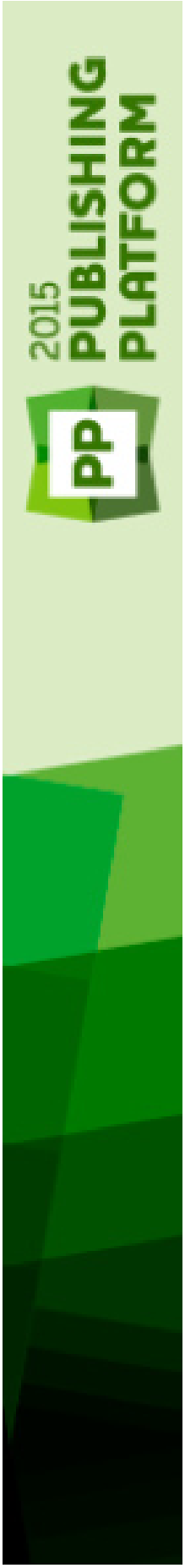

Quark Publishing Platform 2015 August 2019 Update

2019/09/04

Contents

REST framework 1

Session management with REST interface 2

HTTP GET and POST Requests 2

Request/Response parameter formats 3

REST Framework Services 26

Session management 27

logOn 27

getSession 28

getAllSessions 29

keepSessionAlive 29

terminateSession 30

getMaxIdleTime 31

validatePassword 31

logOff 32

Asset management 33

getAsset 34

getAssetAllVersions 42

importAsset 43

importNewAsset 44

updateAsset 44

duplicateAsset 46

deleteAsset 47

createAssetRelation 48

createAssetRelations 51

getAssetRelations 54

updateAssetRelation 55

deleteAssetRelations 57

deleteAssetRelationsForParent 57

generateTextIndexesOfAllAssets 58

generateTextIndexes 58

getAssetVersionNumbers 59

getAssetRenditionsInfo 59

getAvailableRenditions 61
updateChildVersionForAsset 61
updateChildAssetVersionForRelationId 62
updateAllChildAssetsVersion 63
updateChildVersionLockingFlag 63
checkIn 64
checkInNew 66
checkInArticle 68
checkInNewArticle 70
checkInRendition 72
checkOut 73
abortCheckout 74
saveDraft 75
promoteDraft 75
detectContentType 76
reIndex 77

Collection management 79
createCollection 80
createCollections 82
getCollection 84
getCollectionHierarchy 100
getCollectionDefaultJobJacket 102
getCollectionDefaultTicket 102
moveCollection 103
duplicateCollection 104
updateCollection 105
updateCollections 107
deleteCollection 109
deleteCollectionWithAssets 109
createJobjackets 110
getJobjacket 110
getAllJobJackets 111
getJobJacketContent 112
updateJobJacket 113
updateJobJackets 113
parseJobJacketXml 114
deleteJobJacket 115
getApplicableRoutingUsers 115
getApplicablePrivileges 116
getApplicableRoutingGroups 119
getApplicableAttributeConstraints 120
getApplicableStatusTransition 121
getApplicableWorkflowInitialStatuses 122

Query management 123
search 123
searchAssets 125
searchCollections 135
createSearches 138
getSearch 139
getAllSearches 140
getSearchResultsForQuery 140
getTextSearchResults 144
setSharedSearchTrustees 146
getSharedSearchTrustees 147
updateSearches 148
deleteSearch 149
count 149
countAssets 150
countCollections 151
countSearchResultsForQuery 153
countTextSearchResults 155

Administration 157
createUsers 160
getUser 162
getAllUsers 162
getUsersWithGivenUserClass 163
getAllUserRedliningColor 164
updateUserRedliningColor 165
updateUserEnabled 166
updateUsers 166
deleteUser 168
createGroups 169
getGroup 171
getAllGroups 172
getGroupsWithGivenUserClass 172
getUserGroups 173
updateGroups 174
deleteGroup 176
createContentTypes 177
getContentType 180
getAllContentTypes 181
updateContentTypes 181
deleteContentType 184
createAttributes 184
setAttributes 187
getAttribute 187

getAllAttributes 188
updateAttributes 189
deleteAttribute 192
createForms 192
getAllForms 194
updateForms 196
deleteForm 198
createRoles 199
getRole 200
getAllRoles 201
updateRoles 201
deleteRole 204
createRelationType 204
getRelationType 206
getAllRelationTypes 207
updateRelationTypes 207
deleteRelationType 209
createDomains 209
getDomain 211
getAllDomains 212
updateDomains 213
deleteDomain 215
createLdapProfiles 215
getLdapProfile 219
getAllLdapProfiles 221
updateLdapProfiles 222
deleteLdapProfile 223
getLdapUsers 223
importUsersFromLdap 224
getLdapGroups 226
importGroupsFromLdap 227
synchronizeLdapUsersGroups 229
synchronizeLdapUsersGroupsBasedOnProfile 229
createLdapSearches 230
getLdapSearch 230
getAllLdapSearches 231
updateLdapSearches 232
deleteLdapSearch 233
getTrustee 233
updateTrusteeRule 234
createWorkflows 234
getWorkflow 236
getAllWorkflows 239
updateWorkflows 240

deleteWorkFlow....241
createPreferences....241
getPreference....242
getAllPreferences....243
getPreferenceValues....244
updatePreferences....244
updatePreferenceValues....245
deletePreference....246
createRepositories....246
getRepository....247
getAllRepositories....248
updateRepositories....249
deleteRepository....250
createPrivileges....250
getPrivilege....251
getAllPrivileges....252
getApplicableApplicationPriveleges....252
getApplicableContentPrivileges....253
updatePrivileges....255
deletePrivilege....255
createPrivilegeGroups....256
getPrivilegeGroup....257
getAllPrivilegeGroups....257
updatePrivilegeGroups....258
deletePrivilegeGroup....259
addDomainValue....259
getDomainValue....260
getDomainValueXml....260
setDomainValueXml....261
searchDomainValues....262
updateDomainValue....262
deleteDomainValue....263
getFavorites....263
setFavorites....264
getWorkflowInitiatingStatuses....264
getStatusTransitions....265

Publishing....267
getPublishingChannels....267
getAllPublishingChannels....268
getDeliveryChannels....268
publishAsset....269
publishBasedOnURI....271
publishBasedOnStreamContext....272
createDatadoc....273

getDatadocChannels 274
fetchDatadocAssets 276
executeDatadocChannel 277
updateDatadoc 277
detachDatadocChannel 279

Script execution 281
createScript 281
getScriptContent 282
getAllScripts 282
executeScript 283
deleteScript 284

Archive and Restore management 286
archiveCollectionAssets 286
archiveQueryAssets 287
archiveAssets 288
restoreAssets 289

Office 291
getExcelChart 292
getExcelChartBasedOnStream 293
getExcelChartBasedOnURI 294
getExcelTable 294
getExcelTableBasedOnStream 296
getExcelTableBasedOnURI 296
getExcelWorksheets 297
getExcelWorksheetsBasedOnURI 298
getExcelWorksheetsBasedOnStreamContext 299
getChart 300
getChartBasedOnURI 301
getChartBasedOnStream 302
getTable 303
getTableBasedOnURI 304
getTableBasedOnStreamContext 305
getNamedRange 306
getNamedRangeBasedOnURI 308
getNamedRangeBasedOnStream 309
getDynamicRange 310
getDynamicRangeBasedOnURI 311
getDynamicRangeBasedOnStream 312
getVisioPagesInfo 313
getVisioPagesInfoBasedOnURI 314
getVisioPagesInfoBasedOnStreamContext 315
getVisioPageOutput 315

getVisioPageOutputBasedOnURI 316
getVisioPageOutputBasedOnStreamContext....316
getPowerPointSlidesInfo 317
getPowerPointSlidesInfoBasedOnURI 318
getPowerPointSlidesInfoBasedOnStreamContext....319
getPowerPointSlideOutput....319
getPowerPointSlideOutputBasedOnURI 320
getPowerPointSlideOutputBasedOnStreamContext 321
getPowerPointSlideRangeOutput....321
getPowerPointSlideRangeOutputBasedOnURI 322
getPowerPointSlideRangeOutputBasedOnStreamContext 323

Charting 324
getChart....324
getChartBasedOnUri....324
getChartBasedOnStream 325

Configuration 327
getAllXmlMimeTypes 328
getXmlMimeType 329
createXmlMimeType 329
getMappedMimeTypes 330
mapMimeTypes 331
unmapMimeTypes 331
deleteXmlMimeType 332
getPublishingProcess 332
createPublishingProcess 333
deletePublishingProcess 333
getAllPublishingChannelDefinitions....334
getPublishingChannelDefinition....334
getPublishingChannelDefinitions 335
createPublishingChannels 336
createContentChannelMappings....337
deleteContentChannelMappings 338
createIndexingChannelMapping 338
getMappedIndexingChannel....339
deleteIndexingChannelMapping 339
deletePublishingChannels 340
getDeploymentProperties 340
getDeploymentVersion 341
reInitialize 341

Clipboard....343
addClipboardData....343
addFormattedData 344

getAllClipboardChannelsMapping 345
getClipboardData 346
getAllClipboardDataInfos 346
getAllClipboardFormats.....347
getClipboardDataInfo 348
getClipboardChannelsMapping 348
getFormattedData 349
getLatestClipboardData 350
getLatestClipboardDataInfo 350
getLatestFormattedData 351
promoteClipboardDataToAsset 352

JSON support in request parameters for POST requests. 354
createCollection 354
checkinNew.....356
createDomains 357
createSearches 358

Change Log 360
Changes in version 14.0 360
Changes in version 13.2 360
Changes in version 13.1 361
Changes in version 13.0 361
Changes in version 12.0 362
Changes in version 11.2 364
Changes in version 11.0 364

Contacting Quark 367
In the Americas 367
Outside the Americas 367

Legal notices 368

REST framework

The Quark Publishing Platform REST interface lets you access the Platform's functionality using simple HTTP requests. Rather than having to depend on complex remote access libraries such as RMI and SOAP, you can directly invoke functionality and access resources via HTTP and take advantage of standard HTTP mechanisms for transport, session management, security, and error handling. In response to REST calls, the Platform returns any requested data as XML, HTML, or a binary resource stream. If an error occurs, the Platform returns an XML response with an error message and a complete stack trace.

REST URL pattern used in Quark Publishing Platform is:

```
http://server:port/rest/service/typeofcall/resource?op=[value]&[p
arameter 1]=[value1]
```

For secure access to these requests, the platform server should be configured with SSL. Please refer to the "Enabling SSL for Quark Publishing Platform Web applications" chapter in the **System Admin Guide** in order to learn how to configure the Platform server with SSL.

- Request parameters, including *op* are not required for all calls.
- The REST framework is extensible, so you can expose custom services over HTTP.

REST requests return XML responses (by default) or a binary file. The schema of the XML response is specific to the request. These schemas with examples are available in the "*Request parameter formats*" chapter.

For all requests that return XML you can add an additional paramater (i.e. **view=jsonView**) to return **json** instead of an XML response.

You can also apply formatting to returned XML by passing a `view` parameter that points to an XSL transformation or a .jsp page that generates HTML from the XML. The following example points to a .jsp page. This example can be used with a request that returns a search result:

```
http://[server]:[port]/[type of call]?view=thumbnail
```

This example can be used with a request that returns a search result.

Sample .jsp files are provided in the Quark Publishing Platform application folder at `QPP Server/webapps/rest/jsp`

- Users can add their own .jsp file to this folder.

To use custom XSLT, you need to write a custom bean and specify the path of the XSLT file. The XSLT file can be anywhere. The classpath/file location must be mentioned in the bean definition.

Already defined beans allowing you to view responses using XSLT can be found here:

`http://ServerName:ServerPort/rest/service/sessions?view=sessionXsltView`

Support for Transit Session by providing QPP sessionId or username, password as request parameters. User can make a request to server without logging-in by adding request parameters loginname, loginpassword in their requests. Value for loginname and loginpassword would be the username and password of the user. Another request parameter qppsessionid can be used by providing valid qpp session id.

Session management with REST interface

The following types of sessions can be managed using the REST interface:

- Transient sessions - these sessions are not actively managed they are created just for the time period it takes to work on a request.

  Users can receive support for a transient session by providing the Quark Publishing Platform's **sessionId**, **username** or **password** as request parameters. Users can make a request to the server without logging-in by adding **loginname** and **loginpassword** as request parameters.

  Values for **loginname** and **loginpassword** would be the **username** and **password** of the user.

  The request parameter, **qppsessionid** can be used by providing a valid Platform session id.

- Stateful sessions

  Create a stateful session by using the Session Management service. A stateful session created with this approach will last for the maximum idle time. The session can be kept alive by using the **keepsessionalive** request. When working with the session, a user does not need to pass the **qppsessionid** parameter to all requests for the session just created.

HTTP GET and POST Requests

You access the API and manipulate resources by using standard REST conventions. Each API method is invoked using one of the following HTTP methods:

- `GET` - performs a read operation

  `GET` is for retrieving data. When you use the `GET` method, the browser encodes form data into a URL. Use the `GET` method only when the form processing is important.

- `POST` - performs a create operation.

  `POST` can involve storing or updating data. When you use the `POST` method, form data is passed within the message body.

Request/Response parameter formats

Some response and request parameters require an XML body. The format for such parameters is described in the following table.

The same request parameters can also be sent in JSON format.

In order to pass request parameters in JSON format, either a request parameter or a request header with the name `qpp-content-type` can be passed in the request.

Following is the fallback sequence used for resolving the `qpp-content-type`:

1. Request Header `qpp-content-type`.
2. Request Parameter `qpp-content-type`.
3. Auto detection by Tika.

If none of the above resolves to the correct content type, the default value used is XML.

➡ Response in JSON format already exists by specifying extra request parameter `view=jsonView` in the REST request.

Object Name	Schema Definition	Example

`assetRelationInfo`	`<xs:element name="assetRelationInfo">`
	`<xs:complexType>`
	`<xs:sequence>`
	`<xs:element ref="attributeValueList"/>`
	`</xs:sequence>`
	`<xs:attribute name="id" type="xs:long" use="optional"/>`
	`<xs:attribute name="parentAssetId" type="xs:long" use="optional" default="0"/>`
	`<xs:attribute name="parentAssetName" type="xs:string" use="optional"/>`
	`<xs:attribute name="childAssetId" type="xs:long" use="optional"/>`
		XML based relations: XML Component Reference
		`<assetRelationInfoList>`
		`<assetRelationInfo relationType="XML Component Reference" childAssetId="227" parentAssetId="225">`
		`<attributeValueList>`
		`<attributeValue id="417"> xpathValue </attributeValue>`
		`</attributeValueList>`
		`</assetRelationInfo>`
		`</assetRelationInfoList>`
		Primary Attachment:
		`<attributeValueList>`
		`<attributeValue id="401"> 1 </attributeValue>`
		`<attributeValue id="402"> 5 </attributeValue>`

Object Name	Schema Definition	Example
	<xs:attribute name="childAssetName" type="xs:string" use="optional"/> <xs:attribute name="childAssetMajorVersion" type="xs:long" use="optional" default="0"/> <xs:attribute name="childAssetMinorVersion" type="xs:long" use="optional" default="0"/> <xs:attribute name="relationType" type="xs:string" use="optional" default=""/> <xs:attribute name="relationTypeId" type="xs:long" use="optional" default="0"/> <xs:attribute name="relationState" type="xs:string" use="optional"/> <xs:attribute name="relationStateId" type="xs:int" use="optional"/> <xs:attribute name="lockedToChildVersion" type="xs:boolean" use="optional" default="false"/> </xs:complexType> </xs:element>	<attributeValue id="403"> 1 </attributeValue> <attributeValue id="404"> 1 </attributeValue> <attributeValue id="412"> 1 </attributeValue> </attributeValueList>
assetRelationInfoList	<xs:element name="assetRelationInfoList"> <xs:complexType> <xs:sequence> <xs:element maxOccurs="unbounded" ref="assetRelationInfo"/> </xs:sequence> </xs:complexType> </xs:element>	**XML based relations: XML Component Reference** <assetRelationInfoList> <assetRelationInfo relationType="XML Component Reference" childAssetId="227" parentAssetId="225">> <attributeValueList> <attributeValue id="417"> xpathValue </attributeValue> </attributeValueList> </assetRelationInfo> </assetRelationInfoList> **Primary Attachment:** <attributeValueList> <attributeValue id="401"> 1 </attributeValue> <attributeValue id="402"> 5 </attributeValue> <attributeValue id="403"> 1 </attributeValue>

Object Name	Schema Definition	Example
		<attributeValue id="404"> 1 </attributeValue> <attributeValue id="412"> 1 </attributeValue> </attributeValueList>
attributeValue	<xs:element name="attribuuteValue">\ <xs:complexType> <xs:simpleContent> <xs:extension base="xs:string">\ <xs:attribute name="id" type="xs:long" use="optional" default="0"/> <xs:attribute name="name" type="xs:string" use="optional" default=""/> <xs:attribute name="type" type="xs:string" use="optional" default=""/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> <xs:element name="attributeValueList">\ <xs:complexType> <xs:sequence> <xs:element ref="attributeValue" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element>	<attributeValue id="2">\ Hydra.jpg </attributeValue> <attributeValue name="workflow">\ publish </attributeValue> <attributeValue id="55"> 1 </attributeValue>
attributeValueList	<xs:element name="attribuuteValueList">\ <xs:complexType> <xs:sequence> <xs:element ref="attributeValue" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element>	<attributeValueList> <attributeValue valueId="0" type="numeric" name="Id" id="1">\ 222 </attributeValue> <attributeValue valueId="0" type="text" name="Name" id="2">\ 1511623dd8c_Duplicate.qxp </attributeValue> <attributeValue valueId="0" type="dateTime" name="Created" id="3"> 2013-10-16T06:19:38Z \ </attributeValue> <attributeValue valueId="0" type="dateTime" name="Last modified" id="4"> 2013-10-16T06:19:38Z </attributeValue> <attributeValue valueId="1" type="popup" name="Creator" id="5"> Admin </attributeValue> ..... </attributeValueList>
domainValue	<xs:element name="domainValue">\ <xs:complexType> <xs:sequence>	<domainValue id="1">\ <name>Desktop</name> <valueHierarchy> Desktop \ </valueHierarchy> <domainValueList>

Object Name	Schema Definition	Example
	<xs:element name="name" type="xs:string"/> <xs:element name="valueHierarchy" type="xs:string"/> <xs:element ref="domainValueList"/> <xs:element name="name" type="xs:string"/> <xs:element name="valueHierarchy" type="xs:string"/> </xs:sequence> </xs:complexType> </xs:element>	<domainValue id="2"> <name>Research</name> <valueHierarchy> Desktop;Research </valueHierarchy> </domainValue> <domainValue id="3"> <name>Development</name> <valueHierarchy> Desktop;Development </valueHierarchy> </domainValue> </domainValueList> </domainValue>
groupInfo	<xs:element name="groupInfo"><xs:complexType> <xs:sequence> <xs:element name="name" type="xs:string"/> <xs:element name="emailAddress" type="xs:string"/> <xs:element name="nativeTrustee" type="xs:boolean"/> <xs:element name="defaultUserClassId" type="xs:int"/> <xs:element name="defaultUserClass" type="xs:string"/> <xs:element name="ldapProfileId" type="xs:int"/> <xs:element name="ldapProfile" type="xs:string"/> <xs:element name="ldapDistinguished-Name" type="xs:string"/> <xs:element ref="memberList"><xs:element name="nameId" </xs:memberList> </xs:sequence> </xs:complexType> </xs:element>	<groupInfo id="61"><name>Editors</name> <emailAddress> Editors@quark.com </emailAddress> <nativeTrustee> false </nativeTrustee> <defaultUserClassId> 2 </defaultUserClassId> <defaultUserClass> NoPrivileges </defaultUserClass> <ldapProfileId> 10 </ldapProfileId> <ldapProfile> Profile </ldapProfile> <ldapDistinguishedName> DN </ldapDistinguishedName> <memberList> <nameId id="62"><name>John</name> </nameId> </memberList> </groupInfo>
groupInfoList	<xs:element name="groupInfoList"><xs:complexType> <xs:sequence> <xs:element ref="groupInfo" maxOccurs="unbounded"/>	

Object Name	Schema Definition	Example

	`</xs:sequence>`
	`</xs:complexType>`
	`</xs:element>`	

memberList	`<xs:element name="memberList">`
	`<xs:complexType>`
	`<xs:sequence>`
	`<xs:element ref="nameId" maxOccurs="unbounded"/>`
	`</xs:sequence>`
	`</xs:complexType>`
	`</xs:element>`	

ldapProfileInfo	`<xs:element name="lapProfileInfo">`
	`<xs:complexType>`
	`<xs:sequence>`
	`<xs:element name="name" type="xs:string"/>`
	`<xs:element name="realm" type="xs:string"/>`
	`<xs:element name="userName" type="xs:boolean"/>`
	`<xs:element name="authenticationType" type="xs:int"/>`
	`<xs:element ref="ldapServerInfoList">`
	`<xs:element ref="ldapServerInfo">`
	
	`<xs:element name="scheme" type="xs:string"/>`
	
	`<xs:element name="serverName" type="xs:string"/>`
	
	`<xs:element name="port" type="xs:int"/>`
	
	`</xs:ldapServerInfo>`
	
	`</xs:ldapServerInfoList>`
	`<xs:element ref= "ldapObjectsIdentificationCriteria">`
	`<xs:element ref= "userIdentificationCriteria"/>`
	`<xs:element ref= "groupIdentificationCriteria"/>`
	`<xs:element name="groupMemberAttrName" type="xs:string"/>`
	`<xs:element name="membershipI`
		`<ldapProfileInfo id="10">`
		`<name>Quark</name>`
		`<realm> Editors@quark.com </realm>`
		`<userName> realm.com </userName>`
		`<authenticationType> 2 </authenticationType>`
		`<ldapServerInfoList>`
		`<ldapServerInfo>`
		
		`<scheme>LDAP</scheme>`
		`<serverName> SERVER1 </serverName>`
		`<port>389</port>`
		`</ldapServerInfo>`
		`</ldapServerInfoList>`
		`<ldapObjectsIdentificationCriteria>`
		`<userIdentificationCriteria> ... </userIdentificationCriteria>`
		`<groupIdentificationCriteria> ... </groupIdentificationCriteria>`
		`<groupMemberAttrName> Member </groupMemberAttrName>`
		`<membershipIdentifyingAttrName> memberof </membershipIdentifyingAttrName>`
		`<baseDn> BDN </baseDn>`
		`<filterToGetAllUsersOfGroup> ... </filterToGetAllUsersOfGroup>`
		`<filterToGetAllSubGroupsOfGroup> ... </filterToGetAllSubGroupsOfGroup>`
		`<filterToGetAllGroupsOfUser> ... </filterToGetAllGroupsOfUser>`
		`</ldapObjectsIdentificationCriteria>`
		`<ldapAttributeInfoList>`
		`<ldapAttributeInfo id="5">`
		
		`<name>USER_EMAIL</name>`
		`<value>mail</value>`
		`</ldapAttributeInfo>`

Object Name	Schema Definition	Example
	dentifyingAttrName" type="xs:string"/> <xs:element name="baseDn" type="xs:string"/> <xs:element ref="filterToGetAllUsersOfGroup"/> <xs:element ref="filterToGetAllSubGroupsOfGroup"/> <xs:element ref="filterToGetAllGroupsOfUser"/> </xs:ldapObjectsIdentificationCriteria> <xs:element ref="ldapAttributeInfoList"> <xs:element ref="ldapAttrubuteInfo"> <xs:element name="name" type="xs:string"/> <xs:element name="value" type="xs:string"/> </xs:ldapAttributeInfo> </xs:ldapAttributeInfoList> </xs:sequence> </xs:complexType> </xs:element>	</ldapAttributeInfoList> </ldapProfileInfo>
ldapSearchCriteria	<xs:element name="ldapSearchCriteria"> <xs:complexType> <xs:sequence> <xs:element name="ldapProfile" <xs:complexType> <xs:simpleContent> <xs:extension base="xs:string">> <xs:attribute name="id" type="xs:long"/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> <xs:element name="searchilter"	<ldapSearchCriteria> <ldapProfile id="10"> QPP LDAP Profile </ldapProfile> <searchFilter> cn=qpp server </searchFilter> <baseString> dc=india, dc=quark, dc=com </baseString> <searchType id="2"> GROUPS_SEARCH </searchType> </ldapSearchCriteria>

Object Name	Schema Definition	Example
	type="xs:string"/> <xs:element name="baseString" type="xs:string"/> <xs:element name="searchType"> <xs:complexType> <xs:simpleContent> <xs:extension base="xs:string"> <xs:attribute name="id" type="xs:int"/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType> </xs:element>	
ldapSearchInfo	<xs:element name="ldapSearchInfo">\ <xs:complexType> <xs:sequence> <xs:element name="name" type="xs:string"/> <xs:element ref="ldapSearchCriteria"/> </xs:sequence> <xs:attribute name="id" type="xs:long"/> </xs:complexType> </xs:element>	<ldapSearchInfo id="11">\ <name>LDAP Group Search</name> <ldapSearchCriteria> <ldapProfile id="10">QPP LDAP Profile </ldapProfile> <searchFilter> cn=qpp server </searchFilter> <baseString> dc=india, dc=quark, dc=com </baseString> <searchType id="2">GROUPS_SEARCH </searchType> </ldapSearchCriteria> </ldapSearchInfo>
ldapSearchInfoList	<xs:element name="ldapSearchInfoList">\ <xs:complexType> <xs:sequence> <xs:element ref="ldapSearchInfo" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element>	
preferenceInfo	<xs:element name="preferenceInfo">\ <xs:complexType> <xs:sequence> <xs:element name="name" type="xs:string"/>	<preferenceInfo id="1007">\ <name> Display all Redline Deletions in One Color </name> <category id="0"> SYSTEM </category> <possibleValuesExist> false </possibleValuesExist>

Object Name	Schema Definition	Example
	<xs:element name="category">\<xs:complexType\> <xs:simpleContent> <xs:extension base="xs:string"> <xs:attribute name="id" type="xs:int"/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> <xs:element name="possibleValuesExist" type="xs:boolean"/> <xs:element name="defaultValue">\<xs:complexType\> <xs:simpleContent> <xs:extension base="xs:string"> <xs:attribute name="id" type="xs:long"/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> <xs:element ref="preferenceValueList"/> <xs:element name="systemDefined" type="xs:boolean"/> </xs:sequence> <xs:attribute name="id" type="xs:long"/> </xs:complexType> </xs:element>	<defaultValue> No </defaultValue> <systemDefined> true </systemDefined> </preferenceInfo>
preferenceInfoList	<xs:element name="preferenceInfoList">\<xs:complexType\> <xs:sequence> <xs:element ref="preferenceInfo" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType>	<preferenceInfoList> <preferenceInfo id="1007">\<name\> Display all Redline Deletions in One Color \</name\> <category id="0"> SYSTEM \</category\> <possibleValuesExist> false \</possibleValuesExist\>

Object Name	Schema Definition	Example
	</xs:element>	<defaultValue> No </defaultValue> <systemDefined> true </systemDefined> </preferenceInfo> </preferenceInfoList>
privilegeDefinition	<xs:element name="privilegeDefinition">\ <xs:complexType> <xs:sequence> <xs:element name="group">\ <xs:complexType> <xs:simpleContent> <xs:extension base="xs:string">\ <xs:attribute name="id" type="xs:long"/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> <xs:element name="statusBased" type="xs:boolean"/> <xs:element name="systemDefined" type="xs:boolean"/> <xs:element name="contentPrivilege" type="xs:boolean"/> <xs:element ref="contentTypeInfoList"/> </xs:sequence> <xs:attribute name="id" type="xs:long"/> <xs:attribute name="name" type="xs:string"/> </xs:complexType> </xs:element>	<privilegeDefinition name="Check In New" id="1000001">\ <group id="100000"> General </group> <statusBased> false </statusBased> <systemDefined> true </systemDefined> <contentPrivilege> true </contentPrivilege> <contentTypeInfoList> <contentTypeInfo id="1100">\ <name>Asset</name> </contentTypeInfo> </contentTypeInfoList> </privilegeDefinition>
privilegeDefinitionList	<xs:element name="privilegeDefinitionList">\ <xs:complexType> <xs:sequence> <xs:element ref="privilegeDefinition" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element>	<privilegeDefinitionList> <privilegeDefinition name="Check In New" id="1000001">\ <group id="100000"> General </group> <statusBased> false </statusBased> <systemDefined> true </systemDefined> <contentPrivilege> true </contentPrivilege>

Object Name	Schema Definition	Example
		<contentTypeInfoList> <contentTypeInfo id="1100"> <name>Asset</name> </contentTypeInfo> </contentTypeInfoList> </privilegeDefinition> </privilegeDefinitionList>
privilegeGroupDefinition	<xs:element name="privilegeGroupDefinition"> <xs:complexType> <xs:sequence> <xs:element ref="privilegeDefinitionList"/> <xs:element ref="privilegeGroupDefinitionList"/> <xs:element name="systemDefined" type="xs:boolean"/> <xs:element name="parentGroup"> <xs:complexType> <xs:simpleContent> <xs:extension base="xs:string">> <xs:attribute name="id" type="xs:long"/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> </xs:sequence> <xs:attribute name="id" type="xs:long">> </xs:attribute> <xs:attribute name="name" type="xs:string">> </xs:attribute> </xs:complexType> </xs:element>	<privilegeGroupDefinition name="Attributes" id="300000">> <privilegeDefinitionList> <privilegeDefinition name="View" id="3000007">> <group id="300000">> Attributes </group> <statusBased> false </statusBased> <systemDefined> true </systemDefined> <contentPrivilege> false </contentPrivilege> </privilegeDefinition> <privilegeDefinition name="Create/Modify" id="3000008">> <group id="300000">> Attributes </group> <statusBased> false </statusBased> <systemDefined> true </systemDefined> <contentPrivilege> false </contentPrivilege> </privilegeDefinition> <privilegeDefinition name="Delete" id="3000009">> <group id="300000">> Attributes </group> <statusBased> false </statusBased> <systemDefined> true </systemDefined> <contentPrivilege> false </contentPrivilege> </privilegeDefinition> </privilegeDefinitionList> <privilegeGroupDefinitionList> <systemDefined> true </systemDefined> <parentGroup id="99998">> Administration </parentGroup> </privilegeDefinitionList> </privilegeGroupDefinition>
privilegeGroupDefinitionList	<xs:element name="privilegeGroupDefinitionList">> <xs:complexType> <xs:sequence> <xs:element ref="privilegeGroupDefinition" maxOccurs="un	<privilegeGroupDefinitionList> <privilegeGroupDefinition name="Administration" id="99998">> <privilegeDefinitionList/>

Object Name	Schema Definition	Example
	bounded"/> </xs:sequence> </xs:complexType> </xs:element>	<privilegeGroupDefinitionList> <privilegeGroupDefinition name="Attributes" id="300000"> <privilegeDefinitionList> <privilegeDefinition name="View" id="3000007"> <group id="300000"> Attributes </group> <statusBased> false </statusBased> <systemDefined> true </systemDefined> <contentPrivilege> false </contentPrivilege> </privilegeDefinition> <privilegeDefinition name="Create/Modify" id="3000008"> <group id="300000"> Attributes </group> <statusBased> false </statusBased> <systemDefined> true </systemDefined> <contentPrivilege> false </contentPrivilege> </privilegeDefinition> <privilegeDefinition name="Delete" id="3000009"> <group id="300000"> Attributes </group> <statusBased> false </statusBased> <systemDefined> true </systemDefined> <contentPrivilege> false </contentPrivilege> </privilegeDefinition> </privilegeDefinitionList> <privilegeGroupDefinition> <systemDefined> true </systemDefined> <parentGroup id="99998"> Administration </parentGroup> </privilegeGroupDefinition>

Object Name	Schema Definition	Example
		</privilegeGroupDefinitionList>
revisionControlInfo	<xs:element name="revisionControlInfo">\<xs:complexType\>\<xs:sequence\>\<xs:element name="minorVersioningAllowed" type="xs:boolean"/\>\<xs:element name="allRevisions" type="xs:boolean"/\>\<xs:element name="initialRevisionsCount" type="xs:int"/\>\<xs:element name="recentRevisionsByPeriod" type="xs:boolean"/\>\<xs:element name="recentRevisionsCount" type="xs:int"/\>\<xs:element name="recentRevisionsDuration" type="xs:int"/\>\</xs:sequence\>\<xs:attribute name="contentTypeId" type="xs:long"/\>\<xs:attribute name="contentType" type="xs:string"/\>\</xs:complexType\>\</xs:element\>	<revisionControlInfo contentTypeId="110211">\<minorVersioningAllowed\> true \</minorVersioningAllowed\>\<allRevisions\> false \</allRevisions\>\<initialRevisionsCount\> 0 \</initialRevisionsCount\>\<recentRevisionsByPeriod\> false \</recentRevisionsByPeriod\>\<recentRevisionsCount\> 0 \</recentRevisionsCount\>\<recentRevisionsDuration\> 0 \</recentRevisionsDuration\>\</revisionControlInfo\>
revisionControlInfoList	<xs:element name="revisionControlInfoList">\<xs:complexType\>\<xs:sequence\>\<xs:element ref="revisionControlInfo" maxOccurs="unbounded"/\>\</xs:sequence\>\</xs:complexType\>\</xs:element\>	<revisionControlInfoList>\<revisionControlInfo contentTypeId="110211">\<minorVersioningAllowed\> true \</minorVersioningAllowed\>\<allRevisions\> false \</allRevisions\>\<initialRevisionsCount\> 0 \</initialRevisionsCount\>\<recentRevisionsByPeriod\> false \</recentRevisionsByPeriod\>\<recentRevisionsCount\> 0 \</recentRevisionsCount\>\<recentRevisionsDuration\> 0 \</recentRevisionsDuration\>\</revisionControlInfo\>\</revisionControlInfoList\>
redLineInfo	<xs:element name="redlineInfo">\<xs:complexType\>\<xs:sequence\>\<xs:element name="color"\>\<xs:complexType\>\<xs:simpleContent\>\<xs:extension base="xs:string"\>	<redlineInfo>\<color blue="0" green="0" red="0"/\>\<user id="1"\>Admin\</user\>\</redlineInfo\>

Object Name	Schema Definition	Example
	<xs:attribute name="red" type="xs:int"/> <xs:attribute name="green" type="xs:int"/> <xs:attribute name="blue" type="xs:int"/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> <xs:element name="user"> <xs:complexType> <xs:simpleContent> <xs:extension base="xs:string"> <xs:attribute name="id" type="xs:long"/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType> </xs:element>	
redLineInfoList	<xs:element name="redlineInfoList"> <xs:complexType> <xs:sequence> <xs:element ref="redlineInfo" maxOccurs="unbounded" minOccurs="0"/> </xs:sequence> </xs:complexType> </xs:element>	<redlineInfoList> <redlineInfo> <color blue="0" green="0" red="0"/> <user id="1">Admin</user> </redlineInfo> </redlineInfoList>
roleInfo	<xs:element name="roleInfo"> <xs:complexType> <xs:sequence> <xs:element ref="roleInfo" macOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element>	<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <roleInfo name="Administrator" id="1"> <applicationPrivList> <privilege id="190000017"> <name> Remove Link with Collection Template </name> </privilege> <privilege id="190000016"> <name> Status Based Routings </name> </privilege> ..... </applicationPrivList> <contentTypePrivList> <contentTypePriv contentType="Asset" contentTypeId="1100"> <privilege id="10000015"> <name> Get Original Assets </name> </privilege> <privilege id="10000013"> <name> Enable Asset Archiving </name> </privilege>

Object Name	Schema Definition	Example
		<privilege id="1000001"> <name> Check In New </name> </privilege> ..... </content-TypePriv> </contentTypePrivList> </roleInfo>
routingInfo	<xs:element name="routingInfo">> <xs:complexType> <xs:sequence> <xs:element name="trusteeId" type="xs:long"/> <xs:element name="trustee" type="xs:string"/> <xs:element name="routingTypeId" type="xs:int"/> <xs:element name="routingType" type="xs:string"/> </xs:sequence> <xs:attribute name="statusId" type="xs:long"/> <xs:attribute name="status" type="xs:string"/> </xs:complexType> </xs:element>	<routingInfo status="New Status 001" statusId="10">> <trusteeId>1</trusteeId> <routingTypeId> 1 </routing-TypeId> </routingInfo>
routingInfoList	<xs:element name="routingInfoList">> <xs:complexType> <xs:sequence> <xs:element ref="routingInfo" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element>	<routingInfoList> <routingInfo status="New Status 001" statusId="10">> <trusteeId>1</trusteeId> <routingTypeId> 1 </routing-TypeId> </routingInfo> </routingInfoList>
savedSearchList	<xs:element name="savedSearchList">> <xs:complexType> <xs:sequence> <xs:element ref="savedSearch" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element>	<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <savedSearchList> <savedSearch id="100">> <searchName> Default </search-Name> <ownerId>1</ownerId> <ownerName> Admin </ownerName> </savedSearch> </savedSearchList>
scriptInfo	<xs:element name="scriptInfo">> <xs:complexType> <xs:sequence> <xs:element name="name" type="xs:string"/> <xs:element name="language" type="xs:string"/> <xs:element name="content" type="xs:string"/>	<?xml version="1.0" encoding="UTF-8"?> <scriptInfo id="11">> <name>NewScript.js</name> <language> javascript </language> <createdDate> 2015-02-02T09:28:32Z </createdDate> <lastModified> 2015-02-02T09:40:44Z </lastModified>

Object Name	Schema Definition	Example
	<xs:element name="cre- atedDate" type="xs:string"/> <xs:element name="last- Modified" type="xs:string"/> <xs:element name="lastEx- ecutionTime" type="xs:string"/> <xs:element name="lastEx- ecutionLog" type="xs:string"/> <xs:element name="cre- ator"> <xs:complexType> <xs:simpleContent> <xs:extension base="xs:string"> <xs:attribute name="id" type="xs:long"/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> <xs:element name="last- ModifiedBy"> <xs:complexType> <xs:simpleContent> <xs:extension base="xs:string"> <xs:attribute name="id" type="xs:long"/> </xs:extension> </xs:simpleContent> </xs:complexType> </xs:element> <xs:element ref="scrip- tExecutionInfo" maxOc- curs="1" minOccurs="0"/> </xs:sequence> <xs:attribute name="id" type="xs:long"/> </xs:complexType> </xs:element>	<lastExecutionTime> 2015-02- 03T05:55:09Z </lastExecution- Time> <lastExecutionLog> PASSED </las- tExecutionLog> <creator id="1"> Admin </cre- ator> <scriptExecutionInfo trigger- Type="SCHEDULED"> <scheduleInfo scheduleGroup="SCRIPT_MANAGER_GR OUP" id="11"> <cronExpressionList> <cronExpression> 0/30 * * * * ? </cronExpression> </cronExpressionList> </scheduleInfo> </scriptExecutionInfo> </scriptInfo>
scriptInfoList	<xs:element name="script- InfoList"> <xs:complexType> <xs:sequence> <xs:element ref="script- Info" maxOccurs="un- bounded" minOccurs="0"/> </xs:sequence> </xs:complexType> </xs:element>	<?xml version="1.0" encoding="UTF-8"?> <scriptInfoList> <scriptInfo id="11"> <name>NewScript.js</name> <language> javascript </lan- guage> <createdDate> 2015-02- 02T09:28:32Z </createdDate> <lastModified> 2015-02- 02T09:40:44Z </lastModified> <lastExecutionTime> 2015-02- 03T05:55:09Z </lastExecution- Time> <lastExecutionLog> PASSED </las- tExecutionLog> <creator id="1"> Admin </cre- ator>

Object Name	Schema Definition	Example
		<scriptExecutionInfo trigger-Type="SCHEDULED">\<scheduleInfo scheduleGroup="SCRIPT_MANAGER_GROUP" id="11"\>\<cronExpressionList\>\<cronExpression\> 0/30 * * * * ? \</cronExpression\>\</cronExpressionList\>\</scheduleInfo\>\</scriptExecutionInfo\>\</scriptInfo\>\</scriptInfoList\>
scriptExecution Info	<xs:element name="scrip-tExecutionInfo">\<xs:complexType\>\<xs:sequence\>\<xs:element ref="eventInfo"/\>\<xs:element ref="sched-uleInfo"/\>\</xs:sequence\>\<xs:attribute name="trig-gerType" type="xs:string"/\>\</xs:complexType\>\</xs:element\>	<scriptExecutionInfo trigger-Type="SCHEDULED">\<scheduleInfo scheduleGroup="SCRIPT_MANAGER_GROUP" id="11"\>\<cronExpressionList\>\<cronExpression\> 0/30 * * * * ? \</cronExpression\>\</cronExpressionList\>\</scheduleInfo\>\</scriptExecutionInfo\>
searchInfo	<xs:element name="search-Info">\<xs:complexType\>\<xs:sequence\>\<xs:element name="search-Name" type="xs:string"/\>\<xs:element name="owner"\>\<xs:complexType\>\<xs:simpleContent\>\<xs:extension base="xs:string"\>\<xs:attribute name="id" type="xs:long"/\>\</xs:extension\>\</xs:simpleContent\>\</xs:complexType\>\</xs:element\>\<xs:element ref="queryContext"/\>\<xs:element ref="queryConditionList"/\>\<xs:element ref="query-Display"/\>\<xs:element ref="trustee-InfoList"/\>\</xs:sequence\>\<xs:attribute name="id" type="xs:long"/\>\<xs:attribute name="para-meterized" type="xs:boolean"/\>\</xs:complexType\>\</xs:element\>	<searchInfoList>\<searchInfo parameterized="false" id="100"\>\<searchName\> Default \</search-Name\>\<owner id="1"\>Admin\</owner\>\<queryContext context="ASSET"/\>\<queryDisplay exploreMode="PLAIN" display-Mode="PLAIN"\>\<displayColumnList\>\<displayColumn attributeCol-umn="true" width="0" colum-nId="2"\> Name \</displayColumn\>\<displayColumn attributeCol-umn="true" width="0" colum-nId="55"\> Collection \</displayColumn\>\<displayColumn attributeCol-umn="true" width="0" colum-nId="54"\> Workflow \</displayColumn\>\<displayColumn attributeCol-umn="true" width="0" colum-nId="24"\> Status \</displayColumn\>\</displayColumnList\>\<sortInfoList\>\<sortInfo attributeColumn="true" order="descending" columnId="2"\> Name \</sortInfo\>\</sortInfoList\>\</queryDisplay\>\<trusteeInfoList\>\<trusteeInfo id="11"\>

Object Name	Schema Definition	Example
		<user>true</user> <deleted>false</deleted> <name>DKumar</name> </trusteeInfo> </trusteeInfoList> </searchInfo> </searchInfoList>
searchInfoList	<xs:element name="search-InfoList">\ <xs:complexType> <xs:sequence> <xs:element ref="search-Info" maxOccurs="un-bounded"/> </xs:sequence> </xs:complexType> </xs:element>	
sessionInfo	<xs:element name="ses-sionInfo">\ <xs:complexType> <xs:sequence> <xs:element name="ses-sionId" type="xs:string"/> <xs:element name="userId" type="xs:long"/> <xs:element name="user-Name" type="xs:string"/> <xs:element name="time-Created" type="xs:string"/> <xs:element name="lastAc-cessedTime" type="xs:string"/> <xs:element name="machi-neName" type="xs:string"/> <xs:element name="appli-cationName" type="xs:string"/> <xs:element name="timeZo-neOffset" type="xs:string"/> </xs:sequence> </xs:complexType> </xs:element>	<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <sessionInfo> <sessionId> 8d9676d9-1300-433c-b305-509cb2fa2bf9 </sessionId> <userId>1</userId> <userName>Admin</userName> <timeCreated> 2013-10-16T07:25:52Z </timeCreated> <lastAccessedTime> 2013-10-16T07:25:52Z </lastAccessedTime> <machineName> localhost </machi-neName> <applicationName> Rest Client </applicationName> <timeZoneOffset> +05:30 </time-ZoneOffset> </sessionInfo>
sessionInfoList	<xs:element name="ses-sionInfoList">\ <xs:complexType> <xs:sequence> <xs:element ref="session-Info" maxOccurs="un-bounded"/> </xs:sequence> </xs:complexType> </xs:element>	<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <sessionInfoList> <sessionInfo> <sessionId> c191521e-20c2-495f-91ad-bace1681d466 </sessionId> <userId>1</userId> <userName>Admin</userName> <timeCreated> 2013-10-16T05:47:23Z </timeCreated> <lastAccessedTime> 2013-10-16T12:58:07+05:30 </lastAc-cessedTime>

Object Name	Schema Definition	Example
		<machineName> 0:0:0:0:0:0:0:1 </machineName> <applicationName> Quark Publishing Platform Admin </applicationName> <timeZoneOffset> +05:30 </timeZoneOffset> </sessionInfo> <sessionInfo> <sessionId> 1f8e5d68-e915-47eb-a504-ef51333c24e4 </sessionId> ..... </sessionInfo> <sessionInfo> <sessionId> de4323a6-d4a6-4f15-9dfd-5c8d69f3b7d9 </sessionId> ..... </sessionInfo> </sessionInfoList>
trusteeInfo	<xs:element name="trusteeInfo">> <xs:complexType> <xs:sequence> <xs:element name="user" type="xs:boolean"/> <xs:element name="deleted" type="xs:boolean"/> <xs:element name="name" type="xs:string"/> <xs:element name="emailAddress" type="xs:string"/> <xs:element name="nativeTrustee" type="xs:boolean"/> <xs:element name="defaultUserClassId" type="xs:int"/> <xs:element name="ldapProfileId" type="xs:int"/> <xs:element name="ldapDistinguishedName" type="xs:string"/> </xs:sequence> </xs:complexType> </xs:element>	<xs:element name="trusteeInfo">> <xs:complexType> <xs:sequence> <xs:element name="user" type="xs:boolean"/> <xs:element name="deleted" type="xs:boolean"/> <xs:element name="name" type="xs:string"/> <xs:element name="emailAddress" type="xs:string"/> <xs:element name="nativeTrustee" type="xs:boolean"/> <xs:element name="defaultUserClassId" type="xs:int"/> <xs:element name="ldapProfileId" type="xs:int"/> <xs:element name="ldapDistinguishedName" type="xs:string"/> </xs:sequence> </xs:complexType> </xs:element>
userInfo	<xs:element name="userInfo">> <xs:complexType> <xs:sequence> <xs:element name="name" type="xs:string"/> <xs:element name="domainId" type="xs:int"/> <xs:element name="firstName" type="xs:string"/> <xs:element name="lastName" type="xs:string"/> <xs:element name="phoneNumber" type="xs:string"/>	<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <userInfo id="1">> <name>Admin</name> <domainId>0</domainId> <firstName> Administrator </firstName> <emailAddress> admin@quark.com </emailAddress> <nativeTrustee> false </nativeTrustee> <enabled>true</enabled> <defaultUserClassId> 1 </defaultUserClassId>

Object Name	Schema Definition	Example
	<xs:element name="emailAddress" type="xs:string"/> <xs:element name="nativeTrustee" type="xs:boolean"/> <xs:element name="enabled" type="xs:boolean"/> <xs:element name="defaultUserClassId" type="xs:long"/> <xs:element name="defaultUserClass" type="xs:string"/> <xs:element name="ldapProfileId" type="xs:long"/> <xs:element name="nameForAuthentication" type="xs:string"/> <xs:element name="user" type="xs:boolean"/> <xs:element name="deleted" type="xs:boolean"/> <xs:element name="overriddenUserClassId" type="xs:long"/> <xs:element name="ldapDistinguishedName" type="xs:long"/> </xs:sequence> <xs:attribute name="id" type="xs:long"/> </xs:complexType> </xs:element>	<defaultUserClass> Administrator </defaultUserClass> <ldapProfileId> 0 </ldapProfileId> <user>true</user> <deleted>false</deleted> <ldapDistinguishedName> DN </ldapDistinguishedName> <overriddenUserClassId> 0 </overriddenUserClassId> </userInfo>
userInfoList	<xs:element name="userInfoList"> <xs:complexType> <xs:sequence> <xs:element maxOccurs="unbounded" ref="userInfo"/> </xs:sequence> </xs:complexType> </xs:element>	
workflowInfo	<xs:element name="workflowInfo"> <xs:complexType> <xs:sequence> <xs:element name="name" type="xs:string"/> <xs:element name="domainId" type="xs:int"/> <xs:element name="created" type="xs:string"/> <xs:element name="createdBy" type="xs:long"/> <xs:element name="creatorName" type="xs:string"/>	<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <workflowInfoList> <workflowInfo id="10"> <name>Workflow1</name> <domainId>18</domainId> <created> 2014-12-24T06:12:37Z </created> <createdBy>1</createdBy> <creatorName> Admin </creatorName> <lastModifiedBy> 1 </lastModifiedBy>

Object Name	Schema Definition	Example
	<xs:element name="last-ModifiedBy" type="xs:long"/> <xs:element name="last-ModifierName" type="xs:string"/> <xs:element name="last-Modified" type="xs:string"/> <xs:element name="con-strainedStatusTransition" type="xs:boolean" minOc-curs="0"/> <xs:element name="role-BasedRouting" type="xs:boolean" minOc-curs="0"/> <xs:element ref="status-InfoList"/> <xs:element ref="content-TypeInfoList"/> <xs:element ref="status-TransitionInfoList"/> <xs:element ref="role-BasedStatusTransitionIn-foList"/> <xs:element ref="status-RoutingRolesInfoList"/> </xs:sequence> <xs:attribute name="id" type="xs:long"/> </xs:complexType> </xs:element>	<lastModifierName> Admin </last-ModifierName> <lastModified> 2015-01-22T07:16:16Z </lastModified> <constrainedStatusTransition> true </constrainedStatusTransi-tion> <roleBasedRouting> true </role-BasedRouting> <statusInfoList> <statusInfo id="10"> <name>Status1</name> <domainId>6</domainId> <colorRed>50</colorRed> <colorGreen> 150 </colorGreen> <colorBlue>10</colorBlue> <position>1</position> <redlineTracking> false </red-lineTracking> <checkInLayoutEvaluation> false </checkInLayoutEvaluation> <outputLayoutEvaluation> false </outputLayoutEvaluation> <privilegeInfoList> <privilegeInfo> <roleName> Administrator </role-Name> <roleId>1</roleId> <overridden> false </overridden> <contentTypePrivList/> </privilegeInfo> <privilegeInfo> <roleName> Editor </roleName> <roleId>20</roleId> <overridden> false </overridden> <contentTypePrivList/> </privilegeInfo> <privilegeInfo> <roleName> Publisher </roleName> <roleId>21</roleId> <overridden> false </overridden> <contentTypePrivList/> </privilegeInfo> </privilegeInfoList> </statusInfo> </statusInfoList> <contentTypeInfoList> <contentTypeInfo id="1102"> <name>Project</name> <contentTypeHierarchy> System;Asset;Project; </content-TypeHierarchy> <systemDefined> true </systemDe-fined> <childrenAvailable> true </chil-drenAvailable> <extensible> true </extensible> </contentTypeInfo> </contentTypeInfoList> <statusTransitionInfoList> <statusTransitionInfo id="10"> <name>Status1</name>

Object Name	Schema Definition	Example
		<workflowInitiatingStatus> true </workflowInitiatingStatus> </statusTransitionInfo> </statusTransitionInfoList> <roleBasedStatusTransitionIn-foList> <roleBasedStatusTransitionInfo id="20"> <name>Editor</name> <statusTransitionInfoList> <statusTransitionInfo id="10"> <name>Status1</name> <workflowInitiatingStatus> true </workflowInitiatingStatus> </statusTransitionInfo> </statusTransitionInfoList> </roleBasedStatusTransitionInfo> </roleBasedStatusTransitionIn-foList> <statusRoutingRolesInfoList> <statusRoutingRolesInfo id="10"> <name>Status1</name> <routingRolesInfoList> <routingRoleInfo id="1"> <name> Administrator </name> </routingRoleInfo> <routingRoleInfo id="20"> <name>Editor</name> </routingRoleInfo> </routingRolesInfoList> </statusRoutingRolesInfo> </statusRoutingRolesInfoList> </workflowInfo> </workflowInfoList>
workflowInfoLis t	<xs:element name="work-flowInfoList"> <xs:complexType> <xs:sequence> <xs:element ref="work-flowInfo" maxOccurs="un-bounded"/> </xs:sequence> </xs:complexType> </xs:element>	
visioPageInfo	<xs:element name="vi-sioPageInfo"> <xs:complexType> <xs:sequence> <xs:element name="page-Name" type="xs:string"/> </xs:sequence> <xs:attribute name="index" type="xs:int"/> </xs:complexType> </xs:element>	<visioPageInfo index="1"> <page-Name>Overview</pageName> </vi-sioPageInfo> <visioPageInfo index="2"> <page-Name>Final</pageName> </vi-sioPageInfo>
mimeTypeInfo	<xs:element name="mime-TypeInfo"> <xs:complexType> <xs:sequence>	<mimeTypeInfo type="applica-tion/xml; format=smarttable" parentMimeType="application/xml; ">

Object Name	Schema Definition	Example
	<xs:element ref="pi" minOccurs="0" maxOccurs="1"/> <xs:element name="xpath" type="xs:string" minOccurs="0" maxOccurs="1"/> </xs:sequence> <xs:attribute name="type" type="xs:string" use="required"/> <xs:attribute name="parentMimeType" type="xs:string" use="required"/> </xs:complexType> </xs:element>	<xpath>/*[local-name()='table']/@type='smart-table'</xpath> </mimeTypeInfo>
visioPageInfoList	<xs:element name="visioPageInfoList">\ <xs:complexType> <xs:sequence> <xs:element ref="visioPageInfo" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element>	<visioPageInfoList> <visioPageInfo index="1">\ <pageName>Overview</pageName> </visioPageInfo> <visioPageInfo index="2">\ <pageName>Final</pageName> </visioPageInfo> </visioPageInfoList>
mimeTypeInfoList	<xs:element name="mimeTypeInfoList">\ <xs:complexType> <xs:sequence> <xs:element ref="mimeTypeInfo" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element>	<mimeTypeInfoList> <mimeTypeInfo type="application/xml; format=smarttable" parentMimeType="application/xml; "> <xpath>/*[local-name()='table']/ @type='smart-table'</xpath> </mimeTypeInfo> <mimeTypeInfo type="application/xml; format=auxiliarydictionary" parentMimeType="format=hyphenationexceptions">\ <xpath>/*[local-name()='Dictionary']/ @Type='AuxiliaryDictionary'</xpath> </mimeTypeInfo> </mimeTypeInfoList>
powerPointSlideInfo	<xs:element name="powerPointSlideInfo">\ <xs:complexType> <xs:sequence> <xs:element name="slideId" type="xs:long"/> </xs:sequence> <xs:attribute name="index" type="xs:int"/> </xs:complexType> </xs:element>	<powerPointSlideInfo index="1">\ <slideId>256</slideId> </powerPointSlideInfo> <powerPointSlideInfo index="2">\ <slideId>257</slideId> </powerPointSlideInfo> <powerPointSlideInfo index="3">\ <slideId>316</slideId> </powerPointSlideInfo> <powerPointSlideInfo index="4">\ <slideId>330</slideId> </powerPointSlideInfo> <powerPointSlideInfo index="5">

Object Name	Schema Definition	Example
		<slideId>325</slideId> </powerPointSlideInfo>
powerPointSlideInfoList	<xs:element name="powerPointSlideInfoList">\ <xs:complexType>\ <xs:sequence>\ <xs:element ref="powerPointSlideInfo" maxOccurs="unbounded"/>\ </xs:sequence>\ </xs:complexType>\ </xs:element>	<powerPointSlideInfoList>\ <powerPointSlideInfo index="1">\ <slideId>256</slideId>\ </powerPointSlideInfo>\ <powerPointSlideInfo index="2">\ <slideId>257</slideId>\ </powerPointSlideInfo>\ <powerPointSlideInfo index="3">\ <slideId>316</slideId>\ </powerPointSlideInfo>\ <powerPointSlideInfo index="4">\ <slideId>330</slideId>\ </powerPointSlideInfo>\ <powerPointSlideInfo index="5">\ <slideId>325</slideId>\ </powerPointSlideInfo>\ </powerPointSlideInfoList>

REST Framework Services

The API includes support for the following functional groups:

- *Session Management*
- *Asset Management*
- *Collection Management*
- *Query Management*
- *Administration*
- *Publishing*
- *Script Execution*
- *Archive & Restore Management*
- *Office Document Management*
- *Charts Management*
- *Configuration Management*
- *Clipboard Management*

The input parameters for all functions have been identified in their description as being either mandatory or optional. An input value designated with [*value*], indicates a value that you must provide.

Many functions require an `op` parameter as input. The `op` parameter allows you to specify which operation is to be performed. The value reguired for the `op` parameter for each function is given.

Session management

The session management capabilities of the API allow application developers to manage sessions and includes the following REST calls:

- *logOn*
- *getSession*
- *getAllSessions*
- *keepSessionAlive*
- *terminateSession*
- *getMaxIdleTime*
- *validatePassword*
- *logOff*

logOn

Function	Log in		
Description	This request starts a session.		
Type	POST		
URL Format	`.../sessions/[user name]`		
Java API	**Method Name**		**Class Name**
	logOn		`com.quark.qpp.service.facadeClassSessionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=logon`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`password=[password]`	string	Users password (Mandatory).
	`asusername=[name]`	string	If password is set to the super admin password, you can perform actions as a different user by supplying that user name here. (Optional)
	`machinename=[name]`	string	Lets you pass the name of the computer from which you have logged in. (Optional) If this value is not provided

			then the machinename is set as localhost for this session.
	appname=[name]	string	Lets you pass the name of the application from which you have logged in. (Optional) If this value is not provided then the appname is set as Rest Client for this session.
Returns	Returns a sessionInfo object in XML format. See the table in the "Request/Response parameter formats" chapter.		
Examples	**URL:** http://serverName:serverPort/rest/service/sessions/Admin?op=logon&password=Admin **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <sessionInfo> <sessionId>30609864-6dd2-4eba-9031-c80ef56f3a45</sessionId> <userId>1</userId> <userName>Admin</userName> <timeCreated>2013-04-10T07:32:06Z</timeCreated> <lastAccessedTime>2013-04-10T07:32:06Z</lastAccessedTime> <machineName>localhost</machineName> <applicationName>Rest Client</applicationName> <timeZoneOffset>+05:30</timeZoneOffset> </sessionInfo>		

getSession

Function	Get information about a specific session	
Description	This request returns information about a specific session.	
Type	GET	
URL Format	.../sessions/[session ID]	
Java API	**Method Name**	**Class Name**
	getSession	com.quark.qpp.service.facade Class SessionFacade
Parameters	This request requires no input parameters	
Returns	Returns a sessionInfo object in XML format. See the table in the "Request/Response parameter formats" chapter.	
Examples	**URL:** http://serverName:serverPort/rest/service/sessions/30609864-6dd2-4eba-9031-c80ef56f3a45 **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <sessionInfo> <sessionId>30609864-6dd2-4eba-9031-c80ef56f3a45</sessionId> <userId>1</userId> <userName>Admin</userName> <timeCreated>2013-04-10T07:32:06Z</timeCreated>	

	`<lastAccessedTime>2013-04-10T07:32:06Z</lastAccessedTime>`
	`<machineName>localhost</machineName>`
	`<applicationName>Rest Client</applicationName>`
	`<timeZoneOffset>+05:30</timeZoneOffset>`
	`</sessionInfo>`

getAllSessions

Function	Get all sessions	
Description	This request returns a list of all the active sessions.	
Type	GET	
URL Format	`.../sessions`	
Java API	**Method Name**	**Class Name**
	`getAllSessions`	`com.quark.qpp.service.facade Class SessionFacade`
Parameters	This request requires no input parameters	
Returns	Returns a `sessionInfoList` object containing an array of active sessions in XML format. See the table in the "Request/Response parameter formats" chapter.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/sessions`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<sessionInfoList>`
	`<sessionInfo>`
	`<sessionId>e1a89f4c-aa30-488d-9e0e-e067e3431be4</sessionId>`
	`<userId>1</userId>`
	`<userName>Admin</userName>`
	`<timeCreated>2013-04-10T06:11:56Z</timeCreated>`
	`<lastAccessedTime>2013-04-10T13:02:57+05:30</lastAccessed-Time>`
	`<machineName>0:0:0:0:0:0:0:1</machineName>`
	`<applicationName>Quark Publishing Platform Web Client</ap-plicationName>`
	`<timeZoneOffset>+05:30</timeZoneOffset>`
	`</sessionInfo>`
	`....`
	`</sessionInfoList>`	

keepSessionAlive

Function	Keep a specific session alive	
Description	This request keeps the current session alive. The maximum idle time for a session is 60 seconds, which can be updated in the {qpp_home}/serverApp.properties file.	
Type	GET	
URL Format	`.../sessions/`	
Java API	**Method Name**	**Class Name**
	`keepSessionAlive`	`com.quark.qpp.service.facade`

			Class SessionFacade
Parameters	Name	Type	Description
	op=keepalive	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
Returns	Returns a sessionInfo object in XML format. See the table in the "Request/Response parameter formats" chapter.		
Examples	**URL:** http://serverName:serverPort/rest/service/sessions?op=keepalive **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <sessionInfo> <sessionId>30609864-6dd2-4eba-9031-c80ef56f3a45</sessionId> <userId>1</userId> <userName>Admin</userName> <timeCreated>2013-04-10T07:32:06Z</timeCreated> <lastAccessedTime>2013-04-10T07:32:06Z</lastAccessedTime> <machineName>localhost</machineName> <applicationName>Rest Client</applicationName> <timeZoneOffset>+05:30</timeZoneOffset> </sessionInfo>		

terminateSession

Function	Delete a session		
Description	This request terminates the session with the given sessionID.		
Type	POST		
URL Format	.../sessions/[session ID]		
Java API	Method Name		Class Name
	terminateSession		com.quark.qpp.service.facade Class SessionFacade
Parameters	Name	Type	Description
	op=delete	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** http://serverName:serverPort/rest/service/sessions/fee9ea88-68cc-47b9-a576-dab6ad0181eb?op=delete		

getMaxIdleTime

Function	Get the maximum idle time for all sessions		
Description	This request returns the value for maximum idle time setting for all sessions in seconds. User can update the value for the `session.maxIdle` property by editing the `(qpp_home/conf) serverApp.properties` file.		
Type	GET		
URL Format	`.../sessions`		
Java API	**Method Name**		**Class Name**
	`getMaxIdleTime`		`com.quark.qpp.service.facade Class SessionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=maxidletime`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	Returns the maximum idle time set for sessions (for example, 60 seconds (Default)).		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/sessions?op=maxidletime`
	Result:
	`60`		

validatePassword

Function	Validate a password		
Description	This request validates a password for a user.		
Type	POST		
URL Format	`.../sessions/[user name]`		
Java API	**Method Name**		**Class Name**
	`validatePassword`		`com.quark.qpp.service.facade Class SessionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=validate`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`password=[password]`	string	Password to be validated. (Mandatory)
Returns	Returns a string value of `true` or `false`.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/sessions/Admin?op=validate&password=Admin`
	Result:
	`True/False`		

logOff

Function	Log out		
Description	This request ends the session that you are currently logged into.		
Type	POST		
URL Format	`.../sessions/[session ID]`		
Java API	**Method Name**		**Class Name**
	`logOff`		`com.quark.qpp.service.facade Class SessionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=logoff`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)This operation will run only for a valid session id.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/sessions/fee9ea88-68cc-47b9-a576-dab6ad0181eb?op=logoff`		

Asset management

The asset management capabilities of the API allow application developers to manage assets and relations and includes the following REST calls:

- *getAsset*
- *getAssetAllVersions*
- *importAsset*
- *importNewAsset*
- *updateAsset*
- *duplicateAsset*
- *deleteAsset*
- *createAssetRelation*
- *createAssetRelations*
- *getAssetRelations*
- *updateAssetRelation*
- *deleteAssetRelation*
- *deleteAssetRelationsForParent*
- *generateTextIndexesOfAllAssets*
- *generateTextIndexes*
- *getAssetVersionNumbers*
- *getAssetRenditionsInfo*
- *getAvailableRenditions*
- *updateChildVersionForAsset*
- *updateChildAssetVersionForRelationId*
- *updateAllChildAssetsVersions*
- *updateChildVersionLockingFlag*
- *checkIn*
- *checkInNew*
- *checkInArticle*
- *checkInNewArticle*

- *checkInRendition*
- *checkOut*
- *abortCheckout*
- *saveDraft*
- *promoteDraft*
- *detectContentType*
- *generateTextIndexes*
- *generateTextIndexesOfAllAssets*
- *reIndex*

➡ If you are retrieving information about a custom relationship that could be renamed, or if you are retrieving system relationship names that have been localized, use the relationship's ID number instead of the name. To get the IDs of all relationships, use a call to `/admin/relationtypes`.

getAsset

Function	Get all metadata for an asset.		
Description	This request returns all of the metadata associated with an asset.		
Type	GET		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`getAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`attributes=all`	string	(Mandatory)
Returns	Returns an `assetInfo` object with the metadata of all attributes.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/assets/4?attributes=all`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfo id="4">`
	`<assetVersion minorVersion="1" majorVersion="0"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">4</attributeValue>`
	`<attributeValue type="text" name="Name" id="2">1quark2.jpg</attributeValue>`
	`....`
	`</attributeValueList>`
	`</assetInfo>`		

Function	Get a specific piece of metadata for an asset.
Description	This request returns a specific piece of metadata associated with an asset.

