IslandoraFedoraAPI

1.0

Generated by Doxygen 1.7.6.1

Thu Jun 21 2012 17:44:02

Contents

Chapter 1

Todo List

Member AbstractFedoraDatastream ($mime)

test if this covers all cases.

Member AbstractRepository (array $search)

Flesh out the function definition for this.

Member CurlConnection ($url, $type= 'none', $data=NULL, $content_type=NULL)

Test this for posting general form data. (Other then files.)

Member FedoraApiM ($params=array())

This function is a problem in Fedora $<$ 3.5 where ownerId does not properly get set. `https://jira.duraspace.org/browse/FCREPO-963`. We should deal with this.

Member FedoraDatastream ($function, $value)

maybe add functionality to set it to auto

Member FedoraDatastream ($function, $value)

We should perhaps cache this? depending on size?

Class FedoraRelationships

potentially we should validate the predicate URI

Member FedoraRepository ($id=NULL)

validate the ID

catch the getNextPid errors

Member FedoraRepository ($id)

perhaps we should check if an object exists instead of catching the exception

Member FedoraRepository (NewFedoraObject &$object)

error handling

Member FoxmlDocument ()

Implement fedora upload function to allow strings to be added as managed datastreams

Member HttpConnection ($url, $type= 'none', $data=NULL, $content_type=NULL)

Test this for posting general form data. (Other then files.)

Member NewFedoraDatastream ($id, $control_group, AbstractFedoraObject $object, FedoraRepository $repository)

test for valid identifiers. it can't start with a number etc.

Member NewFedoraDatastream ($function, $value)

this should be refined a bit

Class SimpleCache

Replace this with something more interesting like memcached

Try some other intersting caching strategies like LRU.

Chapter 2

Class Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

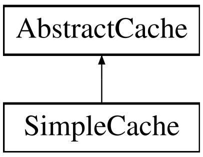
Chapter 5

Class Documentation

5.1 AbstractCache Class Reference

Simple abstract Cache defintion providing basic key value caching functionality.

Inheritance diagram for AbstractCache:



Public Member Functions

- add ($key, $data)

  Add data to the cache.
- get ($key)

  Retrieve data from the cache.
- set ($key, $data)

  Set data in the cache.
- delete ($key)

  Delete key from the cache.

5.1.1 Detailed Description

Simple abstract Cache defintion providing basic key value caching functionality.

5.1.2 Member Function Documentation

5.1.2.1 AbstractCache::add ($ *key*, $ *data*) `[abstract]`

Add data to the cache.

Parameters

string	*$key*	The key to add to the cache.
mixed	*$data*	The data to store with the key.

Returns

> boolean TRUE if the data was added to the cache. FALSE if $key already exists in the cache or if there was an error.

Reimplemented in SimpleCache.

5.1.2.2 AbstractCache::delete ($ *key*) `[abstract]`

Delete key from the cache.

Parameters

string	*$key*	The key to delete.

Returns

> boolean TRUE if they key existed and was deleted. FALSE otherwise.

Reimplemented in SimpleCache.

5.1.2.3 AbstractCache::get ($ *key*) `[abstract]`

Retrieve data from the cache.

Parameters

string	*$key*	They key to retrieve from the cache.

Returns

> mixed FALSE if the data wasn't found in the cache. Otherwise it returns the data assoctiated with the key.

Reimplemented in SimpleCache.

5.1.2.4 AbstractCache::set ($ *key,* $ *data*) [abstract]

Set data in the cache.

This will create new keys if they don't already exist, or update existing keys.

Parameters

string	*$key*	The key to add/update.
mixed	*$data*	The data to store with the key.

Returns

boolean TRUE on success. FALSE on failure.

Reimplemented in SimpleCache.

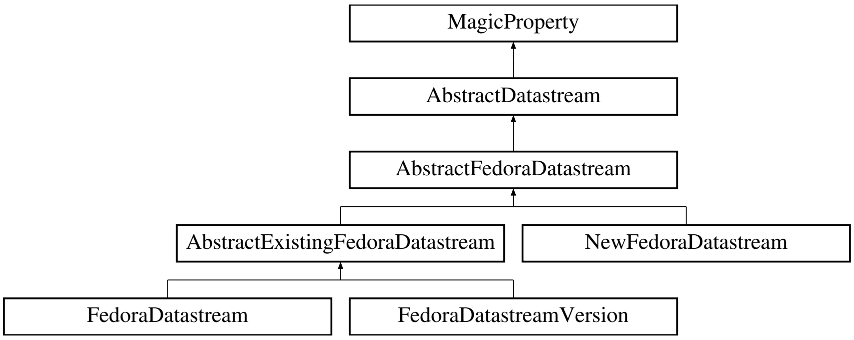
The documentation for this class was generated from the following file:

- Cache.php

5.2 AbstractDatastream Class Reference

This abstract class can be overriden by anything implementing a datastream.

Inheritance diagram for AbstractDatastream:



Public Member Functions

- delete ()

  This will set the state of the datastream to deleted.
- setContentFromFile ($file)

  Set the contents of the datastream from a file.
- setContentFromUrl ($url)

  Set the contents of the datastream from a URL.
- setContentFromString ($string)

  Set the contents of the datastream from a string.

Public Attributes

- **$id**
- **$label**
- **$controlGroup**
- **$versionable**
- **$state**
- **$mimetype**
- **$format**
- **$size**
- **$checksum**
- **$checksumType**
- **$createdDate**
- **$content**
- **$url**

5.2.1 Detailed Description

This abstract class can be overriden by anything implementing a datastream.

5.2.2 Member Function Documentation

5.2.2.1 AbstractDatastream::setContentFromFile ($ *file*) [abstract]

Set the contents of the datastream from a file.

Parameters

string	*$file*	The full path of the file to set to the contents of the datastream.

Reimplemented in FedoraDatastream, FedoraDatastreamVersion, and NewFedoraDatastream.

5.2.2.2 AbstractDatastream::setContentFromString ($ *string*) [abstract]

Set the contents of the datastream from a string.

Parameters

string	*$string*	The string whose contents will become the contents of the datastream.

Reimplemented in FedoraDatastream, FedoraDatastreamVersion, and NewFedoraDatastream.

5.2.2.3 AbstractDatastream::setContentFromUrl ($ *url*) [abstract]

Set the contents of the datastream from a URL.

The contents of this URL will be fetched, and the datastream will be updated to contain the contents of the URL.

Parameters

string	*$url*	The full URL to fetch.

Reimplemented in FedoraDatastream, FedoraDatastreamVersion, and NewFedoraDatastream.

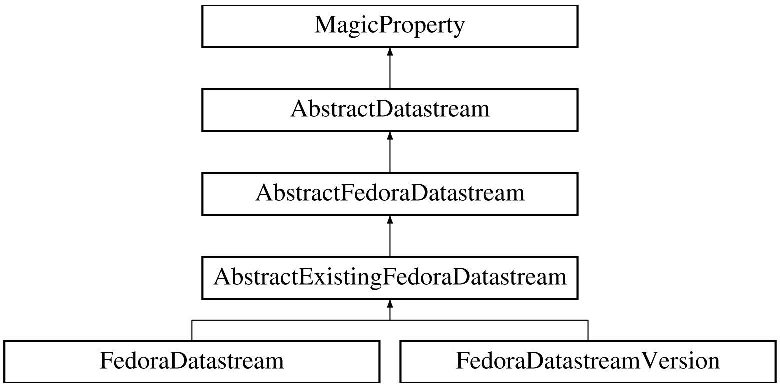
The documentation for this class was generated from the following file:

- Datastream.php

5.3 AbstractExistingFedoraDatastream Class Reference

This abstract class defines some shared functionality between all classes that implement exising fedora datastreams.

Inheritance diagram for AbstractExistingFedoraDatastream:



Public Member Functions

- __construct ($id, FedoraObject $object, FedoraRepository $repository)
  Class constructor.

Protected Member Functions

- getDatastreamContent ($version=NULL)
  Wrapper for the APIA getDatastreamDissemination function.

- getDatastreamHistory ()

    Wrapper around the APIM getDatastreamHistory function.
- modifyDatastream (array $args)

    Wrapper around the APIM modifyDatastream function.
- purgeDatastream ($version)

    Wrapper around the APIM Purge function.

5.3.1 Detailed Description

This abstract class defines some shared functionality between all classes that implement exising fedora datastreams.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 AbstractExistingFedoraDatastream::__construct ($ *id,* FedoraObject $ *object,* FedoraRepository $ *repository*)

Class constructor.

Parameters

string	*$id*	Unique identifier for the DS.
FedoraObject	*$object*	The FedoraObject that this DS belongs to.
FedoraRepository	*$repository*	The FedoraRepository that this DS belongs to.

5.3.3 Member Function Documentation

5.3.3.1 AbstractExistingFedoraDatastream::getDatastreamContent ($ *version =* `NULL`) `[protected]`

Wrapper for the APIA getDatastreamDissemination function.

Parameters

string	*$version*	The version fo the content to retreve.

Returns

string String containing the content.

Reimplemented in FedoraDatastream.

5.3.3.2 AbstractExistingFedoraDatastream::getDatastreamHistory ( ) [protected]

Wrapper around the APIM getDatastreamHistory function.

Returns

array Array containing datastream history.

5.3.3.3 AbstractExistingFedoraDatastream::modifyDatastream (array $ *args*) [protected]

Wrapper around the APIM modifyDatastream function.

Parameters

array	*$args*	Args to pass to the function.

Returns

array Datastream history array.

Reimplemented in FedoraDatastream.

5.3.3.4 AbstractExistingFedoraDatastream::purgeDatastream ($ *version*) [protected]

Wrapper around the APIM Purge function.

Parameters

string	*$version*	The version to purge.

Returns

array The versions purged.

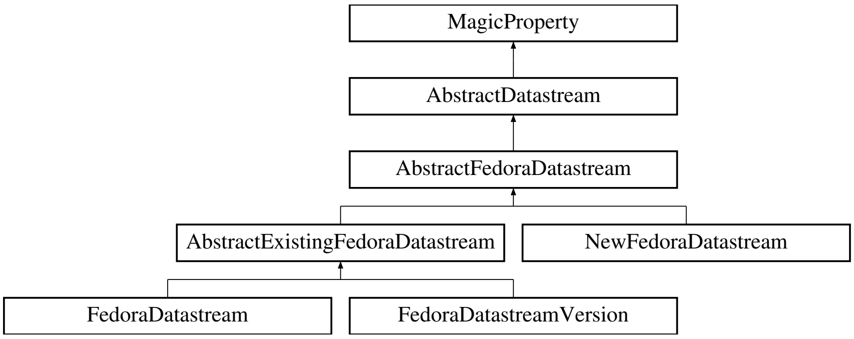
The documentation for this class was generated from the following file:

- Datastream.php

5.4 AbstractFedoraDatastream Class Reference

Abstract base class implementing a datastream in Fedora.

Inheritance diagram for AbstractFedoraDatastream:



Public Member Functions

- __construct ($id, AbstractFedoraObject $object, FedoraRepository $repository)

  The constructor for the datastream.
- delete ()

Public Attributes

- **$repository**
- **$parent**
- **$relationships**

Protected Member Functions

- idMagicProperty ($function, $value)
- isDatastreamProperySet ($actual, $unsetval)

  This is a replacement for isset when things can't be unset.
- validateMimetype ($mime)

  Validates a mimetype using a regular expression.
- validateState ($state)

  Validates and normalizes the datastream state.
- validateVersionable ($versionable)

  Validates the versionable setting of a datastream.
- validateChecksumType ($type)

  Validates and normalizes the checksumType arguement.
- controlGroupMagicProperty ($function, $value)

Protected Attributes

- **$datastreamId** = NULL
- **$datastreamInfo** = NULL

5.4.1 Detailed Description

Abstract base class implementing a datastream in Fedora.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 AbstractFedoraDatastream::__construct ($ *id*, AbstractFedoraObject $ *object*, FedoraRepository $ *repository*)

The constructor for the datastream.

Parameters

string	*$id*	The identifier of the datastream.

5.4.3 Member Function Documentation

5.4.3.1 AbstractFedoraDatastream::controlGroupMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::controlGroup

Reimplemented in FedoraDatastream, and NewFedoraDatastream.

5.4.3.2 AbstractFedoraDatastream::delete ( )

See also

AbstractDatastream::delete()

Reimplemented from AbstractDatastream.

5.4.3.3 AbstractFedoraDatastream::idMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::id

5.4.3.4 AbstractFedoraDatastream::isDatastreamProperySet ($ *actual*, $ *unsetval*) [protected]

This is a replacement for isset when things can't be unset.

So we define a default value, then return TRUE or FALSE based on if it is set.

Parameters

anything	*$actual*	The value we are testing.
anything	*$unsetval*	The value it would be if it was unset.

Returns

boolean TRUE or FALSE

5.4.3.5 AbstractFedoraDatastream::validateChecksumType ($ *type*) [protected]

Validates and normalizes the checksumType arguement.

Parameters

string	*$type*	The input string

Returns

mixed FALSE if validation fails. The checksumType string otherwise.

5.4.3.6 AbstractFedoraDatastream::validateMimetype ($ *mime*) [protected]

Validates a mimetype using a regular expression.

Parameters

string	*$mime*	A string representing a mimetype

Returns

boolean TRUE if the string looks like a mimetype.

Todo test if this covers all cases.

5.4.3.7 AbstractFedoraDatastream::validateState ($ *state*) [protected]

Validates and normalizes the datastream state.

Parameters

string	*$state*	The input state

Returns

string Returns FALSE if validation fails, otherwise it returns the normalized datastream state.

5.4.3.8 AbstractFedoraDatastream::validateVersionable ($ *versionable*) [protected]

Validates the versionable setting of a datastream.

Parameters

mixed	*$versionable*	The input versionable arguement.

Returns

boolean Returns TRUE if the arguement is a boolean, FALSE otherwise.

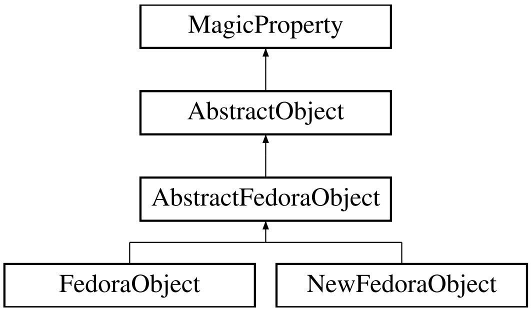
The documentation for this class was generated from the following file:

- Datastream.php

5.5 AbstractFedoraObject Class Reference

This is the base class for a Fedora Object.

Inheritance diagram for AbstractFedoraObject:



Public Member Functions

- __construct ($id, FedoraRepository $repository)

  Constructosaurus.
- delete ()
- constructDatastream ($id, $control_group= 'M')

Public Attributes

- **$relationships**
- **$repository**

Protected Member Functions

- idMagicProperty ($function, $value)
- stateMagicProperty ($function, $value)
- labelMagicProperty ($function, $value)
- ownerMagicProperty ($function, $value)

Protected Attributes

- **$objectId**
- **$objectProfile**

5.5.1 Detailed Description

This is the base class for a Fedora Object.

5.5.2 Member Function Documentation

5.5.2.1 AbstractFedoraObject::constructDatastream ($ *id,* $ *control_group* = 'M')

See also

AbstractObject::constructDatastream()

Reimplemented from AbstractObject.

