LivCos

Cosmos Access

Table of Contents

1 Chapter

...

2 Cosmos URI

The data of the Cosmos can be addressed from extern or from within by a URL.

2.1 URL Decoding

To decode a URL into the cosmos a special XSLT is used.

? at least every project should be able to add and override individual decoding rules.

2.1.1 Object ID

The most direct way to address an object via it's ID. Such an URL does not contain a type extension.

/some.domain.net/parent/child/Object

This default access is in view scope and possible content gets transformed for the result.

2.1.2 Access Scope

The objects in the Cosmos can be access in different scopes. They define what part of the object's content will be provided as result.

Scope	Description	Branch
meta	The meta data for an object. Includes the ObjectNode itself, DataAccessNodes, child objects (also in the meta scope) and other meta information.	active
content	Includes only the content of the object, no meta data, no history. The root node of the content is the root node returned. An object containing only other objects (e.g. a folder) results in null (no content).	active
view	Returns the processed content of the object. The result of the first containing ViewNode forms the root node of the result. If the content is not a ViewNode this scope equals a simple content access.	active
history	The change history of all the entities, contained in the object. No meta data, no content. The most actual revision ID and date is also contained in the meta data.	active
acl	-- not yet implemented -- The Access Control List for the object.	active
cosmos	All the data of the object. Includes meta data, content, ACL and history of all the known branches.	all

Scope	Description	Branch
cache	All the data of the object, that is already in the memory cache. Nothing is read from the persistence layer. The object hierarchy may not be complete.	all
data	Combination of the "meta" and the "content" scope. All the data except the administrative parts like history or ACL. Allows access to the contents of an object tree (the content, all the contents of the child objects and so on).	active

Currently defined access scopes.

With certain type extensions the object can be accessed in specific scopes.

/some.domain.net/parent/child/Object.meta

/some.domain.net/parent/child/Object.cosmos

/some.domain.net/parent/child/Object.cache

/some.domain.net/parent/child/Object.content

/some.domain.net/parent/child/Object.view

2.1.3 Type Extension

With a type extension the object can be seen in a specific view.

/some.domain.net/parent/child/Object.pdf

2.1.4 Entities

...

/some.domain.net/parent/child/Object#entity_1

/some.domain.net/parent/child/Object$entity_1

2.1.5 Query

...

/some.domain.net/parent/child/Object?xpath=*[@name='foo']

/some.domain.net?q=hello+world

2.2 base-uri()

The base-uri() XPath function referees to the object URI of the document source. If the source actually is a view, the view's object URI forms this base. The object URI (cosmos:/domain/someObject) without the protocol equals the object ID (/domain/someObject).

-- object-view-view-base-uri() picture --

3 sys/resolver.xsl

The resolver system object tries to resolve URIs to flexibly access certain subtrees of the Cosmos.

3.1 Input

The input for the resolver transformer is the calling URI, broken into peaces for easier matching.

Eg. the URI /livcos.org/web/concept/LivingSoftware.html?param1=testVal#chapter_1 would produce:

```
<uri full="/livcos.org/web/concept/LivingSoftware.html" path="/livcos.org/web/concept" file="LivingSoftware.html"
      name="LivingSoftware" ext="html" relative="concept/LivingSoftware.html" relativeName="concept/LivingSoftware"
      fragment="chapter_1">
   <param key="param1" value="test value"/>
</uri>
```

for the /livcos.org/web/sys/resolver.xsl.

3.2 Output

The result of the resolve-transformation contains the exact instructions to access the cosmos.

An example could looks like this:

```
<access obj="/livcos.org/web/document" scope="view">
   <param key="ref" value="/livcos.org/LivCos/data/concept/LivingSoftware"/>
</access>
```

to access the /livcos.org/web/document object in the view scope and pass the 'ref' parameter with the data object ID.

<source ...>

4 Node ID

...

4.1 Entity ID

...

4.2 Content Node

...

4.3 Attribute Node

...

5 URI View Encoding

When accessing the Cosmos' content via a specific view, the URI must include the content and the view reference.

5.1 'ref' Parameter

The main URI addresses the view object and specifies the content reference as a parameter.

/livcos.org/web/doc/view.html?ref=/livcos.org/LivCos/data/Intro
/livcos.org/web/doc/view.html?ref=/livcos.org/LivCos/data/project/Cosmos/design/CosmosAccess

/livcos.org/page/doc/view.pdf?ref=/livcos.org/LivCos/data/Intro
/livcos.org/page/doc/view.pdf?ref=/livcos.org/LivCos/data/project/Cosmos/design/CosmosAccess

5.2 'view' Parameter

The main URI addresses the content object and specifies the view object as a parameter.

/livcos.org/LivCos/data/Intro.html?view=/livcos.org/web/doc
/livcos.org/LivCos/data/project/Cosmos/design/CosmosAccess.html?view=/livcos.org/web/doc

/livcos.org/LivCos/data/Intro.pdf?view=/livcos.org/page/doc
/livcos.org/LivCos/data/project/Cosmos/design/CosmosAccess.pdf?view=/livcos.org/page/doc

5.3 URI Split with resolver.xsl

The main URI will be split by the first resolver system object in the hierarchy.

5.3.1 View/Data.type

The resolver in the view object strips the URI and provides the reminding part as the content reference.

/livcos.org/web/doc/livcos.org/LivCos/data/Intro.html
/livcos.org/web/doc/livcos.org/LivCos/data/project/Cosmos/design/CosmosAccess.html

5.3.2 Path/View/Object.type

The resolver in the data object extracts and removes the view object from the URI and builds the according access URI.

/livcos.org/LivCos/data/livcos.org/web/doc/Intro.html
/livcos.org/LivCos/data/project/Cosmos/design/livcos.org/web/doc/CosmosAccess.html

Every data folder object would need to define a resolver for the extraction.

6 Default Domain

When an absolute URI does not address an available domain name, the default domain is taken as the context for that URI.

We rather have a more extensive default resolver (/localhost/system/resolver.xsl), handling the "unknown" URI calls.

7 Access to a Specific Revision

--content--

7.1 Revision ID, Timestamp or History Tag

A revision can be identified by it's ID (content hash), a timestamp (the revision at that time) or a history tag.

7.2 '_rev' Parameter

Eg. /livcos.org/data/MyObject?_rev=2010-01-02

7.3 URI Encoded

Eg. something like this: /livcos.org/data/MyObject_rev_2010-01-02

More like this: /livcos.org/data/MyObject$$2010-01-02

8 Language Code

An object with it's text content, translated in another language, is an object, separate from the original one. It should somehow remember a special relationship to all it's translations though.

Eg. /livcos.org/data/MyObject in the original language and /livcos.org/data/de/MyObject translates the content text into German.

8.1 View by Culture

Certain cultures might want to represent the content in a different style, depending on the language or region. => just another view on the same data.

Eg. /livcos.org/web/doc/arabic/livcos.org/data/MyDocument