load-objects

Command Reference

Table of contents

1. Description

Load object data from a file into the project model on the server.

The load-objects command provides the capability to read type instance data from a file and load it into the active project model running on the server.

Each type's module directory can contain an `objects` sub directory where files containing object data can be stored.

static: This command can be run outside of an object context.

2. Usage

```
ctl -m ProjectBuilder -c load-objects [-basedir <>] [-basetype
<>] [-delimiter <:>] [-filename <>] [-format <projectxml>]
[-type <all>]
```

2.1. Options

Option	Description	Type	Default
`basedir`	*dir containing modules*	string	`${entity.attribute.basedir}`
`basetype`	*specifies the basetype of object data* If type is not specified but filename is, type will default to base of the ${opts.filename} arugment minus the extension.	string	
`delimiter`	*field delimiter. Only relevant if format is tabular. Defaults to ":" (colon)*	string	`:`
`filename`	*filename containing object data* If filename is not specified but type is, filename will default to ${opts.type}.tab	string	

`format`	*data format. Either "tabular" or "projectxml".*	string	`projectxml`
`type`	*type name* If type not specified, the object data for all types in basedir will be loaded	string	`all`

3. Examples

The example below shows setting data being loaded on to the server.

The first example loads data defined as project XML format. Here a single file containing many object definitions of mixed type can be supplied to the load-objects command as follows:

```
ctl -p project -m ProjectBuilder -c load-objects -- -format projectxml
-filename PathToFile
```

Alternatively, the tabular format can be used:

```
ctl -p project -t ProjectBuilder -o object -c load-objects --\
          -type TypeX -format tabular -filename setting.tab -basetype
setting
```

Load-objects can default option values depending on other existing args. For example, if the file base is called *setting* the resource type will automatically be set to that. The reverse is also true. For example:

```
            ctl -p project -t ProjectBuilder -o object -c load-objects --\
               -type TypeX  -format tabular -filename setting.tab

            vs.

            ctl -p project -t ProjectBuilder -o object -c load-objects --\
               -type TypeX  -format tabular -basetype setting
```

If all object data files should be processed, one can leave out both options:

```
ctl -p project -t ProjectBuilder -o object -c load-objects -- -type TypeX
```