
invenio-sip2 Documentation

Release 0.6.19

UCLouvain

Dec 19, 2022

Contents

1	Implemented SIP2 Features	3
1.1	User's Guide	3
1.2	API Reference	7
1.3	Additional Notes	22
	Python Module Index	29
	Index	31

Invenio module that add a SIP2 interface between a library's Automated Circulation System and library automation devices.

This project is in work in progress. Some features may not yet be implemented.

Further documentation is available on <https://invenio-sip2.readthedocs.io/>

Implemented SIP2 Features

- Login
- Selfcheck Status
- Request Resend
- Patron Status
- Patron Enable
- End Patron Session
- Patron Information
- Item Information
- Checkout
- Checkin
- Renew
- Fee Paid

1.1 User's Guide

This part of the documentation will show you how to get started in using Invenio-SIP2.

1.1.1 Invenio-SIP2 Installation

Invenio-SIP2 is on PyPI so all you need is:

```
$ pip install invenio-sip2
```

1.1.2 Configuration

Configuration variables for defining invenio-sip2.

<code>SIP2_DATASTORE_HANDLER</code>	datastore, default: <i>Sip2RedisDatastore</i>
<code>SIP2_DATASTORE_REDIS_PREFIX</code>	Prefix for redis keys, default <i>sip2</i>
<code>SIP2_DATASTORE_REDIS_URL</code>	Redis Datastore URL
<code>SIP2_MESSAGE_ACTIONS</code>	Dictionary of all selfcheck actions.
<code>SIP2_REMOTE_ACTION_HANDLERS</code>	Dictionary of remote action handlers. See example below.
<code>SIP2_MESSAGE_TYPES</code>	Define all message types conforming to SIP2 protocol.
<code>SIP2_FIXED_FIELD_DEFINITION</code>	All fixed field available.
<code>SIP2_VARIABLE_FIELD_DEFINITION</code>	All variable field available.

Datastore handlers

Use `SIP2_DATASTORE_HANDLER` to define your custom datastore.

Provided datastore by invenio-sip2 is:

```
class invenio_sip2.datastore:Sip2RedisDatastore
```

Remote action handlers

Handlers allow customizing endpoints for each selfcheck actions.

Each custom handler actions must be defined in the `SIP2_ACTIONS_HANDLERS` dictionary, where the keys are the application names and the values the configuration parameters for the application.

```
SIP2_REMOTE_ACTION_HANDLERS = dict(
    myapp=dict(
        # configuration values for myapp ...
    ),
)
```

The application name is used to start invenio-sip2 server and call customized handlers.

Configuration of a single remote application is a dictionary with the following keys:

- `login_handler` - Import path to login selfcheck callback handler.
- `logout_handler` - Import path to logout selfcheck callback handler.
- **`system_status_handler` - Import path to automated system status callback handler.**
- **`patron_handlers` - A dictionary of import path to patron callback handler.**
 - `validate_patron` - Import path to validate patron callback handler.
 - `authorize_patron` - Import path to authorize patron callback handler.
 - `enable_patron` - Import path to enable patron callback handler.
 - `patron_status` - Import path to patron status callback handler.
 - `account` - Import path to retrieve patron account callback handler.
- **`item_handlers` - A dictionary of import path to item callback handler.**
 - `item` - Import path to retrieve item callback handler.

- **circulation_handlers** - A dictionary of import path to circulation callback handler.
 - checkout - Import path to checkout item callback handler.
 - checkin - Import path to checkin item callback handler.
 - hold - Import path to hold item callback handler.
 - renew - Import path to renew item callback handler.
 - renew_all - Import path to renew_all items callback handler.
- fee_paid_handler - Import path to fee paid callback handler.“

```
SIP2_REMOTE_ACTION_HANDLERS = dict(
    app=dict(
        login_handler="...",
        logout_handler="...",
        system_status_handler="...",
        patron_handlers=dict(
            validate_patron="...",
            authorize_patron="...",
            enable_patron="...",
            patron_status="...",
            account="...",
        ),
        item_handlers=dict(
            item="..."
        ),
        circulation_handlers=dict(
            checkout="...",
            checkin="...",
            hold="...",
            renew="...",
        ),
        fee_paid_handler="...",
    )
)
```

`invenio_sip2.config.BABEL_DEFAULT_LANGUAGE = 'en'`
Default language

`invenio_sip2.config.SIP2_CIRCULATION_DATE_FORMAT = '%Y%m%d %H%M%S'`
SIP2 date format for circulation.

`invenio_sip2.config.SIP2_DATE_FORMAT = '%Y%m%d %H%M%S'`
SIP2 date format for transaction.

`invenio_sip2.config.SIP2_DEFAULT_LANGUAGE = 'en'`
Default SIP2 language

`invenio_sip2.config.SIP2_DEFAULT_SECURITY_MARKER = '00'`
SIP2 default security marker type.

`invenio_sip2.config.SIP2_ERROR_DETECTION = True`
Enable error detection on message.

`invenio_sip2.config.SIP2_LINE_TERMINATOR = '\r'`
Message line separator.

`invenio_sip2.config.SIP2_LOGGING_CONSOLE = True`
Enable logging to the console.

`invenio_sip2.config.SIP2_LOGGING_CONSOLE_LEVEL = 'INFO'`
Console logging level.

All requests and responses will be written to the console if the level is on info mode. Otherwise, they will not be logged.

```
invenio_sip2.config.SIP2_LOGGING_FS_BACKUPCOUNT = 5
    Number of rotated log files to keep.

invenio_sip2.config.SIP2_LOGGING_FS_LEVEL = 'INFO'
    Console logging level.

    Defaults to write all requests and responses.

invenio_sip2.config.SIP2_LOGGING_FS_LOGFILE = None
    Enable logging to the filesystem.

    A valid filesystem path is required to enable logging.

invenio_sip2.config.SIP2_LOGGING_FS_MAXBYTES = 104857600
    Maximum size of logging file. Default: 100MB.

invenio_sip2.config.SIP2_PERMISSIONS_FACTORY(action)
    Define factory permissions.

invenio_sip2.config.SIP2_PROTOCOL = '2.00'
    SIP2 protocol version.

invenio_sip2.config.SIP2_REMOTE_ACTION_HANDLERS = {}
    Configuration of remote handlers.

invenio_sip2.config.SIP2_RETRIES_ALLOWED = 10
    Number of retries allowed.

invenio_sip2.config.SIP2_SOCKET_BUFFER_SIZE = '1024'
    Socket buffer size.

invenio_sip2.config.SIP2_SUPPORT_CHECKIN = True
    Support check in items.

invenio_sip2.config.SIP2_SUPPORT_CHECKOUT = True
    Support check out items.

invenio_sip2.config.SIP2_SUPPORT_OFFLINE_STATUS = True
    Support off line operation.

invenio_sip2.config.SIP2_SUPPORT_ONLINE_STATUS = True
    Support online status send by automatic circulation system.

invenio_sip2.config.SIP2_SUPPORT_RENEWAL_POLICY = True
    Support patron renewal requests as a policy.

invenio_sip2.config.SIP2_SUPPORT_STATUS_UPDATE = True
    Support patron status updating by the selfcheck.

invenio_sip2.config.SIP2_TEXT_ENCODING = 'UTF-8'
    Message text charset encoding.

invenio_sip2.config.SIP2_TIMEOUT_PERIOD = 10
    Server timeout.
```