Type	GET		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`getAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`attributes= [comma-separated list of attribute names or IDs]`	string	For a list of attributes, use a call to admin/attributes. (Mandatory)
	`majorversion` and `minorversion`	string	Specify the required major and minor version for which the given rendition of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
	`includechildren`	boolean	Returns the relation information of all child assets for the given asset. (Optional)
	`includeparents`	boolean	Returns the relation information of all parent assets for the given asset. (Optional)
Returns	Returns an `assetInfo` object with the metadata of specified attributes.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/assets/4?attributes=1,name,3,4,5,6,7,8,54,collection`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <assetInfo id="4">`
	`  <assetVersion minorVersion="1" majorVersion="0"/>`
	`  <attributeValueList>`
	`  <attributeValue type="numeric" name="Id" id="1">4</attributeValue>`
	`  ....`
	`  </attributeValueList>`
	`  </assetInfo>`
	...with the `majorversion` and `minorversion` parameters
	URL:
	`http://serverName:serverPort/rest/service/assets/292?attributes=1,name,3,4,5,6,7,8,54,collection&majorversion=1&minorversion=0`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <assetInfo id="292">`
	`  <assetVersion minorVersion="0" majorVersion="1"/>`
	`  <attributeValueList>`		

```
      <attributeValue valueId="0" type="numeric" name="Id"
id="1">292</attributeValue>
      <attributeValue valueId="0" type="text" name="Name"
id="2">15116462.qxp</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Created"
id="3">2013-10-16T11:48:52Z</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Last mod-
ified" id="4">2013-10-16T11:54:56Z</attributeValue>
      <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
      <attributeValue valueId="1" type="popup" name="Last modi-
fier" id="6">Admin</attributeValue>
      <attributeValue valueId="0" type="numeric" name="Major ver-
sion" id="7">1</attributeValue>
      <attributeValue valueId="0" type="text" name="File path"
id="8">/01/24/292.1-0.1.1.qxp</attributeValue>
      <attributeValue valueId="10" type="popup" name="Workflow"
id="54">Workflow1</attributeValue>
      <attributeValue valueId="1" type="popup" name="Collection"
id="55">Home</attributeValue>
    </attributeValueList>
  </assetInfo>
```

...with the `includechildren` parameter

URL:

```
http://serverName:serverPort/rest/service/assets/292?attributes=1
,name,3,4,5,6,7,8,54,collection&includechildren=true
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <assetInfo id="292">
    <assetVersion minorVersion="0" majorVersion="1"/>
    <attributeValueList>
      <attributeValue valueId="0" type="numeric" name="Id"
id="1">292</attributeValue>
      <attributeValue valueId="0" type="text" name="Name"
id="2">15116462.qxp</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Created"
id="3">2013-10-16T11:48:52Z</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Last mod-
ified" id="4">2013-10-16T11:54:56Z</attributeValue>
      <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
      <attributeValue valueId="1" type="popup" name="Last modi-
fier" id="6">Admin</attributeValue>
      <attributeValue valueId="0" type="numeric" name="Major ver-
sion" id="7">1</attributeValue>
      <attributeValue valueId="0" type="text" name="File path"
id="8">/01/24/292.1-0.1.1.qxp</attributeValue>
      <attributeValue valueId="10" type="popup" name="Workflow"
id="54">Workflow1</attributeValue>
      <attributeValue valueId="1" type ="popup" name="Collection"
id="55">Home</attributeValue>
    </attributeValueList>
    <assetRelationInfoList>
      <assetRelationInfo relationStateId="1" relationState="Per-
manent" relationTypeId="1" relationType="Primary Attachment"
childAssetMinorVersion="2" childAssetMajorVersion="0" childAsset-
Name="153486e3.qcd" childAssetId="290"
parentAssetName="15116462.qxp" parentAssetId="292" id="109">
      <attributeValueList>
        <attributeValue valueId="0" type="numeric" name="Attached
Layout Uid" id="401">1</attributeValue>
```

```
        <attributeValue valueId="0" type="numeric" name="Box Uid"
id="402">5</attributeValue>
        <attributeValue valueId="0" type="text" name="Page Name"
id="403">Page 1</attributeValue>
        <attributeValue valueId="0" type="numeric" name="Page
Index" id="404">0</attributeValue>
        <attributeValue valueId="0" type="numeric" name="Attached
Component Id" id="412">136</attributeValue>
      </attributeValueList>
    </assetRelationInfo>
  </assetRelationInfoList>
</assetInfo>
```


...with the `includechildren` and `includeparents` parameters

URL:

```
http://serverName:serverPort/rest/service/assets/290?attributes=1
,name,3,4,5,6,7,8,54,collection&includechildren=true&includeparen
ts=true
```

Result:


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <assetInfo id="290">
    <assetVersion minorVersion="2" majorVersion="0"/>
    <attributeValueList>
      <attributeValue valueId="0" type="numeric" name="Id"
id="1">290</attributeValue>
      <attributeValue valueId="0" type="text" name="Name"
id="2">153486e3.qcd</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Created"
id="3">2013-10-16T11:48:14Z</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Last mod-
ified" id="4">2013-10-16T11:54:57Z</attributeValue>
      <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
      <attributeValue valueId="1" type="popup" name="Last modi-
fier" id="6">Admin</attributeValue>
      <attributeValue valueId="0" type="numeric" name="Major ver-
sion" id="7">0</attributeValue>
      <attributeValue valueId="0" type="text" name="File path"
id="8">/01/22/290.0-2.1.1.qcd</attributeValue>
      <attributeValue valueId="10" type="popup" name="Workflow"
id="54">Workflow1</attributeValue>
      <attributeValue valueId="1" type="popup" name="Collection"
id="55">Home</attributeValue>
    </attributeValueList>
    <assetRelationInfoList>
      <assetRelationInfo relationStateId="1" relationState="Per-
manent" relationTypeId="4" relationType="Article Component Refer-
ence" childAssetMinorVersion="1" childAssetMajorVersion="0"
childAssetName="Picture2850.jpg" childAssetId="291" parentAsset-
Name="153486e3.qcd" parentAssetId="290" id="108">
        <attributeValueList>
          <attributeValue valueId="0" type="numeric" name="At-
tached Component Id" id="412">136</attributeValue>
          <attributeValue valueId="0" type="numeric" name="Arti-
cle Component id" id="305">2</attributeValue>
          <attributeValue valueId="0" type="numeric" name="Arti-
cle Id" id="413">290</attributeValue>
        </attributeValueList>
      </assetRelationInfo>
      <assetRelationInfo relationStateId="1" relationState="Per-
manent" relationTypeId="1" relationType="Primary Attachment"
```

	`childAssetMinorVersion="2" childAssetMajorVersion="0" childAsset-Name="153486e3.qcd" childAssetId="290" parentAssetName="15116462.qxp" parentAssetId="292" id="109">`
	`<attributeValueList>`
	`<attributeValue valueId="0" type="numeric" name="At-tached Layout Uid" id="401">1</attributeValue>`
	`<attributeValue valueId="0" type="numeric" name="Box Uid" id="402">5</attributeValue>`
	`<attributeValue valueId="0" type="text" name="Page Name" id="403">Page 1</attributeValue>`
	`<attributeValue valueId="0" type="numeric" name="Page Index" id="404">0</attributeValue>`
	`<attributeValue valueId="0" type="numeric" name="At-tached Component Id" id="412">136</attributeValue>`
	`</attributeValueList>`
	`</assetRelationInfo>`
	`</assetRelationInfoList>`
	`</assetInfo>`

Function	Get a rendition of an asset.		
Description	This request returns a particular rendition of an asset.		
Type	GET		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`getAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`rendition=highres or rendition=1`		Specify the required rendition type. (Optional) If the paramater is not passed the system returns the highres rendition type of the asset. Available renditions types are: • HIGHRES • PREVIEW • THUMBNAIL • ARTICLE_PAGE_PICTURE • DECONSTRUCTED_XML • TEXT
	`layout=1`		Specify layout ID in case of QuarkXPress document or CopyDesk article. (Optional) Layout only applies to a content type of QuarkPress, QuarkXPress Template , InDesign Document and InDesign Document Template. If this parameter is not passed it is defaulted to 1.
	`page=2`		Specify page in case of QuarkXPress document or CopyDesk article. (Optional) Page applies to all assets that have multi-

			page rendtions. If this parameter is not passed it is defaulted to 1.
	`contentDisposition =[option]`		Controls whether the response body part displays inline (automatically upon receipt of the message). Defaults to inline. (Optional)
	`majorversion=[x]` and `minorversion=[x]`	long	Specify the required major and minor version for which the given rendition of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
Returns	Returns a filestream of required assets based on given parameters.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assetsbypath/Home/15116.qxp?rendition=preview` **Result:** A filestream of a generated preview of the given asset.		

Function	Get relationship with parent asset.		
Description	This request allows you to fetch the information of an asset and its relationship to parent assets.		
Type	GET		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`getAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`includeparents=true`	boolean	Parameter to fetch metadata of its parent assets. (Mandatory)
	`relationtypes=[comma-separated list of relationship names or IDs]`	long (relation type id) or string (relation type name)	You can retrieve the list of relationship types with `/admin/relationtypes`. (Optional) If not secified, the metadata related to all types of existing relations will be returned. The available relation types are: • Primary Attachment • Secondary Attachment

			• Article Component Ref Attachment • Overlay Attachment • XML Component Reference • InDesign Behavior Reference • InDesinn Hyperlink Reference
Returns	Returns an `assetInfo` object containing metadata of the relationship to the parent assets.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/[asset id]?includeparents=true` **Result:** The `assetInfoobject` containing metadata of the relationship to the parent assets.		

Function	Get relationship with child assets.		
Description	This request allows you to fetch the information of an asset and its relationship to its child assets.		
Type	GET		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`getAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`includechildren=true`	boolean	Parameter to fetch metadata of its child assets. (Mandatory)
	`relationtypes =[comma-separated list of relationship names or IDs]`	long (relation type id) or string (relation type name)	You can retrieve the list of relationship types with `/admin/relationtypes`. (Optional) If not secified, the metadata related to all types of existing relations will be returned. The available relation types are: • Primary Attachment • Secondary Attachment • Article Component Ref Attachment • Overlay Attachment • XML Component Reference • InDesign Behavior Reference • InDesinn Hyperlink Reference

Returns	Returns an `assetRelationInfoList`. object containing metadata of relations to parent and child assets. See the table in the "Request/Response parameter formats" chapter.
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/[asset id]?includechildren=true` **Result:** The `assetInfo` object containing metadata of the relationship to child assets.

Function	Get relationship with parent and child assets.		
Description	This request allows you to retrieve an assets information and information about its relationship to parent and child assets.		
Type	GET		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`getAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	includeparents=true& includechildren=true	boolean	Specify if the relation metadata to parent assets or child assets is to be included or not. (Mandatory)
Returns	Returns an `assetRelationInfoList`. object containing metadata of relations to parent and child assets. See the table in the "Request/Response parameter formats" chapter.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/[assetid]?includechildren=true&includeparents=true` **Result:** The `assetInfo` object containing metadata of relations to parent and child assets.		

Function	Get a Draft copy of an asset.		
Description	This request returns a draft version of the required asset. If the draft doesn't exists, the last saved version of the asset will be returned.		
Type	GET		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`getAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`attributes= [comma-separated list of attribute names or`	string	For a list of attributes, use a call to `admin/attributes`. (Mandatory)

	IDs]		
	draft=[true]	boolean	(Mandatory)
Returns	Returns an assetInfo object with the metadata of all attributes.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/assets/4?attributes=all&draft=true
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfo id="4">`
	`<assetVersion minorVersion="1" majorVersion="0"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id "id="1">4</attributeValue>`
	`<attributeValue type="text" name="Name" id="2">1quark2.jpg</attributeValue>`
	`....`
	`</attributeValueList>`
	`</assetInfo>`		

getAssetAllVersions

Function	Get all attributes for all versions of an asset.		
Description	This request returns all attributes for all versions of an asset.		
Type	GET		
URL Format	.../assetsbypath/[asset path] or /assets/[asset ID]		
Java API	**Method Name**		**Class Name**
	getAssetAllVersions		com.quark.qpp.service.facade Class AssetFacade
Parameters	**Name**	**Type**	**Description**
	op=allversions	string	This operation will list all the metadata of all the versions. (Mandatory)
Returns	Returns an assetInfoList object containg an array of all versions of the same asset with all of its' metadata. See the table in the "Request/Response parameter formats" chapter.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/assets/4?op=allversions
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfoList>`
	`<assetInfo id="4"assetVersion minorVersion="1"majorVersion="0"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">4</attributeValue>`
	`....`
	`</attributeValueList>`
	`</assetInfo>`		

	`<assetInfo id="4" assetVersion minorVersion="2" majorVersion="0"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">4</attributeValue>`
	`....`
	`</attributeValueList>`
	`</assetInfo>`
	`</assetInfoList>`

importAsset

Function	Import a new version of an existing asset.		
Description	This request imports the new version of an asset.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`importAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=import`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request will insert an asset into the system without giving its stream.
	`attributevalues=<xml/>`	string	The set of attribute values. See “Request/Response parameter formats.”(Mandatory)
	`createminorversion=[true|false]`	boolean	Defaults to `false`.(Optional)
	`index=[true|false]`	boolean	Defaults to `true`.(Optional)
Returns	Returns an `assetInfo` object in XML format.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/[asset name]?op=import`
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfo id="4">`
	`<assetVersion minorVersion="1" majorVersion="0"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">4</attributeValue>`
	`<attributeValue type="text" name="Name" id="2">1quark2.jpg</attributeValue>`
	`....`
	`</attributeValueList>`
	`</assetInfo>`
	Result:
	The `assetInfo` object in XML format.		

importNewAsset

Function	Import a new asset		
Description	This request imports a new asset.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset name]`		
Java API	**Method Name**		**Class Name**
	`importNewAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=importnew`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request will insert an asset into the system without giving its stream.
	`attributevalues=<xml/>`	string	The set of attribute values. See "Request/Response parameter formats." (Mandatory)
	`index=[true|false]`	boolean	Defaults to `true`. (Optional)
Returns	Returns an `assetInfo` object in XML format.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/[asset name]?op=importnew`
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <assetInfo id="4">`
	`  <assetVersion minorVersion="1" majorVersion="0"/>`
	`  <attributeValueList>`
	`  <attributeValue type="numeric" name="Id" id="1">4</attrib-uteValue>`
	`  <attributeValue type="text" name="Name" id="2">1quark2.jpg</attributeValue>`
	`  ....`
	`  </attributeValueList>`
	`  </assetInfo>`
	Result:
	The `assetInfo` object in XML format.		

updateAsset

Function	Update repository attributes.
Description	This requests updates the attribute values that are owned by a given repository adaptor.
Type	POST
URL Format	`.../assets/[asset id]?op=update&repotoken=[repo token]&attributes=....`

Java API	**Method Name**		**Class Name**
	`updateAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`attributes=<xml/>`	string	The set of attribute values. S ee "Request/Response parameter formats."(Mandatory)
	`repotoken=[token]`	GUID	The repository token provided to the repository adaptor during repository adaptor registration. (Mandatory)
	`majorversion=[x]` and `minorversion=[x]`	long	Specify the required major and minor version for which the given rendition of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
Returns	Returns an `assetInfo` object in XML format.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/assets/24?op=update&attributevalues=`
	`<attributeValueList>`
	`<attributeValue name="name">blag_new</attributeValue>`
	`<attributeValue name="status">Status2</attributeValue>`
	`<attributeValue name="workflow">Workflow2</attributeValue>`
	`</attributeValueList>`
	`&repotoken=[repository token]`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfo id="4">`
	`<assetVersion minorVersion="1" majorVersion="0"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">4</attributeValue>`
	`<attributeValue type="text" name="Name" id="2">1quark2.jpg</attributeValue>`
	`....`
	`</attributeValueList>`
	`</assetInfo>`		

Function	Update an asset's attributes.
Description	This request allows you to update attributes of specific version of an asset.

Type	POST		
URL Format	`/assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`updateAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`attributevalues=<xml/>`	string	The set of attribute values. See "Request/Response parameter formats."(Mandatory)
Returns	Return an `assetInfo` object of the specified asset with updated metadata.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/assets/[parent asset id]?op=update&attributevalues=`
	`<attributeValueList>`
	`<attributeValue name="name">blag_new</attributeValue>`
	`<attributeValue name="status">Status2</attributeValue>`
	`<attributeValue name="workflow">Workflow2</attributeValue>`
	`</attributeValueList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfo id="4">`
	`<assetVersion minorVersion="1" majorVersion="0"/> <attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">4</attributeValue>`
	`<attributeValue type="text" name="Name" id="2">1quark2.jpg</attributeValue>`
	`....`
	`</attributeValueList>`
	`</assetInfo>`		

duplicateAsset

Function	Duplicate an asset		
Description	This request creates a duplicate of an asset.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`duplicateAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=duplicate`	string	The op parameter allows you to specify which operation is to be performed.

			(Mandatory)
	`attributevalues=<xml/>`	string	The set of attribute values. See "Request/Response parameter formats." The following attributes are Mandatory: • NAME • COLLECTION (There is a choice between collection **ID**, **Name** and **Path** when providing a value for the **Collection** attribute).
	`createminorversion=[true|false]`	boolean	specify if the duplicated asset is to be created as a Major or Minor Version. Defaults to `false`. (Optional)
	`relationtypes=[comma-separated list of relationship names or IDs]`	string or long	You can retrieve the list of relationship types with `/admin/relationtypes`. (Optional)
Returns	Returns an `assetInfo` object of the newly created asset in XML format.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/4?op=duplicate&attributevalues=` `<attributeValueList> <attributeValue name="name">Duplicate_asset</attributeValue> <attributeValue name="collection">1</attributeValue> </attributeValueList> <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <assetInfo id="11">` `<assetVersion minorVersion="0" majorVersion="1"/>` `<attributeValueList>` `<attributeValue type="numeric" name="Id" id="1">11</attributeValue>` `<attributeValue type="text" name="Name" id="2">Duplicate_asset</attributeValue>` `....` `</attributeValueList>` `</assetInfo>` **Result:** The `assetInfo` object of the newly created asset in XML format.		

deleteAsset

Function	Delete an asset.
Description	This request deletes a specified asset from the QPP server.
Type	POST
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`

Java API	**Method Name**		**Class Name**
	`deleteAsset`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request will lock the asset internally and delete it from system.
	`majorversion=[x]` and `minorversion=[x]`	long	Specify the required major and minor version for which the given rendition of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
	`deleteminorver=True/False`	boolean	Indicates if all minor versions of the given asset version are to be deleted or not. If `True` - all minor versions (associated with the given major version) of the asset would also be deleted. If `False`, only the specified version will be deleted. This parameter would be considered only when the given asset version is major. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/25?op=delete`		

createAssetRelation

Function	Create a relationship between two assets.		
Description	This request creates a relationship between two assets.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	createAssetRelation		com.quark.qpp.service.facade Class AssetFacade
Parameters	**Name**	**Type**	**Description**

	`op=createrelation`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation requires parent asset to be in a Checked-Out state.
	`childassetid=[asset id]`	long	The ID of the other asset in the relationship. (Mandatory)
	`relationtype=[x]`	long	1 = primary attachment, 2 = secondary attachment. (Mandatory) The availabel relation types are: • Primary Attachment • Secondary Attachment • Article Component Ref Attachment • Overlay Attachment • XML Component Reference • InDesign Behavior Reference • InDesinn Hyperlink Reference
	`attributevalues=<xml/>`	string	The set of attribute values. See “Request/Response parameter formats.” For **Secondary Attachment** and **Overlay Attachment** the following attributes are Mandatory: • ATTACHED_LAYOUT_ID • BOX_ID • PAGE_NAME • PAGE_INDEX For **Primary Attachment** the following attribute is also Mandatory ins addition to the above mentioned four attributes: • ATTACHED_COMPONENT_ID For **ArticleComponentRef Attachment** the following two attributes are also Mandatory ins addition to the above mentioned four attributes: • ARTICLE_ID • ARTICLE_COMPONENT_ID
	`childmajorversion=[x]` and	**long**	Specify the required childmajor and childminor version for which the given

	`childminorversion=[x]`		rendition of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
	`lockedtochildversion=[true|false]`	**boolean**	Defaults to false. Allows you to pin relationships to a particular version if true. (Optional)
Returns	Returns an `assetRelationInfo` object containing metadata of the relation, parent and child asset. See the table in the "Request/Response parameter formats" chapter.		

Examples	**URL:**
	
	`http://serverName:serverPort/rest/service/assets/[parent asset Id]?op=createrelation&childassetid=[child asset Id]&relationtype=Secondary Attachment&attributevalues=`
	
	`<attributeValueList>`
	`  <attributeValue id="401">1</attributeValue>`
	`  <attributeValue id="402">5</attributeValue>`
	`  <attributeValue id="403">1</attributeValue>`
	`  <attributeValue id="404">1</attributeValue>`
	`  <attributeValue id="412">1</attributeValue>`
	`</attributeValueList>`
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <assetRelationInfo`
	`  relationStateId="3"`
	`  relationState="Related Temporarily"`
	`  relationTypeId="2"`
	`  relationType="Secondary Attachment"`
	`  childAssetMinorVersion="2"`
	`  childAssetMajorVersion="0"`
	`  childAssetName="1quark2.jpg"`
	`  childAssetId="4"`
	`  parentAssetName="15116.qxp"`
	`  parentAssetId="5"`
	`  id="13">`
	`  <attributeValueList>`
	`  <attributeValue type="numeric" name="Attached Layout Uid" id="401">1</attributeValue>`
	`  <attributeValue type="numeric" name="Box Uid" id="402">5</attributeValue>`
	`  <attributeValue type="text" name="Page Name" id="403">1</attributeValue>`
	`  <attributeValue type="numeric" name="Page Index" id="404">1</attributeValue>`
	`  <attributeValue type="numeric" name="Attached Component Id" id="412">1</attributeValue>`
	`  </attributeValueList>`
	`  </assetRelationInfo>`		

createAssetRelations

Function	Create several relationships between an asset and other assets at one time.		
Description	This request creates several relationships between an asset and other assets in a single request.		
Type	POST		
URL Format	`.../assetsbypath or /assets`		
Java API	**Method Name**		**Class Name**
	`createAssetRelations`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=createrelations`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request requires that the parent asset to be in checked-out state.

	`relations=<xml/>`		See "Request/Response parameter formats." (Mandatory)
			The `relationInfoList` returned will contain the metadata of all the relations that have to be created.
			The following is an example of creating several relationships at one time:
			Overlay Attachment:
			`<?xml version="1.0" encoding="UTF-8"?>`
			`<assetRelationInfoList>`
			`<assetRelationInfo id="112" parentAssetId="306" parentAssetName="15116d33.qxp" childAssetId="307" childAssetName="Picture2503.jpg" childAssetMinorVersion="1" relationType="Overlay Attachment" relationTypeId="3" relationState="Related Temporarily" relationStateId="3">`
			`<attributeValueList>`
			`<attributeValue id="401" name="Attached Layout Uid" type="numeric">1</attributeValue>`
			`<attributeValue id="402" name="Box Uid" type="numeric">17</attributeValue>`
			`<attributeValue id="403" name="Page Name" type="text">Page 1</attributeValue>`
			`<attributeValue id="404" name="Page Index" type="numeric">0</attributeValue>`

			`</attributeValueList> </assetRelationInfo> <assetRelationInfo id="113" parentAssetId="306" parentAssetName="15116d33.qxp" childAssetId="308" childAssetName="Picture2aba.jpg" childAssetMinorVersion="1" relationType="Overlay Attachment" relationTypeId="3" relationState="Related Temporarily" relationStateId="3"> <attributeValueList> <attributeValue id="401" name="Attached Layout Uid" type="numeric">1</attributeValue> <attributeValue id="402" name="Box Uid" type="numeric">17</attributeValue> <attributeValue id="403" name="Page Name" type="text">Page 1</attributeValue> <attributeValue id="404" name="Page Index" type="numeric">0</attributeValue> </attributeValueList> </assetRelationInfo> <assetRelationInfo id="114" parentAssetId="306" parentAssetName="15116d33.qxp" childAssetId="309" childAssetName="Picture2810.jpg" childAssetMinorVersion="1" relationType="Overlay Attachment" relationTypeId="3" relationState="Related Temporarily" relationStateId="3"> <attributeValueList> <attributeValue id="401" name="Attached Layout Uid" type="numeric">1</attributeValue> <attributeValue id="402" name="Box Uid" type="numeric">17</attributeValue> <attributeValue id="403" name="Page Name" type="text">Page 1</attributeValue> <attributeValue id="404" name="Page Index" type="numeric">0</attributeValue> </attributeValueList> </assetRelationInfo> <assetRelationInfo id="115" parentAssetId="306" parentAssetName="15116d33.qxp" childAssetId="310" childAssetName="Picture225d.jpg" childAssetMinorVersion="1" relationType="Overlay Attachment" relationTypeId="3" relationState="Related Temporar`

```
ily" relationStateId="3">
        <attributeValueList>
            <attributeValue
id="401" name="Attached Layout
Uid" type="numeric">1</attribute-
Value>
            <attributeValue
id="402" name="Box Uid" type="nu-
meric">17</attributeValue>
            <attributeValue
id="403" name="Page Name"
type="text">Page 1</attribute-
Value>
            <attributeValue
id="404" name="Page Index"
type="numeric">0</attributeValue>
        </attributeValueList>
    </assetRelationInfo>
    <assetRelationInfo id="116"
parentAssetId="306" parentAsset-
Name="15116d33.qxp" childAs-
setId="311"
childAssetName="Picture24da.jpg"
childAssetMinorVersion="1" rela-
tionType="Overlay Attachment" re-
lationTypeId="3"
relationState="Related Temporar-
ily" relationStateId="3">
        <attributeValueList>
            <attributeValue
id="401" name="Attached Layout
Uid" type="numeric">1</attribute-
Value>
            <attributeValue
id="402" name="Box Uid" type="nu-
meric">17</attributeValue>
            <attributeValue
id="403" name="Page Name"
type="text">Page 1</attribute-
Value>
            <attributeValue
id="404" name="Page Index"
type="numeric">0</attributeValue>
        </attributeValueList>
    </assetRelationInfo>
</assetRelationInfoList>
```


Primary Attachment:


```
<?xml version="1.0"
encoding="UTF-8"?>
<assetRelationInfoList>
    <assetRelationInfo parentAs-
setId="304" childAssetId="305"
relationType="Primary Attach-
ment">
        <attributeValueList>
            <attributeValue
id="401">1</attributeValue>
            <attributeValue
id="402">5</attributeValue>
            <attributeValue
id="403">Page 1</attributeValue>
            <attributeValue
id="404">0</attributeValue>
```

			`<attributeValue id="412">145</attributeValue> </attributeValueList> </assetRelationInfo> </assetRelationInfoList>`
Returns	Returns an `assetRelationInfo` object containing metadata of the relation, parent and child asset. See the table in the "RequestResponse parameter formats" chapter.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/assets/[parent asset id]?op=createrelations`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <assetRelationInfoList> <assetRelationInfo relationStateId="3" relationState="Related Temporarily" relationTypeId="2" relationType="Secondary Attachment" childAssetMinorVersion="2" childAssetMajorVersion="0" childAssetName="1quark2.jpg" childAssetId="4" parentAssetName="15116.qxp" parentAssetId="24" id="14"> <attributeValueList> <attributeValue type="numeric" name="Attached Layout Uid" id="401">1</attributeValue> <attributeValue type="numeric" name="Box Uid" id="402">5</attributeValue> <attributeValue type="text" name="Page Name" id="403">1</attributeValue> <attributeValue type="numeric" name="Page Index" id="404">1</attributeValue> <attributeValue type="numeric" name="Attached Component Id" id="412">1</attributeValue> </attributeValueList> </assetRelationInfo> </assetRelationInfoList>`		

getAssetRelations

Function	Get metadata of the relations between two assets	
Description	This request returns information about the relations between two assets.	
Type	GET	
URL Format	`.../assetsbypath/relations/[RelationId] or /assets/relations/[RelationId]`	
Java API	**Method Name**	**Class Name**
	`getAssetRelations`	`com.quark.qpp.service.facade Class AssetFacade`
Parameters	This request requires no input parameters.	
Returns		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/assets/relations/1`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <assetRelationInfo`
	`relationStateId="1"`
	`relationState="Permanent"`
	`relationTypeId="1"`
	`relationType="Primary Attachment"`
	`childAssetMinorVersion="1"`
	`childAssetMajorVersion="0"`
	`childAssetName="15348eb3.qcd"`
	`childAssetId="51"`
	`parentAssetName="15116bb0.qxp"`
	`parentAssetId="50" id="1">`
	`<attributeValueList>`
	`<attributeValue valueId="0" type="numeric" name="Attached Layout Uid" id="401">1</attributeValue>`
	`<attributeValue valueId="0" type="numeric" name="Box Uid" id="402">5</attributeValue>`
	`<attributeValue valueId="0" type="text" name="Page Name" id="403">Page 1</attributeValue>`
	`<attributeValue valueId="0" type="numeric" name="Page Index" id="404">0</attributeValue>`
	`<attributeValue valueId="0" type="numeric" name="Attached Component Id" id="412">10</attributeValue>`
	`</attributeValueList>`
	`</assetRelationInfo>`

updateAssetRelation

Function	Update the relation type and attributes of a relationship.		
Description	This request updates an asset relation with the given Id. In order to update the relation type, specify `newRelationType` and the corresponding attributes to be set. If the relation type is not supplied then it will update only the relation attributes for the current relation type of the given relation.		
Type	POST		
URL Format	`.../assets/relations/[relation ID]`		
Java API	**Method Name**		**Class Name**
	`updateAssetRelation`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request requires that the parent asset is in a checked-out state.
	`relationtype=[relation type name or ID]`	long (relation type id)	The new relation type. (Mandatory)

		or string (relation type name)	
	`attributevalues=<xml/>`	string	The set of relation attribute values. See “Request/Response parameter formats.” For **Secondary Attachment** and **Overlay Attachment** the following attributes are Mandatory: • ATTACHED_LAYOUT_ID • BOX_ID • PAGE_NAME • PAGE_INDEX For **Primary Attachment** the following attribute is also Mandatory ins addition to the above mentioned four attributes: ATTACHED_COMPONENT_ID For **ArticleComponent Ref Attachment** the following two attributes are also Mandatory ins addition to the above mentioned four attributes: • ARTICLE_ID • ARTICLE_COMPONENT_ID
Returns	Returns an `assetRelationInfo` object in XML format with an updated set of attributes. See the table in the “Request/Response parameter formats” chapter.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/relations/[relation id]?op=update`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetRelationInfo relationStateId="3"`
	`relationState="Related Temporarily"`
	`relationTypeId="2"`
	`relationType="Secondary Attachment"`
	`childAssetMinorVersion="2"`
	`childAssetMajorVersion="0"`
	`childAssetName="1quark2.jpg"`
	`childAssetId="4"`
	`parentAssetName="15116.qxp"`
	`parentAssetId="5" id="13">`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Attached Layout Uid" id="401">1</attributeValue>`
	`<attributeValue type="numeric" name="Box Uid" id="402">5</attributeValue>`		

	`<attributeValue type="text" name="Page Name" id="403">1</attributeValue>`
	`<attributeValue type="numeric" name="Page Index" id="404">1</attributeValue>`
	`<attributeValue type="numeric" name="Attached Component Id" id="412">1</attributeValue>`
	`</attributeValueList>`
	`</assetRelationInfo>`

deleteAssetRelations

Function	Delete a relationship by its id.		
Description	This request deletes a specific relationship between two assets.		
Type	POST		
URL Format	`.../assets/relations/[relation ID]`		
Java API	**Method Name**		**Class Name**
	deleteAssetRelations		com.quark.qpp.service.facade Class AssetFacade
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request requires that the parent asset is in a checked-out state.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/relations/[relation id]?op=delete`		

deleteAssetRelationsForParent

Function	Delete an asset's relationship.		
Description	This request removes all child relations of a specified asset.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`deleteAssetRelationsForParent`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=deleterelations`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request requires that the parent asset

			is in a checked-out state.
	`childassetid=[x]`	long	The ID of the child asset in the relationship to be deleted. (Optional) If this parameter is not supplied, all the asset's relation to its child assets will be deleted.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/27?op=deleterelations`		

generateTextIndexesOfAllAssets

Function	Generate text indexes for all assets.		
Description	This request asynchronously initiates text indexes for all Platform Server assets, including asset attributes and content.		
Type	POST		
URL Format	`.../assets/`		
Java API	**Method Name**		**Class Name**
	`generateTextIndexesOfAllAssets`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=generatetextindexes`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution, along with the following message: **The text indexes generation process has been initiated successfully.**		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets?op=generatetextindexes`		

generateTextIndexes

Function	Generate text indexes for a specified asset.	
Description	This requests initiates text indexes for a specified asset in Platform Server, including asset attributes and content.initiates.	
Type	POST	
URL Format	`.../assets/[asset id]?op=generatetextindexes...`	
Java API	**Method Name**	**Class Name**
	`generateTextIndexes`	`com.quark.qpp.service.facade`

			Class AssetFacade
Parameters	Name	Type	Description
	op=generatetextindexes	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	assetid=[asset Id]	long	The ID of the asset. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: http://serverName:serverPort/rest/service/assets/<assetId>?op=generatetextindexes		

getAssetVersionNumbers

Function	Get a list of all versions of an asset.		
Description	This request returns a list of versions for an asset.		
Type	GET		
URL Format	.../assetsbypath/[asset path] or /assets/[asset ID]		
Java API	Method Name		Class Name
	getAssetVersionNumbers		com.quark.qpp.service.facade Class AssetFacade
Parameters	Name	Type	Description
	op=listversions	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	Returns an assetVersionList object containing an array of all versions of the same asset.		
Examples	URL: http://serverName:serverPort/rest/service/assets/4?op=listversions Result: <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <assetVersionList> <assetVersion minorVersion="1" majorVersion="0"/> <assetVersion minorVersion="2" majorVersion="0"/> </assetVersionList>		

getAssetRenditionsInfo

Function	Get information about an asset's renditions.
Description	This request returns all information about an asset's reditions.
Type	GET
URL Format	.../assetsbypath/[asset path] or /assets/[asset ID]

Java API	**Method Name**		**Class Name**
	`getAssetRenditionsInfo`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=renditioninfo`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`rendition=preview` or `rendition=2`	string	By specifying rendition type, metadata of that rendition will be returned. (Optional) Available renditions are: • PREVIEW • THUMBNAIL • ARTICLE_PAGE_PICTURE • DECONSTRUCTED_XML • TEXT
	`majorversion=[x]` and `minorversion=[x]`	string	Specify the required major and minor version for which the given rendition of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
Returns	Returns an `assetRenditionInfoList` object containing an array of all available renditions of the same asset.		

Examples	**URL**:
	`http://serverName:serverPort/rest/service/assets/4?op=renditioninfo`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetRenditionInfoList>`
	`<assetRenditionInfo fileSize="80357" fileExtension="jpg" filePath="/00/04/4.0-2.2.1.jpg" repositoryId="1" pageNumber="1" layoutNumber="1" renditionTypeId="2" renditionType="PREVIEW" minorVersion="2" majorVersion="0" assetId="4"/>`
	`<assetRenditionInfo fileSize="47142" fileExtension="jpg" filePath="/00/04/4.0-2.3.1.jpg" repositoryId="1" pageNumber="1" layoutNumber="1" renditionTypeId="3" renditionType="THUMBNAIL" minorVersion="2" majorVersion="0" assetId="4"/>`
	`</assetRenditionInfoList>`		

getAvailableRenditions

Function	Get a list of an asset's available rendition types		
Description	This request returns a list of all the rendition types that are applicatble to a specified asset.		
Type	GET		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`getAvailableRenditions`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=listrenditiontypes`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`majorversion=[x]` and `minorversion=[x]`	string	Specify the required major and minor version for which the given rendition of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
Returns	Returns a `namedList` object containg an array of all renditions for the specified asset.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/assets/4?op=listrenditiontypes`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<nameIdList>`
	`<nameId id="2" name="PREVIEW"/>`
	`<nameId id="3" name="THUMBNAIL"/>`
	`</nameIdList>`		

updateChildVersionForAsset

Function	Update the version number of an asset's child asset.	
Description	This request allows you to update the version number of an asset's child asset. The parent asset must be in a checked out state.	
Type	POST	
URL Format	`.../assetsbypath/[parent asset path] or /assets/[parent asset ID]`	
Java API	**Method Name**	**Class Name**

	updateChildVersionForAsset		com.quark.qpp.service.facade Class AssetFacade
Parameters	**Name**	**Type**	**Description**
	op=updatechildversion	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)This operation requires parent asset to be in a Checked-Out state.
	childassetid=[asset ID]	long	The ID of the child asset. (Mandatory)
	majorversion=[x] and minorversion=[x]	long	Specify the required major and minor version for which the given rendition of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** http://serverName:serverPort/rest/service/assets/[parent asset id]?op=updatechildversion&childassetid=4		

updateChildAssetVersionForRelationId

Function	Update the version number of a relationship's child asset.		
Description	This request updates a child asset toa given asset version for the given asset relation.		
Type	POST		
URL Format	.../assets/relations/[relation ID]		
Java API	**Method Name**		**Class Name**
	updateChildAssetVersionForRelationId		com.quark.qpp.service.facade Class AssetFacade
Parameters	**Name**	**Type**	**Description**
	op=updatechildversion	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation requires the parent asset to be in a checked-out state.
	majorversion=[x] and minorversion=[x]	string	Specify the required major and minor version for which the given rendition of an asset is required. (Optional)

			These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/relations/[relation id]?op=updatechildversion`		

updateAllChildAssetsVersion

Function	Update the version number of all of an asset's child assets to the parent asset's current version.		
Description	Ths request allows you to update the asset versions of all the children of an asset at one time. the parent asset must be in a checked out state.		
Type	POST		
URL Format	`.../assetsbypath/[parent asset path] or /assets/[parent asset ID]`		
Java API	**Method Name**		**Class Name**
	updateAllChildAssetsVersion		com.quark.qpp.service.facade Class AssetFacade
Parameters	**Name**	**Type**	**Description**
	`op=updateallchildversion`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`relationtype=[x]`	long (relation type id) or string (relation type name)	You can retrieve the list of relationship types with /admin/relationtypes. (Optional) If the relation types are not specified, then the child asset version will be updated in all the relations with the given parent asset id.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/[parent asset id]?op=updateallchildversion`		

updateChildVersionLockingFlag

Function	Update the "locked" flag for a child asset in a relationship.

Description	This request updates the lockedToChildVersion flag of the given Asset relation.		
Type	POST		
URL Format	.../assets/relations/[relation ID]		
Java API	**Method Name**		**Class Name**
	updateChildVersionLockingFlag		com.quark.qpp.service.facade Class AssetFacade
Parameters	**Name**	**Type**	**Description**
	`op=updatelockflag`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`flag=[true or false]`	boolean	**True** for locked, **false** for unlocked. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/relations/19?op=updatelockflag&flag=true`		

checkIn

Function	Check in a checked out asset.		
Description	This request checks in a checked out asset. Include the file in the request body or as file part.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	checkIn		com.quark.qpp.service.facade Class AssetFacade
Parameters	**Name**	**Type**	**Description**
	`op=checkin`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request will not work if the asset is not checked out.
	`attributevalues=<xml/>`		See "Request/Response parameter formats." List of attribute values which need to be updated.
	`relations=<xml/>`		See "Request/Response parameter formats." If given a value in this request parameter, the Quark Publishing Platform Server will

			check-in a checked-out asset and create a relation with the secified parent asset at the same time. (Optional) For **Secondary Attachment** and **Overlay Attachment** the following attributes are Mandatory: • ATTACHED_LAYOUT_ID • BOX_ID • PAGE_NAME • PAGE_INDEX For **Primary Attachment** the following attribute is also Mandatory ins addition to the above mentioned four attributes: • ATTACHED_COMPONENT_ID For **ArticleComponent Ref Attachment** the following two attributes are also Mandatory ins addition to the above mentioned four attributes: • ARTICLE_ID • ARTICLE_COMPONENT_ID
	`createminorversion =[true | false]`	boolean	Specify whether to create a minor version or not. Defaults to **false**. (Optional)
Returns	Returns the ass`etInfo` object of the latest version of the asset to be checked into the QPP server.		

Examples	**URL:**
	
	`http://serverName:serverPort/rest/service/assets/4?op=checkin&attributevalues=`
	
	`<attributeValueList >`
	`<attributeValue id="2">1quark2.jpg</attributeValue>`
	`<attributeValue id="54">10</attributeValue>`
	`<attributeValue id="55">1</attributeValue>`
	`<attributeValue id="24">10</attributeValue>`
	`<attributeValue id="11">jpg</attributeValue>`
	`<attributeValue id="33">1quark2.jpg</attributeValue>`
	`<attributeValue id="25">admin</attributeValue>`
	`</attributeValueList>`
	`<assetInfo id="4">`
	`<assetVersion minorVersion="1" majorVersion="0"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">4</attributeValue>`
	`<attributeValue type="text" name="Name" id="2">1quark2.jpg</attributeValue>`
	`....`
	`</attributeValueList>`
	`</assetInfo>`		

	Result: The `assetInfo` object of the latest version of the asset to be checked into the QPP server.

checkInNew

Function	Check in a new asset.		
Description	Include the file in the request body or as file part.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[New Asset Name]`		
Java API	**Method Name**		**Class Name**
	`checkInNew`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=checkinnew`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)

	`attributevalues=<xml/>`		See "Request/Response parameter formats."
			`<attributeValueList>`
			`<attributeValue id="62"> 1103 </attributeValue>`
			`<attributeValue id="54"> 10 </attributeValue>`
			`<attributeValue id="24"> 10 </attributeValue>`
			`<attributeValue id="11"> jpg </attributeValue>`
			`<attributeValue id="33"> Original file </attributeValue>`
			`<attributeValue id="25"> 1 </attributeValue>`
			`</attributeValueList>`
			The following attributes are Mandatory:
			• Content type (There is a choice between content type **ID**, **Name** and **Hierarchy** when providing a value for the **Content type** attribute.)
			• Collection (There is a choice between collection **ID**, **Name** and **Path** when providing a value for the **Collection** attribute).
			• Workflow
			• Status
			• File extension

			• Original filename
	`relations=<xml/>`		See "Request/Response parameter formats." If this parameter is passed, the QPP Server will check-in the asset and create child relations with the new asset being checked in at the same time. (Optional).
	`createminorversion =[true | false]`	boolean	Specify if this version of the asset is to be checked in to the QPP server as a major or a minor version. (Optional)
Returns	Returns an `assetInfo` object of the newly checked in asset with its metadata.		

Examples	**URL**:
	
	`http://serverName:serverPort/rest/service/assetsbypath/Home/Tulips?attributevalues=`
	
	`<attributeValueList>`
	`<attributeValue id="62">System;Asset;Picture;</attributeValue>`
	`<attributeValue id="54">10</attributeValue>`
	`<attributeValue id="24">10</attributeValue>`
	`<attributeValue id="11">jpg</attributeValue>`
	`<attributeValue id="33">Original file</attributeValue>`
	`<attributeValue id="25">1</attributeValue>`
	`</attributeValueList>`
	`&op=checkinnew`
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfo id="10">`
	`<assetVersion minorVersion="0" majorVersion="1"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">10</attributeValue>`
	`<attributeValue type="text" name="Name" id="2">Tulips</attributeValue>`
	`<attributeValue type="dateTime" name="Created" id="3">2013-04-11T05:12:51Z</attributeValue>`
	`<attributeValue type="dateTime" name="Last modified" id="4">2013-04-11T05:12:51Z</attributeValue>`
	`<attributeValue type="popup" name="Creator" id="5">Admin</attributeValue>`
	`<attributeValue type="popup" name="Last modifier" id="6">Admin</attributeValue>`
	`<attributeValue type="numeric" name="Major version" id="7">1</attributeValue>`
	`<attributeValue type="text" name="File path" id="8">/00/0a/10.1-0.1.1.jpg</attributeValue>`
	`<attributeValue type="text" name="File extension" id="11">jpg</attributeValue>`
	`<attributeValue type="numeric" name="File size" id="12">115191</attributeValue>`
	`<attributeValue type="boolean" name="Is checked out" id="13">false</attributeValue>`
	`<attributeValue type="dateTime" name="Last content update" id="19">2013-04-11T05:12:51Z</attributeValue>`
	`<attributeValue type="popup" name="Last content updated by" id="20">Admin</attributeValue>`		

```
      <attributeValue type="popup" name="Routed to"
id="25">Admin</attributeValue>
      <attributeValue type="popup" name="Status"
id="24">Status1</attributeValue>
      <attributeValue type="popup" name="Repository type"
id="27">fileRepositoryAdapter</attributeValue>
      <attributeValue type="popup" name="Indexing status"
id="29">Pending</attributeValue>
      <attributeValue type="popup" name="Repository"
id="28">QPS_Repo</attributeValue>
      <attributeValue type="boolean" name="Is placeholder"
id="30">false</attributeValue>
      <attributeValue type="popup" name="Content creator"
id="32">Admin</attributeValue>
      <attributeValue type="text" name="Original filename"
id="33">Original file</attributeValue>
      <attributeValue type="popup" name="Collection"
id="55">Home</attributeValue>
      <attributeValue type="popup" name="Workflow" id="54">Work-
flow1</attributeValue>
      <attributeValue type="popup" name="Content Type"
id="62">Picture</attributeValue>
      <attributeValue type="numeric" name="Minor version"
id="61">0</attributeValue>
      <attributeValue type="boolean" name="Has Children"
id="65">false</attributeValue>
      <attributeValue type="text" name="Collection Path"
id="57">Home</attributeValue>
      <attributeValue type="text" name="Content Type Hierarchy"
id="63">System;Asset;Picture</attributeValue>
    </attributeValueList>
  </assetInfo>
```


checkInArticle

Function	Check in a checked-out QuarkCopyDesk article with components.		
Description	This request checks in a checked out article. Include the file in the request body or as file part.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset name]`		
Java API	**Method Name**		**Class Name**
	`checkInArticle`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=checkinarticle`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request will not work if the asset is not checked out.
	`attributevalues=<xml/>`		See “Request/Response parameter formats.” List of attribute values which need to be

			updated.
	relations=<xml/>		See "Request/Response parameter formats." If given a value in this request parameter, the Quark Publishing Platform Server will check-in a checked-out asset and create a relation with the secified parent asset at the same time. (Optional)
	createminorversion =[true | false]	boolean	Specify whether to create a minor version or not. Defaults to **false**. (Optional)
Returns	Returns the assetInfo object of the latest version of the asset to be checked into the QPP server.		

Examples

URL:

```
http://serverName:serverPort/rest/service/assetsbypath/Home/15116d12.qcd?op=checkin&attributevalues=
  <attributeValueList>
    <attributeValue id="2">15348a77.qcd</attributeValue>
    <attributeValue id="54">Workflow1</attributeValue>
    <attributeValue id="55">Home</attributeValue>
    <attributeValue id="24">Status1</attributeValue>
    <attributeValue id="11">qcd</attributeValue>
    <attributeValue id="33">15348a77.qcd</attributeValue>
    <attributeValue id="25">1</attributeValue>
  </attributeValueList>
& components=
  <articleComponentInfoList>
    <articleComponentInfo componentId="15">
      <attributeValueList>
        <attributeValue name="Component Name">Component 1</attributeValue>
       <attributeValue name="Content Type">1250</attributeValue>
        <attributeValue name="Article Component Id">1</attributeValue>
      </attributeValueList>
    </articleComponentInfo>
    <articleComponentInfo componentId="14">
      <attributeValueList>
        <attributeValue name="Component Name">Component2</attributeValue>
        <attributeValue name="Content Type">1300</attributeValue>
        <attributeValue name="Article Component Id">2</attributeValue>
        </attributeValueList>
    </articleComponentInfo>
  </articleComponentInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <assetInfo id="53">
    <assetVersion minorVersion="0" majorVersion="1"/>
    <attributeValueList>
      <attributeValue valueId="0" type="numeric" name="Id" id="1">53</attributeValue>
```

```
      <attributeValue valueId="0" type="text" name="Name"
id="2">15116d12.qcd</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Created"
id="3">2014-04-09T09:51:45Z</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Last mod-
ified" id="4">2014-04-09T09:51:45Z</attributeValue>
      <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
      <attributeValue valueId="1" type="popup" name="Last modi-
fier" id="6">Admin</attributeValue>
      <attributeValue valueId="0" type="numeric" name="Major ver-
sion" id="7">1</attributeValue>
      <attributeValue valueId="0" type="text" name="File path"
id="8">/00/35/53.1-0.1.1.qcd</attributeValue>
      <attributeValue valueId="0" type="text" name="File exten-
sion" id="11">qcd</attributeValue>
      <attributeValue valueId="0" type="numeric" name="File size"
id="12">284672</attributeValue>
      <attributeValue valueId="0" type="boolean" name="Is checked
out" id="13">false</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Last con-
tent update" id="19">2014-04-09T09:51:45Z</attributeValue>
      <attributeValue valueId="1" type="popup" name="Last content
updated by" id="20">Admin</attributeValue>
      <attributeValue valueId="1" type="popup" name="Routed to"
id="25">Admin</attributeValue>
      <attributeValue valueId="10" type="popup" name="Status"
id="24">Status1</attributeValue>
      <attributeValue valueId="1" type="popup" name="Repository
type" id="27">fileRepositoryAdapter</attributeValue>
    </attributeValueList>
  </assetInfo>
```


checkInNewArticle

Function	Check in a new QuarkCopyDesk Article with components.		
Description	Include the file in the request body or as file part.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset name]`		
Java API	**Method Name**		**Class Name**
	`checkInNewArticle`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=checkinnewarticle`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)

	`attributevalues=<xml/>`		See "Request/Response parameter formats."
			 `<attributeValueList>`
			 `<attributeValue id="62"> 1103 </attributeValue>`
			 `<attributeValue id="54"> 10 </attributeValue>`
			 `<attributeValue id="24"> 10 </attributeValue>`

			<attributeValue id="11"> jpg </attributeValue> <attributeValue id="33">Original file </attributeValue> <attributeValue id="25"> 1 </attributeValue> </attributeValueList> The following attributes are Mandatory: • Content type (There is a choice between content type **ID**, **Name** and Hierarchy when providing a value for the **Content type** attribute.) • Collection (There is a choice between collection **ID**, **Name** and **Path** when providing a value for the **Collection** attribute) • Workflow • Status • File extension • Original filename
	`relations=<xml/>`		See "Request/Response parameter formats." If this parameter is passed, the QPP Server will check-in the asset and create child relations with the new asset being checked in at the same time. (Optional).
	`createminorversion =[true | false]`	boolean	Specify if this version of the asset is to be checked in to the QPP server as a major or a minor version. (Optional)
Returns	Returns an `assetInfo` object of the newly checked in asset with its metadata.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/15116d12.qcd?op=checkinnew&attributevalues=` <?xml version="1.0" encoding="UTF-8"?> <attributeValueList> <attributeValue id="62">11011</attributeValue> <attributeValue name="workflow">Workflow1</attributeValue> <attributeValue name="collection">1</attributeValue> <attributeValue id="24">10</attributeValue> <attributeValue id="11">qcd</attributeValue> <attributeValue id="33">15348.qcd</attributeValue> <attributeValue name="routed to">1</attributeValue> </attributeValueList> &createminorversion=false&components= <articleComponentInfoList> <articleComponentInfo> <attributeValueList> <attributeValue name="Component Name">Component1</attributeValue> <attributeValue name="Content Type">1250</attributeValue>		

```
        <attributeValue name="Article Component Id">1</attribute-
Value>
      </attributeValueList>
    </articleComponentInfo>
    <articleComponentInfo>
       <attributeValueList>
        <attributeValue name="Component Name">Component 2</at-
tributeValue>
        <attributeValue name="Content Type">1300</attributeValue>
        <attributeValue name="Article Component Id">2</attribute-
Value>
      </attributeValueList>
    </articleComponentInfo>
  </articleComponentInfoList>
```


Result:


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <assetInfo id="53">
    <assetVersion minorVersion="0"majorVersion="1"/>
    <attributeValueList>
      <attributeValue valueId="0" type="numeric" name="Id"
id="1">53</attributeValue>
      <attributeValue valueId="0" type="text" name="Name"
id="2">15116d12.qcd</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Created"
id="3">2014-04-09T09:51:45Z</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Last mod-
ified" id="4">2014-04-09T09:51:45Z</attributeValue>
      <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
      <attributeValue valueId="1" type="popup" name="Last modi-
fier" id="6">Admin</attributeValue>
      <attributeValue valueId="0" type="numeric" name="Major ver-
sion" id="7">1</attributeValue>
      <attributeValue valueId="0" type="text" name="File path"
id="8">/00/35/53.1-0.1.1.qcd</attributeValue>
      <attributeValue valueId="0" type="text" name="File exten-
sion" id="11">qcd</attributeValue>
      <attributeValue valueId="0" type="numeric" name="File size"
id="12">284672</attributeValue>
      <attributeValue valueId="0" type="boolean" name="Is checked
out" id="13">false</attributeValue>
      <attributeValue valueId="0" type="dateTime" name="Last con-
tent update" id="19">2014-04-09T09:51:45Z</attributeValue>
      <attributeValue valueId="1" type="popup" name="Last content
updated by" id="20">Admin</attributeValue>
      <attributeValue valueId="1" type="popup" name="Routed to"
id="25">Admin</attributeValue>
      <attributeValue valueId="10" type="popup" name="Status"
id="24">Status1</attributeValue>
      <attributeValue valueId="1" type="popup" name="Repository
type" id="27">fileRepositoryAdapter</attributeValue>
    </attributeValueList>
  </assetInfo>
```


checkInRendition

Function	Check in a rendition of an asset.
Description	Include the file in the request body or as file part.
Type	POST

URL Format	`.../assetsbypath/[asset path] or /assets/[assetId]`		
Java API	**Method Name**		**Class Name**
	`checkInRendition`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=checkinrendition`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`rendition=highres` or `rendition=1`	string	Specify the required rendition type. (Optional) If the paramater is not passed the system returns the highres rendition type of the asset.Available renditions types are:PREVIEWTHUMBNAILARTICLE_PAGE_PICTUREDECONSTRUCTED_XMLTEXT
	`fileext=[file extension]`	string	The file extension of the rendition files being checked in. (Mandatory)
	`pages=[pages]`	long	The number of pages contained in the rendition being created. (Optional)
	`majorversion=[x]` and `minorversion=[x]`	long	Specify the required major and minor version for which the given rendition of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/4?rendition=preview&fileext=jpg& majorversion=0&minorversion=2&op=checkinrendition`		

checkOut

Function	Check out an asset		
Description	This request checks out the specified asset and returns its stream.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`checkOut`		`com.quark.qpp.service.facade Class AssetFacade`

	Name	Type	Description
Parameters	op=checkout	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)

	attributevalues=<xml/>	string in XML format	See "Request/Response parameter formats."
			`<attributeValueList>`
			`<attributeValue id="16">`
			`E:\REST\QPP Rest\TestData\Check-outFilePath </attributeValue>`
			`<attributeValue id="34"> REST _CLIENT </attributeValue>`
			`</attributeValueList>`
			The following attributes are Mandatory:
			• CHECKOUT_FILE_PATH
			• CHECKOUT_APPLICATION

Returns	Returns an asset stream after marking it checked out.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/assets/4?attributevalues=
	`<attributeValueList>`
	`<attributeValue id="16">E:\REST\QPP Rest\TestData\Checkout-FilePath</attributeValue>`
	`<attributeValue id="34">REST _CLIENT</attributeValue>`
	`</attributeValueList>`
	&op=checkout
	Result:
	An asset stream after marking it checked out.		

abortCheckout

Function	Abort checkout of an asset		
Description	This request aborts the check-out of an asset that is marked as checked-out.		
Type	POST		
URL Format	.../assetsbypath/[asset path] or /assets/[asset ID]		
Java API	Method Name		Class Name
	abortCheckout		com.quark.qpp.service.facade Class AssetFacade
Parameters	Name	Type	Description
	op=abortcheckout	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request will not work if the asset is not checked out.
Returns	Returns an assetInfo object in XML format.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/assets/4?op=abortcheckout`
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <assetInfo id="4">`
	`  <assetVersion minorVersion="2" majorVersion="0"/>`
	`  <attributeValueList>`
	`  <attributeValue type="numeric" name="Id"id="1">4</attributeValue>`
	`  <attributeValue type="text" name="Name"id="2">1quark2.jpg</attributeValue>`
	`  <attributeValue type="dateTime" name="Created" id="3">2013-04-10T09:37:56Z</attributeValue>`
	`  .....`
	`  </attributeValueList>`
	`  </assetInfo>`
	Result:
	The `assetInfo` object in XML format.

saveDraft

Function	Saves the draft copy of an asset.		
Description	This request saves the asset without creating a new version of the asset.		
Type	POST		
URL Format	`/assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`saveDraft`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=savedraft`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) Include the file in the request body or as file part.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/{asset Id}?op=savedraft`		

promoteDraft

Function	Promotes the draft version to the actual version of an asset.
Description	This request saves the asset without creating a new version of the asset.
Type	POST
URL Format	`/assets/[asset ID]`

Java API	Method Name		Class Name
	promoteDraft		com.quark.qpp.service.facade Class AssetFacade
Parameters	Name	Type	Description
	op=promotedraft	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	attributes= [comma-separated list of attribute names or IDs]	string	List of attribute values to be set for this new asset version. For a list of attributes, use a call to admin/attributes. (Mandatory)
	Createminorversion={true|false}	string	By default value is false.
Returns	Returns an assetInfo object along with a list of attribute values for this new asset version.		

Examples	URL:
	http://serverName:serverPort/rest/service/assets/{assetId}?op=promotedraft
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfo id="40">`
	`<assetVersion minorVersion="0" majorVersion="3"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">40</attributeValue>`
	`<attributeValue type="text" name="Name" id="2">abc.xlsx</attributeValue>`
	`<attributeValue type="dateTime" name="Created" id="3">2014-08-18T07:23:30Z</attributeValue>`
	`....`
	`<attributeValueList/>`
	`</assetInfo>`		

detectContentType

Function	Detects content type of provided asset. Pass asset as stream in the request body.		
Description	This request detects the content type of a provided asset.		
Type	POST		
URL Format	.../assets		
Java API	Method Name		Class Name
	detectContentType		com.quark.qpp.service.facade Class AssetFacade
Parameters	Name	Type	Description
	fileextension	string	The file extension of the asset.
	mimetype	string	The mime type of the asset.
Returns			

Examples	**URL:**
	`http://serverName:serverPort/rest/service/assets?op=detectcontent type`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <contentTypeInfo id="11011">`
	`  <name>QuarkCopyDesk Article</name>`
	`  <contentTypeHierarchy>System;Asset;Document;QuarkCopyDesk Article;</contentTypeHierarchy>`
	`  <systemDefined>true</systemDefined>`
	`  <childrenAvailable>false</childrenAvailable>`
	`  <extensible>true</extensible>`
	`  </contentTypeInfo>`

reIndex

Function	Re-index an asset using the asset id.		
Description	Re-indexes the given asset. It generates renditions of an asset such as previews, thumbnails, text renditions etc.		
Type	POST		
URL Format	`/assets/[asset ID]`		
Java API	**Method Name**		**Class Name**
	`reIndex`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=reindex`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assets/{asset Id}?op=reindex`		

Function	Re-index an asset using the asset path.		
Description	Re-indexes the given asset. It generates renditions of an asset such as previews, thumbnails, text renditions etc.		
Type	POST		
URL Format	`/assetsbypath/{collectionPath}/[AssetName]`		
Java API	**Method Name**		**Class Name**
	`reIndex`		`com.quark.qpp.service.facade Class AssetsByPathFacade`
Parameters	**Name**	**Type**	**Description**
	`op=reindex`	string	The op parameter allows you to specify which operation is to be performed.

			(Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/assetsbypath/{Collection Hierarchy}/{AssetName}?op=reindex`		

Collection management

The collection management capabilities of the API allow application developers to manage collections and includes the following REST calls:

- *createCollection*
- *createCollections*
- *getCollection*
- *getCollectionHierarchy*
- *getCollectionDefaultJobJacket*
- *getCollectionDefaultTicket*
- *moveCollection*
- *duplicateCollection*
- *updateCollection*
- *updateCollections*
- *deleteCollection*
- *deleteCollectionWithAssets*
- *createJobJackets*
- *getJobJacket*
- *getAllJobJackets*
- *getJobJacketContent*
- *updateJobJacket*
- *updateJobJackets*
- *parseJobJacketXml*
- *deleteJobJacket*
- *getApplicableRoutingUsers*
- *getApplicablePrivileges*
- *getApplicableRoutingGroups*
- *getApplicableAttributeConstraints*
- *getApplicableStatusTransition*
- *getApplicableWorkflowInitialStatuses*

createCollection

Function	Create a collection.		
Description	This request creates a new collection according to the given attributes.		
Type	POST		
URL Format	`.../collections/[collection name] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`createCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`targetparent=[collection path or ID]`	long or string	The parent collection for the new collection. (Mandatory) There is a choice between collection **ID**, **Name** and **Path** when providing a value.
	`attributevalues=<xml/>`	string	The set of attribute values. See "Request/Response parameter formats."(All atributes are Optional)
	`contenttype=[content type name]`	string	Defaults to **GENERAL**. The default content type IDs are documented in the JavaDoc SDK at `com.quark.qpp.core.content.service.constants.DefaultContentTypes`. To get a list of content types, use a call to `../admin/contenttypes`.(Optional) There is a choice between content type **ID**, **Name** and **Hierarchy** when providing a value.
Returns	Returns the `collectionInfo` object of the newly created collection.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/collections/newChildCollection? op=create&targetparent=Home/childCollection`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<collectionInfo templateId="0"`
	`collectionClass="General"`
	`collectionClassId="1416"`
	`collectionTemplate="false"`
	`accessibleChildrenAvailable="true"`
	`domainId="19"`
	`name="newChildCollection"`		

```
  id="15">
   <attributeValueList>
     <attributeValue type="numeric"
      name="Id"
      id="1">1</attributeValue>
     ...
   </attributeValueList>
   <workflowInfoList>
     <workflowInfo id="10">
     <name>Workflow1</name>
     ...
       <statusInfoList>
         <statusInfo id="10">
         <name>Status1</name>
         ...
       </statusInfo>
     </statusInfoList>
   </workflowInfo>
   ....
   ....
 </collectionInfo>
```

Function	Create a collection template.		
Description	This request creates a new collection template under the given parent collection.		
Type	POST		
URL Format	`.../collections/[collection name] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`createCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`istemplate=[true|false]`	boolean	Defaults to **false**. (Mandatory)
	`targetparent=[collection path or ID]`	long or string	The parent collection for the new collection. (Mandatory)
	`attributevalues=<xml/>`	string	The set of attribute values. See "RequestResponse parameter formats." All attributes are Optional.
	`contenttype=[content type name]`	string	Defaults to **GENERAL**. The default content type IDs are documented in the JavaDoc SDK at `com.quark.qpp.core.content.service.constants.DefaultContentTypes`. To get a list of content types, use a call to `../admin/contenttypes`.(Optional)
Returns	Returns the `collectionInfo` object of the newly created collection template.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/collections/newChildCollection?op=create&targetparent=Home/childCollection&istemplate=true`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <collectionInfo templateId="0"`
	`  collectionClass="General"`
	`  collectionClassId="1416"`
	`  collectionTemplate="false"`
	`  accessibleChildrenAvailable="true"`
	`  domainId="19"`
	`  name="newChildCollection"`
	`  id="15">`
	`  <attributeValueList>`
	`  <attributeValue type="numeric" name="Id" id="1">1</attributeValue>`
	`  ...`
	`  </attributeValueList>`
	`  <workflowInfoList>`
	`  <workflowInfo id="10">`
	`  <name>Workflow1</name>`
	`  ...`
	`  <statusInfoList>`
	`  <statusInfo id="10">`
	`  <name>Status1</name>`
	`  ...`
	`  </statusInfo>`
	`  </statusInfoList>`
	`  </workflowInfo>`
	`  ....`
	`  ....`
	`  </collectionInfo>`

createCollections

Function	Create multiple collections.		
Description	This request creates multiple collections in a single request.		
Type	POST		
URL Format	`.../collections`		
Java API	**Method Name**		**Class Name**
	`createCollections`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
	`collectioninfolist=<xml/>`		See “Request/Response parameter formats.”
Returns	Returns a `collectioninfolist` object in XML format that contains the metadata		

	of all successfully created collections.

Examples	**URL:**
	https://serverName:serverPort/rest/service/collections?op=create&collectioninfolist=
	`<?xml version="1.0" encoding="UTF-8"?>`
	`  <collectionInfoList>`
	`  <collectionInfo collectionClassId="1416"`
	`  collectionTemplate="false"`
	`  collectionPath="Home/Home3" >`
	`  <attributeValueList>`
	`  <attributeValue valueId="0" type="text" name="Name" id="2">Home3</attributeValue>`
	`  <attributeValue valueId="0" type="dateTime" name="Created" id="3">2010-04-01T10:00:00Z</attributeValue>`
	`  <attributeValue valueId="0" type="dateTime" name="Last modified" id="4">2010-04-01T10:00:00Z</attributeValue>`
	`  <attributeValue valueId="1" type="popup" name="Creator" id="5">Admin</attributeValue>`
	`  <attributeValue valueId="1" type="popup" name="Last modifier" id="6">Admin</attributeValue>`
	`  <attributeValue valueId="0" type="boolean" name="Is Template" id="59">false</attributeValue>`
	`  <attributeValue valueId="0" type="text" name="Collection Path" id="57">Home/Home1</attributeValue>`
	`  <attributeValue valueId="1416" type="popup" name="Content Type" id="62">General</attributeValue>`
	`  <attributeValue valueId="0" type="text" name="Content Type Hierarchy" id="63">System;Collection;General</attributeValue>`
	`  </attributeValueList>`
	`  </collectionInfo>`
	`  </collectionInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <collectionInfoList>`
	`  <collectionInfo`
	`  defaultJobJacketTicket="Default Job Ticket"`
	`  defaultJobJacketName="Default Job Jacket"`
	`  defaultJobJacketId ="1"`
	`  templateId="0"`
	`  collectionClass="General"`
	`  collectionClassId="1416"`
	`  collectionTemplate="false"`
	`  accessibleChildrenAvailable="false"`
	`  domainId="19"`
	`  collectionPath="Home/Home3"`
	`  name="Home3"`
	`  id="23">`
	`  <attributeValueList>`
	`  <attributeValue valueId="0" type="numeric" name="Id" id="1">23</attributeValue>`
	`  <attributeValue valueId="0" type="text" name="Name" id="2">Home3</attributeValue>`
	`  <attributeValue valueId="0" type="dateTime" name="Created" id="3">2014-04-10T06:09:46Z</ attributeValue >`
	`  ....`
	`  </attributeValueList>`
	`  <workflowInfoList/>`
	`  <routingInfoList/>`
	`  <userInfoList>`
	`  <userInfo id="1">`

```
      <name>Admin</name >
       .....
    </userInfo>
  </userInfoList>
<groupInfoList/>
  <jobJacketInfoList>
    <jobJacketInfo id="1">
      <name>Default Job Jacket</name>
       .....
    </jobJacketInfo>
  </jobJacketInfoList>
</collectionInfoList>
</collectionInfo>
</collectionInfoList>
```

getCollection

Function	Get all of a collection's metadata.	
Description	This request returns all of the metadata associated with an entire collection.	
Type	GET	
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`	
Java API	**Method Name**	**Class Name**
	`getCollection`	`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `collectionInfo` object with all its metadata and a list of its attributes.	
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/1` **Result:**	

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <collectionInfoList>
    <collectionInfo
     defaultJobJacketTicket="Default Job Ticket"
     defaultJobJacketName="Default Job Jacket"
     defaultJobJacketId="1"
     templateId="0"
     collectionClass="General"
     collectionClassId="1416"
     collectionTemplate="false"
     accessibleChildrenAvailable="true"
     domainId="19"
     collectionPath="Home"
     name="Home"
     id="1">
      <attributeValueList>
        <attributeValue valueId="0" type="numeric" name="Id"
id="1">1</attributeValue>
        <attributeValue valueId="0" type="text" name="Name"
id="2">Home</attributeValue>
        <attributeValue valueId="0" type="dateTime" name="Cre-
ated" id="3">2010-04-01T10:00:00Z</attributeValue>
        <attributeValue valueId="0" type="dateTime" name="Last
modified" id="4">2013-12-23T12:07:27Z</attributeValue>
```

```
        <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
        <attributeValue valueId="1" type="popup" name="Last modi-
fier" id="6">Admin</attributeValue>
        <attributeValue valueId="0" type="boolean" name="Is Tem-
plate" id="59">false</attributeValue>
        <attributeValue valueId="0" type="text" name="Collection
Path" id="57">Home</attributeValue>
        <attributeValue valueId="1416" type="popup" name="Content
Type" id="62">General</attributeValue>
        <attributeValue valueId="0" type="text" name="Content
Type Hierarchy" id="63">System;Collection;General</attribute-
Value>
      </attributeValueList>
    </collectionInfo>
  </collectionInfoList>
```

Function	Get specific collection attributes.		
Description	This request returns all of the attrubutes associated with a specific collection.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`attributes=[attribute name or ID], [attribute name or ID],...`	string	To get a list of all attribute IDs, use a call to `../admin/attributes`. (Mandatory)
Returns	Returns a `collentionInfo` object with the specified attributes.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/collections/1?attributes=1,name,creator`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <collectionInfo id="1">`
	`  <attributeValueList>`
	`  <attributeValue type="numeric" name="Id" id="1">1</attributeValue>`
	`  <attributeValue type="text" name="Name" id="2">Home</attributeValue>`
	`  <attributeValue type="popup" name="Creator" id="5">Admin</attributeValue>`
	`  </attributeValueList>`
	`  </collectionInfo>`		

Function	Get a collection's workflows.
Description	This request returns all of the workflows associated with a collection.
Type	GET
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection`

	path]		
Java API	**Method Name**		**Class Name**
	getCollection		com.quark.qpp.service.facade Class CollectionFacade
Parameters	**Name**	**Type**	**Description**
	workflows=[true|false]	boolean	Defaults to **false**. (Mandatory)
Returns	Returns a collectionInfo object with the metadata of the workflows only.		

Examples

URL:

http://serverName:serverPort/rest/service/collections/1?workflows=true

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <collectionInfoList>
     <collectionInfo
      defaultJobJacketTicket="Default Job Ticket"
      defaultJobJacketName="Default Job Jacket"
      defaultJobJacketId="1"
      templateId="0"
      collectionClass="General"
      collectionClassId="1416"
      collectionTemplate="false"
      accessibleChildrenAvailable="true"
      domainId="19"
      collectionPath="Home"
      name="Home"
      id="1">
       <attributeValueList>
         <attributeValue valueId="0" type="numeric" name="Id"
id="1">1</attributeValue>
         <attributeValue valueId="0" type="text" name="Name"
id="2">Home</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Cre-
ated" id="3">2010-04-01T10:00:00Z</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Last
modified" id="4">2013-12-23T12:07:27Z</attributeValue>
         <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
         <attributeValue valueId="1" type="popup" name="Last mod-
ifier" id="6">Admin</attributeValue>
         <attributeValue valueId="0" type="boolean" name="Is Tem-
plate" id="59">false</attributeValue>
         <attributeValue valueId="0" type="text" name="Collection
Path" id="57">Home</attributeValue>
         <attributeValue valueId="1416" type="popup" name="Con-
tent Type" id="62">General</attributeValue>
         <attributeValue valueId="0" type="text" name="Content
Type Hierarchy" id="63">System;Collection;General</attribute-
Value>
       </attributeValueList>
       <workflowInfoList>
         <workflowInfo id="10">
         <name>Workflow1</name>
         <domainId>18</domainId>
         .......
           <statusInfoList>
             <statusInfo id="10">
               <name>Status1</name>
               <domainId>6</domainId>
```

```
                ........
              </statusInfo>
            </statusInfoList>
            <contentTypeInfoList>
              <contentTypeInfo id="1100">
                ......
              </contentTypeInfo>
              <contentTypeInfo id="1101">
                ......
              </contentTypeInfo>
            </contentTypeInfoList>
          </workflowInfo>
        </workflowInfoList>
      </collectionInfo>
    </collectionInfoList>
```

Function	Get a collection's workflow for a specific content type.		
Description	This request returns all of the workflows associated with a specific content type.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`workflows=[true|false]`	boolean	Defaults to false. (Mandatory)
	`contenttypes=[content type ID] or [content type name]`	long	The default content type IDs are documented in the JavaDoc SDK at `com.quark.qpp.core.content.service.constants.DefaultContentTypes`. To get a list of content types, use a call to `../admin/contenttypes`.(Optional) If not provided, the metadata of all workflows will be returned.
Returns	Returns a `collectionInfo` object with the metadata of the workflows associated with the specified content type.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/collections/1?workflows=true&contenttype=Picture`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<collectionInfoList>`
	`<collectionInfo`
	`defaultJobJacketTicket="Default Job Ticket"`
	`defaultJobJacketName="Default Job Jacket"`
	`defaultJobJacketId="1"`
	`templateId="0"`
	`collectionClass="General"`
	`collectionClassId="1416"`		

```
      collectionTemplate="false"
      accessibleChildrenAvailable="true"
      domainId="19"
      collectionPath="Home"
      name="Home"
      id="1">
       <attributeValueList>
         <attributeValue valueId="0" type="numeric" name="Id"
id="1">1</attributeValue>
         <attributeValue valueId="0" type="text" name="Name"
id="2">Home</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Cre-
ated" id="3">2010-04-01T10:00:00Z</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Last
modified" id="4">2013-12-23T12:07:27Z</attributeValue>
         <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
         <attributeValue valueId="1" type="popup" name="Last mod-
ifier" id="6">Admin</attributeValue>
         <attributeValue valueId="0" type="boolean" name="Is Tem-
plate" id="59">false</attributeValue>
         <attributeValue valueId="0" type="text" name="Collection
Path" id="57">Home</attributeValue>
         <attributeValue valueId="1416" type="popup" name="Con-
tent Type" id="62">General</attributeValue>
         <attributeValue valueId="0" type="text" name="Content
Type hierarchy" id="63">System;Collection;General</attribute-
Value>
       </attributeValueList>
       <workflowInfoList>
         <workflowInfo id="10">
           <name>Workflow1</name>
           <domainId>18</domainId>
           .......
           <statusInfoList>
             <statusInfo id="10">
               <name>Status1</name>
               <domainId>6</domainId>
               ........
             </statusInfo>
           </statusInfoList>
           <contentTypeInfoList>
             <contentTypeInfo id="1103">
               ......
             </contentTypeInfo>
           </contentTypeInfoList>
         </workflowInfo>
       </workflowInfoList>
     </collectionInfo>
   </collectionInfoList>
```


Function	Get a collection's users.	
Description	This request returns a list of the users of a collection.	
Type	GET	
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`	
Java API	**Method Name**	**Class Name**
	`getCollection`	`com.quark.qpp.service.facade Class CollectionFacade`

Parameters	**Name**	**Type**	**Description**
	`users=[true|false]`	boolean	Defaults to **false**. (Mandatory)
Returns	Returns a `collectionInfo` object with the metadata of users only.		

