Updating a data package

This chapter will teach you how to edit and update an existing data package in R. Earlier, we updated metadata. In this section we will learn how to update a data file, and how to update a package by adding an additional data file.

Update a data object

To update a data file associated with a data package, you need to do three things:

1) update the object itself,
2) update the resource map (which affiliates the object with the metadata), and
3) update the metadata that describes that object

The `datapack::replaceMember` function takes care of the first two of these tasks. First you need to get the pid of the file you want to replace by using `datapack::selectMember`:

```
metadataId <- selectMember(dp, name="sysmeta@formatId", value="https://eml.ecoinformatics.
```

Then use `replaceMember`:

```
dp <- replaceMember(dp, metadataId, replacement=file_path)
```

If you want to remove some files from the data package we can use `datapack::removeMember`. If we wanted to remove all the zip files associated with this data package, we can use `datapack::removeMember`:

```
zipId <- selectMember(dp, name="sysmeta@formatId", value="application/vnd.shp+zip")
removeMember(dp, zipId, removeRelationships = T)
```

> **Note**
>
> You will need to be explicit about your `format_id` here based on the file type. A list of format IDs can be found here on the DataONE website. Use line 2 (Id:) exactly, character for character.

To accomplish the second task, you will need to update the metadata using the EML package. This is covered in Chapter 4. After you update a file, you will always need to update the metadata because parts of the `physical` section (such as the file size, checksum) will be different, and it may also require different attribute information.

Once you have updated your metadata and saved it, you can update the package itself.

Update a package with a new data object

Once you have updated the data objects and saved the metadata to a file, we can update the metadata and use `replaceMember` to update the package with the new metadata.

Make sure you have the package you want to update loaded into R using `dataone::getDataPackage()`.

Publish update

Now we can update your data package to include the new data object. Assuming you have updated your data package earlier something like the below:

```
d1c_test <- dataone::D1Client("STAGING", "urn:node:mnTestARCTIC")
packageId <- "the resource map"

dp <- getDataPackage(d1c_test, identifier=packageId, lazyLoad=TRUE, quiet=FALSE)
metadataId <- selectMember(dp, name="sysmeta@formatId", value="https://eml.ecoinformatics.

#some modification to the EML here

eml_path <- "path/to/your/saved/eml.xml"
write_eml(doc, eml_path)

dp <- replaceMember(dp, metadataId, replacement=eml_path)
```

You can then upload your data package:

```
myAccessRules <- data.frame(subject="CN=arctic-data-admins,DC=dataone,DC=org", permission=
packageId <- uploadDataPackage(d1c_test, dp, public=FALSE, accessRules=myAccessRules, quie
```

If a package is ready to be public, you can change the `public` argument in the `datapack::uploadDataPackage()` call to `TRUE`.

If you want to publish with a DOI (Digital Object Identifier) instead of a UUID (Universally Unique Identifier), you need to do this when replacing the metadata. **This should only be done after the package is finalized and has been thoroughly reviewed!**

```
doi <- dataone::generateIdentifier(d1c_test@mn, "DOI")
dp <- replaceMember(dp, metadataId, replacement=eml_path, newId=doi)

newPackageId <- uploadDataPackage(d1c_test, dp, public=TRUE, quiet=FALSE)
```

If there is a pre-issued DOI (researcher requested the DOI for the publication first), please do the following:

```
dp <- replaceMember(dp, metadataId, replacement=eml_path, newId="your pre-issued doi previ

newPackageId <- uploadDataPackage(d1c_test, dp, public=TRUE, quiet=FALSE)
```

If the package has children, see how to do this using `arcticdatautils::publish_update` in the nesting section of the reference manual.

Refresh the landing page at test.arcticdata.io/#view/... for this package and then follow the "newer version" link to view the latest.

Exercise 4

What if the researcher notices that some information needed to be updated in the data file? We can use `replaceMember` to do just that!

If you haven't already:

- Locate the data package you published in the previous exercise by navigating to the URL test.arcticdata.io/#view/...
- Load the package and EML into R using the above commands.

Make a slightly different data file to upload to `test.arcticdata.io` for this exercise:

- Load the data file associated with the package into R as a `data.frame`. (Hint: use `read.csv()` to upload the data file from your computer/the server.)
- Make an edit to the data in R (e.g. change one of the colnames to `"TEST"`).
- Save the edited data. (Hint: use `write.csv(data, row.names = FALSE)`.)

Upload the new csv file, get a new pid and publish those updates:

- Update the data file in the package with the edited data file using `replaceMember`.
- Update your package using `uploadDataPackage`.