1.1.3 Usage

Start SIP2 server with CLI example:

```
$ invenio selfcheck start <server_name> -h 127.0.0.1 -p 3004 -r your-remote-app
```

1.1.4 Example application

First install Invenio-SIP2, setup the application and load fixture data by running:

```
$ pip install -e .[all]
$ cd examples
$ ./app-setup.sh
$ ./app-fixtures.sh
```

Next, start the development server:

```
$ export FLASK_APP=app.py FLASK_DEBUG=1
$ flask run
```

and open the example application in your browser:

```
$ open http://127.0.0.1:5000/
```

To reset the example application run:

```
$ ./app-teardown.sh
```

1.2 API Reference

If you are looking for information on a specific function, class or method, this part of the documentation is for you.

1.2.1 API Docs

Flask extension for Invenio-SIP2.

class `invenio_sip2.ext.InvenioSIP2` (*app=None*)

Invenio-SIP2 extension.

Extension initialization.

add_console_handler (*app*)

Add console handler to logger.

add_fs_handler (*app*)

Add file handler to logger.

classmethod `get_logging_formatter` ()

Return logging formatter.

init_app (*app*)

Flask application initialization.

init_config (*app*)

Initialize configuration.

is_error_detection_enabled

Check if error detection is enabled.

line_terminator

Line terminator used for message.

retries_allowed

Number of retries allowed by the automated circulation system.

sip2

Return the SIP2 action machine.

sip2_current_date

Get current date from system.

sip2_handlers

Return the SIP2 handler machine.

sip2_language

Get default language from system.

sip2_message_types

Message type configuration.

support_checkin

Support of checkin by the automated circulation system.

support_checkout

Support of checkout by the automated circulation system.

support_offline_status

Support of offline status by the automated circulation system.

support_online_status

Support of online status by the automated circulation system.

support_renewal_policy

Support of renewal policy by the automated circulation system.

support_status_update

Support of status update by the automated circulation system.

supported_messages (*remote_app*)

Supported messages by the automated circulation system.

supported_protocol

Supported protocol by the automated circulation system.

text_encoding

Message text charset encoding.

timeout_period

Timeout period allowed by the automated circulation system.

`invenio_sip2.ext.load_fixed_field(app)`

Load fixed field configuration.

`invenio_sip2.ext.load_variable_field(app)`

Load variable field configuration.

Actions

Invenio-SIP2 custom actions.

```
class invenio_sip2.actions.actions.AutomatedCirculationSystemStatus (command,  
re-  
sponse,  
message,  
**kwargs)
```

Action to get status from automated circulation system.

Init action object.

execute (*message, client*)

Execute automated circulation system status action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.BlockPatron (command,      response,      message,  
**kwargs)
```

Action to block patron.

Init action object.

execute (*message, client, **kwargs*)

Execute block patron action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.Checkin (command, response, message, **kwargs)
```

Action to checkin an item.

Init action object.

execute (*message, client, **kwargs*)

Execute checkin action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.Checkout (command, response, message, **kwargs)
```

Action to checkout an item.

Init action object.

execute (*message, client, **kwargs*)

Execute checkout action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.EndPatronSession(command, response, message,  
                                                    **kwargs)
```

Action to end patron session on automated circulation system.

Init action object.

```
execute (message, client)
```

Execute end patron session action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.FeePaid (command, response, message, **kwargs)
```

Action to paid fee.

Init action object.

```
execute (message, client, **kwargs)
```

Execute fee paid action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.Hold (command, response, message, **kwargs)
```

Action to hold an item.

Init action object.

```
execute (message, client, **kwargs)
```

Execute hold action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.ItemInformation (command, response, message,  
                                                    **kwargs)
```

Action to get item information from automated circulation system.

Init action object.

```
execute (message, client)
```

Execute item information action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.ItemStatusUpdate(command, response, message,  
                                                    **kwargs)
```

Action to update item status.

Init action object.

```
execute (message, client, **kwargs)  
    Execute item status action.
```

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.PatronEnable(command, response, message,  
                                                **kwargs)
```

Action to enable patron on automated circulation system.

Init action object.

```
execute (message, client)  
    Execute enable patron action.
```

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.PatronInformation(command, response, message,  
                                                    **kwargs)
```

Action to get patron information from automated circulation system.

Init action object.

```
execute (message, client)  
    Execute patron information action.
```

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

```
class invenio_sip2.actions.actions.PatronStatus(command, response, message,  
                                                **kwargs)
```

Action to get patron status from automated circulation system.

Init action object.

```
execute (message, client)  
    Execute patron status action.
```

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

class invenio_sip2.actions.actions.**Renew**(*command, response, message, **kwargs*)
 Action to renew an item.

Init action object.

execute(*message, client, **kwargs*)
 Execute renew action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

class invenio_sip2.actions.actions.**RenewAll**(*command, response, message, **kwargs*)
 Action to renew all items.

Init action object.

execute(*message, client, **kwargs*)
 Execute renew all action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the current action

class invenio_sip2.actions.actions.**RequestResend**(*command, message*)
 Action to resend last message.

Init action object.

execute(*message, client*)
 Execute resend action.

Parameters

- **message** – message receive from the client
- **client** – the client

Returns message class representing the response of the last action send tho the client

class invenio_sip2.actions.actions.**SelfCheckLogin**(*command, response, message, **kwargs*)

Action to selfcheck login.

Init action object.

execute(*message, **kwargs*)
 Execute login action.

Parameters **message** – message receive from the client

Returns message class representing the response of the current action

Records API

API for manipulating the client.

class invenio_sip2.records.record.**Client** (*data*, ***kwargs*)

class for selfcheck client.

Initialize instance with dictionary data.

Parameters **data** – Dict with record metadata.

clear_patron_session ()

Shortcut to library name.

get_current_patron_session ()

Shortcut to patron session.

get_key ()

Get generated key for Client object.

get_server ()

Get server object.

institution_id

Shortcut to institution id.

is_authenticated

Shortcut to check if the selfcheck client is authenticated.

last_request_message

Shortcut to user id.

last_response_message

Shortcut to user id.

last_sequence_number

Shortcut to user id.

library_language

Shortcut for library language.

library_name

Shortcut to library name.

remote_app

Shortcut for remote app.

server_id

Get server identifier.

terminal

Shortcut to terminal.

transaction_user_id

Shortcut to user id.

class invenio_sip2.records.record.**Server** (*data*, ***kwargs*)

class for SIP2 server.

Initialize instance with dictionary data.

Parameters **data** – Dict with record metadata.

clear_all_clients ()

Clear all clients.

classmethod **create** (*data*, *id_=None*, ***kwargs*)

Create record.