Examples

URL:

`http://serverName:serverPort/rest/service/collections/1?users=true`

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <collectionInfoList>
     <collectionInfo
      defaultJobJacketTicket="Default Job Ticket"
      defaultJobJacketName="Default Job Jacket"
      defaultJobJacketId="1"
      templateId="0"
      collectionClass="General"
      collectionClassId="1416"
      collectionTemplate="false"
      accessibleChildrenAvailable="true"
      domainId="19"
      collectionPath="Home"
      name="Home"
      id="1">
       <attributeValueList>
         <attributeValue valueId="0" type="numeric" name="Id"
id="1">1</attributeValue>
         <attributeValue valueId="0" type="text" name="Name"
id="2">Home</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Cre-
ated" id="3">2010-04-01T10:00:00Z</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Last
modified" id="4">2013-12-23T12:07:27Z</attributeValue>
         <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
         <attributeValue valueId="1" type="popup" name="Last mod-
ifier" id="6">Admin</attributeValue>
         <attributeValue valueId="0" type="boolean" name="Is Tem-
plate" id="59">false</attributeValue>
         <attributeValue valueId="0" type="text" name="Collection
Path" id="57">Home</attributeValue>
         <attributeValue valueId="1416" type="popup" name="Con-
tent Type" id="62">General</attributeValue>
         <attributeValue valueId="0" type="text" name="Content
Type Hierarchy" id="63">System;Collection;General</attribute-
Value>
       </attributeValueList>
       <userInfoList>
         <userInfo id="1">
           <name>Admin</name>
           <firstName>Administrator</firstName>
           <nativeTrustee>trie</nativeTrustee>
           <enabled>true</enabled>
           <defaultUserClassid>1</defaultUserClassid>
           <IdapProfileId>0</firstIdapProfileIdtName>
           <deleted>false</deleted>
           <ldapDistinguishedName>Dist Name</ldapDistinguished-
Name>
         </userInfo>
           .......
       </userInfoList>
     </collectionInfo>
```

	`</collectionInfoList>`

Function	Get a collection's groups.		
Description	This request returns a list of the groups associated with a collection.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`groups=[true|false]`	boolean	Defaults to **false**. (Mandatory)
Returns	Returns a `collectionInfo` object with the metadata of groups only.		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/1?groups=true` **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <collectionInfoList>
     <collectionInfo
      defaultJobJacketTicket="Default Job Ticket"
      defaultJobJacketName="Default Job Jacket"
      defaultJobJacketId="1"
      templateId="0"
      collectionClass="General"
      collectionClassId="1416"
      collectionTemplate="false"
      accessibleChildrenAvailable="true"
      domainId="19"
      collectionPath="Home"
      name="Home"
      id="1">
       <attributeValueList>
         <attributeValue valueId="0" type="numeric" name="Id"
id="1">1</attributeValue>
         <attributeValue valueId="0" type="text" name="Name"
id="2">Home</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Cre-
ated" id="3">2010-04-01T10:00:00Z</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Last
modified" id="4">2013-12-23T12:07:27Z</attributeValue>
         <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
         <attributeValue valueId="1" type="popup" name="Last mod-
ifier" id="6">Admin</attributeValue>
         <attributeValue valueId="0" type="boolean" name="Is Tem-
plate" id="59">false</attributeValue>
         <attributeValue valueId="0" type="text" name="Collection
Path" id="57">Home</attributeValue>
         <attributeValue valueId="1416" type="popup" name="Con-
tent Type" id="62">General</attributeValue>
         <attributeValue valueId="0" type="text" name="Content
Type Hierarchy" id="63">System;Collection;General</attribute-
Value>
```

```
        </attributeValueList>
        <groupInfoList>
          <groupInfo id="12">
            <name>Group1</name>
            <emailAddress>Editors@company.com</emailAddress>
            <nativeTrustee>false</nativeTrustee>
            <defaultUserClassId>2</defaultUserClassId>
            <defaultUserClass>NoPrivileges</defaultUserClass>
            <ldapProfileId>10</ldapProfileId>
            <ldapProfile>Profile Name</ldapProfile>
            <ldapDistinguishedName>Dist Name</ldapDistinguished-
Name>
            <memberList>
            <nameId id="1">
            <name>Admin</name>
          </nameId>
            ........
          </memberList>
        </userInfoList>
     </collectionInfo>
    </collectionInfoList>
```

Function	Get the routings for a collection's workflow.		
Description	This request returns all of the routings for a collection's workflow.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`routings=[true|false]`	boolean	Defaults to **false**. (Mandatory)
	`workflow=[workflow name]`	long or string	Specify the workflow **name** or **ID**. (Mandatory)
Returns	Returns a `collectionInfo` object with the metadata of the routings of the specified workflow.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/collections/1?routings=true&workflow=Workflow1`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <collectionInfoList>`
	`  <collectionInfo`
	`  defaultJobJacketTicket="Default Job Ticket"`
	`  defaultJobJacketName="Default Job Jacket"`
	`  defaultJobJacketId="1"`
	`  templateId="0"`
	`  collectionClass="General"`
	`  collectionClassId="1416"`
	`  collectionTemplate="false"`
	`  accessibleChildrenAvailable="true"`
	`  domainId="19"`
	`  collectionPath="Home"`		

```
      name="Home"
      id="1">
       <attributeValueList>
         <attributeValue valueId="0" type="numeric" name="Id"
id="1">1</attributeValue>
         <attributeValue valueId="0" type="text" name="Name"
id="2">Home</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Cre-
ated" id="3">2010-04-01T10:00:00Z</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Last
modified" id="4">2013-12-23T12:07:27Z</attributeValue>
         <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
         <attributeValue valueId="1" type="popup" name="Last mod-
ifier" id="6">Admin</attributeValue>
         <attributeValue valueId="0" type="boolean" name="Is Tem-
plate" id="59">false</attributeValue>
         <attributeValue valueId="0" type="text" name="Collection
Path" id="57">Home</attributeValue>
         <attributeValue valueId="1416" type="popup" name="Con-
tent Type" id="62">General</attributeValue>
         <attributeValue valueId="0" type="text" name="Content
Type Hierarchy" id="63">System;Collection;General</attribute-
Value>
       </attributeValueList>
       <routingInfoList>
         <routingInfo>
           <trusteeId>1</trusteeId>
           <trustee>Admin</trustee>
           <routingTypeId>1</routingTypeId>
           <routingType>User</routingType>
           <workflowId>10</workflowId>
           <workflow>Workflow1</workflow>
           <statusId>10</statusId>
           <status>Status1</status>
         </routingInfo>
           .....
       </routingInfoList>
     </collectionInfo>
   </collectionInfoList>
```


Function	Get a collection's revision controls.		
Description	This request returns all of the revision controls for a collection.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`revisioncontrols=[true|false]`	boolean	Defaults to **false**. (Mandatory)
Returns	Returns a `collectionInfo` object with metadata of the revision controls settings.		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/1?revisioncontrols=true`		

Result:


```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <collectionInfoList>
     <collectionInfo
      defaultJobJacketTicket="Default Job Ticket"
      defaultJobJacketName="Default Job Jacket"
      defaultJobJacketId="1"
      templateId="0"
      collectionClass="General"
      collectionClassId="1416"
      collectionTemplate="false"
      accessibleChildrenAvailable="true"
      domainId="19"
      collectionPath="Home"
      name="Home"
      id="1">
       <attributeValueList>
         <attributeValue valueId="0" type="numeric" name="Id"
id="1">1</attributeValue>
         <attributeValue valueId="0" type="text" name="Name"
id="2">Home</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Cre-
ated" id="3">2010-04-01T10:00:00Z</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Last
modified" id="4">2013-12-23T12:07:27Z</attributeValue>
         <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
         <attributeValue valueId="1" type="popup" name="Last mod-
ifier" id="6">Admin</attributeValue>
         <attributeValue valueId="0" type="boolean" name="Is Tem-
plate" id="59">false</attributeValue>
         <attributeValue valueId="0" type="text" name="Collection
Path" id="57">Home</attributeValue>
         <attributeValue valueId="1416" type="popup" name="Con-
tent Type" id="62">General</attributeValue>
         <attributeValue valueId="0" type="text" name="Content
Type Hierarchy" id="63">System;Collection;General</attribute-
Value>
       </attributeValueList>
       <revisionControlInfoList>
         <revisionControlInfo contentType="Asset" content-
TypeId="1100">
           <minorVersioningAllowed>true</minorVersioningAllowed>
           <allRevisions>true</allRevisions>
           <initialRevisionsCount>0</initialRevisionsCount>
         <recentRevisionsByPeriod>false</recentRevisionsByPeriod>
           <recentRevisionsCount>0</recentRevisionsCount>
           <recentRevisionsDuration>0</recentRevisionsDuration>
         </revisionControlInfo>
           .....
       </revisionControlInfoList>
     </collectionInfo>
   </collectionInfoList>
```


Function	Get a collection's Job Jackets files.	
Description	This request returns all of the job jackets for a collection.	
Type	GET	
URL Format	.../collections/[collection ID] or collectionsbypath/[collection path]	
Java API	**Method Name**	**Class Name**
	getCollection	com.quark.qpp.service.facade

			Class CollectionFacade
Parameters	**Name**	**Type**	**Description**
	jobjackets=[true|false]	boolean	Defaults to **false**. (Mandatory)
Returns	Returns a collectionInfo object with the metadata of associated job jackets.		
Examples	**URL:** http://serverName:serverPort/rest/service/collections/1?jobjackets=true **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <collectionInfoList>
    <collectionInfo
     defaultJobJacketTicket="Default Job Ticket"
     defaultJobJacketName="Default Job Jacket"
     defaultJobJacketId="1"
     templateId="0"
     collectionClass="General"
     collectionClassId="1416"
     collectionTemplate="false"
     accessibleChildrenAvailable="true"
     domainId="19"
     collectionPath="Home"
     name="Home"
     id="1">
      <attributeValueList>
        <attributeValue valueId="0" type="numeric" name="Id"
id="1">1</attributeValue>
        <attributeValue valueId="0" type="text" name="Name"
id="2">Home</attributeValue>
        <attributeValue valueId="0" type="dateTime" name="Cre-
ated" id="3">2010-04-01T10:00:00Z</attributeValue>
       <attributeValue valueId="0" type="dateTime" name="Last
modified" id="4">2013-12-23T12:07:27Z</attributeValue>
        <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
        <attributeValue valueId="1" type="popup" name="Last modi-
fier" id="6">Admin</attributeValue>
        <attributeValue valueId="0" type="boolean" name="Is Tem-
plate" id="59">false</attributeValue>
        <attributeValue valueId="0" type="text" name="Collection
Path" id="57">Home</attributeValue>
        <attributeValue valueId="1416" type="popup" name="Content
Type" id="62">General</attributeValue>
        <attributeValue valueId="0" type="text" name="Content
Type Hierarchy" id="63">System;Collection;General</attribute-
Value>
      </attributeValueList>
      <jobJacketInfoList>
        <jobJacketInfo id="1">
          <name>Default Job Jacket</name>
          <domainId>0</domainId>
          <defaultTicketTemplate>Default Job Ticket</defaultTick-
etTemplate>
          ......
          <ticketTemplateList>
            <ticketTemplate>Default Job Ticket</ticketTemplate>
          </ticketTemplateList>
          <content>
             ......
```

```
            </content>
          </jobJacketInfo>
        </jobJacketInfoList>
      </collectionInfo>
    </collectionInfoList>
```

Function	Get a collection's child collections.		
Description	This request returns immediate child collections accessible to the logged on user.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`childcollections =[true|false]`	boolean	Defaults to **false**. (Mandatory)
Returns	Returns a `collectionInfo` object with the list of child collections that is accessible to the logged-in user.		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/1?childcollections=true` **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <collectionInfoList>
     <collectionInfo
      defaultJobJacketTicket="Default Job Ticket"
      defaultJobJacketName="Default Job Jacket"
      defaultJobJacketId="1"
      templateId="0"
      collectionClass="General"
      collectionClassId="1416"
      collectionTemplate="false"
      accessibleChildrenAvailable="true"
      domainId="19"
      collectionPath="Home"
      name="Home"
      id="1">
       <attributeValueList>
         <attributeValue valueId="0" type="numeric" name="Id"
id="1"></attributeValue>
         <attributeValue valueId="0" type="text" name="Name"
id="2">Home</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Cre-
ated" id="3">2010-04-01T10:00:00Z</attributeValue>
         <attributeValue valueId="0" type="dateTime" name="Last
modified" id="4">2013-12-23T12:07:27Z</attributeValue>
         <attributeValue valueId="1" type="popup" name="Creator"
id="5">Admin</attributeValue>
         <attributeValue valueId="1" type="popup" name="Last mod-
ifier" id="6">Admin</attributeValue>
         <attributeValue valueId="0" type="boolean" name="Is Tem-
plate" id="59">false</attributeValue>
```

```
          <attributeValue valueId="0" type="text" name="Collection
Path" id="57">Home</attributeValue>
          <attributeValue valueId="1416" type="popup" name="Con-
tent Type" id="62">General</attributeValue>
          <attributeValue valueId="0" type="text" name="Content
Type Hierarchy" id="63">System;Collection;General</attribute-
Value>
        </attributeValueList>
        <collectionInfoList>
          <collectionInfo
           defaultJobJacketTicket="Default Job Ticket"
           defaultJobJacketName="Default Job Jacket"
           defaultJobJacketId="1"
           templateId="0"
           collectionClass="General"
           collectionClassId="1416"
           collectionTemplate="false"
           accessibleChildrenAvailable="true"
           domainId="19"
           collectionPath="Home/Collection"
           name="Collection"
           id="10">
            <attributeValueList>
              <attributeValue valueId="0" type="numeric" name="Id"
id="1">10</attributeValue>
              .....
            </attributeValueList>
          </collectionInfo>
          <collectionInfo
           templateId="0"
           collectionClass="General"
           collectionClassId="1416"
           collectionTemplate="false"
           accessibleChildrenAvailable="false"
           domainId="19"
           collectionPath="Home/Collection_84546"
           name="Collection_84546"
           id="13">
            <attributeValueList>
              <attributeValue valueId="0" type="numeric" name="Id"
id="1">13</attributeValue>
              ......
            </attributeValueList>
          </collectionInfo>
        </collectionInfoList>
      </collectionInfo>
    </collectionInfoList>
```


Function	Get a collection's parent collection.		
Description	This request returns all of the accessible parent collections for a collection.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`parentcollections`	boolean	Defaults to **false**. (Mandatory)

	=[true|false]		
Returns	Returns a `collectionInfo` object of ancestor collections up the hierarchy based on access control. While going up the hierarchy, if there exists any inaccessible collection (for which the logged on user does not have access to), the collection path array will contain `collectionInfo` objects of only those collections which are lying below that collection.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/collections/10?parentcollections=true`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<collectionInfoList>`
	`<collectionInfo`
	`defaultJobJacketTicket="Default Job Ticket"`
	`defaultJobJacketName="Default Job Jacket"`
	`defaultJobJacketId="1"`
	`templateId="0"`
	`collectionClass="General"`
	`collectionClassId="1416"`
	`collectionTemplate="false"`
	`accessibleChildrenAvailable="true"`
	`domainId="19"`
	`collectionPath="Home"`
	`name="Home" id="1">`
	`<attributeValueList>`
	`<attributeValue valueId="0" type="numeric" name="Id" id="1">1</attributeValue>`
	`<attributeValue valueId="0" type="text" name="Name" id="2">Home</attributeValue>`
	`<attributeValue valueId="0" type="dateTime" name="Created" id="3">2010-04-01T10:00:00Z</attributeValue>`
	`<attributeValue valueId="0" type="dateTime" name="Last modified" id="4">2013-12-23T12:07:27Z</attributeValue>`
	`<attributeValue valueId="1" type="popup" name="Creator" id="5">Admin</attributeValue>`
	`<attributeValue valueId="1" type="popup" name="Last modifier" id="6">Admin</attributeValue>`
	`<attributeValue valueId="0" type="boolean" name="Is Template" id="59">false</attributeValue>`
	`<attributeValue valueId="0" type="text" name="Collection Path" id="57">Home</attributeValue>`
	`<attributeValue valueId="1416" type="popup" name="Content Type" id="62">General</attributeValue>`
	`<attributeValue valueId="0" type="text" name="Content Type Hierarchy" id="63">System;Collection;General</attributeValue>`
	`</attributeValueList>`
	`<parentColInfoList>`
	`<collectionInfo`
	`defaultJobJacketTicket="Default Job Ticket"`
	`defaultJobJacketName="Default Job Jacket"`
	`defaultJobJacketId="1"`
	`templateId="0"`
	`collectionClass="General"`
	`collectionClassId="1416"`
	`collectionTemplate="false"`
	`accessibleChildrenAvailable="true"`
	`domainId="0"`
	`collectionPath="Home"`		

```
            name="Home"
            id="1">
             <attributeValueList>
               <attributeValue valueId="0" type="numeric" name="Id"
id="1">1</attributeValue>
               <attributeValue valueId="0" type="text" name="Name"
id="2">Home</attributeValue>
                .....
             </attributeValueList>
           </collectionInfo>
         </parentColInfoList>
       </collectionInfo>
     </collectionInfoList>
```

Function	Get collection assets and specify asset display attributes.		
Description	This request returns collection assets by considering asset display attributes.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`assets=[true]`	boolean	(Mandatory)
	`assetattributes`	long or string	An array of display attributes IDs or names to be used while querying for assets within a collection.
Returns	Returns a `collectionInfoList` object containing a list of collections assets.		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/1?assets=true&assetattributes=Name,collection` **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <collectionInfoList>
    <collectionInfo
     id="1"
     name="Home"
     collectionPath="Home"
     domainId="19"
     accessibleChildrenAvailable="true"
     collectionTemplate="false"
     collectionClassId="1416"
     collectionClass="General"
     templateId="0"
     defaultJobJacketId="1"
     defaultJobJacketName="Default Job Jacket"
     defaultJobJacketTicket="Default Job Ticket">
      <attributeValueList>
         <attributeValue id="1" name="Id" type="numeric">1</at-
tributeValue>
          ................
      </attributeValueList>
      <assetInfoList>
```

```
        <assetInfo id="10">
          <attributeValueList>
            <attributeValue id="2" name="Name" type="text"> Work-
flows.json.txt </attributeValue>
            <attributeValue id="55" name="Collection"
type="popup" valueId="1" multiValued="false"> Home </attribute-
Value>
          </attributeValueList>
        </assetInfo>
        <assetInfo id="11">
          <attributeValueList>
            <attributeValue id="2" name="Name" type="text"> Work-
flows.json.txt </attributeValue>
            <attributeValue id="55" name="Collection"
type="popup" valueId="1" multiValued="false"> Home </attribute-
Value>
          </attributeValueList>
        </assetInfo>
       </assetInfoList>
    </collectionInfo>
  </collectionInfoList
```

Function	Get all collections in the hierarchy of the given collection along with assets and specific asset display attributes.		
Description	This request returns a complete collection hierarchy of the given collection alon with collection assets with specific asset display attributes.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`hierarchy=[true]`	boolean	(Mandatory)
	`assets=[true]`	boolean	(Mandatory)
	`assetattributes`	long or string	An array of display attributes IDs or names to be used while querying for assets within a collection.
Returns	Returns a `collectionInfoList` object containing all collections in the hierarchy of a given collection along with assets specific metadata as specified in the assetattributes parameter.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/collections/1?hierarchy=true& assets=true&assetattributes=Name,collection`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <collectionInfoList>`
	`  <collectionInfo id="1" name="Home" collectionPath="Home">`
	`  <assetInfoList>`
	`  <assetInfo id="10">`
	`  .....`
	`  </assetInfoList>`		

```
        <collectionInfoList>
          <collectionInfo id="10" name="Demo"            collec-
tionPath="Home/Demo>
              <assetInfoList>
                <assetInfo id="1">
                  <attributeValueList>
                    <attributeValue id="2" name="Name"
type="text">Hydrangeas.PNG </attributeValue>
                    <attributeValue id="55" name="Collection"
type="popup" valueId+'10" multivalued="false">Demo</attribute-
Value>
                  </attributeValueList>
                </assetInfo>
                  <assetInfo id="3">
                  ..................
                </assetInfoList>
              </collectionInfo>
            </collectionInfoList>
          </collectionInfo>
```

getCollectionHierarchy

Function	Get the metadata of each collection in the hierarchy.		
Description	This request returns all of the metadata associated with each collection in the hierarchy		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollectionHierarchy`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters Returns	**Name**	**Type**	**Description**
	`hierarchy=[true|false]`	boolean	Defaults to true. (Mandatory)
	`attributes=[attribute name or ID], [attribute name or ID],...`	string	To get a list of all attribute IDs, use a call to `../admin/attributes`. (Optional)
	`workflows=[true|false]`	boolean	Defaults to **false**. (Optional)
	`users=[true|false]`	boolean	Defaults to **false**. (Optional)
	`groups=[true|false]`	boolean	Defaults to **false**. (Optional)
	`routings=[true|false]`	boolean	Defaults to **false**. (Optional)
	`revisioncontrols=[true|false]`	boolean	Defaults to **false**. (Optional)
	`jobjackets=[true|false]`	boolean	Defaults to **false**. (Optional)
	`contenttypes=[content type ID] or [content`	string	The default content type IDs are documented in the JavaDoc SDK at

	type name]		com.quark.qpp.core.content.service.constants.DefaultContentTypes. To get a list of content types, use a call to ../admin/contenttypes.(Optional) This parameter can be concatenated with above mentioned parameters workflows and revision controls.
	Returns a collectionInfoList object with the metadata of each collection in the hierarchy.		
Examples	**URL:** http://serverName:serverPort/rest/service/collections/1?hierarchy=true **Result:** <?xml version="1.0" encoding="UTF-8"?> <collectionInfoList> <collectionInfo defaultJobJacketTicket="Default Job Ticket" defaultJobJacketName="Default Job Jacket" defaultJobJacketId="1" templateId="0" collectionClass="General" collectionClassId="1416" collectionTemplate="false" accessibleChildrenAvailable="true" domainId="19" collectionPath="Home" name="Home" id="1"> <attributeValueList> <attributeValue valueId="0" type="numeric" name="Id" id="1">1</attributeValue> <attributeValue valueId="0" type="text" name="Name" id="2">Home</attributeValue> ..... </attributeValueList> <collectionInfo defaultJobJacketTicket="Default Job Ticket" defaultJobJacketName="Default Job Jacket" defaultJobJacketId="1" templateId="0" collectionClass="General" collectionClassId="1416" collectionTemplate="false" accessibleChildrenAvailable="true" domainId="19" collectionPath="Home/Collection" name="Collection" id="10"> <attributeValueList> <attributeValue valueId="0" type="numeric" name="Id" id="1">10</attributeValue> <attributeValue valueId="0" type="text" name="Name" id="2">Collection</attributeValue> ..... </attributeValueList> <collectionInfo defaultJobJacketTicket="Default Job Ticket" defaultJobJacketName="Default Job Jacket" defaultJobJacketId="1" templateId="0" collectionClass="Book" collectionClassId="1403" collectionTemplate="false" accessibleChildrenAvailable="false" domainId="19" collectionPath="Home/Collection/123" name="123" id="14"> <attributeValueList> <attributeValue valueId="0" type="numeric" name="Id" id="1">14</attributeValue> <attributeValue valueId="0" type="text" name="Name" id="2">123</attributeValue> . ..... </attributeValueList> </collectionInfo> </collectionInfo> </collectionInfo> </collectionInfoList>		

getCollectionDefaultJobJacket

Function	Get the default Job Jackets file for a collection.		
Description	This request returns the default job jacket set for a collection. Returns null if no job jacket is specified as default for this collection or the content of the job jacket is not found.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getCollectionDefaultJobJacket`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	op=`defaultjobjacket`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	Returns a `jobJacketInfo` object with its metadata.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/collections/1?op=defaultjobjacket`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<jobJacketInfo id="1">`
	`<name>Default Job Jacket</name>`
	`<domainId>0</domainId>`
	`<defaultTicketTemplate>Default Job Ticket</defaultTicketTemplate>`
	`<createdBy>1</createdBy>`
	`<creatorName>Admin</creatorName>`
	`<created>2008-06-01T10:00:00Z</created>`
	`<lastModifiedBy>1</lastModifiedBy>`
	`<lastModifierName>Admin</lastModifierName>`
	`<lastModified>2008-06-01T10:00:00Z</lastModified>`
	`<ticketTemplateList>`
	`<ticketTemplate>Default Job Ticket</ticketTemplate>`
	`</ticketTemplateList>`
	`<content> ... </content>`
	`</jobJacketInfo>`		

getCollectionDefaultTicket

Function	Get the default Job Ticket for a collection.	
Description	This request returns the default Job Ticket for the collection.	
Type	GET	
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`	
Java API	**Method Name**	**Class Name**
	`getCollectionDefaultTicket`	`com.quark.qpp.service.facade`

			Class CollectionFacade
Parameters	Name	Type	Description
	op=defaultticket	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	Returns the name of the default job ticket for a collection.		
Examples	URL: http://serverName:serverPort/rest/service/collections/1?op=defaulticket Result: Default Job Ticket. The name of the default job ticket for a collection.		

moveCollection

Function	Move a collection.		
Description	This request moves a collection or collection template from one level to another.		
Type	POST		
URL Format	.../collections/[collection ID] or collectionsbypath/[collection path]		
Java API	Method Name		Class Name
	moveCollection		com.quark.qpp.service.facade Class CollectionFacade
Parameters	Name	Type	Description
	op=move	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request moves a collection along with its child collections and assets under a new parent collection.
	targetparent=[collection path or ID]	long or string	The collection to which this collection should be moved. (Mandatory)
Returns	Returns a collectionInfo object of the collection or the collection template with updated metadata.		
Examples	URL: http://serverName:serverPort/rest/service/collections/colTemplate? op=move&targetparent=Home/Collection/XDK Result: <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <collectionInfo templateId="0" collectionClass="General" collectionClassId="1416" collectionTemplate="true" accessibleChildrenAvailable="true" domainId="19" name="colTemplate" id="16"> <attributeValueList> <attributeValue type="numeric" name="Id" id="1">1</attributeValue> ... </attributeValueList> <workflowInfoList>		

	`<workflowInfo id="10">`
	`<name>Workflow1</name> ... <statusInfoList>`
	`<statusInfo id="10">`
	`<name>Status1</name> ... </statusInfo>`
	`</statusInfoList>`
	`</workflowInfo> .... ....</workflowInfoList>`
	`</collectionInfo>`

duplicateCollection

Function	Duplicate a collection.		
Description	This request duplicates a given collection or collection template.		
Type	POST		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`duplicateCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=duplicate`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`name=[new collection name]`	string	The name of the new collection. (Mandatory)
	`targetparent=[collection path or ID]`	long or string	The collection where the duplicate collection should be put. (Mandatory)There is a choice between collection **ID**, **Name** and **Path** when providing a value.
	`assets=[true or false]`	boolean	True to also duplicate assets. Defaults to **false**.(Optional) If assets = **true**, then the metadata flags like `duplicateHierarchy`, `duplicateWorkflows`, `duplicateMembers`, etc will be over written as **true**.
	`createminorversion =[true|false]`	boolean	This parameter will only work if `assets` parameter is given. Defaults to **false**.(Optional)
	`duphierarachy =[true|false]`	boolean	Parameter to duplicate collection with or without its child collections. Defaults to **true**.(Optional)
	`dupworkflows =[true|false]`	boolean	Parameter to duplicate collection with or without workflows. Defaults to **true**.(Optional)
	`dupmembers`	boolean	Parameter to duplicate collection with or

	=[true|false]		without users. Defaults to **true**.(Optional)
	duproutings =[true|false]	boolean	Parameter to duplicate collection with or without Status Routing Settings. Defaults to **true**.(Optional)
	dupjobjackets =[true|false]	boolean	Parameter to duplicate collection with or without JobJackets. Defaults to **true**.(Optional)
	duprevisioncontrols =[true|false]	boolean	Parameter to duplicate collection with or without RevisionControl Settings. Defaults to **true**.(Optional)
Returns	Returns the `collectionInfo` object of the duplicated collection.		
Examples	**URL:** http://serverName:serverPort/rest/service/collections/24?op=duplicate&name=dup12_Children&targetparent=Home **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <collectionInfo templateId="0" collectionClass="General" collectionClassId="1416" collectionTemplate="true" accessibleChildrenAvailable="true" domainId="19" name="dup12_Children" id="17"> <attributeValueList> <attributeValue type="numeric" name="Id" id="1">1</attributeValue> ... </attributeValueList> <workflowInfoList> <workflowInfo id="10">\ <name>Workflow1</name> ... <statusInfoList> <statusInfo id="10">\ <name>Status1</name> ... </statusInfo> </statusInfoList> </workflowInfo> .... </workflowInfoList> </collectionInfo>		

updateCollection

Function	Update a collection.		
Description	This request updates the metadata of a given collection or collection template.		
Type	POST		
URL Format	.../collections/[collection ID] or collectionsbypath/[collection path]		
Java API	**Method Name**		**Class Name**
	updateCollection		com.quark.qpp.service.facade Class CollectionFacade
Parameters	**Name**	**Type**	**Description**
	op=udpate	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	attributevalues=<xml/	string	The set of attribute values. See

	`>`		"Request/Response parameter formats."(Mandatory)
	`workflows =[comma-separated list of workflow names or IDs]`	long or string	To update given collection's workflows. To get the IDs of all workflows, use a call to `../admin/workflows`. (Optional)
	`groups=[group name or ID], [group name or ID],...`	GroupInfoList	To update given collection's workflows.To get the IDs of all groups, use a call to `../admin/groups.` (Optional)
	`jobjackets =[comma-separated list of Job Jackets IDs]`	long or string	To update given collection's JobJackets. To get a list of Job Jackets and their IDs for a collection, use a call to `../collections/colID?jobjackets=true`. (Optional)
	`routings=<xml/>`	xml	To update given collection's Status Routing Settings.See "Request/Response parameter formats."(Optional)
	`ticketname=[name of default Job Ticket]`	string	To update given collection's Default JobJacket Ticket Name.To get the name of Job Ticket for a collection, use a call to `../collections/1?op=defaultticket.` (Optional)
	`applytochildren =[true|false]`	boolean	Parameter to override settings on its child collection. Defaults to **false**. (Optional) If this parameter is supplied, it will override the previous settings on its child collections. Used parameters (e.g. Workflows) will not be removed in case the used workflow doesn't exist in the newly supplied parameters.
	`appendtochildren =[true|false]`	boolean	Parameter to merge settings on its child collection. (Optional)Defaults to false. true is equivalent to Apply settings to child Collections: Merge in Platform. Supplied parameter settings will be merged to previous settings of its child collections.
	`users=<xml/>`	xml	To update given collection's Users. See "Request/Response parameter formats."(Optional)
	`revisioncontrols=<xml/>`	xml	To update given collection's Revision Control Settings. See "Request/Response parameter formats."(Optional)
Returns	Returns the `collectionInfo` object of the updated collection.		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/13?`		

	op=update&attributevalues= <attributeValueList><attributeValue name="name">dup_col</attributeValue></attributeValueList> &workflows=10&users= <userInfoList><userInfo id="1"></userInfo></userInfoList> &groups=Group1 **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <collectionInfo templateId="0" collectionClass="General" collectionClassId="1416" collectionTemplate="true" accessibleChildrenAvailable="true" domainId="19" name="dup12_Children" id="17"> <attributeValueList> <attributeValue type="numeric" name="Id" id="1">1</attributeValue> ... </attributeValueList> <workflowInfoList> <workflowInfo id="10">< <name>Workflow1</name> ... <statusInfoList> <statusInfo id="10">< <name>Status1</name> ... </statusInfo> </statusInfoList> </workflowInfo> ....</workflowInfoList> </collectionInfo> <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <collectionInfo templateId="0" collectionClass="General" collectionClassId="1416" collectionTemplate="false" accessibleChildrenAvailable="false" domainId="19" collectionPath="Home/Collection/XDK/dup_col" name="dup_col" id="13">< <attributeValueList> <attributeValue valueId="0" type="numeric" name="Id" id="1">13</attributeValue> <attributeValue valueId="0" type="text" name="Name" id="2">dup_col</attributeValue> . ..... </attributeValueList> <workflowInfoList> <workflowInfo id="10">< <name>Workflow1</name> <domainId>18</domainId> <created>2013-12-23T12:06:33Z</created> <createdBy>1</createdBy> <lastModifiedBy>1</lastModifiedBy> <lastModified>2013-12-23T12:06:33Z</lastModified> </workflowInfo> </workflowInfoList> ..... </collectionInfo>

updateCollections

Function	Update multiple collections.		
Description	This request updates multiple collections in a single request.		
Type	POST		
URL Format	.../collections		
Java API	**Method Name**		**Class Name**
	updateCollections		com.quark.qpp.service.facade Class CollectionFacade
Parameters	**Name**	**Type**	**Description**
	op=update	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)

			This operation will run only for a valid session id.
	`collectioninfolist=<xml/>`		See "Request/Response parameter formats."
Returns	Returns a `collectioninfolist` object in XML format that contains the metadata of all successfully updated collections.		
Examples	**URL:** `https://serverName:serverPort/rest/service/collections?op=update&collectioninfolist=` `<?xml version="1.0" encoding="UTF-8"?>` `<collectionInfoList>` `<collectionInfo collectionClassId="1416" collectionTemplate="false" collectionPath="Home/Home3">` `<attributeValueList>` `<attributeValue valueId="0" type="text" name="Name" id="2">Home3</attributeValue>` `<attributeValue valueId="0" type="dateTime" name="Created" id="3">2010-04-01T10:00:00Z</attributeValue>` `<attributeValue valueId="0" type="dateTime" name="Last modified" id="4">2010-04-01T10:00:00Z</attributeValue>` `<attributeValue valueId="1" type="popup" name="Creator" id="5">Admin</attributeValue>` `<attributeValue valueId="1" type="popup" name="Last modifier" id="6">Admin</attributeValue>` `<attributeValue valueId="0" type="boolean" name="Is Template" id="59">false</attributeValue>` `<attributeValue valueId="0" type="text" name="Collection Path" id="57">Home/Home1</attributeValue>` `<attributeValue valueId="1416" type="popup" name="Content Type" id="62">General</attributeValue>` `<attributeValue valueId="0" type="text" name="Content Type Hierarchy" id="63">System;Collection;General</attributeValue>` `</attributeValueList>` `</collectionInfo>` `</collectionInfoList>` **Result:** `<?xml version="1.0" encoding="UTF-8" standalone="yes"?>` `<collectionInfoList>` `<collectionInfo defaultJobJacketTicket="Default Job Ticket" defaultJobJacketName="Default Job Jacket" defaultJobJacketId="1" templateId="0" collectionClass="General" collectionClassId="1416" collectionTemplate="false" accessibleChildrenAvailable="false" domainId="19" collectionPath="Home/Home3" name="Home3" id="23">` `<attributeValueList>` `<attributeValue valueId="0" type="numeric" name="Id" id="1">23</attributeValue>` `<attributeValue valueId="0" type="text" name="Name" id="2">Home3</attributeValue>` `<attributeValue valueId="0" type="dateTime" name="Created" id="3">2014-04-10T06:09:46Z</attributeValue> ....` `</attributeValueList>` `<workflowInfoList/>` `<routingInfoList/>` `<userInfoList>` `<userInfo id="1">` `<name>Admin</name> ..... </userInfo>` `</userInfoList>` `<groupInfoList/>`		

	`<jobJacketInfoList>`
	`<jobJacketInfo id="1">`
	`<name>Default Job Jacket</name> ..... </jobJacketInfo>`
	`</jobJacketInfoList>`
	`</collectionInfo>`
	`<collectionInfo> ..... </collectionInfo>`
	`</collectionInfoList>`

deleteCollection

Function	Delete a collection.		
Description	This request deletes the specified collection or collection template.		
Type	POST		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`deleteCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request works only for empty collections. This means a collection can only be deleted if the collection and all of its child collection do not have any assets in them.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/142?op=delete`		

deleteCollectionWithAssets

Function	Delete a collection with assets.	
Description	This request deletes the given collection from the system along with its sub-collections and assets. Only users who have access to the given collection are allowed to delete the collection.	
Type	POST	
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`	
Java API	**Method Name**	**Class Name**
	`deleteCollectionWithAssets`	`com.quark.qpp.service.facade`

			`Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=deletewithassets`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/142?op=deletewithassets`		

createJobjackets

Function	Create a Job Jackets file.		
Description	This request creates new job jacket after parsing the content supplied and returns and object of the newly created Job Jacket. Stream the Job Jackets XML into the request body or send it as a file part.		
Type	POST		
URL Format	`.../collections/jobjackets`		
Java API	**Method Name**		**Class Name**
	`createJobjackets`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create:`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	Returns a `jobJacketInfoList` object in XML format.		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/jobjackets?op=create` **Result:** A `jobJacketInfoList` object in XML format.		

getJobjacket

Function	Get a Job Jackets file with a specific ID.	
Description	This request returns the specified job jacket.	
Type	GET	
URL Format	`.../collections/jobjackets/[Job Jackets ID]`	
Java API	**Method Name**	**Class Name**
	`getJobjacket`	`com.quark.qpp.service.facade`

		Class CollectionFacade
Parameters	This request requires no input parameters.	
Returns	Returns a jobJacketInfoList object with its metadata.	
Examples	URL: http://serverName:serverPort/rest/service/collections/jobjackets/1 Result: <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <jobJacketInfoList> <jobJacketInfo id="1"> <name>Default Job Jacket</name> <domainId>0</domainId> <defaultTicketTemplate>Default Job Ticket</defaultTicket-Template> <createdBy>1</createdBy> <creatorName>Admin</creatorName> <created>2008-06-01T10:00:00Z</created> <lastModifiedBy>1</lastModifiedBy> <lastModifierName>Admin</lastModifierName> <lastModified>2008-06-01T10:00:00Z</lastModified> <ticketTemplateList> <ticketTemplate>Default Job Ticket</ticketTemplate> </ticketTemplateList> <content>.... </content> </jobJacketInfo> </jobJacketInfoList>	

getAllJobJackets

Function	Get all Job Jackets.	
Description	This request returns all the metadata of Job Jackets available in the system.	
Type	GET	
URL Format	.../collections/jobjackets	
Java API	**Method Name**	**Class Name**
	getAllJobJackets	com.quark.qpp.service.facade Class CollectionFacade
Parameters	This request requires no input parameters.	
Returns	Returns a jobJacketInfoList object that contains a list of Job Jackets.	
Examples	**URL:** http://serverName:serverPort/rest/service/collections/jobjackets **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <jobJacketInfoList> <jobJacketInfo id="1"> <name>Default Job Jacket</name> <domainId>0</domainId> <defaultTicketTemplate>Default Job Ticket</defaultTicket-Template> <createdBy>1</createdBy>	

	`<creatorName>Admin</creatorName>`
	`<created>2008-06-01T10:00:00Z</created>`
	`<lastModifiedBy>1</lastModifiedBy>`
	`<lastModifierName>Admin</lastModifierName>`
	`<lastModified>2008-06-01T10:00:00Z</lastModified>`
	`<ticketTemplateList>`
	`<ticketTemplate>Default Job Ticket</ticketTemplate>`
	`</ticketTemplateList>`
	`<content>.....</content>`
	`</jobJacketInfo> ..... </jobJacketInfoList>`

getJobJacketContent

Function	Get the contents of a Job Jackets file.		
Description	This request returns the actual content of a specified job jacket in xml format.		
Type	GET		
URL Format	`.../collections/[collection ID] or collectionsbypath/[collection path]`		
Java API	**Method Name**		**Class Name**
	`getJobJacketContent`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=jobjacketcontent`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`jobjacket=[Job Jackets file ID]`		To get a list of Job Jackets and their IDs for a collection, use a call to `../collections/colID?jobjackets=true`. (Mandatory)
Returns	Returns the actual content of a specified job jacket in XML format		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/1?op=jobjacketcontent&jobjacket=1` **Result:** `<?xml version="1.0" encoding="UTF-8"?>` `<JDF xmlns="http://www.CIP4.org/JDFSchema_1_1" xmlns:qxp="http://www.quark.com/QuarkXPress/JDF_1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="1.2" JacketVersion="1.0" ID="QuarkXPressDefault_Job_Jacket" Type="QuarkXPressJacket" xsi:type="qxp:QuarkXPressJacket" Status="Waiting" Name="Default Job Jacket" Template="true">` `<ResourcePool> </ResourcePool>` `<JDF Name="Default Job Ticket" ID="JobTicket_0" IsDefault="1" Status="Waiting" Template="true" Type="QuarkXPressTemplateTicket" xsi:type="qxp:QuarkXPressTemplateTicket">` `<ResourceLinkPool> </ResourceLinkPool>` `<ResourcePool> </ResourcePool>` `</JDF>` `</JDF>`		

updateJobJacket

Function	Update a job jacket.		
Description	This request updates the contents of a specified job jacket.		
Type	POST		
URL Format	`.../collections/jobjackets/[jobJacketId]`		
Java API	**Method Name**		**Class Name**
	`updateJobJacket`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.Pass the xml of the job jacket file as file stream in the request body.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/collections/jobjackets/{jobJacketId}?op=update`		

updateJobJackets

Function	Update multiple job jackets		
Description	This request updates the contents of multiple job jackets in a single request.		
Type	POST		
URL Format	`.../collections/jobjackets`		
Java API	**Method Name**		**Class Name**
	`updateJobJackets`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
	`jobjacketinfolist=<xml/>`		See “Request/Response parameter formats.”
Returns	Returns a `jobjacketinfolist` object in XML format that contains the metadata of all successfully updated job jackets.		
Examples	**URL:** `http://serverName:serverPort/collections/jobjackets?op=update&job`		

	jacketinfolist=<xml />
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<jobJacketInfoList>`
	`  <jobJacketInfo id="1">`
	`  <name>Default Job Jacket</name>`
	`  <domainId>0</domainId>`
	`  <defaultTicketTemplate>Default Job Ticket</defaultTicket-`
	`Template>`
	`  <createdBy>1</createdBy>`
	`  <creatorName>Admin</creatorName>`
	`  <created>2008-06-01T10:00:00Z</created>`
	`  <lastModifiedBy>1</lastModifiedBy>`
	`  <lastModifierName>Admin</lastModifierName>`
	`  <lastModified>2008-06-01T10:00:00Z</lastModified>`
	`  <ticketTemplateList>`
	`  <ticketTemplate>Default Job Ticket</ticketTemplate>`
	`  </ticketTemplateList>`
	`  <content><?xml version="1.0" encoding="UTF-8"?><JDF`
	`xmlns="http://www.CIP4.org/JDFSchema_1_1"`
	`xmlns:qxp="http://www.quark.com/QuarkXPress/JDF_1"`
	`xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Ver-`
	`sion="1.2" JacketVersion="1.0" ID="QuarkXPressDefault_Job_Jacket"`
	`Type="QuarkXPressJacket" xsi:type="qxp:QuarkXPressJacket" Sta-`
	`tus="Waiting" Name="Default Job Jacket" Template="true"> <Resour-`
	`cePool></ResourcePool> <JDF Name="Default Job Ticket"`
	`ID="JobTicket_0" IsDefault="1" Status="Waiting" Template="true"`
	`Type="QuarkXPressTemplateTicket" xsi:type="qxp:QuarkXPressTem-`
	`plateTicket"> <ResourceLinkPool></ResourceLinkPool> <Resource-`
	`Pool></ResourcePool> </JDF></JDF> </content>`
	`  </jobJacketInfo>`
	`</jobJacketInfoList>`

parseJobJacketXml

Function	Parse a Job Jackets file. (Include the file in the request body or as file part.)		
Description	This request parses the jobjacket xml and returns a JobJacket object containing the jobjacket name and ticket names.		
Type	POST		
URL Format	.../collections/jobjackets		
Java API	**Method Name**		**Class Name**
	`parseJobJacketXml`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=parsexml`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	Returns a `jobjacketinfo` object in XML format.		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/jobjackets?op=parsexml`		

	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<jobJacketInfo id="0">`
	`  <name>Project1 Job Jacket</name>`
	`  <domainId>0</domainId>`
	`  <defaultTicketTemplate>Default Job Ticket</defaultTicketTem-`
	`plate>`
	`  <createdBy>0</createdBy>`
	`  <lastModifiedBy>0</lastModifiedBy>`
	`  <ticketTemplateList>`
	`  <ticketTemplate>Default Job Ticket</ticketTemplate>`
	`  </ticketTemplateList>`
	`</jobJacketInfo>`

deleteJobJacket

Function	Delete a Job Jackets file.		
Description	This request deletes the jobjacket with the given id.		
Type	POST		
URL Format	`.../collections/jobjackets/[Job Jacket ID]`		
Java API	**Method Name**		**Class Name**
	`deleteJobJacket`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/collections/jobjackets/10?op=delete`		

getApplicableRoutingUsers

Function	Get Applicable Routing Users		
Description	This request returns a list of applicable routing users for the given collection for specified workflow and status. This will return all collection mapped users having the same userclass as any of the status mapped userclasses.		
Type	GET		
URL Format	`.../collections/{collectionId}`		
Java API	**Method Name**		**Class Name**
	`getApplicableRoutingUsers`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`routingusers =true`	boolean	

	workflow={workflowIdOrName}	string	The ID or Name of the workflow whose Routing Users you wish to be retrieved.
	status={statusIdOrName} or workflowStatusCombination	string	The ID or Name of the status whose Routing Users you wish to be retrieved.
Returns	Returns an userInfoList object containing a list of all collection users applicable for given workflow or status.		
Examples	**URL:** http://serverName:serverPort/rest/service/collections/1?routingusers=true&status=Status1 **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <userInfoList> <userInfo id="1"> <name>Admin</name> <firstName>Administrator</firstName> <nativeTrustee>true</nativeTrustee> <enabled>true</enabled> <defaultUserClassId>1</defaultUserClassId> <ldapProfileId>0</ldapProfileId> <deleted>false</deleted> <ldapDistinguishedName>Dist Name</ldapDistinguishedName> </userInfo> .... </userInfoList>		

getApplicablePrivileges

Function	Get applicable privileges for the logged in user.		
Description	This request returns the privileges applicable for the logged in user for the given collection. These privileges are a union of the privileges of the following user clases:Userclass of the logged in user in the given collection, only if the user is mapped to the given collection.Userclasses of collection mapped groups of which the logged in user is a member.		
Type	GET		
URL Format	.../collections/[collection ID]		
Java API	**Method Name**		**Class Name**
	getApplicablePrivileges		com.quark.qpp.service.facade Class CollectionFacade
Parameters	**Name**	**Type**	**Description**
	applicableprivileges=[true]	boolean	
	status=[statusId] or [Name] or [StatusWorkFlowCombination]	long	The **ID** or **Name** of the status whose privileges you wish to be retrieved.
	contenttype=[contenTypeId] or [Name] or	long	The default content type IDs are documented in the JavaDoc SDK at

	[Path]		com.quark.qpp.core.content.service.constants.DefaultContentTypes. To get a list of content types, use a call to ../admin/contenttypes.(Optional) If not provided, the metadata of all conent types will be returned.
	privileges=[comma-separated list of privilege ids or names]	string	
Returns	Returns a applicablePrivilegesInfo object containing either application and content level privileges or both, depending upong the parameters.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/collections/192?applicableprivileges=true
	Result:
	`<applicablePrivilegesInfo>`
	`<applicationPrivList>`
	`<privilege id="140000023">`
	`<name>Delete</name>`
	`<privilegeHierarchy> Administration;Storage;StorageRules;Delete </privilegeHierarchy>`
	`</privilege>`
	`<privilege id="140000022">`
	`<name>Create/Modify</name>`
	`<privilegeHierarchy> Administration;Storage;StorageRules;Create/Modify </privilegeHierarchy>`
	`</privilege> ..... </applicationPrivList>`
	`<contentTypePrivList>`
	`<contentTypePriv contentTypeId="1100" contentType="Asset" contentTypeHierarchy="System;Asset;">`
	`<privilege id="3000006">`
	`<name>Change Workflow of Checked In Asset</name>`
	`<privilegeHierarchy> Edit Attributes;Change Workflow of Checked In Asset </privilegeHierarchy>`
	`</privilege> ..... </contentTypePriv>`
	`</contentTypePrivList>`
	`</applicablePrivilegesInfo>`		

Function	Get enabled content type privileges.	
Description	This request returns only contenttype based privileges applicable for the logged in user for the given collection. These privileges are a union of the privileges of the following user clases:Userclass of the logged in user in the given collection, only if the user is mapped to the given collection.Userclasses of collection mapped groups of which the logged in user is a member.	
Type	GET	
URL Format	.../collections/[collection ID]	
Java API	**Method Name**	**Class Name**
	getApplicablePrivileges	com.quark.qpp.service.facade Class CollectionFacade

	Name	Type	Description
Parameters	`applicableprivileges=[true]`	boolean	(Mandatory)
	`contenttype=[contenTypeId]` or `[Name]` or `[Path]`	long	The default content type IDs are documented in the JavaDoc SDK at `com.quark.qpp.core.content.service.constants.DefaultContentTypes`. To get a list of content types, use a call to `../admin/contenttypes`.(Optional) If not provided, the metadata of all conent types will be returned.
Returns	Returns a `applicablePrivilegesInfo` object containing either application and content level privileges or both, depending upong the parameters.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/collections/192?applicableprivileges=true&contenttype=Picture`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<applicablePrivilegesInfo>`
	`  <contentTypePrivList>`
	`  <contentTypePriv contentTypeId="1103" contentType="Picture" contentTypeHierarchy="System;Asset;Picture;">`
	`  <privilege id="10000015">`
	`  <name>Get Original Assets</name>`
	`  <privilegeHierarchy>General;Get Original Assets</privilegeHierarchy>`
	`  </privilege>`
	`  <privilege id="10000012">`
	`  <name>Check In Asset without Displaying the Check In Dialog Box</name>`
	`  <privilegeHierarchy> General;Check In Asset without Displaying the Check In Dialog Box </privilegeHierarchy>`
	`  </privilege>.....`
	`  </contentTypePriv>`
	`  </contentTypePrivList>`
	`</applicablePrivilegesInfo>`		

Function	Get all enabled status privileges.	
Description	This request returns the privileges applicable for the logged in user for the given collection and status (Status based overridden privileges) . These privileges are a union of the privileges of the following user clases: Userclass of the logged in user in the given collection, only if the user is mapped to the given collection.Userclasses of collection mapped groups of which the logged in user is a member.	
Type	GET	
URL Format	`.../collections/[collection ID]`	
Java API	**Method Name**	**Class Name**
	`getApplicablePrivileges`	`com.quark.qpp.service.facade Class CollectionFacade`

	Name	Type	Description
Parameters	applicableprivileges=[true]	boolean	
	status=[statusId] or [Name] or [StatusWorkFlowCombination]	boolean	The **ID** or **Name** of the status whose privileges you wish to be retrieved.
	privileges=[comma-separated list of privilege ids or names]	long or string	
Returns	Returns a applicablePrivilegesInfo object containing either application and content level privileges or both, depending upong the parameters.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/collections/192?applicableprivileges=true&status=10&privileges=...,...,...
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<applicablePrivilegesInfo>`
	`<contentTypePrivList>`
	`<contentTypePriv contentTypeId="1103" contentType="Picture" contentTypeHierarchy="System;Asset;Picture;">`
	`<privilege id="10000015">`
	`<name>Get Original Assets</name>`
	`<privilegeHierarchy>General;Get Original Assets</privilegeHierarchy>`
	`</privilege>`
	`<privilege id="10000012">`
	`<name>Check In Asset without Displaying the Check In Dialog Box</name>`
	`<privilegeHierarchy> General;Check In Asset without Displaying the Check In Dialog Box </privilegeHierarchy>`
	`</privilege> .....</contentTypePriv> </contentTypePrivList>`
	`</applicablePrivilegesInfo>`		

getApplicableRoutingGroups

Function	Get applicable routing groups for the logged in user.		
Description	This request returns an array of applicable routings groups for the given collection for the specified workflow and status. This will return all collection mapped groups whose userclass is the same as one of the status mapped userclasses.		
Type	GET		
URL Format	.../collections/[collection ID]		
Java API	Method Name		Class Name
	getApplicableRoutingGroups		com.quark.qpp.service.facade Class CollectionFacade
Parameters	Name	Type	Description

	routinggroups=[true]	boolean	
	workflow=[Id] or [Name]	long or string	The **ID** or **Name** of the workflow whose routing groups you wish to be retrieved.
	status=[statusId] or [Name] or [StatusWorkFlowCombination]	long or string	The **ID** or **Name** of the status whose routing groups you wish to be retrieved.
Returns	Returns a list of applicable routing `GroupInfo` objects for the given collection and status.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/collections/1?routinggroups=true&workflow=Default Workflow&status=Edit
	Result:
	`<groupInfoList>`
	`  <groupInfo id="60">`
	`  <name>Editors</name>`
	`  <emailAddress/>`
	`  <nativeTrustee>true</nativeTrustee>`
	`  <defaultUserClassId>1</defaultUserClassId>`
	`  <defaultUserClass>Administrator</defaultUserClass>`
	`  </groupInfo>`
	`</groupInfoList>`		

getApplicableAttributeConstraints

Function	Get applicable attribute constraints for the logged in user.		
Description	This request returns attribute constraints applicable to the given status and collection for the logged in user. The user may have multiple userclasses, due to its membership in multiple groups which are mapped to the collection with different userclasses. Since attribute constraints are defined per userclass, multiple constraints in such cases are evaluated and the most permissive set of attribute constraints is returned.		
Type	GET		
URL Format	.../collections/[collection ID]		
Java API	**Method Name**		**Class Name**
	getApplicableAttributeConstraints		com.quark.qpp.service.facade Class CollectionFacade
Parameters	**Name**	**Type**	**Description**
	applicableconstraints=[true]	boolean	
	status=[statusId] or [Name] or [StatusWorkFlowCombination]	long or string	The **ID** or **Name** of the status whose attributes constraints you wish to be retrieved.
	attributes=[comma-separated list of	long or string	

	attributes ids or names]		
Returns	Returns a list of applicable AttributeConstraintsInfo objects.		
Examples	**URL:** http://serverName:serverPort/rest/service/collections/191?applicableconstraints=true&status=Default Workflow&attributes=2,...,... **Result:** <attributeConstraintsInfoList> <attributeConstraintsInfo> <attributeId>2</attributeId> <attributeName>Name</attributeName> <constraintId>1</constraintId> <constraintName>PREVENT CHANGE</constraintName> </attributeConstraintsInfo> ..... </attributeConstraintsInfoList>		

getApplicableStatusTransition

Function	Get applicable status transitions for the logged in user.		
Description	This request returns the applicable status transition for the given workflow, status and collection for the logged in user. If the status transition is overridden for the given userclass, then overridden will be considered.		
Type	GET		
URL Format	.../collections/[collection ID]		
Java API	**Method Name**		**Class Name**
	getApplicableStatusTransition		com.quark.qpp.service.facade Class CollectionFacade
Parameters	**Name**	**Type**	**Description**
	statustransitions=[true]	boolean	
	workflow=[workflowId] or [Name]	long or string	The **ID** or **Name** of the workflow whose status transitions you wish to be retrieved.
	status=[statusId] or [Name]	long or string	The **ID** or **Name** of the status whose status transitions you wish to be retrieved.
Returns	Returns a statusTransitionInfoList object containing a list of applicable status transitions.		
Examples	**URL:** http://serverName:serverPort/rest/service/collections/1?statustransitions=true&workflow=Default Workflow **Result:** <statusTransitionInfoList> <statusTransitionInfo id="10"> <name>Edit</name> <workflowInitiatingStatus>true</workflowInitiatingStatus>		

```
        <nextPossibleStatusInfoList>
            <nextPossibleStatusInfo id="11">
                <name>Publish</name>
            </nextPossibleStatusInfo>
        </nextPossibleStatusInfoList>
    </statusTransitionInfo>
    <statusTransitionInfo id="11">
        <name>Publish</name>
        <workflowInitiatingStatus>true</workflowInitiatingStatus>
        <nextPossibleStatusInfoList>
            <nextPossibleStatusInfo id="10">
                <name>Edit</name>
            </nextPossibleStatusInfo>
        </nextPossibleStatusInfoList>
    </statusTransitionInfo>
</statusTransitionInfoList>
```

getApplicableWorkflowInitialStatuses

Function	Get applicable workflow initial statuses.		
Description	This request returns the initial applicable statuses for the given workflow and collection.		
Type	GET		
URL Format	`.../collections/[collection ID]`		
Java API	**Method Name**		**Class Name**
	`getApplicableWorkflowInitialStatuses`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`workflowinitiatingstatuses=[true]`	boolean	
	`workflow=[workflowId] or [Name]`	long or string	The **ID** or **Name** of the workflow whose workflow Initiating Statuses you wish to be retrieved.
Returns	Returns a list of `StatusInfo` objects referring to the initial applicable statuses.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/collections/1?workflowinitiatingstatuses=true&workflow=Default Workflow`
	Result:
	`<statusInfoList>`
	`  <statusInfo id="10">`
	`  <name>Edit</name> ..... </statusInfo>`
	`  <statusInfo id="11">`
	`  <name>Publish</name> ..... </statusInfo>`
	`</statusInfoList>`		

Query management

The query management capabilities of the API allow application developers to manage searches and includes the following REST calls:

- *search*
- *searchAssets*
- *searchCollections*
- *createSearches*
- *getSearch*
- *getAllSearches*
- *getSearchResultsForQuery*
- *getTextSearchResults*
- *setSharedSearchTrustees*
- *getSharedSearchTrustees*
- *updateSearches*
- *deleteSearch*
- *count*
- *countAssets*
- *countCollections*
- *countSearchResultsForQuery*
- *countTextResults*

search

Function	Search assets/collection using a searchDefinition.	
Description	This request searches assets/collections based on a given searchDefinition. This API has been exposd as both Get and POST. In the case of a large payload, POST API can be used.	
Type	GET, POST	
URL Format	`.../search`	
Java API	**Method Name**	**Class Name**

	search		com.quark.qpp.service.facade Class QueryFacade
Parameters	**Name**	**Type**	**Description**
	searchinfo=[xml]	string	An XML object containing metadata of the asset/collections to be found.
Returns	Returns a assetInfolist object containing the metadata of the asset to be found , if the context of the given searchInfo is **ASSET**. Returns a collectionInfolist object containing the metadata of the asset to be found , if the context of the given searchInfo is **COLLECTION**.		
Examples	**URL**: http://serverName:serverPort/rest/service/search?search=<xml/>&op=count **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <asset-InfoList> <assetInfo id="66"> <attributeValueList> <attributeValue valueId="0" type="text"name="Name" id="2">Alberi.jpg</attributeValue> </attributeValueList> </assetInfo> <assetInfo id="72"> <attributeValueList> <attributeValue valueId="0" type="text"name="Name"id="2">QuarkImage.jpg</attributeValue> </attributeValueList> </assetInfo> </assetInfoList>		

Function	Search assets/collection using pagination from searchDefinition.		
Description	This request searches assets/collections based on a given searchDefinition in a paginated manner. This APi has been exposed as both GET and POST.		
Type	GET, POST		
URL Format	.../search/paginated		
Java API	**Method Name**		**Class Name**
	search		com.quark.qpp.service.facade Class QueryFacade
Parameters	**Name**	**Type**	**Description**
	searchinfo=[xml]	string	An XML object containing metadata of the asset/collections to be found.
	numberOfRows=[int]	int	
	offset=[int]	int	
Returns	Returns a assetInfolist object containing the metadata of the number of assets		

	defined through the `numberofrows` parameter starting from the value defined in the `offset` parameter.

Examples	**URL**:
	`http://serverName:serverPort/rest/service/search/paginated?search=<xml/>&offset=0&numberofows=100`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfoList>`
	`<assetInfo id="1004">`
	`<attributeValueList>`
	`<attributeValue value="XML2c4.xml>`
	`</attributeValueList>`
	`</assetInfo>`
	`</assetInfoList>`

searchAssets

Function	Search assets for a particular attribute value.		
Description	This request searchs for assets matching specfic search criteria.		
Type	GET		
URL Format	`.../search/assets`		
	Method Name		**Class Name**
	`searchAssets`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`[attribute name or id]=[attribute value]`	long or string	The attribute name and value that you are searching for. (Mandatory) The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version • Article Component Reference Asset Minor version • Article Id • Asset Checksum • Attached Component Id

			• Attached Layout Uid • Attachment Checksum • Auxillary Data • Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page NameRelationship Status • Status • XPath Functions to get assets based on two non-queryable atributes (**Collection**, **Status**) are given below. You can pass more than one attribute as a paramater with the value to search for.
	`includechildcontenttypes = [true|false]`	boolean	Parameter to include child content types in search criteria.
	`attributes =[comma-seperated list of attribute names or IDs]`		You can pass the appropriate value for any queryable attribute. For a list of attributes, use a call to `..admin/attributes`.
	`displaymode=[x]`	long	The default display modes are documented in the JavaDoc SDK at `com.quark.qpp.core.query.service.constants.DefaultDisplayModes`. (Optional) Defaults to **1**.
	`exploremode=[x]`	long	For a list of explore modes, see `ExploreModeTypes.java`. (Optional) Defaults to **0**.
	`collections`	string	Comma separated ids of collections in which assets are to be searched.
	`recursive=[true|false]`	boolean	Indicates whether all of the hierarchy below and including collections are to be

			searched. Deffault value=**true**
Returns	Returns search results based upon specified attribute values.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/assets?Name=1quark2.jpg&Collection=Home`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfoList>`
	`  <assetInfo id="4">`
	`  <attributeValueList>`
	`  <attributeValue type="popup" name="Workflow" id="54">Workflow1</attributeValue>`
	`  <attributeValue type="popup" name="Collection" id="55">Home</attributeValue>`
	`  <attributeValue type="popup" name="Routed to" id="25">Admin</attributeValue>`
	`  <attributeValue type="popup" name="Status" id="24">Status1</attributeValue>`
	`  <attributeValue type="popup" name="Content Type" id="62">Picture</attributeValue>`
	`  </attributeValueList>`
	`  </assetInfo>`
	`</assetInfoList>`		

Function	Search for assets that contain a particular text string.		
Description	This request will search for assets that contain a given text string. This is specific to content based searches in workspace and ConnectClient.		
Type	GET		
URL Format	`.../search/assets`		
Java API	**Method Name**		**Class Name**
	`searchAssets`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`text=[text to search for]`	string	Specify the string to search for. (Mandatory)
	`attributes=[comma-seperated list of attribute names or IDs]`	string	You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`. The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id

			• Article Component Reference Asset Major Version • Article Component Reference Asset Minor version • Article Id • Asset Checksum • Attached Component Id • Attached Layout Uid • Attachment Checksum • Auxillary Data • Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page Name • Relationship Status • Status • XPath
Returns	Returns an `assetInfoList` object containing the assets that contain the specified text string.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/assets?text=box&attributes=Name`
	Result:
	Example of a text search with specific attribute:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfoList>`
	`  <assetInfo id="292">`
	`  <attributeValueList>`
	`  <attributeValue valueId="0" type="text" name="Name" id="2">15116462.qxp</attributeValue>`
	`  </attributeValueList>`
	`  </assetInfo>`		

	`<assetInfo id="304">`
	`<attributeValueList>`
	`<attributeValue valueId="0" type="text" name="Name" id="2">15116c9a.qxp</attributeValue>`
	`</attributeValueList>`
	`</assetInfo>`
	`<assetInfo id="306">`
	`<attributeValueList>`
	`<attributeValue valueId="0" type="text" name="Name" id="2">15116d33.qxp</attributeValue>`
	`</attributeValueList>`
	`</assetInfo>`
	`</assetInfoList>`

Function	Search for a particular attachment.		
Description	This request searches for all attachments for a given document.		
Type	GET		
URL Format	`.../search/assets`		
Java API	**Method Name**		**Class Name**
	`searchAssets`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`document=[ID of the project]`	long	ID of the document whose attachments you want found. (Mandatory)
	`attributes=[comma-seperated list of attribute names or IDs]`	long or string	You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`. The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version • Article Component Reference Asset Minor version • Article Id • Asset Checksum • Attached Component Id • Attached Layout Uid • Attachment Checksum

			• Auxillary Data • Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page Name • Relationship Status • Status • XPath
Returns	Returns an `assetInfoList` object containing the assets that contain child assets.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/assets?document=22`
	Result:
	A `QueryResultInfo` object in XML format.
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfoList>`
	`  <assetInfo id="290">`
	`  <attributeValueList>`
	`  <attributeValue valueId="0" type="text" name="Name" id="2">153486e3.qcd</attributeValue>`
	`  </attributeValueList>`
	`  </assetInfo>`
	`</assetInfoList>`		

Function	Search within an existing query.		
Description	This request searchs an existiing query in the platform.		
Type	GET		
URL Format	.../`search`/assets		
Java API	**Method Name**		**Class Name**
	`searchAssets`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`query=[ID of the`	long or	Use /search to get IDs. (Mandatory)

	`query]`	string	
	`attributes=[comma-seperated list of attribute names or IDs]`	long or string	You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`. The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version • Article Component Reference Asset Minor Version • Article Id • Asset Checksum • Attached Component Id • Attached Layout Uid • Attachment Checksum • Auxillary Data • Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page Name • Relationship Status • Status • XPath

Returns	Returns a `QueryResultInfo` object in XML format.
Examples	**URL:** `http://serverName:serverPort/rest/service/search/assets?query=12` **Result:** A `QueryResultInfo` object in XML format.

Function	Search for assets with a particular status.		
Description	This request returns all assets associated with given status.		
Type	GET		
URL Format	`.../search/assets`		
Java API	**Method Name**		**Class Name**
	`searchAssets`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`status=[status name]` and `workflow=[workflow name]`	string	Specify Status of a workflow. Specify Name or ID of a workflow. (Mandatory) Status and workflow will go hand in hand.
	`attributes=[comma-seperated list of attribute names or IDs]`	long or string	You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`. The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version • Article Component Reference Asset Minor Version • Article Id • Asset Checksum • Attached Component Id • Attached Layout Uid • Attachment ChecksumAuxillary Data

			• Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page Name • Relationship Status • Status • XPath
Returns	Returns an `assetInfoList` object containing an array of assets related to a given status.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/assets?workflow=Workflow1 &status=Status1&attributes=name,5,7,62,54,collection`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfoList>`
	`  <assetInfo id="4">`
	`  <attributeValueList>`
	`  <attributeValue type="text" name="Name" id="2">1quark2.jpg</attributeValue> ... </attributeValueList>`
	`  </assetInfo>`
	`  <assetInfo id="10">`
	`  <attributeValueList>`
	`  <attributeValue type="text" name="Name" id="2">Tulips</attributeValue> ... </attributeValueList>`
	`  </assetInfo>`
	`</assetInfoList>`		

Function	Search for assets in a particular collection.		
Description	This request searchs for the assets checked-in at given collection.		
Type	GET		
URL Format	`.../search/assets`		
Java API	**Method Name**		**Class Name**
	`searchAssets`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**

	`collections =[comma-separated list of collection names or IDs]`	long or string	To get a list of collection IDs for a particular collection, use a call to `..collectionsbypath/[path]`.(Mandatory)
	`recursive=[true|false]`	boolean	Parameter to include child collections. Defaults to **true**. (Optional)
	`attributes=[comma-seperated list of attribute names or IDs]`	long or string	You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`. The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version • Article Component Reference Asset Minor Version • Article Id • Asset Checksum • Attached Component Id • Attached Layout Uid • Attachment ChecksumAuxillary Data • Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index

			• Page Name • Relationship Status • Status • XPath
Returns	Returns an `assetInfoList` object containing an array of assets checked-in at a specified collection.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/assets?collections=Home <?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	Result:
	`<assetInfoList>`
	`  <assetInfo id="4">`
	`  <attributeValueList>`
	`  <attributeValue type="text" name="Name" id="2">1quark2.jpg</attributeValue> ... </attributeValueList>`
	`  </assetInfo>`
	`  <assetInfo id="10">`
	`  <attributeValueList>`
	`  <attributeValue type="text" name="Name" id="2">Tulips</attributeValue> ... </attributeValueList>`
	`  </assetInfo>`
	`</assetInfoList>`		

searchCollections

Function	Search for collections in a particular collection.		
Description	This request searchs all nested collections accessible to the user under a given collection ID or Name.		
Type	GET		
URL Format	`.../search/collections`		
Java API	**Method Name**		**Class Name**
	`searchCollections`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`collections =[comma-separated list of collection names or IDs]`	long or string	Parameter to include child collections. To get a list of collection IDs for a particular collection, use a call to `..collectionsbypath/[path]`.(Mandatory)
	`recursive=[true|false]`	boolean	Defaults to **true**.(Optional)
	`includechildcontenttypes=[true | false]`	boolean	Parameter to include child content types in search criteria.
	`attributes=[comma-seperated list of`	long or string	Specify the string to search for. (Mandatory)

	`attribute names or IDs]`		You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`. The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version • Article Component Reference Asset Minor Version • Article Id • Asset Checksum • Attached Component Id • Attached Layout Uid • Attachment ChecksumAuxillary Data • Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page Name • Relationship Status • Status • XPath
Returns	Returns a `collectionInfoList` object containing an array of existing and accessible collections.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/collections?collection path=Home/Collection&Creator=Admin`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<collectionInfoList>`
	`<collectionInfo templateId="0" collectionClass="General" collectionClassId="1416" collectionTemplate="false" accessibleChildrenAvailable="true" domainId="19" name="Collection" id="10">`
	`<attributeValueList>`
	`<attributeValue type="text" name="Name" id="2">Collection</attributeValue>`
	`<attributeValue type="popup" name="Workflow" id="54"/>`
	`<attributeValue type="popup" name="Collection" id="55"/>`
	`<attributeValue type="popup" name="Routed to" id="25"/>`
	`<attributeValue type="popup" name="Status" id="24"/>`
	`<attributeValue type="popup" name="Content Type" id="62">General</attributeValue>`
	`</attributeValueList>`
	`</collectionInfo>`
	`</collectionInfoList>`

Function	Search for collections in a collection with particular attributes.		
Description	This request searchs for assets in a specified collection and displasy results with the specified attributes.		
Type	GET		
URL Format	`.../search/collections`		
Java API	**Method Name**		**Class Name**
	`searchCollections`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`attributes =[comma-separated list of attribute names or IDs]`	long or string	For a list of attributes, use a call to `..admin/attributes`.(Mandatory)
	`collecions =[comma-separated list of collections names or IDs]`	long or string	Parameter to include child collections. To get a list of collection IDs for a particular collection, use a call to `..collectionsbypath/[path]`. (Mandatory)
	`recursive=[true|false]`	boolean	Defaults to **true**. (Optional)
Returns	Returns a `collectionInfoList` object containing an array of existing and accessible collections.		
Examples	**URL:** `http://serverName:serverPort/rest/service/search/`		

	`collections?collection path=Home&Creator=Admin&attributes=Name`
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<collectionInfoList>`
	`  <collectionInfo templateId="0" collectionClass="General" collectionClassId="1416" collectionTemplate="false" accessibleChildrenAvailable="true" domainId="19" name="Home" id="1">`
	`  <attributeValueList>`
	`  <attributeValue valueId="0" type="text" name="Name" id="2">Home </attributeValue>`
	`  </attributeValueList>`
	`  <groupInfoList/>`
	`  </collectionInfo>`
	`</collectionInfoList>`

createSearches

Function	Create searches.		
Description	This request creates a single or multiple searches at one time.		
Type	POST		
URL Format	`.../searchdefinitions`		
Java API	**Method Name**		**Class Name**
	`createSearches`		`com.quark.qpp.service.facade Class QueryDefinitionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`searchinfolist=[xml]`	string	An XML object containing metadata of searches to be created.
Returns	Returns a `searchInfoList` object containing metadata of newly created searches.		

Examples	**URL:**
	
	`http://serverName:serverPort/rest/service/searchdefinitions?op=create&searchinfilist=<xml/>`
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8"?>`
	`<searchInfoList>`
	`  <searchInfo parameterized="false" id="110">`
	`  <searchName>AllPictures</searchName>`
	`  <owner id="1">Admin</owner>`
	`  <queryContext context="ASSET"/>`
	`  <queryConditionList>`
	`  <queryCondition recursive="false" parameterized="false" negated="false" type="CONTENT_TYPE_CONDITION" comparisonOperator="IS" logicalOperator="OR" nestingLevel="1">`
	`  <searchValue id="1103">System;Asset;Picture;</searchValue>`
	`  </queryCondition>`
	`  </queryConditionList>`		

