Superdesk Documentation

Release 1.0b1

Sourcefabric

May 17, 2023

Contents

This documentation is technical, by developers for developers. It focuses on Superdesk backend (server part), without much info about Superdesk client.

Contents:

CHAPTER 1

Contributing

1.1 Documentation

Superdesk documentation is written using rst format and generated via Sphinx. It's organized by topic, using autodoc as much as possible to include docstrings from python code.

When working on docs, you can use live preview. In docs folder with virualenv enabled run:

```
$ make livehtml
```

This will build docs and run a server on `localhost:8000`. It will refresh as you modify documentation, but not when you modify python docstrings, in order to see some changes done there you still have to make some changes in rst files.

1.1.1 Updating docs

Documentation should be added/updated together with code changes in a single PR to superdesk-planning repository. There can be also PRs with only documentation.

1.1.2 New topic/module

To add a new topic or module docs, you create a file eg. `foo.rst` in `docs` folder and then you have to add it to `index.rst` toctree. This will make it appear in table of contents in both sidebar and on homepage.

1.1.3 Docs conventions

Again - we should use autodocs as much as possible so that documentation is close to code and thus should get updated with it. Thus to document a class or function, use autoclass and autofunction:

```
.. autoclass:: apps.publish.content.common.BasePublishService
    :members:

.. autofunction:: superdesk.publish.transmit
```

If you want to document multiple classes/functions from same module, you should use automodule or module first:

```
.. module:: superdesk.io.feed_parsers

.. autoclass:: ANPAFeedParser
.. autoclass:: IPTC7901FeedParser
```

If you want to document whole module with all its members, you can just use automodule:

```
.. automodule:: superdesk.io.feed_parsers
    :members:
```

This will document all public members from the module which have a docstring.

CHAPTER 2

Schema

Superdesk Planning internally uses a JSON schema that is based on the NewsML-G" / EventsML-G2 spec, so on ingest everything is converted to this schema, and on publishing it's converted to different formats.

2.1 Events Schema

Superdesk uses internally item schema that is an extension of ninjs, so on ingest everything is converted to this schema, and on publishing it's converted to different formats.

See IPTC-G2-Implementation_Guide (version 2.21) Section 15.4 for further information about the Events Schema.

2.1.1 Identifiers

`guid` *string*

Globally unique id. Using external id for ingested content.

`unique_id` *integer*

Internally unique id.

`unique_name` *string*

Internally unique name. By default same as `unique_id`.

`version` *integer*

Set by client - used to create items with version `0` which are used as drafts.

`ingest_id` *string*

Ingest item id from which item was fetched.

`recurrence_id` *string*

Internal id for recurrence events. All recurrence events for a particular recurring event will share this id.

2.1.2 Audit Information

`original_creator` *id*

User who created/fetched item.

`version_creator` *id*

User who created current version.

`firstcreated` *datetime*

When the item was created.

`versioncreated` *datetime*

When current version was created.

2.1.3 Ingest Details

`ingest_provider` *id*

Ingest provider id.

`source` *string*

Ingest provider source value. Using `DEFAULT_SOURCE_VALUE_FOR_MANUAL_ARTICLES` config for items created locally.

`original_source` *string*

Source value from ingested item.

`ingest_provider_sequence` *integer*

Counter for ingest items.

2.1.4 Event Details

`name` *string*

Name or title of the event.

`description_short` *string*

Text for short description.

`description_long` *string*

Text for long description.

`anpa_category` *list*

Optional, repeatable. The category classification(s) of the event.

`files` *list*

Optional, repeatable. Files attached to the event.

`relationships` *dict*

Details for relationships dict:

```
'relationships': {
    'broader': 'string',
    'narrower': 'string',
    'related': 'string',
}
```

`links` *list*

Optional, repeatable. Links attached to the event.

`dates` *dict*

Details of dates dict:

```
'date': {
    'start': 'dateimte',
    'end': 'dateimte',
    'duration': 'string',
    'confirmation': 'string',
    'recurring_date': [ 'datetime' ],
    'recurring_rule': {
        'frequency': 'string',
        'interval': 'string',
        'until': 'string',
        'count': 'string',
        'bymonth': 'string',
        'byday': 'string',
        'byhour': 'string',
        'byminute': 'string'
    },
    'occur_status': 'dict',
    'ex_date': [ 'datetime' ],
    'ex_rule': {
        'frequency': 'string',
        'interval': 'string',
        'until': 'string',
        'count': 'string',
        'bymonth': 'string',
        'byday': 'string',
        'byhour': 'string',
        'byminute': 'string'
    }
}
```

`occur_status` *dict*

Optional, non-repeatable property to indicate the provider's confidence that the event will occur.

`news_coverage_status` *dict*

Optional, non-repeatable element to indicate the status of planned news coverage of the event by the provider, using a QCODE and optional Name.

`registration` *string*

Optional, repeatable indicator of any registration details required for the event.