Reimplemented in FedoraObject, and NewFedoraObject.

5.5.2.2 AbstractFedoraObject::delete ( )

See also

AbstractObject::delete()

Reimplemented from AbstractObject.

5.5.2.3 AbstractFedoraObject::idMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractObject::id

5.5.2.4 AbstractFedoraObject::labelMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractObject::label

Reimplemented in FedoraObject.

5.5.2.5 AbstractFedoraObject::ownerMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractObject::owner

Reimplemented in FedoraObject.

5.5.2.6 AbstractFedoraObject::stateMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractObject::state

Reimplemented in FedoraObject.

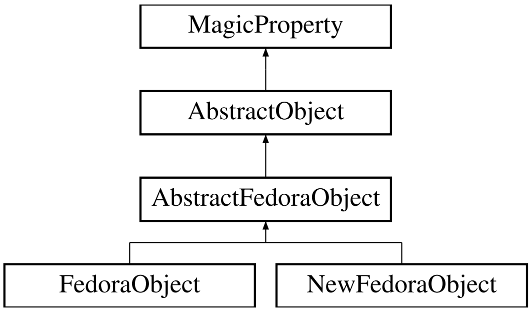
The documentation for this class was generated from the following file:

- Object.php

5.6 AbstractObject Class Reference

An abstract class defining a Object in the repository.

Inheritance diagram for AbstractObject:



Public Member Functions

- delete ()

  Set the state of the object to deleted.
- getDatastream ($id)

  Get a datastream from the object.
- purgeDatastream ($id)

  Purges a datastream.
- constructDatastream ($id, $control_group= 'M')

  Factory to create new datastream objects.
- ingestDatastream (&$ds)

  Ingests a datastream object into the repository.

Public Attributes

- **$label**
- **$owner**
- **$state**
- **$id**
- **$createdDate**
- **$lastModifiedDate**

5.6.1 Detailed Description

An abstract class defining a Object in the repository.

This is the class that needs to be implemented in order to create new repository backends that can be accessed using Tuque.

These classes implement the php object array interfaces so that the object can be accessed as an array. This provides access to datastreams. The object is also traversable with foreach, so that each datastream can be accessed.

```
$object = new AbstractObject()

// access every object
foreach($object as $dsid => $dsObject) {
  // print dsid and set contents to "foo"
  print($dsid);
  $dsObject->content = 'foo';
}

// test if there is a datastream called 'DC'
if(isset($object['DC'])) {
  // if there is print its contents
  print($object['DC']->content);
}
```

5.6.2 Member Function Documentation

5.6.2.1 AbstractObject::constructDatastream ($ *id*, $ *control_group* = 'M') [abstract]

Factory to create new datastream objects.

Creates a new datastream object, this object is not ingested into the repository until you call ingestDatastream.

Parameters

string	*$id*	The identifier of the new datastream.
string	*$control_-group*	The control group the new datastream will be created in.

Returns

AbstractDatastream Returns an instantiated Datastream object.

Reimplemented in FedoraObject, NewFedoraObject, and AbstractFedoraObject.

5.6.2.2 AbstractObject::getDatastream ($ *id*) [abstract]

Get a datastream from the object.

Parameters

string	*$id*	The id of the datastream to retreve.

Returns

AbstractDatastream Returns FALSE if the datastream could not be found. - Otherwise it return an instantiated Datastream object.

Reimplemented in FedoraObject, and NewFedoraObject.

5.6.2.3 AbstractObject::purgeDatastream ($ *id*) [abstract]

Purges a datastream.

Parameters

string	*$id*	The id of the datastream to purge.

Returns

boolean TRUE on success. FALSE on failure.

Reimplemented in FedoraObject, and NewFedoraObject.

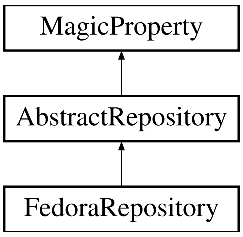
The documentation for this class was generated from the following file:

- Object.php

5.7 AbstractRepository Class Reference

An abstract repository interface.

Inheritance diagram for AbstractRepository:



Public Member Functions

- constructObject ($id=NULL)

  This method is a factory that will return a new repositoryobject object that can be manipulated and then ingested into the repository.
- ingestObject (NewFedoraObject &$object)

  This ingests a new object into the repository.
- getObject ($id)

  Gets a object from the repository.
- purgeObject ($id)

  Removes an object from the repository.
- findObjects (array $search)

  Search the repository for objects.

5.7.1 Detailed Description

An abstract repository interface.

This can be used to override the implementation of the Repository.

5.7.2 Member Function Documentation

5.7.2.1 AbstractRepository::constructObject ($ *id* = `NULL`) `[abstract]`

This method is a factory that will return a new repositoryobject object that can be manipulated and then ingested into the repository.

Parameters

string	*$id*	The ID to assign to this object. There are three options: • NULL: An ID will be assigned. • A namespace: An ID will be assigned in this namespace. • A whole ID: The whole ID must contains a namespace and a identifier in the form NAMESPACE:IDENTIFIER

Returns

AbstractObject Returns an instantiated AbstractObject object that can be manipulated. This object will not actually be created in the repository until the ingest method is called.

Reimplemented in FedoraRepository.

5.7.2.2 AbstractRepository::findObjects (array $ *search*) `[abstract]`

Search the repository for objects.

This function isn't implemented yet.

Todo Flesh out the function definition for this.

Reimplemented in FedoraRepository.

5.7.2.3 AbstractRepository::getObject ($ *id*) `[abstract]`

Gets a object from the repository.

Parameters

string	*$id*	The identifier of the object.

Returns

AbstractObject The requested object.

Reimplemented in FedoraRepository.

5.7.2.4 AbstractRepository::ingestObject (NewFedoraObject &$ *object*) [abstract]

This ingests a new object into the repository.

Parameters

- Abstract-Object	*&$object*	The instantiated AbstractObject to ingest into the repository. This object is passed by reference, and the reference will be replaced by an object representing the ingested AbstractObject.

Returns

AbstractObject The ingested abstract object.

Reimplemented in FedoraRepository.

5.7.2.5 AbstractRepository::purgeObject ($ *id*) [abstract]

Removes an object from the repository.

This function removes an object from the repository premententy. It is a dangerous function since it remvoes an object and all of its history from the repository permenently.

Parameters

string	*$id*	The identifier of the object.

Returns

boolean TRUE if object was purged.

Reimplemented in FedoraRepository.

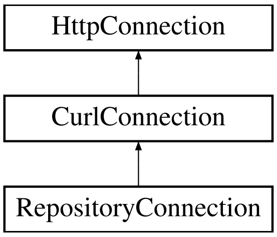
The documentation for this class was generated from the following file:

- Repository.php

5.8 CurlConnection Class Reference

This class defines a abstract HttpConnection using the PHP cURL library.

Inheritance diagram for CurlConnection:



Public Member Functions

- __construct ()

  Constructor for the connection.
- __destruct ()

  Destructor for the connection.
- postRequest ($url, $type= 'none', $data=NULL, $content_type=NULL)

  Post a request to the server.
- putRequest ($url, $type= 'none', $file=NULL)

  Send a HTTP PUT request to URL.
- getRequest ($url)

  Send a HTTP GET request to URL.
- deleteRequest ($url)

  Send a HTTP DELETE request to URL.

Public Attributes

- const **COOKIE_LOCATION** = 'curl_cookie'

Protected Member Functions

- getCurlContext ()

  This function sets up the context for curl.
- unallocateCurlContext ()

  Unallocate curl context.
- setupCurlContext ($url)

  This sets the curl options.
- doCurlRequest ()

  This function actually does the cURL request.

Protected Attributes

- **$cookieFile** = NULL
- **$curlContext** = NULL

5.8.1 Detailed Description

This class defines a abstract HttpConnection using the PHP cURL library.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 CurlConnection::__construct ( )

Constructor for the connection.

Exceptions

HttpConnectionException	

5.8.3 Member Function Documentation

5.8.3.1 CurlConnection::deleteRequest ($ *url*)

Send a HTTP DELETE request to URL.

Parameters

string	*$url*	The URL to post the request to. Should start with the protocol. For example: `http://`.

Exceptions

HttpConnectionException	

Returns

array Associative array containing: ∗ $return['status'] = The HTTP status code ∗ $return['headers'] = The HTTP headers of the reply ∗ $return['content'] = The body of the HTTP reply

Reimplemented in RepositoryConnection.

5.8.3.2 CurlConnection::doCurlRequest ( ) `[protected]`

This function actually does the cURL request.

It is a private function meant to be called by the public get, post and put methods.

Exceptions

HttpConnectionException	

Returns

array Array has keys: (status, headers, content).

5.8.3.3 CurlConnection::getRequest ($ *url*)

Send a HTTP GET request to URL.

Parameters

string	*$url*	The URL to post the request to. Should start with the protocol. For example: `http://`.

Exceptions

HttpConnectionException	

Returns

array Associative array containing: ∗ $return['status'] = The HTTP status code ∗ $return['headers'] = The HTTP headers of the reply ∗ $return['content'] = The body of the HTTP reply

Reimplemented from HttpConnection.

Reimplemented in RepositoryConnection.

5.8.3.4 CurlConnection::postRequest ($ *url*, $ *type* = 'none', $ *data* = NULL, $ *content_type* = NULL)

Post a request to the server.

This is primarily used for sending files.

Todo Test this for posting general form data. (Other then files.)

Parameters

string	*$url*	The URL to post the request to. Should start with the protocol. For example: `http://`.
string	*$type*	This paramerter must be one of: string, file.
string	*$data*	What this parameter contains is decided by the $type parameter.

Exceptions

HttpConnectionException	

Returns

array Associative array containing: ∗ $return['status'] = The HTTP status code ∗ $return['headers'] = The HTTP headers of the reply ∗ $return['content'] = The body of the HTTP reply

Reimplemented from HttpConnection.

Reimplemented in RepositoryConnection.

5.8.3.5 CurlConnection::putRequest ($ *url*, $ *type* = 'none', $ *file* = NULL)

Send a HTTP PUT request to URL.

Parameters

string	*$url*	The URL to post the request to. Should start with the protocol. For example: `http://`.
string	*$type*	This paramerter must be one of: string, file.
string	*$file*	What this parameter contains is decided by the $type parameter.

Exceptions

HttpConnectionException	

Returns

array Associative array containing: ∗ $return['status'] = The HTTP status code ∗ $return['headers'] = The HTTP headers of the reply ∗ $return['content'] = The body of the HTTP reply

Reimplemented from HttpConnection.

Reimplemented in RepositoryConnection.

The documentation for this class was generated from the following file:

- HttpConnection.php

5.9 FedoraApi Class Reference

This is a simple class that brings FedoraApiM and FedoraApiA together.

Public Member Functions

- __construct (RepositoryConnection $connection=NULL, FedoraApiSerializer $serializer=NULL)

  Constructor for the FedoraApi object.

Public Attributes

- **$a**
- **$m**

5.9.1 Detailed Description

This is a simple class that brings FedoraApiM and FedoraApiA together.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 FedoraApi::__construct (RepositoryConnection $ *connection* = NULL, FedoraApiSerializer $ *serializer* = NULL)

Constructor for the FedoraApi object.

Parameters

- RepositoryConnection	*$connection*	(Optional) If one isn't provided a default one will be used.
FedoraApiSerializer	*$serializer*	(Optional) If one isn't provided a default will be used.

The documentation for this class was generated from the following file:

- FedoraApi.php

5.10 FedoraApiA Class Reference

This class implements the Fedora API-A interface.

Public Member Functions

- __construct (RepositoryConnection $connection, FedoraApiSerializer $serializer)

  Constructor for the new FedoraApiA object.

- describeRepository ()

  Returns basic information about the Repository.
- findObjects ($type, $query, $max_results=NULL, $display_fields=array('pid', 'title'))

  Query fedora to return a list of objects.
- resumeFindObjects ($session_token)

  Given a session key returned from findObject, this will return the next set of objects.
- getDatastreamDissemination ($pid, $dsid, $as_of_date_time=NULL)

  Get the default dissemination of a datastream.
- getDissemination ($pid, $sdef_pid, $method, $method_parameters=NULL)

  Get a datastream dissemination from Fedora.
- getObjectHistory ($pid)

  Get the change history for the object.
- getObjectProfile ($pid, $as_of_date_time=NULL)

  Implements the getObjectProfile Fedora API-A method.
- listDatastreams ($pid, $as_of_date_time=NULL)

  List all the datastreams that are associated with this PID.
- listMethods ($pid, $sdef_pid= '', $as_of_date_time=NULL)

  Implements the listMethods Fedora API-A method.

Protected Attributes

- **$connection**
- **$serializer**

5.10.1 Detailed Description

This class implements the Fedora API-A interface.

This is a light wrapper around the Fedora interface. Very little attempt is put into putting things into native PHP datastructures.

See this page for more information: https://wiki.duraspace.org/display/FEDORA35/REST+API

5.10.2 Constructor & Destructor Documentation

5.10.2.1 FedoraApiA::__construct (RepositoryConnection $ *connection,* FedoraApiSerializer $ *serializer*)

Constructor for the new FedoraApiA object.

Parameters

-RepositoryConnection	*$connection*	Takes the Respository Connection object for the -Respository this API should connect to.
FedoraApiSerializer	*$serializer*	Takes the serializer object to that will be used to serialze the XML Fedora returns.

5.10.3 Member Function Documentation

5.10.3.1 FedoraApiA::describeRepository ( )

Returns basic information about the Repository.

This is listed as an unimplemented function in the official API for Fedora. However other libraries connecting to the Fedora REST interaface use this so we are including it here. It may change in the future.

Exceptions

RepositoryException	

Returns

array() An array describing the repository.

```
Array
(
    [repositoryName] => Fedora Repository
    [repositoryBaseURL] => http://localhost:8080/fedora
    [repositoryVersion] => 3.4.1
    [repositoryPID] => Array
        (
            [PID-namespaceIdentifier] => changeme
            [PID-delimiter] => :
            [PID-sample] => changeme:100
            [retainPID] => *
        )

    [repositoryOAI-identifier] => Array
        (
            [OAI-namespaceIdentifier] => example.org
            [OAI-delimiter] => :
            [OAI-sample] => oai:example.org:changeme:100
        )

    [sampleSearch-URL] => http://localhost:8080/fedora/objects
    [sampleAccess-URL] => http://localhost:8080/fedora/objects/demo:5
    [sampleOAI-URL] => http://localhost:8080/fedora/oai?verb=Identify
    [adminEmail] => Array
        (
            [0] => bob@example.org
            [1] => sally@example.org
        )

)
```

5.10.3.2 FedoraApiA::findObjects ($ *type*, $ *query*, $ *max_results* = NULL, $ *display_fields* = array('pid', 'title'))

Query fedora to return a list of objects.