Parameters

- **data** – Dict with metadata.
- **id** – Specify a UUID to use for the new record.

delete()

Delete server and all attached clients.

down()

Set server status to *Down* and clear all clients data.

classmethod find_server(kwargs)**

Find server depending kwargs.

get_clients()

Return clients.

is_running

Check if server is running.

number_of_clients

Shortcut for number of clients.

up()

Set server status to *running* and clear all clients data.

class invenio_sip2.records.record.Sip2RecordMetadata(data, **kwargs)

Sip2RecordMetadata class.

Initialize instance with dictionary data.

Parameters data – Dict with record metadata.

classmethod count()

Return number of all records based on record type.

classmethod create(data, id_=None, **kwargs)

Create record.

Parameters

- **data** – Dict with metadata.
- **id** – Specify a UUID to use for the new record.

delete()

Delete record by uuid.

dumps(kwargs)**

Return pure Python dictionary with record metadata.

classmethod get_all_records()

Get all records.

get_key()

Get generated key for Sip2RecordMetadata object.

classmethod get_record_by_id(id_)

Get record by uuid.

id

Shortcut for id.

search(query='*', index_type=None, filter_query=None)

Search record by query.

update (*data*)

Update instance with dictionary data.

Parameters *data* – Dict with metadata.

Models

Models for Invenio-SIP2.

class invenio_sip2.models.PatronStatus

Class to define patron status.

Constructor.

add_patron_status_type (*patron_status_type*)

Add patron status.

Parameters *patron_status_type* – Enum of patron status type

Add *patron_status_type* indicates that the condition is true. raise exception if patron status type does not exist.

class invenio_sip2.models.PatronStatusTypes

Enum class to list all possible patron status types.

CARD_REPORTED_LOST = 'card_reported_lost'

CHARGE_PRIVILEGES_DENIED = 'charge_privileges_denied'

EXCESSIVE_OUTSTANDING_FEES = 'excessive_outstanding_fees'

EXCESSIVE_OUTSTANDING_FINES = 'excessive_outstanding_fines'

HOLD_PRIVILEGES_DENIED = 'hold_privileges_denied'

RECALL_OVERDUE = 'recall_overdue'

RECALL_PRIVILEGES_DENIED = 'recall_privileges_denied'

RENEWAL_PRIVILEGES_DENIED = 'renewal_privileges_denied'

TOO_MANY_CLAIMS_OF_ITEMS_RETURNED = 'too_many_claims_of_items_returned'

TOO_MANY_ITEMS_BILLED = 'too_many_items_billed'

TOO_MANY_ITEMS_CHARGED = 'too_many_items_charged'

TOO_MANY_ITEMS_LOST = 'too_many_items_lost'

TOO_MANY_ITEMS_OVERDUE = 'too_many_items_overdue'

TOO_MANY_RENEWALS = 'too_many_renewals'

class invenio_sip2.models.SelfcheckCheckin (*permanent_location*, *checkin=False*,
alert=False, *magnetic_media='unknown'*,
resensitize='unknown', ***kwargs*)

Class representing checkin handler response.

Constructor.

:param *permanent_location* - permanent_location of the item :param *checkin* - checkin operation is success or not :param *alert* - indicate if the selcheck will generate sound alert :param *magnetic_media* - indicate the presence of magnetic media :param *resensitize* - resensitize an item ? :param *kwargs* - optional fields

has_magnetic_media

Shortcut for desensitize.

is_success

Shortcut for checkin.

resensitize

Shortcut for resensitize.

sound_alert

Shortcut for alert.

```
class invenio_sip2.models.SelfcheckCheckout (title_id, checkout=False, renewal=False,  
                                              magnetic_media='unknown', desensi-  
                                              tize='unknown', **kwargs)
```

Class representing checkout handler response.

Constructor.

:param title_id - title_id (e.g. title, identifier, ...) :param checkout - checkout operation is success or not :param renewal - renewal operation is success or not :param magnetic_media - indicate the presence of magnetic media :param desensitize - desensitize an item ? :param kwargs - optional fields

desensitize

Shortcut for desensitize.

due_date

Shortcut for due date.

has_magnetic_media

Shortcut for desensitize.

is_renewal

Shortcut for renewal ok.

is_success

Shortcut for checkout ok.

```
class invenio_sip2.models.SelfcheckCirculationStatus
```

Class to handle all available circulation status of an item.

AVAILABLE = '03'

CHARGED = '04'

CHARGED_RECALL = '05'

CLAIMED_RETURNED = '11'

IN_PROCESS = '06'

IN_TRANSIT = '10'

LOST = '12'

MISSING = '13'

ON_ORDER = '02'

OTHER = '01'

RECALLED = '07'

WAITING_ON_HOLD_SHELF = '08'

WAITING_TO_RESHELF = '09'

```
class invenio_sip2.models.SelfcheckEnablePatron (patron_id, institution_id, patron_name="", patron_status=None, language='und', **kwargs)
```

Class representing patron information handler response.

Constructor.

:param *patron_id* - patron identifier (e.g. id, barcode, ...) :param *institution_id* - institution id (or code) of the patron :param *patron_name* - full name of the patron :param *patron_status* - status of the patron :param *language* - iso-639-2 language :param *kwargs* - optional fields

```
class invenio_sip2.models.SelfcheckFeePaid (accepted=False, **kwargs)
```

Class representing fee paid handler response.

Constructor.

Parameters

- **accepted** – payment operation is accepted or not should be set to True if the ACS accept the payment. should be set to 0 if the ACS did not accept the payment.
- **kwargs** – optional fields

is_accepted

Shortcut for *payment_accepted* operation.

```
class invenio_sip2.models.SelfcheckFeeType
```

Class to handle all available fee type.

ADMINISTRATIVE = '02'

COMPUTER_ACCESS_CHARGE = '08'

DAMAGE = '03'

HOLD_FEE = '09'

OTHER = '01'

OVERDUE = '04'

PROCESSING = '05'

RENTAL = '06'

REPLACEMENT = '07'

```
class invenio_sip2.models.SelfcheckHold (hold=False, available=False, **kwargs)
```

Class representing hold handler response.

Constructor.

:param *hold* - hold operation is success or not :param *available* - item available or not :param *kwargs* - optional fields

is_available

Shortcut for *available*.

is_success

Shortcut for *hold ok*.

```
class invenio_sip2.models.SelfcheckItemInformation (item_id, title_id=None, circulation_status=None, fee_type=None, security_marker=None, **kwargs)
```

Class representing item information handler response.

Constructor.

