creating-cmip6-experiment-definitions-with-pyesdoc

September 10, 2015

0.1 Setup

```
In [ ]: # Import standard python modules
        import os
        import tempfile
        import uuid
```

```
In [ ]: # Import pyesdoc & cim
        import pyesdoc
        import pyesdoc.ontologies.cim as cim
```

```
In [ ]: # Set default document project code.
        _PROJECT = u'CMIP6-TEST'

        # Set default document source.
        _SOURCE = 'test-script'

        # Set default I/O directory.
        _IO_DIR = tempfile.mkdtemp()
```

0.2 Document Creation

```
In [ ]: # Create author.
        author = pyesdoc.create(cim.v2.Party, source=_SOURCE)
        author.email = "momipsl@ipsl.jussieu.fr"
        author.name = "Mark A. Greenslade"
```

```
In [ ]: # Create experimental requirement.
        tc = pyesdoc.create(cim.v2.TemporalConstraint, project=_PROJECT, author=author, source=_SOURCE)
        tc.canonical_name = "stc.027"
        tc.conformance_is_requested = False
        tc.description = "Begin in 1979 and run for 30 years"
        tc.duration = cim.v2.TimePeriod()
        tc.duration.length = 30
        tc.duration.units = "years"
        tc.duration.date_type = "unused"
        tc.name = "3.3.stc.1979_30yr"
```

```
In [ ]: # Create experiment.
        experiment = \
            pyesdoc.create(cim.v2.NumericalExperiment, project=_PROJECT, author=author, source=_SOURCE)
        experiment.canonical_name = "amip"
        experiment.description = "AMIP (1979 - at least 2008). Impose SSTs and sea ice from observations
        experiment.long_name = "AMIP"
        experiment.name = "amip"
        experiment.rationale = "AMIP. Baseline simulation for model evaluation."
        experiment.requirements.append(tc)
```

0.3 Document Validation

```
In [ ]: # Validate document & print errors
        errors = pyesdoc.validate(experiment)
        for err in errors:
            print(err)
```

0.4 Document Serialization

```
In [ ]: # Encode as JSON
        experiment_as_json = pyesdoc.encode(experiment, pyesdoc.ESDOC_ENCODING_JSON)
```

```
In [ ]: # Encode as XML
        experiment_as_xml = pyesdoc.encode(experiment, pyesdoc.ESDOC_ENCODING_XML)
```

```
In [ ]: # Encode as HTML
        experiment_as_html = pyesdoc.encode(experiment, pyesdoc.ESDOC_ENCODING_HTML)
```

```
In [ ]: # Decode from JSON
        experiment_from_json = pyesdoc.decode(experiment_as_json, pyesdoc.ESDOC_ENCODING_JSON)
```

```
In [ ]: # Decode from XML
        experiment_from_xml = pyesdoc.decode(experiment_as_xml, pyesdoc.ESDOC_ENCODING_XML)
```

0.5 Document I/O

```
In [ ]: # Set I/O directory.
        pyesdoc.set_option("output_dir", _IO_DIR)
```

```
In [ ]: # Write to file system.
        experiment_fpath_json = pyesdoc.write(experiment, pyesdoc.ESDOC_ENCODING_JSON)
        experiment_fpath_xml = pyesdoc.write(experiment, pyesdoc.ESDOC_ENCODING_XML)
```

```
In [ ]: # Read from file system.
        experiment_from_json_file = pyesdoc.read(experiment_fpath_json)
        experiment_from_xml_file = pyesdoc.read(experiment_fpath_xml)
```

0.6 Document Publishing

```
In [ ]: # Publish to ES-DOC web-service
        pyesdoc.publish(experiment)
```

```
In [ ]: # Retrieve from ES-DOC web-service
        # ... retrieve latest version
        experiment = pyesdoc.retrieve(experiment.meta.id)
        # ... retrieve specific version
        experiment = pyesdoc.retrieve(experiment.meta.id, experiment.meta.version)
```

```
In [ ]: # Republish to ES-DOC web-service
        # ... update document
        experiment.name += "-UPDATED"
        # ... cache current document version
        experiment_version = experiment.meta.version
```

```
         # ... republish
         pyesdoc.publish(experiment)
         # ... assert document version was updated
         assert experiment_version + 1 == experiment.meta.version
```

```
In [ ]: # Unpublish.
         pyesdoc.unpublish(experiment.meta.id)

         # Assert document can no longer be retrieved from web-service.
         assert pyesdoc.retrieve(experiment.meta.id) is None
```