Parameters

string	*$type*	The type of query. Decides the format of the next parameter. Valid options are: • query: specific query on certain fields • terms: search in any field
string	*$query*	The format of this parameter depends on what was passed to type. The formats are: • query: A sequence of space-separated conditions. - A condition consists of a metadata element name followed directly by an operator, followed directly by a value. Valid element names are (pid, label, state, ownerId, cDate, mDate, dcmDate, title, creator, subject, description, publisher, contributor, date, type, format, identifier, source, language, relation, coverage, rights). Valid operators are: contains (∼), equals (=), greater than (>), less than (<), greater than or equals (>=), less than or equals (<=). The contains (∼) operator may be used in combination with the ? and ∗ wildcards to query for simple string patterns. Values may be any string. If the string contains a space, the value should begin and end with a single quote character ('). If all conditions are met for an object, the object is considered a match. • terms: A phrase represented as a sequence of characters (including the ? and ∗ wildcards) for the search. If this sequence is found in any of the fields for an object, the object is considered a match.
int	*$max_-results*	(optional) Default: 25. The maximum number of results that the server should provide at once.
array	*$display_-fields*	(optional) Default: array('pid', 'title'). The fields to be returned as an indexed array. Valid element names are the same as the ones given for the query parameter.

Exceptions

RepositoryException	

Returns

array() The results are returned in an array key called 'results'. If there are more results that aren't returned then the search session information is contained in a key called 'session'. Note that it is possible for some display fields to be multivalued, such as identifier (DC allows multiple DC identifier results) in the case there are multiple results an array is returned instread of a string, this indexed array contains all of the values.

```
Array
(
   [session] => Array
       (
           [token] => 96b2604f040067645f45daf029062d6e
           [cursor] => 0
           [expirationDate] => 2012-03-07T14:28:24.886Z
       )

   [results] => Array
       (
           [0] => Array
               (
                   [pid] => islandora:collectionCModel
                   [title] => Islandora Collection Content Model
                   [identifier] => Contents of DC:Identifier
               )

           [1] => Array
               (
                   [pid] => islandora:testCModel
                   [title] => Test content model for Ari
                   [identifier] => Array
                       (
                           [0] => Contents of first DC:Identifier
                           [1] => Contents of seconds DC:Identifier
                       )

               )

       )

 )
```

5.10.3.3 FedoraApiA::getDatastreamDissemination ($ *pid,* $ *dsid,* $ *as_of_date_time* = NULL)

Get the default dissemination of a datastream.

(Get the contents.)

Parameters

String	*$pid*	Persistent identifier of the digital object.
String	*$dsid*	Datastream identifier.
array	*$as_of_-date_time*	(optional) Indicates that the result should be relative to the digital object as it existed at the given date and time. - Defaults to the most recent version.

Exceptions

Respository-Exception	

Returns

string The response from Fedora with the contents of the datastream.

5.10.3.4 FedoraApiA::getDissemination ($ *pid,* $ *sdef_pid,* $ *method,* $ *method_parameters* = NULL)

Get a datastream dissemination from Fedora.

Parameters

String	*$pid*	Persistent identifier of the digital object.
String	*$sdef_pid*	Persistent identifier of the sDef defining the methods.
String	*$method*	Method to invoke.
String	*$method_parameters*	A key-value paired array of parameters required by the method.

Exceptions

Respository-Exception	

Returns

string The response from Fedora.

5.10.3.5 FedoraApiA::getObjectHistory ($ *pid*)

Get the change history for the object.

Parameters

String	*$pid*	Persistent identifier of the digital object.

Exceptions

RepositoryException	

Returns

array() An array containing the different revisions of the object.

```
Array
(
    [0] => 2011-07-08T18:01:40.384Z
    [1] => 2011-07-08T18:01:40.464Z
    [2] => 2011-07-08T18:01:40.552Z
    [3] => 2011-07-08T18:01:40.694Z
    [4] => 2012-02-22T15:07:15.305Z
    [5] => 2012-02-29T14:20:28.857Z
    [6] => 2012-02-29T14:22:18.239Z
    [7] => 2012-02-29T14:22:46.545Z
    [8] => 2012-02-29T20:52:33.069Z
)
```

5.10.3.6 FedoraApiA::getObjectProfile ($ *pid*, $ *as_of_date_time* = NULL)

Implements the getObjectProfile Fedora API-A method.

Parameters

String	*$pid*	Persistent identifier of the digital object.
String	*$as_of_date_time*	(Optional) Indicates that the result should be relative to the digital object as it existed on the given date. Date Format: yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ

Exceptions

RepositoryException	

Returns

array() Returns information about the digital object.

```
Array
(
    [objLabel] => Islandora strict PDF content model
    [objOwnerId] => fedoraAdminnnn
    [objModels] => Array
        (
            [0] => info:fedora/fedora-system:ContentModel-3.0
            [1] => info:fedora/fedora-system:FedoraObject-3.0
        )

    [objCreateDate] => 2011-07-08T18:01:40.384Z
    [objLastModDate] => 2012-03-02T20:50:13.534Z
    [objDissIndexViewURL] => http://localhost:8080/fedora/objects/
      islandora%3Astrict_pdf/methods/fedora-system%3A3/viewMethodIndex
    [objItemIndexViewURL] => http://localhost:8080/fedora/objects/
      islandora%3Astrict_pdf/methods/fedora-system%3A3/viewItemIndex
    [objState] => A
)
```

5.10.3.7 FedoraApiA::listDatastreams ($ *pid,* $ *as_of_date_time* = NULL)

List all the datastreams that are associated with this PID.

Parameters

string	*$pid*	Persistent identifier of the digital object.
string	*$as_of_date_time*	(optional) Indicates that the result should be relative to the digital object as it existed on the given date. Date Format: yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ.

Exceptions

RepositoryException	

Returns

array() An associative array with the dsid of the datastreams as the key and the mimetype and label as the value.

```
Array
(
    [DC] => Array
        (
            [label] => Dublin Core Record for this object
            [mimetype] => text/xml
        )

    [RELS-EXT] => Array
        (
            [label] => Fedora Object-to-Object Relationship Metadata
            [mimetype] => text/xml
        )

    [ISLANDORACM] => Array
        (
            [label] => ISLANDORACM
            [mimetype] => text/xml
        )

)
```

5.10.3.8 FedoraApiA::listMethods ($ *pid,* $ *sdef_pid* = ' ' , $ *as_of_date_time* = NULL)

Implements the listMethods Fedora API-A method.

Parameters

String	*$pid*	Persistent identifier of the digital object.
String	*$sdef_pid*	(Optional) Persistent identifier of the SDef defining the methods.
String	*$as_of_date_time*	(Optional) Indicates that the result should be relative to the digital object as it existed on the given date. Date Format: yyyy-MM-dd or yyyy-MM-ddTHH:mm:ssZ.

Exceptions

RepositoryException	

Returns

array() An array containing data about the methods that can be called. The result array is an associative array where the sdef pid is the key and the value is a indexed array of methods.

```
Array
(
    [ilives:viewerSdef] => Array
        (
            [0] => getViewer
        )

    [ilives:jp2Sdef] => Array
        (
            [0] => getMetadata
            [1] => getRegion
        )

    [fedora-system:3] => Array
       (
            [0] => viewObjectProfile
            [1] => viewMethodIndex
            [2] => viewItemIndex
            [3] => viewDublinCore
        )

)
```

5.10.3.9 FedoraApiA::resumeFindObjects ($ *session_token*)

Given a session key returned from findObject, this will return the next set of objects.

Parameters

string	*$session_token*	Session token returned from previous search call.

Exceptions

Respository-Exception	

Returns

array() The result format is the same as findObjects.

See also

FedoraApiA::findObjects

The documentation for this class was generated from the following file:

- FedoraApi.php

5.11 FedoraApiM Class Reference

This class implements the Fedora API-M interface.

Public Member Functions

- __construct (RepositoryConnection $connection, FedoraApiSerializer $serializer)

  Constructor for the new FedoraApiM object.
- addDatastream ($pid, $dsid, $type, $file, $params)

  Add a new datastream to a fedora object.
- addRelationship ($pid, $relationship, $is_literal, $datatype=NULL)

  Add a RDF relationship to a Fedora object.
- export ($pid, $params=array())

  Export a Fedora object with the given PID.
- getDatastream ($pid, $dsid, $params=array())

  Returns information about the datastream.
- getDatastreamHistory ($pid, $dsid)

  Get information on the different versions of a datastream that are avilable in Fedora.
- getNextPid ($namespace=NULL, $numpids=NULL)

  Get a new unused PID.
- getObjectXml ($pid)

  Get the Fedora Objects XML (Foxml)
- getRelationships ($pid, $relationship=array())

  Query relationships for a particular fedora object.
- ingest ($params=array())

  Create a new object in Fedora.
- modifyDatastream ($pid, $dsid, $params=array())

  Update a datastream.
- modifyObject ($pid, $params=NULL)

  Update Fedora Object parameters.
- purgeDatastream ($pid, $dsid, $params=array())

  Purge a datastream from from Fedora.
- purgeObject ($pid, $log_message=NULL)

  Purge an object.
- **upload** ($file)

5.11.1 Detailed Description

This class implements the Fedora API-M interface.

This is a light wrapper around the Fedora interface. Very little attempt is put into putting things into native PHP datastructures.

See this page for more information: https://wiki.duraspace.org/display/FEDORA35/REST+API

5.11.2 Constructor & Destructor Documentation

5.11.2.1 FedoraApiM::__construct (RepositoryConnection $ *connection,* FedoraApiSerializer $ *serializer*)

Constructor for the new FedoraApiM object.

Parameters

RepositoryConnection	*$connection*	Takes the Respository Connection object for the Respository this API should connect to.
FedoraApiSerializer	*$serializer*	Takes the serializer object to that will be used to serialze the XML Fedora returns.

5.11.3 Member Function Documentation

5.11.3.1 FedoraApiM::addDatastream ($ *pid,* $ *dsid,* $ *type,* $ *file,* $ *params*)

Add a new datastream to a fedora object.

The datastreams are sent to Fedora using a multipart post if a string or file is provided otherwise Fedora will go out and fetch the URL

Parameters

string	*$pid*	Persistent identifier of the digital object.
string	*$dsid*	Datastream identifier.
string	*$type*	This parameter tells the function what type of arguement is given for file. It must be one of: • string: The datastream is passed as a string. • file: The datastream is contained in a file. • url: The datastream is located at a URL, which is passed as a string. this is the only option that can be used for R and E type datastreams.

string	*$file*	This parameter depends on what is selected for $type. • string: A string containing the datastream. • file: A string containing the file name that contains the datastream. The file name must be a full path. • url: A string containing the publically accessable URL that the datastream is located at.
	array()	$params (optional) An array that can have one or more of the following elements: • controlGroup: one of "X", "M", "R", or "E" (Inline ∗X∗ML, ∗M∗anaged Content, ∗R∗edirect, or ∗E∗xternal Referenced). Default: X. • altIDs: alternate identifiers for the datastream. - A space seperated list of alternate identifiers for the datastream. • dsLabel: the label for the datastream. • versionable: enable versioning of the datastream (boolean). • dsState: one of "A", "I", "D" (∗A∗ctive, ∗I∗nactive, ∗D∗eleted). • formatURI: the format URI of the datastream. • checksumType: the algorithm used to compute the checksum. One of DEFAULT, DISABLED, MD5, SHA-1, SHA-256, SHA-385, SHA-512. • checksum: the value of the checksum represented as a hexadecimal string. • mimeType: the MIME type of the content being added, this overrides the Content-Type request header. • logMessage: a message describing the activity being performed.

Exceptions

RepositoryException	

Returns

array() Returns an array describing the new datastream. This is the same array returned by getDatastream. This may also contain an dsAltID key, that contains any alternate ids if any are specified.

```
Array
(
    [dsLabel] =>
    [dsVersionID] => test.3
```

```
            [dsCreateDate] => 2012-03-07T18:03:38.679Z
            [dsState] => A
            [dsMIME] => text/xml
            [dsFormatURI] =>
            [dsControlGroup] => M
            [dsSize] => 22
            [dsVersionable] => true
            [dsInfoType] =>
            [dsLocation] => islandora:strict_pdf+test+test.3
            [dsLocationType] => INTERNAL_ID
            [dsChecksumType] => DISABLED
            [dsChecksum] => none
    )
```

See also

FedoraApiM::getDatastream

5.11.3.2 FedoraApiM::addRelationship ($ *pid,* $ *relationship,* $ *is_literal,* $ *datatype =* NULL)

Add a RDF relationship to a Fedora object.

Parameters

string	*$pid*	Persistent identifier of the digital object.
array	*$relationship*	An array containing the subject, predicate and object for the relationship. • subject: (optional) Subject of the relationship. Either a URI for the object or one of its datastreams. If none is given then the URI for the current object is used. • predicate: Predicate of the relationship. • object: Object of the relationship.
boolean	*$is_literal*	true if the object of the relationship is a literal, false if it is a URI
string	*$datatype*	(optional) if the object is a literal, the datatype of the literal.

Exceptions

RepositoryException	

See also

FedoraApiM::getRelationships
FedoraApiM::purgeRelationships

5.11.3.3 FedoraApiM::export ($ *pid,* $ *params* = `array()`)

Export a Fedora object with the given PID.

Parameters

string	*$pid*	Persistent identifier of the digital object.
array	*$params*	(optional) An array that can have one or more of the following elements: • format: The XML format to export. One of info-:fedora/fedora-system:FOXML-1.1 (default), info-:fedora/fedora-system:FOXML-1.0, info:fedora/fedora-system:METSFedoraExt-1.1, info:fedora/fedora-system:METSFedoraExt-1.0, info:fedora/fedora-system:ATOM-1.1, info:fedora/fedora-system:ATOM-Zip-1.1 • context: The export context, which determines how datastream URLs and content are represented. -Options: public (default), migrate, archive. • encoding: The preferred encoding of the exported X-ML.

Exceptions

RepositoryException	

Returns

string A string containing the requested XML.

5.11.3.4 FedoraApiM::getDatastream ($ *pid,* $ *dsid,* $ *params* = `array()`)

Returns information about the datastream.

Parameters

string	*$pid*	Persistent identifier of the digital object.
string	*$dsid*	Datastream identifier.
array	*$params*	(optional) An array that can have one or more of the following elements: • asOfDateTime: Indicates that the result should be relative to the digital object as it existed on the given date. • validateChecksum: verifies that the Datastream content has not changed since the checksum was initially computed.

Exceptions

RepositoryException	

Returns

array() An array containing information about the datastream. This may also contains a key dsAltID which contains alternate ids if any are specified.

```
Array
(
    [dsLabel] =>
    [dsVersionID] => test.3
    [dsCreateDate] => 2012-03-07T18:03:38.679Z
    [dsState] => A
    [dsMIME] => text/xml
    [dsFormatURI] =>
    [dsControlGroup] => M
    [dsSize] => 22
    [dsVersionable] => true
    [dsInfoType] =>
    [dsLocation] => islandora:strict_pdf+test+test.3
    [dsLocationType] => INTERNAL_ID
    [dsChecksumType] => DISABLED
    [dsChecksum] => none
)
```

5.11.3.5 FedoraApiM::getDatastreamHistory ($ *pid*, $ *dsid*)

Get information on the different versions of a datastream that are avilable in Fedora.

Parameters

string	*$pid*	Persistent identifier of the digital object.
string	*$dsid*	Datastream identifier.