```
        <queryDisplay exploreMode="PLAIN" displayMode="SNIPPET">
            <displayColumnList>
                <displayColumn attributeColumn="true" width="150"
columnId="2">Name</displayColumn>
                <displayColumn attributeColumn="false"
width="150" columnId="29"/>
                <displayColumn attributeColumn="false"
width="150" columnId="62"/>
                <displayColumn attributeColumn="false"
width="150" columnId="54"/>
                <displayColumn attributeColumn="false"
width="150" columnId="25"/>
                <displayColumn attributeColumn="false"
width="150" columnId="55"/>
            </displayColumnList>
            <sortInfoList>
                <sortInfo attributeColumn="true" order="ascend-
ing" columnId="2">Name</sortInfo>
            </sortInfoList>
        </queryDisplay>
    </searchInfo>
</searchInfoList>
```

getSearch

Function	Search for a saved search.		
Description	This request returns the metadata of a specified search.		
Type	GET		
URL Format	`.../searchdefinitions/{searchIdOrName}?`		
Java API	**Method Name**		**Class Name**
	`getSearch`		`com.quark.qpp.service.facade Class QueryDefinitionFacade`
Parameters	**Name**	**Type**	**Description**
	`context=[true|false]`	boolean	Defaults to **false**.
	`display=[true|false]`	boolean	Defaults to **false**.
	`conditions=[true|false]`	boolean	Defaults to **false**.
	`getSharedSearchTrustees=[true|false]`	boolean	Defaults to **false**.
Returns	Returns a `searchInfoList` object containing metadata of all saved and shared searches. Metadata of QueryContext, QueryDisplay, QueryConditions will be added per the parameters.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/searchdefinitions/Default`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<searchInfoList>`
	`  <searchInfo parameterized="false" id="100">`
	`  <searchName>Default</searchName>`		

	`<owner id="1">Admin</owner>`
	`</searchInfo>`
	`</searchInfoList>`

getAllSearches

Function	Get all saved searches.		
Description	This request returns all saved, shared searches.		
Type	GET		
URL Format	`.../searchdefinitions?`		
Java API	**Method Name**		**Class Name**
	`getAllSearches`		`com.quark.qpp.service.facade QueryDefinitionFacade`
Parameters	**Name**	**Type**	**Description**
	`context=[true|false]`	boolean	Defaults to **false**.
	`display=[true|false]`	boolean	Defaults to **false**.
	`conditions=[true|false]`	boolean	Defaults to **false**.
	`getSharedSearchTrustees=[true|false]`	boolean	Defaults to **false**.
Returns	Returns a `searchInfoList` object containing metadata of all saved and shared searches. Metadata of QueryContext, QueryDisplay, QueryConditions will be added per the parameters.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/searchdefinitions`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<searchInfoList>`
	`<searchInfo parameterized="false" id="100">`
	`<searchName>Default</searchName>`
	`<owner id="1">Admin</owner>`
	`</searchInfo>`
	`<searchInfo parameterized="false" id="101">`
	`<searchName>Expired Content</searchName>`
	`<owner id="1">Admin</owner>`
	`</searchInfo>`
	`</searchInfoList>`		

getSearchResultsForQuery

Function	Get the results of an existing search.
Description	This request returns the results of the execution of an already saved search.
Type	GET

URL Format	`.../search/[search name or ID]`		
Java API	**Method Name**		**Class Name**
	`getSearchResultsForQuery`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`displaymode=[x].`	long	Defaults to 1. The default display modes are documented in the `JavaDoc SDK at com.quark.qpp.core.query.service.constants.DefaultDisplayModes`.(Optional)
	`exploremode=[x]`	long	Defaults to 0. For a list of explore modes, see `ExploreModeTypes.java`.
Returns	Returns an `assetInfoList` object containing the results of a specified search.		
Examples	**URL:** `http://serverName:serverPort/rest/service/search/Default` **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assetInfoList>
    <assetInfo id="10">
        <attributeValueList>
            <attributeValue type="text" name="Name"
id="2">Tulips</attributeValue>
            <attributeValue type="popup" name="Collection"
id="55">Home</attributeValue>
            <attributeValue type="popup" name="Workflow"
id="54">Workflow1</attributeValue>
            <attributeValue type="popup" name="Status"
id="24">Status1</attributeValue>
        </attributeValueList>
    </assetInfo>
    <assetInfo id="11">
        <attributeValueList>
            <attributeValue type="text" name="Name" id="2">Dupli-
cate_asset</attributeValue>
            <attributeValue type="popup" name="Collection"
id="55">Home</attributeValue>
            <attributeValue type="popup" name="Workflow"
id="54">Workflow1</attributeValue>
            <attributeValue type="popup" name="Status"
id="24">Status1</attributeValue>
        </attributeValueList>
    </assetInfo>
    <assetInfo id="4">
        <attributeValueList>
            <attributeValue type="text" name="Name"
id="2">1quark2.jpg</attributeValue>
            <attributeValue type="popup" name="Collection"
id="55">Home</attributeValue>
            <attributeValue type="popup" name="Workflow"
id="54">Workflow1</attributeValue>
            <attributeValue type="popup" name="Status"
id="24">Status1</attributeValue>
        </attributeValueList>
    </assetInfo>
    <assetInfo id="5">
        <attributeValueList>
```

```
            <attributeValue type="text" name="Name"
id="2">15116.qxp</attributeValue>
            <attributeValue type="popup" name="Collection"
id="55">Home</attributeValue>
            <attributeValue type="popup" name="Workflow"
id="54">Workflow1</attributeValue>
            <attributeValue type="popup" name="Status"
id="24">Status1</attributeValue>
        </attributeValueList>
    </assetInfo>
</assetInfoList>
```

Function	Get the results of an existing search with specific display attributes.		
Description	This request returns the results of the execution of an already saved search and displays only the specified attributes.		
Type	GET		
URL Format	`.../search/[search name or ID]`		
Java API	**Method Name**		**Class Name**
	`getSearchResultsForQuery`		com.quark.qpp.service.facade Class QueryFacade
Parameters	**Name**	**Type**	**Description**
	`attributes =[comma-separated list of attribute names or IDs]`	long or string	For a list of attributes, use a call to `..admin/attributes`.(Mandatory)
Returns	Returns an `assetInfoList` object containing the results of a specified search with only the specified attributes.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/Default?attributes=Name,2,10,Collection`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfoList>`
	`  <assetInfo id="10">`
	`  <attributeValueList>`
	`  <attributeValue type="text" name="Name" id="2">Tulips</attributeValue>`
	`  <attributeValue type="text" name="Name" id="2">Tulips</attributeValue>`
	`  <attributeValue type="dateTime" name="Job jacket last modified" id="10"/>`
	`  <attributeValue type="popup" name="Collection" id="55">Home</attributeValue>`
	`  </attributeValueList>`
	`  </assetInfo>`
	`</assetInfoList>`		

Function	Get the results of an already saved “Ask” queries.
Description	This request returns the result of an already saved search (Ask Queries) by accepting required values of search criteria as its parameters.
Type	GET

URL Format	`.../search/[search name OR id]`		
Java API	**Method Name**		**Class Name**
	`getSearchResultsForQuery`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`text=[text to search for]`	string	Provide value for search criteria as its parameter. Parameter will be based upon the saved query. (Mandatory)
	`attributes=[comma-seperated list of attribute names or IDs]`	string or long	You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`.
Returns	Returns an `assetInfoList` containing the results of a specified search based on the given parameter and its value.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/myAskQuery?File%20extension=qxp`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <assetInfoList>`
	`  <assetInfo id="1">`
	`  <attributeValueList>`
	`  <attributeValue valueId="0"`
	`  type="text" name="Name" id="2">Sample.qxp</attributeValue>`
	`  </attributeValueList>`
	`  </assetInfo>`
	`  </assetInfoList>`		

Function	Search assets using pagination from a saved query.		
Description	This request returns the results of the execution of an already saved search in a paginated manner.		
Type	GET		
URL Format	`.../search/paginated/[search name OR id]`		
Java API	**Method Name**		**Class Name**
	`getSearchResultsForQuery`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`numberOfRows=[int]`	int	. (Mandatory)
	`offset=[int]`	int	.
Returns	Returns an `assetInfoList` containing the results of a specified search based on the given parameter and its value.		
Examples	**URL:**		

	http://serverName:serverPort/rest/service/search/paginated/{QueryIdorName}?offset=0&numberofrows=100
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <assetInfoList>`
	`  <assetInfo id="1004">`
	`  <attributeValueList>`
	`  <attributeValue value="XML2c4.xml>`
	`  </attributeValueList>`
	`  </assetInfo>`
	`  </assetInfoList>`

getTextSearchResults

Function	Search for assets using wild cards.		
Description	This request executes a quick search for assets based on their name. Both simple and wildcard based comparison is performed for name.		
Type	GET		
URL Format	.../search/quicksearch/		
Java API	**Method Name**		**Class Name**
	getTextSearchResults		com.quark.qpp.service.facade Class QueryFacade
Parameters	**Name**	**Type**	**Description**
	collections =[comma-separated list of collection names or IDs]	long or string	To get a list of collection IDs for a particular collection, use a call to ..collectionsbypath/[path].(Mandatory)
	recursive=[true|false]	boolean	Parameter to include child collections. Defaults to **true**. (Optional)
	attributes=[comma-seperated list of attribute names or IDs]	long or string	You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to ..admin/attributes. The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version

			• Article Component Reference Asset Minor Version • Article Id • Asset Checksum • Attached Component Id • Attached Layout Uid • Attachment ChecksumAuxillary Data • Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page Name • Relationship Status • Status • XPath
	`includechildcontenttypes=[true | false]`	boolean	Parameter to include child content types in search criteria.
	`sortby`	long or string	attributeIds used for determining the sequence of asset elements in query results.
Returns	Returns an `assetInfoList` object .		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/quicksearch?text=*5*>`
	Result:
	`<assetInfoList>`
	`<assetInfo id="1">`
	`<attributeValueList>`
	`<attributeValue valueId="0" type="text" name="Name" id="2">151161dc.qxp</attributeValue>`
	`<attributeValue valueId="10" type="popup" name="Workflow" id="54">Workflow1</attributeValue>`		

```
            <attributeValue valueId="1" type="popup" name="Col-
lection" id="55">Home</attributeValue>
            <attributeValue valueId="1" type="popup" name="Routed
to" id="25">Admin</attributeValue>
            <attributeValue valueId="10" type="popup" name="Sta-
tus" id="24">Status1</attributeValue>
            <attributeValue valueId="11021" type="popup"
name="Content Type" id="62">QuarkXPress Project</attributeValue>
        </attributeValueList>
    </assetInfo>
</assetInfoList>
```


setSharedSearchTrustees

Function	Share a search.		
Description	This request shares any specified search to any user or group.		
Type	POST		
URL Format	`.../searchdefinitions/{searchIdOrName}?`		
Java API	**Method Name**		**Class Name**
	`getSharedSearchTrustees`		`com.quark.qpp.service.facade Class QueryDefinitionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=setsharedsearchtrustees`	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`trusteeinfolist=[xml]`	string	An XML object containing metadata of all trustees (users, groups) that the search will be shared with.
Returns	Returns a `trusteeinfolist` object containing metadata of all trustees (users, groups) that the search will be shared with.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/searchdefinitions?op=setsharedsearchtrustees&trusteeinfolist=<xml/>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<trusteeInfoList>`
	`  <trusteeInfo id="10">`
	`  <user>true</user>`
	`  <deleted>false</deleted>`
	`  <name>Admin2</name>`
	`  <emailAddress>Editors@company.com</emailAddress>`
	`  <nativeTrustee>false</nativeTrustee>`
	`  <defaultUserClassId>2</defaultUserClassId>`
	`  <ldapProfileId>10</ldapProfileId>`
	`  <ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`  </trusteeInfo>`
	`  <trusteeInfo id="12">`
	`  <user>false</user>`
	`  <deleted>false</deleted>`
	`  <name>Group1</name>`
	`  <emailAddress>Editors@company.com</emailAddress>`		

	`<nativeTrustee>false</nativeTrustee>`
	`<defaultUserClassId>2</defaultUserClassId>`
	`<ldapProfileId>10</ldapProfileId>`
	`<ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`</trusteeInfo>`
	`</trusteeInfoList>`

getSharedSearchTrustees

Function	Get the trustees of a shared search.		
Description	This request returns the trustees that a specfied search has been shared with.		
Type	GET		
URL Format	`.../searchdefinitions/{searchIdOrName}?`		
Java API	**Method Name**		**Class Name**
	`getSharedSearchTrustees`		`com.quark.qpp.service.facade Class QueryDefinitionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=getsharedsearchtrustees`	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	Returns a `trusteeInfoList` object containing metadata of all trustees (users, groups) that a specified search has been shared with.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/searchdefinitions?op=getsharedsearchtrustees`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<trusteeInfoList>`
	`<trusteeInfo id="10">`
	`<user>true</user>`
	`<deleted>false</deleted>`
	`<name>Admin2</name>`
	`<emailAddress>Editors@company.com</emailAddress>`
	`<nativeTrustee>false</nativeTrustee>`
	`<defaultUserClassId>2</defaultUserClassId>`
	`<ldapProfileId>10</ldapProfileId>`
	`<ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`</trusteeInfo>`
	`<trusteeInfo id="12">`
	`<user>false</user>`
	`<deleted>false</deleted>`
	`<name>Group1</name>`
	`<emailAddress>Editors@company.com</emailAddress>`
	`<nativeTrustee>false</nativeTrustee>`
	`<defaultUserClassId>2</defaultUserClassId>`
	`<ldapProfileId>10</ldapProfileId>`
	`<ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`</trusteeInfo>`
	`</trusteeInfoList>`		

updateSearches

Function	Update searches.		
Description	This request updates a single or multiple searches at one time.		
Type	POST		
URL Format	`.../searchdefinitions`		
Java API	`Method Name`		`Class Name`
	`updateSearch`		`com.quark.qpp.service.facade Class QueryDefinitionFacade`
Parameters	`Name`	`Type`	`Description`
	`op=update`	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`searchinfolist=[xml]`	string	An XML object containing metadata of searches to be updated.
Returns	Returns a `searchinfolist` object containing metadata of successfully updated searches.		
Examples	**URL:** `http://serverName:serverPort/rest/service/searchdefinitions?op=update&searchinfilist=<xml/>` **Result:**		

```
<?xml version="1.0" encoding="UTF-8"?>
<searchInfoList>
    <searchInfo parameterized="false" id="110">
        <searchName>AllPictures</searchName>
        <owner id="1">Admin</owner>
        <queryContext context="ASSET"/>
        <queryConditionList>
            <queryCondition recursive="false"
parameterized="false" negated="false" type="CONTENT_TYPE_CONDI-
TION" comparisonOperator="IS" logicalOperator="OR" nestin-
gLevel="1">
                <searchValue
id="1103">System;Asset;Picture;</searchValue>
            </queryCondition>
        </queryConditionList>
        <queryDisplay exploreMode="PLAIN" displayMode="SNIPPET">
            <displayColumnList>
                <displayColumn attributeColumn="true" width="150"
columnId="2">Name</displayColumn>
                <displayColumn attributeColumn="false"
width="150" columnId="29"/>
                <displayColumn attributeColumn="false"
width="150" columnId="62"/>
                <displayColumn attributeColumn="false"
width="150" columnId="54"/>
                <displayColumn attributeColumn="false"
width="150" columnId="25"/>
                <displayColumn attributeColumn="false"
width="150" columnId="55"/>
            </displayColumnList>
            <sortInfoList>
                <sortInfo attributeColumn="true" order="ascend-
ing" columnId="2">Name</sortInfo>
```

	`</sortInfoList>`
	`</queryDisplay>`
	`</searchInfo>`
	`</searchInfoList>`

deleteSearch

Function	Delete searches.		
Description	This request deletes a specified search.		
Type	POST		
URL Format	`.../searchdefinitions/{searchIdOrName}?`		
Java API	**Method Name**		**Class Name**
	`updateSearch`		`com.quark.qpp.service.facade Class QueryDefinitionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/searchdefinitions/{SavedSearchIdORName}?op=delete`		

count

Function	Count assets/collection using a searchDefinition.		
Description	This request counts assets/collections based on a given searchDefinition. This API has been exposd as both Get and POST. In the case of a large payload, POST API can be used.		
Type	GET, POST		
URL Format	`.../search`		
Java API	**Method Name**		**Class Name**
	`count`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`searchinfo=[xml]`	string	An XML object containing metadata of the asset/collections to be counted.
Returns	Returns a count of the assets, if the context of the given searchInfo is **ASSET**. Returns a count of the collections, if the context of the given searchInfo is **COLLECTION**		
Examples	**URL:** `http://serverName:serverPort/rest/service/search?search=<xml/>&op`		

	`=count`
	Result:
	`30`

countAssets

Function	Count assets.		
Description	This request counts assets based on given conditions.		
Type	GET		
URL Format	`.../search`		
Java API	**Method Name**		**Class Name**
	`countAssets`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`op=count`	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`assets =[comma-separated list of assets names or IDs]`	long or string	(Mandatory)
	`recursive=[true|false]`	boolean	Defaults to **true**.(Optional)
	`includechildcontenttypes=[true | false]`	boolean	Parameter to include child content types in search criteria.
	`attributes=[comma-seperated list of attribute names or IDs]`	long or string	Specify the string to search for. (Mandatory) You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`. The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version • Article Component Reference Asset Minor Version

			• Article Id • Asset Checksum • Attached Component Id • Attached Layout Uid • Attachment ChecksumAuxillary Data • Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page Name • Relationship Status • Status • XPath
Returns	Returns a count of the assets satisfying the condition parameters specified.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/assets?op=count`
	Result:
	`25`		

countCollections

Function	Count collections.		
Description	This request counts collections based on given conditions.		
Type	GET		
URL Format	`.../search`		
Java API	**Method Name**		**Class Name**
	`countCollections`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**

	`op=count`	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`collections =[comma-separated list of collection names or IDs]`	long or string	Parameter to include child collections. To get a list of collection IDs for a particular collection, use a call to `..collectionsbypath/[path]`.(Mandatory)
	`recursive=[true|false]`	boolean	Defaults to **true**.(Optional)
	`includechildcontenttypes=[true | false]`	boolean	Parameter to include child content types in search criteria.
	`attributes=[comma-seperated list of attribute names or IDs]`	long or string	Specify the string to search for. (Mandatory) You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`. The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version • Article Component Reference Asset Minor Version • Article Id • Asset Checksum • Attached Component Id • Attached Layout Uid • Attachment ChecksumAuxillary Data • Box Height • Box Uid • Box Width • Collection • Component position

			• Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page Name • Relationship Status • Status • XPath
Returns	Returns a count of the collections satisfying the condition parameters specified.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/collections?op=count`
	Result:
	`25`		

countSearchResultsForQuery

Function	Count assets based on saved query.		
Description	This request counts resultant assets of saved query.		
Type	GET		
URL Format	`.../search/[Query ID or Name]`		
Java API	**Method Name**		**Class Name**
	`countSearchRestultsForQuery`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`op=count`	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`attributes=[comma-seperated list of attribute names or IDs]`	long or string	Specify the string to search for. (Mandatory) You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`. The attributes must be queryable. The following is a lit of non-queryable

			attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version • Article Component Reference Asset Minor Version • Article Id • Asset Checksum • Attached Component Id • Attached Layout Uid • Attachment ChecksumAuxillary Data • Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page Name • Relationship Status • Status • XPath
Returns	Returns a count of the assets satisfying the specified query.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/AllAssets?op=count`
	Result:
	`2`		

countTextSearchResults

Function	Count text search results.		
Description	This request counts assets based on given conditions.		
Type	GET		
URL Format	`.../search/quicksearch`		
Java API	**Method Name**		**Class Name**
	`countTextSearchRestults`		`com.quark.qpp.service.facade Class QueryFacade`
Parameters	**Name**	**Type**	**Description**
	`op=count`	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`collections =[comma-separated list of collections names or IDs]`	long or string	Parameter to include child collections. To get a list of collection IDs for a particular collection, use a call to `..collectionsbypath/[path]`.(Mandatory)
	`recursive=[true|false]`	boolean	Defaults to **true**.(Optional)
	`includechildcontenttypes=[true | false]`	boolean	Parameter to include child content types in search criteria.
	`attributes=[comma-seperated list of attribute names or IDs]`	long or string	Specify the string to search for. (Mandatory) You can pass the appropriate value for any queryable attribute to fetch specific attributes. For a list of attributes, use a call to `..admin/attributes`. The attributes must be queryable. The following is a lit of non-queryable attributes: • Article Component id • Article Component Reference Asset Id • Article Component Reference Asset Major Version • Article Component Reference Asset Minor Version • Article Id • Asset Checksum • Attached Component Id

			• Attached Layout Uid • Attachment ChecksumAuxillary Data • Box Height • Box Uid • Box Width • Collection • Component position • Dependent On Collection Resources • InDesign Object UID • Item Name • Item Type • Page Index • Page Name • Relationship Status • Status • XPath
Returns	Returns a count of the assets satisfying the condition parameters specified.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/search/quicksearch?text=*quark*&op=count`
	Result:
	`2`		

Administration

The administration capabilities of the API allow application developers to perform administrative tasks and includes the following REST calls:

- *createUsers*
- *getUser*
- *getAllUsers*
- *getUsersWithGivenUserClass*
- *getAllUserRedliningColor*
- *updateUserRedliningColor*
- *updateUserEnabled*
- *updateUsers*
- *deleteUser*
- *createGroups*
- *getGroup*
- *getAllGroups*
- *getGroupsWithGivenUserClass*
- *getuserGroups*
- *updateGroups*
- *deleteGroup*
- *createContentTypes*
- *getContentType*
- *getAllContentTpes*
- *updateContentTypes*
- *deleteContentType*
- *createAttributes*
- *setAttributes*
- *getAttribute*
- *getAllAttributes*
- *updateAttributes*

- *deleteAttribute*
- *createForms*
- *getAllForms*
- *updateForms*
- *deleteForm*
- *createRoles*
- *getRole*
- *getAllRoles*
- *updateRoles*
- *deleteRole*
- *createRelationType*
- *getRelationType*
- *getAllRelationTypes*
- *updateRelationTypes*
- *deleteRelationType*
- *createDomains*
- *getDomain*
- *getAllDomains*
- *updateDomains*
- *deleteDomain*
- *createLdapProfiles*
- *getLdapProfile*
- *getAllLdapProfiles*
- *updateLdapProfiles*
- *deleteLdapProfile*
- *getLdapUsers*
- *importUsersFromLdap*
- *getLdapGroups*
- *importGroupsFromLdap*
- *synchronizeLdapUsersGroups*
- *synchronizeLdapUsersGroupsBasedOnProfile*
- *createLdapSearches*
- *getLdapSearch*
- *getAllLdapSearches*

- *updateLdapSearches*
- *deleteLdapSearch*
- *getTrustee*
- *updateTrusteeRule*
- *createWorkflows*
- *getWorkflow*
- *getAllWorkflows*
- *updateWorkflows*
- *deleteWorkFlow*
- *createPreferences*
- *getPreference*
- *getAllPreferences*
- *updatePreferences*
- *getPreferenceValues*
- *updatePreferenceValues*
- *deletePreference*
- *createRepositories*
- *getRepository*
- *getAllRepositories*
- *updateRepositories*
- *deleteRepository*
- *createPrivileges*
- *getPrivilege*
- *getAllPrivileges*
- *getApplicableApplicationPrivileges*
- *getApplicableContentPrivileges*
- *updatePrivileges*
- *deletePrivilege*
- *createPrivilegeGroups*
- *getPrivilegeGroup*
- *getAllPrivilegeGroups*
- *updatePrivilegeGroups*
- *deletePrivilegeGroup*
- *addDomainValue*

- *getDomainValue*
- *getDomainValueXML*
- *setDomainValueXML*
- *searchDomainValues*
- *updateDomainValue*
- *deleteDomainValue*
- *getFavorites*
- *setFavorites*
- *getWorkflowInitiatingStatuses*
- *getStatusTransitions*

createUsers

Function	Create users.		
Description	This request is used to create users. The new user will be created without a password, client needs to set the password for the user separately.		
Type	POST		
URL Format	`.../admin/users`		
Java API	**Method Name**		**Class Name**
	`createUsers`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`userinfolist`		Object containing a list of the users (userInfo) to be created. See the userInfoList paramater in “Request/Response parameter formats”. The following are mandatory attributes required to create a user: • Name • DefaultUserClass • Enabled,NativeTrustee • DefaultUserClassId
Returns	Returns a `userInfoList` object containing metadata of a newly created user.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/users?op=create&userinfolist=`		

```
<?xml version="1.0" encoding="UTF-8"?>
<userInfoList>
    <userInfo>
        <name>SampleQPPGroup</name>
        <nativeTrustee>true</nativeTrustee>
        <enabled>true</enabled>
        <defaultUserClassId>1</defaultUserClassId>
        <defaultUserClass>Administrator</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
</userInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<userInfoList>
    <userInfo id="14">
        <name>SampleQPPGroup</name>
        <nativeTrustee>true</nativeTrustee>
        <enabled>true</enabled>
        <defaultUserClassId>1</defaultUserClassId>
        <defaultUserClass>Administrator</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
</userInfoList>
Example with multiple
usersURL:http://serverName:serverPort/rest/service/admin/users?op
=create&userinfolist= <?xml version="1.0" encoding="UTF-8"?>
<userInfoList>
    <userInfo>
        <name>SampleQPPGroup1</name>
        <nativeTrustee>true</nativeTrustee>
        <enabled>true</enabled>
        <defaultUserClassId>1</defaultUserClassId>
        <defaultUserClass>Administrator</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
    <userInfo>
        <name>SampleQPPGroup2</name>
        <nativeTrustee>true</nativeTrustee>
        <enabled>true</enabled>
        <defaultUserClassId>1</defaultUserClassId>
        <defaultUserClass>Administrator</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
</userInfoList>
Result: <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<userInfoList>
    <userInfo id="14">
        <name>SampleQPPGroup1</name>
        <nativeTrustee>true</nativeTrustee>
        <enabled>true</enabled>
        <defaultUserClassId>1</defaultUserClassId>
        <defaultUserClass>Administrator</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
```

	`<userInfo id="15">`
	`<name>SampleQPPGroup2</name>`
	`<nativeTrustee>true</nativeTrustee>`
	`<enabled>true</enabled>`
	`<defaultUserClassId>1</defaultUserClassId>`
	`<defaultUserClass>Administrator</defaultUserClass>`
	`<ldapProfileId>0</ldapProfileId>`
	`<deleted>false</deleted>`
	`<ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`</userInfo>`
	`</userInfoList>`

getUser

Function	Get specific user.	
Description	This request is used to fet the metadata of a specified user.	
Type	GET	
URL Format	`.../admin/users/[userIdORuserName]`	
Java API	**Method Name**	**Class Name**
	`getUser`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters	
Returns	Returns an `userInfoList` object with its metadata.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/users/Admin`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<userInfoList>`
	`<userInfo id="1">`
	`<name>Admin</name>`
	`<firstName>Administrator</firstName>`
	`<nativeTrustee>true</nativeTrustee>`
	`<enabled>true</enabled>`
	`<defaultUserClassId>1</defaultUserClassId>`
	`<defaultUserClass>Administrator</defaultUserClass>`
	`<ldapProfileId>0</ldapProfileId>`
	`<deleted>false</deleted>`
	`<ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`</userInfo>`
	`</userInfoList>`	

getAllUsers

Function	Get all users.	
Description	This request is used to fetch all users.	
Type	GET	
URL Format	`.../admin/users`	
Java API	**Method Name**	**Class Name**

	getAllUsers	com.quark.qpp.service.facade Class AdminFacade
Parameters	This request requires no input parameters	
Returns	Returns an userInfoList object containing metadata of users in the form of an array.	

Examples	**URL:**
	http://serverName:serverPort/rest/service/users
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<userInfoList>`
	`  <userInfo id="1">`
	`  <name>Admin</name>`
	`  <domainId>0</domainId>`
	`  <firstName>Administrator</firstName>`
	`  <phoneNumber>+911722299000</phoneNumber>`
	`  <emailAddress>platformadmin@quark.com</emailAddress>`
	`  <nativeTrustee>true</nativeTrustee>`
	`  <enabled>true</enabled>`
	`  <defaultUserClassId>1</defaultUserClassId>`
	`  <defaultUserClass>Administrator</defaultUserClass>`
	`  <ldapProfileId>0</ldapProfileId>`
	`  <user>true</user>`
	`  <deleted>false</deleted>`
	`  <ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`  <overriddenUserClassId>0</overriddenUserClassId>`
	`  </userInfo>`
	`</userInfoList>`	

getUsersWithGivenUserClass

Function	Get users with the default given userclass.		
Description	This request returns the platform users with the specified user class.		
Type	GET		
URL Format	../admin/roles/{roleIdORroleName}		
Java API	**Method Name**		**Class Name**
	getUsersWithGivenUserClass		com.quark.qpp.service.facade Class AdminFacade
Parameters	**Name**	**Type**	**Description**
	users	string	true
Returns	Returns a userInfoList containing metadata of all the users belonging to the specified userClass in array format.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/admin/roles/Administrator?users=true
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<userInfoList>`
	`  <userInfo id="1">`		

```
        <name>Admin</name>
        <domainId>0</domainId>
        <firstName>Administrator</firstName>
        <emailAddress>qppadministrator@quark.com</emailAddress>
        <nativeTrustee>true</nativeTrustee>
        <enabled>true</enabled>
        <defaultUserClassId>1</defaultUserClassId>
        <defaultUserClass>Administrator</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <user>true</user>
        <deleted>false</deleted>
        <overriddenUserClassId>0</overriddenUserClassId>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
    <userInfo id="10">
        <name>Admin2</name> ...... </userInfo>
</userInfoList>
```

getAllUserRedliningColor

Function	Get a list of all users along with redline colors.	
Description	This request returns a list of all the users along with their redlining colors.	
Type	GET	
URL Format	`.../redline`	
Java API	**Method Name**	**Class Name**
	`getAllUserRedliningColor`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns list of all the users along with their redlining colors.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/redline`
	Result:
	`<redlineInfoList>`
	`  <redlineInfo>`
	`  <color blue="0" green="0" red="0"/>`
	`  <user id="11">BOT</user>`
	`  </redlineInfo>`
	`  <redlineInfo>`
	`  <color blue="0" green="0" red="0"/>`
	`  <user id="1">Admin</user>`
	`  </redlineInfo>`
	`  <redlineInfo>`
	`  <color blue="0" green="0" red="0"/>`
	`  <user id="10">Admin2</user>`
	`  </redlineInfo>`
	`</redlineInfoList>`	

Function	Get Redline color for a given user.
Description	This request returns the defined redlining color for the given user.
Type	GET
URL Format	`.../redline`

Java API	**Method Name**		**Class Name**
	`getAllUserRedliningColor`		`com.quark.qpp.service.facadeClass AdminFacade`
Parameters	**Name**	**Type**	**Description**
	user	string	Name of the user for which redline color information is to be fetched.
Returns	Returns the defined redlining color for the given user. Returns specific privilege(s) in system with given id or name. If privilege name is specified and there exists multiple privileges in different privilege groups, with the given name, list of all such privileges will be returned.		

Examples	**URL:**
	
	`http://serverName:serverPort/rest/service/admin/redline?user=Admin`
	
	Result:
	
	`<redlineInfoList>`
	`  <redlineInfo>`
	`  <color blue="252" green="8" red="126"/>`
	`  <user id="1">Admin</user>`
	`  </redlineInfo>`
	`</redlineInfoList>`		

updateUserRedliningColor

Function	Updates the redlining color for the given user(s).		
Description	This request updates the redlining color for the given user(s).		
Type	POST		
URL Format	`.../redline? op=update`		
Java API	**Method Name**		**Class Name**
	`updateUserRedliningColor`		`com.quark.qpp.service.facadeClass AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`redlineinfolist= <XML/>`	string	A redLineInfoList object in XML format. See “Request/Response parameter formats.”
Returns	Returns the defined redlining color for the given user(s).		

Examples	URL:
	
	`http://serverName:serverPort/rest/service/admin/redline?user=BOT`
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<redlineInfoList>`
	`  <redlineInfo>`
	`  <color blue="10" green="56" red="115"/>`
	`  <user id="11">BOT</user>`
	`  </redlineInfo>`		

	`</redlineInfoList>`

updateUserEnabled

Function	Enables/disables the Platform Server Logon for the given user.		
Description	This request enables/disables the Platform Server logon for the given user.		
Type	POST		
URL Format	`.../admin/users/{userIdOrName}`		
Java API	**Method Name**		**Class Name**
	`updateUserEnabled`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=setenabled`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
	`enabled=[True] or [False]`	boolean	
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/users/John?op=setenabled&enabled=true:`		

updateUsers

Function	Update users.		
Description	This request is used to update a single user or multiple users at one time.		
Type	POST		
URL Format	`..../admin/users`		
Java API	**Method Name**		**Class Name**
	`updateUsers`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`userinfolist=<xml/>`		Object containing a list of the successfully updated users .
Returns	Returns a `UserInfoList` object containing a list of all the successfully updated users.		

Examples

URL:

```
http://serverName:serverPort/rest/service/admin/users?userinfolist=
```

```
<userInfoList>
    <userInfo id="33">
        <name>Tom</name>
        <firstName>Tom</firstName>
        <lastName>Rodrigues</lastName>
        <nativeTrustee>true</nativeTrustee>
        <enabled>false</enabled>
        <defaultUserClassId>12</defaultUserClassId>
        <defaultUserClass>Editor</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
</userInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<userInfoList>
    <userInfo id="33">
        <name>Tom</name>
        <firstName>Tom</firstName>
        <lastName>Rodrigues</lastName>
        <nativeTrustee>true</nativeTrustee>
        <enabled>false</enabled>
        <defaultUserClassId>12</defaultUserClassId>
        <defaultUserClass>Editor</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
</userInfoList>
```

Example with multiple content types

URL:

```
http://serverName:serverPort/rest/service/admin/users?userinfolist=
```

```
<userInfoList>
    <userInfo id="33">
        <name>Tom</name>
        <firstName>Tom</firstName>
        <lastName>Rodrigues</lastName>
        <nativeTrustee>true</nativeTrustee>
        <enabled>false</enabled>
        <defaultUserClassId>12</defaultUserClassId>
        <defaultUserClass>Editor</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
    <userInfo id="34">
        <name>Sam</name>
        <firstName>Sam</firstName>
        <lastName>Rodrigues</lastName>
        <nativeTrustee>true</nativeTrustee>
        <enabled>false</enabled>
        <defaultUserClassId>12</defaultUserClassId>
        <defaultUserClass>Editor</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
```

```
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
</userInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<userInfoList>
    <userInfo id="33">
        <name>Tom</name>
        <firstName>Tom</firstName>
        <lastName>Rodrigues</lastName>
        <nativeTrustee>true</nativeTrustee>
        <enabled>false</enabled>
        <defaultUserClassId>12</defaultUserClassId>
        <defaultUserClass>Editor</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
    <userInfo id="34">
        <name>Sam</name>
        <firstName>Sam</firstName>
        <lastName>Rodrigues</lastName>
        <nativeTrustee>true</nativeTrustee>
        <enabled>false</enabled>
        <defaultUserClassId>12</defaultUserClassId>
        <defaultUserClass>Editor</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
    </userInfo>
</userInfoList>
```

deleteUser

Function	Delete a user.		
Description	This request is used to delete the specified user.		
Type	POST		
URL Format	`.../admin/users/[userIdORName]`		
Java API	**Method Name**		**Class Name**
	`deleteUser`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`userIdORName`		Id or Name of the user to be deleted.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/users/32?op=delete`		

createGroups

Function	Create groups.		
Description	This request is used to create a single or multiple groups at one time.		
Type	POST		
URL Format	`.../admin/groups`		
Java API	**Method Name**		**Class Name**
	`createGroups`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`groupinfolist`		Object containing list of groups to be created. See "Request/Response parameter formats."
Returns	Returns a `groupInfoList` object containing its metadata and memberlist.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/groups?op=create&groupInfo=`
	`<groupInfoList>`
	`  <groupInfo>`
	`  <name>Editorial</name>`
	`  <emailAddress>Editors@company.com</emailAddress>`
	`  <nativeTrustee>false</nativeTrustee>`
	`  <defaultUserClassId>2</defaultUserClassId>`
	`  <defaultUserClass>NoPrivileges</defaultUserClass>`
	`  <ldapProfileId>10</ldapProfileId>`
	`  <ldapProfile>Profile Name</ldapProfile>`
	`  <ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`  <memberList>`
	`  <nameId>`
	`  <name>Admin</name>`
	`  </nameId> ..... </memberList>`
	`  </groupInfo>`
	`</groupInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<groupInfo id="48">`
	`  <name>SampleQPdzxcdPGroup</name>`
	`  <domainId>0</domainId>`
	`  <emailAddress>Editors@company.com</emailAddress>`
	`  <nativeTrustee>false</nativeTrustee>`
	`  <defaultUserClassId>2</defaultUserClassId>`
	`  <defaultUserClass>NoPrivileges</defaultUserClass>`
	`  <ldapProfileId>10</ldapProfileId>`
	`  <ldapProfile>Profile Name</ldapProfile>`
	`  <ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`  <memberList>`
	`  <nameId id="1" name="Admin"/>`
	`  <nameId id="10" name="Admin2"/>`
	`  </memberList>`
	`</groupInfo>`
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`		

```
<groupInfoList>
    <groupInfo id="20">
        <name>Editoria</name>
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId id="1">
                <name>Admin</name>
            </nameId> ........... </memberList>
    </groupInfo>
</groupInfoList>
```

Multiple Groups Example:

URL:

```
http://serverName:serverPort/rest/service/admin/groups?op=create&
groupinfolist=
```

```
<groupInfoList>
    <groupInfo>
        <name>Department</name>
        <deleted>false</deleted>
        <memberList>
            <nameId id="1">
                <name>Admin</name>
            </nameId> ........... </memberList>
    </groupInfo>
    <groupInfo>
        <name>Production</name>
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId id="1">
                <name>Admin</name>
            </nameId> ........... </memberList>
    </groupInfo>
</groupInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<groupInfoList>
    <groupInfo id="21">
        <name>Department</name>
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId id="1">
                <name>Admin</name>
            </nameId> ........... </memberList>
    </groupInfo>
```

	`<groupInfo id="22">`
	`<name>Production</name>`
	`<emailAddress>Editors@company.com</emailAddress>`
	`<nativeTrustee>false</nativeTrustee>`
	`<defaultUserClassId>2</defaultUserClassId>`
	`<defaultUserClass>NoPrivileges</defaultUserClass>`
	`<ldapProfileId>10</ldapProfileId>`
	`<ldapProfile>Profile Name</ldapProfile>`
	`<ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`<memberList>`
	`<nameId id="1">`
	`<name>Admin</name>`
	`</nameId> ..... </memberList>`
	`</groupInfo>`
	`</groupInfoList>`

getGroup

Function	Get a group with a particular ID or Name.	
Description	This request returns all the metadata of specified group.	
Type	GET	
URL Format	`.../admin/groups/[groupIDORgroupName]`	
Java API	**Method Name**	**Class Name**
	`getGroup`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `groupInfoList` object in XML format of specified group. See the table in the "Request/Response parameter formats" chapter.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/groups/21`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<groupInfoList>`
	`<groupInfo id="21">`
	`<name>Department</name>`
	`<emailAddress>Editors@company.com</emailAddress>`
	`<nativeTrustee>false</nativeTrustee>`
	`<defaultUserClassId>2</defaultUserClassId>`
	`<defaultUserClass>NoPrivileges</defaultUserClass>`
	`<ldapProfileId>10</ldapProfileId>`
	`<ldapProfile>Profile Name</ldapProfile>`
	`<ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`<memberList>`
	`<nameId id="1">`
	`<name>Admin</name>`
	`</nameId> ..... </memberList>`
	`</groupInfo>`
	`</groupInfoList>`	

getAllGroups

Function	Get all groups.	
Description	This request returns all groups includin its members.	
Type	GET	
URL Format	`.../admin/groups`	
Java API	**Method Name**	**Class Name**
	`getAllGroups`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `groupInfoList` object containing an array of groups that exist on the QPP server. See the table in the "Request/Response parameter formats" chapter.	
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/groups` **Result:**	

```
<groupInfoList>
    <groupInfo>
        <name>gp34</name>
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId name="Sam"/>
            <nameId id="1"/>
        </memberList>
    </groupInfo>
    <groupInfo>
        <name>gp35</name>
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId name="Sam"/>
            <nameId id="1"/>
        </memberList>
    </groupInfo>
</groupInfoList>
```

getGroupsWithGivenUserClass

Function	Get groups with the given role.
Description	This request returns the Platform Server groups with the given role.
Type	GET

URL Format	`.../admin/roles/{roleIdOrName}`		
Java API	**Method Name**		**Class Name**
	`getGroupsWithGivenUserClass`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`groups=[true]`	boolean	
Returns	Returns a `groupInfoList` object containing a list of groups that have the default role.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/roles/Administrator? groups=true`
	Result:
	`<groupInfoList>`
	`  <groupInfo id="12">`
	`  <name>Editors</name>`
	`  <nativeTrustee>true</nativeTrustee>`
	`  <defaultUserClassId>1</defaultUserClassId>`
	`  <defaultUserClass>Administrator</defaultUserClass>`
	`  <memberList>`
	`  <nameId id="3680">`
	`  <name>John</name>`
	`  </nameId>`
	`  </memberList>`
	`  </groupInfo>`
	`  <groupInfo id="13">`
	`  <name>Publishers</name>`
	`  <nativeTrustee>true</nativeTrustee>`
	`  <defaultUserClass>Administrator</defaultUserClass>`
	`  <memberList>`
	`  <nameId id="10">`
	`  <name>Mathias</name>`
	`  </nameId> ..... </memberList>`
	`  </groupInfo>`
	`</groupInfoList>`		

getUserGroups

Function	Get groups of a user.		
Description	This request returns all groups to which the given user belongs.		
Type	GET		
URL Format	`.../admin/users/{userIdOrName}`		
Java API	**Method Name**		**Class Name**
	`getUserGroups`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`userIdOrName`	string	
Returns	Returns a `groupInfoList` object containing a list of groups of which the given user is a member.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/users/John?groups=true`
	Result:
	`<groupInfoList>`
	`  <groupInfo id="12">`
	`  <name>Editors</name>`
	`  <nativeTrustee>true</nativeTrustee>`
	`  <defaultUserClassId>123/defaultUserClassId></defaultUserClassId>`
	`  <defaultUserClass>Editors</defaultUserClass>`
	`  </groupInfo>`
	`  <groupInfo id="3681">`
	`  <name>QPP Server</name>`
	`  <nativeTrustee>false</nativeTrustee>`
	`  <defaultUserClassId>10/defaultUserClassId></defaultUserClassId>`
	`  <defaultUserClass>UserClass2</defaultUserClass>`
	`  <ldapProfileId>764</ldapProfileId>`
	`  <ldapProfile>QPP LDAP Profile</ldapProfile>`
	`  <ldapDistinguishedName> CN=QPP Server,OU=Distribution-Lists,OU=QMHI,DC=india,DC=quark,DC=com </ldapDistinguishedName>`
	`  </groupInfo>`
	`</groupInfoList>`

updateGroups

Function	Update groups.		
Description	This request is used to update a single or multiple groups at one time.		
Type	POST		
URL Format	`.../admin/groups`		
Java API	**Method Name**		**Class Name**
	`updateGroups`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`groupinfolist`	XML	Object containing information for single or multiple groups. See “Request/Response parameter formats.”
Returns	Returns a `groupInfoList` object containing a list of all successfully updated groups.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/groups?op=update&groupInfo=`
	`<groupInfoList>`
	`  <groupInfo id="22">`
	`  <name>Production</name>`		

```
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId id="1">
                <name>Admin</name>
            </nameId> ........... </memberList>
    </groupInfo>
</groupInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<groupInfoList>
    <groupInfo id="22">
        <name>Production</name>
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId id="1">
                <name>Admin</name>
            </nameId> ........... </memberList>
    </groupInfo>
</groupInfoList>
```

Example with multiple groups:

URL:

```
http://serverName:serverPort/rest/service/admin/groups?op=update&
groupInfo=
```

```
<groupInfoList>
    <groupInfo id="21">
        <name>Department</name>
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId id="1">
                <name>Admin</name>
            </nameId> ........... </memberList>
    </groupInfo>
    <groupInfo id="20">
        <name>Editoria</name>
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId id="1">
```

```
                <name>Admin</name>
            </nameId> ........... </memberList>
    </groupInfo>
</groupInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<groupInfoList>
    <groupInfo id="21">
        <name>Department</name>
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId id="1">
                <name>Admin</name>
            </nameId> ........... </memberList>
    </groupInfo>
    <groupInfo id="20">
        <name>Editoria</name>
        <emailAddress>Editors@company.com</emailAddress>
        <nativeTrustee>false</nativeTrustee>
        <defaultUserClassId>2</defaultUserClassId>
        <defaultUserClass>NoPrivileges</defaultUserClass>
        <ldapProfileId>10</ldapProfileId>
        <ldapProfile>Profile Name</ldapProfile>
        <ldapDistinguishedName>Dist Name</ldapDistinguishedName>
        <memberList>
            <nameId id="1">
                <name>Admin</name>
            </nameId> ........... </memberList>
    </groupInfo>
</groupInfoList>
```

deleteGroup

Function	Delete a group.		
Description	This request is used to delete the group with the given id or name.		
Type	POST		
URL Format	`..../admin/group/[groupIdORgroupName]`		
Java API	**Method Name**		**Class Name**
	`deleteGroup`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/groups/[groupIdOR`		

	groupName]?op=delete

createContentTypes

Function	Create content types.		
Description	This request is used to create one or more content types and its child content types according to the parent content type ID.		
Type	POST		
URL Format	.../admin/contenttypes		
Java API	**Method Name**		**Class Name**
	createContentTypes		com.quark.qpp.service.facade Class AdminFacade
Parameters	**Name**	**Type**	**Description**
	op=create	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	contenttypeinfolist=<xml/>		Object containing a list of the content types to be created. After the creation of a content type, the child content types defined will be considered for creation. See the contentTypeInfoList paramater in “Request/Response parameter formats”.
Returns	Returns a contenTypeInfoList object containing containing the successfully created content types..		

Examples	**URL:**
	http://serverName:serverPort/rest/service/admin/contenttypes?op=create&contenttypeinfolist=
	`<contentTypeInfoList>`
	`  <contentTypeInfo>`
	`  <name>Pru Child1</name>`
	`  <extensible>true</extensible>`
	`  <parentContentTypeId>10010</parentContentTypeId>`
	`  <contentTypeInfo>`
	`  <name>Child of Project 1</name>`
	`  <extensible>true</extensible>`
	`  </contentTypeInfo>`
	`  </contentTypeInfo>`
	`</contentTypeInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<contentTypeInfoList>`
	`  <contentTypeInfo id="10011">`
	`  <name>Pru Child1</name>`
	`  <contentTypeHierarchy>System;Asset;Prudencial;Pru Child1;</contentTypeHierarchy>`
	`  <parentContentTypeId>10010</parentContentTypeId>`
	`  <systemDefined>false</systemDefined>`
	`  <childrenAvailable>true</childrenAvailable>`
	`  <extensible>true</extensible>`		

```
        <created>2013-11-25T07:21:00Z</created>
        <creator>1</creator>
        <creatorName>Admin</creatorName>
        <lastModified>2013-11-25T07:21:00Z</lastModified>
        <lastModifier>1</lastModifier>
        <lastModifierName>Admin</lastModifierName>
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="false" attributeId="1">Id </contentTypeAttribut-
eMappingInfo>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="true" attributeId="2">Name </contentTypeAttribut-
eMappingInfo> ........... </contentTypeAttributeMappingInfoList>
    </contentTypeInfo>
    <contentTypeInfo id="10012">
        <name>Child of Project 1</name>
        <contentTypeHierarchy>System;Asset;Prudencial;Pru
Child1;Child of Project 1;</contentTypeHierarchy>
        <parentContentTypeId>10011</parentContentTypeId>
        <systemDefined>false</systemDefined>
        <extensible>true</extensible>
        <created>2013-11-25T07:21:00Z</created>
        <creator>1</creator>
        <creatorName>Admin</creatorName>
        <lastModified>2013-11-25T07:21:00Z</lastModified>
        <lastModifier>1</lastModifier>
        <lastModifierName>Admin</lastModifierName>
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="false" attributeId="1">Id </contentTypeAttribut-
eMappingInfo>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="true" attributeId="2">Name </contentTypeAttribut-
eMappingInfo> ........... </contentTypeAttributeMappingInfoList>
    </contentTypeInfo>
</contentTypeInfoList>
```

Example with multiple content types:

URL:

```
http://serverName:serverPort/rest/service/admin/contenttypes?op=c
reate&contenttypeinfolist=
```

```
<contentTypeInfoList>
    <contentTypeInfo>
        <name>Pru 2</name>
        <parentContentTypeId>10010</parentContentTypeId>
        <childrenAvailable>true</childrenAvailable>
        <extensible>true</extensible>
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="false" attributeId="1">Id </contentTypeAttribut-
eMappingInfo> ........... </contentTypeAttributeMappingInfoList>
    </contentTypeInfo>
    <contentTypeInfo>
        <name>Child 1 of Pru 2</name>
        <parentContentTypeId>10011</parentContentTypeId>
        <extensible>true</extensible>
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo userMapped="true"
valueMandatory="false" attributeId="509">Master List </contentTy-
peAttributeMappingInfo> ........... </contentTypeAttributeMap-
pingInfoList>
    </contentTypeInfo>
```

	`</contentTypeInfoList>` `<>`:

Function	Create content types through content type hierarchy		
Description	This request is used to create one or more content types and its child content types according to the specified hierarchy		
Type	POST		
URL Format	`.../admin/contenttypes`		
Java API	**Method Name**		**Class Name**
	`createContentTypes`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`contentTypeInfoList=<xml/>`		Object containing a list of the content types to be created. After the creation of a content type, the child content types defined will be considered for creation. See the contentTypeInfoList paramater in "Request/Response parameter formats".
Returns	Returns a `contentTypeInfoList` object containing the successfully created content types.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/contenttypes?op=create&contenttypeinfolist=`
	`<contentTypeInfoList>`
	`  <contentTypeInfo>`
	`  <name>Prudencial</name>`
	`  <contentTypeHierarchy>System;Asset;Prudential;</contentTypeHierarchy>`
	`  <extensible>true</extensible>`
	`  <contentTypeAttributeMappingInfoList>`
	`  <contentTypeAttributeMappingInfo valueMandatory="true">Pru_Id</contentTypeAttributeMappingInfo>`
	`  <contentTypeAttributeMappingInfo userMapped="true" valueMandatory="false" attributeId="501">Master Text </contentTypeAttributeMappingInfo> ..... </contentTypeAttributeMappingInfoList>`
	`  </contentTypeInfo>`
	`</contentTypeInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<contentTypeInfoList>`
	`  <contentTypeInfo id="10010">`
	`  <name>Prudencial</name>`
	`  <contentTypeHierarchy>System;Asset;Prudencial;</contentTypeHierarchy>`
	`  <parentContentTypeId>1100</parentContentTypeId>`
	`  <systemDefined>false</systemDefined>`
	`  <extensible>true</extensible>`		

```
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="false" attributeId="1">Id </contentTypeAttribut-
eMappingInfo>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="true" attributeId="2">Name </contentTypeAttribut-
eMappingInfo> ........... </contentTypeAttributeMappingInfoList>
    </contentTypeInfo>
</contentTypeInfoList>
```

getContentType

Function	Get content type.		
Description	This request returns complete information on the specified content type. This will also include the child content types.		
Type	Get		
URL Format	`.../admin/contenttypes/{contenttypeIdorName}`		
Java API	**Method Name**		**Class Name**
	`getContentType`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`contentTypeIdOrName`	string	The ID or Name of the content type to be retrieved.
Returns	Returns a `contentTypeInfoList` object containing information on the specified content type inclusive of its children .		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/contenttypes/10000` **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<contentTypeInfoList>
    <contentTypeInfo id="10000">
        <name>Master</name>
        <contentTypeHierarchy>System;Article Component;Picture
Component;Master;</contentTypeHierarchy>
        <parentContentTypeId>1300</parentContentTypeId>
        <systemDefined>false</systemDefined>
        <extensible>false</extensible>
        <created>2013-11-22T07:04:29Z</created>
        <creator>1</creator>
        <creatorName>Admin</creatorName>
        <lastModified>2013-11-22T07:04:29Z</lastModified>
        <lastModifier>1</lastModifier>
        <lastModifierName>Admin</lastModifierName>
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="false" attributeId="1">Id </contentTypeAttribut-
eMappingInfo>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="false" attributeId="51">First page </contentTy-
peAttributeMappingInfo> ........... </contentTypeAttributeMap-
pingInfoList>
```

	`</contentTypeInfo>` `</contentTypeInfoList>`

getAllContentTypes

Function	Get all content types.	
Description	This request returns complete information of all content types in the system in a hierarchical structure.	
Type	GET	
URL Format	`.../admin/contenttypes`	
Java API	**Method Name**	**Class Name**
	`getAllContentTypes`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters	
Returns	Returns a `contenTypeInfoList` object containing an array of content types (contentTypeInfo).	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/contenttypes`
	Result:
	`<contentTypeInfoList>`
	`<contentTypeInfo id="1502">`
	`<name>Web</name>`
	`<contentTypeHierarchy>System;Layout;Web;</contentType-Hierarchy>`
	`<parentContentTypeId>1500</parentContentTypeId>`
	`<systemDefined>true</systemDefined>`
	`<extensible>false</extensible>`
	`<created>2012-04-01T10:00:00Z</created>`
	`<creator>1</creator>`
	`<creatorName>Admin</creatorName>`
	`<lastModified>2012-04-01T10:00:00Z</lastModified>`
	`<lastModifier>1</lastModifier>`
	`<lastModifierName>Admin</lastModifierName>`
	`<contentTypeAttributeMappingInfoList>`
	`<contentTypeAttributeMappingInfo userMapped="false" valueMandatory="false" attributeId="1">Id</contentTypeAttribut-eMappingInfo>`
	`<contentTypeAttributeMappingInfo userMapped="false" valueMandatory="false" attributeId="43">Number of pages</content-TypeAttributeMappingInfo> ..... </contentTypeAttributeMap-pingInfoList>`
	`</contentTypeInfo>`
	`</contentTypeInfoList>`	

updateContentTypes

Function	Update content types.
Description	This request is used to update content types and their children according to the specified id or contentTypeHierarchy.

Type	POST		
URL Format	`..../admin/contenttypes`		
Java API	**Method Name**		**Class Name**
	`updateContentTypes`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`contenttypeinfolist=<xml/>`		Object containing a list of the successfully updated content types .
Returns	Returns a `contentTypeInfoList` object containing a list of all the successfully updated content types..		

Examples

URL:

```
http://serverName:serverPort/rest/service/admin/contenttypes?op=update&contenttypeinfolist=
```

```
<contentTypeInfoList>
    <contentTypeInfo id="10015">
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo>
                <attributeName>Id</attributeName>
                <valueMandatory>true</valueMandatory>
            </contentTypeAttributeMappingInfo>
        </contentTypeAttributeMappingInfoList>
    </contentTypeInfo>
</contentTypeInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<contentTypeInfoList>
    <contentTypeInfo id="10015">
        <name>Prudential</name>
        <contentTypeHierarchy>System;Asset;Prudential;</content-
TypeHierarchy>
        <parentContentTypeId>1100</parentContentTypeId>
        <systemDefined>false</systemDefined>
        <childrenAvailable>true</childrenAvailable>
        <extensible>true</extensible>
        <created>2013-11-25T07:35:37Z</created>
        <creator>1</creator>
        <creatorName>Admin</creatorName>
        <lastModified>2013-11-25T07:35:37Z</lastModified>
        <lastModifier>1</lastModifier>
        <lastModifierName>Admin</lastModifierName>
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="false" attributeId="1">Id </contentTypeAttribut-
eMappingInfo> ........... </contentTypeAttributeMappingInfoList>
    </contentTypeInfo>
    <contentTypeInfo id="10016">
        <name>Pru Nest1</name>
        <contentTypeHierarchy>System;Asset;Prudential;Pru
Nest1;</contentTypeHierarchy>
        <parentContentTypeId>10015</parentContentTypeId>
        <systemDefined>false</systemDefined>
        <extensible>true</extensible>
```

```
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo userMapped="false"
valueMandatory="false" attributeId="1">Id </contentTypeAttribut-
eMappingInfo> ........... </contentTypeAttributeMappingInfoList>
    </contentTypeInfo>
</contentTypeInfoList>
```

Example with multiple content types:

URL:

```
http://serverName:serverPort/rest/service/admin/users?op=create&u
serinfolist=
```

```
<?xml version="1.0" encoding="UTF-8"?>
<userInfoList>
    <userInfo>
        <name>SampleQPPGroup1</name>
        <nativeTrustee>true</nativeTrustee>
        <enabled>true</enabled>
        <defaultUserClassId>1</defaultUserClassId>
        <defaultUserClass>Administrator</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
    </userInfo>
    <userInfo>
        <name>SampleQPPGroup2</name>
        <nativeTrustee>true</nativeTrustee>
        <enabled>true</enabled>
        <defaultUserClassId>1</defaultUserClassId>
        <defaultUserClass>Administrator</defaultUserClass>
        <ldapProfileId>0</ldapProfileId>
        <deleted>false</deleted>
    </userInfo>
</userInfoList>
```

Result:

```
http://serverName:serverPort/rest/service/admin/contenttypes?op=u
pdate&contenttypeinfolist=
```

```
<contentTypeInfoList>
    <contentTypeInfo>
        <name>project aaa new name</name>
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo>
                <attributeName>Master List</attributeName>
                <valueMandatory>true</valueMandatory>
            </contentTypeAttributeMappingInfo>
        </contentTypeAttributeMappingInfoList>
        <contentTypeHierarchy>System;Asset;Prudential;project aaa
new name;</contentTypeHierarchy>
    </contentTypeInfo>
    <contentTypeInfo>
        <name>Pru Nest 1</name>
        <contentTypeAttributeMappingInfoList>
            <contentTypeAttributeMappingInfo>
                <attributeName>Master Day</attributeName>
                <valueMandatory>true</valueMandatory>
            </contentTypeAttributeMappingInfo>
        </contentTypeAttributeMappingInfoList>
        <contentTypeHierarchy>System;Asset;Prudential;Pru Nest 1
New Name;</contentTypeHierarchy>
    </contentTypeInfo>
</contentTypeInfoList>
```

deleteContentType

Function	Delete a content type.		
Description	This request is used to delete the specified content type.		
Type	POST		
URL Format	`.../admin/contenttypes/{contenttypeIdorName}`		
Java API	**Method Name**		**Class Name**
	`deleteContentType`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`contentTypeIdOrName`	string	Id or Name of the content type to be deleted.
	`attributes= [comma-separated list of attribute names or IDs]`	string	Attributes Ids or Names.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/contenttypes/10015?op=delete:`		

createAttributes

Function	Create Attributes.		
Description	This request is used to create a single or multiple attributes at one time.		
Type	POST		
URL Format	`.../admin/attributes`		
Java API	**Method Name**		**Class Name**
	`createAttributes`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`attributeinfolist`	string	List of attributes to create.
Returns	Returns a `attributeInfoList` object containing a list of successfully created attributes.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/attributes?op=cre`		

```
ate&attributeinfolist=

<attributeInfoList>
    <attributeInfo>
        <name>Collections Sequence</name>
        <valueType>TEXT</valueType>
        <queryable>true</queryable>
        <creatorName>Admin</creatorName>
        <displayable>true</displayable>
        <valuePreferencesChangeable>true</valuePreferencesChange-
able>
        <attributeValuePreferenceInfo>
            <defaultValue>From Creation to Publishing</default-
Value>
        </attributeValuePreferenceInfo>
    </attributeInfo>
</attributeInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<attributeInfoList>
    <attributeInfo id="519">
        <name>Collections Sequence</name>
        <valueTypeId>1</valueTypeId>
        <valueType>TEXT</valueType>
        <userDefined>true</userDefined>
        <queryable>true</queryable>
        <modificationLevel>USER_MODIFIABLE</modificationLevel>
        <created>2013-12-05T05:52:57Z</created>
        <creator>1</creator>
        <creatorName>Admin</creatorName>
        <lastModified>2013-12-05T05:52:57Z</lastModified>
        <lastModifier>1</lastModifier>
        <lastModifierName>Admin</lastModifierName>
        <displayable>true</displayable>
        <limitedAccess>false</limitedAccess>
        <limitedAccessChangeable>true</limitedAccessChangeable>
        <constraintsChangeable>true</constraintsChangeable>
        <valuePreferencesChangeable>true</valuePreferencesChange-
able>
        <attributeValuePreferenceInfo>
            <defaultValue>From Creation to Publishing</default-
Value>
            <textSearchEnabled>false</textSearchEnabled>
        </attributeValuePreferenceInfo>
    </attributeInfo>
</attributeInfoList>
```

Example with multiple attributes:

URL:

```
http://serverName:serverPort/rest/service/admin/attributes?op=cre
ate&attributeinfolist=

<attributeInfoList>
    <attributeInfo>
        <name>Collections Sequence 1</name>
        <valueType>TEXT</valueType>
        <queryable>true</queryable>
        <creatorName>Admin</creatorName>
        <displayable>true</displayable>
        <valuePreferencesChangeable>true</valuePreferencesChange-
able>
        <attributeValuePreferenceInfo>
```

```
            <defaultValue>From Creation to Publishing</default-
Value>
        </attributeValuePreferenceInfo>
    </attributeInfo>
    <attributeInfo>
        <name>Collections Sequence 2</name>
        <valueType>TEXT</valueType>
        <queryable>true</queryable>
        <creatorName>Admin</creatorName>
        <displayable>true</displayable>
        <valuePreferencesChangeable>true</valuePreferencesChange-
able>
        <attributeValuePreferenceInfo>
            <defaultValue>From Creation to Publishing</default-
Value>
        </attributeValuePreferenceInfo>
    </attributeInfo>
</attributeInfoList>
```


Result:


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<attributeInfoList>
    <attributeInfo id="519">
        <name>Collections Sequence 1</name>
        <valueTypeId>1</valueTypeId>
        <valueType>TEXT</valueType>
        <userDefined>true</userDefined>
        <queryable>true</queryable>
        <modificationLevel>USER_MODIFIABLE</modificationLevel>
        <created>2013-12-05T05:52:57Z</created>
        <creator>1</creator>
        <creatorName>Admin</creatorName>
        <lastModified>2013-12-05T05:52:57Z</lastModified>
        <lastModifier>1</lastModifier>
        <lastModifierName>Admin</lastModifierName>
        <displayable>true</displayable>
        <limitedAccess>false</limitedAccess>
        <limitedAccessChangeable>true</limitedAccessChangeable>
        <constraintsChangeable>true</constraintsChangeable>
        <valuePreferencesChangeable>true</valuePreferencesChange-
able>
        <attributeValuePreferenceInfo>
            <defaultValue>From Creation to Publishing</default-
Value>
            <textSearchEnabled>false</textSearchEnabled>
        </attributeValuePreferenceInfo>
    </attributeInfo>
    <attributeInfo id="520">
        <name>Collections Sequence 2</name>
        <valueTypeId>1</valueTypeId>
        <valueType>TEXT</valueType>
        <userDefined>true</userDefined>
        <queryable>true</queryable>
        <modificationLevel>USER_MODIFIABLE</modificationLevel>
        <created>2013-12-05T05:52:57Z</created>
        <creator>1</creator>
        <creatorName>Admin</creatorName>
        <lastModified>2013-12-05T05:52:57Z</lastModified>
        <lastModifier>1</lastModifier>
        <lastModifierName>Admin</lastModifierName>
        <displayable>true</displayable>
        <limitedAccess>false</limitedAccess>
        <limitedAccessChangeable>true</limitedAccessChangeable>
        <constraintsChangeable>true</constraintsChangeable>
```

	`<valuePreferencesChangeable>true</valuePreferencesChangeable>`
	`<attributeValuePreferenceInfo>`
	`<defaultValue>From Creation to Publishing</defaultValue>`
	`<textSearchEnabled>false</textSearchEnabled>`
	`</attributeValuePreferenceInfo>`
	`</attributeInfo>`
	`</attributeInfoList>`

setAttributes

Function	Map ContentType with attributes.		
Description	This request is used to update the attribute mapping of content types and their children with the specified map of attributes.		
Type	POST		
URL Format	`..../admin/contenttypes`		
Java API	**Method Name**		**Class Name**
	`setAttributes`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=setattributes`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`contentTypes= [comma-separated list of Content Types]`	string	Content type IDs or Names.
	`attributes= [comma-separated list of attribute names or IDs]`	string	Attributes IDs or Names.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/contenttypes?op=setattributes&contenttypes=1000&attributes= 180,305:`		

getAttribute

Function	Get an attribute with a specific ID or Name.	
Description	This request returns an attribute with the specified ID or Name.	
Type	GET	
URL Format	`.../admin/attributes/{attributeIdORattributeName}`	
Java API	**Method Name**	**Class Name**
	`getAttribute`	`com.quark.qpp.service.facade`

		Class AdminFacade
Parameters	This request requires no input parameters.	
Returns	Returns an attributeInfoList object in XML format.	

