Oracle® DIVA Enterprise Connect

Web Services API Programmer's Guide

Release 1.0

E88109-02

August 2017

ORACLE®

Oracle DIVA Enterprise Connect Web Services API Programmer's Guide, Release 1.0

E88109-02

Primary Author: Lou Bonaventura

Contents

4 Digital Asset Requests

5 Device Requests

6 Informational Commands

7 Miscellaneous Commands

List of Tables

Preface

This document describes using the DIVA Web Services API. You can use this API to connect to Oracle DIVA services on-premise and in the cloud. The Oracle DIVA Enterprise Connect software provides a binding for this API. This book includes a high-level introduction to the SOAP and REST services, and a detailed reference for all of the calls.

See the *Oracle DIVA Enterprise Connect Installation, Configuration, and Administration Guide* in the *Oracle DIVA Enterprise Connect 1.0 documentation* library for details on the installation, configuration, and administration of the DIVA Enterprise Connect.

The first chapter of this document provides an overview of the SOAP and REST services. The remaining chapters are references for each service, including services that process content (Digital Asset Requests), services that return information (Informational Commands), and so on.

Audience

This document is intended for customers, developers, and partners writing software against the DIVA Web Services API.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the Oracle DIVArchive documentation set for this release located at https://docs.oracle.com/en/storage/#csm.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction

This chapter describes an overview of the Oracle DIVA Enterprise Connect, the DIVA Web Services API, and includes the following information:

DIVA Web Services API Overview

The DIVA Web Services API is a standards-based Web Service API that connects to Oracle DIVArchive or Oracle DIVAnet systems, acting as a web service binding for the DIVArchive API. The DIVA Web Services API provides a language and operating system independent method of submitting requests to archive, restore, copy, or delete digital assets. The API also provides services to list archive objects, list archive device status (tape libraries and disks), and obtain the status of requests issued to DIVArchive. The API includes both SOAP and REST bindings.

The web services use basic authentication by default. Oracle recommends using basic authentication with an SSL/TLS transport (the default is port 9444 in Web Services). *See Authentication for more information.*

Oracle DIVArchive and Oracle DIVAnet

Oracle DIVArchive is a Content Management Solution specifically engineered for archiving, tracking, and restoring large amounts of rich media and broadcast digital assets. It integrates with a multitude of industry standard software, archive media, transfer protocols, hardware platforms, and the Cloud. Oracle DIVArchive enables organizations to manage the lifecycle of their digital assets through automated policies that move data to the appropriate tier of storage based on access patterns, age, and storage utilization.

Oracle DIVAnet provides a unified view of archived digital assets across multiple, distributed DIVArchive systems, and the Cloud. It facilitates moving content back and forth among DIVArchive sites, and to and from customer Source/Destination servers and disks. DIVAnet performs its tasks for disaster recovery, content distribution, access control, performance, and content availability.

By connecting to multiple DIVArchive sites, DIVAnet creates a virtual archive system that spans geographical locations.

Understanding Oracle DIVA Enterprise Connect

Oracle DIVA Enterprise Connect is a software package that provides an implementation of the DIVA Web Services API. This implementation enables client applications to submit requests to a remote Oracle DIVArchive or Oracle DIVAnet system. Web services are software modules that you can invoke remotely using a standard messaging format; in this case, XML. The client applications can use standard internet protocols (for example, HTTP and URLs). The DIVA Enterprise Connect platform consists of Oracle WebLogic, installation and administration scripts, data files, and configuration.

When a request is sent to DIVA Enterprise Connect, the request is forwarded to either DIVArchive or DIVAnet. The response is returned as an XML document back to the caller, in either REST or SOAP format.

Oracle DIVA Enterprise Connect Release Compatibility

DIVA Enterprise Connect 1.0 implements the DIVA Web Services 2.2 API, and is compatible with DIVArchive 7.4 and later (Oracle recommends DIVArchive 7.5 and later), and DIVAnet 2.1 and later. Earlier versions of the DIVA Web Services API are not supported on the DIVA Enterprise Connect platform. DIVAnet 2.2 and later is required for the HTTP-based transport to the DIVAnet ManagerAdapter.

The current DIVA Enterprise Connect software release is 1.0. DIVA Enterprise Connect 1.0 supports the 2.2 version of the DIVArchive Web Services API; earlier API versions are not supported on release 1.0. This is true for both SOAP and REST.

The DIVArchive Web Services 2.2 API contains the same basic information elements as the 2.1 version, but applications written against version 2.1 may not be compatible with version 2.2. Some minor changes from version 2.1 can be expected; for example, different return code behavior, and slightly different naming for namespaces in XML responses. Other modifications include changes in service authentication, HTTP return codes, HTTP header fields that are now strictly required (for example, `Content-Type`), a new 2.2 WSDL document, and new security restrictions. SOAP 1.2 bindings are no longer supported.

Some deviations from version 2.1 will require more effort to remedy. For example, passing parameters in the URL is no longer supported, and the URL format has changed slightly from the 2.1 release. *See Chapter 2 for more details on these changes*.

Deprecated Commands

The following commands available in the DIVArchive Web Services 2.1 API are deprecated:

- `insertTapeShort()`
- `addGroupShort()`
- `deleteInstanceByMediaName()`

See Chapter 7 for more information on the deprecated commands.

2

Using the DIVArchive Web Services API

This chapter describes an overview of the DIVArchive Web Services API. *See Chapter 1 for an overview of DIVA Enterprise Connect, the platform, and release compatibility.*

DIVArchive Web Services API Conventions

The DIVArchive Web Services API has both SOAP and REST implementations. The SOAP and REST APIs are both based on XML, and have similar parameter names and data types. To understand how to effectively use the DIVArchive Web Services API, it is useful to first understand some conventions employed by both the SOAP and REST implementations.

Session Management

Every DIVArchive Web Services API request contains a `sessionCode` parameter, which must contain a valid and active session code. The session code is passed as content in the payload of the HTTP request, not in the HTTP request header. You can establish a session using the `registerClient()` call. A successful `registerClient()` response contains a new `sessionCode`. You can issue the following REST call to create a session:

```
curl -v -X POST \
-H "Content-Type: application/xml" \
--data \
'<p:registerClient xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
```

```
   <p:appName>MediaDatabase2</p:appName>
   <p:locName>USWest2</p:locName>
   <p:processId>5828</p:processId>
</p:registerClient>' \
 http://127.0.0.1:9443/diva/service/rest/2.2/DIVArchiveWS_REST/registerClient
```

All sessions expire after a certain period (the default is 30 minutes). The DIVArchive Web Services configuration file has two timeout parameters. The `divas.service.session.maxAge` parameter determines the maximum total age of a session (in seconds) before it is terminated. The `divas.service.session.maxIdle` parameter determines how long a session can be idle before being terminated. Oracle recommends that you set these two parameters to the same value.

When a session is no longer valid, it is the client application's responsibility to create another session by calling `registerClient()` again.

Content Type and Character Sets

The `Content-Type` parameter is required for all service requests. The `Content-Type` parameter is an HTTP header field that identifies the type of content being sent. You must specify `text/xml` as the `Content-Type` to invoke the services using XML input. If you want to pass parameters using REST form parameters, specify a `Content-Type` of `application/x-www-form-urlencoded`.

The character set for all input parameters should be encoded using `UTF-8`. Appending `charset: UTF-8` to the `Content-Type` is optional.

The returned parameters are in XML format, encoded using the `UTF-8` character set. The services do not support XML returned in alternate character sets, for example, `iso-8859-1`.

Defaults and Required Parameters

Most DIVArchive Web Services API request parameters are required to be present in the API request. Very few parameters are optional in the XML schema (for both SOAP and REST). However, some required parameters can be empty or nil. These parameters are identified as *nillable* in the XSDs that are referenced in the WSDL. Some parameters have defaults that are applied when no value is provided. For example, if the priority parameter is passed as `nil`, the system uses the default specified in the DIVArchive configuration. If the `filePathRoot` is not supplied, some requests use the default stored in the archived object.

To pass an empty or nil value for *String* parameters, the client can pass an empty string as follows:

```
<filePathRoot></filePathRoot>
```

However, this technique is not valid for numeric values. You must use the `nil` attribute to pass an empty numeric value. The following example passes an empty priority value:

```
<priorityLevel xsi:nil="true"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
```

You can omit the namespace declaration previously described if the namespace is declared at the top of the XML.

You can use `-1` as the value for some parameters; the `-1` indicates that the value is not specified. Parameters that accept `-1` include `priorityLevel` and `instanceID`. *Check the documentation if you choose to use this method*.

Other parameters in the API contain entire data structures that are optional. You may be able to omit these sections in their entirety in the request, because the sections may not apply, depending the request. These parameters and sections are identified in the XSDs with `minOccurs=0`.

The order that you specify the XML parameters in both SOAP and REST requests is important. You must use the parameter order defined in the WSDL or `wsTest` scripts. However, the order of REST form parameters is not important.

Authentication

The DIVArchive Web Services API is configured to require basic authentication by default. You create the API users using the Key Generator tool. The Key Generator generates a client ID and authentication key, and then stores the user information in the configured WebLogic directory.

API clients must pass the credentials (the client ID and authentication key) in the standard Basic authentication format to have service calls properly authenticated. Both SOAP and REST are authenticated using the same approach; passing the basic authentication credentials in an HTTP header. The following is an example of a REST service invocation with credentials supplied:

```
curl -v -X POST \
-H "Content-Type: application/x-www-form-urlencoded" \
-u "4926d1cc-4f92-3b9c-d421-f958a60a832b:k89kcQWPbDES/tX10ged" \
--data sessionCode=35c3f814-cbe8-421c-ac07-9209709c960c&requestNumber=1678  \
 https://127.0.0.1:9444/diva/service/rest/2.2/DIVArchiveWS_REST/cancelRequest
```

In the previous call, the cURL `-u` parameter provides basic authentication information in the format `user_name:password`. The authentication information is encoded by the cURL utility before it is actually transmitted.

DIVArchive Web Services requires that the client ID and authentication key credentials be passed on every Web Service call, including the `registerClient()` call and WSDL document requests. The `sessionCode` does not appear in the HTTP header; it is present in the body of the HTTP request.

DIVArchive SOAP Services

The DIVArchive SOAP web services are based on SOAP version 1.1 (Simple Object Access Protocol), a standardized way of structuring XML messages. SOAP messages have a structure composed of an envelope, a header, and body section. The interface of SOAP version 1.1 services is specified in a WSDL document (Web Services Description Language). The WSDL document is a standard way of programmatically describing a set of web services.

WSDL Documents

The WSDL describes the available services (request messages), the response messages for those services, and the data flowing on those messages. A WSDL in turn can refer to zero or more XSD (XML Schema Definition) documents. These provide detailed descriptions of the specific input parameters, output parameters, and data types present in the services.

To access or view the WSDL document for the 2.2 Web Services, access the following URL in a web browser (substituting the correct network address, or port, or both):

`http://127.0.0.1:9443/diva/service/soap/2.2/DIVArchiveWS_SOAP?WSDL`

If you use a web browser, you will likely need to supply a username and password to access the WSDL. Pass the client ID as the username, and the authentication key as the password. You can also access the individual XSD documents using the following URLs (the XSD documents are numbered in the WSDL document):

http://127.0.0.1:9443/diva/service/soap/2.2/DIVArchiveWS_SOAP?xsd=1

The advantage of a WSDL document is that clients can download it, and use it to generate client-side code. The code takes care of parsing and generating XML messages, and facilitates programmatic access to the services. Many software development environments have tools to generate client-side code from the WSDL. The following link shows how to generate a simple Web Service client from a WSDL document using JAX-WS (a standard Oracle Java Web Services library):

https://docs.oracle.com/cd/E17802_01/webservices/webservices/docs/2.0/tutorial/doc/JAXWS3.html

Then following link describes how to generate Java artifacts for using standard Oracle Java SE 8 tools:

https://docs.oracle.com/javase/8/docs/technotes/tools/unix/wsimport.html

SOAP XML Requests

The DIVArchive Web Services accept SOAP request messages sent using HTTP, with POST as the request method. The content of these messages are in XML SOAP format. The service endpoint is the URL used in the POST. The actual Web Service name (method name) is not specified in the URL; it is specified in the XML request. You must specify text/xml as the *Content-Type* to invoke the services. If you are generating client-side code using the WSDL, it is likely that this step is performed for you. The following is an example of a simple SOAP HTTP request issued to DIVArchive Web Services:

```
curl -v -X POST \
-H "Content-Type: text/xml" \
-u "4926d1cc-4f92-3b9c-d421-f958a60a832b:k89kcQWPbDES/tX10ged" \
--data \
'<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd"
<soapenv:Header/>
   <soapenv:Body>
      <p:cancelRequest>
            <p:sessionCode>35c3f814-cbe8-421c-ac07-9209709c960c</p:sessionCode> \
            <p:requestNumber>1678</p:requestNumber>\
      </p:cancelRequest>
      </soapenv:Body>
   </soapenv:Envelope>' \
 http://127.0.0.1:9443/diva/service/soap/2.2/DIVArchiveWS_SOAP \
```

Error Handling

There are two main ways errors are handled in the DIVArchive SOAP services:

SOAP Fault

For fundamental errors such as authentication issues, XML message formatting issues, and some validation-related issues, the DIVArchive SOAP Web Services generates a *SOAP Fault*. A SOAP fault is an XML block returned as a response to a request. The

structure of the fault is detailed in the WSDL document. Clients using libraries to generate code against the WSDL often have the ability to have a client-side exception generated (in the client's programming language) when a fault is returned as a response. If a request generates a SOAP fault, the HTTP return code will be HTTP code 500.

DIVArchive API Error

If the request is parsed correctly and sent to DIVArchive, the response may still contain an error. The request may reference an invalid object, or contain an illegal combination of parameters. In these cases, the DIVArchive SOAP Web Services return an architected DIVArchive API response that contains a DIVArchive API error code. The architected response for each request is documented in the reference chapters. The HTTP return code for these responses will be HTTP code 200. *See Appendix A for the full list of DIVArchive API status codes*.

DIVArchive REST Services

DIVArchive REST (REpresentational State Transfer) services are similar to the DIVArchive SOAP services. You can invoke the REST services in two ways. The first method involves an HTTP POST with an XML-formatted request message. This message looks very much like the SOAP messages (minus the wrapping envelope and body tags). The second method of invocation involves passing input parameters using HTTP form parameters. In both methods, the URL contains the name of the method (for example, `getArrayList`).

Regardless of how the request is submitted, either an XML architected response message is returned (with a successful HTTP error code), or a response with a non-successful HTTP error code is returned. When an HTTP error code is returned, the body may have additional error text that can assist in diagnosing the issue.

REST XML Requests

The primary method of invoking a DIVArchive REST call is by submitting an XML request message as an HTTP POST to a request URL representing a DIVArchive API method. You specify text/xml as the *Content-Type* (`application/xml` is also accepted). The following example is a simple HTTP request issued to DIVArchive Web Services:

```
curl -v -X POST \
-H "Content-Type: application/xml" \
-u "4926d1cc-4f92-3b9c-d421-f958a60a832b:k89kcQWPbDES/tX10ged" \
--data \
'<p:cancelRequest xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>35c3f814-cbe8-421c-ac07-9209709c960c</p:sessionCode>
   <p:requestNumber>1678</p:requestNumber>
</p:cancelRequest>' \
 http://127.0.0.1:9443/diva/service/rest/2.2/DIVArchiveWS_REST/cancelRequest
```

REST Form Parameter Requests

The second way of invoking DIVArchive REST services involves supplying request parameters as form-encoded parameter and value pairs. The form-encoded parameters are sent as an HTTP POST request, using `application/x-www-form-urlencoded` as the *Content-Type*. The following is a sample cancel request that is invoked using form parameters:

```
curl -v -X POST \
-H "Content-Type: application/x-www-form-urlencoded" \
```

```
-u "4926d1cc-4f92-3b9c-d421-f958a60a832b:k89kcQWPbDES/tX10ged" \
--data sessionCode=35c3f814-cbe8-421c-ac07-9209709c960c&requestNumber=1678  \
 http://127.0.0.1:9443/diva/service/rest/2.2/DIVArchiveWS_REST/cancelRequest
```

The information passed in the data section includes the `requestNumber` and the `sessionCode`. Each parameter value must be URL-encoded so it can be correctly parsed by the services. The information returned by all REST services is represented in XML format.

Form parameters cannot be used to invoke all services. The `partialRestore()`, `associativeCopy()`, and `multipleRestoreObject()` requests cannot be invoked this way because of the nested structure of the information required on the call. You can invoke all other requests using form parameters. Some requests may require repeated parameter names in the request. For example, you can invoke an `archiveObject()` request that has multiple file arguments by including a `fileNamesList` parameter for each file to be passed.

Error Handling

The response to a REST request contains either an architected response message in XML format (with a successful HTTP error code), or a response with a non-successful HTTP error code. When an HTTP error is returned, the body may have additional error text that can assist in diagnosing the issue.

HTTP Errors

For fundamental errors such as authentication issues, XML formatting issues, and server issues, the DIVArchive REST Web Services return a non-successful HTTP error code (a successful code is between 200 and 300). The following list identifies of some HTTP error codes that may appear in a non-successful request.

HTTP Error 200

If 200 is returned, DIVArchive or DIVAnet has returned an architected response (see Architected Errors). The DIVArchive API status code in the response indicates the status of the call.

HTTP Error 400

The request was invalid, due to an apparent client-side error. In some cases, DWS will be able to identify the error, and also return an architected response in the message body.

HTTP Error 401

Authentication is required; credentials, session code, or both must be supplied.

HTTP Error 403

Authentication was provided, but the credentials supplied did not grant access to the requested service or resource.

HTTP Error 404

The URL is not found or is invalid.

HTTP Error 405

The HTTP method (for example, `GET`, `POST`) is invalid for the request.

HTTP Error 500

An error encountered on the server has caused the inability to satisfy the request. In some cases, the error can be identified and returned in an architected response in the message body.

Architected Errors

For errors that are actually generated by DIVArchive, the DIVArchive REST Web Services return a DIVArchive API status code, and the architected responses identified in this document. If the HTTP return code is between 200 and 300, that does not necessarily indicate that the request was issued successfully. *See Appendix A for a full list of DIVArchive API status codes*.

3

Testing the Installation

This chapter describes testing your installation, and incudes the following information:

Testing Overview

You can test DIVA Enterprise Connect using standard SOAP or REST test clients that can consume WSDL (Web Services Description Language) documents, WADL (Web Application Description Language) documents, or both. The DIVArchive Web Services also contains a set of scripts that make testing the DIVArchive Web Services easier.

Using Automated Testing Tools

Many open source and commercial tools exist for testing Web Services. Many of them can parse a WSDL or WADL document to enable Web Services testing. You must supply the following URL of the WSDL or WADL to use these tools:

```
http://127.0.0.1:9443/diva/service/soap/2.2/DIVArchiveWS_SOAP?WSDL

http://127.0.0.1:9443/diva/service/rest/2.2/DIVArchiveWS_REST/application.wadl
```

Substitute the actual network address of the DIVArchive Web Services platform in the URL. You may need to provide a user name and password to access the WSDL, or WADL, document from.

Using the `wsTest` Scripts

The DIVA Enterprise Connect release includes Linux shell scripts that can be used for testing. If you have access to the DIVA Enterprise Connect release, these scripts can make testing DIVArchive Web Services easier.

The wsTest scripts use the standard cURL utility to invoke the DIVArchive Web Services. The scripts automatically create a session for each call by default. This simplifies the testing process for the Web Service calls. The wsTest scripts are located in the $DIVAS_HOME/scripts/wsTest directory.

You use the following syntax to execute the scripts:

```
./wsTestRest.sh {baseUrl} {version} {templateCommand}
./wsTestSoap.sh {baseUrl} {version} {templateCommand}
```

The following is a list of the parameters used with the scripts:

baseUrl
This parameter identifies the base URL of the DIVArchive Web Service (for example, http://127.0.0.1:9443/diva/service).

version
This parameter identifies the protocol version level of the DIVArchive Web Service API (the current protocol release is the string v2.2). *This is not the software release level of the DIVA Enterprise Connect*.

templateCommand
This parameter identifies the Web Services command to execute. The parameters for the command are looked up in the template file, which contains a section for each call. The templates are located in the soapTemplate.sh script for SOAP, and in the restTemplate.sh script for REST. Each template depicts a sample call with all parameters populated with test data. You must edit these scripts to supply the desired parameters for each command.

--nosess
This is an optional parameter. Normally, the scripts will attempt to obtain a Web Services session ID before executing each command. You can use this argument to supply your own session ID in the template.

Example Call

The following is an example execution of the wsTest scripts. The getRequestInfo() call is being tested in the example. In this example, a new session ID will be fetched if needed. The parameters for this call are retrieved from the restTemplate.sh file.

```
./wsTestRest.sh http://127.0.0.1:9443/diva/service v2.2 getRequestInfo
```

Creating Sessions Automatically

Caution: You must only use the automatic session creation feature for testing, and not in production.

The scripts are configured to automatically create the required session ID to run each service. The scripts generate the new session ID, and then pass it in the chosen web service request. The wsTest scripts save the new session ID in a file, so it can be reused on the next call. Sessions (by default) last for 30 minutes, and then they expire.

If you do not want the wsTest scripts to automatically create the session ID, you must include the --nosess parameter in the command line when executing the test script. You can assign a fixed value for the *SESS_ID* variable in the template file. If you call

registerClient() yourself, and populate your own session code, the code will eventually expire, and you will have to repeat the process.

Editing the Templates

If you view the restTemplate.sh or soapTemplate.sh shell scripts, you will see a shell variable representing each Web Service call, and XML parameters. You can edit the XML to customize the arguments to the Web Service. Some parameter values in the template are derived from a shell variable. For example, the following is a section in the restTemplate.sh file that defines the parameters for the getObjectInfo() request:

```
getObjectInfo='
<p:getObjectInfo xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>'$SESS_ID'</p:sessionCode>
   <p:objectName>'$OBJ_NAME'</p:objectName>
   <p:objectCategory>'$OBJ_CATEGORY'</p:objectCategory>
</p:getObjectInfo>'
```

The $OBJ_NAME variable is defined at the top of the template script. The variables let you easily customize parameters for many commands. You can choose to populate the variable at the top of the template, or remove the variable and hard code the objectName directly in the getObjectInfo() command.

Authentication

Because basic authentication for Web Services is turned on by default, you must populate the *$CLIENT_ID* and *$AUTH_KEY* variables at the top of restTemplate.sh, or soapTemplate.sh if you are using SOAP. You must use values that were created by executing the DIVA Enterprise Connect Key Generator, or preset values assigned to you by Oracle Support.

Calling the cURL Tool Directly

The cURL command line tool enables transferring data to, or from, a server using one of the supported protocols. The cURL utility supports HTTP transfers and basic authentication. The wsTest scripts use cURL in the implementation, but you can call cURL directly as needed. The cURL tool is a standard utility in Oracle Linux 7 and later. *See Chapter 2 for examples of calls using cURL.*

4

Digital Asset Requests

This chapter details the DIVA Enterprise Connect services that process archived assets, which include the following services:

- Archive: `archiveObject()`
- Associative Copy: `associativeCopy()`
- Copy Object and Copy To Group: `copy()`, `copyToGroup()`
- Copy To New: `copyToNewObject()`
- Delete Object: `deleteObject()`
- Delete Instance: `deleteInstance()`
- Restore Object: `restoreObject()`
- Restore Instance: `restoreInstance()`
- Multiple Restore Object: `multipleRestoreObject()`
- Oracle DIVArchive Partial File Restore: `partialRestore()`
- Transcode Archived Object: `transcodeArchive()`
- Transfer Files: `transferFiles()`

Overview

The following Web Services allow digital assets to be archived into the digital archive, restored to Source/Destinations, copied, or deleted. These commands typically run for a long time. When a call is successful, these services return a request ID (similar to a job ID). The request ID can then be passed as an argument to monitor the request as it is executed in the system.