Exceptions

RepositoryException	

Returns

array Returns a indexed array with the same keys as getDatastream.

```
Array
(
    [0] => Array
        (
            [dsLabel] =>
            [dsVersionID] => test.3
            [dsCreateDate] => 2012-03-07T18:03:38.679Z
            [dsState] => A
            [dsMIME] => text/xml
            [dsFormatURI] =>
            [dsControlGroup] => M
            [dsSize] => 22
```

```
                [dsVersionable] => true
                [dsInfoType] =>
                [dsLocation] => islandora:strict_pdf+test+test.3
                [dsLocationType] => INTERNAL_ID
                [dsChecksumType] => DISABLED
                [dsChecksum] => none
            )

        [1] => Array
            (
                [dsLabel] =>
                [dsVersionID] => test.2
                [dsCreateDate] => 2012-03-07T18:03:13.722Z
                [dsState] => A
                [dsMIME] => text/xml
                [dsFormatURI] =>
                [dsControlGroup] => M
                [dsSize] => 22
                [dsVersionable] => true
                [dsInfoType] =>
                [dsLocation] => islandora:strict_pdf+test+test.2
                [dsLocationType] => INTERNAL_ID
                [dsChecksumType] => DISABLED
                [dsChecksum] => none
            )

    )
```

See also

FedoraApiM::getDatastream

5.11.3.6 FedoraApiM::getNextPid ($ *namespace* = NULL, $ *numpids* = NULL)

Get a new unused PID.

Parameters

string	*$namespace*	The namespace to get the PID in. This defaults to default namespace of the repository. This should not contain the PID seperator, for example it should be islandora not islandora:.
int	*$numpids*	The number of pids being requested.

Exceptions

RepositoryException	

Returns

array/string If one pid is requested it is returned as a string. If multiple pids are requested they they are returned in an array containg strings.

```
Array
```

```
        (
            [0] => test:7
            [1] => test:8
        )
```

5.11.3.7 FedoraApiM::getObjectXml ($ *pid*)

Get the Fedora Objects XML (Foxml)

Parameters

string	*$pid*	Persistent identifier of the digital object.

Exceptions

RepositoryException	

Returns

string A string containing the objects foxml

See also

FedoraApiM::export

5.11.3.8 FedoraApiM::getRelationships ($ *pid*, $ *relationship* = `array()`)

Query relationships for a particular fedora object.

Parameters

string	*$pid*	Persistent identifier of the digital object.
array	*$relationship*	(Optional) An array defining the relationship: • subject: subject of the relationship(s). Either a URI for the object or one of its datastreams. defaults to the URI of the object. • predicate: predicate of the relationship(s), if missing returns all predicates.

Exceptions

RepositoryException	

Returns

array An indexed array with all the relationships.

```
Array
(
    [0] => Array
        (
            [subject] => islandora:strict_pdf
            [predicate] => Array
                (
                    [predicate] => hasModel
                    [uri] => info:fedora/fedora-system:def/model#
                    [alias] =>
                )

            [object] => Array
                (
                    [literal] => FALSE
                    [value] => fedora-system:FedoraObject-3.0
                )

        )

    [1] => Array
        (
            [subject] => islandora:strict_pdf
            [predicate] => Array
                (
                    [predicate] => bar
                    [uri] => http://woot/foo#
                    [alias] =>
                )

            [object] => Array
                (
                    [literal] => TRUE
                    [value] => thedude
                )

        )

)
```

5.11.3.9 FedoraApiM::ingest ($ *params* = `array()`)

Create a new object in Fedora.

This could be ingesting a XML file as a string or a file. Executing this request with no XML file content will result in the creation of a new, empty object (with either the specified PID or a system-assigned PID). The new object will contain only a minimal DC datastream specifying the dc:identifier of the object.

Parameters

array	*$params*	(optional) An array that can have one or more of the following elements: • pid: persistent identifier of the object to be created. - If this is not supplied then either a new PID will be created for this object or the PID to be used is encoded in the XML included as the body of the request • string: The XML file defining the new object as a string • file: The XML file defining the new object as a string containing the full path to the XML file. This must not be used with the string parameter • label: the label of the new object • format: the XML format of the object to be ingested. One of info:fedora/fedora-system:FO-XML-1.1, info:fedora/fedora-system:FOXML-1.0, info:fedora/fedora-system:METSFedoraExt-1.1, info:fedora/fedora-system:METSFedoraExt-1.0, info-:fedora/fedora-system:ATOM-1.1, info:fedora/fedora-system:ATOMZip-1.1 • encoding: the encoding of the XML to be ingested. If this is specified, and given as anything other than UTF-8, you must ensure that the same encoding is declared in the XML. • namespace: The namespace to be used to create a PID for a new empty object: if a 'string' parameter is included with the request, the namespace parameter is ignored. • ownerId: the id of the user to be listed at the object owner. • logMessage: a message describing the activity being performed.

Exceptions

RepositoryException	

Returns

string The PID of the newly created object.

Todo This function is a problem in Fedora < 3.5 where ownerId does not properly get set. `https://jira.duraspace.org/browse/FCREPO-963`. We should deal with this.

5.11.3.10 FedoraApiM::modifyDatastream ($ *pid,* $ *dsid,* $ *params* = `array()`)

Update a datastream.

Either changing its metadata, updaing the datastream contents or both.

Parameters

string	*$pid*	Persistent identifier of the digital object.
string	*$dsid*	Datastream identifier.
array	*$params*	(optional) An array that can have one or more of the following elements: • dsFile: String containing the full path to a file that will be used as the new contents of the datastream. • dsString: String containing the new contents of the datastream. • dsLocation: String containing a URL to fetch the new datastream from. Only ONE of dsFile, dsString or dsLocation should be used. • altIDs: alternate identifiers for the datastream. This is a space seperated string of alternate identifiers for the datastream. • dsLabel: the label for the datastream. • versionable: enable versioning of the datastream. • dsState: one of "A", "I", "D" (∗A∗ctive, ∗I∗nactive, ∗D∗eleted) • formatURI: the format URI of the datastream • checksumType: the algorithm used to compute the checksum. This has to be one of: DEFAULT, DISABLED, MD5, SHA-1, SHA-256, SHA-384, SHA-512. If this parameter is given and no checksum is given the checksum will be computed. • checksum: the value of the checksum represented as a hexadecimal string. This checksum must be computed by the algorithm defined above. • mimeType: the MIME type of the content being added, this overrides the Content-Type request header. • logMessage: a message describing the activity being performed • lastModifiedDate: date/time of the last (known) modification to the datastream, if the actual last modified date is later, a 409 response is returned. This can be used for opportunistic object locking.

Exceptions

RepositoryException	

Returns

array An array contianing information about the updated datastream. This array is the same as the array returned by getDatastream.

See also

FedoraApiM::getDatastream

5.11.3.11 FedoraApiM::modifyObject ($ *pid,* $ *params* = `NULL`)

Update Fedora Object parameters.

Parameters

string	*$pid*	Persistent identifier of the digital object.
array	*$params*	(optional) An array that can have one or more of the following elements: • label: object label. • ownerId: the id of the user to be listed at the object owner. • state: the new object state - ∗A∗ctive, ∗I∗nactive, or ∗D∗eleted. • logMessage: a message describing the activity being performed. • lastModifiedDate: date/time of the last (known) modification to the datastream, if the actual last modified date is later, a 409 response is returned. This can be used for opportunistic object locking.

Exceptions

RepositoryException	

Returns

string A string containg the timestamp of the object modification.

5.11.3.12 FedoraApiM::purgeDatastream ($ *pid*, $ *dsid*, $ *params* = `array()`)

Purge a datastream from from Fedora.

This perminatly removes the datastream and all its associated data.

Parameters

string	*$pid*	Persistent identifier of the digital object.
string	*$dsid*	Datastream identifier
array	*$params*	(optional) An array that can have one or more of the following elements: • startDT: the (inclusive) start date-time stamp of the range. If not specified, this is taken to be the lowest possible value, and thus, the entire version history up to the endDT will be purged. • endDT: the (inclusive) ending date-time stamp of the range. If not specified, this is taken to be the greatest possible value, and thus, the entire version history back to the startDT will be purged. • logMessage: a message describing the activity being performed.

Exceptions

RepositoryException	

Returns

array An array containing the timestamps of the datastreams that were removed.

```
Array
(
    [0] => 2012-03-08T18:44:15.214Z
    [1] => 2012-03-08T18:44:15.336Z
)
```

5.11.3.13 FedoraApiM::purgeObject ($ *pid*, $ *log_message* = `NULL`)

Purge an object.

Parameters

string	*$pid*	Persistent identifier of the digital object.
string	*$log_message*	(optional) A message describing the activity being performed.

Exceptions

RepositoryException	

Returns

string Timestamp when object was deleted.

The documentation for this class was generated from the following file:

- FedoraApi.php

5.12 FedoraApiSerializer Class Reference

A class to Serialize the XML responses from Fedora into PHP arrays.

Public Member Functions

- domDocumentExceptionHandler ($errno, $errstr, $errfile= '', $errline=NULL, $errcontext=NULL)

  This is a simple exception handler, that will throw an exception if there is a problem parsing XML from Fedora.
- describeRepository ($request)

  Serializes the data returned in FedoraApiA::describeRepository()
- findObjects ($request)

  Serializes the data returned in FedoraApiA::findObjects()
- resumeFindObjects ($request)

  Serializes the data returned in FedoraApiA::resumeFindObjects()
- getDatastreamDissemination ($request)

  Serializes the data returned in FedoraApiA::getDatastreamDissemination()
- getDissemination ($request)

  Serializes the data returned in FedoraApiA::getDissemination()
- getObjectHistory ($request)

  Serializes the data returned in FedoraApiA::getObjectHistory()
- getObjectProfile ($request)

  Serializes the data returned in FedoraApiA::getObjectProfile()
- listDatastreams ($request)

  Serializes the data returned in FedoraApiA::listDatastreams()

- listMethods ($request)

  Serializes the data returned in FedoraApiA::listMethods()
- addDatastream ($request)

  Serializes the data returned in FedoraApiM::addDatastream()
- addRelationship ($request)

  Serializes the data returned in FedoraApiM::addRelationship()
- export ($request)

  Serializes the data returned in FedoraApiM::export()
- getDatastream ($request)

  Serializes the data returned in FedoraApiM::getDatastream()
- getDatastreamHistory ($request)

  Serializes the data returned in FedoraApiM::getDatastreamHistory()
- getNextPid ($request)

  Serializes the data returned in FedoraApiM::getNextPid()
- getObjectXml ($request)

  Serializes the data returned in FedoraApiM::getObjectXml()
- getRelationships ($request)

  Serializes the data returned in FedoraApiM::getRelationships()
- ingest ($request)

  Serializes the data returned in FedoraApiM::ingest()
- modifyDatastream ($request)

  Serializes the data returned in FedoraApiM::modifyDatastream()
- modifyObject ($request)

  Serializes the data returned in FedoraApiM::modifyObject()
- purgeDatastream ($request)

  Serializes the data returned in FedoraApiM::purgeDatastream()
- purgeObject ($request)

  Serializes the data returned in FedoraApiM::purgeObject()
- **upload** ($request)

Protected Member Functions

- loadSimpleXml ($xml)

  Simple function that takes an XML string and returns a SimpleXml object.
- loadDomDocument ($xml)

  Simple function that takes an XML string and returns a DomDocument object.
- flattenDocument ($xml, $make_array=array())

  Flatten a simplexml object returning an array containing the text from the XML.

5.12.1 Detailed Description

A class to Serialize the XML responses from Fedora into PHP arrays.

5.12.2 Member Function Documentation

5.12.2.1 FedoraApiSerializer::domDocumentExceptionHandler (*$ errno, $ errstr, $ errfile = ' ' , $ errline =* NULL*, $ errcontext =* NULL)

This is a simple exception handler, that will throw an exception if there is a problem parsing XML from Fedora.

This is nice so we can catch it.

Parameters

int	*$errno*	The error number
string	*$errstr*	String describing an error.
string	*$errfile*	(optional) The third parameter is optional, errfile, which contains the filename that the error was raised in, as a string.
int	*$errline*	(optional) The fourth parameter is optional, errline, which contains the line number the error was raised at, as an integer.
array	*$errcontext*	(optional) The fifth parameter is optional, errcontext, which is an array that points to the active symbol table at the point the error occurred. In other words, errcontext will contain an array of every variable that existed in the scope the error was triggered in. User error handler must not modify error context.

Returns

boolean TRUE if we through an exception FALSE otherwise

See also

php.net/manual/en/function.set-error-handler.php

5.12.2.2 FedoraApiSerializer::flattenDocument (*$ xml, $ make_array =* array()) [protected]

Flatten a simplexml object returning an array containing the text from the XML.

This is often used to get data back from fedora. It also makes sure to cast everything to string.

Parameters

SimpleXmlElement	*$xml*	The simplexml elemnt to be processed.
	array()	$make_array (optional) This parameter specifies tags that should become an array instead of an element in an array. This is used to get consistant values for things that are multivalued when there is only one value returned.

Returns

array An array representation of the XML.

5.12.2.3 FedoraApiSerializer::loadDomDocument ($ *xml*) [protected]

Simple function that takes an XML string and returns a DomDocument object.

It makes sure no PHP errors or warnings are issued and instead throws an exception if the XML parse failed.

Parameters

string	*$xml*	The XML as a string

Exceptions

RepositoryXmlError	

Returns

DomDocument Return an istantiated DomDocument

5.12.2.4 FedoraApiSerializer::loadSimpleXml ($ *xml*) [protected]

Simple function that takes an XML string and returns a SimpleXml object.

It makes sure no PHP errors or warnings are issued and instead throws an exception if the XML parse failed.

Parameters

string	*$xml*	The XML as a string

Exceptions

RepositoryXmlError	

Returns

SimpleXmlElement Return an istantiated simplexml

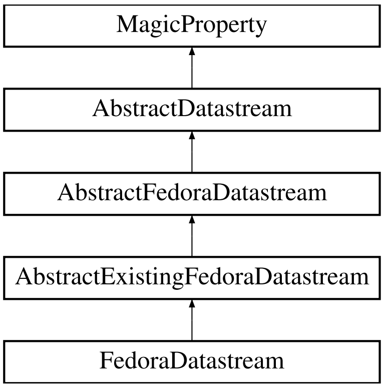
The documentation for this class was generated from the following file:

- FedoraApiSerializer.php

5.13 FedoraDatastream Class Reference

This class implements a fedora datastream.

Inheritance diagram for FedoraDatastream:



Public Member Functions

- __construct ($id, FedoraObject $object, FedoraRepository $repository, array $datastream_info=NULL)

    Domo arigato, Mr.
- refresh ()

    This function clears the datastreams caches, so everything will be requested directly from fedora again.
- setContentFromFile ($file)
- setContentFromUrl ($url)
- setContentFromString ($string)
- count ()
- offsetExists ($offset)
- offsetGet ($offset)
- offsetSet ($offset, $value)
- offsetUnset ($offset)
- getIterator ()

    IteratorAggregate::getIterator()

Public Attributes

- **$forceUpdate** = FALSE

Protected Member Functions

- populateDatastreamHistory ()

  This populates datastream history if it needs to be populated.
- populateDatastreamInfo ()

  This function uses datastream history to populate datastream info.
- modifyDatastream (array $args)

  This function modifies the datastream in fedora while adding the parameters needed to respect datastram locking and making sure that we keep the internal class cache of the datastream up to date.
- getDatastreamContent ($version=NULL)

  A light wrapper around getting the datastream content from fedora.
- controlGroupMagicProperty ($function, $value)
- stateMagicProperty ($function, $value)
- labelMagicProperty ($function, $value)
- versionableMagicProperty ($function, $value)
- mimetypeMagicProperty ($function, $value)
- formatMagicProperty ($function, $value)
- sizeMagicProperty ($function, $value)
- checksumMagicProperty ($function, $value)
- checksumTypeMagicProperty ($function, $value)
- createdDateMagicProperty ($function, $value)
- contentMagicProperty ($function, $value)
- urlMagicProperty ($function, $value)

Protected Attributes

- **$datastreamHistory** = NULL

5.13.1 Detailed Description

This class implements a fedora datastream.

