LDP-PCDM-F4 In Action - Book

Books In Action

1: Final State - Book

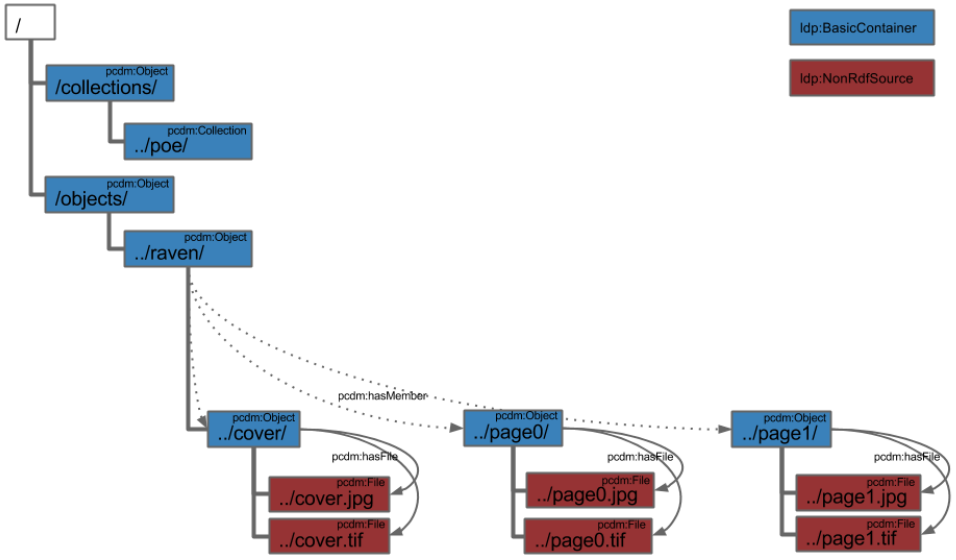
The ldp:BasicContainers are simply containers of other resources. BasicContainers can contain both other containers as well as ldp:NonRdfSources (or "binaries").
There are three PCDM types here:

- pcdm:Object
- pcdm:Collection
- pcdm:File

Additionally, there are two PCDM relationships that indicate resource membership and file membership:

- pcdm:hasMember
- pcdm:hasFile

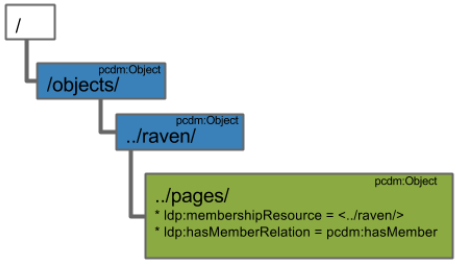
The descriptions of these resource types and relationships may be found in the detailed Portland Common Data Model page.



Book - Create DirectContainer

Book - Create DirectContainer

Here we will begin to walk through the mechanics of creating the structures that will facilitate creation of the book and its pages.



First, create the top-level "objects/" pcdm:Object, which is also an ldp:BasicContainer.

```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @pcdm-object.ttl localhost:8080/fcrepo/rest/objects/
```

Where "pcdm-object.ttl" follows:

pcdm-object.ttl

```
@prefix pcdm: <http://pcdm.org/models#>

<> a pcdm:Object .
```

Second, create the nested "raven/" pcdm:Object, which is also another ldp:BasicContainer.

```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @pcdm-object.ttl localhost:8080/fcrepo/rest/objects
/raven/
```

Lastly, create an ldp:DirectContainer, "pages/" that will facilitate the establishment of relationships between "raven/" and its constituent pages.

```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @ldp-direct.ttl localhost:8080/fcrepo/rest/objects
/raven/pages/
```

Where "ldp-direct.ttl" follows:

ldp-direct.ttl

```
@prefix ldp: <http://www.w3.org/ns/ldp#>
@prefix pcdm: <http://pcdm.org/models#>

<> a ldp:DirectContainer, pcdm:Object ;
  ldp:membershipResource </fcrepo/rest/objects/raven/> ;
  ldp:hasMemberRelation pcdm:hasMember .
```

An ldp:DirectContaner is an LDP construct that activates the creation of certain RDF triples when a new resource is added as a child of this container. Specifically, when a new resource is added inside of the "pages/" DirectContainer, a new triple on the ldp:membershipResource ("raven/") will be created with the predicate defined by the ldp:hasMemberRelation property ("pcdm:hasMember") and an object that is a reference to the new resource.