Archive: `archiveObject()`

This request enables digital assets to be transferred from a Source/Destination (for example, FTP or CIFS file system) to DIVArchive, resulting in a new archived object. If the request is created successfully, the call returns a request ID. You can monitor the progress of the call throughout its lifecycle using the `getRequestInfo()` call. This request is available in DIVArchive and DIVAnet. DIVAnet will archive to the local site by default.

Input Parameters

The following is a list of `archiveObject()` request input parameters:

Table 4–1 archiveObject() Request Input Parameters

Parameter	Description	Data Type and Defaults
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object.	String (1 - 96 characters) This field is required.
source	This parameter identifies the DIVArchive Source/Destination name (representing a server, or disk, such as FTP or CIFS) from where the files are ingested.	String (1 - 96 characters) This field is required.
mediaName	This parameter identifies the DIVArchive media name, and refers to a collection of object instances stored on the same, or similar, disk or tape media. You can also specify a DIVArchive Storage Plan here. In this case, DIVArchive performs processing to determine which media to use for the archive.	String (1 - 96 characters) This field is required.
filesPathRoot	This parameter identifies the top-level subdirectory within the Source/Destination from where files should be archived.	String (0 - 4000 characters) This can be empty.
fileNamesList	This parameter identifies a list of files (or wildcard patterns) to be archived. Paths for the files are relative to the specified filePathRoot. The <fileNamesList> tag delimits each file, and the tag is repeated for each file in the list. For example: <fileNamesList>misc/*</fileNamesList> <fileNamesList>t.mov</fileNamesList> If the -gcinfilelist option is passed, a genuine checksum value separated by a colon, may be supplied after the file name. For example, t.mov:a6f62b73f5a9bf380d32f062f2d71cbc.	String (1 - 4000 characters) This tag can appear one or more times.

Table 4–1 (Cont.) archiveObject() Request Input Parameters

Parameter	Description	Data Type and Defaults
qualityOfService	This parameter controls how requests are cached during request processing. The possible integer values are as follows: 0: *Default* - Use the default configured in DIVArchive. 1: *Cache and Direct* - Use cache first, then direct if cache is unavailable. 2: *Cache Only* - This is a two stage transfer to tape. 3: *Direct and Cache* - Use direct first, then cache if direct is unavailable. 4: *Direct Only* - This is direct to disk or tape. 5: *Nearline and Direct* - Use Nearline first, then direct if Nearline is unavailable. 6: *Nearline Only* - Create a disk instance if media is tape.	Integer (0 - 6) This can be nil. The *Default* is used when this value is nil.
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.
comments	This parameter is an optional string containing information describing the object.	String (0 - 4000 characters) This can be empty.
archiveOptions	This parameter identifies additional options for the archive command. These options are typically prefixed with a dash (similar to command line options). Some options can override parameters for the Source/Destination. Some typical archive options include: -r: Recursively archive subdirectories. -gcinfilelist {gctype}: Specify the type of genuine checksum. The default type is MD5.	String (0 - 768 characters) This can be empty.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–2 archiveObject() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	

DIVArchive Status Codes

The following are typical DIVArchive archiveObject() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1016: Object Already Exists
An object with the name provided already exists in the DIVArchive or DIVAnet system.

1017: Group Does Not Exist
The tape group provided does not exist in the target DIVArchive system.

1018: Source/Destination Does Not Exist
The Source/Destination provided does not exist in the target DIVArchive system.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST archiveObject() request:

```
<p:archiveObject xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:objectCategory>PROD</p:objectCategory>
   <p:source>ftp_001</p:source>
   <p:mediaName>Post4LTO7</p:mediaName>
   <p:filesPathRoot></p:filesPathRoot>
   <!--1 or more repetitions:-->
   <p:fileNamesList>preprod_jul.mov</p:fileNamesList>
   <p:qualityOfService>1</p:qualityOfService>
   <p:priorityLevel>50</p:priorityLevel>
   <p:comments>This object was archived through a DIVA WS REST call</p:comments>
   <p:archiveOptions></p:archiveOptions>
</p:archiveObject>
```

REST Response Example

The following is an example of a REST response for an archiveObject() request:

```
<p:archiveObjectResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:archiveObjectResponse>
```

Associative Copy: associativeCopy()

This request enables storing a list of archived objects together on the same tape media. The request completes when all objects have been copied to the target tape. If the request is created successfully, the call returns a request ID. You can monitor the progress of the call throughout its lifecycle using the getRequestInfo() call. This request is available in DIVArchive, but not in DIVAnet (MultiDIVA Mode).

Input Parameters

The following is as list of associativeCopy() request input parameters:

Table 4–3 associativeCopy() Request Input Parameters

Section	Parameter	Description	Data Type and Defaults
	sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectsInfo		This section identifies a list of objects to be stored together. Each entry is a separate object, identified by an object name and category pair. The following is an example of two objects to be stored together: <objectsInfo> <objectName>OBJ1</objectName> <objectCategory>Air3</objectCategory> </objectsInfo> <objectsInfo> <objectName>OBJ2</objectName> <objectCategory>Air3</objectCategory> </objectsInfo>	Can appear one or more times.
objectsInfo	objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
objectsInfo	objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object. The request will fail if this field is empty, and more than one object in DIVArchive matches the objectName.	String (1 - 96 characters) This field is can be empty.
	groupName	This parameter identifies the DIVArchive tape group name. This refers to a collection of object instances stored on the same, or similar, tape media.	String (1 - 96 characters) This field is required.

Table 4–3 (Cont.) associativeCopy() Request Input Parameters

Section	Parameter	Description	Data Type and Defaults
	priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–4 associativeCopy() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	

DIVArchive Status Codes

The following are typical DIVArchive associativeCopy() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in DIVArchive.

1017: Group Does Not Exist
The tape group provided does not exist in the target DIVArchive system.

1023: Object Offline
There are no available instances of the object. For example, this could possibly occur because a required tape was ejected from a tape library.

1031: Object In Use
One or more objects are currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST associativeCopy() request:

```
<p:associativeCopy xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd"
xmlns:p1="http://model.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectsInfo>
      <p1:objectName>PREPROD_July_Edits4</p1:objectName>
      <p1:objectCategory>PROD</p1:objectCategory>
   </p:objectsInfo>
   <p:objectsInfo>
      <p1:objectName>PREPROD_July_Edits5</p1:objectName>
      <p1:objectCategory>PROD</p1:objectCategory>
   </p:objectsInfo>
   <p:groupName>Post4LTO7</p:groupName>
   <p:priorityLevel>50</p:priorityLevel>
</p:associativeCopy>
```

REST Response Example

The following is an example of a REST response for an associativeCopy() request:

```
<p:associativeCopyResponse
xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:associativeCopyResponse>
```

Copy Object and Copy To Group: copy(), copyToGroup()

This request creates a copy of an archived asset. This results in a *new instance* of the object within DIVArchive, *not a new object*. If you use DIVAnet, you can use this command to copy an object from one site to another. The target media of the new instance is passed. The call returns a request ID if the request is created successfully. You can monitor the progress of the call throughout its lifecycle using the getRequestInfo() call. This request is available in both DIVArchive and DIVAnet.

Input Parameters

The following is a list of copy() and copyToGroup() request input parameters:

Table 4–5 copy() and copyToGroup() Request Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with the objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
categoryName	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object. If this field is empty, and more than one object in DIVArchive matches the objectName, the request will fail.	String (0 - 96 characters) You can leave this field empty.
instanceID	Specifying a value for this parameter enables the caller to select a specific object instance to copy from.	Integer (0 - 100000). If you use nil or -1 for this value, DIVArchive will choose the best object instance to copy from.
mediaName	This parameter identifies the DIVArchive media name. This refers to a collection of object instances stored on the same, or similar, disk or tape media. In DIVAnet, you can prefix a DIVArchive site name to the media using an underscore to separates the two. This allows objects to be copied to a particular site.	String (1 - 96 characters) This field is required.
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–6 copy() or copyToGroup() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	

DIVArchive Status Codes

The following are typical DIVArchive copy() and copyToGroup() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1017: Group Does Not Exist
The tape group provided does not exist in the target DIVArchive system.

1023: Object Offline
There are no available instances of the object. For example, this could possibly occur because a required tape was ejected from a tape library.

1027: Instance Does Not Exist
The specified object instance does not exist in DIVArchive or DIVAnet.

1028: Instance Offline
One or more object instances are currently offline and cannot be processed.

1031: Object In Use
One or more objects are currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST copy() and copyToGroup() request:

```
<p:copy xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:categoryName>PROD</p:categoryName>
   <p:instanceID>-1</p:instanceID>
   <p:priorityLevel>50</p:priorityLevel>
   <p:mediaName>Post4LTO7</p:mediaName>
</p:copy>
```

REST Response Example

The following is an example of a REST response for a copy() and copyToGroup() request:

```
<p:copyResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:copyResponse>
```

Copy To New: copyToNewObject()

This request creates a copy of an archived object. This results in a new object within DIVArchive, not just a new object instance. You provide the target name, target category, and target media for the new object. The call returns a request ID if it the request is created successfully. You can monitor the progress of the call throughout its lifecycle using the getRequestInfo() call. This request is available in DIVArchive, but not in DIVAnet (MultiDIVA mode).

Input Parameters

The following is a list of copyToNewObject() request input parameters:

Table 4–7 copyToNew() Request Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object. If this field is empty, and more than one object in DIVArchive matches the objectName, the request will fail.	String (0 - 96 characters) You can leave this field empty.
objectMedia	Specifying a value for this parameter enables the caller to select which media to perform the copy from. If this value and objectInstanceID are empty, DIVArchive chooses the best object instance to copy from. *This value cannot be combined with objectInstanceID.*	String (0 - 96 characters) You can leave this field empty.

Table 4–7 (Cont.) copyToNew() Request Input Parameters

Parameter	Description	Data Type and Default
objectInstanceID	Specifying a value for this parameter enables the caller to select a specific object instance to copy from. If this value and objectMedia are empty, DIVArchive chooses the best object instance to copy from. *This value cannot be combined with objectMedia.*	Integer (0 - 100000). Entering -1 indicates that this value is not specified.
newObjectName	This parameter identifies the target object name to be created.	String (1 - 192 characters) This field is required.
newObjectCategory	This parameter identifies the target object category to be created.	String (1 - 96 characters) This field is required.
newObjectInstanceMedia	This parameter identifies the DIVArchive media name (either a tape group or array name). This refers to a collection of object instances stored on the same, or similar, disk or tape media. The target object is stored on the identified media.	String (1 - 96 characters) This field is required.
comments	This parameter is an optional string containing information describing the object.	String (0 - 4000 characters) You can leave this field empty.
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–8 copyToNew() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	

DIVArchive Status Codes

The following are typical DIVArchive copyToNew() request status codes. *See Appendix A for all DIVArchive Status Codes.*

1000: Success

The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1017: Group Does Not Exist
The tape group provided does not exist in the target DIVArchive system.

1023: Object Offline
There are no available instances of the object. For example, this could possibly occur because a required tape was ejected from a tape library.

1027: Instance Does Not Exist
The specified object instance does not exist in DIVArchive or DIVAnet.

1028: Instance Offline
One or more object instances are currently offline and cannot be processed.

1031: Object In Use
One or more objects are currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST `copyToNewObject()` request:

```
<p:copyToNewObject xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:objectCategory>PROD</p:objectCategory>
   <p:objectMedia>Post4LTO7</p:objectMedia>
   <p:objectInstanceID></p:objectInstanceID>
   <p:newObjectName>PREPROD_July_Edits4</p:newObjectName>
   <p:newObjectCategory>PROD</p:newObjectCategory>
   <p:newObjectInstanceMedia>Post4LTO7</p:newObjectInstanceMedia>
   <p:comments></p:comments>
   <p:priorityLevel>50</p:priorityLevel>
</p:copyToNewObject>
```

REST Response Example

The following is an example of a REST response for a `copyToNewObject()` request:

```
<p:copyToNewResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
```

```
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:copyToNewResponse>
```

Delete Object: deleteObject()

The request deletes an archived object from DIVArchive. If you use DIVAnet, you can use this command to delete objects from all sites. The call returns a request ID if the request is created successfully. You can monitor the progress of the call throughout its lifecycle using the getRequestInfo() call. This request is available in both DIVArchive and DIVAnet.

Input Parameters

The following is a list of deleteObject() request input parameters:

Table 4–9 deleteObject() Request Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object. If this field is empty, and more than one object in DIVArchive matches the objectName, the request will fail.	String (0 - 96 characters) You can leave this field empty.
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–10 deleteObject() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	

DIVArchive Status Codes

The following are typical DIVArchive deleteObject() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1017: Group Does Not Exist
The tape group provided does not exist in the target DIVArchive system.

1028: Instance Offline
One or more object instances are currently offline and cannot be processed.

1031: Object In Use
One or more objects are currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST deleteObject() request:

```
<p:deleteObject xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:objectCategory>PROD</p:objectCategory>
</p:deleteObject>
```

REST Response Example

The following is an example of a REST response for a deleteObject() request:

```
<p:deleteObjectResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:deleteObjectResponse>
```

Delete Instance: deleteInstance()

The request deletes an archived object, or a single object instance from DIVArchive. If you use DIVAnet, you can delete an asset from all sites, a single site, or a single object instance on a site. You can provide an instance ID or media in addition to object name and category to select a particular object instance. The call returns a request ID if the request is created successfully. You can monitor the progress of the call throughout its lifecycle using the getRequestInfo() call. This request is available in both DIVArchive and DIVAnet.

Input Parameters

The following is a list of deleteInstance() request input parameters:

Table 4–11 deleteInstance() Request Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
categoryName	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object. If this field is empty, and more than one object in DIVArchive matches the objectName, the request will fail.	String (0 - 96 characters) You can leave this field empty.
mediaName	This parameter identifies the DIVArchive media name. This refers to a collection of object instances stored on the same, or similar, disk or tape media. In DIVAnet, you can prefix a DIVArchive site name to the media using an underscore to separates the two. This allows objects to be deleted from a particular site. *You cannot specify both this value and the instanceID at the same time.*	String (1 - 96 characters) This field is required.
instanceID	Specifying a value for this parameter enables the caller to select a specific object instance to delete. *You cannot specify both this value and the mediaName at the same time.*	Integer (0 - 100000). Entering -1 indicates that the parameter is not specified.
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–12 deleteInstance() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	

DIVArchive Status Codes

The following are typical DIVArchive deleteInstance() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1017: Group Does Not Exist
The tape group provided does not exist in the target DIVArchive system.

1023: Object Offline
There are no available instances of the object. For example, this could possibly occur because a required tape was ejected from a tape library.

1027: Instance Does Not Exist
The specified object instance does not exist in DIVArchive or DIVAnet.

1028: Instance Offline
One or more object instances are currently offline and cannot be processed.

1031: Object In Use
One or more objects are currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST deleteInstance() request:

```
<p:deleteInstance xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:categoryName>PROD</p: categoryName >
   <p:mediaName>Post4LTO7</p:mediaName>
   <p:priorityLevel>50</p:priorityLevel>
</p:deleteInstance>
```

REST Response Example

The following is an example of a REST response for a deleteInstance() request:

```
<p:deleteInstanceResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:deleteInstanceResponse>
```

Restore Object: restoreObject()

This request enables restoring an object within DIVArchive to a Source/Destination (such as FTP or CIFS). This request restores all files and folders associated with the archived object. Optionally, you can select a specific DIVArchive object instance to read, and multiple Source/Destinations using the restoreInstance() call (see Restore Instance: restoreInstance() for information). This call returns a request ID if the request is created successfully. You can monitor the progress of the call throughout its lifecycle using the getRequestInfo() call. This request is available in both DIVArchive and DIVAnet.

Input Parameters

The following is a list of restoreObject() request input parameters:

Table 4–13 restoreObject() Request Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.

Table 4–13 (Cont.) restoreObject() Request Input Parameters

Parameter	Description	Data Type and Default
objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object. If this field is empty, and more than one object in DIVArchive matches the objectName, the request will fail.	String (0 - 96 characters) You can leave this field empty.
destination	This parameter identifies a DIVArchive Source/Destination name (representing a server or disk, such as FTP or CIFS). The archived object files are restored to this destination.	String (1 - 96 characters) This field is required
filesPathRoot	This parameter identifies the top-level subdirectory within the Source/Destination where files should be restored. If this value is not specified, the default stored in the archivedObject parameter is used (if specified).	String (0 - 4000 characters) This can be empty.
qualityOfService	This parameter controls how requests are cached during request processing. The possible integer values are as follows: 0: *Default* - Use the default configured in DIVArchive. 1: *Cache and Direct* - Use cache first, then direct if cache is unavailable. 2: *Cache Only* - This is a two stage transfer to tape. 3: *Direct and Cache* - Use direct first, then cache if direct is unavailable. 4: *Direct Only* - This is direct to disk or tape. 5: *Nearline and Direct* - Use Nearline first, then direct if Nearline is unavailable. 6: *Nearline Only* - Create a disk instance if media is tape.	Integer (0 - 6) This can be nil. The *Default* is used when this value is nil.
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Table 4–13 (Cont.) restoreObject() Request Input Parameters

Parameter	Description	Data Type and Default

restoreOptions	This parameter identifies additional restore command options. These options typically are prefixed with a dash (similar to command line options). Some options can override parameters for the Source/Destination. Some typical restore options are a follows:
	Rename certain files:
	`-rest_renaming <renamerule>`
	Use certain transcoders:
	`-tr_names <transcoder_list>`
	Restored format:
	`-tr_restore_format <format>`
	Generate a metadata file on restore:
	`-rm`
		String (0 - 768 characters)
		You can leave this field empty.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–14 restoreObject() Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 -1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	

DIVArchive Status Codes

The following are typical DIVArchive restoreObject() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1018: Source/Destination Does Not Exist
The Source/Destination name provided does not exist in DIVArchive.

1023: Object Offline
There are no available instances of the object. For example, this could possibly occur because a required tape was ejected from a tape library.

1028: Instance Offline
One or more object instances are currently offline and cannot be processed.

1031: Object In Use
One or more objects are currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST `restoreObject()` request:

```
<p:restoreObject xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:objectCategory>PROD</p:objectCategory>
   <p:destination>ftp_001</p:destination>
   <p:filesPathRoot></p:filesPathRoot>
   <p:qualityOfService>0</p:qualityOfService>
   <p:priorityLevel></p:priorityLevel>
   <p:restoreOptions></p:restoreOptions>
</p:restoreObject>
```

REST Response Example

The following is an example of a REST response for a `restoreObject()` request:

```
<p:restoreObjectResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:restoreObjectResponse>
```

Restore Instance: `restoreInstance()`

This request transfers an archived object from DIVArchive to a specified Source/Destination (such as FTP or CIFS). You can specify an `instanceID` to select a specific DIVArchive object instance to read. These calls return a request ID if the request is created successfully. You can monitor the progress of the call throughout its lifecycle using the `getRequestInfo()` call. This request is available in both DIVArchive and DIVAnet.

Input Parameters

The following is a list of restoreInstance() request input parameters:

Table 4–15 restoreInstance() Request Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object. If this field is empty, and more than one object in DIVArchive matches the objectName, the request will fail.	String (0 - 96 characters) You can leave this field empty.
instanceID	Specifying a value for this parameter enables the caller to select a specific object instance to delete. If DIVAnet is used, pass the DIVAnet object instance ID, which is unique among all sites.	Integer (0 - 100000). This field is required.
destination	This parameter identifies a DIVArchive Source/Destination name (representing a server or disk, such as FTP or CIFS). The archived object files are restored to this destination.	String (1 - 96 characters) This field is required
filesPathRoot	This parameter identifies the top-level subdirectory within the Source/Destination where files should be restored. Setting this value overrides the default in the object. If this value is not specified, the default stored in the archivedObject parameter is used (if specified).	String (0 - 4000 characters) This can be empty.
qualityOfService	This parameter controls how requests are cached during request processing. The possible integer values are as follows: 0: *Default* - Use the default configured in DIVArchive. 1: *Cache and Direct* - Use cache first, then direct if cache is unavailable. 2: *Cache Only* - This is a two stage transfer to tape. 3: *Direct and Cache* - Use direct first, then cache if direct is unavailable. 4: *Direct Only* - This is direct to disk or tape. 5: *Nearline and Direct* - Use Nearline first, then direct if Nearline is unavailable. 6: *Nearline Only* - Create a disk instance if media is tape.	Integer (0 - 6) This can be nil. The *Default* is used when this value is nil.

Table 4–15 (Cont.) restoreInstance() Request Input Parameters

Parameter	Description	Data Type and Default
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.
restoreOptions	This parameter identifies additional restore command options. These options typically are prefixed with a dash (similar to command line options). Some options can override parameters for the Source/Destination. Some typical restore options are a follows: Rename certain files: -rest_renaming <renamerule> Use certain transcoders: -tr_names <transcoder_list> Restored format: -tr_restore_format <format> Generate a metadata file on restore: -rm	String (0 - 768 characters) You can leave this field empty.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–16 restoreInstance() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	Integer (1 - 10,000,000)

DIVArchive Status Codes

The following are typical DIVArchive restoreInstance() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1018: Source/Destination Does Not Exist
The Source/Destination name provided does not exist in DIVArchive.

1023: Object Offline
There are no available instances of the object. For example, this could possibly occur because a required tape was ejected from a tape library.

1027: Instance Does Not Exist
The specified object instance does not exist in DIVArchive or DIVAnet.

1028: Instance Offline
One or more object instances are currently offline and cannot be processed.

1031: Object In Use
One or more objects are currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST `restoreInstance()` request:

```
<p:restoreInstance xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:objectCategory>PROD</p:objectCategory>
   <p:destination>ftp_001</p:destination>
   <p:instanceID>-1</p:instanceID>
   <p:filesPathRoot></p:filesPathRoot>
   <p:qualityOfService>0</p:qualityOfService>
   <p:priorityLevel></p:priorityLevel>
   <p:restoreOptions></p:restoreOptions>
</p:restoreInstance>
```

REST Response Example

The following is an example of a REST response for a `restoreInstance()` request:

```
<p:restoreInstanceResponse
xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:restoreInstanceResponse>
```

Multiple Restore Object: multipleRestoreObject()

This request transfers an archived object from DIVArchive to a specified Source/Destination (such as FTP or CIFS). This call allows the object to be transferred to multiple Source/Destinations. The call return a request ID if the request is created successfully. You can monitor the progress of the call throughout its lifecycle using the getRequestInfo() call. This command completes when the object has been transferred to all Source/Destinations (successfully or not). This request is available in both DIVArchive and DIVAnet. DIVAnet (MultiDIVA mode) will restore from exactly one site to fulfill the request.

Input Parameters

The following is a list of multipleRestoreObject() request input parameters:

Table 4–17 multipleRestoreObject() Request Input Parameters

Section	Parameter	Description	Data Type and Default
	sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
	objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
	objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object. If this field is empty, and more than one object in DIVArchive matches the objectName, the request will fail.	String (0 - 96 characters) You can leave this field empty.
destinations		This section identifies a list of one or more DIVArchive Source/Destination names and path roots for each destination. The following is an example of two Source/Destinations: <p:destinations> <p:destination>POST2</p:destination> <p:filePathRoot>content</p:filePathRoot> </p:destinations> <p:destinations> <p:destination>POST3</p:destination> <p:filePathRoot></p:filePathRoot> </p:destinations>	String (1 - 200 entries) This section is required.