It also lets old versions of datastreams be accessed using array notation. For example to see how many versions of a datastream there are:

```
count($datastream)
```

Old datastreams are indexed newest to oldest. The current version is always index 0, and older versions are indexed from that. Old versions can be discarded using the unset command.

These functions respect datastream locking. If a datastream changes under your feet then an exception will be raised.

5.13.2 Constructor & Destructor Documentation

5.13.2.1 FedoraDatastream::__construct ($ *id*, FedoraObject $ *object*, FedoraRepository $ *repository*, array $ *datastream_info* = NULL)

Domo arigato, Mr.

Roboto. Constructor.

5.13.3 Member Function Documentation

5.13.3.1 FedoraDatastream::checksumMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::checksum

Todo maybe add functionality to set it to auto

5.13.3.2 FedoraDatastream::checksumTypeMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::checksumType

5.13.3.3 FedoraDatastream::contentMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::content

Todo We should perhaps cache this? depending on size?

5.13.3.4 FedoraDatastream::controlGroupMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::controlGroup

Reimplemented from AbstractFedoraDatastream.

5.13.3.5 FedoraDatastream::count ( )

See also

Countable::count

5.13.3.6 FedoraDatastream::createdDateMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractDatastream::createdDate

5.13.3.7 FedoraDatastream::formatMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractDatastream::format

5.13.3.8 FedoraDatastream::labelMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractDatastream::label

5.13.3.9 FedoraDatastream::mimetypeMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractDatastream::mimetype

5.13.3.10 FedoraDatastream::offsetExists ($ *offset*)

See also

ArrayAccess::offsetExists

5.13.3.11 FedoraDatastream::offsetGet ($ *offset*)

See also

ArrayAccess::offsetGet

5.13.3.12 FedoraDatastream::offsetSet ($ *offset*, $ *value*)

See also

ArrayAccess::offsetSet

5.13.3.13 FedoraDatastream::offsetUnset ($ *offset*)

See also

ArrayAccess::offsetUnset

5.13.3.14 FedoraDatastream::setContentFromFile ($ *file*)

See also

AbstractDatastream::setContentFromFile

Reimplemented from AbstractDatastream.

5.13.3.15 FedoraDatastream::setContentFromString ($ *string*)

See also

AbstractDatastream::setContentFromString

Reimplemented from AbstractDatastream.

5.13.3.16 FedoraDatastream::setContentFromUrl ($ *url*)

See also

AbstractDatastream::setContentFromUrl

Reimplemented from AbstractDatastream.

5.13.3.17 FedoraDatastream::sizeMagicProperty ($ *function*, $ *value*) `[protected]`

See also

AbstractDatastream::size

5.13.3.18 FedoraDatastream::stateMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::state

5.13.3.19 FedoraDatastream::urlMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::url

5.13.3.20 FedoraDatastream::versionableMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::versionable

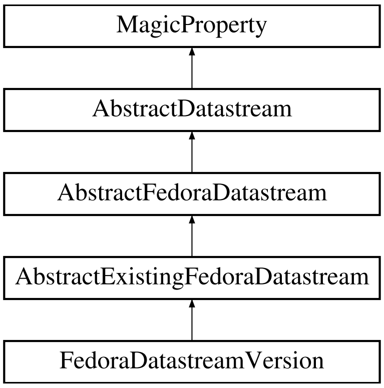
The documentation for this class was generated from the following file:

- Datastream.php

5.14 FedoraDatastreamVersion Class Reference

This class implements an old version of a fedora datastream.

Inheritance diagram for FedoraDatastreamVersion:



Public Member Functions

- __construct ($id, array $datastream_info, FedoraDatastream $datastream, FedoraObject $object, FedoraRepository $repository)

  The Constructor! Sounds like a superhero doesn't it.
- setContentFromFile ($file)
- setContentFromString ($string)
- setContentFromUrl ($url)

Public Attributes

- **$parent**

Protected Member Functions

- error ()

  This function gives us a consistant error across this whole clas.
- generalReadOnly ($offset, $unset_val, $function, $value)

  Since this whole class is read only, this is a general implementation of the MagicPropery function that is ready only.
- stateMagicProperty ($function, $value)
- labelMagicProperty ($function, $value)
- versionableMagicProperty ($function, $value)
- mimetypeMagicProperty ($function, $value)
- formatMagicProperty ($function, $value)
- sizeMagicProperty ($function, $value)
- checksumMagicProperty ()
- createdDateMagicProperty ($function, $value)
- contentMagicProperty ($function, $value)

5.14.1 Detailed Description

This class implements an old version of a fedora datastream.

Its properties are the same of a normal fedora datastream, except since its an older verion everything is read only.

5.14.2 Constructor & Destructor Documentation

5.14.2.1 FedoraDatastreamVersion::__construct ($ *id,* array $ *datastream_info,* FedoraDatastream $ *datastream,* FedoraObject $ *object,* FedoraRepository $ *repository*)

The Constructor! Sounds like a superhero doesn't it.

Constructor away!

5.14.3 Member Function Documentation

5.14.3.1 FedoraDatastreamVersion::checksumMagicProperty ( ) [protected]

See also

AbstractDatastream::checksum

5.14.3.2 FedoraDatastreamVersion::contentMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::content

5.14.3.3 FedoraDatastreamVersion::createdDateMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::createDate

5.14.3.4 FedoraDatastreamVersion::formatMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::format

5.14.3.5 FedoraDatastreamVersion::labelMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::label

5.14.3.6 FedoraDatastreamVersion::mimetypeMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::mimetype

5.14.3.7 FedoraDatastreamVersion::setContentFromFile ($ *file*)

See also

AbstractDatastream::setContentFromFile()

Reimplemented from AbstractDatastream.

5.14.3.8 FedoraDatastreamVersion::setContentFromString ($ *string*)

See also

AbstractDatastream::setContentFromString()

Reimplemented from AbstractDatastream.

5.14.3.9 FedoraDatastreamVersion::setContentFromUrl ($ *url*)

See also

AbstractDatastream::setContentFromUrl()

Reimplemented from AbstractDatastream.

5.14.3.10 FedoraDatastreamVersion::sizeMagicProperty ($ *function,* $ *value*) `[protected]`

See also

AbstractDatastream::size

5.14.3.11 FedoraDatastreamVersion::stateMagicProperty ($ *function,* $ *value*) `[protected]`

See also

AbstractDatastream::state

5.14.3.12 FedoraDatastreamVersion::versionableMagicProperty ($ *function,* $ *value*) `[protected]`

See also

AbstractDatastream::versionable

The documentation for this class was generated from the following file:

- Datastream.php

5.15 FedoraDate Class Reference

This file represents a Fedora Date.

Public Member Functions

- **__toString** ()
- **__construct** ($time)

5.15.1 Detailed Description

This file represents a Fedora Date.

It wraps the PHP datetime functions specifically for the date format used in Fedora. This allows the easy comparison of dates for example.

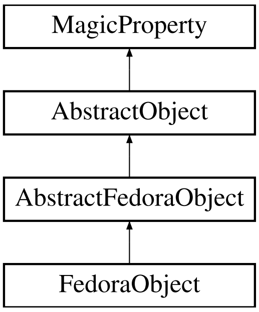
The documentation for this class was generated from the following file:

- FedoraDate.php

5.16 FedoraObject Class Reference

Represents a Fedora Object.

Inheritance diagram for FedoraObject:



Public Member Functions

- __construct ($id, FedoraRepository $repository)

  The class constructor.
- purgeDatastream ($id)

  Purge a datastream.
- getDatastream ($id)
- constructDatastream ($id, $control_group= 'M')

- ingestDatastream (&$ds)
- count ()
- offsetExists ($offset)
- offsetGet ($offset)
- offsetSet ($offset, $value)
- offsetUnset ($offset)
- getIterator ()

  IteratorAggregate::getIterator()

Public Attributes

- **$forceUpdate** = FALSE

Protected Member Functions

- populateDatastreams ()

  this will populate the datastream list the first time it is needed.
- modifyObject ($params)

  This is a wrapper on APIM modify object to make sure we respect locking.
- stateMagicProperty ($function, $value)
- labelMagicProperty ($function, $value)
- ownerMagicProperty ($function, $value)
- createdDateMagicProperty ($function, $value)
- lastModifiedDateMagicProperty ($function, $value)
- modelsMagicProperty ($function, $value)

Protected Attributes

- **$datastreams** = NULL

5.16.1 Detailed Description

Represents a Fedora Object.

Should be created by the factory method in the repository. Respects object locking. Will throw an exception if an object is modified under us.

5.16.2 Constructor & Destructor Documentation

5.16.2.1 FedoraObject::__construct ($ *id*, FedoraRepository $ *repository*)

The class constructor.

Should be instantiated by the repository.

Reimplemented from AbstractFedoraObject.

5.16.3 Member Function Documentation

5.16.3.1 FedoraObject::constructDatastream ($ *id*, $ *control_group* = 'M')

See also

AbstractObject::constructDatastream()

Reimplemented from AbstractFedoraObject.

5.16.3.2 FedoraObject::count ( )

See also

Countable::count

5.16.3.3 FedoraObject::createdDateMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractObject::createdDate

5.16.3.4 FedoraObject::getDatastream ($ *id*)

See also

AbstractObject::getDatastream()

Reimplemented from AbstractObject.

5.16.3.5 FedoraObject::ingestDatastream (&$ *ds*)

See also

AbstractObject::ingestDatastream()

Reimplemented from AbstractObject.

5.16.3.6 FedoraObject::labelMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractObject::label

Reimplemented from AbstractFedoraObject.

5.16.3.7 FedoraObject::lastModifiedDateMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractObject::lastModifiedDate

5.16.3.8 FedoraObject::modelsMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractObject::models

5.16.3.9 FedoraObject::offsetExists ($ *offset*)

See also

ArrayAccess::offsetExists

5.16.3.10 FedoraObject::offsetGet ($ *offset*)

See also

ArrayAccess::offsetGet

5.16.3.11 FedoraObject::offsetSet ($ *offset,* $ *value*)

See also

ArrayAccess::offsetSet

5.16.3.12 FedoraObject::offsetUnset ($ *offset*)

See also

ArrayAccess::offsetUnset

5.16.3.13 FedoraObject::ownerMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractObject::owner

Reimplemented from AbstractFedoraObject.

5.16.3.14 FedoraObject::purgeDatastream ($ *id*)

Purge a datastream.

Parameters

string	*$id*	The id of hte datastream to purge.

Returns

boolean Returns TRUE on success and FALSE on failure.

Reimplemented from AbstractObject.

5.16.3.15 FedoraObject::stateMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractObject::state

Reimplemented from AbstractFedoraObject.

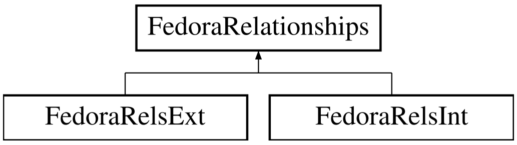
The documentation for this class was generated from the following file:

- Object.php

5.17 FedoraRelationships Class Reference

This is the base class for Fedora Relationships.

Inheritance diagram for FedoraRelationships:



Public Member Functions

- __construct (array $namespaces=NULL)

  The constructor.
- registerNamespace ($alias, $uri)

  Add a new namespace to the relationship xml.

Public Attributes

- **$datastream** = NULL

Protected Member Functions

- getXpath ($document)

  This function returns a domXPath object with all the current namespaces already registered.
- getDom ()

  Sets up a domdocument for the functions.
- updateDatastream ($document)

  This updates the associated datastreams content.
- internalAdd ($subject, $predicate_uri, $predicate, $object, $literal=FALSE)

  Add a new relationship.
- getXpathResults ($xpath_object, $subject, $predicate_uri, $predicate, $object, $literal)

  This function is used to create an xpath expression based on the input.
- internalGet ($subject, $predicate_uri=NULL, $predicate=NULL, $object=NULL, $literal=FALSE)

  This function queries the relationships in the assocaited datastream.
- internalRemove ($subject, $predicate_uri, $predicate, $object, $literal=FALSE)

  This function removes relationships that match the pattern from the datastream.

Protected Attributes

- **$namespaces**

5.17.1 Detailed Description

This is the base class for Fedora Relationships.

Todo potentially we should validate the predicate URI

5.17.2 Constructor & Destructor Documentation

5.17.2.1 FedoraRelationships::__construct (array $ *namespaces* = `NULL`)

The constructor.

This will usually be called by one of its subclasses.

Parameters

array	*$namespaces*	An array of default namespaces.

5.17.3 Member Function Documentation

5.17.3.1 FedoraRelationships::getDom ( ) [protected]

Sets up a domdocument for the functions.

Returns

DomDocument The domdocument to modify

5.17.3.2 FedoraRelationships::getXpath ($ *document*) [protected]

This function returns a domXPath object with all the current namespaces already registered.

Returns

DomXPath The object

5.17.3.3 FedoraRelationships::getXpathResults ($ *xpath_object,* $ *subject,* $ *predicate_uri,* $ *predicate,* $ *object,* $ *literal*) [protected]

This function is used to create an xpath expression based on the input.

Returns

DOMNodeList The node list

5.17.3.4 FedoraRelationships::internalAdd ($ *subject,* $ *predicate_uri,* $ *predicate,* $ *object,* $ *literal* = FALSE) [protected]

Add a new relationship.

Parameters

string	*$subject*	The subject. This can be a PID, or a PID/DSID combo. This string does not contain the info:fedora/ part of the URI this is added automatically.
string	*$predicate_-uri*	The URI to use as the namespace of the predicate. - If you would like the XML to use a prefix instead of the full predicate call the FedoraRelationships::registerNamespace() function first.
string	*$predicate*	The predicate tag to add.
string	*$object*	The object for the relationship that is being created.
boolean	*$literal*	Specifies if the object is a literal or not.

5.17.3.5 FedoraRelationships::internalGet ($ *subject*, $ *predicate_uri* = NULL, $ *predicate* = NULL, $ *object* = NULL, $ *literal* = FALSE) [protected]

This function queries the relationships in the assocaited datastream.

Any parameter except for $subject can be set to NULL to act as a wildcard. Calling with just $subject will return all relationships.

Parameters

string	*$subject*	The subject. This can be a PID, or a PID/DSID combo. This string does not contain the info:fedora/ part of the URI this is added automatically.
string	*$predicate_uri*	The URI to use as the namespace of the predicate. This is ignored if predicate is NULL.
string	*$predicate*	The predicate tag to filter by.
string	*$object*	The object for the relationship to filter by.
boolean	*$literal*	Defines if the $object is a literal or not.