The auto-created triple resulting from the addition of a new child resource within "pages/" will take the form:

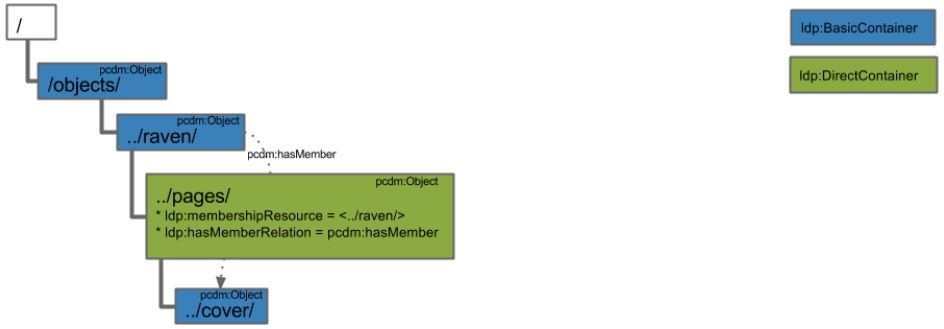
```
<http://localhost:8080/fcrepo/rest/objects/raven/> <pcdm:hasMember> <new-resource>
```

We will see this in action next!

Book - Create Cover

Book - Create cover

Create a new pcdm:Object, "cover/", that is also an ldp:BasicContainer within the "pages/" DirectContainer.




```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @pcdm-object.ttl localhost:8080/fcrepo/rest/objects
/raven/pages/cover/
```

Where "pcdm-object.ttl" follows:

pcdm-object.ttl

```
@prefix pcdm: <http://pcdm.org/models#>

<> a pcdm:Object .
```

As described in the previous step, the addition of "cover/" automatically creates the following new triple on "raven/"

```
<http://localhost:8080/fcrepo/rest/objects/raven/> pcdm:hasMember <http://localhost:8080/fcrepo/rest/objects
/raven/pages/cover/>
```

Restating from the previous step,

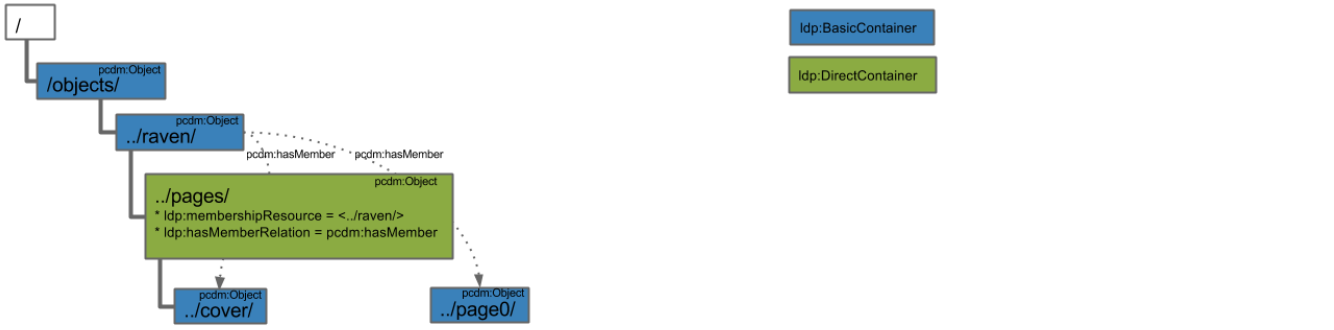
- the subject of the triple comes from the "ldp:membershipResource" defined on "pages/"
- the predicate of the triple comes from the "ldp:hasMemberRelation" defined on "pages/", and
- the object of the triple is the new resource ("cover/") that was added to the ldp:DirectContainer ("pages/")