`access_status` *list*

Optional, repeatable property indicating the accessibility, the ease (or otherwise) of gaining physical access to the event, for example, whether easy, restricted, difficult.

`subject` *list*

Optional, repeatable. The subject classification(s) of the event.

`location` *list*

Repeatable property indicating the location of the event with an optionali Name.

`participant` *list*

Optional, repeatable, The people and/or organisations taking part in the event.

`participant_requirement` *list*

Optional, repeatable element for expressing any required conditions for participation in, or attendance at, the event, expressed by a URI or QCode.

`organiser` *list*

Optional, repeatable. Describes the organiser of the event.

`event_contact_info` *list*

Indicates how to get in contact with the event. This may be a web site, or a temporary office established for the event, not necessarily the organiser or any participant.

`language` *string*

Optional, describes the language(s) associated with the event.

2.2 Planning Schema

Superdesk uses internally item schema that is an extension of ninjs, so on ingest everything is converted to this schema, and on publishing it's converted to different formats.

See IPTC-G2-Implementation_Guide (version 2.21) Section 16 for further information about the Planning Schema.

This collection is storage for individual planning items as well as agendas. The *planning_type* field is used to determine the type.

2.2.1 Identifiers

`guid` *string*

Globally unique id. Using external id for ingested content.

`unique_id` *integer*

Internally unique id.

`unique_name` *string*

Internally unique name. By default same as `unique_id`.

`version` *integer*

Set by client - used to create items with version `0` which are used as drafts.

`ingest_id` *string*

Ingest item id from which item was fetched.

2.2.2 Audit Information

`original_creator` *id*

User who created/fetched item.

`version_creator` *id*

User who created current version.

`firstcreated` *datetime*

When the item was created.

`versioncreated` *datetime*

When current version was created.

Ingest Details

`ingest_provider` *id*

Ingest provider id.

`source` *string*

Ingest provider source value. Using `DEFAULT_SOURCE_VALUE_FOR_MANUAL_ARTICLES` config for items created locally.

`original_source` *string*

Source value from ingested item.

`ingest_provider_sequence` *integer*

Counter for ingest items.

2.2.3 Agenda Item Details

`planning_type` *string*

Text description of the type of planning. Can be null (event), or 'agenda'.

`name` *string*

Name for the agenda.

`planning_items` *list*

List of child planning ids.

2.2.4 Event Item

`event_item` *string*

Internal id of the associated event.

2.2.5 Planning Item Metadata

`ìtem_class`

News codes for the items associated with the planning.

`ednote` *string*

Editorial comment.

`description_text` *string*

Text description of the item. Used for media types.

`anpa_category` *list*

Values from category cv.

`subject` *list*

Values from IPTC subjectcodes plus from custom cvs.

`genre` *list*

Values from genre cv.

`company_codes` *list*

Values from company codes cv.

2.2.6 Content Metadata

`language` *string*

Item language code.

`abstract` *string*

Perex or lead.

`headline` *string*

Item headline.

`slugline` *string*

Item slugline.

`keywords` *list*

List of keywords.

`word_count` *integer*

Word count in `body_html` field.

`priority` *integer*

Item priority.

`urgency` *integer*

Item urgency.

`profile` *string*

Content profile id.

2.3 Locations Schema

Superdesk uses internally item schema that is an extension of ninjs, so on ingest everything is converted to this schema, and on publishing it's converted to different formats.

See IPTC-G2-Implementation_Guide (version 2.21) Section 12.6.3 for further information about the Locations Schema.

2.3.1 Identifiers

`guid` *string*

Globally unique id. Using external id for ingested content.

`unique_id` *integer*

Internally unique id.

`unique_name` *string*

Internally unique name. By default same as `unique_id`.

`version` *integer*

Set by client - used to create items with version `0` which are used as drafts.

`ingest_id` *string*

Ingest item id from which item was fetched.

2.3.2 Audit Information

`original_creator` *id*

User who created/fetched item.

`version_creator` *id*

User who created current version.

`firstcreated` *datetime*

When the item was created.

`versioncreated` *datetime*

When current version was created.

2.3.3 Ingest Details

`ingest_provider` *id*

Ingest provider id.

`source` *string*

Ingest provider source value. Using `DEFAULT_SOURCE_VALUE_FOR_MANUAL_ARTICLES` config for items created locally.

`original_source` *string*

Source value from ingested item.

`ingest_provider_sequence` *integer*

Counter for ingest items.

2.3.4 Location Details

`name` *string*

Plain text name for the location.

`position` *dict*

Details for position dict:

```
'position': {
    'latitude': 'float',
    'longitude': 'float',
    'altitude': 'integer',
    'gps_datum': 'string'
}
```

`address` *dict*

Details of address dict:

```
'address': {
    'line': [ 'string' ],
    'locality': 'string',
    'area': 'string',
    'country': 'string',
    'postal_code': 'string',
    'external': 'dict'
}
```

`access` *list*

Optional, repeatable element to indicate Methods of accessing the POI, including directions.