Examples	**URL:**
	http://serverName:serverPort/rest/service/admin/attributes/collection
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<attributeInfoList>`
	`  <attributeInfo id="55">`
	`  <name>Collection</name>`
	`  <valueTypeId>7</valueTypeId>`
	`  <valueType>DOMAIN</valueType>`
	`  <userDefined>false</userDefined>`
	`  <queryable>false</queryable>`
	`  <modificationLevel>USER_MODIFIABLE</modificationLevel>`
	`  <creator>1</creator>`
	`  <creatorName>Admin</creatorName>`
	`  <lastModifier>1</lastModifier>`
	`  <lastModifierName>Admin</lastModifierName>`
	`  <displayable>true</displayable>`
	`  <limitedAccess>false</limitedAccess>`
	`  <limitedAccessChangeable>false</limitedAccessChangeable>`
	`  <constraintsChangeable>false</constraintsChangeable>`
	`  <valuePreferencesChangeable>false</valuePreferencesChangeable>`
	`  <attributeValuePreferenceInfo>`
	`  <domainId>19</domainId>`
	`  <domainName>Collections</domainName>`
	`  </attributeValuePreferenceInfo>`
	`  </attributeInfo>`
	`</attributeInfoList>`	

getAllAttributes

Function	Get all attributes and attribute IDs.	
Description	This request returns all existing attributes with their metadata of QPP Server.	
Type	GET	
URL Format	.../admin/attributes	
Java API	**Method Name**	**Class Name**
	getAllAttributes	com.quark.qpp.service.facade Class AdminFacade
Parameters	This request requires no input parameters.	
Returns	Returns an attributeInfoList object containing an array of attributes that exist in the QPP server.	

Examples	**URL:**
	http://serverName:serverPort/rest/service/admin/attributes
	`<attributeInfoList>`
	`  <attributeInfo id="180">`
	`  <name>App Studio Issue URN</name> ..... </attributeInfo>`	

	`<attributeInfo id="305">`
	`<name>Article Component id</name> ..... </attributeInfo>`
	`..... </attributeInfoList>`
	Result:
	An `attributeInfoList` object containing an array of attributes that exist in the QPP server.

updateAttributes

Function	Update attributes.		
Description	This request is used to update the metadata of a single or multiple attributes.		
Type	POST		
URL Format	`.../admin/attributes`		
Java API	**Method Name**		**Class Name**
	`updateAttributes`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`attributeinfolist`	string	List of attributes to be updated.
Returns	Returns an `attributeInfoList` object containing a list of successfully updated attributes.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/attributes?op=update&attributevaluelist=`
	`<attributeInfoList>`
	`<attributeInfo id="507">`
	`<name>Release Version</name>`
	`<valueTypeId>5</valueTypeId>`
	`<valueType>MEASUREMENT</valueType>`
	`<userDefined>true</userDefined>`
	`<queryable>true</queryable>`
	`<modificationLevel>USER_MODIFIABLE</modificationLevel>`
	`<displayable>true</displayable>`
	`<limitedAccess>false</limitedAccess>`
	`<limitedAccessChangeable>true</limitedAccessChangeable>`
	`<constraintsChangeable>true</constraintsChangeable>`
	`<valuePreferencesChangeable>true</valuePreferencesChangeable>`
	`<attributeValuePreferenceInfo>`
	`<defaultValue>4</defaultValue>`
	`<maximumPossibleValue>7</maximumPossibleValue>`
	`<minimumPossibleValue>2</minimumPossibleValue>`
	`<measurementUnit>10</measurementUnit>`
	`</attributeValuePreferenceInfo>`
	`</attributeInfo>`
	`</attributeInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`		

```
<attributeInfoList>
    <attributeInfo id="507">
        <name>Release Version</name>
        <valueTypeId>5</valueTypeId>
        <valueType>MEASUREMENT</valueType>
        <userDefined>true</userDefined>
        <queryable>true</queryable>
        <modificationLevel>USER_MODIFIABLE</modificationLevel>
        <created>2013-12-05T05:14:07Z</created>
        <creator>1</creator>
        <creatorName>Admin</creatorName>
        <lastModified>2013-12-05T10:13:29Z</lastModified>
        <lastModifier>1</lastModifier>
        <lastModifierName>Admin</lastModifierName>
        <displayable>true</displayable>
        <limitedAccess>false</limitedAccess>
        <limitedAccessChangeable>true</limitedAccessChangeable>
        <constraintsChangeable>true</constraintsChangeable>
        <valuePreferencesChangeable>true</valuePreferencesChange-
able>
        <attributeValuePreferenceInfo>
            <defaultValue>4.0</defaultValue>
            <maximumPossibleValue>7.0</maximumPossibleValue>
            <minimumPossibleValue>2.0</minimumPossibleValue>
            <measurementUnit>10</measurementUnit>
        </attributeValuePreferenceInfo>
    </attributeInfo>
</attributeInfoList>
```

Example with multiple attributes to update:

URL:

```
http://serverName:serverPort/rest/service/admin/attributes?op=upd
ate&attributevaluelist=
```

```
<attributeInfoList>
    <attributeInfo id="502">
        <name>Article Release 1</name>
        <valueTypeId>2</valueTypeId>
        <valueType>DATE</valueType>
        <userDefined>true</userDefined>
        <queryable>true</queryable>
        <modificationLevel>USER_MODIFIABLE</modificationLevel>
        <displayable>true</displayable>
        <limitedAccess>false</limitedAccess>
        <limitedAccessChangeable>true</limitedAccessChangeable>
        <constraintsChangeable>true</constraintsChangeable>
        <valuePreferencesChangeable>true</valuePreferencesChange-
able>
        <attributeValuePreferenceInfo>
            <defaultValue>2013-12-06</defaultValue>
            <maximumPossibleValue>2013-12-31</maximumPossible-
Value>
            <minimumPossibleValue>2013-12-02</minimumPossible-
Value>
        </attributeValuePreferenceInfo>
    </attributeInfo>
    <attributeInfo id="503">
        <name>Article Release 2</name>
        <valueTypeId>2</valueTypeId>
        <valueType>DATE</valueType>
        <userDefined>true</userDefined>
        <queryable>true</queryable>
        <modificationLevel>USER_MODIFIABLE</modificationLevel>
```

```
        <displayable>true</displayable>
        <limitedAccess>false</limitedAccess>
        <limitedAccessChangeable>true</limitedAccessChangeable>
        <constraintsChangeable>true</constraintsChangeable>
        <valuePreferencesChangeable>true</valuePreferencesChange-
able>
        <attributeValuePreferenceInfo>
            <defaultValue>2013-12-06</defaultValue>
            <maximumPossibleValue>2013-12-31</maximumPossible-
Value>
            <minimumPossibleValue>2013-12-02</minimumPossible-
Value>
        </attributeValuePreferenceInfo>
    </attributeInfo>
</attributeInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<attributeInfoList>
    <attributeInfo id="502">
        <name>Article Release 1</name>
        <valueTypeId>2</valueTypeId>
        <valueType>DATE</valueType>
        <userDefined>true</userDefined>
        <queryable>true</queryable>
        <modificationLevel>USER_MODIFIABLE</modificationLevel>
        <created>2013-12-05T05:11:51Z</created>
        <creator>1</creator>
        <creatorName>Admin</creatorName>
        <lastModified>2013-12-05T10:17:38Z</lastModified>
        <lastModifier>1</lastModifier>
        <lastModifierName>Admin</lastModifierName>
        <displayable>true</displayable>
        <limitedAccess>false</limitedAccess>
        <limitedAccessChangeable>true</limitedAccessChangeable>
        <constraintsChangeable>true</constraintsChangeable>
        <valuePreferencesChangeable>true</valuePreferencesChange-
able>
        <attributeValuePreferenceInfo>
            <defaultValue>2013-12-06</defaultValue>
            <maximumPossibleValue>2013-12-31</maximumPossible-
Value>
            <minimumPossibleValue>2013-12-02</minimumPossible-
Value>
        </attributeValuePreferenceInfo>
    </attributeInfo>
    <attributeInfo id="503">
        <name>Article Release 2</name>
        <valueTypeId>2</valueTypeId>
        <valueType>DATE</valueType>
        <userDefined>true</userDefined>
        <queryable>true</queryable>
        <modificationLevel>USER_MODIFIABLE</modificationLevel>
        <created>2013-12-05T05:11:51Z</created>
        <creator>1</creator>
        <creatorName>Admin</creatorName>
        <lastModified>2013-12-05T10:17:38Z</lastModified>
        <lastModifier>1</lastModifier>
        <lastModifierName>Admin</lastModifierName>
        <displayable>true</displayable>
        <limitedAccess>false</limitedAccess>
        <limitedAccessChangeable>true</limitedAccessChangeable>
        <constraintsChangeable>true</constraintsChangeable>
        <valuePreferencesChangeable>true</valuePreferencesChange-
able>
```

	`<attributeValuePreferenceInfo>`
	`<defaultValue>2013-12-06</defaultValue>`
	`<maximumPossibleValue>2013-12-31</maximumPossible-Value>`
	`<minimumPossibleValue>2013-12-02</minimumPossible-Value>`
	`</attributeValuePreferenceInfo>`
	`</attributeInfo>`
	`</attributeInfoList>`

deleteAttribute

Function	Delete an Attribute.		
Description	This request is used to delete an attribute with the specified id or name.		
Type	POST		
URL Format	`.../admin/attributes/{attributeIdORattributeName}`		
Java API	**Method Name**		**Class Name**
	`deleteAttribute`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`attributeIdORName`	long or string	The Name or ID of the attribute to be deleted.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/attributes/[attributeIdorName]?op=delete`		

createForms

Function	Create forms.		
Description	This request is used to create a new form in the system.		
Type	POST		
URL Format	`..../admin/forms`		
Java API	**Method Name**		**Class Name**
	`createForms`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`forminfolist`	string	List of forms to be created.

Returns	Returns a `formInfoList` object containing a list of successfully created forms.A status of 200 is returned in response to a successful request execution.

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/forms?op=create&forminfolist=`
	`<formInfoList>`
	`  <formInfo>`
	`  <contentTypeName>Project</contentTypeName>`
	`  <workflowName>Quark</workflowName>`
	`  <formAttributeList>`
	`  <formAttribute>`
	`  <name>Collection Path</name>`
	`  <xPos>0</xPos>`
	`  <yPos>175</yPos>`
	`  <totalWidth>450</totalWidth>`
	`  <controlWidth>300</controlWidth>`
	`  <labelWidth>150</labelWidth>`
	`  <valueType>1</valueType>`
	`  </formAttribute> ..... </formAttributeList>`
	`  </formInfo>`
	`</formInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<formInfoList>`
	`  <formInfo id="32">`
	`  <contentTypeId>1102</contentTypeId>`
	`  <contentTypeName>Project</contentTypeName>`
	`  <workflowId>10</workflowId>`
	`  <workflowName>Quark</workflowName>`
	`  <defaultView>false</defaultView>`
	`  <formAttributeList>`
	`  <formAttribute id="57">`
	`  <name>Collection Path</name>`
	`  <xPos>0</xPos>`
	`  <yPos>175</yPos>`
	`  <totalWidth>450</totalWidth>`
	`  <controlWidth>300</controlWidth>`
	`  <labelWidth>150</labelWidth>`
	`  <valueType>1</valueType>`
	`  </formAttribute> ..... </formAttributeList>`
	`  </formInfo>`
	`</formInfoList>`
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<formInfoList>`
	`  <formInfo id="58">`
	`  <contentTypeId>1413</contentTypeId>`
	`  <contentTypeName>Project</contentTypeName>`
	`  <workflowId>0</workflowId>`
	`  <defaultView>false</defaultView>`
	`  <formAttributeList>`
	`  <formAttribute id="57">`
	`  <name>Collection Path</name>`
	`  <xPos>0</xPos>`
	`  <yPos>0</yPos>`
	`  <totalWidth>450</totalWidth>`
	`  <controlWidth>300</controlWidth>`
	`  <labelWidth>150</labelWidth>`
	`  <valueType>1</valueType>`
	`  </formAttribute> ..... </formAttributeList>`
	`  </formInfo> ..... </formInfoList>`

getAllForms

Function	Get all forms.	
Description	This request is used to get all forms saved in the system.	
Type	GET	
URL Format	`.../admin/forms`	
Java API	**Method Name**	**Class Name**
	`getAllForms`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters	
Returns	Returns a `formInfoList` object containing a list of all the forms.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/forms`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<formInfoList>`
	`  <formInfo id="58">`
	`  <contentTypeId>1413</contentTypeId>`
	`  <contentTypeName>Project</contentTypeName>`
	`  <workflowId>0</workflowId>`
	`  <defaultView>false</defaultView>`
	`  <formAttributeList>`
	`  <formAttribute id="57">`
	`  <name>Collection Path</name>`
	`  <xPos>0</xPos>`
	`  <yPos>0</yPos>`
	`  <totalWidth>450</totalWidth>`
	`  <controlWidth>300</controlWidth>`
	`  <labelWidth>150</labelWidth>`
	`  <valueType>1</valueType>`
	`  </formAttribute> ..... </formAttributeList>`
	`  </formInfo> ..... </formInfoList>`	

Function	Get forms with a specific workflow		
Description	This request returns all forms specific to given workflow.		
Type	GET		
URL Format	`..../admin/forms`		
Java API	**Method Name**		**Class Name**
	`getAllForms`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`workflow`	string	ID or name of the workflow.
Returns	Returns a `formInfoList` object containing a list of all the forms specific to a given workflow.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/forms?workflow=Quark`		

	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<formInfoList>`
	`  <formInfo id="53">`
	`  <contentTypeId>1100</contentTypeId>`
	`  <contentTypeName>Asset</contentTypeName>`
	`  <workflowId>11</workflowId>`
	`  <workflowName>Quark</workflowName>`
	`  <defaultView>false</defaultView>`
	`  <formAttributeList>`
	`  <formAttribute id="34">`
	`  <name>Checkedout Application</name>`
	`  <xPos>0</xPos>`
	`  <yPos>0</yPos>`
	`  <totalWidth>450</totalWidth>`
	`  <controlWidth>314</controlWidth>`
	`  <labelWidth>154</labelWidth>`
	`  <valueType>1</valueType>`
	`  </formAttribute> ..... </formAttributeList>`
	`  </formInfo>`
	`</formInfoList>`

Function	Get forms with a specific content type.		
Description	This request is used to get all forms specific to a given content type.		
Type	GET		
URL Format	`..../admin/forms`		
Java API	**Method Name**		**Class Name**
	`getAllForms`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`contenttype`	string	ID or name of the content type. Mandatory.
	`workflow`	string	ID of name of the worflow. Mandatory.
Returns	Returns a `formInfoList` object containing a list of all the forms specific to a given content type.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/forms?`
	`Result`**t:**
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <formInfoList>`
	`  <formInfo id="58">`
	`  <contentTypeId>1413</contentTypeId>`
	`  <contentTypeName>Project</contentTypeName>`
	`  <workflowId>0</workflowId>`
	`  <defaultView>false</defaultView>`
	`  <formAttributeList>`
	`  <formAttribute id="57">`
	`  <name>Collection Path</name>`
	`  <xPos>0</xPos>`
	`  <yPos>0</yPos>`
	`  <totalWidth>450</totalWidth>`
	`  <controlWidth>300</controlWidth>`
	`  <labelWidth>150</labelWidth>`		

```
          <valueType>1</valueType>
        </formAttribute>
        ...........
      </formAttributeList>
    </formInfo>
    .......
  </formInfoList>
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <formInfoList>
    <formInfo id="54">
      <contentTypeId>1101</contentTypeId>
      <contentTypeName>Document</contentTypeName>
      <workflowId>11</workflowId>
      <workflowName>Quark</workflowName>
      <defaultView>false</defaultView>
      <formAttributeList>
        <formAttribute id="13">
          <name>Is checked out</name>
          <xPos>0</xPos>
          <yPos>184.97</yPos>
          <totalWidth>457</totalWidth>
          <controlWidth>304.66666666666663</controlWidth>
          <labelWidth>154</labelWidth>
          <valueType>6</valueType>
        </formAttribute>
        ...........
      </formAttributeList>
    </formInfo>
  </formInfoList>
```

updateForms

Function	Update forms.		
Description	This request is used to updates metadata of a single or multiple forms at one time.		
Type	POST		
URL Format	`..../admin/forms`		
Java API	**Method Name**		**Class Name**
	`updateForms`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	Name	Type	Description
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`forminfolist`	string	List of forms to be updated.
Returns	Returns a `formInfoList` object containing a list of all the forms specific to a given workflow.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/forms?op=update&forminfolist=`
	`<formInfoList>`
	`  <formInfo>`
	`  <contentTypeName>Document</contentTypeName>`		

```
      <workflowName>Quark</workflowName>
      <formAttributeList>
        <formAttribute id="13">
          <name>Is checked out</name>
          <xPos>0</xPos>
          <yPos>184.97</yPos>
          <totalWidth>457</totalWidth>
          <controlWidth>304.66666666666663</controlWidth>
           <labelWidth>154</labelWidth>
           <valueType>6</valueType>
         </formAttribute>
       </formAttributeList>
     </formInfo>
   </formInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>  <formIn-
foList>
   <formInfo id="11">
     <contentTypeId>1101</contentTypeId>
     <contentTypeName>Document</contentTypeName>
     <workflowId>10</workflowId>
     <workflowName>Quark</workflowName>
     <defaultView>false</defaultView>
     <formAttributeList>
       <formAttribute id="13">
         <name>Is checked out</name>
         <xPos>0</xPos>
         <yPos>184.97</yPos>
         <totalWidth>457</totalWidth>
         <controlWidth>304.66666666666663</controlWidth>
         <labelWidth>154</labelWidth>
         <valueType>6</valueType>
       </formAttribute>
     </formAttributeList>
   </formInfo>
 </formInfoList>
```

Example with multiple forms to update

URL:

```
http://serverName:serverPort/rest/service/admin/forms?op=update&f
orminfolist=
```

```
  <formInfoList>
    <formInfo id="91">
      <contentTypeId>10004</contentTypeId>
      <contentTypeName>Archive</contentTypeName>
      <workflowId>0</workflowId>
      <defaultView>false</defaultView>
        <formAttributeList>
          <formAttribute id="501">
          <name>Master Text</name>
          <xPos>0</xPos>
          <yPos>263.5</yPos>
          <totalWidth>457</totalWidth>
          <controlWidth>304.66666666666663</controlWidth>
          <labelWidth>154</labelWidth>
          <valueType>1</valueType>
        </formAttribute>
          ...........
        </formAttributeList>
      </formInfo>
      <formInfo id="92">
        <contentTypeId>1403</contentTypeId>
```

```
        <contentTypeName>Book</contentTypeName>
        <workflowId>0</workflowId>
        <defaultView>false</defaultView>
        <formAttributeList>
          <formAttribute id="3">
          <name>Created</name>
          <xPos>0</xPos>
          <yPos>60.53</yPos>
          <totalWidth>457</totalWidth>
          <controlWidth>304.66666666666663</controlWidth>
          <labelWidth>154</labelWidth>
          <valueType>8</valueType>
        </formAttribute>
          ...........
      </formAttributeList>
    </formInfo>
  </formInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <formInfoList>
    <formInfo id="91">
    <contentTypeId>10004</contentTypeId>
    <contentTypeName>Archive</contentTypeName>
    <workflowId>0</workflowId>
    <defaultView>false</defaultView>
    <formAttributeList>
       <formAttribute id="501">
       <name>Master Text</name>
       <xPos>0</xPos>
       <yPos>263.5</yPos>
       <totalWidth>457</totalWidth>
       <controlWidth>304.66666666666663</controlWidth>
       <labelWidth>154</labelWidth>
       <valueType>1</valueType>
     </formAttribute>
       ...........
     </formAttributeList>
   </formInfo>
   <formInfo id="92">
     <contentTypeId>1403</contentTypeId>
     <contentTypeName>Book</contentTypeName>
     <workflowId>0</workflowId>
     <defaultView>false</defaultView>
     <formAttributeList>
       <formAttribute id="3">
       <name>Created</name>
```

deleteForm

Function	Delete a form.	
Description	This request is used to delete the specified form.	
Type	POST	
URL Format	`..../admin/forms`	
Java API	Method Name	Class Name
	`deleteForm`	`com.quark.qpp.service.facade Class AdminFacade`

Parameters	Name	Type	Description
	op=delete	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** http://serverName:serverPort/rest/service/admin/forms?op=delete&workflow=Quark&contenttype=Document		

createRoles

Function	Create roles.		
Description	This request is used to create one or more roles/userclass in a single request.		
Type	POST		
URL Format	.../admin/roles		
Java API	**Method Name**		**Class Name**
	createRoles		com.quark.qpp.service.facade Class AdminFacade
Parameters	**Name**	**Type**	**Description**
	op=create	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	roleinfolist =<xml/>		Object containing a list of the roles to be created. See the roleInfoList paramater in "*Request/Response parameter formats*".
Returns	Returns a roleInfoList object containing containing a list of the successfully created roles.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/admin/roles?op=create&roleinfolist=
	`<roleInfoList>`
	`  <roleInfo id="11" name="Updated Name of Role">`
	`  <applicationPrivList>`
	`  <privilege id="190000017">`
	`  <name>Remove Link with Collection Template</name>`
	`  </privilege>`
	`  <privilege id="999991">`
	`  <name>Live Preview</name>`
	`  </privilege> .... </applicationPrivList>`
	`  <contentTypePrivList>`
	`  <contentTypePriv contentTypeId="1100" contentType="Asset" contentTypeHierarchy="System;Asset;">`
	`  <privilege id="10000015">`
	`  <name>Get Original Assets</name>`
	`  </privilege> ..... </contentTypePriv> ..... </contentTypePrivList>`
	`  </roleInfo>`
	`</roleInfoList>`		

	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<roleInfoList>`
	`  <roleInfo id="11" name="Updated Name of Role">`
	`  <applicationPrivList>`
	`  <privilege id="190000017">`
	`  <name>Remove Link with Collection Template</name>`
	`  </privilege>`
	`  <privilege id="999991">`
	`  <name>Live Preview</name>`
	`  </privilege> ..... </applicationPrivList>`
	`  <contentTypePrivList>`
	`  <contentTypePriv contentTypeId="1100" contentType="Asset" contentTypeHierarchy="System;Asset;">`
	`  <privilege id="10000015">`
	`  <name>Get Original Assets</name>`
	`  </privilege> ..... </contentTypePriv> .....`
	`</contentTypePrivList>`
	`  </roleInfo>`
	`</roleInfoList>`

getRole

Function	Get a specific userclass/role.		
Description	This request returns the metadata of specified userclass/role.		
Type	GET		
URL Format	`../admin/roles/{roleIdORroleName}`		
Java API	**Method Name**		**Class Name**
	`getRole`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**

	`roleId`
	OR
	`roleName`
		string	The ID or Name of the role to be retrieved.

Returns	Returns a `roleInfoList` object in XML format.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/roles/Administrator`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<roleInfoList>`
	`  <roleInfo name="Administrator" id="1">`
	`  <applicationPrivList>`
	`  <privilege id="190000016">`
	`  <name>Status Based Routings</name>`
	`  </privilege> ..... </applicationPrivList>`
	`  <contentTypePrivList>`
	`  <contentTypePriv contentTypeHierarchy="System;Asset;" contentType="Asset" contentTypeId="1100">`
	`  <privilege id="10000015">`		

	`<name>Get Original Assets</name>`
	`</privilege> ..... </contentTypePriv>`
	`<contentTypePriv contentTypeHierarchy="System;Asset;Media;Video;" contentType="Video" contentTypeId="1152">`
	`<privilege id="10000015">`
	`<name>Get Original Assets</name>`
	`</privilege> ..... </contentTypePriv>`
	`</contentTypePrivList> ..... </roleInfo>`
	`</roleInfoList>`

getAllRoles

Function	Get all userclasses/roles.	
Description	This request returns all the userclasses with its metadata.	
Type	GET	
URL Format	`../admin/roles`	
Java API	**Method Name**	**Class Name**
	`getAllRoles`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `roleInfoList` object containing an array of all roles that exist on the QPP Server.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/roles`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<roleInfoList>`
	`<roleInfo name="Administrator" id="1"/>`
	`<roleInfo name="UserClass2" id="10"/>`
	`</roleInfoList>`	

updateRoles

Function	Update roles.		
Description	This request is used to update the metadata of a specified single or multiple roles.		
Type	POST		
URL Format	`..../admin/roles`		
Java API	**Method Name**		**Class Name**
	`updateRoles`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)

	roleinfolist=<xml/>		Object containing the metadata of the roles to be updated.
Returns	Returns a roleInfoList object containing a list of all the successfully updated roles.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/admin/roles?op=update&roleinfolist=
	`<roleInfoList>`
	`<roleInfo id="11" name="Updated Name of Role">`
	`<applicationPrivList>`
	`<privilege id="190000017">`
	`<name>Remove Link with Collection Template</name>`
	`</privilege>`
	`<privilege id="999991">`
	`<name>Live Preview</name>`
	`</privilege> .... </applicationPrivList>`
	`<contentTypePrivList>`
	`<contentTypePriv contentTypeId="1100" contentType="Asset" contentTypeHierarchy="System;Asset;">`
	`<privilege id="10000015">`
	`<name>Get Original Assets</name>`
	`</privilege> ..... </contentTypePriv> ..... </contentTypePrivList>`
	`</roleInfo>`
	`</roleInfoList>`
	Result:
	`<roleInfoList>`
	`<roleInfo id="11" name="Updated Name of Role">`
	`<applicationPrivList>`
	`<privilege id="190000017">`
	`<name>Remove Link with Collection Template</name>`
	`</privilege>`
	`<privilege id="999991">`
	`<name>Live Preview</name>`
	`</privilege> ..... </applicationPrivList>`
	`<contentTypePrivList>`
	`<contentTypePriv contentTypeId="1100" contentType="Asset" contentTypeHierarchy="System;Asset;">`
	`<privilege id="10000015">`
	`<name>Get Original Assets</name>`
	`</privilege> ..... </contentTypePriv> ..... </contentTypePrivList>`
	`</roleInfo>`
	`</roleInfoList>`
	Example with multiple roles
	URL:
	http://serverName:serverPort/rest/service/admin/roles?op=update&roleinfolist=
	`<roleInfoList>`
	`<roleInfo id="11" name="Updated Name of Role 1">`
	`<applicationPrivList>`
	`<privilege id="190000017">`
	`<name>Remove Link with Collection Template</name>`
	`</privilege>`
	`<privilege id="999991">`
	`<name>Live Preview</name>`
	`</privilege> ..... </applicationPrivList>`		

```
        <contentTypePrivList>
            <contentTypePriv contentTypeId="1100"
contentType="Asset" contentTypeHierarchy="System;Asset;">
                <privilege id="10000015">
                    <name>Get Original Assets</name>
                </privilege> ...... </contentTypePriv> .......
</contentTypePrivList>
    </roleInfo>
    <roleInfo id="12" name="Updated Name of Role 2">
        <applicationPrivList>
            <privilege id="190000017">
                <name>Remove Link with Collection Template</name>
            </privilege>
            <privilege id="999991">
                <name>Live Preview</name>
            </privilege> ...... </applicationPrivList>
        <contentTypePrivList>
            <contentTypePriv contentTypeId="1100"
contentType="Asset" contentTypeHierarchy="System;Asset;">
                <privilege id="10000015">
                    <name>Get Original Assets</name>
                </privilege> ...... </contentTypePriv> .......
</contentTypePrivList>
    </roleInfo>
</roleInfoList>
```

Result:

```
<roleInfoList>
    <roleInfo id="11" name="Updated Name of Role 1">
        <applicationPrivList>
            <privilege id="190000017">
                <name>Remove Link with Collection Template</name>
            </privilege>
            <privilege id="999991">
                <name>Live Preview</name>
            </privilege> ...... </applicationPrivList>
        <contentTypePrivList>
            <contentTypePriv contentTypeId="1100"
contentType="Asset" contentTypeHierarchy="System;Asset;">
                <privilege id="10000015">
                    <name>Get Original Assets</name>
                </privilege> ...... </contentTypePriv> .......
</contentTypePrivList>
    </roleInfo>
    <roleInfo id="12" name="Updated Name of Role 2">
        <applicationPrivList>
            <privilege id="190000017">
                <name>Remove Link with Collection Template</name>
            </privilege>
            <privilege id="999991">
                <name>Live Preview</name>
            </privilege> ...... </applicationPrivList>
        <contentTypePrivList>
            <contentTypePriv contentTypeId="1100"
contentType="Asset" contentTypeHierarchy="System;Asset;">
                <privilege id="10000015">
                    <name>Get Original Assets</name>
                </privilege> ...... </contentTypePriv> .......
</contentTypePrivList>
    </roleInfo>
</roleInfoList>
```

deleteRole

Function	Delete a role.		
Description	This request is used to delete the specified role.		
Type	POST		
URL Format	.../admin/roles/{roleIdorName}		
Java API	**Method Name**		**Class Name**
	deleteRole		com.quark.qpp.service.facade Class AdminFacade
Parameters	**Name**	**Type**	**Description**
	op=delete	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	roleIdorName	string	Id or Name of the role to be deleted.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/roles/23?op=delete`		

createRelationType

Function	Create Relation Types.		
Description	This request is used to create a single or multiple Relation Types at one time.		
Type	POST		
URL Format	`.../admin/relationtypes`		
Java API	**Method Name**		**Class Name**
	`createRelationTypes`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`relationtypeinfolist`	string	List of relation types to create.
Returns	Returns a `relationTypeInfoList` object containing a list of successfully created relation types.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/relationtypes?op=create&relationtypeinfolist=`
	`<relationTypeInfoList>`
	`  <relationTypeInfo>`
	`  <name>Private Relations</name>`
	`  <systemDefined>false</systemDefined>`
	`  <checkOutMandatory>true</checkOutMandatory>`		

```
        <relationAttributeList>
            <relationAttribute valueMandatory="false"
userMapped="true" id="508"> Master DT </relationAttribute>
            <relationAttribute valueMandatory="false"
userMapped="true" id="509"> Master List </relationAttribute>
........... </relationAttributeList>
    </relationTypeInfo>
</relationTypeInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<relationTypeInfoList>
    <relationTypeInfo id="1010">
        <name>Private Relations</name>
        <systemDefined>false</systemDefined>
        <checkOutMandatory>true</checkOutMandatory>
        <relationAttributeList>
            <relationAttribute valueMandatory="false"
userMapped="true" id="508"> Master DT </relationAttribute>
            <relationAttribute valueMandatory="false"
userMapped="true" id="509"> Master List </relationAttribute>
........... </relationAttributeList>
    </relationTypeInfo>
</relationTypeInfoList>
```

Example with multiple relation types

URL:

```
http://serverName:serverPort/rest/service/admin/relationtypes?op=
create&relationtypeinfolist=
```

```
<relationTypeInfoList>
    <relationTypeInfo>
        <name>InDesign 1</name>
        <systemDefined>false</systemDefined>
        <checkOutMandatory>true</checkOutMandatory>
        <relationAttributeList>
            <relationAttribute valueMandatory="false"
userMapped="false" id="410"> Item Name </relationAttribute>
            <relationAttribute valueMandatory="true"
userMapped="false" id="419"> InDesign Object UID </relationAt-
tribute> ........... </relationAttributeList>
    </relationTypeInfo>
    <relationTypeInfo>
        <name>InDesign 2</name>
        <systemDefined>false</systemDefined>
        <checkOutMandatory>true</checkOutMandatory>
        <relationAttributeList>
            <relationAttribute valueMandatory="false"
userMapped="false" id="410"> Item Name </relationAttribute>
            <relationAttribute valueMandatory="true"
userMapped="false" id="419"> InDesign Object UID </relationAt-
tribute> ........... </relationAttributeList>
    </relationTypeInfo>
</relationTypeInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<relationTypeInfoList>
    <relationTypeInfo id="1012">
        <name>InDesign 1</name>
        <systemDefined>false</systemDefined>
        <checkOutMandatory>true</checkOutMandatory>
        <relationAttributeList>
```

```
            <relationAttribute valueMandatory="false"
userMapped="true" id="410"> Item Name </relationAttribute>
            <relationAttribute valueMandatory="true"
userMapped="true" id="419"> InDesign Object UID </relationAt-
tribute> ........... </relationAttributeList>
    </relationTypeInfo>
    <relationTypeInfo id="1013">
        <name>InDesign 2</name>
        <systemDefined>false</systemDefined>
        <checkOutMandatory>true</checkOutMandatory>
        <relationAttributeList>
            <relationAttribute valueMandatory="false"
userMapped="true" id="410"> Item Name </relationAttribute>
            <relationAttribute valueMandatory="true"
userMapped="true" id="419"> InDesign Object UID </relationAt-
tribute> ........... </relationAttributeList>
    </relationTypeInfo>
</relationTypeInfoList>
```

getRelationType

Function	Get Relation Type.		
Description	This request is used to get the metadata of a specified relation type.		
Type	GET		
URL Format	`.../admin/relationtypes/{relationTypeIdOrName}`		
Java API	**Method Name**		**Class Name**
	`getRelationType`		`com.quark.qpp.service.facade` `Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`relationTypeIdOrName`	string	The ID or name of the relation type that is to be retrieved.
Returns	Returns a `relationTypeInfoList` object containing the metadata of the specified relation type.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/relationtypes/1` **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<relationTypeInfoList>
    <relationTypeInfo id="1">
        <name>Primary Attachment</name>
        <systemDefined>true</systemDefined>
        <checkOutMandatory>true</checkOutMandatory>
        <relationAttributeList>
            <relationAttribute valueMandatory="false"
userMapped="false" id="416"> Article Component Reference Asset
Minor version </relationAttribute>
            <relationAttribute valueMandatory="false"
userMapped="false" id="415"> Article Component Reference Asset
Major Version </relationAttribute> ........... </relationAt-
tributeList>
    </relationTypeInfo>
</relationTypeInfoList>
```

getAllRelationTypes

Function	Get all relationship types.	
Description	This request returns all the relation types with their attributes.	
Type	GET	
URL Format	`.../admin/relationtypes`	
Java API	**Method Name**	**Class Name**
	`getAllRelationTypes`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `relationTypeInfoList` object containing an array of all relation types that exist on the QPP Server.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/relationtypes`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<relationTypeInfoList>`
	`  <relationTypeInfo id="1">`
	`  <name>Primary Attachment</name>`
	`  <systemDefined>true</systemDefined>`
	`  <checkOutMandatory>true</checkOutMandatory>`
	`  <relationAttributeList>`
	`  <relationAttribute valueMandatory="false" userMapped="false" id="416"> Article Component Reference Asset Minor version</relationAttribute>`
	`  <relationAttribute valueMandatory="false" userMapped="false" id="414"> Article Component Reference Asset Id</relationAttribute> ..... </relationAttributeList>`
	`  </relationTypeInfo>`
	`  <relationTypeInfo id="2">`
	`  <name>Secondary Attachment</name>`
	`  <systemDefined>true</systemDefined>`
	`  <checkOutMandatory>true</checkOutMandatory>`
	`  <relationAttributeList>`
	`  <relationAttribute valueMandatory="false" userMapped="false" id="415"> Article Component Reference Asset Major Version</relationAttribute>`
	`  <relationAttribute valueMandatory="false" userMapped="false" id="416"> Article Component Reference Asset Minor version</relationAttribute>`
	`  <relationAttribute valueMandatory="false" userMapped="false" id="413"> Article Id</relationAttribute> ..... </relationAttributeList>`
	`  </relationTypeInfo> .... </relationTypeInfoList>`	

updateRelationTypes

Function	Update relation types.
Description	This request is used to update the metadata of user defined relation types.
Type	POST
URL Format	`.../admin/relationtypes`

Java API	**Method Name**		**Class Name**
	`updateRelationTypes`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`relationtypeinfolist`	string	List of relation types to be updated.
Returns	Returns a `relationTypeInfoList` object containing a list of successfully updated relation types.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/relationtypes?op=update&relationtypeinfolist=`
	`<relationTypeInfoList>`
	`  <relationTypeInfo id="1000">`
	`  <name>Custom Relations Updated</name>`
	`  <systemDefined>false</systemDefined>`
	`  <checkOutMandatory>false</checkOutMandatory>`
	`  <relationAttributeList>`
	`  <relationAttribute valueMandatory="false" userMapped="true" id="507"> Release Points </relationAttribute>`
	`  <relationAttribute valueMandatory="false" userMapped="true" id="506"> Release Inches Decimal </relationAttribute> ..... </relationAttributeList>`
	`  </relationTypeInfo>`
	`</relationTypeInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<relationTypeInfoList>`
	`  <relationTypeInfo id="1000">`
	`  <name>Custom Relations Updated</name>`
	`  <systemDefined>false</systemDefined>`
	`  <checkOutMandatory>false</checkOutMandatory>`
	`  <relationAttributeList>`
	`  <relationAttribute valueMandatory="false" userMapped="true" id="507"> Release Points </relationAttribute>`
	`  <relationAttribute valueMandatory="false" userMapped="true" id="506"> Release Inches Decimal </relationAttribute> ..... </relationAttributeList>`
	`  </relationTypeInfo>`
	`</relationTypeInfoList>`
	Example with multiple relation types to update
	URL:
	`http://serverName:serverPort/rest/service/admin/relationtypes?op=update&relationtypeinfolist=`
	`<relationTypeInfoList>`
	`  <relationTypeInfo>`
	`  <name>demo relation 1</name>`
	`  <checkOutMandatory>true</checkOutMandatory>`
	`  </relationTypeInfo>`
	`  <relationTypeInfo>`
	`  <name>demo relation 2</name>`
	`  <checkOutMandatory>true</checkOutMandatory>`
	`  </relationTypeInfo> ..... </relationTypeInfoList>`		

	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<relationTypeInfoList>`
	`  <relationTypeInfo id="1011">`
	`  <name>demo relation 1</name>`
	`  <systemDefined>false</systemDefined>`
	`  <checkOutMandatory>true</checkOutMandatory>`
	`  </relationTypeInfo>`
	`  <relationTypeInfo id="1012">`
	`  <name>demo relation 2</name>`
	`  <systemDefined>false</systemDefined>`
	`  <checkOutMandatory>true</checkOutMandatory>`
	`  </relationTypeInfo> ..... </relationTypeInfoList>`

deleteRelationType

Function	Delete a Relation Type.		
Description	This request is used to delete a relation type with the specified id or name.		
Type	POST		
URL Format	`.../admin/relationtypes/{relationTypeIdOrName}`		
Java API	**Method Name**		**Class Name**
	`deleteRelationType`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`relationTypeORName`	long or string	List of forms to be updated.The Name or ID of the relation type to be deleted.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/relationtypes/1000?op=delete`		

createDomains

Function	Create Attribute Domains.		
Description	This request is used to create a single or multiple attribute domains at one time.		
Type	POST		
URL Format	`.../admin/domains`		
Java API	**Method Name**		**Class Name**
	`createDomains`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**

	op=create	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	attributedomaininfolist	string	List of attribute domains to create.
Returns	Returns a attributeDomainInfoList object containing a list of successfully created attribute domains.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/admin/domains?op=create&attributedomaininfolist=
	`<attributeDomainInfoList>`
	`  <attributeDomainInfo>`
	`  <name>Domain 1</name>`
	`  <systemDefined>false</systemDefined>`
	`  <userModifiable>true</userModifiable>`
	`  <itemsSorted>true</itemsSorted>`
	`  <expandable>true</expandable>`
	`  <warnBeforeExpansion>false</warnBeforeExpansion>`
	`  <domainValueList>`
	`  <nameId>`
	`  <name>Value 1</name>`
	`  </nameId>`
	`  </domainValueList>`
	`  </attributeDomainInfo>`
	`</attributeDomainInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<attributeDomainInfoList>`
	`  <attributeDomainInfo id="504">`
	`  <name>Domain 1</name>`
	`  <systemDefined>false</systemDefined>`
	`  <userModifiable>true</userModifiable>`
	`  <itemsSorted>true</itemsSorted>`
	`  <expandable>true</expandable>`
	`  <warnBeforeExpansion>false</warnBeforeExpansion>`
	`  <domainValueList>`
	`  <nameId id="1">`
	`  <name>Value 1</name>`
	`  </nameId>`
	`  </domainValueList>`
	`  </attributeDomainInfo>`
	`</attributeDomainInfoList>`
	Example with multiple domains
	URL:
	http://serverName:serverPort/rest/service/admin/domains?op=create&attributedomaininfolist=
	`<attributeDomainInfoList>`
	`  <attributeDomainInfo>`
	`  <name>Domain 1</name>`
	`  <systemDefined>false</systemDefined>`
	`  <userModifiable>true</userModifiable>`
	`  <itemsSorted>true</itemsSorted>`
	`  <expandable>true</expandable>`
	`  <warnBeforeExpansion>false</warnBeforeExpansion>`
	`  <domainValueList>`
	`  <nameId>`		

```
                <name>Value 1</name>
            </nameId>
        </domainValueList>
    </attributeDomainInfo>
    <attributeDomainInfo>
        <name>Domain 2</name>
        <systemDefined>false</systemDefined>
        <userModifiable>true</userModifiable>
        <itemsSorted>true</itemsSorted>
        <expandable>true</expandable>
        <warnBeforeExpansion>true</warnBeforeExpansion>
        <domainValueList>
            <nameId>
                <name>Value 1</name>
            </nameId>
        </domainValueList>
    </attributeDomainInfo>
</attributeDomainInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<attributeDomainInfoList>
    <attributeDomainInfo id="502">
        <name>Domain 1</name>
        <systemDefined>false</systemDefined>
        <userModifiable>true</userModifiable>
        <itemsSorted>true</itemsSorted>
        <expandable>true</expandable>
        <warnBeforeExpansion>false</warnBeforeExpansion>
        <domainValueList>
            <nameId id="1">
                <name>Value 1</name>
            </nameId>
        </domainValueList>
    </attributeDomainInfo>
    <attributeDomainInfo id="503">
        <name>Domain 2</name>
        <systemDefined>false</systemDefined>
        <userModifiable>true</userModifiable>
        <itemsSorted>true</itemsSorted>
        <expandable>true</expandable>
        <warnBeforeExpansion>true</warnBeforeExpansion>
        <domainValueList>
            <nameId id="2">
                <name>Value 1</name>
            </nameId>
        </domainValueList>
    </attributeDomainInfo>
</attributeDomainInfoList>
```

getDomain

Function	Get Attribute Domain.	
Description	This request is used to get the metadata of a specified attribute domain.	
Type	GET	
URL Format	`....admin/domains/{domainIdOrName}`	
Java API	**Method Name**	**Class Name**
	`getDomain`	`com.quark.qpp.service.facade`

			Class AdminFacade
Parameters	Name	Type	Description
	domainTypeIdOrName	string	The ID or name of the domain that is to be retrieved.
Returns	Returns an attributeDomainInfoList object containing the metadata of the specified domain.		
Examples	URL: http://serverName:serverPort/rest/service/admin/domains/3 Result: <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <attributeDomainInfoList> <attributeDomainInfo id="3"> <name>Users</name> <systemDefined>true</systemDefined> <userModifiable>false</userModifiable> <itemsSorted>true</itemsSorted> <expandable>false</expandable> <warnBeforeExpansion>false</warnBeforeExpansion> <domainValueList> <nameId id="1"> <name>Admin</name> </nameId> <valueHierarchy>Desktop</valueHierarchy> ..... </domainValueList> </attributeDomainInfo> ..... </attributeDomainInfoList>		

getAllDomains

Function	Get all attribute domains.	
Description	This request is used to get the metadata of all the domains that exist in the system.	
Type	GET	
URL Format	.../admin/domains	
Java API	Method Name	Class Name
	getAllDomains	com.quark.qpp.service.facade Class AdminFacade
Parameters	This request requires no input parameters.	
Returns	Returns an attributeDomainInfoList object containing the metadata of the all the attribute domains.	
Examples	URL: http://serverName:serverPort/rest/service/admin/domains Result: <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <attributeDomainInfoList> <attributeDomainInfo id="3"> <name>Users</name> <systemDefined>true</systemDefined> <userModifiable>false</userModifiable> <itemsSorted>true</itemsSorted>	

```
        <expandable>false</expandable>
        <warnBeforeExpansion>false</warnBeforeExpansion>
        <domainValueList>
            <nameId id="1">
                <name>Admin</name>
            </nameId>
            <valueHierarchy>Desktop</valueHierarchy> ...........
</domainValueList>
    </attributeDomainInfo> ....... </attributeDomainInfoList>
```

updateDomains

Function	Update attribute domains.		
Description	This request is used to update the metadata of a single or multiple attribute domains.		
Type	POST		
URL Format	`.../admin/domains`		
Java API	**Method Name**		**Class Name**
	`updateDomains`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`attributedomaininfolist`	string	List of domains to be updated.
Returns	Returns an attributeDomainInfoList object containing a list of successfully updated domains.		
Examples	**URL:**		

```
<attributeDomainInfoList>
    <attributeDomainInfo id="503">
        <name>Domain 1</name>
        <systemDefined>false</systemDefined>
        <userModifiable>true</userModifiable>
        <itemsSorted>false</itemsSorted>
        <expandable>false</expandable>
        <warnBeforeExpansion>false</warnBeforeExpansion>
        <domainValueList>
            <nameId id="1">
                <name>Value 1</name>
            </nameId>
        </domainValueList>
    </attributeDomainInfo>
</attributeDomainInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<attributeDomainInfoList>
    <attributeDomainInfo id="503">
        <name>Domain 1</name>
        <systemDefined>false</systemDefined>
        <userModifiable>true</userModifiable>
        <itemsSorted>false</itemsSorted>
```

```
        <expandable>false</expandable>
        <warnBeforeExpansion>false</warnBeforeExpansion>
        <domainValueList>
            <nameId id="1">
                <name>Value 1</name>
            </nameId>
        </domainValueList>
    </attributeDomainInfo>
</attributeDomainInfoList>
```

Example with multiple attribute domains to update

URL:

```
http://serverName:serverPort/rest/service/admin/domains?op=update
&attributedomaininfolist=
```

```
<attributeDomainInfoList>
    <attributeDomainInfo id="502">
        <name>Domain 1</name>
        <systemDefined>false</systemDefined>
        <userModifiable>true</userModifiable>
        <itemsSorted>true</itemsSorted>
        <expandable>true</expandable>
        <warnBeforeExpansion>false</warnBeforeExpansion>
        <domainValueList>
            <nameId id="1">
                <name>Value 1</name>
            </nameId>
        </domainValueList>
    </attributeDomainInfo>
    <attributeDomainInfo id="503">
        <name>Domain 2</name>
        <systemDefined>false</systemDefined>
        <userModifiable>true</userModifiable>
        <itemsSorted>true</itemsSorted>
        <expandable>true</expandable>
        <warnBeforeExpansion>true</warnBeforeExpansion>
        <domainValueList>
            <nameId id="2">
                <name>Value 1</name>
            </nameId>
        </domainValueList>
    </attributeDomainInfo>
</attributeDomainInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<attributeDomainInfoList>
    <attributeDomainInfo id="502">
        <name>Domain 1</name>
        <systemDefined>false</systemDefined>
        <userModifiable>true</userModifiable>
        <itemsSorted>true</itemsSorted>
        <expandable>true</expandable>
        <warnBeforeExpansion>false</warnBeforeExpansion>
        <domainValueList>
            <nameId id="1">
                <name>Value 1</name>
            </nameId>
        </domainValueList>
    </attributeDomainInfo>
    <attributeDomainInfo id="503">
        <name>Domain 2</name>
        <systemDefined>false</systemDefined>
        <userModifiable>true</userModifiable>
```

	`<itemsSorted>true</itemsSorted>`
	`<expandable>true</expandable>`
	`<warnBeforeExpansion>true</warnBeforeExpansion>`
	`<domainValueList>`
	`<nameId id="2">`
	`<name>Value 1</name>`
	`</nameId>`
	`</domainValueList>`
	`</attributeDomainInfo>`
	`</attributeDomainInfoList>`

deleteDomain

Function	Delete an Attribute Domain.		
Description	This request is used to delete an attribute domain with the specified id or name.		
Type	POST		
URL Format	`..../admin/domains/{domainIdorName}`		
Java API	**Method Name**		**Class Name**
	`deleteDomain`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`domainIdORName`	long or string	The Name or ID of the domain to be deleted.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/domains/503?op=delete`		

createLdapProfiles

Function	Create LDAP profiles.		
Description	This request is used to create one or more LDAP profiles.		
Type	POST		
URL Format	`.../admin/ldapprofiles`		
Java API	**Method Name**		**Class Name**
	`createLdapProfiles`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)

	`ldapprofileinfolist =<xml/>`		Object containing a list of the LDAP profiles to be created.
Returns	Returns a `ldapProfileInfoList` object containing containing the successfully created LDAp profiles.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/ldapprofiles?op=create&ldapprofileinfolist= <?xml version="1.0" encoding="UTF-8" standalone="yes"?>` **Result:**		

```
<ldapProfileInfoList>
    <ldapProfileInfo>
        <name>QPP LDAP Profile</name>
        <realm>INDIA.QUARK.COM</realm>
        <userName>qppadmin@quark.com</userName>
        <authenticationType>KERBEROS</authenticationType>
        <ldapServerInfoList>
            <ldapServerInfo>
                <scheme>LDAP</scheme>
                <serverName>india-dc1.india.quark.com</server-
Name>
                <port>389</port>
            </ldapServerInfo>
        </ldapServerInfoList>
        <ldapObjectsIdentificationCriteria>
            <userIdentificationCriteria> ... </userIdentifica-
tionCriteria>
            <groupIdentificationCriteria> ... </groupIdentifica-
tionCriteria>
            <groupMemberAttrName>member</groupMemberAttrName>
            <membershipIdentifyingAttrName>memberof</membershipI-
dentifyingAttrName>
            <baseDn>BDN</baseDn>
            <filterToGetAllUsersOfGroup> ... </filterToGetAl-
lUsersOfGroup>
            <filterToGetAllSubGroupsOfGroup> ... </filterToGe-
tAllSubGroupsOfGroup>
            <filterToGetAllGroupsOfUser> ... </filterToGetAll-
GroupsOfUser>
        </ldapObjectsIdentificationCriteria>
        <ldapAttributeInfoList>
            <ldapAttributeInfo id="5">
                <name>USER_EMAIL</name>
                <value>mail</value>
            </ldapAttributeInfo> ......... </ldapAttributeIn-
foList>
    </ldapProfileInfo>
</ldapProfileInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ldapProfileInfoList>
    <ldapProfileInfo id="10">
        <name>QPP LDAP Profile</name>
        <realm>INDIA.QUARK.COM</realm>
        <userName>qppadmin@quark.com</userName>
        <authenticationType>KERBEROS</authenticationType>
        <ldapServerInfoList>
            <ldapServerInfo>
                <scheme>LDAP</scheme>
                <serverName>india-dc1.india.quark.com</server-
Name>
                <port>389</port>
```

```
            </ldapServerInfo>
        </ldapServerInfoList>
        <ldapObjectsIdentificationCriteria>
            <userIdentificationCriteria> ... </userIdentifica-
tionCriteria>
            <groupIdentificationCriteria> ... </groupIdentifica-
tionCriteria>
            <groupMemberAttrName>member</groupMemberAttrName>
            <membershipIdentifyingAttrName>memberof</membershipI-
dentifyingAttrName>
            <baseDn>BDN</baseDn>
            <filterToGetAllUsersOfGroup> ... </filterToGetAl-
lUsersOfGroup>
            <filterToGetAllSubGroupsOfGroup> ... </filterToGe-
tAllSubGroupsOfGroup>
            <filterToGetAllGroupsOfUser> ... </filterToGetAll-
GroupsOfUser>
        </ldapObjectsIdentificationCriteria>
        <ldapAttributeInfoList>
            <ldapAttributeInfo id="5">
                <name>USER_EMAIL</name>
                <value>mail</value>
            </ldapAttributeInfo> ......... </ldapAttributeIn-
foList>
    </ldapProfileInfo>
</ldapProfileInfoList>
```

Example with multiple LDAP profiles

URL:

```
http://serverName:serverPort/rest/service/admin/ldapprofiles?op=c
reate&ldapprofileinfolist= <ldapProfileInfoList> <?xml
version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<ldapProfileInfoList>
    <ldapProfileInfo>
        <name>QPP LDAP Profile India</name>
        <realm>INDIA.QUARK.COM</realm>
        <userName>qppadmin@quark.com</userName>
        <authenticationType>KERBEROS</authenticationType>
        <ldapServerInfoList>
            <ldapServerInfo>
                <scheme>LDAP</scheme>
                <serverName>india-dc1.india.quark.com</server-
Name>
                <port>389</port>
            </ldapServerInfo>
        </ldapServerInfoList>
        <ldapObjectsIdentificationCriteria>
            <userIdentificationCriteria> ... </userIdentifica-
tionCriteria>
            <groupIdentificationCriteria> ... </groupIdentifica-
tionCriteria>
            <groupMemberAttrName>member</groupMemberAttrName>
            <membershipIdentifyingAttrName>memberof</membershipI-
dentifyingAttrName>
            <baseDn>BDN</baseDn>
            <filterToGetAllUsersOfGroup> ... </filterToGetAl-
lUsersOfGroup>
            <filterToGetAllSubGroupsOfGroup> ... </filterToGe-
tAllSubGroupsOfGroup>
            <filterToGetAllGroupsOfUser> ... </filterToGetAll-
GroupsOfUser>
        </ldapObjectsIdentificationCriteria>
```

```
        <ldapAttributeInfoList>
            <ldapAttributeInfo id="5">
                <name>USER_EMAIL</name>
                <value>mail</value>
            </ldapAttributeInfo> ......... </ldapAttributeIn-
foList>
    </ldapProfileInfo>
    <ldapProfileInfo>
        <name>QPP LDAP Profile US</name>
        <realm>INDIA.QUARK.COM</realm>
        <userName>qppadmin@quark.com</userName>
        <authenticationType>KERBEROS</authenticationType>
        <ldapServerInfoList>
            <ldapServerInfo>
                <scheme>LDAP</scheme>
                <serverName>india-dc1.india.quark.com</server-
Name>
                <port>389</port>
            </ldapServerInfo>
        </ldapServerInfoList>
        <ldapObjectsIdentificationCriteria>
            <userIdentificationCriteria> ... </userIdentifica-
tionCriteria>
            <groupIdentificationCriteria> ... </groupIdentifica-
tionCriteria>
            <groupMemberAttrName>member</groupMemberAttrName>
            <membershipIdentifyingAttrName>memberof</membershipI-
dentifyingAttrName>
            <baseDn>BDN</baseDn>
            <filterToGetAllUsersOfGroup> ... </filterToGetAl-
lUsersOfGroup>
            <filterToGetAllSubGroupsOfGroup> ... </filterToGe-
tAllSubGroupsOfGroup>
            <filterToGetAllGroupsOfUser> ... </filterToGetAll-
GroupsOfUser>
        </ldapObjectsIdentificationCriteria>
        <ldapAttributeInfoList>
            <ldapAttributeInfo id="5">
                <name>USER_EMAIL</name>
                <value>mail</value>
            </ldapAttributeInfo> ......... </ldapAttributeIn-
foList>
    </ldapProfileInfo>
</ldapProfileInfoList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ldapProfileInfoList>
    <ldapProfileInfo id="10">
        <name>QPP LDAP Profile India</name>
        <realm>INDIA.QUARK.COM</realm>
        <userName>qppadmin@quark.com</userName>
        <authenticationType>KERBEROS</authenticationType>
        <ldapServerInfoList>
            <ldapServerInfo>
                <scheme>LDAP</scheme>
                <serverName>india-dc1.india.quark.com</server-
Name>
                <port>389</port>
            </ldapServerInfo>
        </ldapServerInfoList>
        <ldapObjectsIdentificationCriteria>
            <userIdentificationCriteria> ... </userIdentifica-
tionCriteria>
```

```
            <groupIdentificationCriteria> ... </groupIdentifica-
tionCriteria>
            <groupMemberAttrName>member</groupMemberAttrName>
            <membershipIdentifyingAttrName>memberof</membershipI-
dentifyingAttrName>
            <baseDn>BDN</baseDn>
            <filterToGetAllUsersOfGroup> ... </filterToGetAl-
lUsersOfGroup>
            <filterToGetAllSubGroupsOfGroup> ... </filterToGe-
tAllSubGroupsOfGroup>
            <filterToGetAllGroupsOfUser> ... </filterToGetAll-
GroupsOfUser>
        </ldapObjectsIdentificationCriteria>
        <ldapAttributeInfoList>
            <ldapAttributeInfo id="5">
                <name>USER_EMAIL</name>
                <value>mail</value>
            </ldapAttributeInfo> ......... </ldapAttributeIn-
foList>
    </ldapProfileInfo>
    <ldapProfileInfo id="11">
        <name>QPP LDAP Profile US</name>
        <realm>INDIA.QUARK.COM</realm>
        <userName>qppadmin@quark.com</userName>
        <authenticationType>KERBEROS</authenticationType>
        <ldapServerInfoList>
            <ldapServerInfo>
                <scheme>LDAP</scheme>
                <serverName>india-dc1.india.quark.com</server-
Name>
                <port>389</port>
            </ldapServerInfo>
        </ldapServerInfoList>
        <ldapObjectsIdentificationCriteria>
            <userIdentificationCriteria> ... </userIdentifica-
tionCriteria>
            <groupIdentificationCriteria> ... </groupIdentifica-
tionCriteria>
            <groupMemberAttrName>member</groupMemberAttrName>
            <membershipIdentifyingAttrName>memberof</membershipI-
dentifyingAttrName>
            <baseDn>BDN</baseDn>
            <filterToGetAllUsersOfGroup> ... </filterToGetAl-
lUsersOfGroup>
            <filterToGetAllSubGroupsOfGroup> ... </filterToGe-
tAllSubGroupsOfGroup>
            <filterToGetAllGroupsOfUser> ... </filterToGetAll-
GroupsOfUser>
        </ldapObjectsIdentificationCriteria>
        <ldapAttributeInfoList>
            <ldapAttributeInfo id="5">
                <name>USER_EMAIL</name>
                <value>mail</value>
            </ldapAttributeInfo> ......... </ldapAttributeIn-
foList>
    </ldapProfileInfo>
</ldapProfileInfoList>
```

getLdapProfile

Function	Get an LDAP profile.

Description	This request returns the metadata of a specified LDAP profile.		
Type	Get		
URL Format	`.../admin/ldapprofiles/{ldapProfileIdOrName}`		
Java API	**Method Name**		**Class Name**
	`getLdapProfile`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`ldapProfileIdOrName`	string	The ID or Name of the LDAP profile to be retrieved.
Returns	Returns a `ldapProfileInfoList` object containing information on the specified LDAP profile.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/ldapprofiles/10` **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ldapProfileInfoList>
    <ldapProfileInfo id="10">
        <name>QPP LDAP Profile</name>
        <realm>INDIA.QUARK.COM</realm>
        <userName>qppadmin@quark.com</userName>
        <authenticationType>KERBEROS</authenticationType>
        <ldapServerInfoList>
            <ldapServerInfo>
                <scheme>LDAP</scheme>
                <serverName>india-dc1.india.quark.com</server-
Name>
                <port>389</port>
            </ldapServerInfo>
        </ldapServerInfoList>
        <ldapObjectsIdentificationCriteria>
            <userIdentificationCriteria> ... </userIdentifica-
tionCriteria>
            <groupIdentificationCriteria> ... </groupIdentifica-
tionCriteria>
            <groupMemberAttrName>member</groupMemberAttrName>
            <membershipIdentifyingAttrName>memberof</membershipI-
dentifyingAttrName>
            <baseDn>BDN</baseDn>
            <filterToGetAllUsersOfGroup> ... </filterToGetAl-
lUsersOfGroup>
            <filterToGetAllSubGroupsOfGroup> ... </filterToGe-
tAllSubGroupsOfGroup>
            <filterToGetAllGroupsOfUser> ... </filterToGetAll-
GroupsOfUser>
        </ldapObjectsIdentificationCriteria>
        <ldapAttributeInfoList>
            <ldapAttributeInfo id="5">
                <name>USER_EMAIL</name>
                <value>mail</value>
            </ldapAttributeInfo> ......... </ldapAttributeIn-
foList>
    </ldapProfileInfo>
</ldapProfileInfoList>
```

getAllLdapProfiles

Function	Get all LDAP profiles.	
Description	This request returns all the LDAP profiles saved on the system.	
Type	GET	
URL Format	`.../admin/ldapprofiles`	
Java API	**Method Name**	**Class Name**
	`getAllLdapProfiles`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters	
Returns	Returns a `ldapProfileInfoList` object containing a list of all the LDAP profiles that exist on the QPP Server.	