Book - Create Page0

Book - Create Page0

In the same fashion as the previous step, adding "page0/" to the DirectContainer, "pages/" results in a new auto-generated triple on "raven/" of the form:

```
<http://localhost:8080/fcrepo/rest/objects/raven/> pcdm:hasMember <http://localhost:8080/fcrepo/rest/objects
/raven/pages/page0/>
```

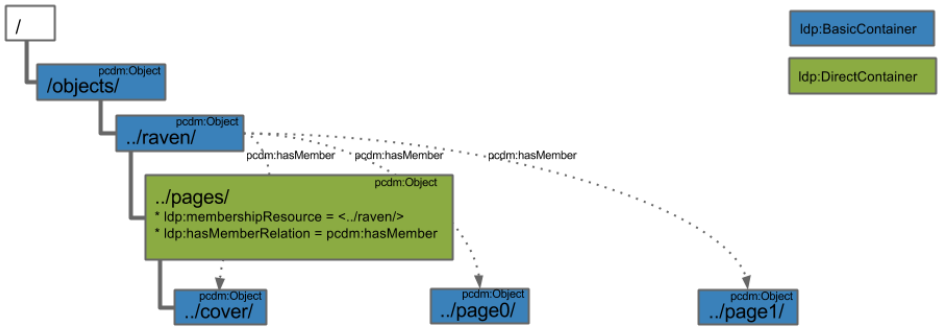



```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @pcdm-object.ttl localhost:8080/fcrepo/rest/objects
/raven/pages/page0/
```

Book - Create Page1

Book - Create Page1

This step in creating the final page, "page1/", follows the same pattern shown in the previous two steps.

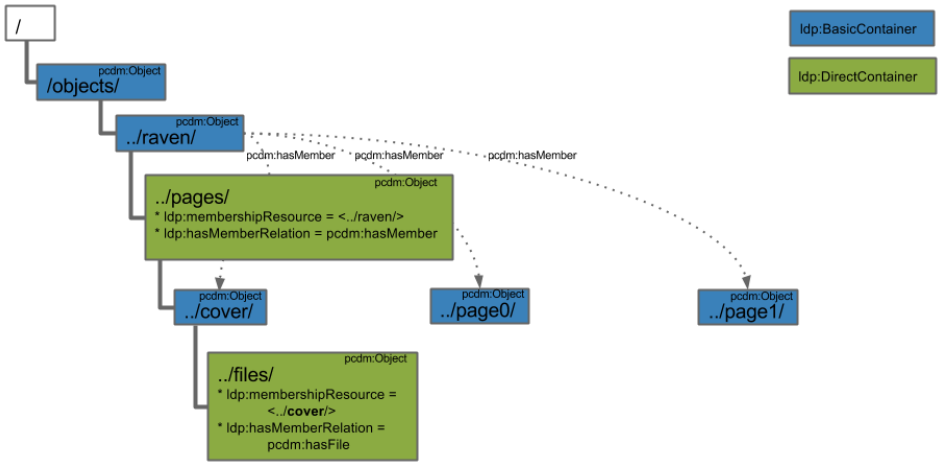



```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @pcdm-object.ttl localhost:8080/fcrepo/rest/objects
/raven/pages/page1/
```

Cover - Create DirectContainer

Cover - Create DirectContainer

In the same way that we used an ldp:DirectContainer to facilitate the auto-generation of triples linking "raven/" to each of the pages, now use the same pattern to auto-generate the creation of triples that link each page pcdm:Object to their various file representations.