Table 4–17 (Cont.) multipleRestoreObject() Request Input Parameters

Section	Parameter	Description	Data Type and Default
destinations	destination	This parameter identifies an item in the list of DIVArchive Source/Destination names. The object is restored to this destination and the others in the list.	String (1 - 96 characters) This field is required.
destinations	filesPathRoot	This parameter identifies the top-level within the Source/Destination where files should be restored. If you supply an empty filePathRoot, DIVArchive uses the original path root in the object.	String (0 - 4000 characters) This can be empty.
	qualityOfService	This parameter controls how requests are cached during request processing. The possible integer values are as follows: 0: *Default* - Use the default configured in DIVArchive. 1: *Cache and Direct* - Use cache first, then direct if cache is unavailable. 2: *Cache Only* - This is a two stage transfer to tape. 3: *Direct and Cache* - Use direct first, then cache if direct is unavailable. 4: *Direct Only* - This is direct to disk or tape. 5: *Nearline and Direct* - Use Nearline first, then direct if Nearline is unavailable. 6: *Nearline Only* - Create a disk instance if media is tape.	Integer (0 - 6) This can be nil. The *Default* is used when this value is nil.
	priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Table 4–17 (Cont.) multipleRestoreObject() Request Input Parameters

Section	Parameter	Description	Data Type and Default
	restoreOptions	This parameter identifies additional restore command options. These options typically are prefixed with a dash (similar to command line options). Some options can override parameters for the Source/Destination. Some typical restore options are a follows: Rename certain files: -rest_renaming <renamerule> Use certain transcoders: -tr_names <transcoder_list> Restored format: -tr_restore_format <format> Generate a metadata file on restore: -rm	String (0 - 768 characters) You can leave this field empty.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–18 multipleRestoreObject() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	Integer (1 - 10,000,000)

DIVArchive Status Codes

The following are typical DIVArchive multipleRestoreObject() request status codes. *See Appendix A for all DIVArchive Status Codes.*

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1018: Source/Destination Does Not Exist
The Source/Destination name provided does not exist in DIVArchive.

1023: Object Offline
There are no available instances of the object. For example, this could possibly occur because a required tape was ejected from a tape library.

1027: Instance Does Not Exist
The specified object instance does not exist in DIVArchive or DIVAnet.

1028: Instance Offline
One or more object instances are currently offline and cannot be processed.

1031: Object In Use
One or more objects are currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST `multipleRestoreObject()` request:

```
<p:multipleRestoreObject xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1="http://model.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:objectCategory>PROD</p:objectCategory>
   <p:destinations>
      <p1:destination>POST2</p1:destination>
      <p1:filePathRoot>content</p1:filePathRoot>
   </p:destinations>
   <p:destinations>
      <p1:destination>POST3</p1:destination>
      <p1:filePathRoot></p1:filePathRoot>
   </p:destinations>
   <p:qualityOfService>0</p:qualityOfService>
   <p:priorityLevel></p:priorityLevel>
   <p:restoreOptions></p:restoreOptions>
</p:multipleRestoreObject>
```

REST Response Example

The following is an example of a REST response for a `multipleRestoreObject()` request:

```
<p:multipleRestoreObjectResponse
xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
```

```
</p:multipleRestoreObjectResponse>
```

Oracle DIVArchive Partial File Restore: partialRestore()

This request enables restoring a portion of an object within DIVArchive to a Source/Destination (such as FTP or CIFS). The caller can define these portions by supplying byte offset ranges, timecode ranges, DPX frame ranges, or entire subdirectories and files. Optionally, the caller can select a specific DIVArchive object instance for the read. The call returns a request ID if the request is created successfully. You can monitor the progress of the call throughout its lifecycle using the getRequestInfo() call. This request is available in both DIVArchive and DIVAnet.

Major Partial File Restore Request Types

DIVArchive supports the following major partialRestore() request types:

Byte Offset

This type of Partial File Restore request extracts a range of bytes from an archived file into a target file. You can supply multiple ranges. You can determine the name of the target file. You can also extract from multiple source files. You cannot extract from multiple source files into one target file.

Timecode

This type of Partial File Restore request extracts a portion of a media clip using timecode start and end values. You can determine the name of the target file to place the resulting media segment. You specify a timecode range, and specify the target media format.

DPX Frame

This type of Partial File Restore request extracts a range of files based on file order (for example, extract all files from the fifth to the twentieth file in the archive). The archived files must be archived in a particular order.

Files and Folder

This type of Partial File Restore request extracts entire subdirectories and files of an archived object, using the original file and folder names that are part of the archive. Recursive folder extraction is supported.

Input Parameters

The following is a list of partialRestore() request input parameters:

Table 4–19 partialRestore() Request Input Parameters

Section	Parameter	Description	Data Type and Default
	sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
	objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.

Table 4–19 (Cont.) partialRestore() Request Input Parameters

Section	Parameter	Description	Data Type and Default
	objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object. If this field is empty, and more than one object in DIVArchive matches the objectName, the request will fail.	String (0 - 96 characters) You can leave this field empty.
	instanceID	Specifying a value for this parameter enables the caller to select a specific object instance to delete. If DIVAnet is used, pass the DIVAnet object instance ID, which is unique among all sites.	Integer (0 - 100000). If this value is nil or -1, it indicates that the parameter is not specified.
fileList		This section identifies a list of files and file offsets, or a list of subdirectories or files within the archived object to restore. The various sections within fileList are detailed. This section can occur multiple times in the request.	Maximum entries are configured in DIVArchive. This field is required.
	destination	This parameter identifies a DIVArchive Source/Destination name representing a server or disk (such as FTP or CIFS). The archived object files are restored to this destination.	String (1 - 96 characters) This field is required.
	filesPathRoot	This parameter identifies the top-level directory within the Source/Destination where files should be restored. If this value is not specified, the default stored in the archivedObject parameter is used (if specified).	String (0 - 4000 characters) This can be empty.
	qualityOfService	This parameter controls how requests are cached during request processing. The possible integer values are as follows: 0: *Default* - Use the default configured in DIVArchive. 1: *Cache and Direct* - Use cache first, then direct if available. 2: *Cache Only* - This is a two stage transfer to disk or tape. 3: *Direct and Cache* - Use direct first, then cache if direct is unavailable. 4: *Direct Only* - This is direct to disk or tape. *Partial File Restore requests do not support NearLine QOS*.	Integer (0 - 4) This can be nil. The *Default* is used when this value is nil.

Table 4–19 (Cont.) partialRestore() Request Input Parameters

Section	Parameter	Description	Data Type and Default
	priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

	restoreOptions	This parameter identifies additional restore command options. These options typically are prefixed with a dash (similar to command line options). Some options can override parameters for the Source/Destination. Some typical restore options are a follows:
		Rename certain files:
		`-rest_renaming <renamerule>`
		Use certain transcoders:
		`-tr_names <transcoder_list>`
		Restored format:
		`-tr_restore_format <format>`
		Generate a metadata file on restore:
		`-rm`
			String (0 - 768 characters)
			You can leave this field empty.

Table 4–19 (Cont.) partialRestore() Request Input Parameters

Section	Parameter	Description	Data Type and Default
	format	This parameter identifies a numeric code that indicates either the type of Partial File Restore to use (for example, file and folder partial file restore), or a particular media format. Selecting a media format implies a timecode based Partial File Restore. **Partial File Restore Type**: 0: *ByteOffset* - Extract by byte ranges 1: *ByteOffsetHeader* - this is deprecated 12: *File and Folder* - extract directories and files 13: *DPX Frame* - extract DPX file ranges **Partial File Restore Formats** (implies Timecode Partial File Restore): 2: *GXF* 3: *Seachange* - SAF format with index 4: *AVI Matrox* - .avi and .wav files 5: *MPEG2* 6: *MXF* 7: *Pinnacle* 8: *Omneon* 9: *Leitch* 10: *Quantel* - QCP format 11: *Autodetect* - use this if unsure of format	Integer (0 - 13) This field is required.

Specifying Partial File Restore Selections <fileList>

The following sections describe how to create and specify Partial File Restore selections for the major types of Partial File Restores (see Major Partial File Restore Request Types).

Byte Offset

You set the <format> parameter value to 0 in the request to extract byte ranges. The following table specifies the parameters you must provide in the <fileList> section:

Table 4–20 Byte Offset Partial File Restore <fileList> Parameters

Section	Parameter	Description	Data Type and Default
	destFile	This parameter identifies the file name to assign to the output of the Partial File Restore. Relative (or absolute) paths in the file name are not allowed.	String (1 - 4000) This field is required.

Table 4–20 (Cont.) Byte Offset Partial File Restore <fileList> Parameters

Section	Parameter	Description	Data Type and Default
offsetVector		This section identifies the byte ranges of the file to restore. The number -1 defaults to the total number of bytes in the file. *There is no enclosing tag for all of the offsets in this section*. <offsetVector> <byteBegin>0</byteBegin> <byteEnd>20</byteEnd> <posType>1</posType> </offsetVector> <offsetVector> <byteBegin>40</byteBegin> <byteEnd>-1</byteEnd> <posType>1</posType> </offsetVector>	
offsetVector	byteBegin	This parameter identifies the starting byte in the range. The first byte starts at zero.	Integer This can be -1, 0, or a positive integer.
offsetVector	byteEnd	This parameter identifies the ending byte in the range. The last byte is indicated with a -1 value.	Integer This can be -1, 0, or a positive integer.
offsetVector	posType	This parameter identifies the type of offset being utilized. Set this value to 1 for Byte Offset Partial File Restore.	Integer (1 - 2) This field is required.
	sourceFile	This parameter identifies the source file name within the archived object.	String (1 - 4000) This field is required.

Timecode

You set the <format> parameter in the request to the desired format (or use autodetect) to extract time regions from media files. The following table specifies the parameters you must provide in the <fileList> section:

Table 4–21 Timecode Partial File Restore <fileList> Parameters

Section	Parameter	Description	Data Type and Default
	destFile	This parameter identifies the file name to assign to the output of the Partial File Restore. Relative (or absolute) paths in the file name are not allowed. If the Partial File Restore results in multiple files being generated (for example, audio files), DIVArchive uses the destFile name to assign a name to the audio files.	String (1 - 4000) This field is required.

Table 4–21 (Cont.) Timecode Partial File Restore <fileList> Parameters

Section	Parameter	Description	Data Type and Default
offsetVector		This section identifies the timecode range to restore. The number -1 defaults to the total number of bytes in the file. *There is no enclosing tag for all of the offsets in this section*. <offsetVector> <timeCodeBegin>00:00:00:00</timeCodeBegin> <timeCodeEnd>00:00:10:00 </timeCodeEnd> <posType>2</posType> </offsetVector> <offsetVector> <timeCodeBegin>00:00:20:00</timeCodeBegin> <timeCodeEnd>99:99:99:99</timeCodeEnd> <posType>2</posType> </offsetVector>	
offsetVector	timeCodeBegin	This parameter identifies the SMPTE timecode value marking the start of the portion to extract, in the format hour:minute:second:frame. For example, 00:00:04:00 to 00:10:04:00 would denote a ten minute segment starting four seconds in and ending at ten minutes and four seconds.	String (hour:min:sec:frame) This field is required.
offsetVector	timeCodeEnd	This parameter identifies the SMPTE timecode value marking the end of the portion to extract, in the format hour:minute:second:frame.	String (hour:min:sec:frame) This field is required.
offsetVector	posType	This parameter identifies the type of offset being utilized. Set this value to 2 for Timecode Partial File Restore.	Integer (1 - 2) This field is required.
	sourceFile	This parameter identifies the source file name within the archived object.	String (1 - 4000) This field is required.

Files and Folders

You set the <format> parameter value in the request to 12 (Files and Folder Partial Restore) to extract entire subdirectories and files. The files are restored with the relative path and file names specified in the archive. For example, a file archived as misc/12-2012/movie.avi would be partially restored to a misc/12-2012 subdirectory with the name movie.avi. *There is no renaming option for Files and Folders Partial File Restore*.

The following table specifies the parameters you must provide in the <fileList> section:

Table 4–22 Files and Folders Partial Files Restore <fileList> Parameters

Section	Parameter	Description	Data Type and Default
fileFolder		This section identifies the list of archived files and folders to restore. This section occurs only once within the <fileList> section. To select multiple file and folders, wrap the fileFolder section in multiple <fileList> tags, as shown: <fileList> <fileFolder> <fileFolder>media/pt3</fileFolder> <option>-r</option> </fileFolder> </fileList> <fileList> <fileFolder> <fileFolder>media/f.txt</fileFolder> <option></option> </fileFolder> </fileList>	Maximum
fileFolder	fileFolder	This parameter identifies the name of the file or folder to restore, including the path (if relevant). When you specify a folder, that folder, and all files in the archive directly within that folder, are restored. To recursively restore all files and folders within the specified folder, use the -r option.	String (1 - 4000) This field is required.
fileFolder	option	This parameter identifies an option for the restore. Using -r for a directory recursively restores all files and folders within the directory.	String (0 - 768) You can leave this field empty.
	sourceFile, destFile	You should not include these two parameters for File and Folder Partial File Restore requests.	This is optional.

DPX Frame

This type of Partial File Restore is specifically designed for extracting ranges of DPX (Digital Picture Exchange) files. Each DPX file corresponds to a frame in a media clip. DIVArchive allows a range of files to be extracted based on file order. For example, you can extract all files from the fifth to the twentieth file. Each file can represent a frame number in a video or media clip. You set the <format> parameter value in the request to 13 (DPX Partial File Restore) to use DPX Partial File Restore.

To be considered a frame, each file in a DPX object must have a .tiff or .dpx extension. DIVArchive bases the frame number on the order the files were archived. If the first file archived is named MDCLIP_000002.dpx, the assigned frame number is 1 (regardless of the file name). The same rule applies if directories appear in the archive. Directories are often sorted alphanumerically at Source/Destinations. However, you can use the -file_order option when archiving DPX objects to ensure proper ordering.

The following table specifies the parameters you must provide in the <fileList> section:

Table 4–23 DPX Frame Partial File Restore <fileList> Parameters

Section	Parameter	Description	Data Type and Default
range		This parameter identifies the frame ranges to restore. This section occurs only once within the <fileList> section. To select multiple frame ranges, wrap the range section in multiple <fileList> tags, as shown: <fileList> <range> <startRange>1</startRange> <endRange>10</endRange> </range> </fileList> <fileList> <range> <startRange>42</startRange> <endRange>-1</endRange> </range> </fileList>	Maximum
range	startRange	This parameter identifies the file order number of the first frame (file) to extract. The first frame starts at 1 (0 also refers to frame 1).	Integer (-1, 0, or any positive integer) This field is required.
range	endRange	This parameter identifies the file order number of the last frame (file) to extract. You use -1 to indicate the last frame in the archive.	Integer (-1, 0, or any positive integer) This field is required.
	sourceFile, destFile	You should not include these two parameters for DPX Frame Partial File Restore requests.	This is optional.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–24 partialRestore() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	Integer (1 - 10,000,000)

DIVArchive Status Codes

The following are typical DIVArchive partialRestore() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1018: Source/Destination Does Not Exist
The Source/Destination name provided does not exist in DIVArchive.

1023: Object Offline
There are no available instances of the object. For example, this could possibly occur because a required tape was ejected from a tape library.

1027: Instance Does Not Exist
The specified object instance does not exist in DIVArchive or DIVAnet.

1028: Instance Offline
One or more object instances are currently offline and cannot be processed.

1031: Object In Use
One or more objects are currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST Timecode `partialRestore()` request:

```
<p:partialRestore xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1="http://model.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:objectCategory>PROD</p:objectCategory>
   <p:instanceID></p:instanceID>
   <p:fileList>
      <p1:destFile>mediaIntro01.mxf</p1:destFile>
      <p1:offsetVector>
         <p1:timeCodeBegin>00:00:00:00</p1:timeCodeBegin>
         <p1:timeCodeEnd>00:00:10:00</p1:timeCodeEnd>
         <p1:posType>2</p1:posType>
      </p1:offsetVector>
      <p1:sourceFile>mediaFile01.xml</p1:sourceFile>
   </p:fileList>
   <p:destination>POST2</p:destination>
   <p:filesPathRoot>partial/out</p:filePathRoot>
   <p:qualityOfService>0</p:qualityOfService>
```

```
   <p:priorityLevel>50</p:priorityLevel>
   <p:restoreOptions></p:restoreOptions>
   <p:format>11</p:format>
</p:partialRestore>
```

REST Response Example

The following is an example of a REST response for a Timecode partialRestore() request:

```
<p:partialRestoreResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:partialRestoreResponse>
```

Transcode Archived Object: transcodeArchive()

This request transcodes an existing archived object to a different media format, and then archives the resulting files as a new object in DIVArchive. You can think of this command as a *CopyToNew* request with *transcode*. The call returns a request ID if the request is created successfully. You can monitor the progress of the call throughout its lifecycle using the getRequestInfo() call. This request is available in DIVArchive, but not DIVAnet (MultiDIVA mode).

Input Parameters

The following is a list of transcodeArchive() request input parameters:

Table 4–25 transcodeArchive() Request Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
parentObjectName	This parameter identifies the name of the object. This parameter, along with parentObjectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
parentObjectCategory	This parameter identifies the name of the object category. This parameter, along with parentObjectName, creates the full formal name of a DIVArchive object. If this field is empty, and more than one object in DIVArchive matches the objectName, the request will fail.	String (1 - 96 characters) This field is required.
instance	This parameter identifies the instance ID in DIVArchive, and enables the caller to select the particular object instance to use for the transcode.	Integer (0 - 1000). If this value is nil or -1, DIVArchive selects the best instance to use.

Table 4–25 (Cont.) transcodeArchive() Request Input Parameters

Parameter	Description	Data Type and Default
objectName	This parameter identifies the name of the new object to create. This object will contain the transcoded files.	String (1 - 192 characters) This field is required.
objectCategory	This parameter identifies the category of the new object to create. This object will contain the transcoded files.	String (1-96 characters) This field is required.
mediaName	This parameter identifies a DIVArchive media name, and enables the caller to select a particular media for the new object containing the transcoded files. You can also specify a DIVArchive Storage Plan. In this case, DIVArchive will perform SPM processing to determine which media is used for the archive.	String (0 - 96 characters) This field is required.
comments	This parameter identifies an optional string containing information describing the object.	String (0 - 4000 characters) You can leave this field empty.

archiveOptions	This parameter identifies additional command options. These options are typically prefixed with a dash (similar to command line options). Some options can override parameters for the Source/Destination. The following options are required top perform the transcode operation:
	Use certain transcoders:
	`-tr_names <transcoder_list>`
	Transcode format:
	`-tr_archive_format <format>`
		String (0 - 768 characters)
		You can leave this field empty.

qualityOfService	This parameter controls how requests are cached during request processing. The possible integer values are as follows: 0: *Default* - Use the default configured in DIVArchive. 1: *Cache and Direct* - Use cache first, then direct if cache is unavailable. 2: *Cache Only* - This is a two stage transfer to tape. 3: *Direct and Cache* - Use direct first, then cache if direct is unavailable. 4: *Direct Only* - This is direct to disk or tape. 5: *Nearline and Direct* - Use Nearline first, then direct if Nearline is unavailable. 6: *Nearline Only* - Create a disk instance if media is tape.	Integer (0 - 6) This can be nil. The *Default* is used when this value is nil.

Table 4–25 (Cont.) transcodeArchive() Request Input Parameters

Parameter	Description	Data Type and Default
bCascadeDelete	If this value is true, the newly created object is linked to the original object as a child, and when the parent is deleted, the child is also deleted.	Boolean (true or false) This field is required.
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–26 transcodeArchive() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	Integer (1 - 10,000,000)

DIVArchive Status Codes

The following are typical DIVArchive transcodeArchive() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1017: Group Does Not Exist
The provided tape group does not exist in the target DIVArchive system.

1023: Object Offline
There are no available instances of the object. For example, this could possibly occur because a required tape was ejected from a tape library.

1027: Instance Does Not Exist
The specified object instance does not exist in DIVArchive or DIVAnet.

1028: Instance Offline
One or more object instances are currently offline and cannot be processed.

1031: Object In Use
One or more objects are currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST transcodeArchive() request:

```
<p:transcodeArchive xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:parentObjectName>PREPROD_July_Edits4</p:parentObjectName>
   <p:parentObjectCategory>PROD</p:parentObjectCategory>
   <p:instance>-1</p:instance>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:objectCategory>PRODProxy</p:objectCategory>
   <p:mediaName>Post4LTO7</p:mediaName>
   <p:comments>Transcoded version of original (PROD)</p:comments>
   <p:archiveOptions>-tr_names TRANS_001 -tr_format MXF</p:archiveOptions>
   <p:qualityOfService>0</p:qualityOfService>
   <p:bCascadeDelete>false</p:bCascadeDelete>
   <p:priorityLevel>50</p:priorityLevel>
</p:transcodeArchive>
```

REST Response Example

The following is an example of a REST response for a transcodeArchive() request:

```
<p:transcodeArchiveResponse
xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:transcodeArchiveResponse>
```

Transfer Files: transferFiles()

This request enables transferring multiple files and directories from a Source/Destination to another Source/Destination without the object being archived in DIVArchive. The call returns a request ID if the request is created successfully. You can monitor the progress of the archive throughout its lifecycle using the getRequestInfo() call. This request is available in DIVArchive, but not DIVAnet (MultiDIVA mode).

Input Parameters

The following is a list of transferFiles() request input parameters:

Table 4–27 transferFiles() Request Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
source	This parameter identifies a DIVArchive Source/Destination name representing a server or disk (such as FTP or CIFS). The files are transferred from this location.	
sourcePathRoot	This parameter identifies the top-level directory of files to transfer as an offset from the Source/Destination.	String (0 - 4000 characters) You can leave this field empty.
fileNamesList	This parameter identifies the list of files (or wildcard patterns) to be transferred. Paths for the files are relative to the specified sourcePathRoot. The <fileNamesList> tag delimits each file. The tag is repeated for each file in the list. You can list a maximum of 1000 files (with or without wildcards). For example: <fileNamesList>misc/*</fileNamesList> <fileNamesList>t.mov</fileNamesList>	String (each file can contain 1 - 400 characters) This field is required.
destination	This parameter identifies a DIVArchive Source/Destination name representing a server or disk (such as FTP or CIFS). The files are transferred to this location.	
destinationPathRoot	This parameter identifies the top-level directory where files are transferred to as an offset from the Source/Destination.	String (0 - 4000 characters) You can leave this field empty.
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 4–28 transferFiles() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)

Table 4–28 (Cont.) transferFiles() Request Returned Values

Parameter	Description	Data Type and Default
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	Integer (1 - 10,000,000)

DIVArchive Status Codes

The following are typical DIVArchive transferFiles() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1018: Source/Destination Does Not Exist
The provided Source/Destination does not exist in the DIVArchive configuration.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST transferFiles() request:

```
<p:transferFiles xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:source>PROD3</p:source?
   <p:sourcePathRoot>jul_03_final</p:sourcePathRoot>
   <p:fileNamesList>ActionPST.avi</p:fileNamesList>
   <p:fileNamesList>ActionPST.wav</p:fileNamesList>
   <p:destination>BCAST1</p:destination>
   <p:destinationPathRoot>wed/allMedia</p:destinationPathRoot>
   <p:priorityLevel>50</p:priorityLevel>
</p:transferFiles>
```

REST Response Example

The following is an example of a REST response for a transferFiles() request:

```
<p:transferFilesResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:transferFilesResponse>
```

5

Device Requests

This chapter describes tape device requests and includes the following services:

- Eject Tape: `ejectTape()`
- Insert Tape: `insertTape()`

Eject Tape: `ejectTape()`

This request ejects media from an archive device; typically tape media from a tape library device. The tapes are usually ejected into a CAP (Cartridge Access Point) located on a tape library. The request is complete after all tapes are successfully ejected.