Examples

URL:

`http://serverName:serverPort/rest/service/admin/workflows`

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ldapProfileInfoList>
    <ldapProfileInfo id="10">
        <name>QPP LDAP Profile</name>
        <realm>INDIA.QUARK.COM</realm>
        <userName>qppadmin@quark.com</userName>
        <authenticationType>KERBEROS</authenticationType>
        <ldapServerInfoList>
            <ldapServerInfo>
                <scheme>LDAP</scheme>
                <serverName>india-dc1.india.quark.com</server-
Name>
                <port>389</port>
            </ldapServerInfo>
        </ldapServerInfoList>
        <ldapObjectsIdentificationCriteria>
            <userIdentificationCriteria> ... </userIdentifica-
tionCriteria>
            <groupIdentificationCriteria> ... </groupIdentifica-
tionCriteria>
            <groupMemberAttrName>member</groupMemberAttrName>
            <membershipIdentifyingAttrName>memberof</membershipI-
dentifyingAttrName>
            <baseDn>BDN</baseDn>
            <filterToGetAllUsersOfGroup> ... </filterToGetAl-
lUsersOfGroup>
            <filterToGetAllSubGroupsOfGroup> ... </filterToGe-
tAllSubGroupsOfGroup>
            <filterToGetAllGroupsOfUser> ... </filterToGetAll-
GroupsOfUser>
        </ldapObjectsIdentificationCriteria>
        <ldapAttributeInfoList>
            <ldapAttributeInfo id="5">
                <name>USER_EMAIL</name>
                <value>mail</value>
            </ldapAttributeInfo> ......... </ldapAttributeIn-
foList>
    </ldapProfileInfo> ....... </ldapProfileInfoList>
```

updateLdapProfiles

Function	Update LDAP profiles.		
Description	This request is used to update the metadata of a specified single or multiple LDAP profiles.		
Type	POST		
URL Format	`.../admin/ldapprofiles`		
Java API	**Method Name**		**Class Name**
	`updateLdapProfiles`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`ldapprofileinfolist=<xml/>`		Object containing a list of the successfully updated LDAP profiles.
Returns	Returns a `ldapProfileInfoList` object containing a list of all the successfully updated LDAP profiles.		

Examples

URL:

```
http://serverName:serverPort/rest/service/admin/workflows?op=update&workflowinfolist=
```

```
<workflowInfoList> <workflowInfo id="22"> <name>DefaultWorkflow New</name> </workflowInfo> </workflowInfoList>
```

Result:

```
http://serverName:serverPort/rest/service/admin/ldapprofiles?op=update&attributevaluelist=
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"? >
<ldapProfileInfoList>
    <ldapProfileInfo>
        <name>QPP LDAP Profile India</name>
        <realm>INDIA.QUARK.COM</realm>
        <userName>qppadmin@quark.com</userName>
        <authenticationType>KERBEROS</authenticationType>
        <ldapServerInfoList>
            <ldapServerInfo>
                <scheme>LDAP</scheme>
                <serverName>india-dc1.india.quark.com</server-
Name>
                <port>389</port>
            </ldapServerInfo>
        </ldapServerInfoList>
    </ldapProfileInfo>
    <ldapProfileInfo>
        <name>QPP LDAP Profile US</name>
        <realm>INDIA.QUARK.COM</realm>
        <userName>qppadmin@quark.com</userName>
        <authenticationType>KERBEROS</authenticationType>
        <ldapServerInfoList>
            <ldapServerInfo>
                <scheme>LDAP</scheme>
                <serverName>india-dc1.india.quark.com</server-
Name>
```

	`<port>389</port>`
	`</ldapServerInfo>`
	`</ldapServerInfoList>`
	`</ldapProfileInfo>`
	`</ldapProfileInfoList>`

deleteLdapProfile

Function	Delete an LDAP profile.		
Description	This request is used to delete the specified LDAP profile from the system. There shouldn't be any LDAP user in the system associated with this profile while executing this request.		
Type	POST		
URL Format	`.../admin/ldapprofiles/{profileId}`		
Java API	**Method Name**		**Class Name**
	`deleteLdapProfile`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`ldapProfileIdOrName`	string	Id or Name of the LDAP profile to be deleted.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/ldapprofiles/10?op=delete`		

getLdapUsers

Function	Get Ldap users.		
Description	This request returns LDAP users using a saved ldap query id/name or a combination of valid ldapprofile id/name, base string & search filter.		
Type	GET		
URL Format	`../admin/ldapusers`		
Java API	**Method Name**		**Class Name**
	`getLdapUsers`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`savedldapquery`	string	Specify the name of the LDAP Query. (Mandatory)
	`ldapProfile`	string	Specify the name of the LDAP profile. (Mandatory)

	baseString	string	Specify the base string to fetch LDAP users.(Mandatory)
	searchFilter	string	Specify the filter to filter the users. (Mandatory)
	defaultUserClass	string	The Quark Publishing Platform Server's UserRole. After fetching users from LDAP Server users will be created with this UserRole. (Mandatory)
	pagesize	string	
Returns	Returns a `userInfoList` object containing the metadata of the specified users.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/users?savedldapquery=LDAP Search&ldapProfile=QPP LDAP Profile&baseString=ou=users,ou=qmhi,dc=india,dc=quark,dc=com&searchFilter=sn=abc&defaultUserClass=UserClass2`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<userInfoList>`
	`  <userInfo id="52">`
	`  <name>abc</name>`
	`  <domainId>0</domainId>`
	`  <user>true</user> ..... <deleted>false</deleted>`
	`  <overriddenUserClassId>0</overriddenUserClassId>`
	`  </userInfo>`
	`  <userInfo id="53">`
	`  <name>abc</name>`
	`  <domainId>0</domainId> ..... </userInfo>`
	`</userInfoList>`		

importUsersFromLdap

Function	Import users from LDAP using LDAP profile.		
Description	This request is used to fetch users metadata from LDAP Server and push it to the Quark Publishing Platform Server.		
Type	POST		
URL Format	`.../admin/users`		
Java API	**Method Name**		**Class Name**
	`importUsersFromLdap`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=importusers`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`ldapProfile`		Specify the name of the LDAP profile. (Mandatory)
	`baseString`		Specify the base string to fetch LDAP

			users. (Mandatory)
	searchFilter		Specify the filter to filter the users. (Mandatory)
	defaultUserClass		The Quark Publishing Platform Server's UserRole. After fetching users from LDAP Server users will be created with this UserRole. (Mandatory)
Returns	Returns a userInfoList object containing metadata of newly fetched\created users in the form of an array.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/admin/users?op=importusers &ldapProfile=QPP LDAP Profile&baseString=ou=users,ou=qmhi,dc=india,dc=quark,dc=com&searchFilter=sn=abc&defaultUserClass=UserClass2
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<userInfoList>`
	`<userInfo id="52">`
	`<name>abc</name>`
	`<domainId>0</domainId>`
	`<user>true</user> ..... <deleted>false</deleted>`
	`<overriddenUserClassId>0</overriddenUserClassId>`
	`</userInfo>`
	`<userInfo id="53">`
	`<name>abc</name>`
	`<domainId>0</domainId> ..... </userInfo>`
	`</userInfoList>`		

Function	Import Users from LDAP using Saved Query		
Description	This request is used to fetch users metadata from the LDAP Server and pust it to the Quark Publishing Platform Server.		
Type	POST		
URL Format	.../admin/users		
Java API	**Method Name**		**Class Name**
	importUsersFromLdap		com.quark.qpp.service.facade Class AdminFacade
Parameters	**Name**	**Type**	**Description**
	op=importusers	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	savedLdapQuery		Specify the name of the saved LDAP Users search. (Mandatory)
Returns	Returns a userInfoList object containing metadata of newly fetched\created users in the form of an array.		
Examples	**URL:** http://serverName:serverPort/rest/service/admin/users?op=importusers &savedLdapQuery=SavedSearch		

	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<userInfoList>`
	`  <userInfo id="52">`
	`  <name>abc</name>`
	`  <domainId>0</domainId>`
	`  <user>true</user> ..... <deleted>false</deleted>`
	`  <overriddenUserClassId>0</overriddenUserClassId>`
	`  </userInfo>`
	`  <userInfo id="53">`
	`  <name>abc</name>`
	`  <domainId>0</domainId> ..... </userInfo>`
	`</userInfoList>`

getLdapGroups

Function	Get LDAP groups.		
Description	This request returns LDAP groups using a saved LDAP query or the combination of basestring, ldapprofile and searchfilter.		
Type	GET		
URL Format	`../admin/ldapgroups`		
Java API	**Method Name**		**Class Name**
	`getLdapGroups`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`savedldapquery=[savedLdapQueryId] or [Name]`	long	
	`ldapprofile=[ldapProfileId] or [Name]`	long	
	`basestring=[dc=xyz,dc=abc,dc=com]`	long	
	`searchfilter`	string	
Returns	Returns a list of LDAP groups on the basis of a saved LDAP query or the combination of LDAP profile, base string and searchfilter.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/ldapgroups?savedldapquery=MySearch`
	Result:
	`<groupInfoList>`
	`  <groupInfo id="0">`
	`  <name>QPP Server</name>`
	`  <emailAddress>QPP Server</emailAddress>`
	`  <nativeTrustee>false</nativeTrustee>`
	`  <defaultUserClassId>0</defaultUserClassId>`
	`  <ldapProfileId>764</ldapProfileId>`
	`  <ldapProfile>QPP LDAP Profile</ldapProfile>`		

	`<ldapDistinguishedName> CN=QPP Server, OU=Distribution-Lists, OU=QMHI, DC=india, DC=quark, DC=com </ldapDistinguished-Name>`
	`<memberList/>`
	`</groupInfo>`
	`</groupInfoList>`

importGroupsFromLdap

Function	Import LDAP groups.		
Description	LDAP groups can be imported into Platform Server along with all of its subgroups and users through the following two options: Option 1. Using LDAP profile, a base string and search filter to form the LDAP search criteria or Option2. On the basis of a saved LDAP query. The LDAP groups can be associated with a role by using the defaultuserclass request parameter while importing the groups. By default, if not mentioned, the Administrator role will be associated to all the imported groups and users.		
Type	POST		
URL Format	`../admin/groups`		
Java API	**Method Name**		**Class Name**
	`importGroupsFromLdap`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	op=importgroups	string	
	Option 1. Mention the following parameters (ldapprofile,basestring,searchfilter) to form LDAP search criteria:-		
	`ldapprofile=[ldapProfileId] or [ldapProfileName]`	long	name or id of the LDAP profile
	`basestring=[dc=xyz,dc=abc,dc=com]`	string	base DN under which groups are to be searched. e.g. basestring=dc=india,dc=myorg,dc=com
	`searchfilter=`	string	a string to specify the value of LDAP attributes on whose basis LDAP server will be searched. e.g. searchfilter=samaccountname=* searchfilter=cn=qpp*

	OR Option2. Mention a saved LDAP query id or name `savedldapquery=[savedLdapQueryId]or[savedLdapQueryname]`	long/string	name or id of the saved LDAP query from which groups & users are to be imported.
	`defaultuserclass=[userClassId]or[userClassName]`	long/string	Id or name of the role to be associated recursively to the LDAP groups and users that will be imported into Platform Server.
Returns	Returns an `importedGroupInfoList` object containing a list of LDAP groups on the basis of a saved LDAP query or the combination of LDAP profile, base string and searchfilter.		

Examples	**URL:**
	
	Option 1-Import based on LDAP profile, search filter and base string
	
	`http://serverName:serverPort/rest/service/admin/groups?op=importgroups&ldapprofile=MyOrgIndia&basestring=dc=india,dc=myorg,dc=com&searchfilter=cn=qpp server`
	
	Option 2 – Import based on saved LDAP query
	
	`http://serverName:serverPort/rest/service/admin/groups?op=importgroups&savedldapquery=SavedSearch_MyOrgGroups`
	
	Result:
	
	`<importedGroupInfoList>`
	`  <importedGroupInfo>`
	`  <groupInfo id="3681">`
	`  <name>QPP Server</name>`
	`  <emailAddress>QPP Server</emailAddress>`
	`  <defaultUserClass>Administrator<defaultUserClass>`
	`  <nativeTrustee>false</nativeTrustee> .....`
	`</groupInfo>`
	`  <subGroupsInfoList>`
	`  <groupInfo id="3682">`
	`  <name>QPP Server R&D</name> .....`
	`</groupInfo>`
	`  </subGroupsInfoList>`
	`  <userInfoList>`
	`  <userInfo id="3687">`
	`  <name>kshankarraman</name> ..... . </userInfo>`
	`  <userInfo id="3688">`
	`  <name>Menon</name> ..... </userInfo>`
	`  </userInfoList>`
	`  </importedGroupInfo>`
	`</importedGroupInfoList>`		

synchronizeLdapUsersGroups

Function	Synchronize all LDAP users and groups.		
Description	This request synchronizes all users and groups that were imported from LDAP Server to Platform Server.		
Type	GET		
URL Format	`../admin/ldapprofiles`		
Java API	**Method Name**		**Class Name**
	`synchronizeLdapUsersGroups`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=synchronize`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/ldapprofiles?op=synchronize`		

synchronizeLdapUsersGroupsBasedOnProfile

Function	Synchronize all LDAP users and groups.		
Description	This request synchronizes all users and groups that were imported from LDAP Server to Platform Server using the given LDAP profile..		
Type	GET		
URL Format	`../admin/ldapprofiles/{ldapprofileIdOrName}`		
Java API	**Method Name**		**Class Name**
	`synchronizeLdapUsersGroupsBasedOnProfile`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=synchronize`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/ldapprofiles/Quark India? op=synchronize`		

createLdapSearches

Function	Create LDAP search definitions.		
Description	This request creates creates LDAP search definitions.		
Type	POST		
URL Format	`../admin/ldapsearchdefinitions/`		
Java API	**Method Name**		**Class Name**
	`createLdapSearches`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
	`ldapsearchinfolist=<xml/>`		See "Request/Response parameter formats."
Returns	Returns a `ldapsearchinfolist` object in XML format that contains the metadata of all successfully created LDAP search definitions.		

Examples	**URL:**
	`https://serverName:serverPort/rest/service/admin/ldapsearchdefinitions?op=create&ldapsearchinfolist=`
	`<ldapSearchInfoList>`
	`  <ldapSearchInfo>`
	`  <name>Demo groups LDAP search</name>`
	`  <ldapSearchCriteria>`
	`  <ldapProfile>Quark India</ldapProfile>`
	`  <searchFilter>cn=qpp cli*</searchFilter>`
	`  <baseString>dc=india,dc=quark,dc=com</baseString>`
	`  <searchType>GROUPS_SEARCH</searchType>`
	`  </ldapSearchCriteria>`
	`  </ldapSearchInfo>`
	`</ldapSearchInfoList>`
	Result:
	`<ldapSearchInfoList>`
	`  <ldapSearchInfo id="123">`
	`  <name>Demo groups LDAP search</name>`
	`  <ldapSearchCriteria>`
	`  <ldapProfile id="18">Quark India</ldapProfile>`
	`  <searchFilter>cn=qpp cli*</searchFilter>`
	`  <baseString>dc=india,dc=quark,dc=com</baseString>`
	`  <searchType>GROUPS_SEARCH</searchType>`
	`  </ldapSearchCriteria>`
	`  </ldapSearchInfo>`
	`</ldapSearchInfoList>`		

getLdapSearch

Function	Get LDAP search definition.
Description	This request returns the LDAP search definition for the given search id or name.

Type	GET	
URL Format	`../admin/ldapsearchdefinitions{searchIdOrName}`	
Java API	**Method Name**	**Class Name**
	`getLdapSearch`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `ldapSearchInfoList` object containing the LDAP search definition for the given search id or name.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/ldapsearchdefinitions/MySearch`
	Result:
	`<ldapSearchInfoList>`
	`  <ldapSearchInfo id="10">`
	`  <name>MySearch</name>`
	`  <ldapSearchCriteria>`
	`  <ldapProfile id="20">shankar</ldapProfile>`
	`  <searchFilter>sn=*gupta</searchFilter>`
	`  <baseString>dc=quark,dc=com</baseString>`
	`  <searchType id="1">USERS_SEARCH</searchType>`
	`  </ldapSearchCriteria>`
	`  </ldapSearchInfo>`
	`</ldapSearchInfoList>`	

getAllLdapSearches

Function	Get all saved LDAP search definitions.	
Description	This request returns all saved LDAP search definitions.	
Type	GET	
URL Format	`../admin/ldapsearchdefinitions`	
Java API	**Method Name**	**Class Name**
	`getAllLdapSearches`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `ldapSearchInfoList` object containing a list of ldapSearchInfo objects containing the LDAP search definition.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/ldapsearchdefinitions`
	Result:
	`<ldapSearchInfoList>`
	`  <ldapSearchInfo id="438">`
	`  <name>LDAP User Search</name>`
	`  <ldapSearchCriteria>`
	`  <ldapProfile id="764">QPP LDAP Profile</ldapProfile>`
	`  <searchFilter>sn=kumar</searchFilter>`	

```
<baseString>ou=users,ou=qmhi,dc=india,dc=quark,dc=com</baseS-
tring>
            <searchType id="1">USERS_SEARCH</searchType>
        </ldapSearchCriteria>
    </ldapSearchInfo>
    <ldapSearchInfo id="439">
        <name>LDAP Group Search</name>
        <ldapSearchCriteria>
            <ldapProfile id="764">QPP LDAP Profile</ldapProfile>
            <searchFilter>cn=qppserver</searchFilter>
            <baseString>dc=india,dc=quark,dc=com</baseString>
            <searchType id="2">GROUPS_SEARCH</searchType>
        </ldapSearchCriteria>
    </ldapSearchInfo>
</ldapSearchInfoList>
```

updateLdapSearches

Function	Update LDAP search definitions.		
Description	This request update the given LDAP search definitions.		
Type	POST		
URL Format	`../admin/ldapsearchdefinitions/`		
Java API	**Method Name**		**Class Name**
	`uodateLdapSearches`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
	`ldapsearchinfolist=<xml/>`		See “Request/Response parameter formats.”
Returns	Returns a `ldapsearchinfolist` object in XML format that contains the metadata of all successfully updated LDAP search definitions.		

Examples	**URL:**
	`https://serverName:serverPort/rest/service/admin/ldapsearchdefinitions?op=update&ldapsearchinfolist=`
	`<ldapSearchInfoList>`
	`  <ldapSearchInfo>`
	`  <name>Demo groups LDAP search</name>`
	`  <ldapSearchCriteria>`
	`  <ldapProfile>Quark India</ldapProfile>`
	`  <searchFilter>cn=qpp cli*</searchFilter>`
	`  <baseString>dc=india,dc=quark,dc=com</baseString>`
	`  <searchType>GROUPS_SEARCH</searchType>`
	`  </ldapSearchCriteria>`
	`  </ldapSearchInfo>`
	`</ldapSearchInfoList>`
	Result:
	`<ldapSearchInfoList>`		

	`<ldapSearchInfo id="123">`
	`<name>Demo groups LDAP search</name>`
	`<ldapSearchCriteria>`
	`<ldapProfile id="10">Quark India</ldapProfile>`
	`<searchFilter>cn=qpp cli*</searchFilter>`
	`<baseString>dc=india,dc=quark,dc=com</baseString>`
	`<searchType id="2">GROUPS_SEARCH</searchType>`
	`</ldapSearchCriteria>`
	`</ldapSearchInfo>`
	`</ldapSearchInfoList>`

deleteLdapSearch

Function	Delete an LDAP search definition.		
Description	This request deletes the given LDAP search definition with the given ID or name.		
Type	POST		
URL Format	`../admin/ldapsearchdefinitions/{searchIdOrName}`		
Java API	**Method Name**		**Class Name**
	`deleteLdapSearch`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/ldapsearchdefinitions/ MySearch?op=delete`		

getTrustee

Function	Get a trustee (a user or group) with a particular name.	
Description	This request returns all the metadata of a specified trustee.	
Type	GET	
URL Format	`.../admin/trustees/[trustee name]`	
Java API	**Method Name**	**Class Name**
	`getTrustee`	com.quark.qpp.service.facade Class AdminFacade
Parameters	This request requires no input parameters.	
Returns	Returns a `trusteeInfo` object in XML format of specified user or group.	
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/trustees/Admin`	

	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<trusteeInfo id="1">`
	`  <user>true</user>`
	`  <deleted>false</deleted>`
	`  <name>Admin</name>`
	`  <emailAddress>Editors@company.com</emailAddress>`
	`  <nativeTrustee>false</nativeTrustee>`
	`  <defaultUserClassId>2</defaultUserClassId>`
	`  <ldapProfileId>10</ldapProfileId>`
	`  <ldapDistinguishedName>Dist Name</ldapDistinguishedName>`
	`  <domainId>0</domainId>`
	`</trusteeInfo>`

updateTrusteeRule

Function	Update role for the given trustee.		
Description	This request sets the default role for the given trustee (User or Group).		
Type	POST		
URL Format	`.../admin/trustees/{trusteeIdOrName}`		
Java API	**Method Name**		**Class Name**
	`updateTrusteeRole`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=setrole`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation will run only for a valid session id.
	`role=[RoldId] or [Name]`	string	
	`overrideuserclass= [True] or [False]`	boolean	This flag is applicable only if the trustee Id provided is that of a group. If true, it will override the userclass of all users belonging to the given group.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/trustees/John?op=setrole&role=Publishers`		

createWorkflows

Function	Create workflows.
Description	This request is used to create one or more workflows and their statuses.
Type	POST
URL Format	`.../admin/workflows`

Java API	**Method Name**		**Class Name**
	`createWorkFlows`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`workflowinfolist =<xml/>`		Object containing a list of the workflows to be created. See the workflowInfoList paramater in “Request/Response parameter formats”.
Returns	Returns a `workflowInfoList` object containing containing the successfully created workflows.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/workflows?op=create&workflowinfolist=` **Result:**		

```
<workflowInfoList>
    <workflowInfo>
        <name>Publsihing Workflow</name>
        <statusInfoList>
            <statusInfo>
                <name>New Status 001</name>
                <colorBlue>255</colorBlue>
                <position>1</position>
                <redlineTracking>false</redlineTracking>
                <checkInLayoutEvaluation>false</checkInLayoutE-
valuation>
                <outputLayoutEvaluation>false</outputLayoutEvalu-
ation>
                <privilegeInfoList>
                    <privilegeInfo>
                        <roleName>Administrator</roleName>
                        <roleId>1</roleId>
                        <overridden>false</overridden>
                        <contentTypePrivList/>
                    </privilegeInfo>
                </privilegeInfoList>
                <attributeConstraintsInfoList>
                    <attributeConstraintsInfo>
                        <attributeName>Name</attributeName>
                        <roleName>Administrator</roleName>
                        <constraintName>PREVENT CHANGE</con-
straintName>
                    </attributeConstraintsInfo>
                </attributeConstraintsInfoList>
            </statusInfo>
        </statusInfoList>
        <contentTypeInfoList>
            <contentTypeInfo>
                <name>Asset</name>
            </contentTypeInfo>
            <contentTypeInfo id="1102"/>
        </contentTypeInfoList>
    </workflowInfo>
</workflowInfoList>
```

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<workflowInfoList>
    <workflowInfo id="20">
        <name>Publishing Workflow</name>
        <domainId>18</domainId>
        <created>2013-11-28T08:58:00Z</created>
        <createdBy>1</createdBy>
        <creatorName>Admin</creatorName>
        <lastModifiedBy>1</lastModifiedBy>
        <lastModifierName>Admin</lastModifierName>
        <lastModified>2013-11-28T08:58:00Z</lastModified>
        <statusInfoList>
            <statusInfo id="20">
                <name>New Status 001</name>
                <domainId>6</domainId>
                <colorRed>0</colorRed>
                <colorGreen>0</colorGreen>
                <colorBlue>255</colorBlue>
                <position>1</position>
                <redlineTracking>false</redlineTracking>
                <checkInLayoutEvaluation>false</checkInLayoutE-
valuation>
                <outputLayoutEvaluation>false</outputLayoutEvalu-
ation>
                <privilegeInfoList>
                    <privilegeInfo>
                        <roleName>Administrator</roleName>
                        <roleId>1</roleId>
                        <overridden>false</overridden>
                        <contentTypePrivList/>
                    </privilegeInfo>
                </privilegeInfoList>
                <attributeConstraintsInfoList>
                    <attributeConstraintsInfo>
                        <attributeId>2</attributeId>
                        <attributeName>Name</attributeName>
                        <roleId>1</roleId>
                        <roleName>Administrator</roleName>
                        <constraintId>1</constraintId>
                        <constraintName>PREVENT CHANGE</con-
straintName>
                    </attributeConstraintsInfo>
                </attributeConstraintsInfoList>
            </statusInfo>
        </statusInfoList>
        <contentTypeInfoList>
            <contentTypeInfo id="1102">
                <name>Project</name>
                <contentTypeHierarchy>System;Asset;Project;</con-
tentTypeHierarchy>
                <domainId>0</domainId>
                <systemDefined>true</systemDefined>
                <childrenAvailable>true</childrenAvailable>
                <extensible>true</extensible>
            </contentTypeInfo>
        </contentTypeInfoList>
    </workflowInfo>
</workflowInfoList>
```


getWorkflow

Function	Get a workflow with a particular ID or name.

Description	This request returns all the metadata of a specified workflow.	
Type	GET	
URL Format	`.../admin/workflows/[workflow name]`	
Java API	**Method Name**	**Class Name**
	`getWorkflow`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `workflowInfoList` object with the metadata of a specified workflow in XML format.	

Examples

URL:

`http://serverName:serverPort/rest/service/admin/workflows/workflow1`

Result:

```
<workflowInfoList>
    <workflowInfo id="10">
        <name>Workflow1</name>
        <domainId>18</domainId>
        <created>2013-12-23T12:06:33Z</created>
        <createdBy>1</createdBy>
        <creatorName>Admin</creatorName>
        <lastModifiedBy>1</lastModifiedBy>
        <lastModifierName>Admin</lastModifierName>
        <lastModified>2013-12-23T12:06:33Z</lastModified>
        <statusInfoList>
            <statusInfo id="10">
                <name>Status1</name>
                <domainId>6</domainId>
                <colorRed>50</colorRed>
                <colorGreen>150</colorGreen>
                <colorBlue>10</colorBlue>
                <position>1</position>
                <redlineTracking>false</redlineTracking>
                <checkInLayoutEvaluation>false</checkInLayoutE-
valuation>
                <outputLayoutEvaluation>false</outputLayoutEvalu-
ation>
                <privilegeInfoList>
                    <privilegeInfo>
                        <roleName>Administrator</roleName>
                        <roleId>1</roleId>
                        <overridden>false</overridden>
                        <contentTypePrivList/>
                    </privilegeInfo>
                    <privilegeInfo>
                        <roleName>UserClass2</roleName>
                        <roleId>10</roleId>
                        <overridden>false</overridden>
                        <contentTypePrivList/>
                    </privilegeInfo>
                </privilegeInfoList>
                <attributeConstraintsInfoList/>
            </statusInfo> ..... </statusInfoList>
        <contentTypeInfoList>
            <contentTypeInfo id="1100">
                <name>Asset</name>
                <contentTypeHierarchy>System;Asset;</contentType-
Hierarchy>
```

```
                <domainId>0</domainId>
                <systemDefined>true</systemDefined>
                <childrenAvailable>true</childrenAvailable>
                <extensible>true</extensible>
            </contentTypeInfo> ...... </contentTypeInfoList>
    </workflowInfo>
</workflowInfoList>
```

Function	Get a workflow		
Description	This request returns the metadata and status of a specified workflow.		
Type	Get		
URL Format	`..../admin/workflows/{workflowIdorName}`		
Java API	**Method Name**		**Class Name**
	`getWorkflow`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`workflowIdOrName`	string	The ID or Name of the workflow to be retrieved.
Returns	Returns a `workflowInfoList` object containing information on the specified workflow.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/workflows/21` **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<workflowInfoList>
    <workflowInfo id="21">
        <name>New Workflow</name>
        <domainId>18</domainId>
        <created>2013-12-09T11:45:42Z</created>
        <createdBy>1</createdBy>
        <creatorName>Admin</creatorName>
        <lastModifiedBy>1</lastModifiedBy>
        <lastModifierName>Admin</lastModifierName>
        <lastModified>2013-12-09T11:45:42Z</lastModified>
        <statusInfoList>
            <statusInfo id="24">
                <name>New Status 001</name>
                <domainId>6</domainId>
                <colorRed>0</colorRed>
                <colorGreen>0</colorGreen>
                <colorBlue>255</colorBlue>
                <position>1</position>
                <redlineTracking>false</redlineTracking>
                <checkInLayoutEvaluation>false</checkInLayoutE-
valuation>
                <outputLayoutEvaluation>false</outputLayoutEvalu-
ation>
                <privilegeInfoList>
                    <privilegeInfo>
                        <roleName>Administrator</roleName>
                        <roleId>1</roleId>
                        <overridden>false</overridden>
                        <contentTypePrivList/>
```

```
                    </privilegeInfo> ......... </privilegeIn-
foList>
                <attributeConstraintsInfoList>
                    <attributeConstraintsInfo>
                        <attributeId>2</attributeId>
                        <attributeName>Name</attributeName>
                        <roleId>1</roleId>
                        <roleName>Administrator</roleName>
                        <constraintId>1</constraintId>
                        <constraintName>PREVENT CHANGE</con-
straintName>
                    </attributeConstraintsInfo>
                </attributeConstraintsInfoList>
            </statusInfo>
        </statusInfoList>
        <contentTypeInfoList>
            <contentTypeInfo id="1100">
                <name>Asset</name>
                <contentTypeHierarchy>System;Asset;</contentType-
Hierarchy>
                <domainId>0</domainId>
                <systemDefined>true</systemDefined>
                <childrenAvailable>true</childrenAvailable>
                <extensible>true</extensible>
            </contentTypeInfo> ......... </contentTypeInfoList>
    </workflowInfo>
</workflowInfoList>
```

getAllWorkflows

Function	Get all workflows.	
Description	This request returns all the metadata of all the workflows that exist on the QPP Server.	
Type	GET	
URL Format	`.../admin/workflows`	
Java API	**Method Name**	**Class Name**
	`getAllWorkflows`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `workflowInfoList` object containing an array of all the workflows that exist on the QPP Server.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/workflows`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<workflowInfoList>`
	`  <workflowInfo id="10">`
	`  <name>Workflow1</name> .... <statusInfoList>`
	`  <statusInfo id="10">`
	`  <name>Status1</name> .... </statusInfo>`
	`  <statusInfo id="11">`
	`  <name>Status2</name> .... </statusInfo>`
	`  </statusInfoList>`	

```
    </workflowInfo>
    <workflowInfo id="11">
        <name>Workflow2</name> .... <statusInfoList>
            <statusInfo id="12">
                <name>Status1</name> .... </statusInfo>
            <statusInfo id="13">
                <name>Status2</name> .... </statusInfo>
        </statusInfoList>
    </workflowInfo>
</workflowInfoList>
```

updateWorkflows

Function	Update workflows.		
Description	This request is used to update the metadata of a specified single or multiple workflows and their statuses.		
Type	POST		
URL Format	`..../admin/workflows`		
Java API	**Method Name**		**Class Name**
	`updateWorkflows`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`workflowinfolist=<xml />`		Object containing a list of the successfully updated workflows .
Returns	Returns a `workflowInfoList` object containing a list of all the successfully updated workflows.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/workflows?op=update&workflowinfolist=`
	`<workflowInfoList>`
	`  <workflowInfo id="22">`
	`  <name>DefaultWorkflow New</name>`
	`  </workflowInfo>`
	`</workflowInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<workflowInfoList>`
	`  <workflowInfo id="22">`
	`  <name>DefaultWorkflow New</name>`
	`  <domainId>18</domainId>`
	`  <created>2013-11-28T09:01:38Z</created>`
	`  <createdBy>1</createdBy>`
	`  <creatorName>Admin</creatorName>`
	`  <lastModifiedBy>1</lastModifiedBy>`
	`  <lastModifierName>Admin</lastModifierName>`
	`  <lastModified>2013-11-28T09:03:26Z</lastModified>`
	`  <statusInfoList/>`
	`  <contentTypeInfoList>`		

```
            <contentTypeInfo id="1100">
                <name>Asset</name>
                <contentTypeHierarchy>System;Asset;</contentType-
Hierarchy>
                <domainId>0</domainId>
                <systemDefined>true</systemDefined>
                <childrenAvailable>true</childrenAvailable>
                <extensible>true</extensible>
            </contentTypeInfo> ......... </contentTypeInfoList>
    </workflowInfo>
</workflowInfoList>
```

deleteWorkFlow

Function	Delete a workflow.		
Description	This request is used to delete the specified workflow.		
Type	POST		
URL Format	`.../admin/workflows/{workflowIdorName}`		
Java API	**Method Name**		**Class Name**
	`deleteWorkFlow`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`workflowIdOrName`	string	ID or Name of the workflow to be deleted.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/workflows/22?op=delete`		

createPreferences

Function	Create Preferences.		
Description	This request is used to create a single or multiple preferences at one time.		
Type	POST		
URL Format	`.../admin/preferences`		
Java API	**Method Name**		**Class Name**
	`createPreferences`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)

	preferenceinfolist	string	List of preferences to create.
Returns	Returns a preferenceInfoList object containing a list of successfully created preferences.		
Examples	**URL:** http://serverName:serverPort/rest/service/admin/preferences?op=create&preferenceinfolist=<xml/> **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <preferenceInfoList> <preferenceInfo id="10005"> <name>Demo user preference 3</name> <category id="1">USER</category> <possibleValuesExist>false</possibleValuesExist> <defaultValue>Ths is the default value.</defaultValue> <systemDefined>false</systemDefined> </preferenceInfo> <preferenceInfo id="10006"> <name>Demo preference without possible values</name> <category id="0">SYSTEM</category> <possibleValuesExist>false</possibleValuesExist> <defaultValue>This is the default new value.</defaultValue> <systemDefined>false</systemDefined> </preferenceInfo> </preferenceInfoList>		

getPreference

Function	Get a specific preference.		
Description	This request returns the metadata of a specified preference.		
Type	GET		
URL Format	../admin/preferences/{preferenceIdOrName}		
Java API	**Method Name**		**Class Name**
	getPreference		com.quark.qpp.service.facade Class AdminFacade
Parameters	**Name**	**Type**	**Description**
	preferenceIdOrName	string	The ID or Name of the preference to be retrieved.
Returns	Returns a preferenceInfoList object containing the metadata of the specified preference.		
Examples	**URL:** http://serverName:serverPort/rest/service/admin/preferences/Content Display View **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <preferenceInfoList> <preferenceInfo id="2006"> <name>Content Display View</name>		

```
        <category id="1">USER</category>
        <possibleValuesExist>true</possibleValuesExist>
        <defaultValue id="1">Keep Last Setting</defaultValue>
        <preferenceValueList>
            <preferenceValue valueId="1" preferenceName="Content
Display View" preferenceId="2006">Keep Last Setting</preference-
Value>
            <preferenceValue valueId="2" preferenceName="Content
Display View" preferenceId="2006">WYSIWYG</preferenceValue>
            <preferenceValue valueId="3" preferenceName="Content
Display View" preferenceId="2006">Galley</preferenceValue>
            <preferenceValue valueId="4" preferenceName="Content
Display View" preferenceId="2006">Full Screen</preferenceValue>
        </preferenceValueList>
        <systemDefined>true</systemDefined>
    </preferenceInfo>
</preferenceInfoList>
```

getAllPreferences

Function	Get all Preferences.	
Description	This request returns all of the existing preference definitions.	
Type	GET	
URL Format	`../admin/preferences`	
Java API	**Method Name**	**Class Name**
	`getAllPreferences`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `preferenceInfoList` object containing metadata of all existing preferences with its possible values.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/preferences`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<preferenceInfoList>`
	`<preferenceInfo id="1007">`
	`<name>Display all Redline Deletions in One Color</name>`
	`<category id="0">SYSTEM</category>`
	`<possibleValuesExist>false</possibleValuesExist>`
	`<defaultValue>No</defaultValue>`
	`<systemDefined>true</systemDefined>`
	`</preferenceInfo>`
	`<preferenceInfo id="1008">`
	`<name>Redline Deletion Color</name>`
	`<category id="0">SYSTEM</category>`
	`<possibleValuesExist>false</possibleValuesExist>`
	`<defaultValue>255,0,0</defaultValue>`
	`<systemDefined>true</systemDefined>`
	`</preferenceInfo> .... </preferenceInfoList>`	

getPreferenceValues

Function	Get Preference values.		
Description	This request is used to get preference values for a given preference corresponding to the logged-in user.		
Type	GET		
URL Format	`....admin/preferencevalues`		
Java API	**Method Name**		**Class Name**
	`getPreferenceValues`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`preferences`	string	Comma separated list of Ids or Names of Preferences to be retrieved.
Returns	Returns an `preferenceValueList` object containing the values set for given preferences for the logged-in user.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/preferencevalues?preferences=3018, 1007, 3011, Archive Revisions`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<preferenceValueList>`
	`<preferenceValue preferenceName="Display all Redline Deletions in One Color" preferenceId="1007">No</preferenceValue>`
	`<preferenceValue valueId="1" preferenceName="Default Action for Double Click on Quark XPress Projects" preferenceId="3011">Do Check Out</preferenceValue>`
	`<preferenceValue valueId="3" preferenceName="Delete Attached Files" preferenceId="3018">Ask Before Deleting</preferenceValue>`
	`<preferenceValue valueId="3" preferenceName="Archive Revisions" preferenceId="4018">Ask</preferenceValue>`
	`</preferenceValueList>`		

updatePreferences

Function	Update Preferences.		
Description	This request is used to update the metadata of a specified single or multiple preferences.		
Type	POST		
URL Format	`..../admin/preferences`		
Java API	**Method Name**		**Class Name**
	`updatePreferences`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)

	preferenceinfolist=<xml/>		Object containing the metadata of the preferences to be updated.
Returns	Returns a preferenceInfoList object containing a list of all the successfully updated preferences.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/admin/preferences?op=update&preferenceinfolist=<xml/>
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<preferenceInfoList>`
	`  <preferenceInfo id="10005">`
	`  <name>Demo user preference 3</name>`
	`  <category id="1">USER</category>`
	`  <possibleValuesExist>false</possibleValuesExist>`
	`  <defaultValue>Ths is the default value.</defaultValue>`
	`  <systemDefined>false</systemDefined>`
	`  </preferenceInfo>`
	`  <preferenceInfo id="10006">`
	`  <name>Demo preference without possible values</name>`
	`  <category id="0">SYSTEM</category>`
	`  <possibleValuesExist>false</possibleValuesExist>`
	`  <defaultValue>This is the default new value.</defaultValue>`
	`  <systemDefined>false</systemDefined>`
	`  </preferenceInfo>`
	`</preferenceInfoList>`		

updatePreferenceValues

Function	Update Preference Values.		
Description	This request updates the preference values corresponding to the logged-in user.		
Type	POST		
URL Format	.../preferenceValues		
Java API	**Method Name**		**Class Name**
	updatePreferenceValues		com.quark.qpp.service.facade Class AdminFacade
Parameters	**Name**	**Type**	**Description**
	op=update	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	preferencevaluelist=[xml]	string	Object containing the metadata of the preference valuess to be updated.
Returns	Returns a preferencevaluelist object containing the preferences values successfully updated.		
Examples	**URL:** http://serverName:serverPort/rest/service/admin/preferencevalues?op=update&preferencevaluelist=<xml/>		

	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<preferenceValueList>`
	`  <preferenceValue valueId="1" preferenceName="user 1" preferenceId="10061">system value 2</preferenceValue>`
	`  <preferenceValue valueId="2" preferenceName="user 2" preferenceId="10062">system value 3</preferenceValue>`
	`</preferenceValueList>`

deletePreference

Function	Delete a preference.		
Description	This request is used to delete the specified preference.		
Type	POST		
URL Format	`.../admin/preferences/{preferenceIdorName}`		
Java API	**Method Name**		**Class Name**
	`deletePreference`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/preference/{preferenceIdOrName}?op=delete`		

createRepositories

Function	Create one or more storage repositories.		
Description	This request is used to create new storage repositories.		
Type	POST		
URL Format	`.../admin/repositories`		
Java API	**Method Name**		**Class Name**
	`createRepositories`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`storagerepositoryinfolist`	string	List of storage repositories to create.
Returns	Returns a `storageRepositoryInfoList` object containing a list of successfully		

	created repositories.

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/repositories?op=create&storagerepositoryinfolist=`
	`<storageRepositoryInfoList>`
	`  <storageRepositoryInfo>`
	`  <name>PlatformRepository</name>`
	`  <connectionInfo><![CDATA[<connectionProperties><property name='url' required='true'>C:\PlatformRepo</property></connectionProperties>]]></connectionInfo>`
	`  <repositoryType>fileRepositoryAdapter</repositoryType>`
	`  </storageRepositoryInfo>`
	`</storageRepositoryInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?><storageRepositoryInfoList>`
	`  <storageRepositoryInfo id="20">`
	`  <name>PlatformRepository</name>`
	`  <connectionInfo><connectionProperties><property name='url' required='true'>C:\PlatformRepo< /property>< /connectionProperties>`
	`  </connectionInfo>`
	`  <repositoryType>fileRepositoryAdapter</repositoryType>`
	`  <repositoryTypeId>1</repositoryTypeId>`
	`  <status>READ_WRITE</status>`
	`  <statusId>1</statusId>`
	`  </storageRepositoryInfo>`
	`</storageRepositoryInfoList>`

getRepository

Function	Get a specific repository.		
Description	This request returns a storage repository with the given Id or Name.		
Type	GET		
URL Format	`../admin/repositories/{repositoryIdOrName}`		
Java API	**Method Name**		**Class Name**
	`getRepository`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`repositoryIdOrName`	string	The ID or Name of the repository to be retrieved.
Returns	Returns a `repositoryInfoList` object containing the metadata of the specified repository.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/repositories/1`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<storageRepositoryInfoList>`
	`  <storageRepositoryInfo id="1">`		

```
        <name>TestRep1</name>
        <connectionInfo><connectionProperties><property
name='url' required='true'> C:\TestRep1</property></connection-
Properties></connectionInfo>
        <repositoryType>fileRepositoryAdapter</repositoryType>
        <repositoryTypeId>1</repositoryTypeId>
        <status>READ_WRITE</status>
        <statusId>1</statusId>
    </storageRepositoryInfo>
    <storageRepositoryInfo id="10">
        <name>TestRepA</name>
        <connectionInfo><connectionProperties><property
name='url' required='true'> C:\TestRepA</property></connection-
Properties></connectionInfo>
        <repositoryType>fileRepositoryAdapter</repositoryType>
        <repositoryTypeId>1</repositoryTypeId>
        <status>READ_WRITE</status>
        <statusId>1</statusId>
    </storageRepositoryInfo>
</storageRepositoryInfoList>
```

getAllRepositories

Function	Get all storage repositories.	
Description	This request returns all the information regarding all repositories registered with the platform.	
Type	GET	
URL Format	`../admin/repositories`	
Java API	**Method Name**	**Class Name**
	`getAllRepositories`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `storageRepositoryInfoList` object containing a list of all the registered repositories.	

Examples

URL:

`http://serverName:serverPort/rest/service/admin/repositories`

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<storageRepositoryInfoList>
    <storageRepositoryInfo id="1">
        <name>TestRep1</name>
        <connectionInfo><connectionProperties><property
name='url' required='true'> C:\TestRep1</property></connection-
Properties></connectionInfo>
        <repositoryType>fileRepositoryAdapter</repositoryType>
        <repositoryTypeId>1</repositoryTypeId>
        <status>READ_WRITE</status>
        <statusId>1</statusId>
    </storageRepositoryInfo>
    <storageRepositoryInfo id="10">
        <name>TestRepA</name>
        <connectionInfo><connectionProperties><property
name='url' required='true'> C:\TestRepA</property></connection
```

	`Properties></connectionInfo>`
	`<repositoryType>fileRepositoryAdapter</repositoryType>`
	`<repositoryTypeId>1</repositoryTypeId>`
	`<status>READ_WRITE</status>`
	`<statusId>1</statusId>`
	`</storageRepositoryInfo>`
	`</storageRepositoryInfoList>`

updateRepositories

Function	Update one or more storage repositories.		
Description	This request is used to update the storage repositories on the system.		
Type	POST		
URL Format	`..../admin/repositories`		
Java API	**Method Name**		**Class Name**
	`updateRepositories`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`storagerepositoryinfolist=<xml/>`		Object containing the metadata of the storage repositories to be updated.
Returns	Returns a `storageRepositoryInfoList` object containing a list of all the successfully updated storage repositories.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/repositories?op=update&storagerepositoryinfolist=`
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<storageRepositoryInfoList>`
	`<storageRepositoryInfo id="24">`
	`<name>QPP_Repo111</name>`
	`<connectionInfo><connectionProperties><property name='url' required='false'> \\dinesh-2k8\Repo1</property></connectionProperties></connectionInfo>`
	`<repositoryType>fileRepositoryAdapter</repositoryType>`
	`<repositoryTypeId>1</repositoryTypeId>`
	`<status>READ_WRITE</status>`
	`<statusId>1</statusId>`
	`</storageRepositoryInfo>`
	`</storageRepositoryInfoList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<storageRepositoryInfoList>`
	`<storageRepositoryInfo id="24">`
	`<name>QPP_Repo111</name>`
	`<connectionInfo><connectionProperties><property name='url' required='false'> \\dinesh-2k8\Repo1</property></connectionProperties></connectionInfo>`
	`<repositoryType>fileRepositoryAdapter</repositoryType>`
	`<repositoryTypeId>1</repositoryTypeId>`		

	`<status>READ_WRITE</status>`
	`<statusId>1</statusId>`
	`</storageRepositoryInfo>`
	`</storageRepositoryInfoList>`

deleteRepository

Function	Delete a storage repository.		
Description	This request is used to delete the specified storage repository.		
Type	POST		
URL Format	`.../admin/repositories/{repositoryIdorName}`		
Java API	**Method Name**		**Class Name**
	`deleteRepository`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/repositories/10?op=delete`		

createPrivileges

Function	Create Privileges.		
Description	This request creates the required privilege definition(s).		
Type	POST		
URL Format	`.../privileges?op=create`		
Java API	**Method Name**		**Class Name**
	`createPrivileges`		`com.quark.qpp.service.facadeClass AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`privilegedefinitionlist =<XML/>`	string	A privilegeDefinitionList object in XML format. See “Request/Response parameter formats.”
Returns	Returns a `privilegeDefinitionList` object containing the metadata of the newly created privileges.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/privileges?op=create` **Result:**		

	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<privilegeDefinitionList>`
	`  <privilegeDefinition name="Sample Privilege" id="2500000">`
	`  <group id="100000">General</group>`
	`  <statusBased>true</statusBased>`
	`  <systemDefined>false</systemDefined>`
	`  <contentPrivilege>true</contentPrivilege>`
	`  <contentTypeInfoList>`
	`  <contentTypeInfo id="1100">`
	`  <name>Asset</name>`
	`  </contentTypeInfo>`
	`  </contentTypeInfoList>`
	`  </privilegeDefinition>`
	`</privilegeDefinitionList>`

getPrivilege

Function	Get Privilege.	
Description	This request returns a specific privilege(s) in system with given id or name.	
Type	GET	
URL Format	`.../privileges/{privilegeIdOrName}`	
Java API	**Method Name**	**Class Name**
	`getPrivilege`	`com.quark.qpp.service.facadeClass AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns specific privilege(s) in system with given ID or name. If privilege name is specified and there exists multiple privileges in different privilege groups, with the given name, list of all such privileges will be returned.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/privileges/Override Another Users Checkout`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<privilegeDefinitionList>`
	`  <privilegeDefinition name="Override Another Users Checkout" id="1000003">`
	`  <group id="100000">General</group>`
	`  <statusBased>true</statusBased>`
	`  <systemDefined>true</systemDefined>`
	`  <contentPrivilege>true</contentPrivilege>`
	`  <contentTypeInfoList>`
	`  <contentTypeInfo id="1100">...</contentTypeInfo>`
	`  <contentTypeInfo id="1101">...</contentTypeInfo>`
	`  <contentTypeInfo id="1102">`
	`  <name>Project</name>`
	`  </contentTypeInfo>`
	`  <contentTypeInfo id="1103">`
	`  <name>Picture</name>`
	`  </contentTypeInfo>`
	`  <contentTypeInfo id="1104">...</contentTypeInfo>`
	`  <contentTypeInfo id="1105">...</contentTypeInfo>`
	`  </contentTypeInfoList>`
	`  </privilegeDefinition> ..... .... </privilegeDefinitionList>`	

getAllPrivileges

Function	Get All Privileges.		
Description	This request returns all privilege definitions in the system.		
Type	GET		
URL Format	`.../privileges`		
Java API	**Method Name**		**Class Name**
	`getAllPrivileges`		`com.quark.qpp.service.facadeClass AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`contenttype=[true|false]`	boolean	Defaul value is true
	`application=[true|false]`	boolean	Defaul value is true
Returns	Returns a `privilegeDefinitionList` object containing the metadata of all privileges in the system.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/privileges` **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<privilegeDefinitionList>
    <privilegeDefinition name="Check In New" id="1000001">
        <group id="100000">General</group>
        <statusBased>false</statusBased>
        <systemDefined>true</systemDefined>
        <contentPrivilege>true</contentPrivilege>
        <contentTypeInfoList>
            <contentTypeInfo id="1100">
                <name>Asset</name>
            </contentTypeInfo>
            <contentTypeInfo id="1101">
                <name>Document</name>
            </contentTypeInfo>
            <contentTypeInfo id="1102">...</contentTypeInfo>
            <contentTypeInfo id="1103">...</contentTypeInfo>
            <contentTypeInfo id="1104">...</contentTypeInfo> ...
</contentTypeInfoList>
    </privilegeDefinition> ..... ..... </privilegeDefinitionList>
```

getApplicableApplicationPriveleges

Function	Get applicable application privileges for the logged in user.	
Description	This request returns enabled applicable application privileges for the logged in user. Applicable privileges are a union of all the privileges granted to him as a user as well as to groups of which the user is a member.	
Type	GET	
URL Format	`../admin/users/`	
Java API	**Method Name**	**Class Name**

	getApplicableApplicationPrivileges		com.quark.qpp.service.facade Class AdminFacade
Parameters	**Name**	**Type**	**Description**
	op= applicableapplication provileges	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	privileges= [comma-separated list of privilege Names or IDs]	string	The set of privilege values. (Optional)
Returns	Returns the applicationPrivList object containing all of the applicable applicaton privileges.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/admin/users?op=applicableapplicationprivileges
	Result:
	`<applicationPrivList>`
	`<privilege id="140000023">`
	`<name>Delete</name>`
	`<privilegeHierarchy> Administration;Storage;Storage Rules;Delete </privilegeHierarchy>`
	`</privilege>`
	`<privilege id="140000022">`
	`<name>Create/Modify</name>`
	`<privilegeHierarchy> Administration;Storage;Storage Rules;Create/Modify </privilegeHierarchy>`
	`</privilege> .....  </applicationPrivList>`
	Get enabled applicable application privileges for the logged in user from the given privileges:URL:http://serverName:serverPort/rest/service/admin/users?op=applicableapplicationprivileges&privileges=140000023,...,Result:
	`<applicationPrivList>`
	`<privilege id="140000023">`
	`<name>Delete</name>`
	`<privilegeHierarchy> Administration;Storage;Storage Rules;Delete </privilegeHierarchy>`
	`</privilege>`
	`<privilege id="140000022">`
	`<name>Create/Modify</name>`
	`<privilegeHierarchy> Administration;Storage;Storage Rules;Create/Modify </privilegeHierarchy>`
	`</privilege> .....  </applicationPrivList>`		

getApplicableContentPrivileges

Function	Get applicable content privileges for the logged in user.
Description	This request returns enabled privileges of the given content type applicable for the logged in user. Applicable privileges are a union of all the privileges granted to him as a user as well as to groups of which the user is a member.
Type	GET

URL Format	`../admin/users/`		
Java API	**Method Name**		**Class Name**
	`getApplicableContentPrivileges`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op= applicablecontentprovileges`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`contenttype=[contenType ID or Name or Hierarch]`	string	
	`privileges= [comma-separated list of privilege Names or IDs]`	string	The set of privilege values. (Optional)
Returns	Returns the `contentPrivList` object containing all of the applicable content privileges.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/users?op=applicablecontentprivileges&contenttype=Picture`
	Result:
	`<contentTypePriv>`
	`  <privilege id="3000006">`
	`  <name>Change Workflow of Checked In Asset</name>`
	`  <privilegeHierarchy> Edit Attributes;Change Workflow of Checked In Asset </privilegeHierarchy>`
	`  </privilege>`
	`  <privilege id="10000015">`
	`  <name>Get Original Assets</name>`
	`  <privilegeHierarchy>General;Get Original Assets</privilegeHierarchy>`
	`  </privilege>`
	`</contentTypePriv>`
	`Get enabled applicable content privileges for the logged in user from the given privileges:URL:http://serverName:serverPort/rest/service/admin/users? op=applicablecontentprivileges&contenttype=Picture&privileges=3000006, 10000015Result:`
	`<contentTypePriv>`
	`  <privilege id="3000006">`
	`  <name>Change Workflow of Checked In Asset</name>`
	`  <privilegeHierarchy> Edit Attributes;Change Workflow of Checked In Asset </privilegeHierarchy>`
	`  </privilege>`
	`  <privilege id="10000015">`
	`  <name>Get Original Assets</name>`
	`  <privilegeHierarchy>General;Get Original Assets</privilegeHierarchy>`
	`  </privilege>`
	`</contentTypePriv>`		

updatePrivileges

Function	Update Privileges.		
Description	This request updates the required privilege definition(s).		
Type	POST		
URL Format	`.../privileges?op=update`		
Java API	**Method Name**		**Class Name**
	`updatePrivileges`		`com.quark.qpp.service.facadeClassAdminFacade`
Parameters	**Name**	**Type**	**Description**
	`privilegedefinitionlist =<XML/>`	string	A privilegeDefinitionList object in XML format. See "Request/Response parameter formats."
Returns	Returns a `privilegeDefinitionList` object containing the metadata of the successfully updated privileges.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/privileges?op=update` **Result:**		

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<privilegeDefinitionList>
    <privilegeDefinition name="Sample Privilege" id="2500000">
        <group id="100000">General</group>
        <statusBased>true</statusBased>
        <systemDefined>false</systemDefined>
        <contentPrivilege>true</contentPrivilege>
        <contentTypeInfoList>
            <contentTypeInfo id="1100">
                <name>Asset</name>
            </contentTypeInfo>
            <contentTypeInfo id="1102">
                <name>Project</name>
            </contentTypeInfo>
        </contentTypeInfoList>
    </privilegeDefinition>
</privilegeDefinitionList>
```

deletePrivilege

Function	Delete Privilege.	
Description	This request deletes the specified privilege Id.	
Type	POST	
URL Format	`.../privileges/{privilegeIdOrName}?op=delete`	
Java API	**Method Name**	**Class Name**
	`deletePrivilege`	`com.quark.qpp.service.facadeClassAdminFacade`
Parameters	This request requires no input parameters.	

Returns	A status of 200 is returned in response to a successful request execution.
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/privileges/{privilegeIdOrName}?op=delete`

createPrivilegeGroups

Function	Creates New Privilege Group.		
Description	This request creates new privilege group along with its subgroups and privileges in the specified hierarchy.		
Type	POST		
URL Format	`.../privilegegroups?op=create`		
Java API	**Method Name**		**Class Name**
	`createPrivilegeGroups`		`com.quark.qpp.service.facadeClass AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`privilegegroupdefinitionlist =<XML/>`	string	A `privilegeGroupDefinitionList` object in XML format. See "Request/Response parameter formats."
Returns	Returns a `privilegeGroupDefinitionList` object containing the metadata of the successfuy created privilege groups.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/privilegegroups?op=create`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<privilegeGroupDefinitionList>`
	`  <privilegeGroupDefinition name="Sample Privilege Group">`
	`  <privilegeDefinitionList>`
	`  <privilegeDefinition name="Check In New" id="1000001">`
	`  <group id="100000">General</group>`
	`  <statusBased>false</statusBased>`
	`  <systemDefined>true</systemDefined>`
	`  <contentPrivilege>true</contentPrivilege>`
	`  <contentTypeInfoList>`
	`  <contentTypeInfo id="1100">`
	`  <name>Asset</name>`
	`  </contentTypeInfo>`
	`  <contentTypeInfo id="1100">`
	`  <name>CustomeContentType_49933</name>`
	`  </contentTypeInfo>`
	`  </contentTypeInfoList>`
	`  </privilegeDefinition>`
	`  </privilegeDefinitionList>`
	`  <parentGroup id="-1"/>`
	`  </privilegeGroupDefinition>`
	`</privilegeGroupDefinitionList>`		

getPrivilegeGroup

Function	Get Privilege Group Definition.	
Description	This request returns all existing privilege group definitions.	
Type	GET	
URL Format	`.../privilegegroups/{privilegeGroupIdOrName}`	
Java API	**Method Name**	**Class Name**
	`getPrivilegeGroup`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns information for the specified privilege group based on its id or name specified.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/privilegegroups/General`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<privilegeGroupDefinitionList>`
	`  <privilegeGroupDefinition name="General" id="100000">`
	`  <privilegeDefinitionList>`
	`  <privilegeDefinition name="Check In New" id="1000001">`
	`  <group id="100000">General</group>`
	`  <statusBased>false</statusBased>`
	`  <systemDefined>true</systemDefined>`
	`  <contentPrivilege>true</contentPrivilege>`
	`  <contentTypeInfoList>`
	`  <contentTypeInfo id="1100">`
	`  <name>Asset</name>`
	`  </contentTypeInfo> ... </contentTypeInfoList>`
	`  </privilegeDefinition> ..... ..... </privilegeDefinitionList>`
	`  </privilegeGroupDefinition>`
	`</privilegeGroupDefinitionList>`	

getAllPrivilegeGroups

Function	Get All Privilege Group Definitions.	
Description	This request returns all existing privilege group definitions.	
Type	GET	
URL Format	`../privilegegroups`	
Java API	**Method Name**	**Class Name**
	`getAllPrivilegeGroups`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns all existing privilege group definitions. Each privilege group contains its information along with its privileges and sub-privilege group definitions.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/privilegegroups`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<privilegeGroupDefinitionList>`
	`  <privilegeGroupDefinition name="Administration" id="99998">...</privilegeGroupDefinition>`
	`  <privilegeGroupDefinition name="Workspace" id="99999"> ... </privilegeGroupDefinition>`
	`  <privilegeGroupDefinition name="General" id="100000">`
	`  <privilegeDefinitionList>`
	`  <privilegeDefinition name="Check In New" id="1000001">`
	`  <group id="100000">General</group>`
	`  <statusBased>false</statusBased>`
	`  <systemDefined>true</systemDefined>`
	`  <contentPrivilege>true</contentPrivilege>`
	`  <contentTypeInfoList>`
	`  <contentTypeInfo id="1100">`
	`  <name>Asset</name>`
	`  </contentTypeInfo> ... </contentTypeInfoList>`
	`  </privilegeDefinition> ..... ..... </privilegeDefinitionList>`
	`  </privilegeGroupDefinition>`
	`</privilegeGroupDefinitionList>`

updatePrivilegeGroups

Function	Updates Given Privilege Group.		
Description	This request updates the given privilege group.		
Type	POST		
URL Format	`.../privilegegroups?op=update`		
Java API	**Method Name**		**Class Name**
	`updatePrivilegeGroups`		`com.quark.qpp.service.facadeClassAdminFacade`
Parameters	**Name**	**Type**	**Description**
	`privilegegroupdefinitionlist =<XML/>`	string	A `privilegeGroupDefinitionList` object in XML format. See "Request/Response parameter formats."
Returns	Returns a `privilegeGroupDefinitionList` object containing the metadata of the successfuy updated privilege groups.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/privilegegroups?op=update`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<privilegeGroupDefinitionList>`
	`  <privilegeGroupDefinition name="Sample Privilege Group">`		

```
        <privilegeDefinitionList>
            <privilegeDefinition name="Check In New"
id="1000001">
                <group id="100000">General</group>
                <statusBased>false</statusBased>
                <systemDefined>true</systemDefined>
                <contentPrivilege>true</contentPrivilege>
                <contentTypeInfoList>
                    <contentTypeInfo id="1100">
                        <name>Asset</name>
                    </contentTypeInfo>
                    <contentTypeInfo id="1100">
                        <name>CustomeContentType_49933</name>
                    </contentTypeInfo>
                </contentTypeInfoList>
            </privilegeDefinition>
        </privilegeDefinitionList>
        <parentGroup id="-1"/>
    </privilegeGroupDefinition>
</privilegeGroupDefinitionList>
```

deletePrivilegeGroup

Function	Deletes Given Privilege Group.	
Description	This request deletes the given privilege group.	
Type	POST	
URL Format	`.../privilegegroups/{privilegeGroupIdOrName}?op=delete`	
Java API	**Method Name**	**Class Name**
	`deletePrivilegeGroup`	`com.quark.qpp.service.facadeClass AdminFacade`
Parameters	This request requires no input parameters.	
Returns	A status of 200 is returned in response to a successful request execution.	
Examples	**URL:** `http://serverName:serverPortt/rest/service/admin/privilegegroups/{privilegeGroupIdOrName}?op=delete`	

addDomainValue

Function	Add a domain value.		
Description	This request is used to add a new value in domain under the mentioned parent value. If the parent value Id is -1, then the domain value will be added at root level.		
Type	POST		
URL Format	`.../domains/{domainIdOrName}/{newValueName}`		
Java API	**Method Name**		**Class Name**
	`addDomainValue`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**

	op=create	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	Parentvalue	string	Id/Name of the domain value under which new domain value is to be added.
	sourcereference	string	Source reference of the new domain value.
	position	string	Position among siblings where the domain value will be added.Can have values like 1, 5,....,(use -1 to add domain value at the end).
Returns	Returns metadata of the newly created domain value.		
Examples	**URL:** http://serverName:serverPort/rest/service/admin/domains/501/SampleQPPDomainVal_a2b **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <domainValue id="1"> <name>SampleQPPDomainVal_a2b</name> <valueHierarchy>Desktop</valueHierarchy> <sourceReference>source</sourceReference> </domainValue>		

getDomainValue

Function	Get Domain Value.	
Description	This request returns desired DomainValue information.	
Type	GET	
URL Format	.../domains/{domainIdOrName}/{valueId}	
Java API	**Method Name**	**Class Name**
	getDomainValue	com.quark.qpp.service.facade Class AdminFacade
Parameters	This request requires no input parameters.	
Returns	Returns metadata of the specified domain value.	
Examples	**URL:** http://serverName:serverPort/rest/service/admin/domains/Users/10 **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <domainValue id="10"> <name>Admin2</name> <valueHierarchy>Desktop</valueHierarchy> </domainValue>	

getDomainValueXml

Function	Get XML Value for Given Domain.
Description	This request returns xml value of the given domain value.

Type	GET		
URL Format	`.../domains/{domainIdOrName}/{valueId}`		
Java API	**Method Name**		**Class Name**
	`getDomainValueXml`		`com.quark.qpp.service.facadeClassSessionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=fetchxml`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	Returns xml value of the given domain value.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/domains/{domainIdOrName}/{valueId}?op=fetchxml`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<sampleXMLvalue/>`		

setDomainValueXml

Function	Set XML Value for Given Domain.		
Description	This request sets xml for the given domain value.		
Type	POST		
URL Format	`.../domains/{domainIdOrName}/{valueId}`		
Java API	**Method Name**		**Class Name**
	`setDomainValueXml`		`com.quark.qpp.service.facadeClassSessionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=setxmlvalue`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`valueId=[]`	string	Id of the domain value whose xml is to be set.
	`xmlvalue=[XML Dat]`	string	XML Content.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/domains/{domainIdOrName}/{valueId}?op=setxmlvalue&valueid={Id of Domain Value}&xmlvalue={XML Data}`		

searchDomainValues

Function	Search Domain with XML Values with matching Xpath Expression.		
Description	This request finds domain values whose xml matches with the given xpath.		
Type	GET		
URL Format	`.../domains/{domainIdOrName}`		
Java API	**Method Name**		**Class Name**
	`searchDomainValues`		`com.quark.qpp.service.facadeClass SessionFacade`
Parameters	**Name**	**Type**	**Description**
	`op=searchvalues`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`xpath=[]`	string	Expression to be used for finding required domain values.
Returns	Returns a `domainValueList` object whose xml matches with the given XPath.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/domains/{domainId OrName} ?op=searchvalues&xpath={xpath to match the XML Value for Defined Domain}`		

updateDomainValue

Function	Update a domain value.		
Description	This request is used to updates the name and source reference of the domain value with the given ID.		
Type	POST		
URL Format	`.../domains/{domainIdOrName}/{ValueId}`		
Java API	**Method Name**		**Class Name**
	`updateDomainValue`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=update`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`name`	string	The new name for the domain value.
	`sourcereference`	string	The source reference of the domain value to be updated.
Returns	Returns metadata of the successfully updated domain value.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/domains/501/1?op=`		

	update &name=NewName&sourcereference=updatedSourceRef
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<domainValue id="1">`
	`  <name>newName</name>`
	`  <valueHierarchy>Desktop</valueHierarchy>`
	`  <sourceReference>updatedSourceRef</sourceReference>`
	`</domainValue>`

deleteDomainValue

Function	Delete a domain value.		
Description	This request is used to delete the value of a domain.		
Type	POST		
URL Format	`.../domains/{domainIdOrName}/{ValueId}`		
Java API	**Method Name**		**Class Name**
	`deleteDomainValue`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/domains/501/1?op=delete`		

getFavorites

Function	Get all Favorites.	
Description	This request returns all of the favorites items of the logged-on user.	
Type	GET	
URL Format	`../admin/favorites`	
Java API	**Method Name**	**Class Name**
	`getFavorites`	`com.quark.qpp.service.facade Class AdminFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `favoriteItemList` object containing metadata of all of the favorites items for the logged-in user.	
Examples	**URL:** `http://serverName:serverPort/rest/service/admin/favorites` **Result:**	

	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<favoriteItemlist>`
	`  <favoriteItem position="1" type="QUERY" id="100">Default</fa-`
	`voriteItem>`
	`  <favoriteItem position="2" type="COLLECTION"`
	`id="10">Home/Pictures</favoriteItem>`
	`</favoriteItemlist>`

setFavorites

Function	Set Favorites.		
Description	This request sets the favorites items for the logged-in user.		
Type	POST		
URL Format	`.../favorites`		
Java API	**Method Name**		**Class Name**
	`setFavorites`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op=setfavorites`	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`favoriteitemlist=[xml]`	string	Object containing the metadata of the preference valuess to be updated.
Returns	Returns a `favoriteItemList` object containing the metadata of favorite items successfully set for logged-in user.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/admin/preferencevalues?op=update&preferencevaluelist=<xml/>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<favoriteItemlist>`
	`  <favoriteItem position="1" type="QUERY" id="100">Default</fa-`
	`voriteItem>`
	`  <favoriteItem position="2" type="COLLECTION"`
	`id="10">Home/Pictures</favoriteItem>`
	`</favoriteItemlist>`		

getWorkflowInitiatingStatuses

Function	Get Workflow initial statuses.
Description	This request returns a list of “Workflow Initiating Statuses” for the given workflow. If the roleIdOrName parameter is specified then “Workflow Initiating Statuses” will be returned on the basis of overridden status transitions for the given userclass, otherwise the default initiating statuses are returned.
Type	GET

URL Format	`.../workflows/{workflowIdOrName}`		
Java API	**Method Name**		**Class Name**
	`getWorkflowInitiatingStatuses`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`workflowinitiatingstatuses =true`	boolean	
	`role={ roleIdOrName }`	string	The ID or Name of the role whose Workflow Initiating Statuses you wish to be retrieved.
Returns	Returns a `statusInfoList` object containing its metadata.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/workflows/Workflow1?workflowinitiatingstatuses=true&role=10`
	Result:
	`<?xml version="1.0" encoding="UTF-8"?>`
	`<statusInfoList>`
	`  <statusInfo id="10">`
	`  <name>Status1</name>`
	`  <domainId>6</domainId>`
	`  <colorRed>50</colorRed>`
	`  <colorGreen>150</colorGreen>`
	`  <colorBlue>10</colorBlue>`
	`  <position>1</position>`
	`  <redlineTracking>false</redlineTracking>`
	`  <checkInLayoutEvaluation>false</checkInLayoutEvaluation>`
	`  <outputLayoutEvaluation>false</outputLayoutEvaluation>`
	`  <privilegeInfoList>`
	`  <privilegeInfo>`
	`  <roleName>Administrator</roleName>`
	`  <roleId>1</roleId>`
	`  <overridden>false</overridden>`
	`  <contentTypePrivList/>`
	`  </privilegeInfo> ... </privilegeInfoList>`
	`  </statusInfo>`
	`</statusInfoList>`		

getStatusTransitions

Function	Get Status Transitions.	
Description	This request returns the status transitions set for the given workflow. If the statusIdOrName parameter is specified, then status transitions for a particular status are returned. If the roleIdOrName parameter is specified, then the overridden transitions for the given role for particular workflow will be returned. Otherwise the default status transitions are returned.	
Type	GET	
URL Format	`.../workflows/{workflowIdOrName}`	
Java API	**Method Name**	**Class Name**
	`getStatusTransistions`	`com.quark.qpp.service.facade`

			Class AdminFacade
Parameters	**Name**	**Type**	**Description**
	statustransitions =true	boolean	
	status={statusIdOrNam e}	string	The ID or Name of the status whose Status Transitions you wish to be retrieved."
	role={ roleIdOrName}	string	The ID or Name of the role whose Status Transitions you wish to be retrieved ."
Returns	Returns an statusTransitionInfoList object containing a list of StatusTransitions set for the given workflow or workflow status for specified role.		

Examples	**URL:**
	http://serverName:serverPort/rest/service/workflows/Workflow1?statustransitions=true&role=20
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<statusTransitionInfoList>`
	`  <statusTransitionInfo id="10">`
	`  <name>Status1</name>`
	`  <workflowInitiatingStatus>true</workflowInitiatingStatus>`
	`  <nextPossibleStatusInfoList>`
	`  <nextPossibleStatusInfo id="11">`
	`  <name>Status2</name>`
	`  </nextPossibleStatusInfo>`
	`  </nextPossibleStatusInfoList>`
	`  </statusTransitionInfo>`
	`  <statusTransitionInfo id="11">`
	`  <name>Status2</name>`
	`  <workflowInitiatingStatus>false</workflowInitiatingSta-tus>`
	`  </statusTransitionInfo>`
	`</statusTransitionInfoList>`		

Publishing

The publishing capabilities of the API allow application developers to publish content and includes the following REST calls:

- *getPublishingChannels*
- *getAllPublishingChannels*
- *getDeliveryChannels*
- *publishAsset*
- *publishBasedOnURI*
- *publishBasedOnStreamContext*
- *createDatadoc*
- *getDatadocChannels*
- *fetchDatadocAssets*
- *executeDatadocChannel*
- *updateDatadoc*
- *detachDatadocChannel*

getPublishingChannels

Function	Get the publishing channels for a particular content type.	
Description	This request returns the metadata of all the available publishing channels for given content type. Null will be returned in case the channel doesn't exist for the given content type.	
Type	GET	
URL Format	`.../publish/publishingchannels/[content type name or ID]`	
Java API	**Method Name**	**Class Name**
	`getPublishingChannels`	`com.quark.qpp.service.facade Class PublishFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `publishingChannelList` object.	
Examples	**URL:** `http://serverName:serverPort/rest/service/publish/publishingchann`	

	`els/Picture`
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<publishingChannelList>`
	`  <publishingChannelInfo name="IM.JPG" id="convertImage">`
	`  <parameter>SOURCE_FILE_EXTENSION</parameter>`
	`  <parameter>OUTPUT_FILE_NAME</parameter>`
	`  </publishingChannelInfo>`
	`  <publishingChannelInfo name="Extract Image Metadata" id="ex-`
	`tractImageAttributes">`
	`  <parameter>SOURCE_FILE_EXTENSION</parameter>`
	`  <parameter>OUTPUT_FILE_NAME</parameter>`
	`  </publishingChannelInfo>`
	`  <publishingChannelInfo name="EPS.JPG" id="epsPreview">`
	`  <parameter>SOURCE_FILE_EXTENSION</parameter>`
	`  <parameter>OUTPUT_FILE_NAME</parameter>`
	`  </publishingChannelInfo>`
	`</publishingChannelList>`

getAllPublishingChannels

Function	Get all publishing channels.	
Description	This request returns the metadata of all existing Publishing Channels with their parameters.	
Type	GET	
URL Format	`.../publish/publishingchannels`	
Java API	**Method Name**	**Class Name**
	`getAllPublishingChannels`	`com.quark.qpp.service.facade Class PublishFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `publishingChannelList` object containing an array of all existing publishing channels.	

Examples	**URL:**
	
	`http://serverName:serverPort/rest/service/publish/publishingchannels`
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<publishingChannelList>`
	`  <publishingChannelInfo name="PDF" id="qxpPdf">`
	`  <parameter>ASSET_ID</parameter> .... </publishingChan-`
	`nelInfo>`
	`  <publishingChannelInfo name="JPEG.ZIP" id="qxpImage">`
	`  <parameter>ASSET_ID</parameter> .... </publishingChan-`
	`nelInfo> ..... </publishingChannelList>`	

getDeliveryChannels

Function	Get all delivery channels.

Description	This request returns the metadata of all existing Delivery Channels with their parameters.	
Type	GET	
URL Format	`.../publish/deliverychannels`	
Java API	**Method Name**	**Class Name**
	`getDeliveryChannels`	`com.quark.qpp.service.facade Class PublishFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `publishingChannelList` object containing an array of all existing delivery channels.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/publish/deliverychannels`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<publishingChannelList>`
	`<publishingChannelInfo name="Publish to Sharepoint" id="checkInToSharepoint">`
	`<parameter>ASSET_ID</parameter> .... </publishingChannelInfo>`
	`<publishingChannelInfo name="Publish to FileNet" id="checkInToFileNet">`
	`<parameter>ASSET_ID</parameter> .... </publishingChannelInfo> ..... </publishingChannelList>`	

publishAsset

Function	Publish an asset.		
Description	This request returns the stream after publishing an asset into a specified format using the publishing channel.		
Type	GET		
URL Format	`.../publish/[asset ID] or /publishbypath/[asset path]`		
Java API	**Method Name**		**Class Name**
	`publishAsset`		`com.quark.qpp.service.facade Class PublishFacade`
Parameters	**Name**	**Type**	**Description**
	`publishingchannel =[Publishing Channel] & [Publishing Channel Parameters]`	string	The publishing channel used for publishing the content along with publishing parameters (specific to provided publishing channel)(Mandatory)
	`majorversion`	int	The major version number of the asset to be published. Optional.
	`minorversion`	int	The minor version number of the asset to be published. Optional.
Returns	Returns the filestream after publishing an asset into required format.		

Examples	**URL:** `http://serverName:serverPort/publish/[asset ID]?publishingchannel=qxpPdf&majorversion=1&minorversion=0` **Result:** The filestream after publishing an asset into required format.

Function	Deliver an asset.		
Description	This request delivers a QPP or Non-QPP asset at any fileserver, sharepoint server, filenet or documentum.		
Type	GET		
URL Format	`.../publish/[asset ID] or /publishbypath/[asset path]`		
Java API	**Method Name**		**Class Name**
	`publishAsset`		`com.quark.qpp.service.facade Class PublishFacade`
Parameters	**Name**	**Type**	**Description**
	`deliverychannel =[Delivery Channel] & [Delivery Channel Parameters]`	string	The delivery channel used to deliver a QPP asset or published content to any of the available delivery channels. (Mandatory)Delivery channel parameters are specific to the channel and all are mandatory.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/publish/[asset ID]?deliverychannel=checkInToFileNet&parameter1=value1?...`		

Function	Publish and deliver an asset.		
Description	This request publishes any QPP or Non-QPP asset into the required format and delivers the published content at any fileserver, sharepoint server, filenet or documentum at the same time.		
Type	GET		
URL Format	`.../publish/[asset ID]/publish/[asset ID] or /publish//publishbypath/[asset path]`		
Java API	**Method Name**		**Class Name**
	`publishAsset`		`com.quark.qpp.service.facade Class PublishFacade`
Parameters	**Name**	**Type**	**Description**
	`publishingchannel =[Publishing Channel] & [Publishing Channel Parameters]`	string	The publishing channel used for publishing the content along with publishing parameters (specific to provided publishing channel).
	`deliverychannel`	string	The delivery channel used to deliver the

	=[Delivery Channel] & [Delivery Channel Parameters]		QPP asset or published content to any of the available delivery channels. Delivery channel parameters are specific to the channel and all are mandatory.
	majorversion	int	The major version number of the asset to be published. Optional.
	minorversion	int	The minor version number of the asset to be published. Optional.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** http://serverName:serverPort/publish/[asset ID]?publishingchannel=qxpPdf& deliverychannel=checkInToFileNet& parameter1=value1?...		

publishBasedOnURI

Function	Publish a non-Platform resource.		
Description	Takes input in the form of URI.		
Type	GET		
URL Format	.../publish		
Java API	**Method Name**		**Class Name**
	publishBasedOnURI		com.quark.qpp.service.facade Class PublishFacade
Parameters	**Name**	**Type**	**Description**
	uri	string	Input a non QPP resource available on any of the URIs. Supported URIs are file, qpp, classpath, SharePoint and http.
	publishingchannel	string	The publishing channel used for publishing the content along with publishing parameters (specific to provided publishing channel).
Returns	Returns the filestream of output after publishing an asset to the required format.		
Examples	**URL:** http://serverName:serverPort/rest/service/publish?uri=file:\\server1\\abc\\abc.xml40& publishingchannel=busDocPdf **Result:** The filestream of output after publishing an asset to the required format.		