:param patron_id - patron identifier (e.g. id, barcode, ...) :param patron_name - full name of the patron :param institution_id - institution id (or code) of the patron :param language - iso-639-2 language :param kwargs - optional fields

class invenio_sip2.models.SelfcheckLanguage

Enum class to list all available language.

```
ARABIC = '016'
BELGIAN = '026'
CANADIAN_FRENCH = '011'
CHINESE = '019'
DANISH = '009'
DUTCH = '005'
ENGLISH = '001'
FINNISH = '007'
FRENCH = '002'
GERMAN = '003'
GREEK = '018'
HEBREW = '013'
ICELANDIC = '025'
ITALIAN = '004'
JAPANESE = '014'
KOREAN = '020'
MALAY = '023'
NORTH_AMERICAN_SPANISH = '021'
NORWEGIAN = '012'
POLISH = '017'
PORTUGUESE = '010'
RUSSIAN = '015'
SPANISH = '008'
SWEDISH = '006'
TAIWANESE = '027'
TAMIL = '022'
UNITED_KINGDOM = '024'
UNKNOWN = '000'
chi = '019'
dan = '009'
dut = '005'
```

```
eng = '001'
fin = '007'
fra = '002'
fre = '002'
ger = '003'
gre = '018'
heb = '013'
ice = '025'
isl = '025'
ita = '004'
jpn = '014'
kor = '020'
may = '023'
msa = '023'
nor = '012'
pol = '017'
por = '010'
rus = '015'
spa = '008'
swe = '006'
tam = '022'
und = '000'
zho = '019'
```

```
class invenio_sip2.models.SelfcheckMediaType
    Class to handle all available media type.
```

```
AUDIO = '004'
BOOK = '001'
BOOK_WHIT_AUDIO_TAPE = '010'
BOOK_WHIT_CD = '009'
BOOK_WHIT_DISKETTE = '008'
BOUND_JOURNAL = '003'
CD_OR_CDROM = '006'
DISKETTE = '007'
MAGAZINE = '002'
OTHER = '000'
VIDEO = '005'
```

```
class invenio_sip2.models.SelfcheckPatronInformation (patron_id,      institution_id,
                                                    patron_name="",      pa-
                                                    tron_status=None,      lan-
                                                    guage='und', **kwargs)
```

Class representing patron information handler response.

Constructor.

:param patron_id - patron identifier (e.g. id, barcode, ...) :param institution_id - institution id (or code) of the patron :param patron_name - full name of the patron :param patron_status - status of the patron :param language - iso-639-2 language :param kwargs - optional fields

charged_items_count

Shortcut for charged items count.

fine_items_count

Shortcut for fine items count.

hold_items_count

Shortcut for hold items count.

overdue_items_count

Shortcut for overdue items count.

patron_id

Shortcut for patron pid.

patron_name

Shortcut for patron pid.

recall_items_count

Shortcut for recall items count.

unavailable_items_count

Shortcut for unavailable items count.

```
class invenio_sip2.models.SelfcheckPatronStatus (patron_id,      institution_id,      pa-
                                                    tron_name="",      patron_status=None,
                                                    language='und', **kwargs)
```

Class representing patron information handler response.

Constructor.

:param patron_id - patron identifier (e.g. id, barcode, ...) :param institution_id - institution id (or code) of the patron :param patron_name - full name of the patron :param patron_status - status of the patron :param language - iso-639-2 language :param kwargs - optional fields

```
class invenio_sip2.models.SelfcheckPaymentType
```

Class to handle all available payment type.

CASH = '00'

CREDIT_CARD = '02'

VISA = '01'

```
class invenio_sip2.models.SelfcheckRenew (title_id, success=False, renewal=False, mag-
                                                    netic_media='unknown', desensitize='unknown',
                                                    **kwargs)
```

Class representing renew handler response.

Constructor.

Parameters

- **title_id** – title_id (e.g. title, identifier, ...)
- **renew** – renew operation is success or not should be set to True if the ACS renew the item. should be set to 0 if the ACS did not renew the item.
- **renewal** – renewal operation. should be set to True if the patron requesting to renew the item already has the item renewed. should be set to False if the item is not already renewed.
- **magnetic_media** – indicate the presence of magnetic media
- **desensitize** – desensitize an item ?
- **kwargs** – optional fields

desensitize
Shortcut for desensitize.

due_date
Shortcut for due date.

has_magnetic_media
Shortcut for desensitize.

is_renewal
Shortcut for renewal ok.

is_success
Shortcut for success operation.

class invenio_sip2.models.**SelfcheckSecurityMarkerType**
Class to handle all available security marker type.

NONE = '01'

OTHER = '00'

TATTLE_TAPE_SECURITY_STRIP = '02'

WHISPER_TAPE = '03'

class invenio_sip2.models.**SelfcheckSummary** (*text*)
Class representing summary.

Init.

fields = ['hold_items', 'overdue_items', 'charged_items', 'fine_items', 'recall_items']

is_needed (*key*)
Check if the given information is needed.

class invenio_sip2.models.**SupportedMessages**
Class to define supported messages from handler config.

Constructor.

add_supported_message (*handler*)
Add supported message.

handlers = ['patron_status', 'checkout', 'checkin', 'block_patron', 'system_status', '']

Rest API

API blueprint for Invenio-SIP2.

```
class invenio_sip2.views.rest.Monitoring
    Monitoring class.

    classmethod get_clients_by_server_id (server_id)
        Get list of clients by server id.

    classmethod get_number_client_by_server (server_id)
        Get total number of clients by server.

    classmethod get_servers ()
        Get list of servers.

    classmethod status ()
        Check status for all servers.

invenio_sip2.views.rest.get_clients ()
    Display all connected clients to server.

invenio_sip2.views.rest.get_server (server_id)
    Display all running SIP2 servers.

invenio_sip2.views.rest.get_servers ()
    Display all running SIP2 servers.

invenio_sip2.views.rest.status ()
    Display status for all SIP2 server.
```

Views

Blueprint for Invenio-SIP2.

```
invenio_sip2.views.views.monitoring ()
    Render a basic view.
```

1.3 Additional Notes

Notes on how to contribute, legal information and changes are here for the interested.

1.3.1 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

Types of Contributions

Report Bugs

Report bugs at <https://github.com/inveniosoftware-contrib/invenio-sip2/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

Write Documentation

Invenio-SIP2 could always use more documentation, whether as part of the official Invenio-SIP2 docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/inveniosoftware-contrib/invenio-sip2/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

Get Started!

Ready to contribute? Here’s how to set up *invenio-sip2* for local development.

1. Fork the *inveniosoftware-contrib/invenio-sip2* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/invenio-sip2.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv invenio-sip2
$ cd invenio-sip2/
$ pip install -e .[all]
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you’re done making changes, check that your changes pass tests:

```
$ ./run-tests.sh
```

The tests will provide you with test coverage and also check PEP8 (code style), PEP257 (documentation), flake8 as well as build the Sphinx documentation and run doctests.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -s
  -m "component: title without verbs"
  -m "* NEW Adds your new feature."
  -m "* FIX Fixes an existing issue."
  -m "* BETTER Improves and existing feature."
  -m "* Changes something that should not be visible in release notes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests and must not decrease test coverage.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring.
3. The pull request should work for python 3.6 to 3.9. Check https://travis-ci.org/inveniosoftware-contrib/invenio-sip2/pull_requests and make sure that the tests pass for all supported Python versions.

