

Reptor Manual

Thomas Zastrow, 2016



Table of Content

1. What it is	3
2. How it works	3
3. Installation.....	3
4. Using Docker	4
5. Configuration	4
6. Persistent Identifiers (PIDs).....	4
7. Metadata	5
8. Evaluation.....	5
9. OAI-PMH.....	5
10. ResourceSync.....	5
11. Permissions	5
12. Cloud Storage Support	6

by Thomas Zastrow, 2016

1. What it is

Reptor is a PHP application which turns a webserver into a data repository. It demonstrates the functionality of a modern *data repository* along the recommendations of the Research Data Alliance (RDA)¹. Currently it offers:

- Untyped metadata handling by supporting several types of serialised metadata (plain text, key-value pairs, XML formats)
- PID handling, independent of the used PID system (DOI, Handle, URN etc.)
- Access to the object data (bitstreams) stored on the filesystem and taking care of permission management
- Integration of Data Type Registries (DTR)
- OAI-PMH
- ResourceSync (partially)

All data is stored in an arbitrary filesystem or a part of it. Only a PHP enabled webserver with access to this filesystem is necessary to run Reptor.

2. How it works

Reptor consists of a bunch of PHP files where **index.php** is the heart of the system. There are some helper files in subdirectories (CSS, images) as well as additional functionality in PHP files (oai.php etc.), but the logic for displaying content is done in „index.php“. Installation is quite simple: just put all the files on a PHP enabled webserver (you can clone the GitHub repository, see below). No database is required.

The file index.php is then exposing a subdirectory with the name „data“ and all its files and subdirectories in an arbitrary depth as data repository via the web. Also external filesystems which are mounted into „data“ are exposed - if the underlying webserver has the permissions to access the mounted filesystem (see chapter about cloud storage below).

So far, index.php is acting very similar to a web server which shows the content of its folders to a webbrowser. But in addition, Reptor checks any folder and subfolder if it contains metadata, PIDs and links to a Data Type Registry beside other files and folders.

¹<https://www.rd-alliance.org>

3. Installation

Simply clone the GitHub repository into an empty folder on a PHP enabled webserver:

```
git clone https://github.com/TomZastrow/reptor.git
```

Beside the basic Reptor files, this will also create some basic example entries in the “data” folder: this folder is holding the whole data, any other (external) filesystem can be mounted on any mountpoint below “data”.

4. Using Docker

The Reptor software needs nothing else than a PHP enabled webserver to run. On DockerHub, you will find a bunch of images which could be used. Here is one way:

- Clone the Reptor software as mentioned in the chapter before from GitHub to your local machine
- Download the “eboraas/apache-php” Docker image:

```
docker pull eboraas/apache-php
```

- Run the image and mount your local clone of the GitHub repository:

```
docker run -p 9999:80 -v /path/to/git-clone/:/var/www/html -d
eboraas/apache-php
```

Here, “9999” is the port on your host machine where the Reptor software is being made available. and “/path/to/git-clone” points to the directory where you made a clone of the Reptor GitHub repository. Now point your browser to:

```
http://localhost:9999
```

5. Configuration

At the beginning of the file „index.php“, some configurations can be set. Most of these settings are coming along with a senseful default setting but can be adjusted accordingly to individual needs:

\$dataTypeRegistry = "http://dtr.pidconsortium.eu:8081/objects/";

The URL to a data type registry: per default, GWDGs DTR is used

\$namePIDFile = "pid.txt";

The name of the file which contains the PID of the current directory

\$nameMetadataText = "metadata.txt";

The name of the file which contents is interpreted and displayed as metadata text

\$namesDataTypes = "metadata.types";

The name of the file which contains the data types, related to the data type registry

mentioned above, to describe the content of the current directory

\$nameDCFile = "metadata.dc";

This file contains Dublin Core metadata in Key-Value format

\$template = "default";

The CSS- and HTML template which is used to render the pages. Can be found in subfolder "templates".

\$showEvaluation = true;

Defines if an evaluation against the RDA specifications is shown (true) or not (false) at the bottom of the page

6. Persistent Identifiers (PIDs)

Reptor is currently not able to manage PIDs on its own. You need an external PID provider, which can be a Handle based service (for example DOI) or any other PID system like ARK². The PID should point to:

```
http://yourserver.com/path/to/reptor/subfolder/subfolder
```

where „subfolder/subfolder“ is the relative path to the folder you want to assign a PID to. Please note that the folder „data“ as the starting folder of the data repository is not part of the path. Also „index.php“ is not part because we assume that your server is configured in such a way that „index.php“ is delivered by default if no other filename is mentioned.

7. Metadata

Metadata has to be stored in files in any (sub-) directory of the „data“ dir. Depending on the filename, the metadata will be handled differently:

- **metadata.txt:** Its content will be displayed as it is in the „Metadata“ box of that directory. Any HTML, XML or whatever content is possible.
- **metadata.dc:** Key-value pairs as described in Dublin Core. Will be included in OAI-PMH output!
- **metadata.types:** semantic explanations of the content of the current directory. Makes use of the „Data Type Registry“ at GWDG

²A good starting point is the free-of-cost service Shortref (www.shortref.org)

8. OAI-PMH

The “Open Archive Initiative - Protocol for Metadata Harvesting” (OAI-PMH) is implemented in Reptor and can be used to offer the metadata for other repositories, catalogue systems etc. for harvesting. The Reptor implementation supports the following OAI-PMH verbs:

- Identify
- ListMetadataFormats
- ListSets
- ListIdentifiers
- ListRecords

The OAI-PMH implementation is complete, it validates against the OAI-PMH validator at oaipmh.com³. Currently, it only offers “Dublin Core” metadata for harvesting.

9. ResourceSync

ResourceSync⁴ is a specification for synchronization of resources between servers. Its implementation in Reptor is rudimentary.

10. Permissions

Permissions can be handled via Apache's .htaccess files: just put a correctly configured one in a Reptor's subdirectory and the Apache webserver will take care of the correct handling of its content.

Following RDA's recommendation: the metadata and the PID in this directory will be displayed always! Only accessing the content itself follows under the rules defined in .htaccess.

11. Cloud Storage Support

Reptor's data just has to be stored somewhere below its „data“ directory. Here, any other (remote) filesystem can be locally mounted. Reptor will then handle these filesystems just as local (sub-) directories. Make sure that the Apache webserver has the file permissions to access the mounted filesystems.

Via this mechanism, a user can add / edit data easily on a remote filesystem while any change is instantaneously visible in Reptor.

³ <http://validator.oaipmh.com/>

⁴ <http://www.openarchives.org/rs/1.0.9/resourcesync>

Reptor Manual