If the request is created successfully the call returns a request ID. You can monitor the progress of the call throughout its lifecycle using the `getRequestInfo()` call. This request is available in DIVArchive, but not DIVAnet.

Input Parameters

The following is a list of `ejectTape()` request input parameters:

Table 5–1 `ejectTape()` Request Input Parameters

Parameter	Description	Data Type and Defaults
`sessionCode`	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.

`vsnList`	This parameter identifies the volume serial number (barcode) of the tapes to be ejected. These must be unique values within each DIVArchive system. The `<vsnList>` tag delimits each barcode, and is repeated for each barcode in the list. For example:
	`<vsnList>DS4903</vsnList>`
	`<vsnList>FS4905</vsnList>`
		String (each barcode can have 1 - 96 characters)
		One, or multiple, instances of this parameter may appear.

`release`	If you set this value to `true`, all object instances located on the tape are marked in DIVArchive as no longer required (released). *This flag is for informational purposes only*.	Boolean (`true` or `false`) This field is required.

Table 5–1 (Cont.) ejectTape() Request Input Parameters

Parameter	Description	Data Type and Defaults
comment	This parameter is a comment that you can use to indicate to DIVArchive operators why the eject tape operation was performed.	String
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 5–2 ejectTape() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	Integer (1 - 10,000,000)

DIVArchive Status Codes

The following are typical DIVArchive ejectTape() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1033: Tape Does Not Exist
The provided barcode does not exist in the DIVArchive configuration.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST ejectTape() request:

```
<p:ejectTape xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:vsnList>DS4903</p:vsnList>
   <p:vsnList>FS4905</p:vsnList>
   <p:release>false</p:release>
```

```
   <p:comment>Externalized for long term storage</p:comment>
   <p:priorityLevel>50</p:priorityLevel>
</p:ejectTape>
```

REST Response Example

The following is an example of a REST response for an ejectTape() request:

```
<p:ejectTapeResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:ejectTapeResponse>
```

Insert Tape: insertTape()

This request facilitates the insertion of media into an archive device; typically tape media into a tape library device. DIVArchive waits for a short period while an operator inserts the tapes, usually into a CAP in a tape library. The request completes when all affected tapes are successfully inserted into the library.

If the request is called successfully it returns a request ID. You can monitor the progress of the call throughout its lifecycle using the getRequestInfo() call. This request is available in DIVArchive, but not DIVAnet (MultiDiva mode).

Input Parameters

The following is a list of insertTape() request input parameters:

Table 5–3 insertTape() Request Input Parameters

Parameter	Description	Data Type and Defaults
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
require	If you set this value to true, all object instances located on the tape are marked in as required in DIVArchive. *This flag is for informational purposes only*.	Boolean (true or false) This field is required.
priorityLevel	This parameter identifies the initial request priority on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.
acsId	This parameter is the ACS ID (Automated Cartridge System ID, which is the tape library ID) of the device where tapes will be inserted. If you enter -1 for this value, DIVArchive attempts an insert in all tape library devices. This ID can be obtained in the getArchiveSystemInfo() call.	Integer

Table 5–3 (Cont.) insertTape() Request Input Parameters

Parameter	Description	Data Type and Defaults
capId	This parameter identifies the numeric ID of the CAP in the tape library where tapes will be inserted. If you enter -1 for this value, DIVArchive attempts an insert into the first available CAP.	Integer

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 5–4 insertTape() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
requestNumber	This parameter is an ID uniquely identifying the running request. You can monitor the progress of the request using this ID.	Integer (1 - 10,000,000)

DIVArchive Status Codes

The following are typical DIVArchive insertTape() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST insertTape() request:

```
<p:insertTape xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:require>false</p:require>
   <p:priorityLevel>50</p:priorityLevel>
   <p:acsId>-1</p:acsId>
   <p:capId>-1</p:capId>
</p:insertTape>
```

REST Response Example

The following is an example of a REST response for an insertTape() request:

```
<p:insertTapeResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:requestNumber>4905</p1:requestNumber>
   </p:return>
</p:insertTapeResponse>
```

6

Informational Commands

This chapter describes information-based commands and includes the following services:

- DIVArchive System Information: `getArchiveSystemInfo()`
- Source/Destination Information: `getSourceDestinationList()`
- Disk Array Information: `getArrayList()`
- Tape Group Information: `getGroupsList()`
- Object Information: `getObjectInfo()`
- Object Query Information: `getObjectDetailsList()`
- Object Files and Folders Information: `getFilesAndFolders()`
- Request Information: `getRequestInfo()`
- Partial File Restore Request Information: `getPartialRestoreRequestInfo()`
- Finished Requests: `getFinishedRequestList()`
- Storage Plan Information: `getStoragePlanList()`

Overview

The informational services described in this chapter supply information about DIVArchive requests, archived assets, Source/Destinations, archive media, and so on. Some commands return large amounts of information in batches. This requires the caller to take information returned in the response, and pass it back in the next request to get the next information batch.

DIVArchive System Information: `getArchiveSystemInfo()`

This command returns information regarding the configuration and status of DIVArchive. The information includes the number of running requests, the operational status of tape drives, libraries, and Actors, and other information (for example, running requests).

If you issue the command to DIVAnet, the command returns information about the local site. If the `-site {site_name}` option appears in the ***Options*** parameter, DIVAnet returns status on the selected site only. *The information returned by DIVAnet only shows status information for a single site, not all sites*.

Input Parameters

The following table lists the `getArchiveSystemInfo()` input parameters:

Table 6–1 `getArchiveSystemInfo()` Command Input Parameters

Parameter	Description	Data Type and Default
`sessionCode`	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
`options`	If you use DIVAnet, and the `-site {site_name}` option appears in this parameter, DIVAnet returns status on the selected site.	String (0 - 768 characters) You can leave this field empty.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, along with the values below. The command returns generic information about the DIVArchive system. For example, the number of available tape drives, the total number of objects, and the total number of blank tapes. It also returns detailed information regarding Actors, tape drives, and tape library LSMs (Library Storage Module).

Table 6–2 `getArchiveSystemInfo()` Command Return Values

Section	Parameter	Description	Data Type and Default
	`divaStatus`	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)
`actorsAndDrivesList`		The section identifies a list of Actor and drive information. See the following items for details.	
`actorsAndDrivesList`	`actorAddress`	The parameter identifies the Actor IP address.	String (1 - 36 characters)
`actorsAndDrivesList`	`actorisAvailable`	This parameter is `true` if the Actor is available to process requests.	Boolean (`true` or `false`)
`actorsAndDrivesList`	`connectedDrives`	This parameter identifies the ID numbers of the drives connected to the Actor. Each ID is formatted as: `acs-lsm-drive` (for example, `0-0-1`).	String (1 - 192 characters)
`actorsAndDrivesList`	`repackEnabled`	This parameter is `true` if the actor is enabled for Repack.	Boolean (`true` or `false`)

Table 6–2 (Cont.) getArchiveSystemInfo() Command Return Values

Section	Parameter	Description	Data Type and Default
actorsAndDrivesList	classicEnabled	*This is for backward compatibility only*. This parameter is true if all of the *classic* operations are enabled. The classic operations are as follows: ■ cacheArchiveEnabled ■ directArchiveEnabled ■ cacheRestoreEnabled ■ directRestoreEnabled ■ deleteEnabled ■ copyToGroupEnabled ■ associativeCopyEnabled	Boolean (true or false)
actorsAndDrivesList	cacheArchiveEnabled	This parameter is true if the Actor is enabled for Cache Archive Requests.	Boolean (true or false)
actorsAndDrivesList	directArchiveEnabled	This parameter is true if the Actor is enabled for Direct Archive Requests.	Boolean (true or false)
actorsAndDrivesList	cacheRestoreEnabled	This parameter is true if the Actor is enabled for Cache Restore Requests.	Boolean (true or false)
actorsAndDrivesList	directRestoreEnabled	This parameter is true if the Actor is enabled for Direct Restore Requests.	Boolean (true or false)
actorsAndDrivesList	deleteEnabled	This parameter is true if the Actor is enabled for Delete Requests.	Boolean (true or false)
actorsAndDrivesList	copyToGroupEnabled	This parameter is true if the Actor is enabled to perform Copy To Group Requests.	Boolean (true or false)
actorsAndDrivesList	associativeCopyEnabled	This parameter is true if the Actor is enabled to perform Associative Copy Requests.	Boolean (true or false)
actorsAndDrivesList	cacheForRepack	This parameter is not used and the value is always zero.	Integer (0)
	capSize	This parameter identifies the number of slots in the default library CAP.	Integer
drivesList		This section identifies the list of information about tape drives. This tag repeats for each drive in the list. The following seven parameters are contained in each driveList entry.	Not applicable
drivesList	driveName	This parameter identifies the name of the drive and formatted as acs-lsm-drive (for example, 0-0-1).	String (1 - 192 characters)

Table 6–2 (Cont.) getArchiveSystemInfo() Command Return Values

Section	Parameter	Description	Data Type and Default
drivesList	driveType	This parameter is a string describing the type of drive (for example, LTO4).	String (1 - 192 characters)
drivesList	driveTypeId	This parameter identifies a DIVArchive ID uniquely identifying the drive type.	Integer
drivesList	lsmID	This parameter identifies the numeric ID of the LSM containing the drive.	Integer
drivesList	driveIsAvailable	This parameter is true if the drive is available.	Boolean (true or false)
drivesList	repackEnabled	This parameter is true if the drive is enabled for repack.	Boolean (true or false)
drivesList	classicEnabled	This parameter is true if the drive is enabled for classic read and write operations.	Boolean (true or false)
	firstUsedRequestId	This parameter identifies the ID number of the first request processed since the Manager was last started.	Integer
	lastUsedRequestId	This parameter identifies the ID number of the last request processed.	Integer
	libStatus	This parameter identifies the current library status. The possible values are as follows: 0: *Online* 1: *Out of Order* 2: *Unknown*	Integer (0 - 2)
lsmList		This parameter identifies the list of information about LSMs (Library Storage Modules). The <lsmList> tag repeats for each LSM in the list. The following three parameters are contained in each lsmList entry.	
lsmList	lsmName	This parameter identifies the configured LSM name.	String (1 - 50 characters)
lsmList	lsmID	This parameter identifies the numeric ID of the LSM.	Integer
lsmList	lsmIsAvailable	This parameter is true if the LSM is available.	Boolean
	numOfAvailableActors	This parameter identifies the number of running, and available, Actors.	Integer
	numOfAvailableDisks	This parameter identifies the number of disks Online.	Integer

Table 6–2 (Cont.) getArchiveSystemInfo() Command Return Values

Section	Parameter	Description	Data Type and Default
	numOfAvailableDrives	This parameter identifies the number of drives Online.	Integer
	numberOfBlankTapes	This parameter identifies the total number of blank tapes in all tape libraries in DIVArchive.	Integer
	remainSizeOnTapes	This parameter identifies the total amount of space (in megabytes) remaining on all online tapes, in all tape libraries, in DIVArchive.	Integer
	siteIpAddress	This parameter identifies the IP address of the DIVArchive Manager.	String
	siteName	This parameter identifies the configured DIVArchive site name. This is a DIVArchive value and different than the DIVAnet site name.	String
	sitePort	This parameter identifies the Manager connection port number.	Integer
	status	This parameter identifies the status of DIVArchive. This integer value is one of the following: 0: *On* 1: *Off* 2: *Unknown*	Integer (0 - 2)
	totalNumberOfObjects	This parameter identifies the total number of objects in DIVArchive.	Integer
	totalSizeOnTapes	This parameter identifies the total amount of space (in megabytes) in all online tapes, in all tape libraries, in DIVArchive.	Integer

DIVArchive Status Codes

The following are typical DIVArchive getArchiveSystemInfo() command status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success

The request has been created successfully.

1008: Invalid Parameter

One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied

This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST getArchiveSystemInfo() command:

```
<p:getArchiveSystemInfo
xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:options>1678</p:options>
</p:getArchiveSystemInfo>
```

REST Response Example

The following is an example of a REST response for a getArchiveSystemInfo() command:

```
<getArchiveSystemInfoResponse
xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
<return>
   <p:divaStatus>1000</p1:divaStatus>
   <p:info>
      <p1:actorsDrivesList>
        <p1:actorName>actor_001_std_74ny</p1:actorName>
        <p1:actorAddress>10.15.62.68</p1:actorAddress>
        <p1:actorIsAvailable>true</p1:actorIsAvailable>
        <p1:connectedDrives>0-0-0</p1:connectedDrives>
        <p1:repackEnabled>true</p1:repackEnabled>
        <p1:classicEnabled>false</p1:classicEnabled>
        <p1:cacheArchiveEnabled>true</p1:cacheArchiveEnabled>
        <p1:directArchiveEnabled>true</p1:directArchiveEnabled>
        <p1:cacheRestoreEnabled>true</p1:cacheRestoreEnabled>
        <p1:directRestoreEnabled>true</p1:directRestoreEnabled>
        <p1:deleteEnabled>true</p1:deleteEnabled>
        <p1:copyToGroupEnabled>true</p1:copyToGroupEnabled>
        <p1:associativeCopyEnabled>true</p1:associativeCopyEnabled>
        <p1:cacheForRepack>0</p1:cacheForRepack>
      </p1:actorsDrivesList>
      <p1:actorsDrivesList>
        <p1:actorName>actor_002_std_74ny</p1:actorName>
        <p1:actorAddress>10.15.62.69</p1:actorAddress>
        <p1:actorIsAvailable>true</p1:actorIsAvailable>
        <p1:connectedDrives>0-0-1</p1:connectedDrives>
        <p1:repackEnabled>true</p1:repackEnabled>
        <p1:classicEnabled>false</p1:classicEnabled>
        <p1:cacheArchiveEnabled>true</p1:cacheArchiveEnabled>
        <p1:directArchiveEnabled>true</p1:directArchiveEnabled>
        <p1:cacheRestoreEnabled>true</p1:cacheRestoreEnabled>
        <p1:directRestoreEnabled>true</p1:directRestoreEnabled>
        <p1:deleteEnabled>true</p1:deleteEnabled>
        <p1:copyToGroupEnabled>true</p1:copyToGroupEnabled>
        <p1:associativeCopyEnabled>true</p1:associativeCopyEnabled>
        <p1:cacheForRepack>0</p1:cacheForRepack>
      </p1:actorsDrivesList>
      <p1:capSize>8</p1:capSize>
      <p1:drivesList>
        <p1:driveName>0-0-0</p1:driveName>
        <p1:driveTypeID>9</p1:driveTypeID>
        <p1:driveType>SIMU_LT03</p1:driveType>
        <p1:lsmID>0</p1:lsmID>
        <p1:driveIsAvailable>true</p1:driveIsAvailable>
```

```
        <p1:repackEnabled>true</p1:repackEnabled>
        <p1:classicEnabled>true</p1:classicEnabled>
      </p1:drivesList>
      <p1:drivesList>
        <p1:driveName>0-0-1</p1:driveName>
        <p1:driveTypeID>9</p1:driveTypeID>
        <p1:driveType>SIMU_LT03</p1:driveType>
        <p1:lsmID>0</p1:lsmID>
        <p1:driveIsAvailable>true</p1:driveIsAvailable>
        <p1:repackEnabled>true</p1:repackEnabled>
        <p1:classicEnabled>true</p1:classicEnabled>
      </p1:drivesList>
      <p1:firstUsedRequestId>31473</p1:firstUsedRequestId>
      <p1:lastUsedRequestId>33079</p1:lastUsedRequestId>
      <p1:libStatus>0</p1:libStatus>
      <p1:lsmList>
        <p1:lsmName>LSM_0</p1:lsmName>
        <p1:lsmID>0</p1:lsmID>
        <p1:lsmIsAvailable>true</p1:lsmIsAvailable>
      </p1:lsmList>
      <p1:lsmList>
        <p1:lsmName>LSM_1</p1:lsmName>
        <p1:lsmID>65536</p1:lsmID>
        <p1:lsmIsAvailable>true</p1:lsmIsAvailable>
      </p1:lsmList>
      <p1:numOfAvailableActors>3</p1:numOfAvailableActors>
      <p1:numOfAvailableDisks>6</p1:numOfAvailableDisks>
      <p1:numOfAvailableDrives>12</p1:numOfAvailableDrives>
      <p1:numberOfBlankTapes>15998</p1:numberOfBlankTapes>
      <p1:remainSizeOnTapes>225117</p1:remainSizeOnTapes>
      <p1:siteIpAddress>10.15.62.67</p1:siteIpAddress>
      <p1:siteName>local</p1:siteName>
      <p1:sitePort>9005</p1:sitePort>
      <p1:status>0</p1:status>
      <p1:totalNumberOfObjects>3532</p1:totalNumberOfObjects>
      <p1:totalSizeOnTapes>226297</p1:totalSizeOnTapes>
    </ns0:info>
  </return>
</getArchiveSystemInfoResponse>
```

Source/Destination Information: `getSourceDestinationList()`

This command returns information regarding all of the configured DIVArchive Source/Destinations. A DIVArchive Source/Destination is a disk or server (for example, an FTP server) that DIVArchive can archive from or restore to.

If the command is issued to DIVAnet, the command returns all DIVArchive Source/Destinations for *all* sites in the DIVAnet network. The Source/Destination name is prefixed with the site name, and delimited with an underscore. If the `-site {site_name}` option appears in the options parameter, DIVAnet returns Source/Destination information for *only* the selected site.

Input Parameters

The following is a list of `getSourceDestinationList()` command input parameters:

Table 6–3 getSourceDestinationList() Command Input Parameters

Parameter	Description	Data Type and Defaults
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
options	For this parameter there are no options defined in DIVArchive. In DIVAnet, the -site {site_name} option indicates the site to return the list of Source/Destinations from.	String (0 - 768 characters)

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 6–4 getSourceDestinationList() Command Returned Values

Section	Parameter	Description	Data Type and Default
	divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer
arraysInfo		This section repeats for each Source/Destination configured in the system. The following table entries describe the parameters for each <arraysInfo> entry.	
arraysInfo	serversAddress	This parameter identifies the network address of the server or disk.	String
arraysInfo	serversConnectOption	This parameter identifies the DIVArchive connect options (similar to command line options) supplied by default on all requests using the Source/Destination. These options are merged with any options that appear in the request itself.	String
arraysInfo	serversMaxAccess	This parameter identifies the maximum number of concurrent operations from and to the Source/Destination.	Integer
arraysInfo	serversMaxReadAccess	This parameter identifies the maximum number of concurrent read operations from the Source/Destination.	Integer
arraysInfo	serversMaxThroughput	This parameter identifies the maximum throughput to and from the Source/Destination (per request) allowed by DIVArchive. This value is for a single request, and is expressed in megabits per second.	Integer

Table 6–4 (Cont.) getSourceDestinationList() Command Returned Values

Section	Parameter	Description	Data Type and Default
arraysInfo	serversMaxWriteAccess	This parameter identifies the maximum number of concurrent write operations to the Source/Destination.	Integer
arraysInfo	serversName	This parameter identifies the name of the Source/Destination. DIVArchive uses this name requests.	String
arraysInfo	serversProductionSystem	This parameter identifies the DIVArchive production system name that the Source/Destination belongs to.	String
arraysInfo	serversRootPath	This parameter identifies the Source/Destination root path. This parameter is not relevant for all Source/Destinations.	String
arraysInfo	serversSourceType	This parameter identifies the type of Source/Destination as defined in the configuration.	String

DIVArchive Status Codes

The following are typical DIVArchive getSourceDestinationList() command status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST getSourceDestinationList() command:

```
<p:getSourceDestinationList
xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:options></p:options>
</p:getSourceDestinationList>
```

REST Response Example

The following is an example of a REST response for a getSourceDestinationList() command:

```
<p:getSourceDestinationListResponse
xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
<p:return>
```

```
   <p:divaStatus>1000</p:divaStatus>
   <p:arraysInfo>
      <p1:serversAddress>127.0.0.1</p1:serversAddress>
      <p1:serversConnectOption -transfer_timeout 1200 -list_timeout
600</p1:serversConnectOption>
      <p1:serversMaxAccess>10</p1:serversMaxAccess>
      <p1:serversMaxReadAccess>10</p1:serversMaxReadAccess>
      <p1:serversMaxThroughput>100000</p1:serversMaxThroughput>
      <p1:serversMaxWriteAccess>10</p1:serversMaxWriteAccess>
      <p1:serversName>ftp_005_vf</p1:serversName>
      <p1:serversProductionSystem>ps_001</p1:serversProductionSystem>
      <p1:serversRootPath/>
      <p1:serversSourceType>FTP_STANDARD</p1:serversSourceType>
    </p:arraysInfo>
    <p:arraysInfo>
      <p1:serversAddress> </p1:serversAddress>
      <p1:serversConnectOption>-allow_delete_on_source</p1:serversConnectOption>
      <p1:serversMaxAccess>10</p1:serversMaxAccess>
      <p1:serversMaxReadAccess>10</p1:serversMaxReadAccess>
      <p1:serversMaxThroughput>100000</p1:serversMaxThroughput>
      <p1:serversMaxWriteAccess>10</p1:serversMaxWriteAccess>
      <p1:serversName>THA_MOBILE_WK37</p1:serversName>
      <p1:serversProductionSystem>ps_001</p1:serversProductionSystem>
      <p1:serversRootPath>c:\diskSrcDests\THA_MOBILE_WK37</p1:serversRootPath>
      <p1:serversSourceType>DISK</p1:serversSourceType>
    </p:arraysInfo>
<p:/return>
</p:getSourceDestinationListResponse>
```

Disk Array Information: `getArrayList()`

This command returns information regarding DIVArchive disk arrays, including which are available, and the used and available space each disk. A DIVArchive disk array is a grouping of one or more physical disks arrays or disks. This command lists the parameters associated with the array, and the parameters for each of the member disks.

If the command is issued to DIVAnet, the command returns all DIVArchive arrays for all sites in the DIVAnet network. The array name is prefixed with the site name delimited with an underscore. If the `-site {site_name}` option appears in the `options` parameter, DIVAnet returns array information only for the selected site.

Input Parameters

The following is a list of `getArrayList()` request input parameters:

Table 6–5 `getArrayList()` Request Input Parameters

Parameter	Description	Data Type and Defaults
`sessionCode`	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.