To begin with, create an ldp:DirectContainer, "files/", which is also a pcdm:Object, as a child of "cover/" as follows:

```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @ldp-cover-direct.ttl localhost:8080/fcrepo/rest
/objects/raven/pages/cover/files/
```

Where "ldp-cover-direct.ttl" follows:

ldp-cover-direct.ttl

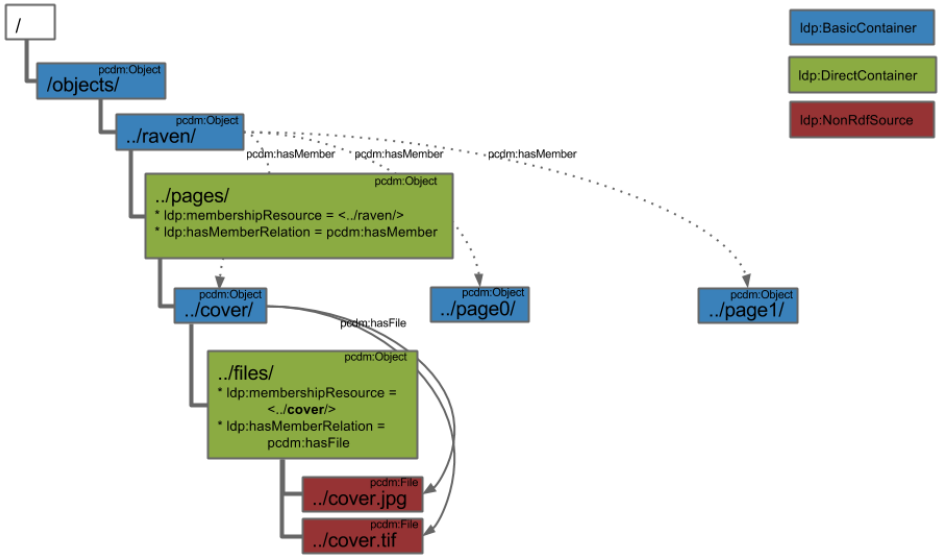
```
@prefix ldp: <http://www.w3.org/ns/ldp#>
@prefix pcdm: <http://pcdm.org/models#>

<> a ldp:DirectContainer, pcdm:Object ;
  ldp:membershipResource </fcrepo/rest/objects/raven/pages/cover/> ;
  ldp:hasMemberRelation pcdm:hasFile .
```

Now, any new resource that is added as a child of the DirectContainer "files/" will cause the auto-generation of a new triple on "cover/" that has a predicate of pcdm:hasFile and an object of the new resource.

Cover - Create Files

Cover - Create Files

Once again, we demonstrate the use of LDP in creating PCDM relationships simply as a result of repository interactions.



Add two pcdm:File resources to the DirectContainer, "files/" as follows:

```
curl -i -XPUT -H"Content-Type: image/jpeg" --data-binary @cover.jpg localhost:8080/fcrepo/rest/objects/raven
/pages/cover/files/cover.jpg
```

Where "cover.jpg" is attached.

If you perform a subsequent HTTP HEAD on this new resource, you will see there is a "Link" header of rel="describedby". Update the RDF metadata of the ldp:NonRdfSource to specify that the resource is a pcdm:File, as follows:

```
curl -i -XPATCH -H"Content-Type: application/sparql-update" --data-binary @pcdm-file.ru localhost:8080/fcrepo
/rest/objects/raven/pages/cover/files/cover.jpg/fcr:metadata
```

Where "pcdm-file.ru" follows:

pcdm-file.ru

```
PREFIX pcdm: <http://pcdm.org/models#>
INSERT {
  <> a pcdm:File
} WHERE {
}
```

Repeat for the attached TIFF, cover.tif

```
curl -i -XPUT -H"Content-Type: image/tiff" --data-binary @cover.tif localhost:8080/fcrepo/rest/objects/raven
/pages/cover/files/cover.tif
curl -i -XPATCH -H"Content-Type: application/sparql-update" --data-binary @pcdm-file.ru localhost:8080/fcrepo
/rest/objects/raven/pages/cover/files/cover.tif/fcr:metadata
```

After creating the two "cover" resources, an HTTP GET on "cover/" will include the two new triples:

```
<http://localhost:8080/fcrepo/rest/objects/raven/pages/cover/> pcdm:hasFile <http://localhost:8080/fcrepo/rest
/objects/raven/pages/cover/files/cover.jpg>
<http://localhost:8080/fcrepo/rest/objects/raven/pages/cover/> pcdm:hasFile <http://localhost:8080/fcrepo/rest
/objects/raven/pages/cover/files/cover.tif>
```