`details` *list*

Optional, repeatable indicatoled information about the location.

`created` *dateimte*

Optional, the date (and optionally a time) on which the physical location was created (not the location item).

`ceased_to_exist` *datetime*

Optional, the date (and optionally a time) on which the physical location ceased to exist.

`open_hours` *string*

Optional, the operational hours of the location.

`capacity` *string*

Optional, location capacity.

`contact_info` *list*

Optional, repeatable. Indicates how to get in contact with the location. This may be a web site, email, phone or any other human readable contact information.

2.4 Coverage Schema

Superdesk uses internally item schema that is an extension of ninjs, so on ingest everything is converted to this schema, and on publishing it's converted to different formats.

See IPTC-G2-Implementation_Guide (version 2.21) Section 16.4 for further information about the Coverage Schema.

2.4.1 Identifiers

`guid` *string*

Globally unique id. Using external id for ingested content.

`unique_id` *integer*

Internally unique id.

`unique_name` *string*

Internally unique name. By default same as `unique_id`.

`version` *integer*

Set by client - used to create items with version `0` which are used as drafts.

`ingest_id` *string*

Ingest item id from which item was fetched.

2.4.2 Audit Information

`original_creator` *id*

User who created/fetched item.

`version_creator` *id*

User who created current version.

`firstcreated` *datetime*

When the item was created.

`versioncreated` *datetime*

When current version was created.

2.4.3 Ingest Details

`ingest_provider` *id*

Ingest provider id.

`source` *string*

Ingest provider source value. Using `DEFAULT_SOURCE_VALUE_FOR_MANUAL_ARTICLES` config for items created locally.

`original_source` *string*

Source value from ingested item.

`ingest_provider_sequence` *integer*

Counter for ingest items.

2.4.4 Planning Item

`planning_item`

Internal id of the associated planning item.

2.4.5 Planning Metadata Hints

`planning.ednote` *string*

Editorial comment.

`planning.g2_content_type`

Optional, non-repeatable element to indicate the MIME type of the intended coverage.

`planning.item_class` *string*

Optional, non-repeatable element indicates the type of content to be delivered, using the IPTC News Item Nature NewsCodes.

`planning.item_count` *string*

The number of items to be delivered, expressed as a range (ex: 1-5)

`planning.scheduled` *dateimte*

Optional, non-repeatable. Indicates the scheduled time of delivery, and may be truncated if the precise date and time is not known.

`planning.service` *list*

Optional, repeatable. The editorial service to which the content has been assigned by the provider and on which the receiver should expect to receive the planned content.

`planning.assigned_to` *string*

Optional, non-repeatable element that holds the details of a person or organisation who has been assigned to create the announced content.

`planning.news_content_characteristics` *list*

Optional, repeatable,enables providers to express physical properties of the planned item using attributes from the News Content Characteristics group.

`planning.planning_ext_property` *list*

For example, the planned item has a proprietary content rating. The rating is expressed using a QCode indicating the nature of the proprietary property, an optional name, and a value.

`planning.by` *list*

Optional, repeatable. Natural language author/creator information.

`planning.credit_line` *list*

Optional, repeatable. A freeform expression of the credit(s) for the content.

`planning.dateline` *list*

Optional, repeatable. Natural language information traditionally placed at the start of a text by some news agencies, indicating the place and time that the content was created.

`planning.description` *list*

Text description of the item. Used for media types.

`planning.genre` *list*

Values from genre cv.

`planning.headline` *string*

Item headline.

`planning.keywords` *list*

List of keywords.

`planning.language` *string*

Item language code.

`planning.slugline` *string*

Item slugline.

`planning.subject` *list*

Values from IPTC subjectcodes plus from custom cvs.

2.4.6 Delivery Metadata

`delivery` *dict*

Optional, repeatable, tells the receiver which parts of the planned coverage has been delivered:

```
'delivery': [
    {
        'rel': 'string',
        'href': 'string',
        'residref': 'string',
        'version': 'string',
        'content_type': 'string',
        'format': 'string',
        'size': 'string',
        'persistent_id_ref': 'string',
        'valid_from': 'datetime',
        'valid_to': 'datetime',
        'creator': 'string',
        'modified': 'datetime',
        'xml_lang': 'string',
        'dir': 'string',
        'rank': 'integer'
    }
]
```

CHAPTER 3

Source

The following documentation is generated from source code comment.

3.1 planning

3.1.1 planning package

Subpackages

planning.assignments package

planning.assignments.assignments module

planning.assignments.assignments_complete module

planning.assignments.assignments_content module

planning.assignments.assignments_history module

planning.assignments.assignments_link module

planning.assignments.assignments_link_tests module

planning.assignments.assignments_lock module

planning.assignments.assignments_revert module