Table 6–5 (Cont.) getArrayList() Request Input Parameters

Parameter	Description	Data Type and Defaults
options	If you use DIVAnet, and the -site {site_name} option appears in this parameter, DIVAnet returns information for the selected site only.	String (0 - 768 characters) You can leave this field empty.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 6–6 getArrayList() Command Returned Values

Section	Parameter	Description	Data Type and Default
	divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer
arraysInfo		This <arraysInfo> section repeats for each Source/Destination configured in the system. The following table entries describe the parameters for each <arraysInfo> entry.	
arraysInfo	arrayDesc	This parameter is a description of the array configured in DIVArchive.	String
arraysInfo	arrayName	This parameter identifies the name of the array. This is the media name of an object stored on this DIVArchive array.	String

Table 6–6 (Cont.) getArrayList() Command Returned Values

Section	Parameter	Description	Data Type and Default
arraysInfo	mediaFormatId	This parameter identifies the format type of the tapes in this group; either DIVArchive legacy format, or AXF (Archive Exchange Format). The possible integer values are as follows: 0: Legacy - DIVArchive Legacy format 1: AXFv0.9 2: AXFv1.0	Integer (0 - 2)
arraysInfo	numberOfDisk	This parameter identifies the number of disks configured in the DIVArchive array.	Integer
arraysInfo/arrayDiskList		This <arrayDiskList> section repeats for each member disk in the DIVArchive array. The following table entries describe each <arrayDiskList> parameter.	List
arraysInfo/arrayDiskList	diskCurrentRemainingSize	The remaining size on disk available to DIVArchive (in MB).	Integer
arraysInfo/arrayDiskList	diskIsWritable	This parameter is true if the disk is writable.	Boolean (true or false)
arraysInfo/arrayDiskList	diskMaxThroughput	This parameter (measured in megabits per second) identifies the maximum configured throughput of the disk. Lower values will place an artificial constraint on disk performance.	Integer
arraysInfo/arrayDiskList	diskMinFreeSpace	This parameter identifies the minimum disk space (in kilobytes) to keep unallocated, as specified by the configuration.	Integer

Table 6–6 (Cont.) getArrayList() Command Returned Values

Section	Parameter	Description	Data Type and Default
arraysInfo/arrayDiskList	diskName	This parameter identifies the name of the disk as it is configured in DIVArchive.	String
arraysInfo/arrayDiskList	diskSite	This parameter identifies the DIVArchive site name. This is different from DIVAnet site name. The DIVAnet site name is prepended to the array name.	String
arraysInfo/arrayDiskList	diskStatus	This parameter identifies the overall current status of the disk. This integer value can be one of the following: 0: *Unknown* 1: *Online* 2: *Offline* 3: *Not Visible*	Integer (0 - 3)
arraysInfo/arrayDiskList	diskTotalSize	This parameter identifies the total disk size in kilobytes.	Long
arraysInfo/arrayDiskList	diskArrayName	This parameter identifies the name of the array to which this disk belongs. This field is currently not populated, but instead will be returned as a nil value.	String

DIVArchive Status Codes

The following are typical DIVArchive getArrayList() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST getArrayList() request:

```
<p:getArrayList xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:options></p:options>
</p:getArrayList>
```

REST Response Example

The following is an example of a REST response for an getArrayList() request:

```
<p:getArrayListResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
<p:return>
   <p:divaStatus>1000</p:divaStatus>
   <p:arraysInfo>
      <p1:arrayDesc>AXF two disk example</p1:arrayDesc>
      <p1:arrayName>PROXY_MEDIA005</p1:arrayName>
      <p:mediaFormatId>2</p1:mediaFormatId>
      <p1:numberOfDisk>2</p1:numberOfDisk>
      <p1:arrayDiskList>
         <p1:diskCurrentRemainingSize>873132317</p1:diskCurrentRemainingSize>
         <p1:diskIsWritable>true</p1:diskIsWritable>
         <p1:diskMaxThroughput>100000</p1:diskMaxThroughput>
         <p1:diskMinFreeSpace>0</p1:diskMinFreeSpace>
         <p1:diskName>disk_005_a005</p1:diskName>
         <p1:diskSite>local</p1:diskSite>
         <p1:diskStatus>1</p1:diskStatus>
         <p1:diskTotalSize>1048214524</p1:diskTotalSize>
         <p1:diskArrayName xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:nil="true"/>
      </p1:arrayDiskList>
      <p1:arrayDiskList>
        <p1:diskCurrentRemainingSize>498093</p1:diskCurrentRemainingSize>
        <p1:diskIsWritable>true</p1:diskIsWritable>
        <p1:diskMaxThroughput>100000</p1:diskMaxThroughput>
        <p1:diskMinFreeSpace>0</p1:diskMinFreeSpace>
        <p1:diskName>disk_006_a005</p1:diskName>
        <p1:diskSite>local</p1:diskSite>
        <p1:diskStatus>1</p1:diskStatus>
        <p1:diskTotalSize>1048214524</p1:diskTotalSize>
        <p1:diskArrayName xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:nil="true"/>
      </p1:arrayDiskList>
    </p:arraysInfo>
    <p:arraysInfo>
       <p1:arrayDesc>AXF one disk example</p1:arrayDesc>
       <p1:arrayName>SPM_MEDIA002</p1:arrayName>
       <p1:mediaFormatId>2</p1:mediaFormatId>
       <p1:numberOfDisk>1</p1:numberOfDisk>
       <p1:arrayDiskList>
          <p1:diskCurrentRemainingSize>98530985</p1:diskCurrentRemainingSize>
          <p1:diskIsWritable>true</p1:diskIsWritable>
          <p1:diskMaxThroughput>100000</p1:diskMaxThroughput>
          <p1:diskMinFreeSpace>0</p1:diskMinFreeSpace>
          <p1:diskName>disk_002_a002</p1:diskName>
          <p1:diskSite>local</p1:diskSite>
          <p1:diskStatus>1</p1:diskStatus>
```

```
          <p1:diskTotalSize>1048214524</p1:diskTotalSize>
          <p1:diskArrayName xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:nil="true"/>
      </p1:arrayDiskList>
    </p:arraysInfo>
  </p:return>
</getArrayListResponse>
```

Tape Group Information: getGroupsList()

This command returns information regarding all of the configured DIVArchive tape groups. A DIVArchive tape group is a logical grouping of tape media with similar characteristics. Each DIVArchive object instance belongs to a single tape group. A DIVArchive object instance written to tape has a DIVArchive media name value equivalent to the tape group name. This command returns abbreviated information about each tape group.

If the command is issued to DIVAnet (MultiDIVA mode), the command returns all DIVArchive tape groups for all sites in the DIVAnet network. The group name will be prefixed with the site name and delimited with an underscore. If the `-site {site_name}` option appears in the options parameter, DIVAnet returns group information only for the selected site.

Input Parameters

The following is a list of `getGroupsList()` request input parameters:

Table 6–7 getGroupsList() Request Input Parameters

Parameter	Description	Data Type and Defaults
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 6–8 getGroupsList() Returned Values

Section	Parameter	Description	Data Type and Default
	divaStatus	This parameter identifies the DIVArchive Status code for the operation.	Integer
groups		The <groups> tag repeats for each tape group configured in the system. The following table entries describe the parameters for each <groups> parameter.	List
groups	groupDesc	This parameter is a text description of the tape group as configured in DIVArchive.	String
groups	groupName	This parameter identifies the name of the tape group. This is the media name of an object instance stored in this tape group.	String

Table 6–8 (Cont.) getGroupsList() Returned Values

Section	Parameter	Description	Data Type and Default
groups	mediaFormatId	This parameter identifies the format type of the tapes in this group; either DIVArchive legacy format, or AXF (Archive Exchange Format). The possible integer values are as follows: 0: *Legacy* - DIVArchive Legacy format 1: *AXFv0.9* 2: *AXFv1.0*	Integer (0 - 2)

DIVArchive Status Codes

The following are typical DIVArchive getGroupsList() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST getGroupsList() request:

```
<p:getGroupsList xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
</p:getGroupsList>
```

REST Response Example

The following is an example of a REST response for an getGroupsList() request:

```
<p:getGroupsListResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
<p:return>
   <p:divaStatus>1000</p:divaStatus>
   <p:groups>
      <p1:groupDesc>Commercial spots 01</p1:groupDesc>
      <p1:groupName>CommercialsSpot01</p1:groupName>
      <p1:mediaFormatId>2</p1:mediaFormatId>
   </p:groups>
   <p:groups>
      <p1:groupDesc>Feature Films with commercials embedded</p1:groupDesc>
      <p1:groupName>FeatureWithSpots04</p1:groupName>
      <p1:mediaFormatId>2</p1:mediaFormatId>
   </p:groups>
<p:/return>
</p:getGroupsListResponse>
```

Object Information: getObjectInfo()

This command returns detailed object information about a single archived object. The returned information includes the size of the object, file names, and object instances.

If the command is issued to DIVAnet (MultiDIVA mode), the command returns all object instances on all DIVArchive sites. In each object instance, the media name is prefixed with the site name where it exists and is delimited with an underscore. DIVAnet does not synchronize some object information, such as the inserted parameter.

Input Parameters

The following is a list of getObjectInfo() request input parameters:

Table 6–9 getObjectInfo() Request Input Parameters

Parameter	Description	Data Type and Defaults
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters)
objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object. If this field is empty, and more than one object in DIVArchive matches the objectName, the request will fail.	String (0 - 96 characters)

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 6–10 getObjectInfo() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This is the DIVArchive Status Code for the operation (see the DIVArchive Status Codes section).	Integer (1000 - 1039)
info	The <info> section contains all of the object information details. One, and only one, instance of this tag exists on a successful call. See Object Detail Parameters for the structure of the information returned within the <info> tag	

Object Detail Parameters

The following table describes the archived object parameters returned as part of either a getObjectInfo() call or a getObjectDetailsList() call. If you call getObjectDetailsList(), some parameters may not be returned according to the selected detail level specified in the request. See Object Query Information: getObjectDetailsList() for more information.

Note: Disk-based object instances are named actorInstances in the following parameters table.

Table 6–11 Object Details Parameters (<info> or <objectInfos>)

Section	Parameter	Description	Data Type and Default
actorInstances		This section identifies a disk instance (a DIVArchive array). The <actorInstances> tag repeats for each array instance that belongs to the object	
actorInstances	actor	This parameter identifies the name of the DIVArchive disk array. This is also the media name of the disk-based object instance.	String
actorInstances	instanceID	This parameter identifies number uniquely identifying the instance within an object. This number is not unique across objects. If DIVAnet is used, each instance has its own instance ID that is unique among all instances on all sites. The DIVAnet instance ID is a projection of (but different than) the DIVArchive instance ID.	Integer (0 - 100,000)
	archivingDate	This parameter identifies the date and time when the request was received. It is reflected as a UTC value in seconds since January 01, 1970.	Integer

Table 6–11 (Cont.) Object Details Parameters (<info> or <objectInfos>)

Section	Parameter	Description	Data Type and Default
	filesList	This parameter identifies the list of files contained in the object. Relative paths for the file names are included if the files were archived with relative paths. The <filesList> tag delimits each file, and is repeated for each file in the list. For example: <filesList>misc/t.txt</filesList> <filesList>t.mov</filesList> If the object is complex, the files in the object do not appear. Instead, a single file appears representing the entire object. In this case, call getFilesAndFolders() to iteratively retrieve all of the files and folders within the object.	String (1 - 4000 characters for each file)
	inserted	This parameter is true if either one of the object instances is a tape instance, and all of the tapes comprising the tape instance are present in a tape library, or a disk instance exists and the disk array is online.	Boolean (true or false)
	isComplex	This parameter is true if the object is complex. Complex objects can have a million files and many nested folders.	Boolean (true or false)
	lockStatus	This parameter identifies a code indicating the lock status of the object as follows: 0: *Unlocked* 1: *Locked for restore*	Integer (0 or 1)

Table 6–11 (Cont.) Object Details Parameters (<info> or <objectInfos>)

Section	Parameter	Description	Data Type and Default
	modifiedOrDeleted	This parameter identifies a code indicating the created, modified, or deleted status of the object: 0: *Undefined* 1: *Created or Modified* 2: *Deleted*	Integer (0, 1 or 2)
	nbFilesInComplexComponent	This parameter identifies the number of actual files in the archived object.	Integer
	nbFoldersInComplexComponent	This parameter identifies the number of folders present in the archived object.	Integer
	objectComments	This parameter identifies the user-supplied text regarding the archived object.	String (0 - 4000 characters)
	objectSizeBytes	This parameter identifies the size of the archived object in bytes. This is defined as the sum total of the size of all files.	Integer
	objectSize	This parameter identifies the object size in kilobytes.	Integer
	objectSource	This parameter identifies the DIVArchive Source/Destination name used to archive the object.	String (1 - 96 characters)
objectSummary		This section identifies the formal name of the DIVArchive object (the object name plus category).	String
objectSummary	objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters)

Table 6–11 (Cont.) Object Details Parameters (<info> or <objectInfos>)

Section	Parameter	Description	Data Type and Default
objectSummary	objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object.	String (1 - 96 characters)
	rootDirectory	This parameter identifies the File Path Root within the Source/Destination that was used to archive the object. This value is used as a default on restores if the Restore request contains no File Path Root value.	String
tapeInstances		This section identifies the list of tape instances. The <tapeInstances> tag is repeated for each tape instance present in the object.	String
tapeInstances	groupName	This parameter identifies the name of the tape group where the object instance was assigned.	String
tapeInstances	inserted	This parameter is true if all of the tapes comprising the tape instance are present in a tape library.	Boolean (true or false)
tapeInstances	instanceID	This parameter is true if all of the tapes comprising the tape instance are present in a tape library.	Boolean (true or false)
tapeInstances	reqStatus	This parameter identifies a code indicating the current *require* status of the tape instance: 0: *Marked as Required* 1: *Marked as Releasable*	Integer (0 or 1)

Table 6–11 (Cont.) Object Details Parameters (<info> or <objectInfos>)

Section	Parameter	Description	Data Type and Default
tapeInstances/tapeDesc		This section identifies a list of tapes. The <tapeDesc> tag is repeated for each tape present in the list. If multiple tapes appear, the object instance is considered spanned.	List
tapeInstances/tapeDesc	externalizationComment	This parameter contains the comment that might have been added to DIVArchive when the tape was ejected from a tape library.	String
tapeInstances/tapeDesc	goingToBeRepacked	This parameter is false unless all instances are going to be repacked.	Boolean (true or false)
tapeInstances/tapeDesc	inserted	This parameter is true if this tape is present in a tape library.	Boolean (true or false)
tapeInstances/tapeDesc	tapeFormatId	This parameter identifies the format type of the tapes in this group. The format can be DIVArchive Legacy format, or AXF (Archive Exchange Format). The possible integer values are as follows: 0: *Legacy* - DIVArchive Legacy format 1: *AXFv0.9* 2: *AXFv1.0*	Integer (0, 1, or 2)
tapeInstances/tapeDesc	vsn	This parameter identifies the tape barcode or VSN (Volume Serial Number). This code uniquely identifies the tape within a DIVArchive system.	String (1 - 96 characters)
	toBeRepacked	This parameter is false unless all instances are going to be repacked.	Boolean (true or false)

Table 6–11 (Cont.) Object Details Parameters (<info> or <objectInfos>)

Section	Parameter	Description	Data Type and Default
	uuid	This parameter identifies the UUID (Universally Unique Identifier) assigned to an object when initially created. When a CopyToNew operation is performed, the UUID from the source object is copied to the target object. Therefore, the UUID in this case cannot be used to uniquely identify an object in DIVArchive.	String

DIVArchive Status Codes

The following are typical DIVArchive getObjectInfo() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
The specified object does not exist in the DIVArchive database.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST getObjectInfo() request:

```
<p:getObjectInfo xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:objectCategory>PROD</p:objectCategory>
</p:getObjectInfo>
```

REST Response Example

The following is an example of a REST response for an getObjectInfo() request:

```
<getObjectInfoResponse xmlns="http://interaction.api.ws.diva.fpdigital.com/xsd"
xmlns:ns0="http://response.model.api.ws.diva.fpdigital.com/xsd"
xmlns:ns1="http://model.api.ws.diva.fpdigital.com/xsd">
  <return>
    <p:divaStatus>1000</p:divaStatus>
    <p:info>
      <p1:actorInstances>
        <p1:actor>array_003_legacy_vw</p1:actor>
        <p1:instanceID>1</p1:instanceID>
      </p1:actorInstances>
      <p1:archivingDate>1478282457</p1:archivingDate>
      <p1:filesList>700M</p1:filesList>
      <p1:inserted>true</p1:inserted>
      <p1:isComplex>false</p1:isComplex>
      <p1:lockStatus>0</p1:lockStatus>
      <p1:modifiedOrDeleted>0</p1:modifiedOrDeleted>
      <p1:nbFilesInComplexComponent>1</p1:nbFilesInComplexComponent>
      <p1:nbFoldersInComplexComponent>0</p1:nbFoldersInComplexComponent>
      <p1:objectComments> </p1:objectComments>
      <p1:objectSizeBytes>734003200</p1:objectSizeBytes>
      <p1:objectSize>716800</p1:objectSize>
      <p1:objectSource>ftp_001</p1:objectSource>
      <p1:objectSummary>
        <p1:objectCategory>PROD</p1:objectCategory>
        <p1:objectName>PREPROD_July_Edits4</p1:objectName>
      </p1:objectSummary>
      <p1:rootDirectory>source</p1:rootDirectory>
      <p1:tapeInstances>
        <p1:groupName>group_001_axf</p1:groupName>
        <p1:inserted>true</p1:inserted>
        <p1:instanceID>0</p1:instanceID>
        <p1:reqStatus>0</p1:reqStatus>
        <p1:tapeDesc>
          <p1:externalizationComment/>
          <p1:goingToBeRepacked>false</p1:goingToBeRepacked>
          <p1:inserted>true</p1:inserted>
          <p1:tapeFormatId>0</p1:tapeFormatId>
          <p1:vsn>3L0020</p1:vsn>
        </p1:tapeDesc>
      </p1:tapeInstances>
      <p1:toBeRepacked>false</p1:toBeRepacked>
      <p1:uuid>ba75378c-a8fb-4cf5-8e2e-da4d9b19acd9</p1:uuid>
    </p:info>
  </return>
</getObjectInfoResponse>
```

Object Query Information: `getObjectDetailsList()`

This command enables querying the DIVArchive database. It returns multiple archived objects or tape instances that satisfy the query. The call can return only the names of objects satisfying the query, or the same detailed information returned by the `getObjectInfo()` call. The `getObjectDetailsList()` call can return information in batches (see the following information in this section).

If the command is issued to DIVAnet (MultiDIVA mode), the command queries each site in turn, and returns a batch of information from each site in a round-robin fashion. In each object instance returned, the media name is prefixed with the site name (where it exists), delimited with an underscore. If the request contains a media value that

corresponds to one of the system site names, DIVAnet returns information only from that site.

The getObjectDetailsList() call is meant for queries seeking objects, or tapes, that have had operations performed on them since a given date and time. Object and tape instance details are provided roughly in date and time order (see the following section for ordering details). This call supports the following types of queries:

Object Queries

Enables querying on all archived objects that satisfy a set of criteria. For example, category, media, and date and time. Object Queries focus on events such as object creation and deletion.

Tape Queries

Enables queries specifically on object tape instances, and supports queries on common tape operations such as insert, eject, repack, date and time, and category.

Object Query Modes

You can perform object queries in one of three modes. The number preceding the mode (in the following list) can be used in the objectListType parameter.

Results are returned based on the time an object was created, modified (in terms of number of copies), or deleted. For example, a user queries batches of objects modified in the last 24 hours, and an object was copied twice during that time. It is possible that as the results of the query are retrieved, the same object will appear in two separate batches. The same object will never appear in the same batch.

The three modes for performing queries are as follows:

1: Deleted Since Mode

This query mode returns objects that have been deleted since a specific date and time, indicating that at a specific time the object was either deleted, or an instance was modified, or both.

The results are ordered by the object deletion date and time, or instance deletion date and time, or both. The object name and category are returned (with detail level zero).

As batches of results are returned, you may see the same object returned twice because an object may have several delete operations performed on it throughout its lifetime. That is, several delete and add operations may have occurred.

2: Created Since Mode

This query enables the caller to return objects that have been created since a specific date and time value. If you supply zero for the date and time value, the call returns all objects in the database that satisfy the other query parameters. The results are ordered by the object archive date and time.

3: Modified Since Mode

This query returns objects that have been modified since a specific date and time, indicating that one or more of the following has occurred at a specific time:

- Object Added
- Object Deleted
- Object Exported
- Object Instance Added
- Object Instance Deleted

- Object Instance Exported

The results are ordered by the event time. That is, the object creation or deletion date and time, or instance creation or deletion date and time, or both. The current state of the object at the time of the query is returned.

As batches of results are returned, you may see the same object returned twice because an object may have had several operations performed on it, including deletion, throughout its lifetime. The current state of the object is returned for every event previously listed.

If the object was actually deleted at a certain time, you will not receive all object details (provided you have requested them) for that event. If the object is currently deleted at the time of the query, you will receive no details at all, because the details are no longer available.

Tape Query Event Types

A tape query can be performed according to one or more event types. The valid event types are as follows:

1: Instance Created
This event type returns tape instances that have been created since a specified date and time.

2: Instance Deleted
This event type returns tape instances that have been deleted since a specified date and time.

4: Instance Repacked
This event type returns tape instances that have been repacked since a specified date and time.

8: Instance Ejected
This event type returns tape instances that have had one or more tapes ejected from a tape library since a specific date and time.

16: Instance Inserted
This event type returns tape instances that have had one or more tapes inserted into a tape library since a specific date and time.

Each event type listed has a number. To select multiple events, you add these numbers together, and you enter the resulting number in the `objectsListType` parameter. For example, a query that returns tape instances that have been created or repacked within a given time would have `5` (the sum of `1: Instance Created` + `4: Instance Repacked`) as the value for the `objectsListType` parameter.

The results are ordered by the event date and time.

Batching and List Position

The query is specified in the initial call, and the results are returned in batches. You specify the batch size in the request. You must not specify any `listPosition` values in the initial query. To get the next batch, take the `listPosition` value returned from the previous call, and pass it on the next. Keep calling `getObjectDetailsList()` to receive each batch until you receive an empty list, indicating there are no more objects.

As batches of results are returned, you may see the same object returned twice in the list. This is especially true when Modified Since Mode is used. In this case, an object

will be returned in the list for every insert or delete performed during the object's lifetime.

Because the getObjectDetailsList() call is stateless, you must maintain the state of the query by taking the returned list position parameters and passing them in as input on the next call. *You must pass all of the returned list position parameters in the order that they were returned*.

Continuous Polling for Updates

If you create a query and receive an empty list as a response, you can try the query again later using the returned list position values. A future call will return new objects if updates have been made to DIVArchive. In the query, use the createdSince parameter to get updates on new objects, or the modifiedSince parameter for updates on new or deleted objects or instances. Do not mix the createdSince parameter with modifiedSince, or deletedSince in the same calling sequence.

You can create a query that returns any new updates in the future, even though no objects currently satisfy the criteria. When you create the new query, provide the current time as the value for the initialTime parameter. It is possible that you may receive an empty list on the first response. In this case, you can archive a new object, and then attempt the same query (with the list position you just received) again. The newly archived object, and a new list position value, will be returned.