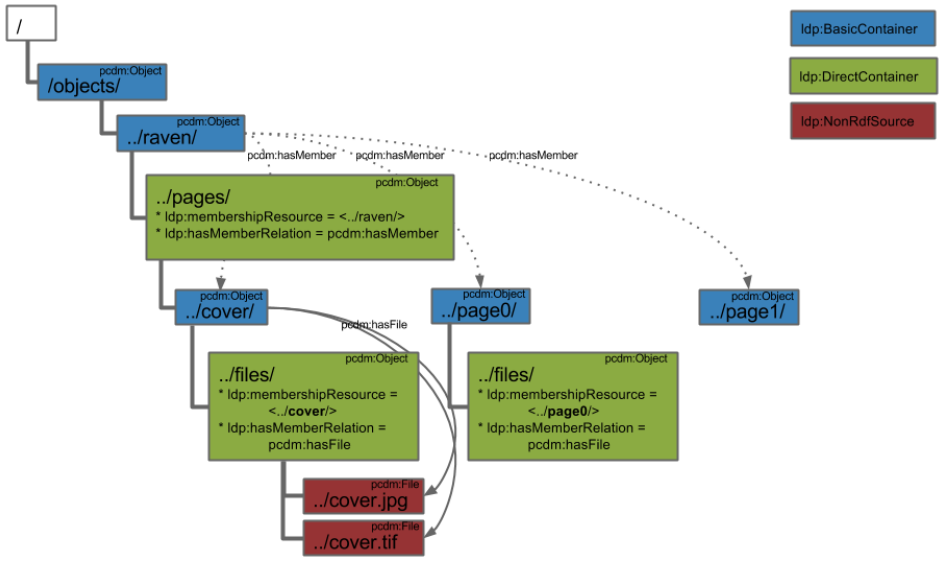
Once again,

- the subject of the triple comes from the "ldp:membershipResource" defined on "files/"
- the predicate of the triple comes from the "ldp:hasMemberRelation" defined on "files/", and
- the object of the triple is the new resource ("cover.jpg" or "cover.tif") that was added to the ldp:DirectContainer ("files/")

Page0 - Create DirectContainer

Page0 - Create DirectContainer

Here we repeat the exact steps as for the "cover/" above, but for "page0/".




```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @ldp-page0-direct.ttl localhost:8080/fcrepo/rest
/objects/raven/pages/page0/files/
```

Where "ldp-page0-direct.ttl" follows:

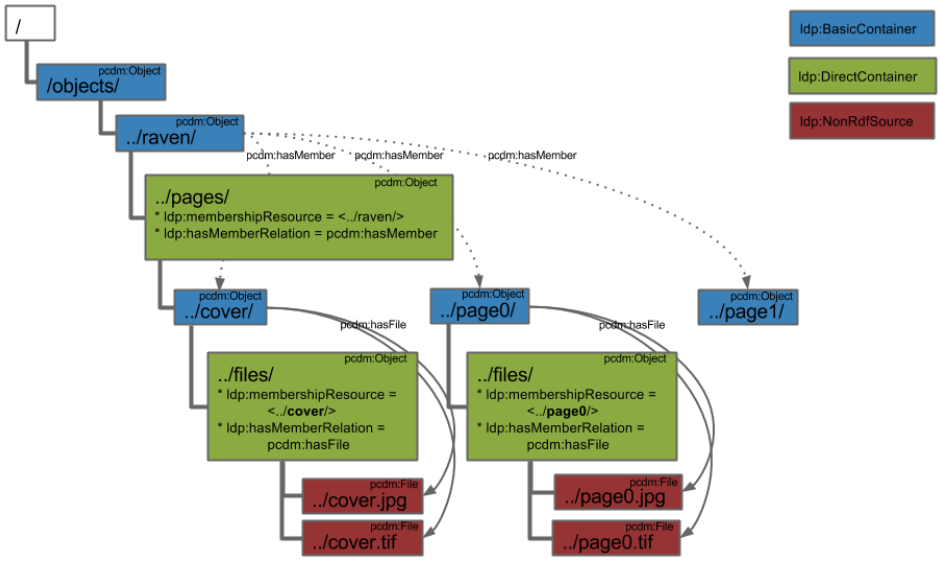
ldp-page0-direct.ttl

```
@prefix ldp: <http://www.w3.org/ns/ldp#>
@prefix pcdm: <http://pcdm.org/models#>

<> a ldp:DirectContainer, pcdm:Object ;
  ldp:membershipResource </fcrepo/rest/objects/raven/pages/page0/> ;
  ldp:hasMemberRelation pcdm:hasFile .
```