1.3.2 Changes

Version 0.6.19 (released 2022-12-19)

Bug fix:

- Fixes patron information summary field.

Version 0.6.18 (released 2022-12-16)

Implemented enhancements:

- Implements fee paid action.

Bug fix:

- Fixes readthedoc compilation.

Minor change:

- Adds more documentation.

Version 0.6.17 (released 2022-11-07)

Minor change:

- Uses http method on monitoring api blueprint.
- Updates Flask dependency.

Version 0.6.16 (released 2022-08-31)

Bug fix:

- Fixes project dependency.

Version 0.6.15 (released 2022-08-30)

Bug fix:

- Fixes sphinx dependency.

Version 0.6.14 (released 2022-08-30)

Bug fix:

- Fixes datastore local proxy.

Minor change:

- Changes logging level for *CommandNotFound* and *UnicodeDecodeError*.
- Moves to poetry.

Version 0.6.13 (released 2021-08-25)

Bug fix:

- Fixes request resend wrong decoding.

Minor change:

- Adds CommandNotFound exception.
- Catches all exceptions when reading the request.

Version 0.6.12 (released 2021-08-24)

Bug fix:

- Fix missing parsing date.
- Fix request resend checksum validation.

Minor change:

- Adds module version in logging formatter.

Version 0.6.11 (released 2021-08-17)

Bug fix:

- Forces text encoding on checksum generation

Version 0.6.10 (released 2021-08-09)

Minor change:

- Improves server logging.
- Ensures that the sequence number is present in the message if the selfcheck terminal sends it.
- Adds sequence number and checksum in dumped message.
- Adds cached property to extension.

Bug fix:

- Fixes circulation date parsing.
- Rewrites error detection for request and response message.

Version 0.6.9 (released 2021-07-28)

Bug fix:

- Fixes invenio-search version.
- Fixes invenio-db version.

Version 0.6.8 (released 2021-07-27)

Minor change:

- Catches runtime error.
- Uses pydocstyle and pycodestyle.
- Increase code coverage.
- Cleans code.

Version 0.6.7 (released 2021-07-19)

Bug fix:

- Fixes missing conversion of i18n language.
- Fixes date format.

Version 0.6.6 (released 2021-07-14)

Minor changes:

- Defines supported messages from handlers config.

Bug fix:

- Fixes empty patron session.
- Improves i18n language.

Version 0.6.5 (released 2021-07-12)

Minor changes:

- Logs more information for debugging

Version 0.6.4 (released 2021-06-30)

Bug fix:

- Fixes wrong circulation messages response.
- Fixes no such process in command line utilities.

Version 0.6.3 (released 2021-06-15)

Bug fix:

- Fixes error on renew action.

Version 0.6.2 (released 2021-06-14)

Minor changes:

- implement summary for patron information.
- Fixes fixed field wrong length.

Version 0.6.1 (released 2021-06-14)

Minor changes:

- Use invenio-sip2 logger for server error logs.

Version 0.6.0 (released 2021-06-11)

Implemented enhancements:

- Implements request resend action.

- Adds CLI to stop the server.
- Implements sequence number error detection.

Version 0.5.1 (released 2021-05-06)

Minor changes:

- Increase code coverage.
- Updates documentation.
- Cleans and rewrites code.

Version 0.5.0 (released 2021-03-25)

Implemented enhancements:

- Adds datastore to save clients and servers state.
- Adds record metadata management.
- Adds APIs to monitor servers and clients.
- Implements specific logger to log selfcheck requests and server responses.

Version 0.4.0 (released 2020-11-26)

Implemented enhancements:

- Implements Patron status action.
- Moves to github action for continuous Integration.

Fixed bugs:

- Increase code coverage

Version 0.3.0 (released 2020-10-13)

Implemented enhancements:

- Adds Item information action.
- Implements circulation actions
- Adds base of patron session.
- Uses pycountry for language management.

Fixed bugs:

- Missing line terminator to tell to client that all bytes are sent.

Version 0.2.0 (released 2020-08-10)

Implemented enhancements:

- Implements Patron information action.
- Adds Remote ILS handlers configuration.

Version 0.1.0 (released 2020-05-25)

- Base of automated circulation system.

1.3.3 License

Note: In applying this license, CERN does not waive the privileges and immunities granted to it by virtue of its status as an Intergovernmental Organization or submit itself to any jurisdiction.

INVENIO-SIP2 Copyright (C) 2020 UCLouvain

This program is free software: you can redistribute it and/or modify it under the terms of the GNU Affero General Public License as published by the Free Software Foundation, version 3 of the License.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Affero General Public License for more details.

You should have received a copy of the GNU Affero General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

1.3.4 Authors

Invenio module that add SIP2 communication for self-check

- Laurent Dubois (UCLouvain) <laurent.dubois@uclouvain.be>

i

- `invenio_sip2.actions.actions`, [8](#)
- `invenio_sip2.config`, [4](#)
- `invenio_sip2.ext`, [7](#)
- `invenio_sip2.models`, [15](#)
- `invenio_sip2.records.record`, [12](#)
- `invenio_sip2.views.rest`, [21](#)
- `invenio_sip2.views.views`, [22](#)

A

`add_console_handler()` (invenio_sip2.ext.InvenioSIP2 method), 7
`add_fs_handler()` (invenio_sip2.ext.InvenioSIP2 method), 7
`add_patron_status_type()` (invenio_sip2.models.PatronStatus method), 15
`add_supported_message()` (invenio_sip2.models.SupportedMessages method), 21
ADMINISTRATIVE (invenio_sip2.models.SelfcheckFeeType attribute), 17
ARABIC (invenio_sip2.models.SelfcheckLanguage attribute), 18
AUDIO (invenio_sip2.models.SelfcheckMediaType attribute), 19
AutomatedCirculationSystemStatus (class in invenio_sip2.actions.actions), 8
AVAILABLE (invenio_sip2.models.SelfcheckCirculationStatus attribute), 16

B

BABEL_DEFAULT_LANGUAGE (in module invenio_sip2.config), 5
BELGIAN (invenio_sip2.models.SelfcheckLanguage attribute), 18
BlockPatron (class in invenio_sip2.actions.actions), 9
BOOK (invenio_sip2.models.SelfcheckMediaType attribute), 19
BOOK_WHIT_AUDIO_TAPE (invenio_sip2.models.SelfcheckMediaType attribute), 19
BOOK_WHIT_CD (invenio_sip2.models.SelfcheckMediaType attribute), 19
BOOK_WHIT_DISKETTE (invenio_sip2.models.SelfcheckMediaType attribute), 19