Input Parameters

The following is a list of getObjectDetailsList() request input parameters:

Table 6–12 getObjectDetailsList() Request Input Parameters

Parameter	Description	Data Type and Defaults
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
isFirstTime	You set this value to true when initiating a new query. You set this value to false as you retrieve batches of results from the initial query.	Boolean (true or false) This field is required.
listPosition	These values mark the position of the returned elements in the list. These values are not specified in the initial call (where isFirstTime=true). The <listPosition> tag is repeated for each string value to be passed in the request. For example: <listPosition>*str1*</listPosition> <listPosition>*str2*</listPosition> <listPosition>*str3*</listPosition> You take the returned listPosition values from one call, and feed them as input into the request to retrieve the next batch.	This field is optional.
listType	This parameter identifies the type of query being performed as follows: 1: *Object List* 2: *Tape Info List*	Integer (1 or 2)

Table 6–12 (Cont.) getObjectDetailsList() Request Input Parameters

Parameter	Description	Data Type and Defaults
initialTime	This call returns objects and tape instances having events that occur after the initialTime value. This is a UTC value in seconds since January 01, 1970.	Integer
objectsListType	If you are performing an object query (listType=1), use one of the following: 1: *Deleted Since* 2: *Created Since* 3: *Modified Since* If you are performing a tape instance query (listType=2), select one or more options by adding the numbers together from the following options: 1: *Instance Created* 2: *Instance Deleted* 4: *Instance Repacked* 8: *Instance Ejected* 16: *Instance Inserted* *See Object Query Modes for detailed information on these options.*	Integer (1 - 32)
maxListSize	This parameter identifies the batch size, or maximum number of objects and tape instances returned in a single call. Do not rely on the number of items returned to predict the end of the list. There is no guarantee that exactly the specified number items will appear in a batch.	Integer (1 - 1000) This field is required.
objectName	This parameter identifies the object name being queried. You can use a wildcard in the value.	String
objectCategory	This parameter identifies the DIVArchive object category being queried. You can use a wildcard in the value.	String
mediaName	This parameter identifies the DIVArchive media name being queried. You can use a wildcard in the value.	String
levelOfDetail	There are four levels of detail that can be returned: 0: *Object Name and Category* 1: *Medium* - this level adds comments, source path root, and archive date. 2: *Component* - this level adds object size and file names. 3: *All* - this level returns all information in getObjectInfo(). The following limitations apply: ■ *Deleted Since* only works at detail level 0. ■ Tape instance queries only support detail levels 0 and 3.	Integer (0 - 3)

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 6–13 getObjectDetailsList() Return Values

Section	Parameter	Description	Data Type and Default
	divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer
objectDetailsList		This section contains the list of objects or tapes requested in the query.	
objectDetailsList	listType	This parameter identifies the type of query being performed as follows: 1: *Object List* 2: *Tape Info List*	Integer (1 or 2)
objectDetailsList	siteId	This parameter identifies site name of the DIVArchive system that satisfied the request. If you use DIVAnet, this will correspond to the DIVAnet site name.	String
objectDetailsList	listPosition	This parameter identifies a list of strings that maintain the state of the query. You pass all of these values as input to the next getObjectDetailsList() call to retrieve the next batch. The <listPosition> tag repeats for each element of the list.	
objectDetailsList /objectInfos		This section only appears when an object query has been performed. This tag repeats for each object returned. See Object Detail Parameters for a full description of the object parameters that appear under the <objectInfos> tag.	
objectDetailsList /objectTapeInfos		This section only appears when a tape instance query has been performed. This tag repeats for each tape instance returned. The following section describes the parameters within the <objectTapeInfos> tag.	

Tape Query Returned Values

The following table describes the values returned when performing a tape query. The parameters are repeated for each tape instance returned.

Table 6–14 <objectTapeInfos> Request Returned Values

Section	Parameter	Description	Data Type and Default
objectSummary		This parameter identifies the formal name of the DIVArchive object (the object name plus category). This occurs once in the objectTapeInfos structure.	
objectSummary	objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters)
objectSummary	objectCategory	This section describes a DIVArchive tape instance. This parameter, along with objectName, creates the full formal name of a DIVArchive object.	String (1 - 96 characters)
tapeInstanceInfo		This parameter identifies the tape instance information. This tag occurs once within the <objectTapeInfos> structure.	
tapeInstanceInfo	req	This parameter indicates the current require status of the tape instance: 0: *Marked as Required* 1: *Marked as Releasable*	Integer (0 or 1)
tapeInstanceInfo	reqDate		
tapeInstanceInfo	instanceID	This parameter is true if all of the tapes comprising the tape instance are present in a tape library.	Boolean (true or false)
tapeInstanceInfo/tapeDesc		This section identifies a list of tapes. The <tapeDesc> tag is repeated for each tape present in the list. If multiple tapes appear, the tape instance is considered to be spanned.	
tapeInstanceInfo/tapeDesc	externalizationComment	This parameter identifies the externalizationComment parameter, and contains the comment that may have been added to DIVArchive when the tape was ejected from a tape library.	
tapeInstanceInfo/tapeDesc	goingToBeRepacked	This parameter is false unless all instances are going to be repacked.	Boolean (true or false)

Table 6–14 (Cont.) <objectTapeInfos> Request Returned Values

Section	Parameter	Description	Data Type and Default
tapeInstanceInfo/tapeDesc	inserted	This parameter is true if this tape is present in a tape library.	Boolean (true or false)
tapeInstanceInfo/tapeDesc	tapeFormatId	This parameter identifies the format type of the tapes in this group; either DIVArchive legacy format, or AXF (Archive Exchange Format). The possible integer values are as follows: 0: *Legacy* - DIVArchive Legacy format 1: *AXFv0.9* 2: *AXFv1.0*	Integer (0, 1 or 2)
tapeInstanceInfo/tapeDesc	vsn	This parameter identifies the barcode of the tape, or VSN (Volume Serial Number). This code uniquely identifies the tape within a DIVArchive system.	String (1 - 96 characters)

DIVArchive Status Codes

The following are typical DIVArchive getObjectDetailsList() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Object Query Request Example

The following is an example of a REST getObjectDetailsList() Object Query request:

```
<p:getObjectDetailsList
xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>9610ff91-8721-4af9-905d-316a60ebdb27</p:sessionCode>
   <p:isFirstTime>1</p:isFirstTime>
   <p:initialTime>1479300291</p:initialTime>
   <!--p:initialTime>0</p:initialTime-->
   <p:listType>1</p:listType>
   <p:objectsListType>2</p:objectsListType>
   <!--1 or more repetitions:-->
   <p:listPosition></p:listPosition>
   <p:maxListSize>3</p:maxListSize>
   <p:objectName>*</p:objectName>
   <p:objectCategory>api</p:objectCategory>
   <p:mediaName>*</p:mediaName>
   <p:levelOfDetail>3</p:levelOfDetail>
</p:getObjectDetailsList>
```

REST Object Query Response Example

The following is an example of a REST response for an getObjectDetailsList() Object Query request:

```
<getObjectDetailsListResponse
xmlns="http://interaction.api.ws.diva.fpdigital.com/xsd"
xmlns:ns0="http://response.model.api.ws.diva.fpdigital.com/xsd"
xmlns:ns1="http://model.api.ws.diva.fpdigital.com/xsd">
  <return>
    <ns0:divaStatus>1000</ns0:divaStatus>
    <ns0:objectDetailsList>
      <ns1:listType>1</ns1:listType>
      <ns1:siteID>DIVA</ns1:siteID>
      <ns1:listPosition>1479304660</ns1:listPosition>
      <ns1:listPosition>1</ns1:listPosition>
      <ns1:listPosition>1479305114000</ns1:listPosition>
      <ns1:objectInfos>
        <ns1:actorInstances>
          <ns1:actor>array_002_axf</ns1:actor>
          <ns1:instanceID>0</ns1:instanceID>
        </ns1:actorInstances>
        <ns1:archivingDate>1479300506</ns1:archivingDate>
        <ns1:filesList>101</ns1:filesList>
        <ns1:filesList>102</ns1:filesList>
        <ns1:filesList>103</ns1:filesList>
        <ns1:inserted>true</ns1:inserted>
        <ns1:isComplex>false</ns1:isComplex>
        <ns1:lockStatus>0</ns1:lockStatus>
        <ns1:modifiedOrDeleted>1</ns1:modifiedOrDeleted>
        <ns1:nbFilesInComplexComponent>-1</ns1:nbFilesInComplexComponent>
        <ns1:nbFoldersInComplexComponent>-1</ns1:nbFoldersInComplexComponent>
        <ns1:objectComments> </ns1:objectComments>
        <ns1:objectSizeBytes>33084</ns1:objectSizeBytes>
        <ns1:objectSize>33</ns1:objectSize>
        <ns1:objectSource>ftp_001</ns1:objectSource>
        <ns1:objectSummary>
          <ns1:objectCategory>api</ns1:objectCategory>
          <ns1:objectName>godl-1</ns1:objectName>
        </ns1:objectSummary>
        <ns1:rootDirectory>source</ns1:rootDirectory>
        <ns1:tapeInstances>
          <ns1:groupName>group_005_legacy</ns1:groupName>
          <ns1:inserted>true</ns1:inserted>
          <ns1:instanceID>1</ns1:instanceID>
          <ns1:reqStatus>0</ns1:reqStatus>
          <ns1:tapeDesc>
            <ns1:externalizationComment/>
            <ns1:goingToBeRepacked>false</ns1:goingToBeRepacked>
            <ns1:inserted>true</ns1:inserted>
            <ns1:tapeFormatId>0</ns1:tapeFormatId>
            <ns1:vsn>3L2020</ns1:vsn>
          </ns1:tapeDesc>
        </ns1:tapeInstances>
        <ns1:tapeInstances>
          <ns1:groupName>group_001_axf</ns1:groupName>
          <ns1:inserted>true</ns1:inserted>
          <ns1:instanceID>2</ns1:instanceID>
          <ns1:reqStatus>0</ns1:reqStatus>
          <ns1:tapeDesc>
            <ns1:externalizationComment/>
```

```
            <ns1:goingToBeRepacked>false</ns1:goingToBeRepacked>
            <ns1:inserted>true</ns1:inserted>
            <ns1:tapeFormatId>0</ns1:tapeFormatId>
            <ns1:vsn>3L0020</ns1:vsn>
          </ns1:tapeDesc>
        </ns1:tapeInstances>
        <ns1:toBeRepacked>false</ns1:toBeRepacked>
        <ns1:uuid>1a72fb52-6a06-40a5-9380-566c9558d56e</ns1:uuid>
      </ns1:objectInfos>
      <ns1:objectInfos>
        <ns1:archivingDate>1479304660</ns1:archivingDate>
        <ns1:filesList>101</ns1:filesList>
        <ns1:filesList>103</ns1:filesList>
        <ns1:filesList>102</ns1:filesList>
        <ns1:inserted>true</ns1:inserted>
        <ns1:isComplex>false</ns1:isComplex>
        <ns1:lockStatus>0</ns1:lockStatus>
        <ns1:modifiedOrDeleted>1</ns1:modifiedOrDeleted>
        <ns1:nbFilesInComplexComponent>-1</ns1:nbFilesInComplexComponent>
        <ns1:nbFoldersInComplexComponent>-1</ns1:nbFoldersInComplexComponent>
        <ns1:objectComments> </ns1:objectComments>
        <ns1:objectSizeBytes>33084</ns1:objectSizeBytes>
        <ns1:objectSize>33</ns1:objectSize>
        <ns1:objectSource>ftp_001</ns1:objectSource>
        <ns1:objectSummary>
          <ns1:objectCategory>api</ns1:objectCategory>
          <ns1:objectName>godl-2</ns1:objectName>
        </ns1:objectSummary>
        <ns1:rootDirectory>source</ns1:rootDirectory>
        <ns1:tapeInstances>
          <ns1:groupName>group_001_axf</ns1:groupName>
          <ns1:inserted>true</ns1:inserted>
          <ns1:instanceID>0</ns1:instanceID>
          <ns1:reqStatus>0</ns1:reqStatus>
          <ns1:tapeDesc>
            <ns1:externalizationComment/>
            <ns1:goingToBeRepacked>false</ns1:goingToBeRepacked>
            <ns1:inserted>true</ns1:inserted>
            <ns1:tapeFormatId>0</ns1:tapeFormatId>
            <ns1:vsn>3L0020</ns1:vsn>
          </ns1:tapeDesc>
        </ns1:tapeInstances>
        <ns1:toBeRepacked>false</ns1:toBeRepacked>
        <ns1:uuid>0ab3afc5-7a77-4b35-a297-b755e740c41d</ns1:uuid>
      </ns1:objectInfos>
    </ns0:objectDetailsList>
  </return>
</getObjectDetailsListResponse>
```

REST Tape Query Request Example

The following is an example of a REST `getObjectDetailsList()` Tape Query request:

```
<p:getObjectDetailsList
xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>0cc686cc-7654-4c02-b747-388edeb0f577</p:sessionCode>
   <p:isFirstTime>true</p:isFirstTime>
   <p:initialTime>0</p:initialTime>
   <p:listType>2</p:listType>
```

```
   <p:objectsListType>1</p:objectsListType>
   <!--1 or more repetitions:-->
   <p:listPosition></p:listPosition>
   <p:maxListSize>3</p:maxListSize>
   <p:objectName>*</p:objectName>
   <p:objectCategory>api</p:objectCategory>
   <p:mediaName>*</p:mediaName>
   <p:levelOfDetail>3</p:levelOfDetail>
</p:getObjectDetailsList>
```

REST Tape Query Response Example

The following is an example of a REST response for a `getObjectDetailsList()` Tape Query request:

```
<getObjectDetailsListResponse
xmlns="http://interaction.api.ws.diva.fpdigital.com/xsd"
xmlns:p="http://response.model.api.ws.diva.fpdigital.com/xsd"
xmlns:p1="http://model.api.ws.diva.fpdigital.com/xsd">
  <return>
    <p:divaStatus>1000</p:divaStatus>
    <p:objectDetailsList>
      <p1:listType>2</p1:listType>
      <p1:siteID>DIVA</p1:siteID>
      <p1:listPosition>0</p1:listPosition>
      <p1:listPosition>godl-2</p1:listPosition>
      <p1:listPosition>api</p1:listPosition>
      <p1:objectTapeInfos>
        <p1:objectSummary>
          <p1:objectCategory>api</p1:objectCategory>
          <p1:objectName>700M</p1:objectName>
        </p1:objectSummary>
        <p1:tapeInstanceInfo>
          <p1:req>1</p1:req>
          <p1:reqDate>1478282456</p1:reqDate>
          <p1:instanceID>0</p1:instanceID>
          <p1:tapeDesc>
            <p1:externalizationComment> </p1:externalizationComment>
            <p1:goingToBeRepacked>false</p1:goingToBeRepacked>
            <p1:inserted>true</p1:inserted>
            <p1:tapeFormatId>2</p1:tapeFormatId>
            <p1:vsn>3L0020</p1:vsn>
          </p1:tapeDesc>
        </p1:tapeInstanceInfo>
      </p1:objectTapeInfos>
      <p1:objectTapeInfos>
        <p1:objectSummary>
          <p1:objectCategory>api</p1:objectCategory>
          <p1:objectName>godl-1</p1:objectName>
        </p1:objectSummary>
        <p1:tapeInstanceInfo>
          <p1:req>1</p1:req>
          <p1:reqDate>1479301156</p1:reqDate>
          <p1:instanceID>1</p1:instanceID>
          <p1:tapeDesc>
            <p1:externalizationComment> </p1:externalizationComment>
            <p1:goingToBeRepacked>false</p1:goingToBeRepacked>
            <p1:inserted>true</p1:inserted>
            <p1:tapeFormatId>0</p1:tapeFormatId>
```

```
            <p1:vsn>3L2020</p1:vsn>
          </p1:tapeDesc>
        </p1:tapeInstanceInfo>
        <p1:tapeInstanceInfo>
          <p1:req>1</p1:req>
          <p1:reqDate>1479301168</p1:reqDate>
          <p1:instanceID>2</p1:instanceID>
          <p1:tapeDesc>
            <p1:externalizationComment> </p1:externalizationComment>
            <p1:goingToBeRepacked>false</p1:goingToBeRepacked>
            <p1:inserted>true</p1:inserted>
            <p1:tapeFormatId>2</p1:tapeFormatId>
            <p1:vsn>3L0020</p1:vsn>
          </p1:tapeDesc>
        </p1:tapeInstanceInfo>
      </p1:objectTapeInfos>
      <p1:objectTapeInfos>
        <p1:objectSummary>
          <p1:objectCategory>api</p1:objectCategory>
          <p1:objectName>godl-2</p1:objectName>
        </p1:objectSummary>
        <p1:tapeInstanceInfo>
          <p1:req>1</p1:req>
          <p1:reqDate>1479304660</p1:reqDate>
          <p1:instanceID>0</p1:instanceID>
          <p1:tapeDesc>
            <p1:externalizationComment> </p1:externalizationComment>
            <p1:goingToBeRepacked>false</p1:goingToBeRepacked>
            <p1:inserted>true</p1:inserted>
            <p1:tapeFormatId>2</p1:tapeFormatId>
            <p1:vsn>3L0020</p1:vsn>
          </p1:tapeDesc>
        </p1:tapeInstanceInfo>
      </p1:objectTapeInfos>
    </p:objectDetailsList>
  </return>
</getObjectDetailsListResponse>
```

Object Files and Folders Information: `getFilesAndFolders()`

This command returns information regarding the files and folders in a DIVArchive archived object. It can return file size, file name (with path offset information if present), and checksum information (if available). It returns information for every file and folder in the archived object.

This command returns information in batches. To traverse the list of files, set the `startIndex` parameter in the request to `1`. In the response, take the `nextStartIndex` value and pass it as the `startIndex` in the next `getFilesAndFolders()` call. You proceed in this manner until the `nextStartIndex` value you receive becomes negative. This negative value means that you should process any returned entries, but you should not call the method again.

Note: If you have reached the end of the list, and pass a negative `startIndex` value to the next call, you will receive an empty list, but also a `1008` error (Invalid Parameter).

This call is available in both DIVArchive and DIVAnet. If you are using DIVAnet (MultiDIVA mode), DIVAnet chooses from which DIVArchive site to retrieve the file and folder information.

You can use the getObjectInfo() call to return the file names of each file in the object, but the following describes why you may want to call getFilesAndFolders() instead:

Folder Information
The getFilesAndFolders() call can optionally return all folders in an object, along with all of the files.

Complex Objects
The getFilesAndFolders() call returns file information for all types of archived objects, including complex objects. The getObjectInfo() call does not return the file names if the object is a complex object.

Batching
The getFilesAndFolders() call returns files in batches, enabling the caller to safely retrieve thousands of files and folders.

Sizes and Checksums
The getFilesAndFolders() call returns the file sizes and checksums for each file in an archived object.

Input Parameters

The following is a list of getFilesAndFolders() request input parameters:

Table 6–15 getFilesAndFolders() Request Input Parameters

Parameter	Description	Data Type and Defaults
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object to provide information on. This parameter, along with category, creates the unique formal name of a DIVArchive object.	String (1 - 192 characters)
category	This parameter identifies the object category of the specified object. This parameter, along with objectName, creates the unique formal name of a DIVArchive object.	String (1 - 96 characters)
startIndex	This parameter identifies a unique file and folder ID referencing the first file or folder to return in the list. You should set this value to 1 to retrieve the beginning of the list. After retrieving a batch of file and folder entries, pass the returned value of nextStartIndex as the startIndex to retrieve the next batch. *You must not try to increment this value in your program because sometimes there are gaps in the IDs that are returned.*	Integer (greater than or equal to 1)

Table 6–15 (Cont.) getFilesAndFolders() Request Input Parameters

Parameter	Description	Data Type and Defaults
batchSize	This parameter identifies the maximum size of the returned list. You must set this to a value no greater than 1000. Oracle recommends setting the value to 500. *This is only a suggestion and may be overridden by the underlying functionality*. This parameter does not guarantee that the list will be a certain size.	Integer (1 - 1000) This field is required.
listType	This parameter specifies what file and folder information is returned. You can select one of the following: 0: *Files* - file information for all files 1: *Folders* - folder information for all folders 2: *Both Files and Folders*	Integer (0, 1 or 2) This field is required.
options	If you use DIVAnet, -site {site_name} enables the caller to choose which site to pull the file and folder information from.	String (0 - 768 characters)

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 6–16 getFilesAndFolders() Request Returned Values

Section	Parameter	Description	Data Type and Default
objectSummary		This section identifies a tag that contains the objectName and objectCategory.	
objectSummary	objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the unique formal name of a DIVArchive object.	String (1 - 192 characters)
objectSummary	objectCategory	This parameter identifies the object category. This parameter, along with objectName, creates the unique formal name of a DIVArchive object.	String (1 - 96 characters)
	isComplex	This parameter is true if the object is a DIVArchive complex object. Complex objects can have hundreds of thousands of files, but the file names are not returned using the getObjectInfo() call.	Boolean (true or false)
	nextStartIndex	After the first call, you use this value as the startIndex input parameter for the following call.	Integer
	siteName	This parameter identifies the DIVArchive site name. If connected to DIVAnet (in MultiDiva mode), then the DIVAnet site name will be populated.	String

Table 6–16 (Cont.) getFilesAndFolders() Request Returned Values

Section	Parameter	Description	Data Type and Default
fileAndFolderList		This section identifies the list of files, folders, or both, in an archived object. This tag repeats for each drive in the list. The following parameters are contained in each file and folder entry.	List
fileAndFolderList	fileOrFolderName	This parameter identifies the file name (with embedded path information, if present), or the folder name.	String
fileAndFolderList	isDirectory	This parameter is true if the component is a directory.	Boolean (true or false)
fileAndFolderList	sizeBytes	This parameter identifies the size of the file in bytes. *This is valid only for files*.	Integer
fileAndFolderList	fileId	This parameter identifies a file or folder ID that uniquely identifies the file or folder within the archived object.	Integer
fileAndFolderList	totalNumFilesAndFolders	This parameter identifies the total number of files in the folder (including subfolders). *This statistic is only valid for folders that are part of complex objects*.	Integer
fileAndFolderList	totalSizeFilesAndFolders	This parameter identifies the total size of all files in the folder (including subfolders). *This statistic is only valid for folders that are part of complex objects*.	Integer
fileAndFolderList/checksums		This section identifies the checksum information for each file. Directories do not have checksums.	Not applicable
fileAndFolderList/checksums	checksumType	This parameter identifies the type of checksum. For example, MD5, SHA1, and so on.	String
fileAndFolderList/checksums	checksumValue	This parameter identifies the value of the checksum in hexadecimal string format. The length of this string varies based on the type of checksum in use.	String
fileAndFolderList/checksums	isGenuine	This parameter is true if the checksum was provided at the time of the archive, and verified as *Genuine*.	Boolean (true or false)

DIVArchive Status Codes

The following are typical DIVArchive getFilesAndFolders() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success

The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1019: End Of List Reached
The end of the list has been reached, and there are no more entries to retrieve.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST getFilesAndFolders() request:

```
<p:getFilesAndFolders xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>4eb23590-c5a8-4af9-97c9-9234f1b571cc</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:objectCategory>PROD</p:objectCategory>
   <p:startIndex>1</p:startIndex>
   <p:batchSize>500</p:batchSize>
   <p:listType>2</p:listType>
   <p:options></p:options>
</p:getFilesAndFolders>
```

REST Response Example

The following is an example of a REST response for an getFilesAndFolders() request:

```
<getFilesAndFoldersResponse
xmlns="http://interaction.api.ws.diva.fpdigital.com/xsd"
xmlns:ns0="http://response.model.api.ws.diva.fpdigital.com/xsd"
xmlns:ns1="http://model.api.ws.diva.fpdigital.com/xsd">
  <return>
    <p:divaStatus>1000</p:divaStatus>
    <p:divaFilesAndFolders>
      <p1:objectSummary>
        <p1:objectCategory>SM_tryit</p1:objectCategory>
        <p1:objectName>tryit_does_not_exist</p1:objectName>
      </p1:objectSummary>
      <p1:isComplex>false</p1:isComplex>
      <p1:nextStartIndex>-1</p1:nextStartIndex>
      <p1:siteName>local</p1:siteName>
      <p1:fileAndFolderList>
        <p1:checksums>
          <p1:checksumType>MD5</p1:checksumType>
          <p1:checksumValue>cf975e6243044b95272a29247046c971</p1:checksumValue>
          <p1:isGenuineChecksum>false</p1:isGenuineChecksum>
        </p1:checksums>
        <p1:fileOrFolderName>kiki.tx2</p1:fileOrFolderName>
        <p1:isDirectory>false</p1:isDirectory>
        <p1:fileId>1</p1:fileId>
        <p1:sizeBytes>12</p1:sizeBytes>
        <p1:totalNumFilesFolders>0</p1:totalNumFilesFolders>
        <p1:totalSizeFilesFolders>0</p1:totalSizeFilesFolders>
      </p1:fileAndFolderList>
      <p1:fileAndFolderList>
        <p1:checksums>
          <p1:checksumType>MD5</p1:checksumType>
```

```
            <p1:checksumValue>5f75b988ec56974f9f6194ad7a5baf64</p1:checksumValue>
            <p1:isGenuineChecksum>false</p1:isGenuineChecksum>
          </p1:checksums>
          <p1:fileOrFolderName>complex.0004.txt</p1:fileOrFolderName>
          <p1:isDirectory>false</p1:isDirectory>
          <p1:fileId>2</p1:fileId>
          <p1:sizeBytes>4090</p1:sizeBytes>
          <p1:totalNumFilesFolders>0</p1:totalNumFilesFolders>
          <p1:totalSizeFilesFolders>0</p1:totalSizeFilesFolders>
        </p1:fileAndFolderList>
      </p:divaFilesAndFolders>
    </return>
</getFilesAndFoldersResponse>
```

Request Information: `getRequestInfo()`

This command returns information about a particular DIVArchive or DIVAnet request, and includes status and error message text. You can call it for requests currently executing, and completed requests. The caller provides the ID of the request (`requestNumber`). The ID is returned after a content request (for example, `archiveObject()`, `copy()`, and so on) is first invoked. The `getRequestInfo()` call indicates whether the request has completed.

