telemetry

Apr 10, 2020

Table of Contents:

Configurable event-logging for Jupyter applications and extensions.

Telemetry provides a configurable traitlets object, EventLog, for structured event-logging in Python. It leverages Python's standard logging library for filtering, handling, and recording events. All events are validated (using jsonschema) against registered JSON schemas.

If you're looking for telemetry in Jupyter frontend applications (like JupyterLab), checkout the work happening in jupyterlab-telemetry!

CHAPTER 1

Installation

Jupyter's Telemetry library can be installed from PyPI.

CHAPTER 2

Basic Usage

Here's a basic example of an EventLog.

```
import logging
from jupyter_telemetry import EventLog

eventlog = EventLog(
    # Use logging handlers to route where events
    # should be record.
    handlers=[
        logging.FileHandler('events.log')
    ],
    # List schemas of events that should be recorded.
    allowed_schemas=[
        'uri.to.event.schema'
    ]
)
```

EventLog has two configurable traits:

- `handlers`: a list of Python's logging handlers.
- `allowed_schemas`: a list of event schemas to record.

Event schemas must be registered with the EventLog for events to be recorded. An event schema looks something like:

```
{
    "$id": "url.to.event.schema",
    "title": "My Event",
    "description": "All events must have a name property.",
    "type": "object",
    "properties": {
        "name": {
            "title": "Name",
            "description": "Name of event",
```

(continues on next page)

(continued from previous page)

```
            "type": "string"
        }
    },
    "required": ["name"],
    "version": 1
}
```

Two fields are required:

- `$id`: a valid URI to identify the schema (and possibly fetch it from a remote address).
- `version`: the version of the schema.

The other fields follow standard JSON schema structure.

Schemas can be registered from a Python dict object, a file, or a URL. This example loads the above example schema from file.

```
# Record an example event.
event = {'name': 'example event'}
eventlog.record_event(
    schema_id='url.to.event.schema',
    version=1,
    event=event
)
```

2.1 Writing a Schema

Schemas should follow valid JSON schema. These schemas can be written in valid YAML or JSON.

At a minimum, valid schemas should have the following keys:

- `$id` : a valid URL where the schema lives.
- `version` : schema version.
- `title` : name of the schema
- `description` : documentation for the schema
- `properties` : attributes of the event being emitted.

  Each property should have the following attributes:

  - `title` : name of the property
  - `description`: documentation for this property.
  - `pii`: (optional) boolean for whether this property is personally identifiable information or not.
- `required`: list of required properties.

Here is a minimal example of a valid JSON schema for an event.

```
$id: url.to.event.schema
version: 1
title: My Event
description: |
  All events must have a name property
type: object
```

(continues on next page)

(continued from previous page)

```
properties:
  name:
    title: Name
    description: |
        Name of event
    type: string
required:
- name
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search