Returns

array This returns an indexed array with all the matching relationships. The array is of the form:

```
Array
(
    [0] => Array
        (
            [predicate] => Array
                (
                    [value] => thepredicate
                    [alias] => thexmlprefix
                    [namespace] => http://crazycool.com#
                )

            [object] => Array
                (
                    [literal] => TRUE
                    [value] => test
                )

        )

)
```

5.17.3.6 FedoraRelationships::internalRemove ($ *subject*, $ *predicate_uri*, $ *predicate*, $ *object*, $ *literal* = FALSE) [protected]

This function removes relationships that match the pattern from the datastream.

Any parameter can be given as NULL which will make it a wildcard.

Parameters

string	*$subject*	The subject. This can be a PID, or a PID/DSID combo. This string does not contain the info:fedora/ part of the URI this is added automatically.
string	*$predicate_-uri*	The URI to use as the namespace of the predicate. This is ignored if predicate is NULL.
string	*$predicate*	The predicate tag to filter removed results by.
string	*$object*	The object for the relationship to filter by.
boolean	*$literal*	Defines if the $object is a literal or not.

Returns

boolean TRUE if relationships were removed, FALSE otherwise.

5.17.3.7 FedoraRelationships::registerNamespace ($ *alias,* $ *uri*)

Add a new namespace to the relationship xml.

Doing this before adding new predicates with differnt URIs makes the XML look a little prettier.

Parameters

string	*$alias*	The alias to add.
string	*$uri*	The URI to associate with the alias.

5.17.4 Member Data Documentation

5.17.4.1 FedoraRelationships::$namespaces [protected]

Initial value:

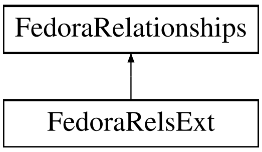
```
array(
   'rdf' => RDF_URI,
  )
```

The documentation for this class was generated from the following file:

- FedoraRelationships.php

5.18 FedoraRelsExt Class Reference

Inheritance diagram for FedoraRelsExt:



Public Member Functions

- __construct (AbstractFedoraObject $object)

    Objects Construct!
- add ($predicate_uri, $predicate, $object, $literal=FALSE)

    Add a new relationship.
- remove ($predicate_uri=NULL, $predicate=NULL, $object=NULL, $literal=FALSE)

    This function removes relationships that match the pattern from the datastream.
- get ($predicate_uri=NULL, $predicate=NULL, $object=NULL, $literal=FALSE)

    This function queries the relationships in the assocaited datastream.

Protected Member Functions

- initializeDatastream ()

    Initialize the datastrem that we are using.

Protected Attributes

- **$new** = FALSE
- **$initialized** = FALSE

5.18.1 Constructor & Destructor Documentation

5.18.1.1 FedoraRelsExt::__construct (AbstractFedoraObject $ *object*)

Objects Construct!

Parameters

-AbstractFedoraObject	*$object*	The object whose relationships we are manipulating

5.18.2 Member Function Documentation

5.18.2.1 FedoraRelsExt::add ($ *predicate_uri*, $ *predicate*, $ *object*, $ *literal* = FALSE)

Add a new relationship.

Parameters

string	*$predicate_uri*	The URI to use as the namespace of the predicate. - If you would like the XML to use a prefix instead of the full predicate call the FedoraRelationships::registerNamespace() function first.
string	*$predicate*	The predicate tag to add.
string	*$object*	The object for the relationship that is being created.
boolean	*$literal*	Specifies if the object is a literal or not.

5.18.2.2 FedoraRelsExt::get ($ *predicate_uri* = NULL, $ *predicate* = NULL, $ *object* = NULL, $ *literal* = FALSE)

This function queries the relationships in the assocaited datastream.

Any parameter except for $subject can be set to NULL to act as a wildcard. Calling with just $subject will return all relationships.

Parameters

string	*$predicate_uri*	The URI to use as the namespace of the predicate. This is ignored if predicate is NULL.
string	*$predicate*	The predicate tag to filter by.
string	*$object*	The object for the relationship to filter by.
boolean	*$literal*	Defines if the $object is a literal or not.

Returns

array This returns an indexed array with all the matching relationships. The array is of the form:

```
Array
(
    [0] => Array
        (
            [predicate] => Array
                (
                    [value] => thepredicate
                    [alias] => thexmlprefix
                    [namespace] => http://crazycool.com#
                )

            [object] => Array
                (
                    [literal] => TRUE
                    [value] => test
                )

        )
```

)

5.18.2.3 FedoraRelsExt::initializeDatastream ( ) [protected]

Initialize the datastrem that we are using.

We use this function to delay this as long as possible, in case it never has be be called.

5.18.2.4 FedoraRelsExt::remove ($ *predicate_uri* = NULL, $ *predicate* = NULL, $ *object* = NULL, $ *literal* = FALSE)

This function removes relationships that match the pattern from the datastream.

Any parameter can be given as NULL which will make it a wildcard.

Parameters

string	*$predicate_uri*	The URI to use as the namespace of the predicate. This is ignored if predicate is NULL.
string	*$predicate*	The predicate tag to filter removed results by.
string	*$object*	The object for the relationship to filter by.
boolean	*$literal*	Defines if the $object is a literal or not.

Returns

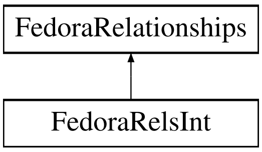
boolean TRUE if relationships were removed, FALSE otherwise.

The documentation for this class was generated from the following file:

- FedoraRelationships.php

5.19 FedoraRelsInt Class Reference

Inheritance diagram for FedoraRelsInt:



Public Member Functions

- __construct (AbstractFedoraDatastream $datastream)

  Objects Construct!

- add ($predicate_uri, $predicate, $object, $literal=FALSE)

  Add a new relationship.
- remove ($predicate_uri=NULL, $predicate=NULL, $object=NULL, $literal=FALSE)

  This function removes relationships that match the pattern from the datastream.
- get ($predicate_uri=NULL, $predicate=NULL, $object=NULL, $literal=FALSE)

  This function queries the relationships in the assocaited datastream.

Protected Member Functions

- initializeDatastream ()

  Delay initialization by waiting to set datastream with this function.

Protected Attributes

- **$new** = FALSE
- **$initialized** = FALSE
- **$aboutDs**

5.19.1 Constructor & Destructor Documentation

5.19.1.1 FedoraRelsInt::__construct (AbstractFedoraDatastream $ *datastream*)

Objects Construct!

Parameters

-Abstract-Fedora-Object	*$datastream*	The datastream whose relationships we are manipulating

5.19.2 Member Function Documentation

5.19.2.1 FedoraRelsInt::add ($ *predicate_uri,* $ *predicate,* $ *object,* $ *literal* = FALSE)

Add a new relationship.

Parameters

string	*$predicate_uri*	The URI to use as the namespace of the predicate. -If you would like the XML to use a prefix instead of the full predicate call the FedoraRelationships::registerNamespace() function first.
string	*$predicate*	The predicate tag to add.
string	*$object*	The object for the relationship that is being created.

boolean	*$literal*	Specifies if the object is a literal or not.

5.19.2.2 FedoraRelsInt::get (*$ predicate_uri* = NULL, *$ predicate* = NULL, *$ object* = NULL, *$ literal* = FALSE)

This function queries the relationships in the assocaited datastream.

Any parameter except for $subject can be set to NULL to act as a wildcard. Calling with just $subject will return all relationships.

Parameters

string	*$predicate_-uri*	The URI to use as the namespace of the predicate. This is ignored if predicate is NULL.
string	*$predicate*	The predicate tag to filter by.
string	*$object*	The object for the relationship to filter by.
boolean	*$literal*	Defines if the $object is a literal or not.

Returns

array This returns an indexed array with all the matching relationships. The array is of the form:

```
Array
(
    [0] => Array
        (
            [predicate] => Array
                (
                    [value] => thepredicate
                    [alias] => thexmlprefix
                    [namespace] => http://crazycool.com#
                )

            [object] => Array
                (
                    [literal] => TRUE
                    [value] => test
                )

        )

)
```

5.19.2.3 FedoraRelsInt::remove (*$ predicate_uri* = NULL, *$ predicate* = NULL, *$ object* = NULL, *$ literal* = FALSE)

This function removes relationships that match the pattern from the datastream.

Any parameter can be given as NULL which will make it a wildcard.

Parameters

string	*$predicate_-uri*	The URI to use as the namespace of the predicate. This is ignored if predicate is NULL.
string	*$predicate*	The predicate tag to filter removed results by.
string	*$object*	The object for the relationship to filter by.
boolean	*$literal*	Defines if the $object is a literal or not.

Returns

boolean TRUE if relationships were removed, FALSE otherwise.

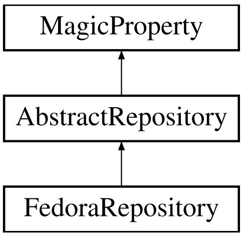
The documentation for this class was generated from the following file:

- FedoraRelationships.php

5.20 FedoraRepository Class Reference

Concrete implementation of the AbstractRepository for Fedora.

Inheritance diagram for FedoraRepository:



Public Member Functions

- __construct (FedoraApi $api, AbstractCache $cache)

  Constructor for the FedoraRepository Object.
- findObjects (array $search)
- constructObject ($id=NULL)
- ingestObject (NewFedoraObject &$object)
- getObject ($id)
- purgeObject ($id)

Public Attributes

- **$ri**

Protected Attributes

- **$cache**

5.20.1 Detailed Description

Concrete implementation of the AbstractRepository for Fedora.

The parent class has more detailed documentation about how this class can be called as an Array.

See also

AbstractRepository

5.20.2 Constructor & Destructor Documentation

5.20.2.1 FedoraRepository::__construct (FedoraApi $ *api*, AbstractCache $ *cache*)

Constructor for the FedoraRepository Object.

Parameters

FedoraApi	*$api*	An instantiated FedoraAPI which will be used to connect to the repository.
- AbstractCache	*$cache*	An instantiated AbstractCache which will be used to cache fedora objects.

5.20.3 Member Function Documentation

5.20.3.1 FedoraRepository::constructObject ($ *id* = `NULL`)

Todo validate the ID

catch the getNextPid errors

See also

AbstractRepository::constructObject

Reimplemented from AbstractRepository.

5.20.3.2 FedoraRepository::findObjects (array $ *search*)

See also

AbstractRepository::findObjects

Reimplemented from AbstractRepository.

5.20.3.3 FedoraRepository::getObject ($ *id*)

See also

AbstractRepository::ingestObject()

Todo perhaps we should check if an object exists instead of catching the exception

Reimplemented from AbstractRepository.

5.20.3.4 FedoraRepository::ingestObject (NewFedoraObject &$ *object*)

See also

AbstractRepository::ingestObject()

Todo error handling

Reimplemented from AbstractRepository.

5.20.3.5 FedoraRepository::purgeObject ($ *id*)

See also

AbstractRepository::purgeObject()

Reimplemented from AbstractRepository.

The documentation for this class was generated from the following file:

- Repository.php

5.21 FoxmlDocument Class Reference

Public Member Functions

- **__construct** (NewFedoraObject $object)
- createDocumentDatastreams ()

  Checks that the content and dsid are valid and then passes the FOXML creation off to the relevant function.

Public Attributes

- const **FOXML** = 'info:fedora/fedora-system:def/foxml#'
- const **xlink** = 'http://www.w3.org/1999/xlink'
- const **xsi** = 'http://www.w3.org/2001/XMLSchema-instance'
- const **xmlns** = 'http://www.w3.org/2000/xmlns/'
- const **rdf** = 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'
- const **rdfs** = 'http://www.w3.org/2000/01/rdf-schema#'
- const **fedora** = 'info:fedora/fedora-system:def/relations-external#'
- const **dc** = 'http://purl.org/dc/elements/1.1/'
- const **oai_dc** = 'http://www.openarchives.org/OAI/2.0/oai_dc/'
- const **fedora_model** = 'info:fedora/fedora-system:def/model#'

Protected Attributes

- **$root**
- **$object**

5.21.1 Member Function Documentation

5.21.1.1 FoxmlDocument::createDocumentDatastreams ( )

Checks that the content and dsid are valid and then passes the FOXML creation off to the relevant function.

Currently any 'string' content that is marked as a managed datastream will be ingested as inline.

Todo Implement fedora upload function to allow strings to be added as managed datastreams

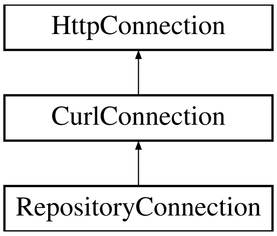
The documentation for this class was generated from the following file:

- FoxmlDocument.php

5.22 HttpConnection Class Reference

Abstract class defining functions for HTTP connections.

Inheritance diagram for HttpConnection:



Public Member Functions

- postRequest ($url, $type= 'none', $data=NULL, $content_type=NULL)

  Post a request to the server.
- getRequest ($url)

  Send a HTTP GET request to URL.
- putRequest ($url, $type= 'none', $file=NULL)

  Send a HTTP PUT request to URL.

Public Attributes

- **$cookies** = TRUE
- **$username** = NULL
- **$password** = NULL
- **$verifyHost** = TRUE
- **$verifyPeer** = TRUE
- **$timeout** = NULL
- **$connectTimeout** = 5
- **$userAgent** = NULL
- **$reuseConnection** = TRUE
- **$debug** = FALSE

5.22.1 Detailed Description

Abstract class defining functions for HTTP connections.

5.22.2 Member Function Documentation

5.22.2.1 HttpConnection::getRequest ($ *url*) `[abstract]`

Send a HTTP GET request to URL.

Parameters

string	*$url*	The URL to post the request to. Should start with the protocol. For example: `http://`.

Exceptions

HttpConnectionException	

Returns

> array Associative array containing: ∗ $return['status'] = The HTTP status code ∗ $return['headers'] = The HTTP headers of the reply ∗ $return['content'] = The body of the HTTP reply

Reimplemented in CurlConnection, and RepositoryConnection.

5.22.2.2 HttpConnection::postRequest (*$ url*, *$ type* = 'none', *$ data* = NULL, *$ content_type* = NULL) [abstract]

Post a request to the server.

This is primarily used for sending files.

Todo Test this for posting general form data. (Other then files.)

Parameters

string	*$url*	The URL to post the request to. Should start with the protocol. For example: `http://`.
string	*$type*	This paramerter must be one of: string, file.
string	*$data*	What this parameter contains is decided by the $type parameter.
string	*$content_type*	The content type header to set for the post request.

Exceptions

HttpConnectionException	

Returns

> array Associative array containing: ∗ $return['status'] = The HTTP status code ∗ $return['headers'] = The HTTP headers of the reply ∗ $return['content'] = The body of the HTTP reply

Reimplemented in CurlConnection, and RepositoryConnection.

5.22.2.3 HttpConnection::putRequest ($ *url*, $ *type* = 'none', $ *file* = NULL) [abstract]

Send a HTTP PUT request to URL.

Parameters

string	*$url*	The URL to post the request to. Should start with the protocol. For example: `http://`.
string	*$type*	This paramerter must be one of: string, file, none.
string	*$data*	What this parameter contains is decided by the $type parameter.

Exceptions

HttpConnection-Exception	

Returns

array Associative array containing: ∗ $return['status'] = The HTTP status code ∗ $return['headers'] = The HTTP headers of the reply ∗ $return['content'] = The body of the HTTP reply

Reimplemented in CurlConnection, and RepositoryConnection.