BOUND_JOURNAL (invenio_sip2.models.SelfcheckMediaType attribute), 19

C

CANADIAN_FRENCH (invenio_sip2.models.SelfcheckLanguage attribute), 18
CARD_REPORTED_LOST (invenio_sip2.models.PatronStatusTypes attribute), 15
CASH (invenio_sip2.models.SelfcheckPaymentType attribute), 20
CD_OR_CDROM (invenio_sip2.models.SelfcheckMediaType attribute), 19
CHARGE_PRIVILEGES_DENIED (invenio_sip2.models.PatronStatusTypes attribute), 15
CHARGED (invenio_sip2.models.SelfcheckCirculationStatus attribute), 16
charged_items_count (invenio_sip2.models.SelfcheckPatronInformation attribute), 20
CHARGED_RECALL (invenio_sip2.models.SelfcheckCirculationStatus attribute), 16
Checkin (class in invenio_sip2.actions.actions), 9
Checkout (class in invenio_sip2.actions.actions), 9
chi (invenio_sip2.models.SelfcheckLanguage attribute), 18
CHINESE (invenio_sip2.models.SelfcheckLanguage attribute), 18
CLAIMED_RETURNED (invenio_sip2.models.SelfcheckCirculationStatus attribute), 16
clear_all_clients() (invenio_sip2.records.record.Server method), 13
clear_patron_session() (invenio_sip2.records.record.Client method),

13
Client (class in invenio_sip2.records.record), 12
COMPUTER_ACCESS_CHARGE (invenio_sip2.models.SelfcheckFeeType attribute), 17
count() (invenio_sip2.records.record.Sip2RecordMetadata class method), 14
create() (invenio_sip2.records.record.Server class method), 13
create() (invenio_sip2.records.record.Sip2RecordMetadata class method), 14
CREDIT_CARD (invenio_sip2.models.SelfcheckPaymentType attribute), 20

D

DAMAGE (invenio_sip2.models.SelfcheckFeeType attribute), 17
dan (invenio_sip2.models.SelfcheckLanguage attribute), 18
DANISH (invenio_sip2.models.SelfcheckLanguage attribute), 18
delete() (invenio_sip2.records.record.Server method), 14
delete() (invenio_sip2.records.record.Sip2RecordMetadata method), 14
desensitize (invenio_sip2.models.SelfcheckCheckout attribute), 16
desensitize (invenio_sip2.models.SelfcheckRenew attribute), 21
DISKETTE (invenio_sip2.models.SelfcheckMediaType attribute), 19
down() (invenio_sip2.records.record.Server method), 14
due_date (invenio_sip2.models.SelfcheckCheckout attribute), 16
due_date (invenio_sip2.models.SelfcheckRenew attribute), 21
dumps() (invenio_sip2.records.record.Sip2RecordMetadata method), 14
dut (invenio_sip2.models.SelfcheckLanguage attribute), 18
DUTCH (invenio_sip2.models.SelfcheckLanguage attribute), 18

E

EndPatronSession (class in invenio_sip2.actions.actions), 9
eng (invenio_sip2.models.SelfcheckLanguage attribute), 18
ENGLISH (invenio_sip2.models.SelfcheckLanguage attribute), 18
EXCESSIVE_OUTSTANDING_FEES (invenio_sip2.models.PatronStatusTypes attribute), 15
EXCESSIVE_OUTSTANDING_FINES (invenio_sip2.models.PatronStatusTypes attribute), 15
execute() (invenio_sip2.actions.actions.AutomatedCirculationSystemStatus method), 9
execute() (invenio_sip2.actions.actions.BlockPatron method), 9
execute() (invenio_sip2.actions.actions.Checkin method), 9
execute() (invenio_sip2.actions.actions.Checkout method), 9
execute() (invenio_sip2.actions.actions.EndPatronSession method), 10
execute() (invenio_sip2.actions.actions.FeePaid method), 10
execute() (invenio_sip2.actions.actions.Hold method), 10
execute() (invenio_sip2.actions.actions.ItemInformation method), 10
execute() (invenio_sip2.actions.actions.ItemStatusUpdate method), 11
execute() (invenio_sip2.actions.actions.PatronEnable method), 11
execute() (invenio_sip2.actions.actions.PatronInformation method), 11
execute() (invenio_sip2.actions.actions.PatronStatus method), 11
execute() (invenio_sip2.actions.actions.Renew method), 12
execute() (invenio_sip2.actions.actions.RenewAll method), 12
execute() (invenio_sip2.actions.actions.RequestResend method), 12
execute() (invenio_sip2.actions.actions.SelfCheckLogin method), 12

F

FeePaid (class in invenio_sip2.actions.actions), 10
fields (invenio_sip2.models.SelfcheckSummary attribute), 21
fin (invenio_sip2.models.SelfcheckLanguage attribute), 19
find_server() (invenio_sip2.records.record.Server class method), 14
fine_items_count (invenio_sip2.models.SelfcheckPatronInformation attribute), 20
FINNISH (invenio_sip2.models.SelfcheckLanguage attribute), 18
fra (invenio_sip2.models.SelfcheckLanguage attribute), 19
fre (invenio_sip2.models.SelfcheckLanguage attribute), 19

FRENCH (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

G

ger (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

GERMAN (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

get_all_records() (*invenio_sip2.records.record.Sip2RecordMetadata class method*), 14

get_clients() (*in module invenio_sip2.views.rest*), 22

get_clients() (*invenio_sip2.records.record.Server method*), 14

get_clients_by_server_id() (*invenio_sip2.views.rest.Monitoring class method*), 22

get_current_patron_session() (*invenio_sip2.records.record.Client method*), 13

get_key() (*invenio_sip2.records.record.Client method*), 13

get_key() (*invenio_sip2.records.record.Sip2RecordMetadata class method*), 14

get_logging_formatter() (*invenio_sip2.ext.InvenioSIP2 class method*), 7

get_number_client_by_server() (*invenio_sip2.views.rest.Monitoring class method*), 22

get_record_by_id() (*invenio_sip2.records.record.Sip2RecordMetadata class method*), 14

get_server() (*in module invenio_sip2.views.rest*), 22

get_server() (*invenio_sip2.records.record.Client method*), 13

get_servers() (*in module invenio_sip2.views.rest*), 22

get_servers() (*invenio_sip2.views.rest.Monitoring class method*), 22

gre (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

GREEK (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

H

handlers (*invenio_sip2.models.SupportedMessages attribute*), 21

has_magnetic_media (*invenio_sip2.models.SelfcheckCheckin attribute*), 15

has_magnetic_media (*invenio_sip2.models.SelfcheckCheckout attribute*), 16

has_magnetic_media (*invenio_sip2.models.SelfcheckRenew attribute*), 21

heb (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

HEBREW (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

Hold (*class in invenio_sip2.actions.actions*), 10

HOLD_FEE (*invenio_sip2.models.SelfcheckFeeType attribute*), 17

hold_items_count (*invenio_sip2.models.SelfcheckPatronInformation attribute*), 20