Page0 - Create Files

Page0 - Create Files

Here we add the attached page0 files (page0.jpg and page0.tif) to the newly created DirectContainer.




```
curl -i -XPUT -H"Content-Type: image/jpeg" --data-binary @page0.jpg localhost:8080/fcrepo/rest/objects/raven
/pages/page0/files/page0.jpg
curl -i -XPUT -H"Content-Type: image/tiff" --data-binary @page0.tif localhost:8080/fcrepo/rest/objects/raven
/pages/page0/files/page0.tif
```

Followed by assigning the type of pcdm:File to the respective RDF Sources found in the "Link; rel=describedby" header of each of the ldp:NonRdfSources.

```
curl -i -XPATCH -H"Content-Type: application/sparql-update" --data-binary @pcdm-file.ru localhost:8080/fcrepo
/rest/objects/raven/pages/page0/files/page0.jpg/fcr:metadata
curl -i -XPATCH -H"Content-Type: application/sparql-update" --data-binary @pcdm-file.ru localhost:8080/fcrepo
/rest/objects/raven/pages/page0/files/page0.tif/fcr:metadata
```

Where "pcdm-file.ru" follows:

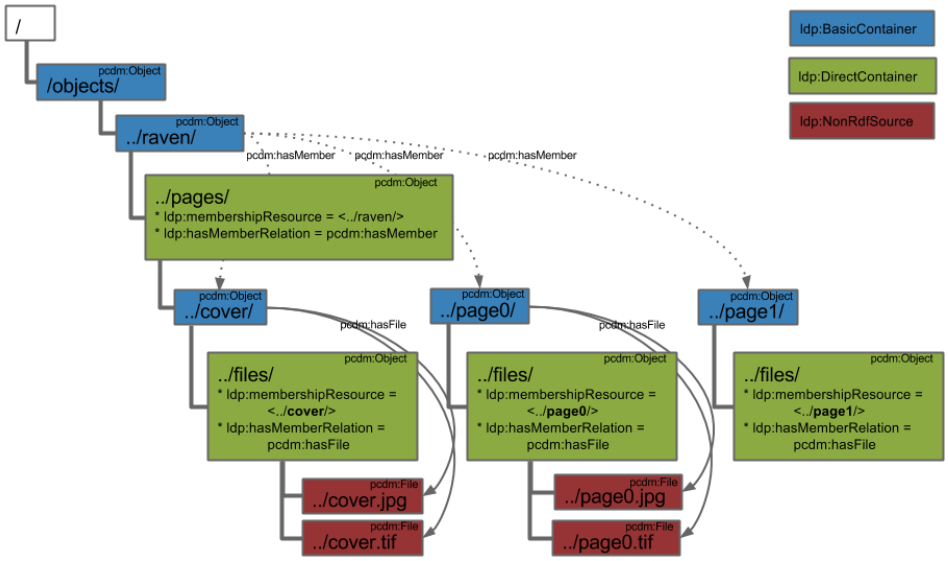
pcdm-file.ru

```
PREFIX pcdm: <http://pcdm.org/models#>
INSERT {
  <> a pcdm:File
} WHERE {
}
```