Function	Deliver a non-Platform resource.
Description	Include the file in the request body or as file part. Takes input in the form of URI.
Type	GET

URL Format	`.../publish`		
Java API	**Method Name**		**Class Name**
	`publishBasedOnURI`		`com.quark.qpp.service.facade Class PublishFacade`
Parameters	**Name**	**Type**	**Description**
	`uri`	string	Input the non QPP resource available on any of the URIs. Supported URIs are file, qpp, classpath, SharePoint and http.
	`deliverychannel`	string	The delivery channel used to deliver the QPP asset or published content to any of the available delivery channels.Delivery channel parameters are specific to the channel and all are mandatory.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/publish?uri=qpp://assets/40&deliverychannel=FileSystem&LOCATION=C:\DATA\temp`		

publishBasedOnStreamContext

Function	Publish a non-platform resource.		
Description	This request will publish a non-platform resource. Takes input in the form of a file stream.		
Type	POST		
URL Format	`.../publish`		
Java API	**Method Name**		**Class Name**
	`publishBasedOnStreamContext`		`com.quark.qpp.service.facade Class PublishFacade`
Parameters	**Name**	**Type**	**Description**
	`publishingchannel`	string	The publishing channel used for publishing the content along with publishing parameters (specific to provided publishing channel).
Returns	Returns the filestream of output after publishing an asset to the required format.		
Examples	**URL:** `http://serverName:serverPort/rest/service/publish?publishingchannel=busDocPdf` **Result:** The filestream of output after publishing an asset to the required format.		

Function	Deliver a non-Platform resource.
Description	Publish and deliver non-platform resource. Takes input in the form of a file stream.

Type	GET		
URL Format	`.../publish`		
Java API	**Method Name**		**Class Name**
	`publishBasedOnStreamContext`		`com.quark.qpp.service.facade Class PublishFacade`
Parameters	**Name**	**Type**	**Description**
	`deliverychannel`	string	The delivery channel used to deliver the QPP asset or published content to any of the available delivery channels.Delivery channel parameters are specific to the channel and all are mandatory.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	**URL:** `http://serverName:serverPort/rest/service/publish?deliverychannel=FileSystem&LOCATION=C:\DATA\temp`		

createDatadoc

Function	Create a Datadoc.		
Description	This request creates a datadoc by executing the given datadoc channel and checks it in to the platform with the given metadata.		
Type	GET		
URL Format	`.../datadocchannels`		
Java API	**Method Name**		**Class Name**
	`createDatadoc`		`com.quark.qpp.service.facadeClass PublishFacade`
Parameters	**Name**	**Type**	**Description**
	`op=createdatadoc`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`datadocchannel={DataDoc Channel Id}`	string	The Datadoc channel. (Mandatory)
	`createMinorVersion={true|false}`	boolean	Indicates whether you should create a Minor version.
	`attributevalues=<XML/>`	string	See “Request/Response parameter formats.”
Returns	Returns the metadata of newly created Datadoc asset.		
Examples	**URL:** `http://serverName:serverPort/rest/service/publish/?op=createdatadoc &datadocchannel=createDataDocAssetFromHttpUrl &CONTENT_URL=http://Server:Port/SampleBusDoc.xml &attributevalues=`		

```
<?xml version="1.0" encoding="UTF-8"?>
<attributeValueList>
    <attributeValue valueId="0" type="text" name="Name"
id="2">SampleBusDoc827574.xml </attributeValue>
    <attributeValue valueId="1" type="popup" name="Routed to"
id="25">Admin</attributeValue>
    <attributeValue valueId="10" type="popup" name="Status"
id="24">Status1</attributeValue>
    <attributeValue valueId="1" type="popup" name="Collection"
id="55">Home</attributeValue>
    <attributeValue valueId="10" type="popup" name="Workflow"
id="54">Workflow1</attributeValue>
    <attributeValue valueId="1104" type="popup" name="Content
Type" id="62">Business Document</attributeValue>
    <attributeValue valueId="0" type="text" name="File extension"
id="11">xml</attributeValue>
    <attributeValue valueId="0" type="text" name="Original file-
name" id="33">SampleBusDoc827.xml</attributeValue>
</attributeValueList>
```

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assetInfo id="61">
    <assetVersion minorVersion="0" majorVersion="1"/>
    <attributeValueList>
        <attributeValue type="numeric" name="Id" id="1">61</at-
tributeValue>
        <attributeValue type="text" name="Name" id="2">SampleBus-
Doc827574.xml</attributeValue>
        <attributeValue type="dateTime" name="Created"
id="3">2014-10-28T11:39:46Z</attributeValue>
        <attributeValue type="dateTime" name="Last modified"
id="4">2014-10-28T11:39:46Z</attributeValue>
        <attributeValue multiValued="false" valueId="1"
type="popup" name="Creator" id="5">Admin</attributeValue>
        <attributeValue multiValued="false" valueId="1"
type="popup" name="Last modifier" id="6">Admin</attributeValue>
        <attributeValue type="numeric" name="Major version"
id="7">1</attributeValue>
        <attributeValue type="text" name="File path"
id="8">/00/3d/61.1-0.1.1.xml</attributeValue>
        <attributeValue type="text" name="File extension"
id="11">xml</attributeValue>
        <attributeValue type="numeric" name="File size"
id="12">1591</attributeValue>
        <attributeValue type="boolean" name="Is checked out"
id="13">false</attributeValue>
        <attributeValue type="dateTime" name="Last content up-
date" id="19">2014-10-28T11:39:46Z</attributeValue> ..... </at-
tributeValueList>
</assetInfo>
```

getDatadocChannels

Function	Get all datadoc channels.	
Description	This request returns all channels of type ‘datadoc’ that exist in the system.	
Type	GET	
URL Format	`.../datadocchannels`	
Java API	**Method Name**	**Class Name**

	getDatadocChannels	com.quark.qpp.service.facade Class PublishFacade
Parameters	This request requires no input parameters.	
Returns	Returns a publishingChannelList object containing the metadata of channels of type datadoc.	
Examples	**URL:** http://serverName:serverPort/rest/service/publish/datadocchannels **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <publishingChannelList> <publishingChannelInfo name="Create Data Document From HTTP URL" id="createDataDocAssetFromHttpUrl"> <parameter>CONTENT_URL</parameter> <parameter>XSL_URI</parameter> </publishingChannelInfo> <publishingChannelInfo name="Fetch Excel Data Object" id="fetchExcelDataObject"> <parameter>ASSET_ID</parameter> <parameter>WORKSHEET_NAME</parameter> <parameter>DATA_OBJECT_TYPE</parameter> <parameter>DATA_OBJECT_NAME</parameter> <parameter>OUTPUT_FORMAT</parameter> <parameter>OUTPUT_FORMAT_PROPERTIES</parameter> </publishingChannelInfo> </publishingChannelList>	

Function	Get DataDoc Channels for Desired Content-Type.	
Description	This request returns channels of type DataDoc which are mapped to the given content type.	
Type	GET	
URL Format	.../datadocchannels/{contentTypeIdOrName}	
Java API	**Method Name**	**Class Name**
	getDatadocChannels	com.quark.qpp.service.facade Class PublishFacade
Parameters	This request requires no input parameters.	
Returns	Returns a publishingChannelList object containing the metadata of channels of type 'datadoc' mapped to a specific content type.	
Examples	**URL:** http://serverName:serverPort/rest/service/publish/datadocchannels/{Contenty-Type Id } **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <publishingChannelList> <publishingChannelInfo name="Fetch Excel Data Object" id="fetchExcelDataObject"> <parameter>ASSET_ID</parameter> <parameter>WORKSHEET_NAME</parameter> <parameter>DATA_OBJECT_TYPE</parameter> <parameter>DATA_OBJECT_NAME</parameter> <parameter>OUTPUT_FORMAT</parameter>	

	`<parameter>OUTPUT_FORMAT_PROPERTIES</parameter>`
	`</publishingChannelInfo>`
	`</publishingChannelList>`

fetchDatadocAssets

Function	Fetch Datadoc Assets.		
Description	This request returns the metadata of the datadoc assets which had been created by executing the given channel and parameters.		
Type	GET		
URL Format	`...publish/`		
Java API	**Method Name**		**Class Name**
	`fetchDatadocAssets`		`com.quark.qpp.service.facadeClassPublishFacade`
Parameters	**Name**	**Type**	**Description**
	`op=fetchdatadocassets`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`datadocchannel={DataDoc Channel Id}`	string	The Datadoc channel. (Mandatory)
Returns	Returns the metadata of the Datadoc asset created through the supplied datadoc channel.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/publish?op=fetchdatadocassets&datadocchannel=createDataDocAssetFromHttpUrl`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfo id="61">`
	`<assetVersion minorVersion="0" majorVersion="1"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">61</attributeValue>`
	`<attributeValue type="text" name="Name" id="2">SampleBusDoc827574.xml</attributeValue>`
	`<attributeValue type="dateTime" name="Created" id="3">2014-10-28T11:39:46Z</attributeValue>`
	`<attributeValue type="dateTime" name="Last modified" id="4">2014-10-28T11:39:46Z</attributeValue>`
	`<attributeValue multiValued="false" valueId="1" type="popup" name="Creator" id="5">Admin</attributeValue>`
	`<attributeValue multiValued="false" valueId="1" type="popup" name="Last modifier" id="6">Admin</attributeValue>`
	`<attributeValue type="numeric" name="Major version" id="7">1</attributeValue>`
	`<attributeValue type="text" name="File path" id="8">/00/3d/61.1-0.1.1.xml</attributeValue>`
	`<attributeValue type="text" name="File extension" id="11">xml</attributeValue>`
	`<attributeValue type="numeric" name="File size" id="12">1591</attributeValue>`		

	`<attributeValue type="boolean" name="Is checked out" id="13">false</attributeValue>`
	`<attributeValue type="dateTime" name="Last content update" id="19">2014-10-28T11:39:46Z</attributeValue> ..... </attributeValueList>`
	`</assetInfo>`

executeDatadocChannel

Function	Execute Datadoc Channels.		
Description	This request executes the given datadoc channel and writes the resulted datadoc content to the given outputstream.		
Type	GET		
URL Format	`...publish/`		
Java API	**Method Name**		**Class Name**
	`executeDatadocChannel`		`com.quark.qpp.service.facadeClassPublishFacade`
Parameters	**Name**	**Type**	**Description**
	`op=executedatadocchannel`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`datadocchannel={DataDoc Channel Id}`	string	The Datadoc channel. (Mandatory)
	`channelparameters=<XML/>`	string	Channel parameters.
Returns	Resultant output is returned in the response body stream.		
Examples	**URL:** `http://serverName:serverPort/rest/service/publish?op=executedatadocchannel&datadocchannel=fetchExcelDataObject&...`		

updateDatadoc

Function	Update a Datadoc.		
Description	This request updates the existing datadoc asset by creating its new version. Latest content of the asset is fetched by executing the desired datadoc channel.		
Type	POST		
URL Format	`...publish/{assetId}`		
Java API	**Method Name**		**Class Name**
	`updateDatadoc`		`com.quark.qpp.service.facadeClassPublishFacade`
Parameters	**Name**	**Type**	**Description**
	`op=updatedatadoc`	string	The op parameter allows you to specify which operation is to be performed.

			(Mandatory)
	`datadocchannel={DataDoc Channel Id}`	string	The Datadoc channel. (Mandatory)
	`createMinorVersion={true|false}`	boolean	Indicates whether you should create a Minor version.
	`attributevalues=<XML/>`	string	See "Request/Response parameter formats."
Returns	Returns the metadata of the successfully updated Datadoc asset.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/publish/?op=createdatadoc &datadocchannel=createDataDocAssetFromHttpUrl &CONTENT_URL=http://Server:Port/SampleBusDoc.xml &attributevalues=`
	`<?xml version="1.0" encoding="UTF-8"?>`
	`<attributeValueList>`
	`<attributeValue valueId="0" type="text" name="Name" id="2">SampleBusDoc827574.xml </attributeValue>`
	`<attributeValue valueId="1" type="popup" name="Routed to" id="25">Admin</attributeValue>`
	`<attributeValue valueId="10" type="popup" name="Status" id="24">Status1</attributeValue>`
	`<attributeValue valueId="1" type="popup" name="Collection" id="55">Home</attributeValue>`
	`<attributeValue valueId="10" type="popup" name="Workflow" id="54">Workflow1</attributeValue>`
	`<attributeValue valueId="1104" type="popup" name="Content Type" id="62">Business Document</attributeValue>`
	`<attributeValue valueId="0" type="text" name="File extension" id="11">xml</attributeValue>`
	`<attributeValue valueId="0" type="text" name="Original file-name" id="33">SampleBusDoc827.xml</attributeValue>`
	`</attributeValueList>`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfo id="61">`
	`<assetVersion minorVersion="0" majorVersion="1"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">61</at-tributeValue>`
	`<attributeValue type="text" name="Name" id="2">SampleBus-Doc827574.xml</attributeValue>`
	`<attributeValue type="dateTime" name="Created" id="3">2014-10-28T11:39:46Z</attributeValue>`
	`<attributeValue type="dateTime" name="Last modified" id="4">2014-10-28T11:39:46Z</attributeValue>`
	`<attributeValue multiValued="false" valueId="1" type="popup" name="Creator" id="5">Admin</attributeValue>`
	`<attributeValue multiValued="false" valueId="1" type="popup" name="Last modifier" id="6">Admin</attributeValue>`
	`<attributeValue type="numeric" name="Major version" id="7">1</attributeValue>`
	`<attributeValue type="text" name="File path" id="8">/00/3d/61.1-0.1.1.xml</attributeValue>`
	`<attributeValue type="text" name="File extension" id="11">xml</attributeValue>`
	`<attributeValue type="numeric" name="File size" id="12">1591</attributeValue>`		

	`<attributeValue type="boolean" name="Is checked out" id="13">false</attributeValue>`
	`<attributeValue type="dateTime" name="Last content update" id="19">2014-10-28T11:39:46Z</attributeValue> ..... </attributeValueList>`
	`</assetInfo>`

detachDatadocChannel

Function	Detach a DataDoc asset from its source.		
Description	This request detaches or unlinks the given datadoc asset from its source content and creates a new version of the asset. After detaching/unlinking, the given asset will no more be a datadoc asset and thus can be checked-out and checked-in by the end user like other conventional assets in the Platform.		
Type	GET		
URL Format	`.../publish/{assetId}/`		
Java API	**Method Name**		**Class Name**
	`detachDatadocChannel`		`com.quark.qpp.service.facadeClassPublishFacade`
Parameters	**Name**	**Type**	**Description**
	`op=detachdatadoc`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`attributevalues=<XML/>`	string	The Datadoc channel. (Mandatory)
	`createMinorVersion={true|false}`	boolean	Indicates whether you should create a Minor version.
Returns	Returns the metadata of newly created asset.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/publish?op=detachdatadoc&attributevalues=<xml/>&createminorversion=false`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<assetInfo id="61">`
	`<assetVersion minorVersion="0" majorVersion="1"/>`
	`<attributeValueList>`
	`<attributeValue type="numeric" name="Id" id="1">61</attributeValue>`
	`<attributeValue type="text" name="Name" id="2">SampleBusDoc827574.xml</attributeValue>`
	`<attributeValue type="dateTime" name="Created" id="3">2014-10-28T11:39:46Z</attributeValue>`
	`<attributeValue type="dateTime" name="Last modified" id="4">2014-10-28T11:39:46Z</attributeValue>`
	`<attributeValue multiValued="false" valueId="1" type="popup" name="Creator" id="5">Admin</attributeValue>`
	`<attributeValue multiValued="false" valueId="1" type="popup" name="Last modifier" id="6">Admin</attributeValue>`
	`<attributeValue type="numeric" name="Major version" id="7">1</attributeValue>`		

	`<attributeValue type="text" name="File path" id="8">/00/3d/61.1-0.1.1.xml</attributeValue>`
	`<attributeValue type="text" name="File extension" id="11">xml</attributeValue>`
	`<attributeValue type="numeric" name="File size" id="12">1591</attributeValue>`
	`<attributeValue type="boolean" name="Is checked out" id="13">false</attributeValue>`
	`<attributeValue type="dateTime" name="Last content update" id="19">2014-10-28T11:39:46Z</attributeValue> ..... </attributeValueList>`
	`</assetInfo>`

Script execution

The script execution capabilities of the API allow application developers to execut Platform scripts and includes the following REST calls:

- *createScript*
- *getScriptContent*
- *getAllScripts*
- *executeScript*
- *deleteScript*

createScript

Function	Create a script.		
Description	This request creates a new script with the given content and name at the platform server. In the script execution information it is mandatory to mention the trigger type and corresponding eventInfo or scheduleInfo depending upon the trigger type. Include the file in the request body or as file part.		
Type	POST		
URL Format	`.../scripts/{scriptName}`		
Java API	**Method Name**		**Class Name**
	`createScript`		`com.quark.qpp.service.facade Class ScriptFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)This operation requires parent asset to be in a Checked-Out state.
	`language`	long	The scripting language
	`scriptexecutioninfo =<xml/>`	string	The set of attribute values. See “Request/Response parameter formats.”
Returns	Returns an `scriptInfo` object containing metadata of the successfully created script.		
Examples	**URL:** `http://serverName:serverPort/rest/service/scripts?op=create`		

	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<scriptInfo id="12">`
	`  <name>abc.js</name>`
	`  <language>javascript</language>`
	`  <createdDate>2015-01-06T10:32:31Z</createdDate>`
	`  <lastModified>2015-01-06T10:32:31Z</lastModified>`
	`  <lastExecutionLog>NOT_RUN</lastExecutionLog>`
	`  <creator id="1">Admin</creator>`
	`  <lastModifiedBy id="1">Admin</lastModifiedBy>`
	`  <scriptExecutionInfo triggerType="MANUAL"/>`
	`</scriptInfo>`

getScriptContent

Function	Get the content of a script.	
Description	This request returns the content of the required script in text format.	
Type	GET	
URL Format	`.../scripts/[script ID or name]`	
Java API	**Method Name**	**Class Name**
	`getScriptContent`	`com.quark.qpp.service.facade Class ScriptFacade`
Parameters	This request requires no input parameters.	
Returns	Returns the actual content of the required script in text format.	
Examples	**URL:** `http://serverName:serverPort/rest/service/scripts/QppFunctions importPackage(Packages.com.quark.qpp.filetransfergateway.service); importPackage(Packages.java.io); importPackage(Packages.com.quark.qpp.core.asset.service.constants); importPackage(Packages.com.quark.qpp.core.query.service.dto); importPackage(Packages.com.quark.qpp.core.query.service.constants); importPackage(Packages.com.quark.qpp.core.content.service.constants); importPackage(com.quark.qpp.common.dto); importPackage(com.quark.qpp.common.constants); .....` **Result:** The actual content of the required script in text format.	

getAllScripts

Function	Get all scripts.
Description	This request returns all scripts stored at the platform server.
Type	GET
URL Format	`.../scripts/`

Java API	**Method Name**	**Class Name**
	`getAllScripts`	`com.quark.qpp.service.facade Class ScriptFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `scriptInfoList` object that contains the metadata of all scripts. See the table in the "Request/Response parameter formats" chapter.	

Examples	**URL:**
	`http://serverName:serverPort/rest/service/scripts`
	Result:
	`<scriptInfoList>`
	`  <scriptInfo id="11">`
	`  <name>GetSavedSearchResults</name>`
	`  <language>javascript</language>`
	`  <createdDate>2015-01-06T10:23:37Z</createdDate>`
	`  <lastModified>2015-01-06T10:23:37Z</lastModified>`
	`  <lastExecutionLog>NOT_RUN</lastExecutionLog>`
	`  <creator id="1">Admin</creator>`
	`  <lastModifiedBy id="1">Admin</lastModifiedBy>`
	`  <scriptExecutionInfo triggerType="MANUAL"/>`
	`  </scriptInfo> ... </scriptInfoList>`	

executeScript

Function	Execute a script.		
Description	This request executes a specified script that is already saved on the QPP Server.		
Type	POST		
URL Format	`... /scripts/[script ID or name]`		
Java API	**Method Name**		**Class Name**
	`executeScript`		`com.quark.qpp.service.facade Class ScriptFacade`
Parameters	**Name**	**Type**	**Description**
	`op=execute`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This operation executes the complete script.
Returns	The response depends upon the output of the script.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/scripts/GetSavedSearchResults?op=execute`
	Result:
	`Total No Of Assets : 4 Name: Tulips Collection: Home Workflow: Workflow1 Status: Status1 Routed To: Admin Name: Duplicate_asset Collection: Home Workflow: Workflow1 Status: Status1 Routed To: Admin Name: 1quark2.jpg Collection: Home Workflow: Workflow1 Status: Status1 Routed To: Admin Name: 15116.qxp Collection: Home`		

	`Workflow: Workflow1 Status: Status1 Routed To: Admin`

Function	Execute a script method.		
Description	This request executes a specified method of a specified script that is already saved on the QPP Server.		
Type	POST		
URL Format	`.../scripts/[script ID or name]`		
Java API	**Method Name**		**Class Name**
	`executeScript`		`com.quark.qpp.service.facade Class ScriptFacade`
Parameters	**Name**	**Type**	**Description**
	`op=execute&`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)This operation executes given the method of the script along with the given parameters.
	`methodname=[method name]&`	string	Specify the method name to be executed. (Mandatory)
	`parameters=x,y,z`	long or string	Specify the method's parameters if required. (Optional)
Returns	The response depends upon the output of the script.		
Examples	**URL:** `http://serverName:serverPort/rest/service/scripts/Sample?op=execute&methodname=abc` **Result:** The response depends upon the output of the script.		

deleteScript

Function	Delete a script.		
Description	This request deletes the specified script.		
Type	POST		
URL Format	`.../scripts/{scriptName}`		
Java API	**Method Name**		**Class Name**
	`deleteScript`		`com.quark.qpp.service.facade Class ScriptFacadeClass AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op=delete`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory) This request will lock the asset internally and delete it from system.

Returns	A status of 200 is returned in response to a successful request execution.
Examples	**URL:** `http://serverName:serverPort/rest/service/scripts/12?op=delete`

Archive and Restore management

The archive and restore management capabilities of the API allow application developers to archive assets (metadata and its high_res) at the collection/query level and provides extensibility to restore/import the same assets into a different QPP server environment. This includes the following REST calls:

- *archiveCollectionAssets*
- *archiveQueryAssets*
- *archiveAssets*
- *restoreAssets*

archiveCollectionAssets

Function	Archive collection assets.		
Description	This request archives all assets belonging to a particular collection, from the given collection ID or complete collection path.		
Type	GET		
URL Format	`../archive/collectionassets?..`		
Java API	**Method Name**		**Class Name**
	`archiveCollectionAssets`		`com.quark.qpp.service.facade Class ArchiveFacade`
Parameters	**Name**	**Type**	**Description**
	`collection`	string	Name/ID of the collection. (Mandatory)
	`destinationfolder`	string	The path. Any file path that is accessible to the server, it could be on a network. (Mandatory)
	`allversions=[true | false]`	boolean	Defaults to true. (Optional) If true, the server will archive all available versions of all assets.
	`archiverelations=[true | false]`	boolean	Defaults to true. (Optional) If true, the server will maintain all the relations of assets to be archived.
	`deleteafterarchive=[true | false]`	boolean	Defaults to false. (Optional) If true, the server will delete all the assets after archiving it.
Returns	Returns an `archiveRestoreInfo` object in XML format.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/archive/collectionassets?collection=Home&destinationfolder=C:\QPP Archive`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <archiveRestoreInfo failureCount="0" successCount="50"> <summary> Archive Summary Log Date: 5/01/13 17:17 PM Archive location: C:\QPP Archive Total assets to be archived: 50 Number of assets successfully archived: 50 Number of assets failed to archive: 0 Per Version Listing of Assets ---------- Asset Versions Successfully Archived: ---------- Home/SampleBusDocdc1.qxp, Version: 1.0 Home/Construct/Construct - Copy.zip, Version: 1.0 Home/Construct/Construct.zip, Version: 1.0 Home/Construct/Construct1/Construct.rar, Version: 1.0 Home/Construct/Construct1/dineshjj.xml, Version: 1.0 </summary> </archiveRestoreInfo>`

archiveQueryAssets

Function	Archive resulted assets of any given saved search.		
Description	This request archives all assets returned in the query result for the given query name.		
Type	GET		
URL Format	`../archive/searchresults?..`		
Java API	**Method Name**		**Class Name**
	`archiveQueryAssets`		`com.quark.qpp.service.facade Class ArchiveFacade`
Parameters	**Name**	**Type**	**Description**
	`searchname`	string	Name of the saved search. (Mandatory)
	`destinationfolder`	string	The path. Any file path that is accessible to server, it could be on network. (Mandatory)
	`allversions=[true | false]`	boolean	Defaults to true.(Optional)If true, the server will archive all available versions of all assets.
	`archiverelations=[true | false]`	boolean	Defaults to true.(Optional)If true, the server will maintain all the relations of assets to be archived.
	`deleteafterarchive=[true | false]`	boolean	Defaults to false.(Optional)If true,the server will delete all the assets after archiving it.
Returns	Returns an `archiveRestoreInfo` object in XML format.		
Examples	**URL:** `http://serverName:serverPort/rest/service/archive/searchresults?searchname=Default&destinationfolder=C:\QPP Archive`		

	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<archiveRestoreInfo failureCount="0" successCount="50"> <summary>`
	`Archive Summary Log Date: 5/01/13 17:17 PM Archive location: C:\QPP Archive Total assets to be archived: 50 Number of assets successfully archived: 50 Number of assets failed to archive: 0 Per Version Listing of Assets ---------- Asset Versions Successfully Archived: ----------`
	`Home/SampleBusDocdc1.qxp, Version: 1.0 Home/Construct/Construct - Copy.zip, Version: 1.0 Home/Construct/Construct.zip, Version: 1.0 Home/Construct/Construct1/Construct.rar, Version: 1.0 Home/Construct/Construct1/dineshjj.xml, Version: 1.0 </summary>`
	`</archiveRestoreInfo>`

archiveAssets

Function	Archive specified assets.		
Description	This request archives an asset from the given asset id or asset path.		
Type	GET		
URL Format	`../archive/assets?..`		
Java API	**Method Name**		**Class Name**
	`archiveAssets`		`com.quark.qpp.service.facade Class ArchiveFacade`
Parameters	**Name**	**Type**	**Description**
	`assets`	long or string	The IDs or Collection Path of assets to be archived. (Mandatory) Comma separated IDs of assets (long), or a collection path of assets (string).
	`destinationfolder`	string	The path. Any file path that is accessible to server, it could be on network. (Mandatory)
	`allversions=[true | false]`	boolean	Defaults to true.(Optional)If true, the server will archive all available versions of all assets.
	`archiverelations=[true | false]`	boolean	Defaults to true.(Optional)If true, the server will maintain all the relations of assets to be archived.
	`deleteafterarchive=[true | false]`	boolean	Defaults to false.(Optional)If true, the server will delete all the assets after archiving it.
Returns	Returns an `archiveRestoreInfo` object in XML format.		
Examples	**URL:** `http://serverName:serverPort/rest/service/archive/assets?assets=1102,1229&destinationfolder=C:\QPP Archive` **Result:**		

	<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <archiveRestoreInfo failureCount="0" successCount="50"> <summary> Archive Summary Log Date: 5/01/13 17:17 PM Archive location: C:\QPP Archive Total assets to be archived: 50 Number of assets successfully archived: 50 Number of assets failed to archive: 0 Per Version Listing of Assets ---------- ---------- Asset Versions Successfully Archived: -- ---------- Home/SampleBusDocdc1.qxp, Version: 1.0 Home/Construct/Construct - Copy.zip, Version: 1.0 Home/Construct/Construct.zip, Version: 1.0 Home/Construct/Construct1/Construct.rar, Version: 1.0 Home/Con- struct/Construct1/dineshjj.xml, Version: 1.0 </summary> </archiveRestoreInfo>

restoreAssets

Function	Restore archived assets into the system.		
Description	This request restores assets from the given folder path. This request requires the same collection hierarchy with same set of workflow, users etc which existed at the time of archiving the assets.		
Type	GET		
URL Format	`../restore?...`		
Java API	**Method Name**		**Class Name**
	`restoreAssets`		`com.quark.qpp.service.facade Class RestoreFacade`
Parameters	**Name**	**Type**	**Description**
	`archivedpath`	string	The full path of the folder which contains the archived assets and its information. (Mandatory)
	`lookup`	string	Specify the lookup strategy to be used for fetching entities, like content type, relation type and attributes. Use name to restore across different servers, else use id. (Optional, Default value is id)
Returns	Returns an `archiveRestoreInfo` object in XML format.		
Examples	**URL:** `http://serverName:serverPort/rest/service/restore?archivedpath=C:\QPP Archive` **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <archiveRestoreInfo failureCount="0" successCount="50"> <summary> Archive Summary Log Date: 5/01/13 17:17 PM Archive location: C:\QPP Archive Total assets to be archived: 50 Number of assets successfully archived: 50 Number of assets failed to archive: 0 Per Version Listing of Assets ---------- ---------- Asset Versions Successfully Archived: -- ---------- Home/SampleBusDocdc1.qxp, Version: 1.0 Home/Construct/Construct - Copy.zip, Version: 1.0 Home/Construct/Construct.zip, Version: 1.0 Home/Construct/Construct1/Construct.rar, Version: 1.0 Home/Con		

	struct/Construct1/dineshjj.xml, Version: 1.0 </summary> </archiveRestoreInfo>

Office

The Office capabilities of the API allow application developers to fetch metadata from Excel worksheets, fetch information about Excel Data objects (like named range, dynamic range, tables and chartes) defined in a worksheet and get and transform Excel Data. The Office capabilities includes the following REST calls:

- *getExcelChart*
- *getExcelChartBasedOnStream*
- *getExcelChartBasedOnURI*
- *getExcelTable*
- *getExcelTableBasedOnStream*
- *getExcelTableBasedOnURI*
- *getExcelWorksheets*
- *getExcelWorksheetsBasedOnURI*
- *getExcelWorksheetsBasedOnStreamContext*
- *getChart*
- *getChartBasedOnURI*
- *getChartBasedOnStream*
- *getTable*
- *getTableBasedOnURI*
- *getTableBasedOnStreamContext*
- *getNamedRange*
- *getNamedRangeBasedOnURI*
- *getNamedRangeBasedOnStream*
- *getDynamicRange*
- *getDynamicRangeBasedOnURI*
- *getDynamicRangeBasedOnStream*
- *getVisioPagesInfo*
- *getVisioPagesInfoBasedOnURI*
- *getVisioPagesInfoBasedOnStreamContext*

- *getVisioPageOutput*
- *getVisioPageOutputBasedOnURI*
- *getVisioPageOutputBasedOnStreamContext*
- *getPowerPointSlidesInfo*
- *getPowerPointSlidesInfoBasedOnURI*
- *getPowerPointSlidesInfoBasedOnStreamContext*
- *getpowerPointSlideOutput*
- *getpowerPointSlideOutputBasedOnURI*
- *getpowerPointSlideOutputBasedOnStreamContext*
- *getPowerPointSlideRangeOutput*
- *getPowerPointSlideRangeOutputBasedOnURI*
- *getPowerPointSlideRangeOutputBasedOnStreamCongtext*

getExcelChart

Function	Get a Microsoft Excel chart based on asset Id		
Description	This request returns a Microsoft Excel Chart in the desired image format. Default image format is PNG.		
Type	GET		
URL Format	`...ofice/excelchart/{assetId}`		
Java API	**Method Name**		**Class Name**
	`getExcelChart`		`com.quark.qpp.service.facade.OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`assetId`	string	The AssetID.
	`majorversion=[x]`	string	Major version of the asset. (Optional)
	`minorversion=[x]`	string	Minor version of the asset. (Optional)
	`outputformat=[image, json, html]`	string	Image output format. (Optional)
	`contentdisposition`	string	Content disposition value to be set on the http response headers.
	`outputStream`	string	Output stream onto which output is to be written (for local API invocatins).
Returns	Returns a stream of the required format based on the given parameters.		
Examples	**URL:** `http://server:port/rest/service/office/excelchart/48?outputformat=image&imageformat=png` **Results:** A filestream of required output format based on given parameters.		

Function	Get a Microsoft Excel chart by path		
Description	This request returns a Microsoft Excel Chart in the desired image format. Default image format is PNG.		
Type	GET		
URL Format	`...ofice/excelchartbypath/{assetPath}`		
Java API	**Method Name**		**Class Name**
	`getExcelChart`		`com.quark.qpp.service.facade.OfficeByPathFacade`
Parameters	**Name**	**Type**	**Description**
	`assetPath`	string	The AssetPath.
	`majorversion=[x]`	string	Major version of the asset. (Optional)
	`minorversion=[x]`	string	Minor version of the asset. (Optional)
	`outputformat=[image, json, html]`	string	Image output format. (Optional)
	`contentdisposition`	string	Content disposition value to be set on the http response headers.
	`outputStream`	string	Output stream onto which output is to be written (for local API invocatins).
Returns	Returns a stream of the required format based on the given parameters.		
Examples	**URL:** `http://server:port/rest/service/office/excelchartbypath/Home/Chart1.xlsx?outputformat=image&imageformat=png` **Results:** A filestream of required output format based on given parameters.		

getExcelChartBasedOnStream

Function	Get a Microsoft Excel chart based on stream		
Description	This request returns a Microsoft Excel Chart in the desired image format. Default image format is PNG.		
Type	POST		
URL Format	`...ofice/excelchart`		
Java API	**Method Name**		**Class Name**
	getExcelChartBasedOnStream		`com.quark.qpp.service.facade.OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`inputStream`	string	The stream of the input file. (Mandatory)
	`outputformat=[image, json, html]`	string	Image output format. (Optional)
	`contentdisposition`	string	Content disposition value to be set on the http response headers.

	`outputStream`	string	Output stream onto which output is to be written (for local API invocatins).
Returns	Returns a stream of the required format based on the given parameters.		
Examples	**URL:** `http://server:port/rest/service/office/excelchart?outputformat=image&imageformat=png` **Results:** A filestream of required output format based on given parameters.		

getExcelChartBasedOnURI

Function	Get a Microsoft Excel chart based on URI		
Description	This request returns a Microsoft Excel Chart in the desired image format. Default image format is PNG.		
Type	GET		
URL Format	`...ofice/excelchart`		
Java API	**Method Name**		**Class Name**
	getExcelChartBasedOnURI		`com.quark.qpp.service.facade.OfficeByPathFacade`
Parameters	**Name**	**Type**	**Description**
	`uri`	string	The URI of the Microsoft Excel chart
	`outputformat=[image, json, html]`	string	Image output format. (Optional)
	`contentdisposition`	string	Content disposition value to be set on the http response headers.
	`outputStream`	string	Output stream onto which output is to be written (for local API invocatins).
Returns	Returns a stream of the required format based on the given parameters.		
Examples	**URL:** `http://server:port/rest/service/office/excelchart?uri=qpp://assetsbypath/Home/Chart1.xlsx&outputformat=image&imageformat=png` **Results:** A filestream of required output format based on given parameters.		

getExcelTable

Function	Get a Microsoft Excel table based on asset Id
Description	This request returns a Microsoft Excel table in the desired output format. Default output format is Smart Table.

Type	GET		
URL Format	`...ofice/exceltable/{assetId}`		
Java API	**Method Name**		**Class Name**
	`getExcelTable`		`com.quark.qpp.service.facade.OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`assetId`	string	The AssetID.
	`majorversion=[x]`	string	Major version of the asset. (Optional)
	`minorversion=[x]`	string	Minor version of the asset. (Optional)
	`outputformat`	string	Output format. (Optional)
	`contentdisposition`	string	Content disposition value to be set on the http response headers.
	`outputStream`	string	Output stream onto which output is to be written (for local API invocatins).
Returns	Returns a stream of the required output format based on the given parameters.		
Examples	**URL:** `http://server:port/rest/service/office/exceltable/48?outputformat=image&imageformat=png` **Results:** A filestream of required output format based on given parameters.		

Function	Get a Microsoft Excel table by path		
Description	This request returns a Microsoft Excel table in the desired output format. Default output format is Smart Table.		
Type	GET		
URL Format	`...ofice/exceltablebypath/{assetPath}`		
Java API	**Method Name**		**Class Name**
	`getExcelTable`		`com.quark.qpp.service.facade.OfficeByPathFacade`
Parameters	**Name**	**Type**	**Description**
	`assetPath`	string	The AssetPath.
	`majorversion=[x]`	string	Major version of the asset. (Optional)
	`minorversion=[x]`	string	Minor version of the asset. (Optional)
	`outputformat`	string	Output format. (Optional)
	`contentdisposition`	string	Content disposition value to be set on the http response headers.
	`outputStream`	string	Output stream onto which output is to be written (for local API invocatins).
Returns	Returns a stream of the required output format based on the given parameters.		
Examples	**URL:**		

	`http://server:port/rest/service/office/exceltablebypath/Home/BudgetDetails.xlsx?outputformat=image&imageformat=png` **Results:** A filestream of required output format based on given parameters.

getExcelTableBasedOnStream

Function	Get a Microsoft Excel chtable art based on stream		
Description	This request returns a Microsoft Excel table in the desired output format. Default image format is PNG.		
Type	GET		
URL Format	`...ofice/exceltable`		
Java API	**Method Name**		**Class Name**
	getExcelTableBasedOnStream		`com.quark.qpp.service.facade.OfficeByPathFacade`
Parameters	**Name**	**Type**	**Description**
	`inputStream`	string	The stream of the input file. (Mandatory)
	`outputformat`	string	Output format. (Optional)
	`contentdisposition`	string	Content disposition value to be set on the http response headers.
	`outputStream`	string	Output stream onto which output is to be written (for local API invocatins).
Returns	Returns a stream of the required output format based on the given parameters.		
Examples	**URL** `http://server:port/rest/service/office/exceltable?outputformat=image&imageformat=png` **Results:** A filestream of required output format based on given parameters.		

getExcelTableBasedOnURI

Function	Get a Microsoft Excel table based on URI	
Description	This request returns a Microsoft Excel table in the desired image format. Default output format is Smart Table.	
Type	GET	
URL Format	`...ofice/exceltable`	
Java API	**Method Name**	**Class Name**
	`getExcelTableBasedOnURI`	`com.quark.qpp.service.facade.Offi`

			ceByPathFacade
Parameters	Name	Type	Description
	uri	string	The URI of the Microsoft Exceltable
	outputformat	string	Output format. (Optional)
	contentdisposition	string	Content disposition value to be set on the http response headers.
	outputStream	string	Output stream onto which output is to be written (for local API invocatins).
Returns	Returns a stream of the required output format based on the given parameters.		
Examples	**URL:** http://server:port/rest/service/office/exceltable?uri=qpp://assetsbypath/Home/BudgetDetails.xlsx&outputformat=image&imageformat=png **Results:** A filestream of required output format based on given parameters.		

getExcelWorksheets

Function	Get all worksheets in Excel document.		
Description	This request returns all worksheets defined in an Excel document.		
Type	GET		
URL Format	..office/excel/[assetId] OR .../ office/ excelbypath /{ coll name1}/{ coll name 2}/../{asset name}		
Java API	Method Name		Class Name
	getExcelWorksheets		com.quark.qpp.service.facadeClassOfficeFacade
Parameters	Name	Type	Description
	majorversion=?	string	Major version to be fetched. (Optional)
	minorversion=?	string	Minor version to be fetched. (Optional)
	worksheets=?	string	Comma separated worksheet names of specific worksheets to be fetched. (Optional)
Returns	Returns all worksheets or required worksheets defined in an Excel document. Each worksheet encapsulates its metadata (such as worksheet index and name.) Each worksheet also provides information about tables, named ranges and charts defined in the worksheet.		
Examples	**URL:** http://serverName:serverPort/rest/service/office/excel/10 **Result:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <worksheetInfoList>		

```
    <worksheetInfo index="0">
      <name>Products</name>
      <chartInfoList>
        <chartInfo>
          <name>Chart 1</name>
        </chartInfo>
      </chartInfoList>
      <namedRangeInfoList>
        <namedRangeInfo>
          <name>Description</name>
        </namedRangeInfo>
        <namedRangeInfo>
          <name>DispatchRange</name>
        </namedRangeInfo>
      </namedRangeInfoList>
    </worksheetInfo>
    <worksheetInfo index="1">
      <name>Sales</name>
      <chartInfoList>
        <chartInfo>
          <name>Chart 1</name>
        </chartInfo>
      </chartInfoList>
      <tableInfoList>
        <tableInfo>
          <name>SalesTable</name>
        </tableInfo>
      </tableInfoList>
    </worksheetInfo>
  </worksheetInfoList>
```

URL:

http://servername:serverport/rest/service/office/excel/10?worksheets=sales,sheet3

Result:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <worksheetInfoList>
    <worksheetInfo index="1">
      <name>Sales</name>
      <chartInfoList>
        <chartInfo>
          <name>Chart 1</name>
        </chartInfo>
      </chartInfoList>
      <tableInfoList>
        <tableInfo>
          <name>SalesTable</name>
        </tableInfo>
      </tableInfoList>
    </worksheetInfo>
    <worksheetInfo index="2">
      <name>Sheet3</name>
    </worksheetInfo>
  </worksheetInfoList>
```

getExcelWorksheetsBasedOnURI

Function	Get all worksheets based on URI of the Excel document.
Description	This request returns all worksheets defined in an Excel document.

Type	GET		
URL Format	`...office/excel`		
Java API	**Method Name**		**Class Name**
	`getExcelWorksheetsBasedOnURI`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`URI`	string	Excel document uri.
	`worksheets`	string	Comma separated worksheet names if specific worksheets are to be fetched. (Optional)
Returns	Returns all worksheets or required worksheets defined in an Excel document. Each worksheet encapsulates its metadata (such as worksheet index and name. Each worksheet also provides information about tables, named ranges and charts defined in the worksheet.		

Examples	**URL:**
	`http://serverName:serverPort/rest/service/office/excel?uri=file:C:\DATA\Documents\MyExcelSamples\DocSample.xlsx`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<worksheetInfoList>`
	`  <worksheetInfo index="0">`
	`  <name>Products</name>`
	`  <chartInfoList>`
	`  <chartInfo>`
	`  <name>Chart 1</name>`
	`  </chartInfo>`
	`  </chartInfoList>`
	`  <namedRangeInfoList>`
	`  <namedRangeInfo>`
	`  <name>Description</name>`
	`  </namedRangeInfo>`
	`  <namedRangeInfo>`
	`  <name>DispatchRange</name>`
	`  </namedRangeInfo>`
	`  </namedRangeInfoList>`
	`  </worksheetInfo>`
	`  <worksheetInfo index="1">`
	`  <name>Sales</name>`
	`  <tableInfoList>`
	`  <tableInfo>`
	`  <name>SalesTable</name>`
	`  </tableInfo>`
	`  </tableInfoList>`
	`  </worksheetInfo>`
	`</worksheetInfoList>`		

getExcelWorksheetsBasedOnStreamContext

Function	Get all worksheets defined in an Excel document based on stream context.
Description	This request returns all worksheets defined in an Excel document.
Type	GET

URL Format	`...office/excel`		
Java API	**Method Name**		**Class Name**
	`getExcelWorksheetsBasedOnStreamContext`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`stream`	string	Excel document in request.
	`worksheets`	string	Comma separated worksheet names if specific worksheets are to be fetched. (Optional)
Returns	Returns all worksheets or required worksheets defined in an Excel document. Each worksheet encapsulates its metadata (such as worksheet index and name). Each worksheet also provides information about tables, named ranges and charts defined a worksheet.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/excel Stream the excel document as input.`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<worksheetInfoList>`
	`  <worksheetInfo index="0">`
	`  <name>Products</name>`
	`  <chartInfoList>`
	`  <chartInfo>`
	`  <name>Chart 1</name>`
	`  </chartInfo>`
	`  </chartInfoList>`
	`  <namedRangeInfoList>`
	`  <namedRangeInfo>`
	`  <name>Description</name>`
	`  </namedRangeInfo>`
	`  <namedRangeInfo>`
	`  <name>DispatchRange</name>`
	`  </namedRangeInfo>`
	`  </namedRangeInfoList>`
	`  </worksheetInfo>`
	`  <worksheetInfo index="1">`
	`  <name>Sales</name>`
	`  <tableInfoList>`
	`  <tableInfo>`
	`  <name>SalesTable</name>`
	`  </tableInfo>`
	`  </tableInfoList>`
	`  </worksheetInfo>`
	`</worksheetInfoList>`		

getChart

Function	Get an Excel chart in image format.
Description	This request returns an Excel chart in image format.
Type	GET

URL Format	`..office/excel/[assetId] OR .../ office/ excelbypath /{ coll name1}/{ coll name 2}/../{asset name}`		
Java API	**Method Name**		**Class Name**
	`getChart`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`majorversion=?`	string	Major version of the asset. (Optional)
	`minorversion=?`	string	Minor version of the asset. (Optional)
	`chart`	string	Name of the Excel chart to be retrieved.
	`worksheet`	string	The name of the worksheet where the chart is defined. By default the first chart in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	`horizontalresolution`	string	Image output horizontal resolution. (Optional)
	`verticalresolution`	string	Image output vetical resolution. (Optional)
	`imageormat`	string	Image output format. By default the image is returned in png format. (Optional)
	`contentDisposition`	string	Used to set the content-disposition header value in the response. Set to “attachment” if downloading of chart is required. (Optional)
Returns	Returns an Excel chart in the image format.		
Examples	URL: `http://serverName:serverPortt/rest/service/office/excel/36?chart=Chart1&horizontalresolution=100&verticalresolution=100&imageformat=png` Result:		

getChartBasedOnURI

Function	Get Excel chart based on the URI of the Excel document.		
Description	This request returns an Excel chart in the image format.		
Type	GET		
URL Format	`...office/excel`		
Java API	**Method Name**		**Class Name**
	`getChartBasedOnURI`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**

	uri	string	URI of the Excel document.
	chart	string	Name of the Excel chart to be retrieved.
	worksheet	string	The name of the worksheet where the chart is defined. By default the first chart in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	horizontalresolution	string	Image output horizontal resolution. (Optional)
	verticalresolution	string	Image output vetical resolution. (Optional)
	imageormat	string	Image output format. By default the image is returned in png format. (Optional)
	contentDisposition	string	Uused to set the content-disposition header value in the response. Set to "attachment" if downloading of chart is required. (Optional)
Returns	Returns an Excel chart in the image format.		
Examples	URL: http://serverName:serverPort/rest/service/office/excel?uri=classpath:SampleExcelFile.xlsx&worksheet=Products&chart=chart1&imageformat=png Result:		

getChartBasedOnStream

Function	Get a chart based on the input stream of an Excel document.		
Description	This request returns an Excel chart in the image format. Include the file in the request body or as file part.		
Type	GET		
URL Format	...office/excel		
Java API	**Method Name**		**Class Name**
	getChartBasedOnStream		com.quark.qpp.service.facadeClassOfficeFacade
Parameters	**Name**	**Type**	**Description**
	chart	string	Name of the Excel chart to be retrieved.
	worksheet	string	The name of the worksheet where the chart is defined. By default the first chart in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	horizontalresolution	string	Image output horizontal resolution.

			(Optional)
	verticalresolution	string	Image output vetical resolution. (Optional)
	imageformat	string	Image output format. By default the image is returned in png format. (Optional)
	contentDisposition	string	Uused to set the content-disposition header value in the response. Set to "attachment" if downloading of chart is required. (Optional)
Returns	Returns an Excel chart in the image format.		
Examples	URL: http://serverName:serverPort/rest/service/office/excel?worksheet=Products& chart=chart1&imageformat=png Result:		

getTable

Function	Get an Excel table.		
Description	This request r eturns an Excel table in the desired output format.		
Type	GET		
URL Format	..office/excel/[assetId] OR .../ office/ excelbypath /{ coll name1}/{ coll name 2}/../{asset name}		
Java API	**Method Name**		**Class Name**
	getTable		com.quark.qpp.service.facadeClass OfficeFacade
Parameters	**Name**	**Type**	**Description**
	majorversion=?	string	Major version to be retrieved. (Optional)
	minorversion=?	string	Minor version to be retrieved. (Optional)
	table	string	Name of the table to be retrieved. (Optional)
	worksheets=?	string	The name of the worksheet where the table is defined. By default the first table in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	outputformat	string	Value of the output format in which the table is to be rendered. Possible values are defined in class com.quark.qpp.office.service.constants.ExcelDataOutputFormats . Default value is "smarttable" .
	contentDisposition	string	Used to set the content-disposition header

			value in the response. Set to “attachment” if downloading of table is required. (Optional)
Returns	Returns an Excel table in the desired output format.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/excel/10?worksheet= Products&table=SalesDetails`
	Result:
	`<?xml version="1.0" encoding="utf-8"?>`
	`<table xmlns="http://quark.com/ smartcontent /1.0" type=" smart-table ">`
	`  <tgroup cols="2">`
	`  <colspec colnum="1" colname="1" colwidth="64px"/>`
	`  <colspec colnum="2" colname="2" colwidth="131px"/>`
	`  <tbody>`
	`  <row>`
	`  <entry valign="bottom" colsep="0" rowsep="0">`
	`  <p>`
	`  <t>Name</t>`
	`  </p>`
	`  </entry>`
	`  <entry valign="bottom" colsep="0" rowsep="0">`
	`  <p>`
	`  <t>Sales (in millions)</t>`
	`  </p>`
	`  </entry>`
	`  </row>`
	`  </tbody>`
	`  </tgroup>`
	`</table>`		

getTableBasedOnURI

Function	Get an Excel table based on an Excel document URI.		
Description	This request returns an Excel table in the desired output format.		
Type	GET		
URL Format	`...office/excel`		
Java API	**Method Name**		**Class Name**
	`getTableBasedOnURI`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`uri`	string	URI of the Excel document.
	`table`	string	Name of the table to be retrieved. (Optional)
	`worksheets=?`	string	The name of the worksheet where the chart is defined. By default the first chart in Excel with the given name is returned if the worksheet is not mentioned. (Optional)

	`outputformat`	string	Value of the output format in which the table is to be rendered. Possible values are defined in class com.quark.qpp.office.service.constants.ExcelDataOutputFormats . Default value is “smarttable” .
	`contentDisposition`	string	Uused to set the content-disposition header value in the response. Set to “attachment” if downloading of chart is required. (Optional)
Returns	Returns an Excel table in the desired output format.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/excel? uri = classpath:SampleExcelFile.xlsx&table = SalesDetails`
	Result:
	`<?xml version="1.0" encoding="utf-8"?>`
	`<table xmlns="http://quark.com/ smartcontent /1.0" type="smart-table">`
	`  <tgroup cols="2">`
	`  <colspec colnum="1" colname="1" colwidth="64px"/>`
	`  <colspec colnum="2" colname="2" colwidth="131px"/>`
	`  <tbody>`
	`  <row>`
	`  <entry valign="bottom" colsep="0" rowsep="0">`
	`  <p>`
	`  <t>Name</t>`
	`  </p>`
	`  </entry>`
	`  <entry valign="bottom" colsep="0" rowsep="0">`
	`  <p>`
	`  <t>Sales (in millions)</t>`
	`  </p>`
	`  </entry>`
	`  </row>`
	`  </tbody>`
	`  </tgroup>`
	`</table>`		

getTableBasedOnStreamContext

Function	Get an Excel table based on stream context.		
Description	This request r eturns an Excel table in the desired output format. Include the file in the request body or as file part.		
Type	GET		
URL Format	`...office/excel`		
Java API	**Method Name**		**Class Name**
	`getTableBasedOnStreamContext`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**

	table	string	Name of the table to be retrieved. (Optional)
	worksheets=?	string	The name of the worksheet where the chart is defined. By default the first chart in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	outputformat	string	Value of the output format in which the table is to be rendered. Possible values are defined in class com.quark.qpp.office.service.constants.ExcelDataOutputFormats . Default value is “smarttable”.
	contentDisposition	string	Uused to set the content-disposition header value in the response. Set to “attachment” if downloading of chart is required. (Optional)
Returns	Returns an Excel table in the desired output format.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/excel ? worksheet= Products&table = SalesInfo & stream the Excel file in input`
	Result:
	`<?xml version="1.0" encoding="utf-8"?>`
	`<table xmlns="http://quark.com/ smartcontent /1.0" type=" smart-table ">`
	`  <tgroup cols="2">`
	`  <colspec colnum="1" colname="1" colwidth="64px"/>`
	`  <colspec colnum="2" colname="2" colwidth="131px"/>`
	`  <tbody>`
	`  <row>`
	`  <entry valign="bottom" colsep="0" rowsep="0">`
	`  <p>`
	`  <t>Name</t>`
	`  </p>`
	`  </entry>`
	`  <entry valign="bottom" colsep="0" rowsep="0">`
	`  <p>`
	`  <t>Sales (in millions)</t>`
	`  </p>`
	`  </entry>`
	`  </row>`
	`  </tbody>`
	`  </tgroup>`
	`</table>`		

getNamedRange

Function	Get an Excel Named Range.

Description	This request r eturns an Excel named range in the desired output format.		
Type	GET		
URL Format	`..office/excel/[assetId]` OR `.../ office/ excelbypath /{ coll name1}/{ coll name 2}/../{asset name}`		
Java API	**Method Name**		**Class Name**
	`getNamedRange`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`majorversion=?`	string	Major version to be retrieved. (Optional)
	`minorversion=?`	string	Minor version to be retrieved. (Optional)
	`namedrange`	string	Name of the Excel named range to be retrieved.
	`worksheets=?`	string	The name of the worksheet where the named ranged is defined. By default the first named ranged in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	`outputformat`	string	Value of the output format in which the named range is to be rendered. Possible values are defined in class com.quark.qpp.office.service.constants.ExcelDataOutputFormats .
	`contentDisposition`	string	Used to set the content-disposition header value in the response. Set to “attachment” if downloading of named range is required. (Optional)
Returns	Returns an Excel named range in the desired output format.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/excel/10 ? worksheet= Products& namedrange = DispatchRange`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<table xmlns="http://quark.com/ smartcontent /1.0" type=" smart-table ">`
	`  <tgroup cols="1">`
	`  <colspec colnum="1" colname="1" colwidth="83x"/>`
	`  <tbody>`
	`  <row>`
	`  <entry align="right" valign="bottom" colsep="0" rowsep="0">`
	`  <p>`
	`  <t>10</t>`
	`  </p>`
	`  </entry>`
	`  </row> ..... </tbody>`
	`  </tgroup>`
	`</table>`		

getNamedRangeBasedOnURI

Function	Get an Excel Named Range based on the URI of the Excel document.		
Description	This request returns an Excel named range in the desired output format.		
Type	GET		
URL Format	`...office/excel`		
Java API	**Method Name**		**Class Name**
	`getNamedRangeBasedOnURI`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`uri`	string	Excel document URI.
	`namedrange`	string	Name of the Excel named range to be retrieved.
	`worksheets=?`	string	The name of the worksheet where the named ranged is defined. By default the first named ranged in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	`outputformat`	string	Value of the output format in which the named range is to be rendered. Possible values are defined in class com.quark.qpp.office.service.constants.ExcelDataOutputFormats .
	`contentDisposition`	string	Used to set the content-disposition header value in the response. Set to “attachment” if downloading of named range is required. (Optional)
Returns	Returns an Excel named range in the desired output format.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/excel? uri = classpath:SampleExcelFile.xlsx& namedrange = DispatchRange`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<table xmlns="http://quark.com/ smartcontent /1.0" type=" smart-table ">`
	`  <tgroup cols="1">`
	`  <colspec colnum="1" colname="1" colwidth="83x"/>`
	`  <tbody>`
	`  <row>`
	`  <entry align="right" valign="bottom" colsep="0" rowsep="0">`
	`  <p>`
	`  <t>10</t>`
	`  </p>`
	`  </entry>`
	`  </row> ..... </tbody>`
	`  </tgroup>`
	`</table>`		

getNamedRangeBasedOnStream

Function	Get an Excel Named Range based on stream context.		
Description	This request returns an Excel named range in the desired output format. Include the file in the request body or as file part.		
Type	GET		
URL Format	`...office/excel`		
Java API	**Method Name**		**Class Name**
	`getNamedRangeBasedOnStream`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`namedrange`	string	Name of the Excel named range to be retrieved.
	`worksheets=?`	string	The name of the worksheet where the named ranged is defined. By default the first named ranged in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	`outputformat`	string	Value of the output format in which the named range is to be rendered. Possible values are defined in class com.quark.qpp.office.service.constants.ExcelDataOutputFormats .
	`contentDisposition`	string	Used to set the content-disposition header value in the response. Set to “attachment” if downloading of named range is required. (Optional)
Returns	Returns an Excel named range in the desired output format.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/excel?worksheet= Sales&namedrange = DispatchRange & stream the Excel file in input`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<table xmlns="http://quark.com/ smartcontent /1.0" type=" smart-table ">`
	`  <tgroup cols="1">`
	`  <colspec colnum="1" colname="1" colwidth="83x"/>`
	`  <tbody>`
	`  <row>`
	`  <entry align="right" valign="bottom" colsep="0" rowsep="0">`
	`  <p>`
	`  <t>10</t>`
	`  </p>`
	`  </entry>`
	`  </row> ..... </tbody>`
	`  </tgroup>`
	`</table>`		

getDynamicRange

Function	Get an Excel Dynamic Range.		
Description	This request r eturns an Excel dynamic range in the desired output format.		
Type	GET		
URL Format	`..office/excel/[assetId]` OR `.../ office/ excelbypath /{ coll name1}/{ coll name 2}/../{asset name}d]`		
Java API	**Method Name**		**Class Name**
	`getDynamicRange`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`majorversion=?`	string	Major version to be retrieved. (Optional)
	`minorversion=?`	string	Minor version to be retrieved. (Optional)
	`dynamicrange`	string	Name of the Excel dynamic range to be retrieved.
	`worksheets=?`	string	The name of the worksheet where the dynamic ranged is defined. By default the first dynamic ranged in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	`outputformat`	string	Value of the output format in which the dynamic range is to be rendered. Possible values are defined in class com.quark.qpp.office.service.constants.ExcelDataOutputFormats .
	`contentDisposition`	string	Used to set the content-disposition header value in the response. Set to “attachment” if downloading of dynamic range is required. (Optional)
Returns	Returns an Excel dynamic range in the desired output format.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/excel/1?worksheet=Sales&dynamicrange=A2:C25`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<table xmlns="http://quark.com/ smartcontent /1.0" type=" smart-table ">`
	`  <tgroup cols="1">`
	`  <colspec colnum="1" colname="1" colwidth="104px"/>`
	`  <tbody>`
	`  <row>`
	`  <entry colsep="0" rowsep="0">`
	`  <p>`
	`  <t/>`
	`  </p>`
	`  </entry>`
	`  </row>`
	`  <row>`		

	`<entry valign="bottom" colsep="0" rowsep="0">`
	`<p>`
	`<t>Product details</t>`
	`</p>`
	`</entry>`
	`</row> ..... ..... </tbody>`
	`</tgroup>`
	`</table>`

getDynamicRangeBasedOnURI

Function	Get an Excel Dynamic Range based on the URI of the Excel document.		
Description	This request returns an Excel dynamic range in the desired output format.		
Type	GET		
URL Format	`...office/excel`		
Java API	**Method Name**		**Class Name**
	`getDynamicRangeBasedOnURI`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`uri`	string	Excel document URI.
	`dynamicrange`	string	Name of the Excel dynamic range to be retrieved.
	`worksheets=?`	string	The name of the worksheet where the dynamic ranged is defined. By default the first dynamic ranged in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	`outputformat`	string	Value of the output format in which the dynamic range is to be rendered. Possible values are defined in class com.quark.qpp.office.service.constants.ExcelDataOutputFormats .
	`contentDisposition`	string	Used to set the content-disposition header value in the response. Set to “attachment” if downloading of named range is required. (Optional)
Returns	Returns an Excel dynamic range in the desired output format.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/excel? uri = classpath:SampleExcelFile.xlsx&worksheet = Sales&dynamicrange =A2:C25`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<table xmlns="http://quark.com/ smartcontent /1.0" type=" smart-table ">`
	`<tgroup cols="1">`		

```
        <colspec colnum="1" colname="1" colwidth="104px"/>
        <tbody>
            <row>
                <entry colsep="0" rowsep="0">
                    <p>
                        <t/>
                    </p>
                </entry>
            </row>
            <row>
                <entry valign="bottom" colsep="0" rowsep="0">
                    <p>
                        <t>Product details</t>
                    </p>
                </entry>
            </row> ...... ...... </tbody>
    </tgroup>
</table>
```

getDynamicRangeBasedOnStream

Function	Get an Excel Dynamic Range based on stream context.		
Description	This request returns an Excel dynamic range in the desired output format. Include the file in the request body or as file part.		
Type	GET		
URL Format	`...office/excel`		
Java API	**Method Name**		**Class Name**
	`getDynamicRangeBasedOnStream`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`dynamicrange`	string	Name of the Excel dynamic range to be retrieved.
	`worksheets=?`	string	The name of the worksheet where the dynamic ranged is defined. By default the first named ranged in Excel with the given name is returned if the worksheet is not mentioned. (Optional)
	`outputformat`	string	Value of the output format in which the dynamic range is to be rendered. Possible values are defined in class com.quark.qpp.office.service.constants.ExcelDataOutputFormats .
	`contentDisposition`	string	Used to set the content-disposition header value in the response. Set to “attachment” if downloading of named range is required. (Optional)
Returns	Returns an Excel dynamic range in the desired output format.		
Examples	URL: `http://serverName:serverPort/rest/service/office/excel?`		

	worksheet= Sales&dynamicrange=r1c1:r5c5 & stream the Excel file in input
	
	Result:
	
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<table xmlns="http://quark.com/ smartcontent /1.0" type=" smart-table ">`
	`  <tgroup cols="1">`
	`  <colspec colnum="1" colname="1" colwidth="104px"/>`
	`  <tbody>`
	`  <row>`
	`  <entry colsep="0" rowsep="0">`
	`  <p>`
	`  <t/>`
	`  </p>`
	`  </entry>`
	`  </row>`
	`  <row>`
	`  <entry valign="bottom" colsep="0" rowsep="0">`
	`  <p>`
	`  <t>Product details</t>`
	`  </p>`
	`  </entry>`
	`  </row> ..... ..... </tbody>`
	`  </tgroup>`
	`</table>`

getVisioPagesInfo

Function	Get metadata of all pages of a Visio asset.		
Description	This request returns all pages defined within a Visio document. To retrieve specific pages, specify a valid page name in the pages parameter.		
Type	GET		
URL Format	`.../office/visio/[assetId]`		
Java API	**Method Name**		**Class Name**
	`getVisioPagesInfo`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`majorversion` and `minorversion`	string	Specify the required major and minor version for which the pages of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
	`pageNames`= [comma-separated list of page names]	string	The set of page names.
Returns	Returns a `VisioPageInfoList` object that contains index number and page name of all pages of a given Visio asset.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/visio/{assetId}`
	Result:
	A VisioPageInfoList object that contains index number and page name of all pages of a given Visio asset.
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<visioPageInfoList>`
	`  <visioPageInfo index="1">`
	`  <pageName>Page-1</pageName>`
	`  </visioPageInfo>`
	`</visioPageInfoList>`

getVisioPagesInfoBasedOnURI

Function	Get metadata of all pages of a Visio asset based on URI.		
Description	This request returns all pages defined within a Visio document. To retrieve specific pages, specify a valid page name in the pages parameter.		
Type	GET		
URL Format	`.../office/visio`		
Java API	**Method Name**		**Class Name**
	`getVisioPagesInfoBasedOnURI`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	Uri	string	The URI of the asset. (Mandatory)
	`pageNames`= [comma-separated list of page names]	string	The set of page names.
Returns	Returns a `VisioPageInfoList` object that contains index number and page name of all pages of a given Visio asset.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/visio?uri=qpp://assets/{assetId}`
	Result:
	A VisioPageInfoList object that contains index number and page name of all pages of a given Visio asset.
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<visioPageInfoList>`
	`  <visioPageInfo index="1">`
	`  <pageName>Page-1</pageName>`
	`  </visioPageInfo>`
	`</visioPageInfoList>`		

getVisioPagesInfoBasedOnStreamContext

Function	Get metadata of all pages of a Visio asset based on stream.		
Description	This request returns all pages defined within a Visio document. To retrieve specific pages, specify a valid page name in the pages parameter. Include the file in the request body or as a file part. (Mandatory)		
Type	GET		
URL Format	`.../office/visio`		
Java API	**Method Name**		**Class Name**
	`getVisioPagesInfoBasedOnStreamContext`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`pageNames`= [comma-separated list of page names]	string	The set of page names.
Returns	Returns a `VisioPageInfoList` object that contains index number and page name of all pages of a given Visio asset.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/visioInclude the file in the request body.`
	Result:
	A VisioPageInfoList object that contains index number and page name of all pages of a given Visio asset.
	`<visioPageInfoList>`
	`  <visioPageInfo index="1">`
	`  <pageName>Page-1</pageName>`
	`  </visioPageInfo>`
	`</visioPageInfoList>`		

getVisioPageOutput

Function	Get page output of a Visio asset.		
Description	This request returns a Visio document page output in the desired output format.		
Type	GET		
URL Format	`.../office/visio/{assetId}`		
Java API	**Method Name**		**Class Name**
	`getVisioPageOutput`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`majorversion` and `minorversion`	string	Specify the required major and minor version for which the pages of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset

			will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
	`pagename`= [comma-separated list of page names]	string	The set of page names.
Returns	Returns a filestream of required output format based on given parameters.		
Examples	URL: `http://serverName:serverPort/rest/service/office/visio/{assetId}?outputformat=image&pagename={pageName}&imageformat=PNG` Result: A filestream of the generated output of the given asset.		

getVisioPageOutputBasedOnURI

Function	Get metadata of all pages of a Visio asset based on URI.		
Description	This request returns a Visio document page output in the desired output format.		
Type	GET		
URL Format	`.../office/visio`		
Java API	**Method Name**		**Class Name**
	`getVisioPagesOutputBasedOnURI`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`Uri`	string	The URI of the asset. (Mandatory)
	`pagename`= [comma-separated list of page names]	string	The set of page names.
Returns	Returns a `VisioPageInfoList` object that contains index number and page name of all pages of a given Visio asset.		
Examples	URL: `http://serverName:serverPort/rest/service/office/visio?uri=qpp://assets/{assetId}&outputformat=image&pagename={pageName}&imageformat=PNG` Result: A filestream of the generated output of the given asset.		

getVisioPageOutputBasedOnStreamContext

Function	Get page output of a Visio asset based on stream.
Description	This request returns a Visio document page output in the desired output format.