If the command is issued to DIVAnet, then DIVAnet returns the status of the DIVAnet request. DIVAnet creates requests on one, or more, DIVArchive sites to fulfill the DIVAnet request.

Input Parameters

The following is a list of `getRequestInfo()` request input parameters:

Table 6–17 `getRequestInfo()` Request Input Parameters

Parameter	Description	Data Type and Defaults
`sessionCode`	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
requestNumber	This parameter identifies an ID that uniquely identifies the request. You can monitor the progress of the request by using this ID.	Integer This parameter is required.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 6–18 getRequestInfo() Request Returned Values

Section	Parameter	Description	Data Type and Default
	divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)
divaRequestInfo		This section identifies the tag containing the request information.	Occurs only one time.
divaRequestInfo/abortionReason		This section identifies the tag indicating the reason why the request terminated.	Occurs only one time.
divaRequestInfo/abortionReason	code	This parameter identifies a code indicating a possible reason why the request terminated: 0: *None* 1: *Tape Drive* 2: *Tape - individual tape issue* 3: *Actor* 4: *Disk* 5: *Disk Full* 6: *Source or Destination* 7: *Resources* 8: *Tape Library* 9: *Input Parameters* 10: *Unknown* 11: *Internal Error*	Integer (0 - 11)
divaRequestInfo/abortionReason	description	This parameter identifies a text representation of the abortionReason code.	String
divaRequestInfo	additionalInfo	This parameter identifies the an unstructured XML value, encoded as a string, that indicates additional information associated with the request (see the following table entries). This information is not necessarily valid after the request completes.	

Table 6–18 (Cont.) getRequestInfo() Request Returned Values

Section	Parameter	Description	Data Type and Default
divaRequestInfo	currentPriority	This parameter identifies the current priority of the request on a scale of 0 to 100 (100 is the highest priority). The current priority of the request increases the longer the request waits to be executed.	Integer (0 - 100)
divaRequestInfo/objectSummary		This section identifies a tag that contains the objectName and objectCategory associated with the request.	
divaRequestInfo/objectSummary	objectName	This parameter identifies the name of the object. This parameter, along with objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192)
divaRequestInfo/objectSummary	objectCategory	This parameter identifies the name of the object category. This parameter, along with objectName, creates the full formal name of a DIVArchive object.	String (1 - 96)
divaRequestInfo	progress	This parameter identifies a value indicating the percentage of the request completed. If multiple transfers are required to process a request (for example, if Verify On Restore was selected), the percentage may reset back to 0 when performing the second transfer. Therefore, a progress of 100 does not necessarily indicate that the request has completed. In addition, -1 may be returned if the request terminated.	Integer (-1, 0 - 100)
divaRequestInfo/repackTapes		This section identifies a tag that applies only to Repack requests.	
divaRequestInfo/repackTapes	destinationTape	This parameter identifies the destination tape to store the repacked information for a repack request.	String (0 - 96)

Table 6–18 (Cont.) getRequestInfo() Request Returned Values

Section	Parameter	Description	Data Type and Default
divaRequestInfo/repackTapes	sourceTape	This parameter identifies the fragmented tape to be repacked for a repack request.	String (0 - 96)
divaRequestInfo	requestNumber	This parameter identifies the ID that uniquely identifies the request.	Integer
divaRequestInfo	requestState	This parameter identifies the overall state of the request. The following states indicate that the request has finished execution: 3: *Completed Successfully* 4: *Aborted* 5: *Cancelled* 11: *Partially Aborted* The following states indicate that the request is running, or the state is unknown: 6: *Unknown* 12: *Running*	Integer (3 - 12)

Table 6–18 (Cont.) getRequestInfo() Request Returned Values

Section	Parameter	Description	Data Type and Default
divaRequestInfo	requestType	This parameter identifies the type of request issued. Common requests include the following: 0: *Archive* 1: *Restore* 2: *Delete* 5: *Copy* 7: *Restore Instance* 8: *Delete Instance* 13: *Partial Restore* Other requests include the following: 3: *Eject Tape* 4: *Insert Tape* 6: *Copy To New* 9: *Unknown* 10: *Automatic Repack* 11: *On Demand Repack* 12: *Associative Copy* 14: *Multiple Restore* 15: *Transcode Archived* 16: *Export* 17: *Transfer Files* 18: *Automatic Verify Tapes* 19: *Manual Verify Tapes*	Integer (0 - 19)

DIVArchive Status Codes

The following are typical DIVArchive getRequestInfo() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success

The request has been created successfully.

1008: Invalid Parameter

One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied

This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST getRequestInfo() request:

```
<p:getRequestInfo xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:requestNumber>1536</p:requestNumber>
</p:getRequestInfo>
```

REST Response Example

The following is an example of a REST response for an getRequestInfo() request:

```
<getRequestInfoResponse xmlns=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p="http://response.model.api.ws.diva.fpdigital.com/xsd"
xmlns:p1="http://model.api.ws.diva.fpdigital.com/xsd">
<return>
    <p:divaStatus>1000</p:divaStatus>
    <p:divaRequestInfo>
      <p1:abortionReason>
        <p1:code>0</p1:code>
        <p1:description>NoError</p1:description>
      </p1:abortionReason>
      <p1:additionalInfo>&lt;?xml version="1.0" encoding="UTF-8"?&gt;
&lt;ADDITIONAL_INFO
xmlns="http://www.fpdigital.com/divarchive/additionalInfoRequestInfo/v1.0"&gt;
      &lt;Object&gt;
           &lt;Name&gt;Stress_223&lt;/Name&gt;
           &lt;Category&gt;1000ArchivesBCAST9&lt;/Category&gt;
           &lt;Instances&gt;
                &lt;TapeInstance&gt;
                     &lt;Id&gt;0&lt;/Id&gt;
                     &lt;Tape&gt;
                          &lt;MediaName&gt;3L0022&lt;/MediaName&gt;
                     &lt;/Tape&gt;
                &lt;/TapeInstance&gt;
           &lt;/Instances&gt;
      &lt;/Object&gt;
&lt;/ADDITIONAL_INFO&gt;</p1:additionalInfo>
      <p1:currentPriority>50</p1:currentPriority>
      <p1:objectSummary>
        <p1:objectCategory>1000ArchivesBCAST9</p1:objectCategory>
        <p1:objectName>Stress_223</p1:objectName>
      </p1:objectSummary>
      <p1:progress>100</p1:progress>
      <p1:repackTapes>
        <p1:destinationTape>FD5309</p1:destinationTape>
        <p1:sourceTape>FD5301</p1:sourceTape>
      </p1:repackTapes>
      <p1:requestNumber>1209</p1:requestNumber>
      <p1:requestState>3</p1:requestState>
      <p1:requestType>1</p1:requestType>
    </p:divaRequestInfo>
  </return>
</getRequestInfoResponse>
```

Partial File Restore Request Information: getPartialRestoreRequestInfo()

This command returns extended information about a particular Partial File Restore request. When a Partial File Restore request is executed, DIVArchive may compute and use offsets that are different than those provided in the request. This command provides a method of obtaining the offsets that were actually used for the Partial File Restore.

This command is available in DIVArchive, but not in DIVAnet (MultiDIVA mode).

Input Parameters

The following is a list of getPartialRestoreRequestInfo() request input parameters:

Table 6–19 getPartialRestoreRequestInfo() Request Input Parameters

Parameter	Description	Data Type and Defaults
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
requestNumber	This parameter identifies the request ID of the Partial File Restore request to obtain information on.	Integer (1 - 10,000,000)

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 6–20 getPartialRestoreRequestInfo() Request Returned Values

Section	Parameters	Description	Data Type and Default
	divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)
	destFile	This parameter identifies the output file name of the Partial File Restore. Relative, or absolute, paths in the file name are not allowed. If the Partial File Restore results in multiple files being generated (such as audio files), DIVArchive uses the destFile name to assign a name to the audio files.	String (1 - 4000 characters) This is required for timecode Partial File Restore.

Table 6–20 (Cont.) getPartialRestoreRequestInfo() Request Returned Values

Section	Parameters	Description	Data Type and Default
offsetVector		This section identifies the actual timecode range for the Partial File Restore. *There is no enclosing tag for all of the offsets in this section.* <offsetVector> <timeCodeBegin>00:00:00:00 </timeCodeBegin> <timeCodeEnd>00:00:10:00 </timeCodeEnd> <posType>2</posType> </offsetVector> <offsetVector> <timeCodeBegin>00:00:20:00 </timeCodeBegin> <timeCodeEnd>-1 </timeCodeEnd> <posType>2</posType> </offsetVector>	
offsetVector	timeCodeBegin	This parameter identifies the SMPTE timecode value marking the start of the extracted portion. The format is hour:minute:second:frame. For example, 00:00:04:00 to 00:10:04:00 would denote a 10 minute segment starting 4 seconds in and ending at 10 minutes and 4 seconds. *This parameter is valid only when posType=1.*	String (hour:minute:second:frame)
offsetVector	timeCodeEnd	This parameter identifies the SMPTE timecode value marking the end of the extracted portion. The format is hour:minute:second:frame. *This parameter is valid only when posType=1.*	String (hour:minute:second:frame)
offsetVector	byteBegin	This parameter identifies the starting byte in the range. The first byte starts at 0. *This parameter is valid only when posType=0.*	Integer (-1, 0 or a positive integer)
offsetVector	byteEnd	This parameter identifies the ending byte in the range. The last byte is indicated with a -1 value. *This parameter is valid only when posType=0.*	Integer (-1, 0 or a positive integer)
offsetVector	posType	This parameter identifies the type of offset in use: 0: *ByteOffset* 1: *Timecode*	Integer (0 or 1)
	sourceFile	This parameter identifies the source file name within the archived object.	String (1 - 4000 characters)

DIVArchive Status Codes

The following are typical DIVArchive getPartialRestoreRequestInfo() request status codes. *See Appendix A for all DIVArchive Status Codes.*

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST getPartialRestoreRequestInfo() request:

```
<p:getPartialRestoreRequestInfo
xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:requestNumber>1536</p:requestNumber>
</p:getPartialRestoreRequestInfo>
```

REST Response Example

The following is an example of a REST response for a getPartialRestoreRequestInfo() request:

```
<p:getPartialRestoreRequestInfoResponse
xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd
xmlns:p2=http://model.api.ws.diva.fpdigital.com/xsd
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
      <p1:fileList>
         <p2:destFile>gxf</p2:destFile>
         <p2:offsetVector>
            <p2:byteBegin xsi:nil="true"/>
            <p2:byteEnd xsi:nil="true"/>
            <p2:timeCodeBegin>00:00:40:00</p2:timeCodeBegin>
            <p2:timeCodeEnd>00:00:59:28</p2:timeCodeEnd>
            <p2:posType>1</p2:posType>
         </p2:offsetVector>
         <p2:sourceFile>horseRace.gxf</p2:sourceFile>
         <p2:fileFolder xsi:nil="true"/>
         <p2:range xsi:nil="true"/>
      </p1:fileList>
   </p:return>
</p:getPartialRestoreRequestInfoResponse>
```

Finished Requests: getFinishedRequestList()

This command returns requests that have finished in the last *n* seconds. The *n* value is the initialTime parameter. The command returns a list of requests that completed

successfully, and unsuccessfully. The returned information is similar to the output of the getRequestInfo() call. It includes the name of the archived object involved in each request, parameters conveying the status of each request, and additional information.

The command returns requests in batches. The size of each batch is governed by the maxFetch parameter. The returned parameter uniqueId holds the current position in the list. You pass the returned uniqueId value as input to the next getFinishedRequestList() call to retrieve the next batch. The end of the list is reached when an empty list of requests is returned. If you are constantly polling for finished requests, and you do not want to miss requests, you should use an initialTime that is slightly greater than how often you are polling.

Input Parameters

The following is a list of getFinishedRequestList() input parameters:

Table 6–21 getFinishedRequestList() Request Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This parameter is required.
maxFetch	This parameter identifies the maximum number of requests to retrieve in a single call. The recommended setting is 500.	Integer This parameter is required.
initialTime	This parameter indicates how far back in time to look for finished requests. This value is measured in seconds. This parameter has effect only on the initial call, and not when fetching batches of requests. On the initial call, if the number 0 is passed, the behavior is different, and all finished requests in the system will be returned.	Integer (0 - 259200; in seconds)
uniqueId	This value should be empty on the first call. If retrieving requests in batches, this parameter should be populated with the uniqueId returned from the previous call.	String

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 6–22 getFinishedRequestList() Request Returned Values

Section	Parameter	Description	Data Type and Default
	divaStatus	This parameter identifies the DIVArchive Status Code for the operation.	Integer (1000 - 1039)

Table 6–22 (Cont.) getFinishedRequestList() Request Returned Values

Section	Parameter	Description	Data Type and Default
	uniqueId	This parameter holds the position in the list when returning requests in batches. You pass this value into the next getFinishedRequestList() call to get the next batch.	String
divaRequestInfo		This section contains the request information.	Occurs only one time.
divaRequestInfo/abortionReason		This section describes the reason why the request terminated.	Occurs only one time.
divaRequestInfo/abortionReason	code	This parameter identifies a code indicating a possible reason why the request terminated as follows: 0: *None* 1: *Tape Drive* 2: *Individual Tape Issue* 3: *Actor* 4: *Disk* 5: *Disk Full* 6: *Source or Destination* 7: *Resources* 8: *Tape Library* 9: *Input Parameters* 10: *Unknown* 11: *Internal Error*	Integer (0 - 11)
divaRequestInfo/abortionReason	description	This parameter is a text representation of the abortionReason code.	String
divaRequestInfo	additionalInfo	This parameter is an unstructured XML value (encoded as a string) indicating additional information associated with the request (see the following table entries). The information here is not necessarily valid after the request completes.	

Table 6–22 (Cont.) getFinishedRequestList() Request Returned Values

Section	Parameter	Description	Data Type and Default
divaRequestInfo	currentPriority	This parameter identifies the current priority of the request on a scale of 0 to 100 (100 being the highest priority). The current priority of the request increases the longer it waits to be executed.	Integer (0 - 100)
divaRequestInfo/objectSummary		This section details the format name of the object (objectName and objectCategory).	Occurs only one time.
divaRequestInfo/objectSummary	objectName	This parameter identifies the name of the object. This parameter, along with the objectCategory, creates the full formal name of a DIVArchive object.	String (1 - 192 characters)
divaRequestInfo/objectSummary	objectCategory	This parameter identifies the object category. This parameter, along with the objectName, creates the full formal name of a DIVArchive object.	String (1 - 96 characters)
divaRequestInfo	progress	The value of this parameter indicates the percentage of the request that is complete. If multiple transfers are required to process a request (for example, if *Verify on Restore* has been selected), the percentage may reset back to 0 when performing the second transfer. A progress of 100 does not necessarily indicate that the request has completed.	Integer (-1, 0 - 100)
divaRequestInfo / repackTapes		This section only applies to Repack requests.	Occurs only one time.
divaRequestInfo / repackTapes	destinationTape	This parameter identifies (on a repack request) the destination tape to store the repacked information.	String (0 - 96 characters)

Table 6–22 (Cont.) getFinishedRequestList() Request Returned Values

Section	Parameter	Description	Data Type and Default
divaRequestInfo / repackTapes	sourceTape	This parameter identifies the fragmented tape being repacked.	String (0 - 96 characters)
divaRequestInfo	requestNumber	This parameter identifies the ID that uniquely identifies the request.	Integer
divaRequestInfo	requestState	This parameter identifies the overall state of the request. The following states indicate that the request finished execution as follows: 3: *Completed Successfully* 4: *Terminated* 5: *Canceled* 11: *Partially Terminated* The following states indicate that the request is running, or the state is unknown: 6: *Unknown* 12: *Running*	Integer (3 - 12)

Table 6–22 (Cont.) getFinishedRequestList() Request Returned Values

Section	Parameter	Description	Data Type and Default
divaRequestInfo	requestType	This parameter identifies the type of request issued. Common requests include the following: 0: *Archive* 1: *Restore* 2: *Delete* 5: *Copy* 7: *Restore Instance* 8: *Delete Instance* 13: *Partial File Restore* Other request types include the following: 3: *Eject Tape* 4: *Insert Tape* 6: *Copy To New* 9: *Unknown* 10: *Automatic Repack* 11: *On Demand Repack* 12: *Associative Copy* 14: *Multiple Restore* 15: *Transcode Archived* 16: *Export* 17: *Transfer Files* 18: *Automatic Verify Tapes* 19: *Manual Verify Tapes*	Integer (0 - 19)

DIVArchive Status Codes

The following are typical DIVArchive getFinishedRequestList() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST getFinishedRequestList() request:

```
<getFinishedRequestList
xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:maxFetch>150</p:maxFetch>
   <p:initialTime>34899</p:initialTime>
   <p:uniqueId></p:uniqueId>
</getFinishedRequestList>
```

REST Response Example

The following is an example of a REST response for an getFinishedRequestList() request:

```
<getFinishedRequestListResponse
xmlns:p="http://response.model.api.ws.diva.fpdigital.com/xsd"
xmlns:p1="http://model.api.ws.diva.fpdigital.com/xsd">
  <return>
    <p:divaStatus>1000</p:divaStatus>
    <p:uniqueId>1489760744000,372</p:uniqueId>
    <p:requestInfoList>
      <p1:abortionReason>
        <p1:code>0</p1:code>
        <p1:description>NoError</p1:description>
      </p1:abortionReason>
      <p1:additionalInfo>&lt;?xml version="1.0" encoding="UTF-8"?&gt;
&lt;ADDITIONAL_INFO
xmlns="http://www.fpdigital.com/divarchive/additionalInfoRequestInfo/v1.0"&gt;
       &lt;Object&gt;
              &lt;Name&gt;CHANNEL17_EDIT3&lt;/Name&gt;
              &lt;Category&gt;Cat&lt;/Category&gt;
              &lt;Instances&gt;
                     &lt;DiskInstance&gt;
                           &lt;Id&gt;0&lt;/Id&gt;
                            &lt;Disk&gt;
                                  &lt;MediaName&gt;disk_002_a002&lt;/MediaName&gt;
                           &lt;/Disk&gt;
                     &lt;/DiskInstance&gt;
              &lt;/Instances&gt;
       &lt;/Object&gt;
&lt;/ADDITIONAL_INFO&gt;</p1:additionalInfo>
      <p1:currentPriority>40</p1:currentPriority>
      <p1:objectSummary>
        <p1:objectCategory>Cat</p1:objectCategory>
        <p1:objectName>CHANNEL17_EDIT3</p1:objectName>
      </p1:objectSummary>
      <p1:progress>100</p1:progress>
      <p1:repackTapes>
        <p1:destinationTape> </p1:destinationTape>
        <p1:sourceTape> </p1:sourceTape>
      </p1:repackTapes>
      <p1:requestNumber>366</p1:requestNumber>
      <p1:requestState>3</p1:requestState>
      <p1:requestType>13</p1:requestType>
    </p:requestInfoList>
    <p:requestInfoList>
      <p1:abortionReason>
```

```
        <p1:code>0</p1:code>
        <p1:description>NoError</p1:description>
      </p1:abortionReason>
      <p1:additionalInfo>&lt;?xml version="1.0" encoding="UTF-8"?&gt;
&lt;ADDITIONAL_INFO
xmlns="http://www.fpdigital.com/divarchive/additionalInfoRequestInfo/v1.0"&gt;
       &lt;Object&gt;
              &lt;Name&gt;CHANNEL17_EDIT3&lt;/Name&gt;
              &lt;Category&gt;Cat&lt;/Category&gt;
              &lt;Instances&gt;
                     &lt;DiskInstance&gt;
                           &lt;Id&gt;0&lt;/Id&gt;
                           &lt;Disk&gt;
                                  &lt;MediaName&gt;disk_002_a002&lt;/MediaName&gt;
                           &lt;/Disk&gt;
                     &lt;/DiskInstance&gt;
              &lt;/Instances&gt;
       &lt;/Object&gt;
&lt;/ADDITIONAL_INFO&gt;</p1:additionalInfo>
      <p1:currentPriority>40</p1:currentPriority>
      <p1:objectSummary>
        <p1:objectCategory>Cat</p1:objectCategory>
        <p1:objectName>CHANNEL17_EDIT3</p1:objectName>
      </p1:objectSummary>
      <p1:progress>100</p1:progress>
      <p1:repackTapes>
        <p1:destinationTape> </p1:destinationTape>
        <p1:sourceTape> </p1:sourceTape>
      </p1:repackTapes>
      <p1:requestNumber>367</p1:requestNumber>
      <p1:requestState>3</p1:requestState>
      <p1:requestType>13</p1:requestType>
    </p:requestInfoList>
</return>
</getFinishedRequestListResponse>
```

Storage Plan Information: `getStoragePlanList()`

This command returns information regarding all of the configured DIVArchive Storage Plans. A DIVArchive Storage Plan is a set of configuration parameters that describe how an archived object is processed (for example, copied, deleted, and so on) during its lifetime in DIVArchive.

If you issue the command to DIVAnet, the command returns all DIVArchive Storage Plans for all sites in the DIVAnet network. The returned Storage Plan name is prefixed with the DIVAnet site name delimited with an underscore. If the `-site {sitename}` option appears in the `options` parameter, DIVAnet returns Storage Plan information for the selected site only.

Input Parameters

The following is a list of `getStoragePlanList()` request input parameters:

Table 6–23 getStoragePlanList() Request Input Parameters

Parameter	Description	Data Type and Defaults
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
options	There are no options defined in DIVArchive. In DIVAnet, the -site option indicates the site to return Storage Plans from.	

