
CoVE Documentation

Release beta

Open Data Services Co-operative Limited

Mar 26, 2018

Contents

1	Deployment Notes	3
1.1	How to create a deployment pull request	3
2	How Cove deals with errors	5
3	Using cove as a library (cove.lib)	7
3.1	common	7
3.2	converters	8
3.3	exceptions	8
3.4	ocds	8
3.5	threesixtygiving	8
3.6	tools	8
4	Indices and tables	9
	Python Module Index	11

Contents:

Deployment Notes

General Django deployment considerations apply to deploying Cove. We deploy using Apache and uwsgi using this [Salt State file](#).

1.1 How to create a deployment pull request

Post a pull request with a title following the appropriate template.:

For the monthly rollout of new features:

```
End of {{Month}} {{Year}} live deployment
```

For bug fixes: Post {{Month}} {{Year}} bug fixes ({{Num}}) - {{optionally, brief description of changes}} - live deployment

In both cases, add a description following this template:

```
URL for testing: http://release-{{YYYYMM}}.dev.cove.opendataservices.coop/

Planned deployment date:

#### Summary of changes for this deployment

#### Tasks in deploy process

Before merge:
- [ ] Re-run translations if any text has changed
- [ ] Create a new branch `release-{{YYYYMM}}` if it doesn't exist.
- [ ] Deploy to a subdomain on `cove-dev` for OCDS http://release-{{YYYYMM}}.dev.cove.
    ↳ opendataservices.coop/
- [ ] Check that the correct commit has been deployed using the link in the footer_
    ↳ http://release-{{YYYYMM}}.dev.cove.opendataservices.coop/
- [ ] Run `BROWSER=PhantomJS CUSTOM_SERVER_URL=http://release-{{YYYYMM}}.dev.cove.
    ↳ opendataservices.coop/ DJANGO_SETTINGS_MODULE=cove_ocds.settings py.test cove_ocds/
    ↳ tests_functional.py` - redo this for each redeploy to the subomdain
```

```
- [ ] Deploy to a subdomain on the 360 dev server http://release-{{YYYYMM}}.cove-360-  
↳dev.default.threesixtygiving.uk0.bigv.io/  
- [ ] Check that the correct commit has been deployed using the link in the footer_  
↳http://release-{{YYYYMM}}.cove-360-dev.default.threesixtygiving.uk0.bigv.io/  
- [ ] Run `BROWSER=PhantomJS CUSTOM_SERVER_URL=http://release-{{YYYYMM}}.cove-360-dev.  
↳default.threesixtygiving.uk0.bigv.io/ DJANGO_SETTINGS_MODULE=cove_360.settings py.  
↳test cove_360/tests_functional.py` - redo this for each redeploy to the subomdain
```

Steps above need redoing for additional commits.

After merge:

```
- [ ] Run salt highstate on `cove-360-live`  
- [ ] Check that the correct commit has been deployed using the link in the footer_  
↳http://dataquality.threesixtygiving.org/  
- [ ] Run `BROWSER=PhantomJS CUSTOM_SERVER_URL=https://dataquality.threesixtygiving.  
↳org/ DJANGO_SETTINGS_MODULE=cove_360.settings py.test cove_360/tests_functional.py`_  
↳on a local copy of the updated live branch  
- [ ] Run salt highstate on `cove-live-ocds`  
- [ ] Check that the correct commit has been deployed using the link in the footer_  
↳http://standard.open-contracting.org/validator/  
- [ ] Run `BROWSER=PhantomJS CUSTOM_SERVER_URL=http://dev.cove.opendataservices.coop/_  
↳DJANGO_SETTINGS_MODULE=cove_ocds.settings py.test cove_ocds/tests_functional.py` on_  
↳a local copy of the updated live branch  
- [ ] Check that changes on live are merged back into master too
```

Where { {{YYYYMM}} } should be replace with the actual year and month numbers - e.g. 201602

Add any extra tasks as appropriate. If they should be recurring update this template.

How Cove deals with errors

Errors in Cove can be broken down into 2 categories:

- Deliberately caught errors - to some extent we're expecting something to go wrong, probably due to people's data
- Uncaught 500 errors - something unexpected breaks

Breaking these down further:

- Deliberately caught errors
 - Custom message - something's wrong with the data, and we know what, so are able to display some custom help text. Since we know this is a data problem, it doesn't get logged in Sentry.
 - Generic message - something went wrong that we think is very likely to be a data issue (e.g. conversion failed), but we don't have a custom error message for it. We show the user the caught error, but also log it to Sentry.
- Uncaught 500 errors
 - Themed 500 error page - an otherwise uncaught exception, but we successfully rendered the friendly, well themed 500 page. These are always reported to Sentry.
 - Unthemed 500 error page - something went very wrong and we couldn't even display the nice 500 error page. The error should be reported to Sentry, but that may be broken too! In general these are serious bugs, and should be reported.

In an ideal world we want to eliminate all error messages except for the custom ones (ie. the only errors are data errors, and we can tell the users how to fix them).

Using cove as a library (cove.lib)

Currently the main user of `cove.lib` is `cove.views` which is responsible for rendering the Cove web application.