	Include the file in the request body or as a file part. (Mandatory)		
Type	GET		
URL Format	`.../office/visio`		
Java API	**Method Name**		**Class Name**
	`getVisioPageOutputBasedOnStreamContext`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	pagename	string	The page name.
Returns	Returns a filestream of the required output format based on the given parameters.		
Examples	URL: `http://serverName:serverPort/rest/service/office/visio/outputformat=image&pagename={pageName}&imageformat=PNG Include the file in the request body.` Result: A filestream of the generated output of the given asset.		

getPowerPointSlidesInfo

Function	Get metadata of all slides of a PowerPoint asset.		
Description	This request returns all slides defined within a PowerPoint document. To retrieve specific slides, specify a valid slide index in the slideIndexes parameter.		
Type	GET		
URL Format	`.../office/powerpoint/[assetId]`		
Java API	**Method Name**		**Class Name**
	`getPowerPointSlidesInfo`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`majorversion` and `minorversion`	string	Specify the required major and minor version for which the pages of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
	`slideindexes`= [comma-separated list of slide indexes]	string	The set of slide indexes.
Returns	Returns a `PowerPointSlidesInfoList` object that contains index numbers and slide IDs of all slides for a given PowerPoint asset.		
Examples	`URL:` `http://serverName:serverPort/rest/service/office/powerpoint{asset`		

	Id}
	
	Result:
	
	A PowerPointSlidesInfoList object that contains index numbers and slide IDs for all slides of a given PowerPoint asset.
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<powerPointSlideInfoList>`
	`  <powerPointSlideInfo index="1">`
	`  <slideId>256</slideId>`
	`  </powerPointSlideInfo>`
	`  <powerPointSlideInfo index="2">`
	`  <slideId>257</slideId>`
	`  </powerPointSlideInfo>`
	`  <powerPointSlideInfo index="3">`
	`  <slideId>258</slideId>`
	`  </powerPointSlideInfo>`
	`  <powerPointSlideInfo index="4">`
	`  <slideId>259</slideId>`
	`  </powerPointSlideInfo>`
	`  <powerPointSlideInfo index="5">`
	`  <slideId>260</slideId>`
	`  </powerPointSlideInfo>`
	`</powerPointSlideInfoList>`

getPowerPointSlidesInfoBasedOnURI

Function	Get metadata of all slides of a PowerPoint asset based on URI.		
Description	This request returns all slides defined within a PowerPoint document. To retrieve specific slides, specify a valid slide id in the slideIndexes parameter.		
Type	GET		
URL Format	`.../office/powerpoint`		
Java API	**Method Name**		**Class Name**
	`getPowerPointSlidesInfoBasedOnURI`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`Uri`	string	The URI of the asset. (Mandatory)
	`slideindexes`= [comma-separated list of slide index numbers	string	The set of index number.
Returns	Returns a `PowerPointSlidesInfoList` object that contains index numbers and slide IDs for all slides of a given PowerPoint asset.		
Examples	URL: `http://serverName:serverPort/rest/service/office/powerpoint?uri=qpp://assets/{assetId}` Result: A PowerPointSlidesInfoList object that contains index numbers and slide IDs for all slides of a given PowerPoint asset.		

getPowerPointSlidesInfoBasedOnStreamContext

Function	Get all slide metadata for a PowerPoint asset based on file stream.		
Description	This request returns all slides defined within a PowerPoint document. To retrieve specific slides, specify a valid index of a slide in the slideIndexes parameter. Include the file in the request body or as a file part. (Mandatory)		
Type	GET		
URL Format	`.../office/powerpoint`		
Java API	**Method Name**		**Class Name**
	`getPowerPointSlidesInfoBasedOnStreamContext`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`slideindexes`= [comma-separated list of slide indexes	string	The set of slide ids.
Returns	Returns a `PowerPointSlideInfoList` object that contains index numbers and slide IDs for all slides of a given PowerPoint asset.		

Examples	URL:
	`http://serverName:serverPort/rest/service/office/powerpointInclud e the file in the request body.`
	Result:
	`A PowerPointSlideInfoList object that contains index numbers and slide ids of all slides of a given PowerPoint asset.`
	`<powerPointSlideInfoList>`
	`  <powerPointSlideInfo index="1">`
	`  <slideId>256</slideId>`
	`  </powerPointSlideInfo>`
	`  <powerPointSlideInfo index="2">`
	`  <slideId>257</slideId>`
	`  </powerPointSlideInfo>`
	`  <powerPointSlideInfo index="3">`
	`  <slideId>258</slideId>`
	`  </powerPointSlideInfo>`
	`  <powerPointSlideInfo index="4">`
	`  <slideId>259</slideId>`
	`  </powerPointSlideInfo>`
	`  <powerPointSlideInfo index="5">`
	`  <slideId>260</slideId>`
	`  </powerPointSlideInfo>`
	`</powerPointSlideInfoList>`		

getPowerPointSlideOutput

Function	Get slide output of a PowerPoint asset.	
Description	This request extracts PowerPoint slide output in the specified format.	
Type	GET	
URL Format	`.../office/powerpoint/[assetId]`	
Java API	**Method Name**	**Class Name**

	`getPowerPointSlidesOutput`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`majorversion` and `minorversion`	string	Specify the required major and minor version for which the slide of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
	`slideid`	string	The slide Id.
	`outputformat`	string	The desired output format. Possible values are defined in class com.quark.qpp.office.service.constants.ExcelDataOutputFormats
Returns	Returns a filestream that contains the required output format based on the given parameters.		
Examples	URL: `http://server:port/rest/service/office/powerpoint/{assetId}?outputformat=image&slideid={slideId}&imageformat=PNG` Result: Returns a filestream that contains the required output format based on the given parameters.		

getPowerPointSlideOutputBasedOnURI

Function	Get slide output of a PowerPoint asset based on URI.		
Description	This request extracts the PowerPoint slide output in the desired output format..		
Type	GET		
URL Format	`.../office/powerpoint`		
Java API	**Method Name**		**Class Name**
	`getPowerPointSlideOutputBasedOnURI`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`Uri`	string	The URI of the asset. (Mandatory)
	`slideid`	string	The slide Id.
	`outputformat`	string	The desired output format.
Returns	Returns a filestream that contains the required output format based on the given parameters.		
Examples	URL: `http://server:port/rest/service/office/powerpoint`		

	`?uri=qpp://assets/{assetId}&outputformat=image&slideid={slideId}&imageformat=PNG` Result: Returns a filestream that contains the required output format based on the given parameters.

getPowerPointSlideOutputBasedOnStreamContext

Function	Get slide output of a PowerPoint asset based on file stream.		
Description	This request extracts the PowerPoint slide output in the desired output format.. Include the file in the request body or as a file part. (Mandatory)		
Type	GET		
URL Format	`.../office/`powerpoint		
Java API	**Method Name**		**Class Name**
	`getPowerPointSlideOutputBasedOnStreamContext`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`slideid`	string	The slide Id.
	`outputformat`	string	The desired output format.
Returns	Returns a filestream that contains the required output format based on the given parameters.		
Examples	URL: `http://server:port/rest/service/office/powerpoint?outputformat=image&slideid={slideId}&imageformat=PNGInclude the file in the request body.` Result: Returns a filestream that contains the required output format based on the given parameters.		

getPowerPointSlideRangeOutput

Function	Get slide range output of a PowerPoint asset.		
Description	This request extracts the PowerPoint slide output in the desired output format.		
Type	GET		
URL Format	`.../office/powerpoint/[assetId]`		
Java API	**Method Name**		**Class Name**
	`getPowerPointSlideRangeOutput`		`com.quark.qpp.service.facadeClass OfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`majorversion` and	string	Specify the required major and minor

	`minorversion`		version for which the slide of an asset is required. (Optional) These parameters will be used collectively. If neither parameter is passed, the latest version of the asset will be considered. If only one of the parameters is not passed, then the value for that parameter is assumed to be 0.
	`sliderange` = [comma-separated list of slide indexes]	string	The set of slide indexes.
	`outputformat`	string	The desired output format.
Returns	Returns a filestream that contains the required output format based on the given parameters.		
Examples	URL: `http://server:port/rest/service/office/powerpoint/{assetId}?outputformat=image&sliderange=1,3,5&imageformat=PNG` Result: Returns a filestream that contains the required output format based on the given parameters.		

getPowerPointSlideRangeOutputBasedOnURI

Function	Get slide range output of a PowerPoint asset based on URI.		
Description	This request extracts the PowerPoint slide output in the desired output format..		
Type	GET		
URL Format	`.../office/powerpoint`		
Java API	**Method Name**		**Class Name**
	`getPowerPointSliderRangeOutputBasedOnURI`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`Uri`	string	The URI of the asset. (Mandatory)
	`sliderange` = [comma-separated list of slide indexes]	string	The set of slide indexes.
	`outputformat`	string	The desired output format.
Returns	Returns a filestream that contains the required output format based on the given parameters.		
Examples	URL: `http://server:port/rest/service/office/powerpoint?uri=qpp://assets/{assetId}&outputformat=image&sliderange=1,3,5&imageformat=PNG` Result:		

	Returns a filestream that contains the required output format based on the given parameters.

getPowerPointSlideRangeOutputBasedOnStreamContext

Function	Get slide range output of a PowerPoint asset based on file stream.		
Description	This request extracts the PowerPoint slide output in the desired output format.. Include the file in the request body or as a file part. (Mandatory)		
Type	GET		
URL Format	`.../office/powerpoint`		
Java API	**Method Name**		**Class Name**
	`getPowerPointSlideRangeOutputBasedOnStreamContext`		`com.quark.qpp.service.facadeClassOfficeFacade`
Parameters	**Name**	**Type**	**Description**
	`sliderange` = [comma-separated list of slide indexes]	string	The set of slide indexes.
	`outputformat`	string	The desired output format.
Returns	Returns a filestream that contains the required output format based on the given parameters.		
Examples	URL: `http://server:port/rest/service/office/powerpoint?sliderange=1,3,5&imageformat=PNGInclude the file in the request body.` Result: Returns a filestream that contains the required output format based on the given parameters.		

Charting

The charting capabilities of the API includes the following REST calls:

- *getChart*
- *getChartBasedOnUri*
- *getChartBasedOnStream*

getChart

Function	Create new chart.		
Description	This request draws a chart for the given asset using the specified template.		
Type	GET		
URL Format	`...charts/{assetId}` or `.../chartsbypath/[asset path]`		
Java API	**Method Name**		**Class Name**
	`getChart`		`com.quark.qpp.service.facadeClass ChartingFacade`
Parameters	**Name**	**Type**	**Description**
	`assetId`	string	The AssetID.
	`majorversion`=[x]	string	Major version of the asset. (Optional)
	`minorversion`=[x]	string	Minor version of the asset. (Optional)
	`templateuri`=[x]	string	Uri for the template file.
	`outputformat`=[image, json, html]	string	Image output format. (Optional)
Returns	Returns a stream of the successfully created chart.		
Examples	URL: `http://serverName:serverPortt/rest/service/charts/408?templateuri = qpp://assets/409&outputformat=html` Result: The file stream of the successfully created chart.		

getChartBasedOnUri

Function	Create new chart By URI.

Description	This request draws a chart for the given data specified in the URI using the specified template.		
Type	GET		
URL Format	`...charts/`		
Java API	**Method Name**		**Class Name**
	`getChartBasedOnURI`		`com.quark.qpp.service.facadeClass ChartingFacade`
Parameters	**Name**	**Type**	**Description**
	`uri`=[x]	string	The URI for the source asset.
	`majorversion`=[x]	string	Major version of the asset. (Optional)
	`minorversion`=[x]	string	Minor version of the asset. (Optional)
	`templateuri`=[x]	string	Uri for the template file.
	`outputformat`=[image, json, html]	string	Image output format. (Optional)File, ClassPath, Http, Qpp are the supported URIs.
Returns	Returns a stream of the successfully created chart.		
Examples	URL: `http://serverName:serverPort/rest/service/charts?uri=qpp://assets/408&templateuri= qpp://assets/409&outputformat=htm` Result: The file stream of the successfully created chart.		

getChartBasedOnStream

Function	Create new chart By Stream		
Description	This request draws a chart for the given data input stream using the specified template. Include the file in the request body or as file part.		
Type	GET		
URL Format	`.../charts/`		
Java API	**Method Name**		**Class Name**
	`getChartBasedOnStream`		`com.quark.qpp.service.facadeClass ChartingFacade`
Parameters	**Name**	**Type**	**Description**
	`majorversion`=[x]	string	Major version of the asset. (Optional)
	`minorversion`=[x]	string	Minor version of the asset. (Optional)
	`templateuri`=[x]	string	Uri for the template file.
	`outputformat`=[image, json, html]	string	Image output format. (Optional)File, ClassPath, Http, Qpp are the supported URIs
Returns	Returns a stream of the successfully created chart.		

Examples	URL: `http://serverName:serverPort/rest/service/charts?templateuri=qpp://assets/409&outputformat=html` Result: The file stream of the successfully created chart.

Configuration

The Configuration service is used to manage the publishing configuration (adding/updating new publishing channels, processes and MIME types) and is responsible for the following:

- Managing the configuration files.
- Capturing updates.
- Reinitializing corresponding bean definitions.

In a multi-server environment, each node in a cluster must have access to a single .ext folder and the shared location must be appended to the classpth system environment variable.

POST requests only alter config files on a configured app server node. Multi-server deployments can leverage a shared file system location for the config files and avoid needing to manually maintain multiple sets of files on various application servers.

The `reInitialize` call can be used to avoid restarts even after changing the configuration files manually on the server .

The configuration capabilities of the API includes the following REST calls:

- *getAllXmlMimeTypes*
- *getXmlMimeType*
- *createXmlMimeType*
- *getMappedMimeTypes*
- *mapMimeTypes*
- *unmapMimeTypes*
- *deleteXmlMimeType*
- *getPublishingProcess*
- *createPublishingProcess*
- *deletePublishingProcess*
- *getAllPublishingChannelDefinitions*
- *getPublishingChannelDefinition*
- *getPublishingChannelDefinitions*
- *createPublishingChannels*

- *createContentChannelMappings*
- *deleteContentChannelMappings*
- *createIndexingChannelMapping*
- *getMappedIndexingChannel*
- *deleteIndexingChannelMapping*
- *deletePublishingChannels*
- *getDeploymentProperties*
- *getDeploymentVersion*
- *reInitialize*

getAllXmlMimeTypes

Function	Get XML MimeTypes	
Description	This request returns all XML MimeTypes that exist on the server.	
Type	GET	
URL Format	`.../config/xmlmimetypes`	
Java API	**Method Name**	**Class Name**
	`getAllXmlMimeTypes`	`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	This request requires no input parameters.	
Returns	Returns the definitions of all xml mime-types.	

Examples	URL:
	`http://serverName:serverPort/rest/service/config/xmlmimetypes`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<mimeTypeInfoList>`
	`  <mimeTypeInfo type="application/xml; format=smarttable" parentMimeType="application/xml;">`
	`  <xpath>/*[local-name()='table']/@type='smarttable'</xpath>`
	`  </mimeTypeInfo>`
	`  <mimeTypeInfo type="application/xml; format=auxiliarydictionary" parentMimeType="format=hyphenationexceptions">`
	`  <xpath>/*[local-name()='Dictionary']/@Type='AuxiliaryDictionary'</xpath>`
	`  </mimeTypeInfo>`
	`  <mimeTypeInfo type="application/dita+xml; format=busdoc" parentMimeType="application/dita+xml; format=topic">`
	`  <pi name="Xpress">productLine="busdoc"</pi>`
	`  </mimeTypeInfo>`
	`  <mimeTypeInfo type="application/xml; format=smartdoc" parentMimeType="application/xml; format=smartcontent">`
	`  <xpath>/*[local-name()='section']/@type='document'</xpath>`
	`  </mimeTypeInfo>`
	`  <mimeTypeInfo type="application/xml; format=smartsection" parentMimeType="application/xml; format= smartcontent">`	

	`<xpath>/*[local-name()='section']/@type='section'</xpath>`
	`</mimeTypeInfo>`
	`</mimeTypeInfoList>`

getXmlMimeType

Function	Get XML MimeType		
Description	This request returns a specified XML MimeType.		
Type	GET		
URL Format	`.../config/xmlmimetypes`		
Java API	**Method Name**		**Class Name**
	`getXmlMimeType`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`Mimetype`	string	The desired MimeType. (Mandatory)
Returns	Returns the definition of the desired XML MimeType, including the parent MimeType, xpath and the processing instruction name and value based on which given MimeType is detected.		

Examples	URL:
	`http://serverName:serverPort/rest/service/config/xmlmimetypes?mimetype=application/xml; format=researchreport`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<mimeTypeInfoList>`
	`<mimeTypeInfo type="application/xml; format=researchreport" parentMimeType="application/xml; format= smartcontent">`
	`<xpath>/*[local-name()='section']/@type='researchreport'</xpath>`
	`</mimeTypeInfo>`
	`</mimeTypeInfoList>`		

createXmlMimeType

Function	Create a custom XML MimeType.		
Description	This request creates a MimeType for an XML document.		
Type	POST		
URL Format	`.../config/xmlmimetypes`		
Java API	**Method Name**		**Class Name**
	`createXmlMimeType`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`op=create`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)

	mimetype	string	The MimeType to be created. (Mandatory)
	parentmimetype	string	The parent MimeType. (Mandatory)
	xpath	string	The xPath. (It is mandatory to specify either the xPath or the piName and piValue.)
	piname	string	The processing instruction. (It is mandatory to specify either the xPath or the piName and piValue.)
	pivalue	string	The processing value. (It is mandatory to specify either the xPath or the piName and piValue.)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: http://serverName:serverPort/rest/service/config/xmlmimetypes?op=create&mimetype=application/xml;format=researchreport&parentmimetype=application/xml;format=smartcontent&xpath=/*[local-name()='section']/@type='researchreport'		

getMappedMimeTypes

Function	Get mapped MimeTypes		
Description	This request returns a list of MimeTypes mapped to the given content type.		
Type	GET		
URL Format	.../config/xmlmimetypes		
Java API	**Method Name**		**Class Name**
	getMappedMimeTypes		com.quark.qpp.service.facade Class ConfigurationFacade
Parameters	**Name**	**Type**	**Description**
	contenttype	string	The content type. (Mandatory)
Returns	Returns a mimeTypeInfoList object that contains the metadata of all mimeTypes mapped to the given content type.		

Examples	URL:
	http://serverName:serverPort/rest/service/config/xmlmimetypes?contenttype=Research Report
	
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<mimeTypeInfoList>`
	`  <mimeTypeInfo type="application/xml; format=researchreport" parentMimeType="application/xml; format= smartcontent">`
	`  <xpath>/*[local-name()='section']/@type='researchreport'</xpath>`
	`  </mimeTypeInfo>`
	`</mimeTypeInfoList>`		

mapMimeTypes

Function	Map custom XML MimeType with ContentType.		
Description	This request maps a content type to a list of mime types.		
Type	POST		
URL Format	`.../config/xmlmimetypes`		
Java API	**Method Name**		**Class Name**
	`mapMimeTypes`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=mapcontenttype	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`contenttype`= [contentTypeIdOrName]	string	The content type to be mapped. (Mandatory)
	`mimetypes`	string	The mimetypes to be mapped. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: `http://serverName:serverPort/rest/service/config/xmlmimetypes?op=mapcontenttype&contenttype=Research Report&mimetypes=application/xml; format=researchreport`		

unmapMimeTypes

Function	Unmap custom XML MimeType with ContentType.		
Description	This request deletes the content type mappings for given mime types.		
Type	POST		
URL Format	`.../config/xmlmimetypes`		
Java API	**Method Name**		**Class Name**
	`unmapMimeTypes`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=unmapcontenttype	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`mimetypes`	string	The mimetypes to be unmapped. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: `http://serverName:serverPort/rest/service/config/xmlmimetypes?op=unmapcontenttype&mimetypes=application/xml;`		

	format=researchreport

deleteXmlMimeType

Function	Delete custom XML MimeType.		
Description	This request deletes an xml mime type and if this mime type is mapped to a content type, that mapping is also deleted.		
Type	POST		
URL Format	.../config/xmlmimetypes		
Java API	**Method Name**		**Class Name**
	deleteXmlMimeType		com.quark.qpp.service.facade Class ConfigurationFacade
Parameters	**Name**	**Type**	**Description**
	op=delete	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	mimeType	string	The mimetype to be deleted. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: http://serverName:serverPort/rest/service/config/xmlmimetypes?op=delete&mimetype=application/xml; format=researchreport		

getPublishingProcess

Function	Get publishing process		
Description	This request returns the xml configuration of the process with the given process name.		
Type	GET		
URL Format	.../config/publishingprocess/{ProcessId}		
Java API	**Method Name**		**Class Name**
	getPublishingProcess		com.quark.qpp.service.facade Class ConfigurationFacade
Parameters	**Name**	**Type**	**Description**
	processName	string	The process name. (Mandatory)
Returns	Returns the xml configuration of the process with the given process name.		

Examples	URL:
	http://serverName:serverPort/rest/service/config/publishingprocess/ researchReportToQxpRenderer
	Result:
	`<beans xmlns="http://www.springframework.org/schema/beans">`
	`  <!-- PROCESS DEFINITIONS -->`		

```
    <pub:process id="researchReportToQxpRenderer">
        <pub:activity
class="com.quark.qpp.publishing.activity.UnzipXaDocumentPackage">
            <description>Unzips and registers XA document file
along with the dependencies supplied in the zip file.</descrip-
tion>
            <property name="activityName">
                <value>UnzipXaDocumentPackage</value>
            </property> ...
    </pub:process>
</beans>
```

createPublishingProcess

Function	Create publishing process		
Description	This request creates a new process based on the publishingPocessXML.		
Type	POST		
URL Format	`.../config/publishingprocess`		
Java API	**Method Name**		**Class Name**
	`createPublishingProcess`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`processxml`= [xml]	string	The publishing process XML (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: `http://serverName:serverPort/rest/service/config/publishingproces s ?op=create&processxml=<xml>`		

deletePublishingProcess

Function	Delete publishing process.		
Description	This request deletes the publishing process with the given process name. Deletes the channels where this process is used. Deletes the corresponding configuration file of the process and updates ProcessConfig-ext.xml.		
Type	POST		
URL Format	`.../config/publishingprocess/{ProcessName}`		
Java API	**Method Name**		**Class Name**
	`deletePublishingProcess`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=delete	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`processName`	string	The name of the process to be deleted.

			(Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: `http://serverName:serverPort/rest/service/config/publishingprocess /researchReportToQxpRenderer?op=delete`		

getAllPublishingChannelDefinitions

Function	Get all publishing channels	
Description	This request returns all publishing of type publish, delver and datadoc.	
Type	GET	
URL Format	`.../config/publishingchannels`	
Java API	**Method Name**	**Class Name**
	`getAllPublishingChannelDefinitions`	`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `publishingChannelList` object that contains the metadata of all publishing channels (Publish, Deliver, Datadoc).	

Examples	URL:
	`http://serverName:serverPort/rest/service/config/publishingchannels`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<publishingChannelList>`
	`<publishingChannelInfo id="qppPreviewConvert" name="QPP.PREVIEW.IM.JPG" process="qppPreviewConverter" type="publish">`
	`<parameter name="SOURCE_FILE_EXTENSION" ask="true"/>`
	`<parameter name="OUTPUT_FILE_NAME" ask="true"/>`
	`</publishingChannelInfo>`
	`<publishingChannelInfo id="smartDocJpeg" name="QXP.JPEG.ZIP" process="smartDocToQxpRenderer" type="publish">`
	`<parameter name="ASSET_ID" ask="true"/>`
	`<parameter name="LAYOUT" ask="true"/> ..... </publishingChannelInfo>`
	`</publishingChannelList>`	

getPublishingChannelDefinition

Function	Get publishing channel
Description	This request returns the definition of the publishing channel with the given Id, including the channel Id, name, channel type, the channel's process name and channel parameters.
Type	GET
URL Format	`.../config/publishingchannels/{channelId}`

Java API	**Method Name**		**Class Name**
	`getPublishingChannelDefinition`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`channelId`	string	The channel Id of the desired publishing channel. (Mandatory)
Returns	Returns a `publishingChannelList` object that contains the metadata of the given publishing channel ID.		

Examples	URL:
	`http://serverName:serverPort/rest/service/config/publishingchannels/smartDocJpeg`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<publishingChannelList>`
	`<publishingChannelInfo id="smartDocJpeg" name="QXP.JPEG.ZIP" process="smartDocToQxpRenderer" type="publish">`
	`<parameter name="ASSET_ID" ask="true"/>`
	`<parameter name="LAYOUT" ask="true"/>`
	`<parameter name="BOX" ask="true"/>`
	`<parameter name="SCALE" ask="true"/> ..... <parameter name="MAX_PAGES" ask="false">0</parameter>`
	`<parameter name="RESOLVE_QPP_RELATIONS" ask="true">true</parameter>`
	`</publishingChannelInfo>`
	`</publishingChannelList>`		

getPublishingChannelDefinitions

Function	Get the publishing channel for a specific content type.		
Description	This request returns the definition of the publishing channels of type publish, deliver and datadoc that are mapped to a particular content type.		
Type	GET		
URL Format	`.../config/publishingchannels`		
Java API	**Method Name**		**Class Name**
	`getPublishingChannelDefinitions`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`contenttype`	string	The content type that is mapped to the desired publishing channel. (Mandatory)
Returns	Returns a `publishingChannelList` object that contains the metadata of all the publishing channels available for the given content type.		
Examples	URL: `http://serverName:serverPort/rest/service/config/publishingchannels ?contenttype=Smart Document` Result:		

	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<publishingChannelList>`
	`  <publishingChannelInfo id="smartDocJpeg" name="QXP.JPEG.ZIP" process="smartDocToQxpRenderer" type="publish">`
	`  <parameter name="ASSET_ID" ask="true"/>`
	`  <parameter name="LAYOUT" ask="true"/>`
	`  <parameter name="BOX" ask="true"/>`
	`  <parameter name="SCALE" ask="true"/> ..... <parameter name="MAX_PAGES" ask="false">0</parameter>`
	`  <parameter name="RESOLVE_QPP_RELATIONS" ask="true">true</parameter>`
	`  </publishingChannelInfo>`
	`</publishingChannelList>`

createPublishingChannels

Function	Create new publishing channels		
Description	This request creates new publishing channels as specified in the publishingChannelList.		
Type	POST		
URL Format	`.../config/publishingchannels`		
Java API	**Method Name**		**Class Name**
	`createPublishingChannels`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=create	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`publishingchannellist` = [publishingChannelList]	string	The publishing channel list. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		

Examples	URL:
	`http://serverName:serverPort/rest/service/config/publishingchannels ?op=create&publishingchannellist=`
	`<publishingChannelList>`publishingChannelList Example: `<publishingChannelInfo id="researchReportPdf" name="PDF" process="smartDocToQxpRenderer" type="publish">`
	`  <parameter name="OUTPUT_STYLE" ask="true">PDF High Quality</parameter>`
	`  <parameter name="ASSET_ID" ask="true"/>`
	`  <parameter name="QXP_TEMPLATE_URI" ask="true">classpath:ResearchReportTemplate.qxp</parameter>`
	`  <parameter name="RENDER_FORMAT">PDF</parameter>`
	`  <parameter name="EXCEL_DATA_IMAGE_FORMAT">PDF</parameter>`
	`  <parameter name="ANNOTATE_ERRORS">true</parameter>`
	`  <parameter name="TRACKED_CHANGES" ask="true">ACCEPT</parameter>`
	`  <parameter name="XSL_URI" ask="true">classpath:SmartDoc2QXPS.xslt</parameter>`
	`  <parameter name="APPEND_ERRORS" ask="true">true</parameter>`
	`  </publishingChannelInfo>`		

```
    <publishingChannelInfo id="researchReportJpeg"
name="QXP.JPEG.ZIP" type="publish" process="smartDocToQxpRen-
derer">
        <parameter name="ASSET_ID" ask="true"/>
        <parameter name="LAYOUT" ask="true"/>
        <parameter name="BOX" ask="true"/>
        <parameter name="SCALE" ask="true"/>
        <parameter name="RENDER_FORMAT">JPEG</parameter>
        <parameter name="XSL_URI"
ask="true">classpath:SmartDoc2QXPS.xslt</parameter>
        <parameter name="QXP_TEMPLATE_URI"
ask="true">classpath:SmartDocTemplate.qxp</parameter>
        <parameter name="EXCEL_DATA_IMAGE_FORMAT"
ask="true">PDF</parameter>
        <parameter name="CHART_IMAGE_FORMAT">PDF</parameter>
        <parameter name="VISIO_DATA_IMAGE_FORMAT">PDF</parameter>
        <parameter name="POWERPOINT_DATA_IMAGE_FORMAT">PDF</para-
meter>
        <parameter name="TRACKED_CHANGES" ask="true">ACCEPT</pa-
rameter>
        <parameter name="RESOLVE_QPP_RELATIONS"
ask="true">true</parameter>
    </publishingChannelInfo>
</publishingChannelList>
```

createContentChannelMappings

Function	Map publishing channel with content type for publishing.		
Description	This request maps content types to channels and updates the “PublishingConfig-ext.xml” file with the new mappings.		
Type	POST		
URL Format	`.../config/publishingchannels`		
Java API	**Method Name**		**Class Name**
	`createContentChannelMappings`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=mapcontenttype	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`contenttypes`	string	The content types to be mapped. (Mandatory)
	`channels`[comma-separated list of channels]	string	The publishing channels. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: `http://serverName:serverPort/rest/service/config/publishingchannels ?op=mapcontenttype&contenttypes=Research Report &channels=researchReportPdf,researchReportJpeg`		

deleteContentChannelMappings

Function	Delete publishing channel and content type mapping.		
Description	This request deletes content types to channel mappins and updates the "PublishingConfig-ext.xml" file. If content type is specified, channel mappings are deleted for this content type only. If content type is not specified, then mappings with all content types of the specified channel are deleted.		
Type	POST		
URL Format	`.../config/publishingchannels`		
Java API	**Method Name**		**Class Name**
	`deleteContentChannelMappings`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=unmapcontenttype	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`contenttype`	string	The content types to be mapped. (Mandatory)
	`channel`	string	The publishing channel. (Mandatory)
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: `http://serverName:serverPort/rest/service/config/publishingchannels ?op=unmapcontenttype&contenttypes=Research Report &channels=researchReportPdf,researchReportJpeg`		

createIndexingChannelMapping

Function	Map publishing channel with content type for indexing.		
Description	This request maps content types to indexing channels and updates the "IndexingChannels-ext.xml" file with the new mappings.		
Type	POST		
URL Format	`.../config/publishingchannels`		
Java API	**Method Name**		**Class Name**
	`createIndexingChannelMapping`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`op=mapindexingchannel`	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`contenttype`	string	The content types to be mapped. (Mandatory)
	`channels[comma-separated list of`	string	The indexing channels. (Mandatory)

	channels]		
	parameterMap	string	
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: http://serverName:serverPort/rest/service/config/publishingchannels ?op=mapindexingchannel&contenttype=Research Report &channel=researchReportJpeg&QXP_TEMPLATE_URI=classpath:ResearchReportTemplate.qxp		

getMappedIndexingChannel

Function	Get mapped indexing channels for a given content type.		
Description	This request returns the definition of the indexing channel mapped to the given content type.		
Type	GET		
URL Format	.../config/publishingchannels		
Java API	**Method Name**		**Class Name**
	getMappedIndexingChannel		com.quark.qpp.service.facade Class ConfigurationFacade
Parameters	**Name**	**Type**	**Description**
	contenttype=[contentTypeIdOrName]	string	The content type. (Mandatory)
	indexingchannel=[true|false]	boolean	
Returns	Returns a publishingChannelList object that contains a list of mapped publishing channels and its metadata.		
Examples	URL: http://serverName:serverPort/rest/service/config/publishingchannels ?contenttype=Research Report&indexingchannel=true		

deleteIndexingChannelMapping

Function	Delete publishing channel and content type mapping for indexing.	
Description	This request deletes content type to indexing channel mapping and updates the "IndexingChannels-ext.xml" file. If content type is specified, channel mapping is deleted for the given content type only. If content type is not specified, then mappings with all content types of the specified channel are deleted.	
Type	POST	
URL Format	.../config/publishingchannels	
Java API	**Method Name**	**Class Name**
	deleteIndexingChannelMapping	com.quark.qpp.service.facade

			Class ConfigurationFacade
Parameters	Name	Type	Description
	op=unmapindexingchannel	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	channel	string	The channel.
	contenttype	string	The content type.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: http://serverName:serverPort/rest/service/config/publishingchannels ?op=unmapindexingchannel&channel=researchReportJpeg&contenttype=Research Report		

deletePublishingChannels

Function	Delete publishing channel.		
Description	This request deletes the channels specified. The "ChannelConfig-ext.xml" file is updated and any mapping for the specified channels is also deleted.		
Type	POST		
URL Format	.../config/publishingchannels		
Java API	Method Name		Class Name
	deletePublishingChannels		com.quark.qpp.service.facade Class ConfigurationFacade
Parameters	Name	Type	Description
	op=delete	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	channelids	string	The channel Ids of the channels to be deleted.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: http://serverName:serverPort/rest/service/config/publishingchannels ?op=delete&channelids=researchReportPdf		

getDeploymentProperties

Function	Get deployment properties
Description	This request returns all properties specified for the deployment in the "Deployment.properties" file.
Type	GET

URL Format	`.../config/deploymentproperties`	
Java API	**Method Name**	**Class Name**
	`getDeploymentProperties`	`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	This request requires no input parameters.	
Returns	Returns a `nameValueList` object that contains all the properties.	

Examples	URL:
	`http://serverName:serverPort/rest/service/config/deploymentproperties`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <nameValueList> <nameValue name="deployment.version" value=""/> </nameValueList>`	

getDeploymentVersion

Function	Get deployment version	
Description	This request returns the version of the custom deployment.	
Type	GET	
URL Format	`.../config/deploymentversion`	
Java API	**Method Name**	**Class Name**
	`getDeploymentVersion`	`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	This request requires no input parameters.	
Returns	Returns the text value of the deployment version specified in the "Deployment.properties" file.	
Examples	URL: `http://serverName:serverPort/rest/service/config/deploymentversion` Result: 1.2	

reInitialize

Function	ReIntialize
Description	This request reinitializes the mime-type, channel and process definitions using their config files. It will reinitialize the content-mime-type mappings configuration, content type to channel mappings configuration and content type to indexing channel mappings configuration. This request may be used to apply settings assigned directly to config files without restarting the server.
Type	POST

URL Format	`.../config/`		
Java API	**Method Name**		**Class Name**
	`reInitialize`		`com.quark.qpp.service.facade Class ConfigurationFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=reinitializeconfigs	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`reinitializemimetypes` = [true|false]	boolean	
	`reinitializechannels`=[true|false]	boolean	
	`reinitializeprocesses` =[true|false]	boolean	
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: `http://serverName:serverPort/rest/service/config/?op=reinitializeconfigs&reinitializemimetypes=true&reinitializechannels=true &reinitializeprocesses=true`		

Clipboard

The Clipboard service is used to manage user-specific clipboard data on the Platform server. The server-side clipboard enables clipboard access across multiple devices and securely maintains data by user; accessible only to the user who copied the data. The service also manages clipboard data conversion to other formats.

The clipboard capabilities of the API includes the following REST calls:

- *addClipboardData*
- *addFormattedData*
- *getAllClipboardChannelsMapping*
- *getClipboardData*
- *getAllClipboardDataInfos*
- *getAllClipboardFormats*
- *getClipboardDataInfo*
- *getClipboardChannelsMappings*
- *getFormattedData*
- *getLatestClipboardData*
- *getLatestClipboardDataInfo*
- *getLatestFormattedData*
- *promoteClipboardDataToAsset*

addClipboardData

Function	Add data to clipboard.	
Description	This request adds data to the logged-in user's clipboard and returns info about the data added .	
Type	POST	
URL Format	`.../clipboard`	
Java API	**Method Name**	**Class Name**
	`addClipboardData`	`com.quark.qpp.service.facade.Clip`

			boardFacade
Parameters	**Name**	**Type**	**Description**
	op=adddata	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	contenttype=[contentTypeId] or [Name]	string	The content type **Id** or **Name** of the data.
	mimetype=[mimeType]	string	The mimeType of the data.
	inputstream	string	The data stream.
Returns	Returns a clipboardDataInfo object in XML format, containing information about the data added to the user's clipboard. See the table in the "Request/Response parameter formats" chapter.		

Examples	**URL:**
	http://server:port/rest/service/clipboard?op=adddata&mimetype=application/vnd.openxmlformats-officedocument.spreadsheetml.sheet&contenttype=Microsoft Excel Table&preview=My Excel table preview...&created by=QppAdmin&create on=12th June 2017
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<clipboardDataInfo dataId="24">`
	`<timestamp>2017-06-12T14:15:34Z</timestamp>`
	`<contentType id="110171" name="Microsoft Excel Table"/>`
	`<mimeType>application/vnd.openxmlformats-officedocument.spreadsheetml.sheet</mimeType>`
	`<clipboardMeta>`
	`<nameValue name="preview" value="My Excel table preview..."/>`
	`<nameValue name="create on" value="12th June 2017"/>`
	`<nameValue name="created by" value="QppAdmin"/>`
	`</clipboardMeta>`
	`</clipboardDataInfo>`		

addFormattedData

Function	Add formatted data.	
Description	This request adds formatted-data to the logged-in user's clipboard for the data corresponding to the data Id.	
Type	POST	
URL Format	.../clipboard/{dataId}	
Java API	**Method Name**	**Class Name**
	addFormattedData	com.quark.qpp.service.facade.Clip

			boardFacade
Parameters	Name	Type	Description
	op=addformatteddata	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	inputstream	string	The data stream.
	format	string	The format of the data.
Returns	A status of 200 is returned in response to a successful request execution.		
Examples	URL: http://server:port/rest/service/clipboard/24?op=addformatteddata&format=jpeg		

getAllClipboardChannelsMapping

Function	Get all clipboard cannel mappings.	
Description	This request retrieves all of the clipboard format to channel mappings.	
Type	GET	
URL Format	.../clipboard/channelsmapping	
Java API	Method Name	Class Name
	getAllClipboardChannelsMapping	com.quark.qpp.service.facade.ClipboardFacade
Parameters	This request requires no input parameters.	
Returns	Returns a clipboardChannelMappingInfoList object in XML format, containing a list of all clipboard format to channel mappings.	
Examples	URL: http://server:port/rest/service/clipboard/channelsmapping Result: <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <clipboardChannelMappingInfoList> <clipboardChannelMappingInfo id="fetchExcelTable" contentType="Microsoft Excel Table" clipboardFormat="svg"/> <channelParameters> <nameValue name="OUTPUT_FORMAT" value="image"/> <nameValue name="OUTPUT_FORMAT_PROPERTIES" value="imageformate=svg"/> </channelParameters> </clipboardChannelMappinInfo> <clipboardChannelMappingInfo id="fetchExcelTable" contentType="Microsoft Excel Table" clipboardFormat="png"/> <channelParameters> <nameValue name="OUTPUT_FORMAT" value="image"/> <nameValue name="OUTPUT_FORMAT_PROPERTIES" value="imageformate=png"/> </channelParameters>	

	`  </clipboardChannelMappinInfo>`
	`.....`
	`</clipboardChannelMappingInfoList>`

getClipboardData

Function	To get the clipboard data.		
Type	Get		
URL Format	`.../clipboard/{dataId}`		
Java API	**Method Name**		**Class Name**
	`getClipboardData`		`com.quark.qpp.service.facade.ClipboardFacade`
Parameters	**Name**	**Type**	**Description**
	`contentDisposition`	string	Optional value used to set content-disposition header value in response
	`outputStream`	string	output stream onto which output is to be written. This parameter is applicable only for the local invocations
Returns	Returns the data associated with the given dataId.		
Examples	**URL:** `http://server:port/rest/service/clipboard/24?`		

getAllClipboardDataInfos

Function	Get all clipboard data information.		
Description	This request fetches information about all the data present in the logged-in user's clipboard.		
Type	GET		
URL Format	`.../clipboard`		
Java API	**Method Name**		**Class Name**
	`getAllClipboardDataInfos`		`com.quark.qpp.service.facade.ClipboardFacade`
Parameters	**Name**	**Type**	**Description**
	`metadata=true`	boolean	(Mandatory)
Returns	Returns a `clipboardDataInfo`List object in XML format, containing information about all the data present in the user's clipboard.		
Examples	**URL:** `http://server:port/rest/service/clipboard? metadata=true`		

	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<clipboardDataInfoList>`
	`  <clipboardDataInfo dataId="24">`
	`  <timestamp>2017-06-12T14:15:34Z</timestamp>`
	`  <contentType id="110171" name="Microsoft Excel Table"/>`
	`  <mimeType>application/vnd.openxmlformats-officedocument.spreadsheetml.sheet</mimeType>`
	`  <clipboardMeta>`
	`  <nameValue name="preview" value="My Excel table preview..."/>`
	`  <nameValue name="create on" value="12th June 2017"/>`
	`  <nameValue name="created by" value="QppAdmin"/>`
	`  </clipboardMeta>`
	`  </clipboardDataInfo>`
	`</clipboardDataInfoList>`

getAllClipboardFormats

Function	Get all clipboard formats	
Description	This request returns all the data formats supported for clipboard data.	
Type	GET	
URL Format	`.../clipboard/formats`	
Java API	**Method Name**	**Class Name**
	`getAllClipboardFormats`	`com.quark.qpp.service.facade.ClipboardFacade`
Parameters	This request requires no input parameters	
Returns	Returns a `clipboardFormatInfoList`object containing an array of all the clipboard data formats in XML format. See the table in the “Request/Response parameter formats” chapter.	

Examples	**URL:**
	`http://server:port/rest/service/clipboard/formats`
	Result:
	`<?xml version="1.0" encoding="UTF-8"?>`
	`<clipboardFormatInfoList>`
	`  <clipboardFormatInfo>`
	`  <name>svg</name>`
	`  <mimeType>image/svg+xml</mimeType>`
	`  </clipboardFormatInfo>`
	`  <clipboardFormatInfo>`
	`  <name>png</name>`
	`  <mimeType>image/png</mimeType>`
	`  </clipboardFormatInfo>`
	`  <clipboardFormatInfo>`
	`  <name>jpeg</name>`
	`  <mimeType>image/jpeg</mimeType>`
	`  </clipboardFormatInfo>`
	`  <clipboardFormatInfo>`
	`  <name>pdf</name>`
	`  <mimeType>application/pdf</mimeType>`
	`  </clipboardFormatInfo>`
	`  <clipboardFormatInfo>`
	`  <name>cals</name>`	

	`<mimeType>application/xml</mimeType>`
	`</clipboardFormatInfo>`
	`</clipboardFormatInfoList>`

getClipboardDataInfo

Function	Get clipboard data information.		
Description	This request fetches information about the logged-in user's clipboard data using the data Id.		
Type	GET		
URL Format	`.../clipboard/{dataId}`		
Java API	**Method Name**		**Class Name**
	`getClipboardDataInfo`		`com.quark.qpp.service.facade.ClipboardFacade`
Parameters	**Name**	**Type**	**Description**
	`metadata=true`	boolean	(Mandatory)
Returns	Returns a `clipboardDataInfo` object in XML format, containing information about the clipboard data using the data **Id**.		

Examples	**URL:**
	`http://server:port/rest/service/clipboard/24? metadata=true`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<clipboardDataInfo dataId="24">`
	`<timestamp>2017-06-12T14:15:34Z</timestamp>`
	`<contentType id="110171" name="Microsoft Excel Table"/>`
	`<mimeType>application/vnd.openxmlformats-officedocument.spreadsheetml.sheet</mimeType>`
	`<clipboardMeta>`
	`<nameValue name="preview" value="My Excel table preview..."/>`
	`<nameValue name="create on" value="12th June 2017"/>`
	`<nameValue name="created by" value="QppAdmin"/>`
	`</clipboardMeta>`
	`</clipboardDataInfo>`		

getClipboardChannelsMapping

Function	Get clipboard channel mappings for content type.	
Description	This request returns the cipboard format to channel mappings based on the content type.	
Type	GET	
URL Format	`.../clipboard/channelsmapping`	
Java API	**Method Name**	**Class Name**
	`getClipboardChannelsMapping`	`com.quark.qpp.service.facade.ClipboardFacade`

Parameters	**Name**	**Type**	**Description**
	`contenttype=contentTypeIdOrName`	string	The content type.(Mandatory)
Returns	Returns a `clipboardChannelMappingInfoList` object in XML format.		

Examples	**URL:**
	`http://server:port/rest/service/clipboard/channelsmapping?contentype=Microsoft Excel Table`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<clipboardChannelMappingInfoList>`
	`<clipboardChannelMappingInfo id="fetchExcelTable" contentType="Microsoft Excel Table" clipboardFormat="svg">`
	`<channelParameters>`
	`<nameValue name="OUTPUT_FORMAT" value="image"/>`
	`<nameValue name="OUTPUT_FORMAT_PROPERTIES" value="imageformat=svg"/>`
	`</channelParameters>`
	`</clipboardChannelMappingInfo>`
	`<clipboardChannelMappingInfo id="fetchExcelTable" contentType="Microsoft Excel Table" clipboardFormat="png">`
	`<channelParameters>`
	`<nameValue name="OUTPUT_FORMAT" value="image"/>`
	`<nameValue name="OUTPUT_FORMAT_PROPERTIES" value="imageformat=png"/>`
	`</channelParameters>`
	`</clipboardChannelMappingInfo>`
	`.....`
	`</clipboardChannelMappingInfoList>`		

getFormattedData

Function	Get formatted data.		
Description	This request retrieves formatted data based on the data Id and format. The formatted data is retrieved by using default Channel Mappings if no publishing channel parameter is specified.		
Type	GET		
URL Format	`.../clipboard/{dataId}`		
Java API	**Method Name**		**Class Name**
	`getFormattedData`		`com.quark.qpp.service.facade.ClipboardFacade`
Parameters	**Name**	**Type**	**Description**
	`format`	string	The format in which clipboard data is to be retrieved.
	`publishingchannel`	string	The publishing channel **Id.**

	parametersMap	string	The publishing channel parameters.
	contentdisposition		The content disposition value to be set on the http response header.
	outputStream	string	The output stream onto which output is to be written (for local API invocations).
Returns	Returns a stream of formatted data.		
Examples	**URL:** http://server:port/rest/service/clipboard/24? format=png **Result:** A filestream of the required output format based on given parameters.		

getLatestClipboardData

Function	Get latest clipboard data.		
Description	This request retrieves the latest data that was added to the logged-in user's clipboard.		
Type	GET		
URL Format	.../clipboard		
Java API	**Method Name**		**Class Name**
	getLatestClipboardData		com.quark.qpp.service.facade.ClipboardFacade
Parameters	**Name**	**Type**	**Description**
	contentdisposition	string	The content disposition value to be set on the http response headers.
	outputStream	string	The output stream onto which output is to be written (for local API invocations).
Returns	Returns a stream of copied data into server.		
Examples	**URL:** http://server:port/rest/service/clipboard **Result:** A filestream of the latest copied data.		

getLatestClipboardDataInfo

Function	Get latest clipboard data information.
Description	This request fetches information about the latest data that was added to the logged-in user's clipboard.
Type	GET
URL Format	.../clipboard

Java API	**Method Name**		**Class Name**
	`getLatestClipboardDataInfo`		`com.quark.qpp.service.facade.ClipboardFacade`
Parameters	**Name**	**Type**	**Description**
	`metadata=true`	boolean	
	`latest=true`	boolean	
Returns	Returns a `clipboardDataInfo` object in XML format, containing information about the latest data added to the user's clipboard . See the table in the "Request/Response parameter formats" chapter.		

Examples	**URL:**
	`http://server:port/rest/service/clipboard?metadata=true&latest=true`
	Result:
	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`  <clipboardDataInfo dataId="24">`
	`  <timestamp>2017-06-12T14:15:34Z</timestamp>`
	`  <contentType id="110171" name="Microsoft Excel Table"/>`
	`  <mimeType>application/vnd.openxmlformats-officedocument.spreadsheetml.sheet</mimeType>`
	`  <clipboardMeta>`
	`  <nameValue name="preview" value="My Excel table preview..."/>`
	`  <nameValue name="create on" value="12th June 2017"/>`
	`  <nameValue name="created by" value="QppAdmin"/>`
	`  </clipboardMeta>`
	`  </clipboardDataInfo>`		

getLatestFormattedData

Function	Get latest formatted data.		
Description	This request retrieves the latest data added to the logged-in user's clipboard in the specified format. The formatted data is retrieved by using default Channel Mappings if no publishing channel parameter is specified.		
Type	GET		
URL Format	`.../clipboard`		
Java API	**Method Name**		**Class Name**
	`getLatestFormattedData`		`com.quark.qpp.service.facade.ClipboardFacade`
Parameters	**Name**	**Type**	**Description**
	`format`	string	The format in which clipboard data is to be retrieved.
	`publishingchannel`	string	The publishing channel **Id**.
	`parametersMap`	string	The publishing channel parameters.

	contentdisposition		The content disposition value to be set on the http response header.
	outputStream	string	The output stream onto which output is to be written (for local API invocations).
Returns	Returns a stream of formatted data.		
Examples	**URL:** http://server:port/rest/service/clipboard? format=png **Result:** A filestream of the required output format based on given parameters.		

promoteClipboardDataToAsset

Function	Promote clipboard data to asset		
Description	This request promotes the user's clipboard data to Platform's asset.		
Type	POST		
URL Format	.../clipboard/{dataId}		
Java API	**Method Name**		**Class Name**
	promoteClipboardDataToAsset		com.quark.qpp.service.facade.ClipboardFacade
Parameters	**Name**	**Type**	**Description**
	op=promotetoasset	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	createminorversion	boolean	
	attributevalues	string	
Returns	Returns an assetInfo object in XML format, containing the metadata of the newly created asset. See the table in the "Request/Response parameter formats" chapter.		
Examples	**URL:** http://server:port/rest/service/clipboard/{DataId}?loginname=Admin&loginpassword=Admin&op=promotetoasset&attributevalues=<xml/> **RESULT:** <?xml version="1.0" encoding="UTF-8" standalone="yes"?> <assetInfo id="220"> <assetVersion majorVersion="1" minorVersion="0"/> <attributeValueList>		

	`<attributeValue id="1" name="Id" type="numeric">220</attribute-Value>`
	`.....`
	`</attributeValueList>`
	`<articleComponentInfoList/>`
	`</assetInfo>`

JSON support in request parameters for POST requests.

Both XML and JSON can be used to pass request parameters.

The following are some examples:

createCollection

Function	Create a collection.		
Description	This request creates a new collection according to the given attributes.		
Type	POST		
URL Format	`.../collections/[collection name] or collectionsbypath/[collection path]`		
Header	qpp-content-type:json		
Java API	**Method Name**		**Class Name**
	`createCollection`		`com.quark.qpp.service.facade Class CollectionFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=create	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`targetparent`=[collection path or ID]	long or string	The parent collection for the new collection. (Mandatory)There is a choice between collection ID, Name and Path when providing a value.
	`attributevalues`=<xml/>	string	The set of attribute values.: See “Request/Response parameter formats.”(All atributes are Optional)
	`contenttype`=[content type name]	string	Defaults to `GENERAL`. The default content type IDs are documented in the JavaDoc SDK at com.quark.qpp.core.content. service.constants.DefaultContentTypes. To get a list of content types, use a call to ../admin/contenttypes.(Optional)There is a choice between content type ID, Name and Hierarchy when providing a value.

Returns	Returns the `collectionInfo` object of the newly created collection.
Examples	**URL:** `http://serverName:serverPort/rest/service/collections?op=create&collectioninfolist=` `{"CollectionInfoList": {"collectionInfo": [{"assetInfoList":null, "attributeValueList": {"attributeValue": [{"id":2,"multiValued":false,"name":"", "sourceReference":null,"type":"", "value":"Collection_5800a","valueId":0}, {"id":59,"multiValued":false,"name":"", "sourceReference":null,"type":"", "value":"False","valueId":0}, {"id":57,"multiValued":false,"name":"", "sourceReference":null,"type":"", "value":"Home/Collection_5800a","valueId":0}, {"id":62,"multiValued":false,"name":"", "sourceReference":null,"type":"", "value":null,"valueId":1416}] },"collectionClass":null,"collectionClassId":1416, "collectionInfoList":null, "collectionPath":"Home/Collection_5800a", "defaultJobJacketId":1,"defaultJobJacketName":null, "defaultJobJacketTicket":"Default Job Ticket", "domainId":19,"groupInfoList":null,"id":0, "jobJacketInfoList":null,"name":"Collection_5800a", "parentColInfoList":null,"revisionControlInfoList":null, "routingInfoList":null,"templateId":0,"userInfoList":null, "workflowInfoList":null }] }` `}Input:{"CollectionInfoList": {"collectionInfo": [{"assetInfoList":null, "attributeValueList": {"attributeValue": [{"id":2,"multiValued":false,"name":"", "sourceReference":null,"type":"", "value":"Collection_5800a","valueId":0}, {"id":59,"multiValued":false,"name":"", "sourceReference":null,"type":"", "value":"False","valueId":0}, {"id":57,"multiValued":false,"name":"", "sourceReference":null,"type":"", "value":"Home/Collection_5800a","valueId":0}, {"id":62,"multiValued":false,"name":"", "sourceReference":null,"type":"", "value":null,"valueId":1416}] }, "collectionClass":null,"collectionClassId":1416, "collectionInfoList":null, "collectionPath":"Home/Collection_5800a", "defaultJobJacketId":1, "defaultJobJacketName":null, "defaultJobJacketTicket":"Default Job Ticket", "domainId":19,"groupInfoList":null,"id":0, "jobJacketInfoList":null, "name":"Collection_5800a", "parentColInfoList":null,"revisionControlInfoList":null, "routingInfoList":null,"templateId":0,"userInfoList":null, "workflowInfoList":null}] } }`|

checkinNew

Function	Check in a new asset.		
Description	Include the file in the request body or as file part.		
Type	POST		
URL Format	`.../assetsbypath/[asset path] or /assets/[New Asset Name]`		
Header	qpp-content-type: json		
Java API	**Method Name**		**Class Name**
	`checkInNew`		`com.quark.qpp.service.facade Class AssetFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=checkinnew	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`attributevalues`		See "checkinNewRequest/Response parameter formats." The following attributes are Mandatory:Content type (There is a choice between content type ID, Name and Hierarchy when providing a value for the Content type) attribute. Collection (There is a choice between collection ID, Name and Path when providing a value for the `Collection` attribute). WorkflowStatusFile extensionOriginal filename
	`relations`		See "Request/Response parameter formats." If this parameter is passed, the Platform Server will check-in the asset and create child relations with the new asset being checked in at the same time. (Optional).
	`createminorversion` =[true | false]	boolean	Specify if this version of the asset is to be checked in to the QPP server as a major or a minor version. (Optional)
Returns	Returns an assetInfo object of the newly checked in asset with its metadata.		

Examples	**URL:**
	`http://myplatform:61400/rest/service/assets/Platform 11.2 What's New.pptx? op=checkinnew&attributevalues=`
	`{"AttributeValueList":{...}}`
	`Input:{"AttributeValueList": {"attributeValue":[{ "id":62, "multiValued":false, "name":"", "sourceReference":null, "type":"", "value":"System;Asset;Project;QuarkXPress Project;", "valueId":0 }, { "id": 54, "multiValued": false, "name": "", "sourceReference": null, "type": "", "value": null, "valueId": 11 }, { "id": 24, "multiValued": false, "name": "", "sourceReference": null, "type": "" , "value": null, "valueId": 13 }, {`		

	"id": 33, "multiValued": false, "name": "", "sourceReference": null, "type": "", "value": "15116.qxp", "valueId": 0 }, { "id": 25, "multiValued": false, "name": "", "sourceReference": null, "type": "", "value": null, "valueId": 1 }, { "id": 11, "multiValued": false, "name": "", "sourceReference": null, "type": "", "value": "qxp", "valueId": 0 }, { "id": 55, "multiValued": false, "name": "", "sourceReference": null, "type": "", "value": null, "valueId": 1 }] } }

createDomains

Function	Create Attribute Domains.		
Description	This request is used to create a single or multiple attribute domains at one time.		
Type	POST		
URL Format	`.../admin/domains`		
Header	qpp-content-type: json		
Java API	**Method Name**		**Class Name**
	`createDomains`		`com.quark.qpp.service.facade Class AdminFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=create	string	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`attributedomaininfolist`	string	List of attribute domains to create.
Returns	Returns a `attributeDomainInfoList` object containing a list of successfully created attribute domains.		
Examples	**URL:** `http://myplatform:61400/rest/service/admin/domains?op=create&attributedomaininfolist= Input:` {AttributeDomainInfoList": {"attributeDomainInfo": [{"domainValueList": {"domainValue": [{"domainValueList": {"domainValue": [{"domainValueList": {"domainValue": [{"domainValueList": {"domainValue":[]}, "id": 0, "name": "7", "sourceReference": "source7", "valueHierarchy": null }] }, "id": 0, "name": "4", "sourceReference": "source4", "valueHierarchy": null }] }, "id": 0, "name": "1", "sourceReference": "source1", "valueHierarchy": null }, {"domainValueList": {"domainValue": [{"domainValueList": {"domainValue": [{"domainValueList": {"domainValue":[]}, "id": 0, "name": "8", "sourceReference": "source8" "valueHierarchy": null }] }, "id": 0, "name": "5", "sourceReference": "source5", "valueHierarchy": null }] }, "id": 0, "name": "2", "sourceReference": "source2", "valueHierarchy": null }, {"domainValueList": {"domainValue":		

	[{"domainValueList": {"domainValue": [{"domainValueList": {"domainValue":[]}, "id": 0, "name": "9", "sourceReference": "source9", "valueHierarchy": null }] }, "id": 0, "name": "6", "sourceReference": "source6", "valueHierarchy": null }] }, "id": 0, "name": "3", "sourceReference": "source3", "valueHierarchy": null }] }, "expandable": true, "hierarchical": true, "id": 0, "itemsSorted": false, "name": "SampleQPP_0cb", "sourceReference": null, "systemDefined": false, "userModifiable": false, "warnBeforeExpansion": true }] } }

createSearches

Function	Create searches.		
Description	This request creates a single or multiple searches at one time.		
Type	POST		
URL Format	`.../searchdefinitions`		
Header	qpp-content-type: json		
Java API	**Method Name**		**Class Name**
	`createSearches`		`com.quark.qpp.service.facade Class QueryDefinitionFacade`
Parameters	**Name**	**Type**	**Description**
	`op`=create	long	The op parameter allows you to specify which operation is to be performed. (Mandatory)
	`searchinfolist`=[xml]	string	An XML object containing metadata of searches to be created.
Returns	Returns a `searchInfoList` object containing metadata of newly created searches.		
Examples	**URL:** `http://myplatform:61400/rest/service/searchdefinitions?op=create&searchinfolist= Input:` {"SearchInfoList": {"searchInfo": [{"id": 0,"owner": {"id": 0, "value": null}, "parameterized": false, "queryConditionList": {"queryCondition":[]}, "queryContext": {"collections": {"collection": [{"id": 0,"name": "Home","path": null}], "recursive": true }, "context": "ASSET" }, "queryDisplay": {"displayColumnList": {"displayColumn": [{"attributeColumn": true, "columnId": 2, "value": null, "width": 0 }, {"attributeColumn": true, "columnId": 55, "value": null, "width": 0 }, {"attributeColumn": true, "columnId": 54, "value": null, "width": 0 }, {"attributeColumn": true, "columnId": 24, "value": null, "width": 0 }] }, "displayMode": "PLAIN", "exploreMode": "PLAIN", "groupingAttributeList": {"groupingAttribute": [{"attributeId": 55, "order": "ascending", "value": null }] }, "sortInfoList": {"sortInfo": [{"attributeColumn": true, "columnId":		

	2, "order": "ascending", "value": null }] } }, "searchName": "SampleQuery_90d", "trusteeInfoList": {"trusteeInfo": null} }] } }

Change Log

Changes in version 14.0

Changes

- The `publishAsset` API has been enhanced to support publishing previous versions of an asset, by adding two parameters:
  - majorversion
  - minorversion
- A correction was made to the example URL for the `createRepositories` method.

Changes in version 13.2

Changes in version 13.2 included the following:

Additions

The following additions were made to the REST API in version 13.2.

Service	Method Name	Function	Reference
Asset facade	reIndex	Re-index an asset using the asset id or the asset path.	"*Asset management*"

Changes

- The `getSearchResultsForQuery` API has been enhanced to support returning results in a paginated manner.
- The `search` API has been enhanced to support returning results in a paginated manner.
- The `search`, count API have been modified to accomodate a large amount of returning results.

Changes in version 13.1

Changes in version 13.1 included the following:

Additions

The following additions were made to the REST API in version 13.1.

Service	Method Name	Function	Reference
Clipboard facade	promoteClipboardDataToAsset	Promote clipboard data to asset.	"*Clipboard*"

Changes in version 13.0

Changes in version 13.0 included the following:

The `PublishingFacade` bean API has been updated which could result in a compile error if using this as a Spring bean.

The method signatures in `PublishFacade` and `PublishByPathFacade` were modified by adding the `httpServletResponse` argument. This affects the following REST calls:

- `publishAsset`
- `publishBasedOnStreamContext`
- `publishBasedOnURI`
- `executeDatadocChannel`

This argument sets the appropriate content-type header values in the http response for publishing requests.

Although this change impacts no REST URLs, any custom code using these methods by injecting `PublishFacade` and `PublishByPathFacade` as a Spring bean will encounter compile errors. These errors can be avoided by passing the argument as null.

Additions

The following additions were made to the REST API in version 13.0.

Service	Method Name	Function	Reference
Office facade	getExcelChart	Get Microsoft Excel chart	"Office"
	getExcelChartBasedOnStream	Get Microsoft Excel chart based on stream	
	getExcelChartBasedOnURI		

Service	Method Name	Function	Reference
		Get Microsoft Excel chart based on URI	
	getExcelTable	Get Microsoft Excel table	
	getExcelTableBasedOnStream	Get Microsoft Excel table based on stream	
	getExcelTableBasedOnURI	Get Microsoft Excel table based on URI	
Clipboard facade	addClipboardData	Add data to clipboard	"Clipboard"
	addFormattedData	Add formatted data	
	getAllClipboardChannelsMappi ng	Get all clipboard format to channel mappings	
	getAllClipboardDataInfos	Get all clipboard data information.	
	getAllClipboardFormats	Get all clipboard formats	
	getClipboardDataInfo	Get clipboard data information	
	getClipboardChannelsMappings	Get clipboard channel mappings for content type	
	getFormattedData	Get formatted data	
	getLatestClipboardData	Get latest clipboard data	
	getLatestClipboardDataInfo	Get latest clipboard data information	
	getLatestFormattedData	Get latest formatted data	

Changes in version 12.0

Changes in version 12.0 included the following:

Additions

The following additions were made to the REST API in version 12.0.

Service	Method Name	Function	Reference
Asset facade	generateTextIndexesOfAllAssets	Generate text indexes for all assets.	"Asset Manageme nt"
	generateTextIndexes	Generate text indexes for a specified asset.	

Service	Method Name	Function	Reference
Office facade	getVisioPagesInfo	Get metadata of all pages of a Visio asset.	"Office"
	getVisioPagesInfoBasedOnURI	Get metadata of all pages of a Visio asset based on URI.	
	getVisioPagesInfoBasedOnStreamContext	Get metadata of all pages of a Visio asset based on stream.	
	getVisioPageOutput	Get page output of a Visio asset.	
	getVisioPageOutputBasedOnURI	Get metadata of all pages of a Visio asset based on URI.	
	getVisioPageOutputBasedOnStreamContext	Get page output of a Visio asset based on stream.	
	getPowerPointSlidesInfo	Get metadata of all slided of a PowerPoint asset.	
	getPowerPointSlidesInfoBasedOnURI	Get metadata of all slides of a PowerPoint asset based on URI.	
	getPowerPointSlidesInfoBasedOnStreamContext	Get metadata of all slides of a PowerPoint asset based on file stream.	
	getPowerPointSlideOutput	Get slide output of a PowerPoint asset.	
	getPowerPointSlideOutputBasedOnURI	Get slide output of a PowerPoint asset based on URI.	
	getPowerPointSlideOutputBasedOnStreamContext	Get slide output of a PowerPoint asset based on file stream.	
	getPowerPointSlideRangeOutput	Get slide range output of a PowerPoint asset.	
	getPowerPointSlideRangeOutputBasedOnURI	Get slide range output of a PowerPoint asset based on URI.	
	getPowerPointSlideRangeOutputBasedOnStreamContext	Get slide range output of a PowerPoint asset based on file stream.	
Configuration facade	getAllXmlMimeTypes	Get XML MimeTypes	"Configuration"
	getXmlMimeType	Get XML MimeType	
	createXmlMimeType	Create a custom XML MimeType.	
	getMappedMimeTypes	Get mapped MimeTypes	
	mapMimeTypes	Map custom XML MimeType with ContentType.	
	unmapMimeTypes	Unmap custom XML MimeType with ContentType.	
	deleteXmlMimeType	Delete custom XML MimeType.	
	getPublishingProcess	Get publishing process	
	createPublishingProcess	Create publishing process	
	deletePublishingProcess	Delete publishing process.	

Service	Method Name	Function	Reference
	getAllPublishingChannelDefinitions	Get all publishing channels	
	getPublishingChannelDefinition	Get publishing channel	
	getPublishingChannelDefinitions	Get the publishing channel of a specific content type.	
	createpublishingChannels	Create new publishing channels	
	createContentChannelMappings	Map publishing channel with content type for publishing.	
	deleteContentChannelMappings	Delete publishing channel and content type mapping.	
	createIndexingChannelMapping	Map publishing channel with content type for indexing.	
	getMappedIndexingChannel	Get mapped indexing channels for a given content type.	
	deleteIndexingChannelMapping	Delete publishing channel and content type mapping for indexing.	
	deletePublishingChannels	Delete publishing channel.	
	getDeploymentProperties	Get deployment properties	
	getDeploymentVersion	Get deployment version	
	reInitialize	ReIntialize the MimeType	

Changes in version 11.2

Changes

- Added support for JSON format.

Changes in version 11.0

Changes in version 11.0 included the following:

Additions

The following additions were made to the REST API in version 11.0.

Service	Method Name	Function	Reference
Collection facade	getApplicablePrivileges	Get applicable privileges for the logged in user.	"Collection Management"
	getApplicableRoutingGroups	Get applicable routing groups for the logged in user.	
	getApplicableAttributeCon	Get applicable attribute constraints	

Service	Method Name	Function	Reference
	straints	for the logged in user.	
	getApplicableStatusTransition	Get applicable status transitions for the logged in user.	
	getApplicableWorkflowInitialStatuses	Get applicable workflow initial statuses.	
Admin facade	updateUserEnabled	Enables/disables the Platform Server Logon for the given user.	"Administration"
	getGroupsWithGivenUserClass	Get groups with the given role.	
	getUserGroups	Get groups of a user.	
	getLdapGroups	Get LDAP groups.	
	importGroupsFromLdap	Get LDAP groups.	
	synchonizeLdapUsesrGroups	Synchronize all LDAP users and groups.	
	synchronizeLdapUsersGroupsBasedOnProfile	Synchronize all LDAP users and groups.	
	createLdapSearches	Create LDAP search definitions.	
	getLdapSearch	Get LDAP search definition.	
	getAllLdapSearches	Get all saved LDAP search definitions.	
	updateLdapSearches	Update LDAP search definitions.	
	deleteLdapSearch	Delet an LDAP search definition.	
	updateTrusteeRule	Update role for the given trustee.	
	getApplicableApplicationPrivileges	Get applicable application privileges for the logged in user.	
	getApplicablecontentPrivileges	Get applicable content privileges for the logged in user.	

Changes

- The `userInfo` XML has been modified to support LDAP groups functionality.
- The `groupInfo` XML has been modified to support LDAP groups functionality.
- The `trusteeInfo` XML has been modified to support LDAP groups functionality.
- The `domainValue` XML has been modified to populate a new element, `valueHierarchy` referring to the hierarchy of a domain value.

➡ Population of this unique domain value hierarchy is helpful in asset metadata update when the domain value id is not known and there exists multiple domain values with the same name.

- The `ldapProfileInfo` XML has been modified to support LDAP groups functionality.

- In order to set the attribute value of a domain type attribute during check in of an asset: asset metadata update, asset checkout etc., the `attributeValue` XML has been enhanced to support the following options in priority:
  - Specify domain value via hierarchy:

```
<attributeValueList>
  <attributeValue
name="Departments">Enterprise;Server</attributeValue>
</attributeValueList>
```

  - Specify domain value via name:

```
<attributeValueList>
  <attributeValue
name="Departments">Server</attributeValue>
</attributeValueList>
```

  - Specify domain value via id:

```
<attributeValueList>
  <attributeValue name="Departments"
valueId="1"></attributeValue>
</attributeValueList>
```

➡ If the CDATA has a value mentioned, it will be assumed to be a domain value hierarchy first and if not resolved, then a domain value name. The valueId attribute will only be considered if the CDATA is empty for that <attributeValue> node for a domain type attribute.

Deletions

- The `getOverriddenUserClass(long collectionId)` request has been removed.

  In the current model, a users privileges is a union of the following roles:

  - Role of the logged in user in the given collection, only if the user is mapped to the given collection.
  - Role of collection mapped groups of which the logged in user is a member.

  The `getApplicablePrivileges` request should be used to get applicable privileges for the logged in user for the given collection.

Contacting Quark

The support portal allows you to log support tickets, track tickets, receive status notifications, chat with a technical support representative, search the Knowledge Base, and access product documentation.

With direct access to documentation across all Quark software - from QuarkXPress and App Studio to Quark Enterprise Solutions - you can find answers to your questions at your convenience. Our support team is also available to help, either through our support portal, or via phone for our maintenance contract customers.

If you are a Quark customer and have a current maintenance or support contract your account has already been created for you using your registered email address. If you do not have a support contract you can purchase a single support incident to get your problem resolved. If you have purchased or registered a supported product, you are eligible for free support for the first 90 days.

In the Americas

For more details, please check out our support website *www.quark.com/support*

Outside the Americas

For countries outside the Americas, please visit the following sites to access your support account:

Support Website

- France - *www.quark.com/fr/support*
- Germany - *www.quark.com/de/support*

Legal notices

© 2022 Quark Software Inc. and its licensors. All rights reserved.

Protected by the following United States Patents: 5,541,991; 5,907,704; 6,005,560; 6,052,514; 6,081,262; 6,633,666 B2; 6,947,959 B1; 6,940,518 B2; 7,116,843; 7,463,793; and other patents pending.

Quark, the Quark logo, Quark Publishing Platform are trademarks or registered trademarks of Quark Software Inc. and its affiliates in the U.S. and/or other countries. All other marks are the property of their respective owners.