The documentation for this class was generated from the following file:

- HttpConnection.php

5.23 HttpConnectionException Class Reference

HTTP Exception.

Public Member Functions

- __construct ($message, $code, $response=NULL, $previous=NULL)

  The constructor for the exception.
- getResponse ()

  Get the HTTP response that caused the exception.

Protected Attributes

- **$response**

5.23.1 Detailed Description

HTTP Exception.

This is thrown when a status code other then 2XX is returned.

Parameters

string	*$message*	A message describing the exception.
int	*$code*	The error code. These are often the HTTP status codes, however less then 100 is defined by the class extending -HttpConnection, for eample cURL.
array	*$response*	The array containing: status, headers, and content of the HTTP request causing the error. This is only set if there was a HTTP response sent.

5.23.2 Constructor & Destructor Documentation

5.23.2.1 HttpConnectionException::__construct ($ *message*, $ *code*, $ *response* = NULL, $ *previous* = NULL)

The constructor for the exception.

Adds a response field.

Parameters

string	*$message*	The error message
int	*$code*	The error code
array	*$response*	The HTTP response
-Exception	*$previous*	The previous exception in the chain

5.23.3 Member Function Documentation

5.23.3.1 HttpConnectionException::getResponse ( )

Get the HTTP response that caused the exception.

Returns

> array Array containing the HTTP response. It has three keys: status, headers and content.

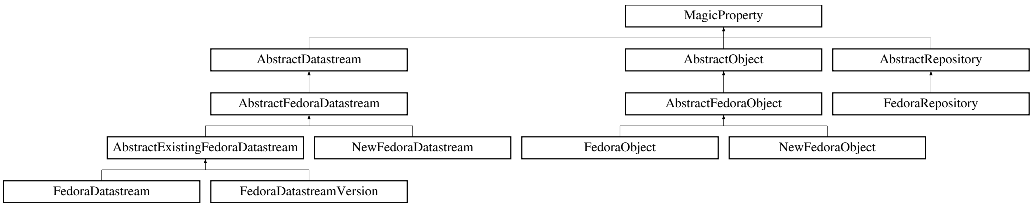
The documentation for this class was generated from the following file:

- HttpConnection.php

5.24 MagicProperty Class Reference

This abstract class allows us to implement PHP magic properties by defining a private method in the class that entends it.

Inheritance diagram for MagicProperty:



Public Member Functions

- __get ($name)

  This implements the PHP __get function which is utilized for reading data from inaccessible properties.
- __isset ($name)

  This implments the PHP __isset function which is utilized for testing if data in inaccessable properties is set.
- __set ($name, $value)

  This implements the PHP __set function which is utilized for setting inaccessable properties.
- __unset ($name)

  This implements the PHP __unset function which is utilized for unsetting inaccessable properties.

Protected Member Functions

- getGeneralMagicPropertyMethodName ($name)

  Returns the name of the magic property.

5.24.1 Detailed Description

This abstract class allows us to implement PHP magic properties by defining a private method in the class that entends it.

It attemtps to make the magic properties behave as much like normal PHP properties as possible.

This code lets the user define a new method that will be called when a property is accessed. Any method that ends in MagicProperty is code that implements a magic property.

Usage Example

```
class MyClass extends MagicProperty {
  private $secret;

  protected function myExampleMagicProperty($function, $value) {
    switch($function) {
      case 'set':
        $secret = $value;
        return;
      case 'get':
        return $secret;
      case 'isset':
        return isset($secret);
      case 'unset':
        return unset($secret);
    }
  }
}

$test = new MyClass();
$test->myExample = 'woot';
print($test->myExample);
```

5.24.2 Member Function Documentation

5.24.2.1 MagicProperty::__get ($ *name*)

This implements the PHP __get function which is utilized for reading data from inaccessible properties.

It wraps it by calling the appropriatly named method in the inherteting class. http://php.net/manual/en/language.oop5.overloading.php

Parameters

string	*$name*	The name of the function being called.

Returns

void The data returned from the property.

5.24.2.2 MagicProperty::__isset ($ *name*)

This implments the PHP __isset function which is utilized for testing if data in inaccessable properties is set.

This function calls the approprietly named method in the inhereting class. http://php.net/manual/en/language.oop5.overloading.php

Parameters

string	*$name*	The name of the function being called.

Returns

boolean If the variable is set.

5.24.2.3 MagicProperty::__set ($ *name*, $ *value*)

This implements the PHP __set function which is utilized for setting inaccessable properties.

http://php.net/manual/en/language.oop5.overloading.php

Parameters

string	*$name*	The property to set.
void	*$value*	The value it should be set with.

5.24.2.4 MagicProperty::__unset ($ *name*)

This implements the PHP __unset function which is utilized for unsetting inaccessable properties.

http://php.net/manual/en/language.oop5.overloading.php

Parameters

string	*$name*	The property to unset

5.24.2.5 MagicProperty::getGeneralMagicPropertyMethodName ($ *name*) `[protected]`

Returns the name of the magic property.

Makes it easy to change what we use as the name.

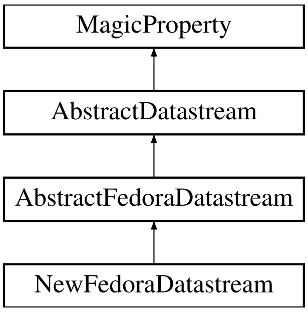
The documentation for this class was generated from the following file:

- MagicProperty.php

5.25 NewFedoraDatastream Class Reference

This defines a new fedora datastream.

Inheritance diagram for NewFedoraDatastream:



Public Member Functions

- __construct ($id, $control_group, AbstractFedoraObject $object, FedoraRepository $repository)

  The constructor for a new fedora datastream.
- setContentFromFile ($file)
- setContentFromUrl ($url)
- setContentFromString ($string)

Protected Member Functions

- validateControlGroup ($value)

  Validates and normalizes the control group.
- validateType ($type)

  Validates and normalizes the contentType.
- controlGroupMagicProperty ($function, $value)
- stateMagicProperty ($function, $value)
- labelMagicProperty ($function, $value)
- versionableMagicProperty ($function, $value)
- mimetypeMagicProperty ($function, $value)
- formatMagicProperty ($function, $value)
- checksumMagicProperty ($function, $value)
- checksumTypeMagicProperty ($function, $value)
- contentTypeMagicProperty ($function, $value)

  Magic Property that creates NewFedoraDatastream->contentType.
- contentMagicProperty ($function, $value)
- urlMagicProperty ($function, $value)

5.25.1 Detailed Description

This defines a new fedora datastream.

This is the class used to contain the inforamtion for a new fedora datastream before it is ingested.

5.25.2 Constructor & Destructor Documentation

5.25.2.1 NewFedoraDatastream::__construct ($ *id*, $ *control_group*, AbstractFedoraObject $ *object*, FedoraRepository $ *repository*)

The constructor for a new fedora datastream.

Parameters

string	*$id*	The unique identifier of the DS.
FedoraObject	*$object*	The FedoraObject that this DS belongs to.
FedoraRepository	*$repository*	The FedoraRepository that this DS belongs to.
string	*$control_group*	The control group this DS will belong to.

Todo test for valid identifiers. it can't start with a number etc.

5.25.3 Member Function Documentation

5.25.3.1 NewFedoraDatastream::checksumMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::checksum

Todo this should be refined a bit

5.25.3.2 NewFedoraDatastream::checksumTypeMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::checksumType

5.25.3.3 NewFedoraDatastream::contentMagicProperty ($ *function*, $ *value*) [protected]

See also

AbstractDatastream::content

5.25.3.4 NewFedoraDatastream::contentTypeMagicProperty ($ *function,* $ *value*) [protected]

Magic Property that creates NewFedoraDatastream->contentType.

This getter/setter lets us access what type of datastream this is so that we can ingest it properly.

5.25.3.5 NewFedoraDatastream::controlGroupMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractDatastream::controlGroup

Reimplemented from AbstractFedoraDatastream.

5.25.3.6 NewFedoraDatastream::formatMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractDatastream::format

5.25.3.7 NewFedoraDatastream::labelMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractDatastream::label

5.25.3.8 NewFedoraDatastream::mimetypeMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractDatastream::mimetype

5.25.3.9 NewFedoraDatastream::setContentFromFile ($ *file*)

See also

AbstractDatastream::setContentFromFile

Reimplemented from AbstractDatastream.

5.25.3.10 NewFedoraDatastream::setContentFromString ($ *string*)

See also

AbstractDatastream::setContentFromString

Reimplemented from AbstractDatastream.

5.25.3.11 NewFedoraDatastream::setContentFromUrl ($ *url*)

See also

AbstractDatastream::setContentFromUrl

Reimplemented from AbstractDatastream.

5.25.3.12 NewFedoraDatastream::stateMagicProperty ($ *function,* $ *value*) `[protected]`

See also

AbstractDatastream::state

5.25.3.13 NewFedoraDatastream::urlMagicProperty ($ *function,* $ *value*) `[protected]`

See also

AbstractDatastream::url

5.25.3.14 NewFedoraDatastream::validateControlGroup ($ *value*) `[protected]`

Validates and normalizes the control group.

Parameters

string	*$value*	The passed in control group.

Returns

mixed The stirng for hte controlgroup or FALSE if validation fails.

5.25.3.15 NewFedoraDatastream::validateType ($ *type*) `[protected]`

Validates and normalizes the contentType.

Parameters

string	*$type*	The passed in value for type.

Returns

mixed The stirng for the type or FALSE if validation fails.

5.25.3.16 NewFedoraDatastream::versionableMagicProperty ($ *function,* $ *value*) [protected]

See also

AbstractDatastream::versionable

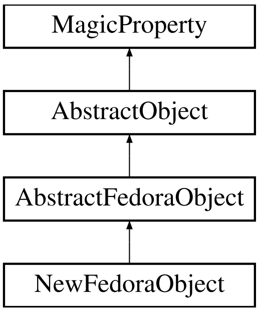
The documentation for this class was generated from the following file:

- Datastream.php

5.26 NewFedoraObject Class Reference

This represents a new fedora object that hasn't been ingested yet.

Inheritance diagram for NewFedoraObject:



Public Member Functions

- __construct ($id, FedoraRepository $repository)

  Constructoman!
- constructDatastream ($id, $control_group= 'M')
- ingestDatastream (&$ds)

  This function doesn't actually ingest the datastream, it just adds it to the queue to be ingested whenever this object is ingested.
- purgeDatastream ($id)
- getDatastream ($id)

- count ()
- offsetExists ($offset)
- offsetGet ($offset)
- offsetSet ($offset, $value)
- offsetUnset ($offset)
- getIterator ()

Protected Attributes

- **$datastreams** = array()

5.26.1 Detailed Description

This represents a new fedora object that hasn't been ingested yet.

It lets us pass around the object and add datastreams before we ingest it. This shouldn't be constructed on its own, it should only be created from a factory class.

5.26.2 Member Function Documentation

5.26.2.1 NewFedoraObject::constructDatastream ($ *id*, $ *control_group* = 'M')

See also

AbstractObject::constructDatastream()

Reimplemented from AbstractFedoraObject.

5.26.2.2 NewFedoraObject::count ( )

See also

Countable::count

5.26.2.3 NewFedoraObject::getDatastream ($ *id*)

See also

AbstractObject::getDatastream()

Reimplemented from AbstractObject.

5.26.2.4 NewFedoraObject::getIterator ( )

See also

IteratorAggregate::getIterator()

5.26.2.5 NewFedoraObject::ingestDatastream (&$ *ds*)

This function doesn't actually ingest the datastream, it just adds it to the queue to be ingested whenever this object is ingested.

See also

AbstractObject::ingestDatastream()

Parameters

NewFedoraDatastream	*$ds*	the datastream to be ingested

Returns

mixed FALSE on failure otherwise a instantiated FedoraDatastream.

Reimplemented from AbstractObject.

5.26.2.6 NewFedoraObject::offsetExists ($ *offset*)

See also

ArrayAccess::offsetExists

5.26.2.7 NewFedoraObject::offsetGet ($ *offset*)

See also

ArrayAccess::offsetGet

5.26.2.8 NewFedoraObject::offsetSet ($ *offset,* $ *value*)

See also

ArrayAccess::offsetSet

5.26.2.9 NewFedoraObject::offsetUnset ($ *offset*)

See also

ArrayAccess::offsetUnset

5.26.2.10 NewFedoraObject::purgeDatastream ($ *id*)

See also

AbstractObject::purgeDatastream()

Reimplemented from AbstractObject.

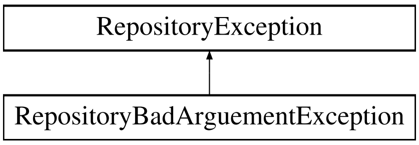
The documentation for this class was generated from the following file:

- Object.php

5.27 RepositoryBadArguementException Class Reference

This is thrown when a bad arguement is passed.

Inheritance diagram for RepositoryBadArguementException:



5.27.1 Detailed Description

This is thrown when a bad arguement is passed.

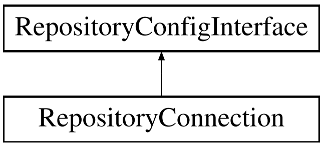
The documentation for this class was generated from the following file:

- RepositoryException.php

5.28 RepositoryConfigInterface Interface Reference

The general interface for a RepositoryConfig object.

Inheritance diagram for RepositoryConfigInterface:



Public Member Functions

- __construct ($url, $username, $password)

  Simple constructor defintion for the repository.

5.28.1 Detailed Description

The general interface for a RepositoryConfig object.

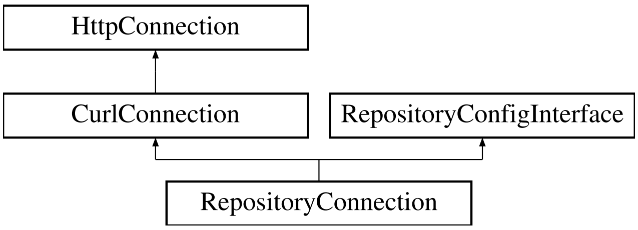
The documentation for this interface was generated from the following file:

- RepositoryConnection.php

5.29 RepositoryConnection Class Reference

Specific RepositoryConfig implementation that extends the CurlConnection class so that we can do specific processing on Curl requests for Fedora.

Inheritance diagram for RepositoryConnection:



Public Member Functions

- __construct ($url= 'http://localhost:8080/fedora', $username=NULL, $password=NULL)

  This constructor for the RepositoryConnection.
- addParamArray (&$request, &$seperator, $params, $name)

  These functions are used a lot when connecting to Fedora to create the correct arguements for REST calls.
- addParam (&$request, &$seperator, $name, $value)

This function adds a specific parameter to a RESTful request.

- getRequest ($url)

  Do a get request.
- postRequest ($url, $type= 'none', $data=NULL, $content_type=NULL)
- putRequest ($url, $type= 'none', $file=NULL)
- deleteRequest ($url)

Public Attributes

- **$url**
- **$username**
- **$password**

Protected Member Functions

- buildUrl ($url)

  This private function makes a fedora URL from a noraml URL.
- parseFedoraExceptions ($e)

  This function attempts to parse the exceptions that are recieved from Fedora into something more reasonable.