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 6–24 getStoragePlanList() Request Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter is the DIVArchive Status Code for the operation (see the following section).	Integer (1000 - 1039)
spList	This tag contains the name of the Storage Plan. The <spList> tag repeats for each Storage Plan configured in the system. For example: <spList>SP_001</spList> <spList>SP_DEFAULT</spList>	List of Strings

DIVArchive Status Codes

The following are typical DIVArchive getStoragePlanList() request status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST getStoragePlanList() request:

```
<p:getStoragePlanList xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:options></p:options>
</p:getStoragePlanList>
```

REST Response Example

The following is an example of a REST response for an getStoragePlanList() request:

```
<p:getStoragePlanListResponse
xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
<p:return>
   <p:divaStatus>1000</p:divaStatus>
   <p:spList>SP_001</spList>
   <p:spList>SP_DEFAULT</spList>
</p:return>
</p:getStoragePlanListResponse>
```

7

Miscellaneous Commands

This chapter describes additional miscellaneous DIVA Enterprise Connect commands and includes the following services:

Overview

The miscellaneous commands int his chapter can perform miscellaneous tasks such as canceling requests, adding or deleting tape groups, and locking or unlocking objects. These commands are simple request and response operations, and do not require returning information in batches, nor monitoring through the `getRequestInfo()` call.

Creating a Session: `registerClient()`

This command creates a client session. A valid client session is necessary to use the DIVArchive Web Services API. The client must provide valid authentication in this step to have a `sessionCode` returned successfully. You must supply a valid `sessionCode` in every Web Service request. This command is specific to the DIVArchive Web Services, and does not cause a call to DIVArchive or DIVAnet.

Input Parameters

The following table lists the `registerClient()` input parameters:

Table 7–1 registerClient() Command Input Parameters

Parameter	Description	Data Type and Default
appName	This parameter identifies the name of the application invoking the services.	String (1 - 96 characters) This field is required.
locName	This parameter identifies a string describing the location of the client (caller).	String (1 - 96 characters) This field is required.
processId	This parameter identifies the ID of the client process that is executing, or another unique ID that identifies the instance of the application that is executing.	String (1 - 96 characters) This field is required.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–2 registerClient() Command Return Values

Parameter	Description	Data Type and Default
divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)
sessionCode	The parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters)

DIVArchive Status Codes

The following are typical DIVArchive registerClient() command status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST registerClient() command:

```
<p:registerClient xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:appName>MediaDatabase2</p:appName>
   <p:locName>USWest2</p:locName>
   <p:processId>5828</p:processId>
</p:registerClient>
```

REST Response Example

The following is an example of a REST response for a registerClient() command:

```
<p:registerClientResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
```

```
   <p1:return>810c85c7-d70c-414e-93a4-655a3e4c89bd</p1:return>
</p:registerClientResponse>
```

Changing the Request Priority: changeRequestPriority()

This command enable you to change the priority of a running or pending request to a new value. The priority of a request that has already finished execution cannot be changed.

In DIVArchive, the effective priority of a request is constantly increased as it sits in the queue waiting to run. It is possible that if the priority of a long running request is changed (for example, from 50 to 60), the effective priority in DIVArchive may actually be decreased.

This command is available in DIVArchive. DIVAnet (MultiDIVA mode) returns success for this operation, but will not process the request.

Input Parameters

The following table lists the changeRequestPriority() input parameters:

Table 7–3 changeRequestPriority() Command Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
requestNumber	This parameter identifies the number of the request to modify.	Integer This field is required.
priorityLevel	This parameter identifies the initial priority of the request on a scale of 0 to 100 (100 is the highest priority). The effective priority of the request increases the longer it waits for execution.	Integer (0 - 100) If you use nil or -1 for this value, the DIVArchive default is used.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–4 changeRequestPriority() Command Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)

DIVArchive Status Codes

The following are typical DIVArchive changeRequestPriority() command status codes. *See Appendix A for all DIVArchive Status Codes.*

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1011: Request Does Not Exist
The request ID you issued does not exist in DIVArchive or DIVAnet.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST changeRequestPriority() command:

```
<p:changeRequestPriority
xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:requestNumber>3904</p:requestNumber>
   <p:priorityLevel>75</p:priorityLevel>
</p:changeRequestPriority>
```

REST Response Example

The following is an example of a REST response for a changeRequestPriority() command:

```
<p:changeRequestPriorityResponse
xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
   </p:return>
</p:changeRequestPriorityResponse>
```

Canceling a Request: cancelRequest()

This command enables canceling a running DIVArchive or DIVAnet request. You must supply the request ID to cancel the request.

Input Parameters

The following table lists the cancelRequest() input parameters:

Table 7–5 cancelRequest() Command Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
requestNumber	This parameter identifies the number of the DIVArchive or DIVAnet request to cancel.	Integer This field is required.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–6 cancelRequest() Command Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)

DIVArchive Status Codes

The following are typical DIVArchive cancelRequest() command status codes. There may be a delay between when a cancel request is issued, and when the request actually becomes canceled. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1011: Request Does Not Exist
The request ID you issued does not exist in DIVArchive or DIVAnet.

1012: Request Not Cancelable
The request you are attempting to cancel cannot be canceled because it is either not currently executing, or is in a state that cannot be interrupted.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST cancelRequest() command:

```
<p:cancelRequest xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:requestNumber>1678</p:requestNumber>
</p:cancelRequest>
```

REST Response Example

The following is an example of a REST response for a cancelRequest() command:

```
<p:cancelRequestResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
   </p:return>
</p:cancelRequestResponse>
```

Adding a Tape Group: addGroup()

This command enables the caller to add a tape group to a running DIVArchive system.

This command is available in DIVArchive, but not in DIVAnet (MultiDIVA mode).

Input Parameters

The following table lists the addGroup() input parameters:

Table 7–7 addGroup() Command Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
groupName	This parameter identifies the name of the group being added.	String (1 - 96 characters) This field is required.
associatedSet	This parameter identifies the DIVArchive tape set number where the group will belong. *A DIVArchive tape set is a pool of tapes with similar characteristics*.	Integer (0 - 99) this field is required.
comment	This parameter identifies the text describing the group. This text is available to users of the API who call getGroupList().	String (1 - 768 characters) This field is required.
toBeRepacked	If this parameter is true, then tapes belonging to this group are eligible for automatic repack.	Boolean (true or false) This field is required.
worstFitEnabled	If this parameter is true, then DIVArchive uses the tape with the largest amount of free space in the group to satisfy the request. Oracle recommends you set this value to false if you are unsure of which setting to use.	Boolean (true or false) This field is optional.
worstFitRepackTapes	This parameter identifies the number of tapes reserved for worst fit repacking.	Integer This field is required.
mediaFormatId	This parameter identifies the format type of the tapes in this group. This is either DIVArchive legacy format, or AXF (Archive Exchange Format). The possible integer values are as follows: 0: *Legacy* - DIVArchive Legacy format 1: *AXF* - AXF 1.0 format	Integer (0 or 1) This field is required.
verifyWriteEnabled	If this parameter is true, then the VerifyOnWrite feature is enabled when writing media to this group. If enabled, DIVArchive reads the media after it has been written, to ensure that the data was written correctly.	Boolean (true or false) This field is optional.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–8 addGroup() Command Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)

DIVArchive Status Codes

The following are typical DIVArchive addGroup() command status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1021: Group Already Exists
The requested group already exists in DIVArchive.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST addGroup() command:

```
<p:addGroup xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:groupName>LTOPOST3</p:groupName>
   <p:associatedSet>1</p:associatedSet>
   <p:comment>Raw footage for post</p:comment>
   <p:toBeRepacked>true</p:toBeRepacked>
   <p:worstFitEnabled>false</p:worstFitEnabled>
   <p:worstFitRepackTapes>1</p:worstFitRepackTapes>
   <p:mediaFormatId>1</p:mediaFormatId>
   <p:verifyWriteEnabled>true</p:verifyWriteEnabled>
</p:addGroup>
```

REST Response Example

The following is an example of a REST response for a addGroup() command:

```
<p:addGroupResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
   </p:return>
</p:addGroupResponse>
```

Deleting a Tape Group: deleteGroup()

This command deletes a DIVArchive tape group. You can only delete a tape group if there are no object instances stored in the group.

This command is available in DIVArchive, but not in DIVAnet (MultiDIVA mode).

Input Parameters

The following table lists the deleteGroup() input parameters:

Table 7–9 deleteGroup() Command Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
groupName	This parameter identifies the name of the DIVArchive tape group to be deleted.	String (1 - 96 characters) This field is required.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–10 deleteGroup() Command Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)

DIVArchive Status Codes

The following are typical DIVArchive deleteGroup() command status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1017: Group Does Not Exist
The tape group you provided does not exist in DIVArchive.

1022: Group In Use
The tape group you provided currently has object instances stored in it.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST deleteGroup() command:

```
<p:deleteGroup xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:groupName>LTOPOST3</p:groupName>
</p:deleteGroup>
```

REST Response Example

The following is an example of a REST response for a deleteGroup() command:

```
<p:deleteGroupResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
```

```
      <p1:divaStatus>1000</p1:divaStatus>
   </p:return>
</p:deleteGroupResponse>
```

Locking an Object: lockObject()

This command allows an object to become locked for restore. Objects cannot be restored while they are locked. You must invoke unlockObject() to allow the object to be restored.

This command is available in DIVArchive. DIVAnet returns success for this operation, but does not process the request.

Input Parameters

The following table lists the lockObject() input parameters:

Table 7–11 lockObject() Command Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with category, creates the unique formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
category	This parameter identifies the name of the object category. This parameter, along with objectName, creates the unique formal name of a DIVArchive object.	String (1 - 96 characters) This field is required.
options	No options are currently defined for this operation.	String (0 - 768 characters) You can leave this field empty.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–12 lockObject() Command Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)

DIVArchive Status Codes

The following are typical DIVArchive lockObject() command status codes. *See Appendix A for all DIVArchive Status Codes.*

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in the DIVArchive.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST lockObject() command:

```
<p:lockObject xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:category>PROD</p:category>
   <p:options></p:options>
</p:lockObject>
```

REST Response Example

The following is an example of a REST response for a lockObject() command:

```
<p:lockObjectResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
   </p:return>
</p:lockObjectResponse>
```

Unlocking an Object: unlockObject()

This command allows the unlocking of an object previously locked for restore. Objects cannot be restored if they are locked. DIVArchive returns success for the operation even if the object is not currently locked. You must invoke lockObject() to lock the object from restore.

This command is available in DIVArchive. DIVAnet returns success for this operation, but does not process the request.

Input Parameters

The following table lists the unlockObject() input parameters:

Table 7–13 unlockObject() Command Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with category, creates the unique formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
category	This parameter identifies the name of the object category. This parameter, along with objectName, creates the unique formal name of a DIVArchive object.	String (1 - 96 characters) This field is required.

Table 7–13 (Cont.) unlockObject() Command Input Parameters

Parameter	Description	Data Type and Default
options	No options are currently defined for this operation.	String (0 - 768 characters) You can leave this field empty.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–14 unlockObject() Command Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)

DIVArchive Status Codes

The following are typical DIVArchive unlockObject() command status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in the DIVArchive.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST unlockObject() command:

```
<p:unlockObject xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName>PREPROD_July_Edits4</p:objectName>
   <p:category>PROD</p:category>
   <p:options></p:options>
</p:unlockObject>
```

REST Response Example

The following is an example of a REST response for a unlockObject() command:

```
<p:unlockObjectResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
   </p:return>
</p:unlockObjectResponse>
```

Linking Objects: linkObjects()

This command links DIVArchive archived objects together, creating a parent-child relationship. The parent and child objects must both exist in DIVArchive before performing the link operation. In addition to maintaining this relationship in DIVArchive, additional options are available to delete the child object if the parent object is deleted, and to restore the child object if the parent object is restored.

This command is available in DIVArchive. DIVAnet returns success for this operation, but does not process the request. There is no unlink operation available in the DIVArchive Web Services API.

Input Parameters

The following table lists the linkObjects() input parameters:

Table 7–15 linkObjects() Command Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
parentName	This parameter identifies the name of the parent object. This parameter, along with parentCategory, creates the unique formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
parentCategory	This parameter identifies the name of the parent object category. This parameter, along with parentName, creates the unique formal name of a DIVArchive object.	String (1 - 96 characters) This field is required.
childName	This parameter identifies the name of the child object. This parameter, along with childCategory, creates the unique formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
childCategory	This parameter identifies the name of the child object category. This parameter, along with childName, creates the unique formal name of a DIVArchive object.	String (1 - 96 characters) This field is required.
cascadeDelete	The value of this parameter is true if the child object should be automatically deleted when the parent object is deleted.	Boolean (true or false) This field is required.
cascadeRestore	The value of this parameter is true if the child object should be automatically restored when the parent object is restored.	Boolean (true or false) This field is required.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–16 linkObjects() Command Return Values

Parameter	Description	Data Type and Default
divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)
requestNumber	The parameter identifies the ID that uniquely identifies the new running request. You can monitor the progress of the request by using this ID.	Integer (1 - 10,000,000)

DIVArchive Status Codes

The following are typical DIVArchive linkObjects() command status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The selected objects have been linked.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
An object with the name provided does not exist in the DIVArchive system.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST linkObjects() command:

```
<p:linkObjects xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:parentName>PREPROD_July_Edits4</p:parentName>
   <p:parentCategory>PROD</p:parentCategory>
   <p:childName>PREPROD_July_Metadata4</p:childName>
   <p:childCategory>PRODMeta</p:childCategory>
   <p:cascadeDelete>true</p:cascadeDelete>
   <p:cascadeRestore>false</p:cascadeRestore>
</p:linkObjects>
```

REST Response Example

The following is an example of a REST response for a linkObjects() command:

```
<p:linkObjectsResponse xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
   </p:return>
</p:linkObjectsResponse>
```

Marking an Object Instance Required: require()

This command sets a flag in the DIVArchive database indicating that an object instance is required to be present in the tape library, and therefore should not be externalized. You can query this flag can be queried through the DIVArchive Control GUI. If the instance has already been marked as required, this call has no effect. The flag is informational only, and is not used to prevent or allow tape operations.

This command is available in DIVArchive, but not in DIVAnet (MultiDIVA mode).

Input Parameters

The following table lists the require() input parameters:

Table 7–17 require() Command Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with categoryName, creates the unique formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
categoryName	This parameter identifies the name of the object category. This parameter, along with objectName, creates the unique formal name of a DIVArchive object.	String (1 - 96 characters) This field is required.
instanceID	This parameter identifies the tape instance to mark as required. If this field is empty, or the value -1 is passed, all instances of the object are marked as required.	String (-1, or 0 - 100000)

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–18 require() Command Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)

DIVArchive Status Codes

The following are typical DIVArchive require() command status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
The request has been created successfully.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
The object name and category pair provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1027: Instance Does Not Exist
The specified instance ID does not exist in DIVArchive.

1029: Instance Must Be On Tape
The specified instance must be a tape instance.

1030: No Tape Instance Exists
There is no valid tape instance for the object in DIVArchive

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST `require()` command:

```
<p:require xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName> PREPROD_July_Edits4</p:objectName>
   <p:categoryName>PROD</p:categoryName>
   <p:instanceID>1</p:instanceID>
</p:require>
```

REST Response Example

The following is an example of a REST response for a `require()` command:

```
<p:require xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
   </p:return>
</p:require>
```

Marking an Object Instance Releasable: `release()`

This command sets a flag in the DIVArchive database indicating that an object instance is available to be externalized. You can query this flag through the DIVArchive Control GUI. If the instance has already been released, this call has no effect. The flag is informational only, and is not used to prevent or allow tape operations.

This command is available in DIVArchive, but not in DIVAnet (MultiDIVA mode).

Input Parameters

The following table lists the `release()` input parameters:

Table 7–19 release() Command Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
objectName	This parameter identifies the name of the object. This parameter, along with categoryName, creates the unique formal name of a DIVArchive object.	String (1 - 192 characters) This field is required.
categoryName	This parameter identifies the name of the object category. This parameter, along with objectName, creates the unique formal name of a DIVArchive object.	String (1 - 96 characters) This field is required.
instanceID	This parameter identifies the tape instance to mark as released. If this field is empty, or the value -1 is passed, all instances of the object are marked as released.	String (-1, or 0 - 100000)

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–20 release() Command Returned Values

Parameter	Description	Data Type and Default
divaStatus	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)

DIVArchive Status Codes

The following are typical DIVArchive release() command status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
There may be some delay between receiving this code and when the request is actually canceled.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1009: Object Does Not Exist
The object name and category pair provided does not exist in DIVArchive.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1027: Instance Does Not Exist
The specified instance ID does not exist in DIVArchive.

1029: Instance Must Be On Tape
The specified instance must be a tape instance.

1030: No Tape Instance Exists
There is no valid tape instance for the object in DIVArchive

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST release() command:

```
<p:release xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:objectName> PREPROD_July_Edits4</p:objectName>
   <p:categoryName>PROD</p:categoryName>
   <p:instanceID>1</p:instanceID>
</p:release>
```

REST Response Example

The following is an example of a REST response for a release() command:

```
<p:release xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
   </p:return>
</p:release>
```

Enabling Automatic Repack: enableAutomaticRepack()

This command enables or disables the scheduled automatic repack service within DIVArchive. When enabled, DIVArchive uses the repack schedule and thresholds defined in the DIVArchive configuration. When disabled, the service is not executed.

This command is available in DIVArchive, but not in DIVAnet (MultiDIVA mode).

Input Parameters

The following table lists the enableAutomaticRepack() input parameters:

Table 7–21 enableAutomaticRepack() Command Input Parameters

Parameter	Description	Data Type and Default
sessionCode	This parameter identifies the unique ID associated with the client's session.	String (1 - 36 characters) This field is required.
enable	If you set this to true, then automatic repack is enabled for execution in DIVArchive.	Boolean (true or false) This field is required.

Returned Values

If a response to the request was returned by DIVArchive or DIVAnet, HTTP status code 200 will be returned, and will include the following values:

Table 7–22 `enableAutomaticRepack()` Command Returned Values

Parameter	Description	Data Type and Default
`divaStatus`	This parameter identifies the DIVArchive status code for the operation.	Integer (1000 - 1039)

DIVArchive Status Codes

The following are typical DIVArchive `enableAutomaticRepack()` command status codes. *See Appendix A for all DIVArchive Status Codes*.

1000: Success
Automatic repack has been enabled or disabled.

1008: Invalid Parameter
One or more parameters contain data that cannot be parsed correctly.

1035: Access Denied
This code is returned if the client does not have permission for the request, or the command has been disallowed.

REST Request Example

The following is an example of a REST `enableAutomaticRepack()` command:

```
<p:enableAutomaticRepack
xmlns:p="http://interaction.api.ws.diva.fpdigital.com/xsd">
   <p:sessionCode>810c85c7-d70c-414e-93a4-655a3e4c89bd</p:sessionCode>
   <p:enable>true</p:enable>
</p:enableAutomaticRepack>
```

REST Response Example

The following is an example of a REST response for a `enableAutomaticRepack()` command:

```
<p:enableAutomaticRepackResponse
xmlns:p=http://interaction.api.ws.diva.fpdigital.com/xsd
xmlns:p1=http://response.model.api.ws.diva.fpdigital.com/xsd>
   <p:return>
      <p1:divaStatus>1000</p1:divaStatus>
   </p:return>
</p:enableAutomaticRepackResponse>
```

Deprecated Commands

The following commands are deprecated:

`InsertTapeShort()`
You can achieve the same result calling `insertTape()`, and omit the `acsId` and `capId` tags entirely. Alternatively, you can set the `acsId` and `capId` parameters to `-1`.

`addGroupShort()`
You can achieve the same result calling `addGroup()`, and omitting the `verifyWriteEnabled` and `worstFitEnabled` tags entirely. Alternatively, you can set both parameters to `false`.

deleteInstanceByMediaName()

You can achieve the same result calling `deleteInstance()`, populating the `mediaName` field, and setting `instanceID` value to `-1`.

A

DIVA Enterprise Connect Status Codes

This appendix describes the DIVArchive API status codes that can appear in an architected web services response. Typically, these codes are returned by either DIVArchive or DIVAnet as the result of a request.

1000: Success
The command has completed successfully, or a request has been created.

1001: Unknown
An unknown error has occurred.

1002: Internal Error
An internal error has been detected. This may be a transient error condition.

1003: DIVA Unavailable
The DIVArchive system, or DIVAnet system, or both are currently unavailable.

1004: Broken Connection
The connection between DIVArchive, or DIVAnet, or both, has been broken. Retrying the request may clear this issue.

1005: Disconnecting
The connection between DIVArchive, or DIVAnet, and the Web Services is currently disconnecting. Retrying the request may clear this issue.

1006: Already Connected
An internal error has occurred.

1007: Wrong Version
The web services are connecting to DIVArchive or DIVAnet with the wrong release level. There is a configuration parameter available to set the release level.

1008: Invalid Parameter
One or more parameters contains data that cannot be parsed correctly. Depending on the error, there may be additional information returned that can help you to isolate the issue.

1009: Object Does Not Exist
An object with the provided name does not exist in the DIVArchive system.

1010: Name Matches Multiple Objects
More than one object with the specified name exists in the DIVArchive or DIVAnet database.

1011: Request Does Not Exist
The request ID you have issued does not exist in DIVArchive or DIVAnet.

1012: Request Not Cancelable
The request you are trying to cancel cannot be canceled. This could be because it is not currently executing, or it is in a state that cannot be interrupted.

1013: System Idle
The DIVArchive, or DIVAnet, system is currently in an idle state and unable to process requests.

1014: Wrong List Size
The list size provided in the call is not within the valid range.

1015: List Not Initialized
This is not currently in use.

1016: Object Already Exists
An object with the name provided already exists in the DIVArchive or DIVAnet system.

1017: Group Does Not Exist
The tape group you have provided does not exist in the target DIVArchive system.

1018: Source/Destination Does Not Exist
The provided Source/Destination name does not exist in the DIVArchive configuration.

1019: No More Objects
The end of the list has been reached on calls that use batching to return large data sets. This is not an error condition.

1020: Not Connected
There is a connection issue between the DIVArchive Web Services and DIVArchive, or DIVAnet. This is possibly a configuration issue.

1021: Group Already Exists
The tape group provided already exists in the target DIVArchive system.

1022: Group In Use
An attempt was made to delete a tape group that has at least one object instance. You must confirm that no objects are using the tape group before deletion.

1023: Object Offline
There are no available instances of the object, possibly because a required tape has been ejected from a tape library.

1024: Timeout
The request has experienced a timeout. Retries may, or may not, have been attempted.

1025: Cannot Delete Last Instance
An attempt was made to delete the last remaining object instance in DIVArchive or DIVAnet. You must submit a Delete Object message to remove the last instance. You must submit either a Global Delete, or Site Delete, in DIVAnet.

1026: Invalid Path
The specified path does not exist in the DIVArchive system.

1027: Instance Does Not Exist
The specified object instance does not exist in DIVArchive or DIVAnet.

1028: Instance Offline
One or more object instances are currently in use and cannot be processed.

1029: Instance Must Be On Tape
The specified instance must be a tape instance.

1030: No Tape Instance Exists
No valid tape instance exists for the operation requested.

1031: Object In Use
One or more objects is currently being used and cannot be processed.

1032: Cannot Accept More Requests
The DIVArchive or DIVAnet system temporarily cannot accept any more requests. You can try the request again later.

1034: Invalid Barcode
The specified tape barcode is invalid.

1035: Access Denied
Returned if the client does not have permission for the request, or the command has been disallowed.

1036: Object Partially Deleted
The delete operation did not complete successfully for all object instances either in DIVArchive, or DIVArchive sites in DIVAnet.

1038: File/Folder Not Found
The specified file, or folder, or both, was not found.

1039: Object Is Locked
The specified operation cannot be performed because the specified object is currently locked.