Page1 - Create DirectContainer

Page1 - Create DirectContainer

Here we repeat the exact steps as for the "page0/" above, but for "page1/".




```
curl -i -XPUT -H"Content-Type: text/turtle" --data-binary @ldp-page1-direct.ttl localhost:8080/fcrepo/rest
/objects/raven/pages/page1/files/
```

Where "ldp-page1-direct.ttl" follows:

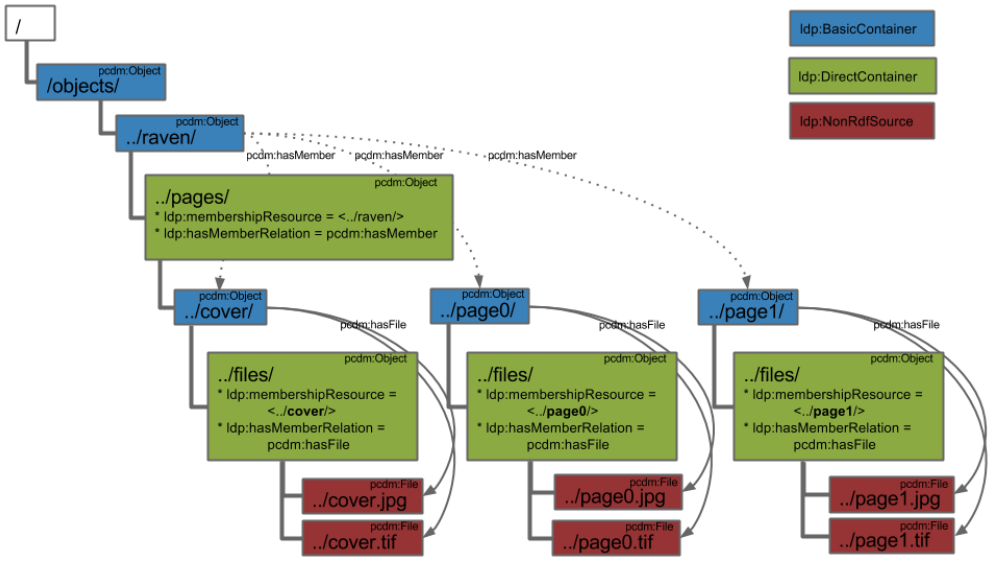
ldp-page1-direct.ttl

```
@prefix ldp: <http://www.w3.org/ns/ldp#>
@prefix pcdm: <http://pcdm.org/models#>

<> a ldp:DirectContainer, pcdm:Object ;
  ldp:membershipResource </fcrepo/rest/objects/raven/pages/page1/> ;
  ldp:hasMemberRelation pcdm:hasFile .
```

Page1 - Create Files

Page1 - Create Files

Finally, we add the attached page1 files (page1.jpg and page1.tif) to the newly created DirectContainer.




```
curl -i -XPUT -H"Content-Type: image/jpeg" --data-binary @page1.jpg localhost:8080/fcrepo/rest/objects/raven
/pages/page1/files/page1.jpg
curl -i -XPUT -H"Content-Type: image/tiff" --data-binary @page1.tif localhost:8080/fcrepo/rest/objects/raven
/pages/page1/files/page1.tif
```

Followed by assigning the type of pcdm:File to the respective RDF Sources found in the "Link; rel=describedby" header of each of the ldp:NonRdfSources.

```
curl -i -XPATCH -H"Content-Type: application/sparql-update" --data-binary @pcdm-file.ru localhost:8080/fcrepo
/rest/objects/raven/pages/page1/files/page1.jpg/fcr:metadata
curl -i -XPATCH -H"Content-Type: application/sparql-update" --data-binary @pcdm-file.ru localhost:8080/fcrepo
/rest/objects/raven/pages/page1/files/page1.tif/fcr:metadata
```

Where "pcdm-file.ru" follows:

pcdm-file.ru

```
PREFIX pcdm: <http://pcdm.org/models#>
INSERT {
  <> a pcdm:File
} WHERE {
}
```

Book - Conclusion

Using LDP in conjunction with PCDM terms, we have created a book, "raven/", with its constituent pages and their file representations.