3.1 common

```
class cove.lib.common.CustomJsonrefLoader (**kwargs)
```

```
    get_remote_json (uri, **kwargs)
```

```
class cove.lib.common.CustomRefResolver (*args, **kw)
```

```
    resolve_remote (uri)
```

```
class cove.lib.common.SchemaJsonMixin
```

```
    deref_schema (schema_str)
```

```
    get_release_pkg_schema_fields ()
```

```
    get_release_pkg_schema_obj (deref=False)
```

```
    get_release_schema_obj (deref=False)
```

```
    release_pkg_schema_str
```

```
    release_schema_str
```

```
cove.lib.common.add_is_codelist (obj)
```

This is needed so that we can detect enums that are arrays as the `jsonschema` library does not give you any parent information and the `codelist` property is on the parent in this case. Only applies to `release.tag` in core schema at the moment.

```
cove.lib.common.fields_present_generator (json_data, prefix="")
```

```
cove.lib.common.get_additional_codelist_values (schema_obj, codelist_url, json_data)
```

```
cove.lib.common.get_counts_additional_fields(json_data, schema_obj, schema_name,
                                             context, fields_regex=False)
cove.lib.common.get_fields_present(*args, **kwargs)
cove.lib.common.get_json_data_deprecated_fields(json_data, schema_obj)
cove.lib.common.get_schema_validation_errors(json_data, schema_obj, schema_name,
                                             cell_src_map, heading_src_map, extra_checkers=None)
cove.lib.common.get_spreadsheet_meta_data(request, data_object, schema, file_type='xlsx',
                                           name='Meta')
cove.lib.common.required_draft4(validator, required, instance, schema)
cove.lib.common.schema_dict_fields_generator(schema_dict)
cove.lib.common.unique_ids(validator, ui, instance, schema)
```

3.2 converters

3.3 exceptions

exception `cove.lib.exceptions.CoveInputDataError` (*context=None*)

An error that we think is due to the data input by the user, rather than a bug in the application.

exception `cove.lib.exceptions.UnrecognisedFileType` (*context=None*)

`context = {'link': 'index', 'sub_title': <django.utils.functional.lazy.<locals>.__proxy__ object>, 'msg': <django.utils.fun`

`cove.lib.exceptions.cove_spreadsheet_conversion_error` (*func*)

`cove.lib.exceptions.cove_web_input_error` (*func*)

3.4 ocds

3.5 threesixtygiving

3.6 tools

`cove.lib.tools.datetime_or_date` (*instance*)

`cove.lib.tools.get_file_type` (*file*)

`cove.lib.tools.get_no_exception` (*item, key, fallback*)

`cove.lib.tools.ignore_errors` (*f*)

`cove.lib.tools.to_list` (*item*)

`cove.lib.tools.update_docs` (*document_parent, counter*)

CHAPTER 4

Indices and tables

- `genindex`
- `modindex`
- `search`

C

`cove.lib.common`, [7](#)
`cove.lib.exceptions`, [8](#)
`cove.lib.tools`, [8](#)

A

`add_is_codelist()` (in module `cove.lib.common`), 7

C

`context` (`cove.lib.exceptions.UnrecognisedFileType` attribute), 8

`cove.lib.common` (module), 7

`cove.lib.exceptions` (module), 8

`cove.lib.tools` (module), 8

`cove_spreadsheet_conversion_error()` (in module `cove.lib.exceptions`), 8

`cove_web_input_error()` (in module `cove.lib.exceptions`), 8

`CoveInputDataError`, 8

`CustomJsonrefLoader` (class in `cove.lib.common`), 7

`CustomRefResolver` (class in `cove.lib.common`), 7

D

`datetime_or_date()` (in module `cove.lib.tools`), 8

`deref_schema()` (`cove.lib.common.SchemaJsonMixin` method), 7

F

`fields_present_generator()` (in module `cove.lib.common`), 7

G

`get_additional_codelist_values()` (in module `cove.lib.common`), 7

`get_counts_additional_fields()` (in module `cove.lib.common`), 7

`get_fields_present()` (in module `cove.lib.common`), 8

`get_file_type()` (in module `cove.lib.tools`), 8

`get_json_data_deprecated_fields()` (in module `cove.lib.common`), 8

`get_no_exception()` (in module `cove.lib.tools`), 8

`get_release_pkg_schema_fields()` (`cove.lib.common.SchemaJsonMixin` method), 7

`get_release_pkg_schema_obj()`

(`cove.lib.common.SchemaJsonMixin` method), 7

`get_release_schema_obj()`

(`cove.lib.common.SchemaJsonMixin` method), 7

`get_remote_json()` (`cove.lib.common.CustomJsonrefLoader` method), 7

`get_schema_validation_errors()` (in module `cove.lib.common`), 8

`get_spreadsheet_meta_data()` (in module `cove.lib.common`), 8

I

`ignore_errors()` (in module `cove.lib.tools`), 8

R

`release_pkg_schema_str` (`cove.lib.common.SchemaJsonMixin` attribute), 7

`release_schema_str` (`cove.lib.common.SchemaJsonMixin` attribute), 7

`required_draft4()` (in module `cove.lib.common`), 8

`resolve_remote()` (`cove.lib.common.CustomRefResolver` method), 7

S

`schema_dict_fields_generator()` (in module `cove.lib.common`), 8

`SchemaJsonMixin` (class in `cove.lib.common`), 7

T

`to_list()` (in module `cove.lib.tools`), 8

U

`unique_ids()` (in module `cove.lib.common`), 8

`UnrecognisedFileType`, 8

`update_docs()` (in module `cove.lib.tools`), 8