5.29.1 Detailed Description

Specific RepositoryConfig implementation that extends the CurlConnection class so that we can do specific processing on Curl requests for Fedora.

This also wraps the exceptions thrown by Curl, so that we keep our exception encapsulation.

It also doesn't take urls but instead uses relative paths that it adds the fedora URL to. This makes is a bit easier to use. It also makes sure that we always send usernames and passwords.

5.29.2 Constructor & Destructor Documentation

5.29.2.1 RepositoryConnection::__construct ($ *url* = 'http://localhost-:8080/fedora', $ *username* = NULL, $ *password* = NULL)

This constructor for the RepositoryConnection.

Parameters

string	*$url*	The URL to Fedora.
string	*$username*	The username to connect with.
string	*$password*	The password to connect with.

Implements RepositoryConfigInterface.

5.29.3 Member Function Documentation

5.29.3.1 RepositoryConnection::addParam (&$ *request*, &$ *seperator*, $ *name*, $ *value*)

This function adds a specific parameter to a RESTful request.

It makes sure that PHP booleans are changes into true and false and that the parameters are properly URL encoded.

Parameters

string	*$request*	The request that is being built.
character	*$seperator*	This is a helper to make sure that the first arguement gets a ? and the rest of them get a &.
string	*$name*	The name of the parameter that is being added
string	*$value*	the value of hte parameter.

5.29.3.2 RepositoryConnection::addParamArray (&$ *request*, &$ *seperator*, $ *params*, $ *name*)

These functions are used a lot when connecting to Fedora to create the correct arguements for REST calls.

This will encode and add an array of arguements to a request URL.

Parameters

string	*$request*	The request that is being built.
character	*$seperator*	This is a helper to make sure that the first arguement gets a ? and the rest of them get a &.
array	*$params*	An array of parameters.
string	*$name*	The name of the parameter that we are adding.

5.29.3.3 RepositoryConnection::deleteRequest ($ *url*)

See also

CurlConnection::deleteRequest()

Reimplemented from CurlConnection.

5.29.3.4 RepositoryConnection::getRequest ($ *url*)

Do a get request.

Parameters

string	*$url*	The URL relative to the fedora path to use.

Returns

array The contents of the get request

See also

CurlConnection::getRequest()

Reimplemented from CurlConnection.

5.29.3.5 RepositoryConnection::parseFedoraExceptions ($ *e*) [protected]

This function attempts to parse the exceptions that are recieved from Fedora into something more reasonable.

This it very hard since really things like this are garbage in garbage out, but we do what we can.

Parameters

- Exception	*$e*	The exception being parsed

5.29.3.6 RepositoryConnection::postRequest ($ *url*, $ *type* = 'none', $ *data* = NULL, $ *content_type* = NULL)

See also

CurlConnection::postRequest()

Reimplemented from CurlConnection.

5.29.3.7 RepositoryConnection::putRequest ($ *url*, $ *type* = 'none', $ *file* = NULL)

See also

CurlConnection::putRequest()

Reimplemented from CurlConnection.

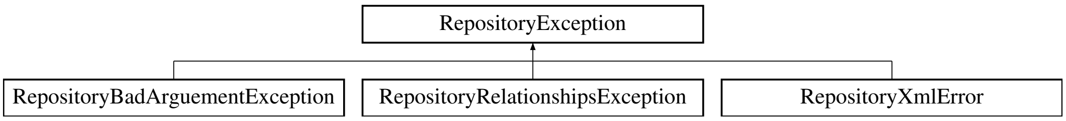
The documentation for this class was generated from the following file:

- RepositoryConnection.php

5.30 RepositoryException Class Reference

The top level exception for the Islandora Fedora API.

Inheritance diagram for RepositoryException:


RepositoryException
RepositoryBadArguementException
RepositoryRelationshipsException
RepositoryXmlError

5.30.1 Detailed Description

The top level exception for the Islandora Fedora API.

The documentation for this class was generated from the following file:

- RepositoryException.php

5.31 RepositoryQuery Class Reference

Public Member Functions

- __construct (RepositoryConnection $connection)

  Construct a new RI object.
- query ($query, $type= 'itql', $limit=-1, $offset=0)

  Performs the given Resource Index query and return the results.
- itqlQuery ($query, $limit=-1, $offset=0)

  Thin wrapper for self::_performRiQuery().
- sparqlQuery ($query, $limit=-1, $offset=0)

  Thin wrapper for self::performRiQuery().

Static Public Member Functions

- static parseSparqlResults ($sparql)

  Parse the passed in Sparql XML string into a more easily usable format.

Public Attributes

- **$connection**

Static Protected Member Functions

- static pidUriToBarePid ($uri)

  Utility function used in self::performRiQuery().

5.31.1 Constructor & Destructor Documentation

5.31.1.1 RepositoryQuery::__construct (RepositoryConnection $ *connection*)

Construct a new RI object.

Parameters

-Repository-Connection	*$connection*	The connection to connect to the RI with.

5.31.2 Member Function Documentation

5.31.2.1 RepositoryQuery::itqlQuery ($ *query*, $ *limit* = `-1`, $ *offset* = `0`)

Thin wrapper for self::_performRiQuery().

See also

self::performRiQuery()

5.31.2.2 static RepositoryQuery::parseSparqlResults ($ *sparql*) `[static]`

Parse the passed in Sparql XML string into a more easily usable format.

Parameters

string	*$sparql*	A string containing Sparql result XML.

Returns

array Indexed (numerical) array, containing a number of associative arrays, with keys being the same as the variable names in the query. URIs beginning with 'info:fedora/' will have this beginning stripped off, to facilitate their use as PIDs.

5.31.2.3 static RepositoryQuery::pidUriToBarePid ($ *uri*) `[static, protected]`

Utility function used in self::performRiQuery().

Strips off the 'info:fedora/' prefix from the passed in string.

Parameters

string	*$uri*	A string containing a URI.

Returns

string The input string less the 'info:fedora/' prefix (if it has it). The original string otherwise.

5.31.2.4 RepositoryQuery::query ($ *query*, $ *type* = 'itql', $ *limit* = -1, $ *offset* = 0)

Performs the given Resource Index query and return the results.

Parameters

string	*$query*	A string containing the RI query to perform.
string	*$type*	The type of query to perform, as used by the risearch interface.
int	*$limit*	An integer, used to limit the number of results to return.
int	*$offset*	An integer, used to offset the results (results should be ordered, to maintain consistency.

Returns

array Indexed (numerical) array, containing a number of associative arrays, with keys being the same as the variable names in the query. URIs beginning with 'info:fedora/' will have this beginning stripped off, to facilitate their use as PIDs.

5.31.2.5 RepositoryQuery::sparqlQuery ($ *query*, $ *limit* = -1, $ *offset* = 0)

Thin wrapper for self::performRiQuery().

See also

self::_performRiQuery()

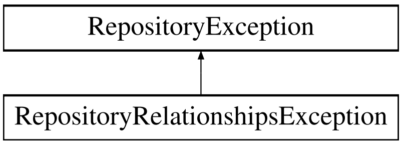
The documentation for this class was generated from the following file:

- RepositoryQuery.php

5.32 RepositoryRelationshipsException Class Reference

This is thrown from the relationships class when something goes wrong.

Inheritance diagram for RepositoryRelationshipsException:



5.32.1 Detailed Description

This is thrown from the relationships class when something goes wrong.

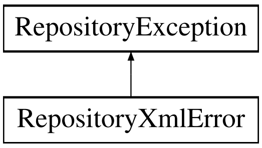
The documentation for this class was generated from the following file:

- RepositoryException.php

5.33 RepositoryXmlError Class Reference

This is thrown when there is an error parsing XML.

Inheritance diagram for RepositoryXmlError:



Public Member Functions

- __construct ($message, $code, $errors, $previous=NULL)

  Same as the default exception constructor except it takes another parameter errors, this is the error returned by the xml parser.

Public Attributes

- **$errors**

5.33.1 Detailed Description

This is thrown when there is an error parsing XML.

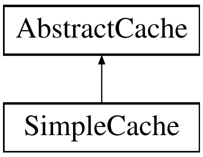
The documentation for this class was generated from the following file:

- RepositoryException.php

5.34 SimpleCache Class Reference

This is a simple cache that uses a static array to hold the cached values.

Inheritance diagram for SimpleCache:



Public Member Functions

- get ($key)
- add ($key, $data)
- set ($key, $data)
- delete ($key)

Static Public Member Functions

- static setCacheSize ($size)

  Set the cache size for the cache.
- static resetCache ()

  Reset the cache flushing it and returning it to its default size.

Public Attributes

- const **CACHESIZE** = 100

Static Protected Attributes

- static **$cache** = array()
- static **$entries** = array()
- static **$size** = SimpleCache::CACHESIZE

5.34.1 Detailed Description

This is a simple cache that uses a static array to hold the cached values.

This means that it will cache across instantiations in the same PHP runtime but not across runtimes. The cache has 100 slots and uses a simple LIFO caching strategy.

Todo Replace this with something more interesting like memcached

Try some other intersting caching strategies like LRU.

5.34.2 Member Function Documentation

5.34.2.1 SimpleCache::add ($ *key*, $ *data*)

See also

AbstractCache::add

Reimplemented from AbstractCache.

5.34.2.2 SimpleCache::delete ($ *key*)

See also

AbstractCache::delete

Reimplemented from AbstractCache.

5.34.2.3 SimpleCache::get ($ *key*)

See also

AbstractCache::get

Reimplemented from AbstractCache.

5.34.2.4 SimpleCache::set ($ *key*, $ *data*)

See also

AbstractCache::set

Reimplemented from AbstractCache.

5.34.2.5 static SimpleCache::setCacheSize ($ *size*) `[static]`

Set the cache size for the cache.

If the size if bigger the cache size is just made bigger. If its smaller, the cache is flushed and the cache size is made smaller.

Parameters

int	*$size*	The new size of the cache.

The documentation for this class was generated from the following file:

- Cache.php

Chapter 6

File Documentation

6.1 Cache.php File Reference

Simple set of classes defining caches.

Classes

- class AbstractCache

  Simple abstract Cache defintion providing basic key value caching functionality.
- class SimpleCache

  This is a simple cache that uses a static array to hold the cached values.

6.1.1 Detailed Description

Simple set of classes defining caches. An abstract cache is defined so that we can use a more interesting caching setup in the future like memcached or the native drupal cache.

6.2 Datastream.php File Reference

This file defines all the classes used to manipulate datastreams in the repository.

Classes

- class AbstractDatastream

  This abstract class can be overriden by anything implementing a datastream.
- class AbstractFedoraDatastream

  Abstract base class implementing a datastream in Fedora.

- class NewFedoraDatastream

  This defines a new fedora datastream.
- class AbstractExistingFedoraDatastream

  This abstract class defines some shared functionality between all classes that implement exising fedora datastreams.
- class FedoraDatastreamVersion

  This class implements an old version of a fedora datastream.
- class FedoraDatastream

  This class implements a fedora datastream.

6.2.1 Detailed Description

This file defines all the classes used to manipulate datastreams in the repository.

6.3 FedoraApi.php File Reference

The RAW API wrappers for the Fedora interface.

Classes

- class FedoraApi

  This is a simple class that brings FedoraApiM and FedoraApiA together.
- class FedoraApiA

  This class implements the Fedora API-A interface.
- class FedoraApiM

  This class implements the Fedora API-M interface.

6.3.1 Detailed Description

The RAW API wrappers for the Fedora interface. This file currently contains fairly raw wrappers around the Fedora REST interface. These could also be reinmplemented to use for example the Fedora SOAP interface. If there are version specific modifications to be made for Fedora, this is the place to make them.

6.4 FedoraApiSerializer.php File Reference

This file contains functions to serialize the XML responses from Fedora into something more easily dealt with in PHP.

Classes

- class FedoraApiSerializer

  A class to Serialize the XML responses from Fedora into PHP arrays.

6.4.1 Detailed Description

This file contains functions to serialize the XML responses from Fedora into something more easily dealt with in PHP.

6.5 FedoraRelationships.php File Reference

This file defines the classes that are used for manipulaing the fedora relationships datastreams.

Classes

- class FedoraRelationships

  This is the base class for Fedora Relationships.
- class FedoraRelsExt
- class FedoraRelsInt

Enumerations

- enum **XMLNS**
- enum **RDF_URI**
- enum **FEDORA_RELS_EXT_URI**
- enum **FEDORA_MODEL_URI**
- enum **ISLANDORA_RELS_EXT_URI**
- enum **ISLANDORA_RELS_INT_URI**

6.5.1 Detailed Description

This file defines the classes that are used for manipulaing the fedora relationships datastreams.

6.6 HttpConnection.php File Reference

This file defines the classes used to make HTTP requests.

Classes

- class HttpConnectionException

  HTTP Exception.
- class HttpConnection

  Abstract class defining functions for HTTP connections.
- class CurlConnection

  This class defines a abstract HttpConnection using the PHP cURL library.

6.6.1 Detailed Description

This file defines the classes used to make HTTP requests.

6.7 MagicProperty.php File Reference

This file contains the class MagicProperty.

Classes

- class MagicProperty

  This abstract class allows us to implement PHP magic properties by defining a private method in the class that entends it.

6.7.1 Detailed Description

This file contains the class MagicProperty.

6.8 Object.php File Reference

This file contains all of the functionality for objects in the repository.

Classes

- class AbstractObject

  An abstract class defining a Object in the repository.
- class AbstractFedoraObject

  This is the base class for a Fedora Object.
- class NewFedoraObject

  This represents a new fedora object that hasn't been ingested yet.
- class FedoraObject

  Represents a Fedora Object.

6.8.1 Detailed Description

This file contains all of the functionality for objects in the repository.

6.9 Repository.php File Reference

This file defines an abstract repository that can be overridden and also defines a concrete implementation for Fedora.

Classes

- class AbstractRepository

  An abstract repository interface.
- class FedoraRepository

  Concrete implementation of the AbstractRepository for Fedora.

6.9.1 Detailed Description

This file defines an abstract repository that can be overridden and also defines a concrete implementation for Fedora.

6.10 RepositoryConnection.php File Reference

This file contains the implementation of a connection to Fedora.

Classes

- interface RepositoryConfigInterface

  The general interface for a RepositoryConfig object.
- class RepositoryConnection

  Specific RepositoryConfig implementation that extends the CurlConnection class so that we can do specific processing on Curl requests for Fedora.

6.10.1 Detailed Description

This file contains the implementation of a connection to Fedora. And the interface for a repository configuration.

6.11 RepositoryException.php File Reference

Contains all of the exceptions thrown by Tuque.

Classes

- class RepositoryException

  The top level exception for the Islandora Fedora API.
- class RepositoryXmlError

  This is thrown when there is an error parsing XML.
- class RepositoryBadArguementException

  This is thrown when a bad arguement is passed.
- class RepositoryRelationshipsException

  This is thrown from the relationships class when something goes wrong.

6.11.1 Detailed Description

Contains all of the exceptions thrown by Tuque.

6.12 RepositoryQuery.php File Reference

This file provides some methods for doing RDF queries.

Classes

- class RepositoryQuery

6.12.1 Detailed Description

This file provides some methods for doing RDF queries. The essance of this file was taken from some commits that Adam Vessy made to Islandora 6.x, so I'd like to give him some credit here.