HOLD_PRIVILEGES_DENIED (*invenio_sip2.models.PatronStatusTypes attribute*), 15

I

ice (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

ICELANDIC (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

id (*invenio_sip2.records.record.Sip2RecordMetadata attribute*), 14

IN_PROCESS (*invenio_sip2.models.SelfcheckCirculationStatus attribute*), 16

IN_TRANSIT (*invenio_sip2.models.SelfcheckCirculationStatus attribute*), 16

init_app() (*invenio_sip2.ext.InvenioSIP2 method*), 7

init_config() (*invenio_sip2.ext.InvenioSIP2 method*), 7

institution_id (*invenio_sip2.records.record.Client attribute*), 13

invenio_sip2.actions.actions (*module*), 8

invenio_sip2.config (*module*), 4

invenio_sip2.ext (*module*), 7

invenio_sip2.models (*module*), 15

invenio_sip2.records.record (*module*), 12

invenio_sip2.views.rest (*module*), 21

invenio_sip2.views.views (*module*), 22

InvenioSIP2 (*class in invenio_sip2.ext*), 7

is_accepted (*invenio_sip2.models.SelfcheckFeePaid attribute*), 17

is_authenticated (*invenio_sip2.records.record.Client attribute*), 13

is_available (*invenio_sip2.models.SelfcheckHold attribute*), 17

is_error_detection_enabled (*invenio_sip2.ext.InvenioSIP2 attribute*), 7

is_needed() (*invenio_sip2.models.SelfcheckSummary method*), 21

is_renewal (*invenio_sip2.models.SelfcheckCheckout attribute*), 16

is_renewal (*invenio_sip2.models.SelfcheckRenew attribute*), 21

is_running (*invenio_sip2.records.record.Server attribute*), 14

is_success (*invenio_sip2.models.SelfcheckCheckin attribute*), 15

is_success (*invenio_sip2.models.SelfcheckCheckout attribute*), 16

is_success (*invenio_sip2.models.SelfcheckHold attribute*), 17

is_success (*invenio_sip2.models.SelfcheckRenew attribute*), 21

isl (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

ita (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

ITALIAN (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

ItemInformation (class in *invenio_sip2.actions.actions*), 10

ItemStatusUpdate (class in *invenio_sip2.actions.actions*), 10

J

JAPANESE (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

jpn (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

K

kor (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

KOREAN (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

L

last_request_message (*invenio_sip2.records.record.Client attribute*), 13

last_response_message (*invenio_sip2.records.record.Client attribute*), 13

last_sequence_number (*invenio_sip2.records.record.Client attribute*), 13

library_language (*invenio_sip2.records.record.Client attribute*), 13

library_name (*invenio_sip2.records.record.Client attribute*), 13

line_terminator (*invenio_sip2.ext.InvenioSIP2 attribute*), 7

load_fixed_field() (*in module invenio_sip2.ext*), 8

load_variable_field() (*in module invenio_sip2.ext*), 8

LOST (*invenio_sip2.models.SelfcheckCirculationStatus attribute*), 16

M

MAGAZINE (*invenio_sip2.models.SelfcheckMediaType attribute*), 19

MALAY (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

may (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

MISSING (*invenio_sip2.models.SelfcheckCirculationStatus attribute*), 16

Monitoring (class in *invenio_sip2.views.rest*), 21

monitoring() (*in module invenio_sip2.views.views*), 22

msa (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

N

NONE (*invenio_sip2.models.SelfcheckSecurityMarkerType attribute*), 21

nor (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

NORTH_AMERICAN_SPANISH (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

NORWEGIAN (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

number_of_clients (*invenio_sip2.records.record.Server attribute*), 14

O

ON_ORDER (*invenio_sip2.models.SelfcheckCirculationStatus attribute*), 16

OTHER (*invenio_sip2.models.SelfcheckCirculationStatus attribute*), 16

OTHER (*invenio_sip2.models.SelfcheckFeeType attribute*), 17

OTHER (*invenio_sip2.models.SelfcheckMediaType attribute*), 19

OTHER (*invenio_sip2.models.SelfcheckSecurityMarkerType attribute*), 21

OVERDUE (*invenio_sip2.models.SelfcheckFeeType attribute*), 17

overdue_items_count (*invenio_sip2.models.SelfcheckPatronInformation attribute*), 20

P

patron_id (*invenio_sip2.models.SelfcheckPatronInformation attribute*), 20

patron_name (*invenio_sip2.models.SelfcheckPatronInformation attribute*), 20

PatronEnable (*class in invenio_sip2.actions.actions*), 11

PatronInformation (*class in invenio_sip2.actions.actions*), 11

PatronStatus (*class in invenio_sip2.actions.actions*), 11

PatronStatus (*class in invenio_sip2.models*), 15

PatronStatusTypes (*class in invenio_sip2.models*), 15

pol (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

POLISH (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

por (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

PORTUGUESE (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

PROCESSING (*invenio_sip2.models.SelfcheckFeeType attribute*), 17

R

recall_items_count (*invenio_sip2.models.SelfcheckPatronInformation attribute*), 20

RECALL_OVERDUE (*invenio_sip2.models.PatronStatusTypes attribute*), 15

RECALL_PRIVILEGES_DENIED (*invenio_sip2.models.PatronStatusTypes attribute*), 15

RECALLED (*invenio_sip2.models.SelfcheckCirculationStatus attribute*), 16

remote_app (*invenio_sip2.records.record.Client attribute*), 13

Renew (*class in invenio_sip2.actions.actions*), 11

RENEWAL_PRIVILEGES_DENIED (*invenio_sip2.models.PatronStatusTypes attribute*), 15

RenewAll (*class in invenio_sip2.actions.actions*), 12

RENTAL (*invenio_sip2.models.SelfcheckFeeType attribute*), 17

REPLACEMENT (*invenio_sip2.models.SelfcheckFeeType attribute*), 17

RequestResend (*class in invenio_sip2.actions.actions*), 12

resensitize (*invenio_sip2.models.SelfcheckCheckin attribute*), 16

retries_allowed (*invenio_sip2.ext.InvenioSIP2 attribute*), 7

rus (*invenio_sip2.models.SelfcheckLanguage attribute*), 19

RUSSIAN (*invenio_sip2.models.SelfcheckLanguage attribute*), 18

S

search () (*invenio_sip2.records.record.Sip2RecordMetadata method*), 14

SelfcheckCheckin (*class in invenio_sip2.models*), 15

SelfcheckCheckout (*class in invenio_sip2.models*), 16

SelfcheckCirculationStatus (*class in invenio_sip2.models*), 16

SelfcheckEnablePatron (*class in invenio_sip2.models*), 16

SelfcheckFeePaid (*class in invenio_sip2.models*), 17

SelfcheckFeeType (*class in invenio_sip2.models*), 17

SelfcheckHold (*class in invenio_sip2.models*), 17

SelfcheckItemInformation (*class in invenio_sip2.models*), 17

SelfcheckLanguage (*class in invenio_sip2.models*), 18

SelfCheckLogin (*class in invenio_sip2.actions.actions*), 12

SelfcheckMediaType (*class in invenio_sip2.models*), 19

SelfcheckPatronInformation (*class in invenio_sip2.models*), 19

SelfcheckPatronStatus (*class in invenio_sip2.models*), 20

SelfcheckPaymentType (*class in invenio_sip2.models*), 20

SelfcheckRenew (*class in invenio_sip2.models*), 20

SelfcheckSecurityMarkerType (*class in invenio_sip2.models*), 21

SelfcheckSummary (*class in invenio_sip2.models*), 21

Server (*class in invenio_sip2.records.record*), 13

server_id (*invenio_sip2.records.record.Client attribute*), 13

sip2 (*invenio_sip2.ext.InvenioSIP2 attribute*), 8

SIP2_CIRCULATION_DATE_FORMAT (*in module invenio_sip2.config*), 5

sip2_current_date (*invenio_sip2.ext.InvenioSIP2 attribute*), 8

SIP2_DATE_FORMAT (*in module invenio_sip2.config*), 5

SIP2_DEFAULT_LANGUAGE (*in module invenio_sip2.config*), 5

SIP2_DEFAULT_SECURITY_MARKER (*in module invenio_sip2.config*), 5

SIP2_ERROR_DETECTION (in module *invenio_sip2.config*), 5
sip2_handlers (*invenio_sip2.ext.InvenioSIP2* attribute), 8
sip2_language (*invenio_sip2.ext.InvenioSIP2* attribute), 8
SIP2_LINE_TERMINATOR (in module *invenio_sip2.config*), 5
SIP2_LOGGING_CONSOLE (in module *invenio_sip2.config*), 5
SIP2_LOGGING_CONSOLE_LEVEL (in module *invenio_sip2.config*), 5
SIP2_LOGGING_FS_BACKUPCOUNT (in module *invenio_sip2.config*), 5
SIP2_LOGGING_FS_LEVEL (in module *invenio_sip2.config*), 6
SIP2_LOGGING_FS_LOGFILE (in module *invenio_sip2.config*), 6
SIP2_LOGGING_FS_MAXBYTES (in module *invenio_sip2.config*), 6
sip2_message_types (*invenio_sip2.ext.InvenioSIP2* attribute), 8
SIP2_PERMISSIONS_FACTORY() (in module *invenio_sip2.config*), 6
SIP2_PROTOCOL (in module *invenio_sip2.config*), 6
SIP2_REMOTE_ACTION_HANDLERS (in module *invenio_sip2.config*), 6
SIP2_RETRIES_ALLOWED (in module *invenio_sip2.config*), 6
SIP2_SOCKET_BUFFER_SIZE (in module *invenio_sip2.config*), 6
SIP2_SUPPORT_CHECKIN (in module *invenio_sip2.config*), 6
SIP2_SUPPORT_CHECKOUT (in module *invenio_sip2.config*), 6
SIP2_SUPPORT_OFFLINE_STATUS (in module *invenio_sip2.config*), 6
SIP2_SUPPORT_ONLINE_STATUS (in module *invenio_sip2.config*), 6
SIP2_SUPPORT_RENEWAL_POLICY (in module *invenio_sip2.config*), 6
SIP2_SUPPORT_STATUS_UPDATE (in module *invenio_sip2.config*), 6
SIP2_TEXT_ENCODING (in module *invenio_sip2.config*), 6
SIP2_TIMEOUT_PERIOD (in module *invenio_sip2.config*), 6
Sip2RecordMetadata (class in *invenio_sip2.records.record*), 14
sound_alert (*invenio_sip2.models.SelfcheckCheckin* attribute), 16
spa (*invenio_sip2.models.SelfcheckLanguage* attribute), 19
SPANISH (*invenio_sip2.models.SelfcheckLanguage* attribute), 18
status() (in module *invenio_sip2.views.rest*), 22
status() (*invenio_sip2.views.rest.Monitoring* class method), 22
support_checkin (*invenio_sip2.ext.InvenioSIP2* attribute), 8
support_checkout (*invenio_sip2.ext.InvenioSIP2* attribute), 8
support_offline_status (*invenio_sip2.ext.InvenioSIP2* attribute), 8
support_online_status (*invenio_sip2.ext.InvenioSIP2* attribute), 8
support_renewal_policy (*invenio_sip2.ext.InvenioSIP2* attribute), 8
support_status_update (*invenio_sip2.ext.InvenioSIP2* attribute), 8
supported_messages() (*invenio_sip2.ext.InvenioSIP2* method), 8
supported_protocol (*invenio_sip2.ext.InvenioSIP2* attribute), 8
SupportedMessages (class in *invenio_sip2.models*), 21
swe (*invenio_sip2.models.SelfcheckLanguage* attribute), 19
SWEDISH (*invenio_sip2.models.SelfcheckLanguage* attribute), 18

T

TAIWANESE (*invenio_sip2.models.SelfcheckLanguage* attribute), 18
tam (*invenio_sip2.models.SelfcheckLanguage* attribute), 19
TAMIL (*invenio_sip2.models.SelfcheckLanguage* attribute), 18
TATTLE_TAPE_SECURITY_STRIP (*invenio_sip2.models.SelfcheckSecurityMarkerType* attribute), 21
terminal (*invenio_sip2.records.record.Client* attribute), 13
text_encoding (*invenio_sip2.ext.InvenioSIP2* attribute), 8
timeout_period (*invenio_sip2.ext.InvenioSIP2* attribute), 8
TOO_MANY_CLAIMS_OF_ITEMS_RETURNED (*invenio_sip2.models.PatronStatusTypes* attribute), 15
TOO_MANY_ITEMS_BILLED (*invenio_sip2.models.PatronStatusTypes* attribute), 15
TOO_MANY_ITEMS_CHARGED (*invenio_sip2.models.PatronStatusTypes* attribute), 15
TOO_MANY_ITEMS_LOST (*invenio_sip2.models.PatronStatusTypes* attribute), 15

15
 TOO_MANY_ITEMS_OVERDUE (invenio_sip2.models.PatronStatusTypes attribute), 15
 TOO_MANY_RENEWALS (invenio_sip2.models.PatronStatusTypes attribute), 15
 transaction_user_id (invenio_sip2.records.record.Client attribute), 13

U

unavailable_items_count (invenio_sip2.models.SelfcheckPatronInformation attribute), 20
 und (invenio_sip2.models.SelfcheckLanguage attribute), 19
 UNITED_KINGDOM (invenio_sip2.models.SelfcheckLanguage attribute), 18
 UNKNOWN (invenio_sip2.models.SelfcheckLanguage attribute), 18
 up() (invenio_sip2.records.record.Server method), 14
 update() (invenio_sip2.records.record.Sip2RecordMetadata method), 14

V

VIDEO (invenio_sip2.models.SelfcheckMediaType attribute), 19
 VISA (invenio_sip2.models.SelfcheckPaymentType attribute), 20

W

WAITING_ON_HOLD_SHELF (invenio_sip2.models.SelfcheckCirculationStatus attribute), 16
 WAITING_TO_RESHELF (invenio_sip2.models.SelfcheckCirculationStatus attribute), 16
 WHISPER_TAPE (invenio_sip2.models.SelfcheckSecurityMarkerType attribute), 21

Z

zho (invenio_sip2.models.SelfcheckLanguage attribute), 19