Indico – Offline Event Storage

Ana Clara L. Siqueira

CERN openlab
16th August 2012

oTN-2011-01 openlab Summer Student Report

Indico – Offline Event Storage

Ana C. Siqueira
Supervisor: Jose Benito G. Lopez
16th August 2012
Version 1
Distribution: **Public**

Table of Contents

Abstract

The goal of this openlab summer student project was to design and implement a flexible, maintainable, and extensible procedure to generate static pages for conferences, events and lectures on Indico, with relative hyperlinks, the same CSS style, functional JavaScript, and so forth. Furthermore, also integrate the chosen procedure into the core of Indico and provide a way by which the user could request the generation of offline events.

Introduction

In this report an introduction to the Indico Project and its features will be given, as also an explanation of its architeture. The second part discusses how it was implemented the new feature during the openlab summer student project and what the next steps are so that the feature can be deployed.

1 Indico Project

1.1 What does Indico Means?

Indico (Integrated Digital Conference) is a web-based, world-wide, multi-platform application to schedule and organise events, from simple lectures to complex meetings, workshops and conferences with sessions and contributions. The tool also includes an advanced user delegation mechanism, allow paper reviewing, archival of conference information and electronic proceedings. The software is used in more than 100 institutes worldwide and extensively at CERN, where it hosts more than 190.000 events and it is visited by around 12.000 users per day.
The Indico software was originally developed in the framework of the EU InDiCo project. Nowadays, Indico is a free software licensed under terms of GNU General Public License (**GPL**).

1.2 Why offline?

Many of the Indico events are important conferences whose organizers would like to distribute USB keys/CDs/DVDs with an offline version of their Indico conference. It can also happen that the conference and its slides needs to be presented/displayed in a place with no Internet connection.

1.3 Features

In order to better understand the necessary modifications to provide an offline event here are some features that can be found at the event pages:

- User interactions

This feature include the registration, abstract submission, paper submission, slide upload and more;

- Flexible event representation

Allows different view of the same event, provides access to the timetable and various listings (participants, contributions...), and stores all kind of files including pictures, transparencies, minutes, videos, etc.

- Full management of conference cycle

Easy setup of access and modifications rights, easy creation of timetable and customization of the conference website;

To generate the offline event pages only the features included at *Flexible event representation* was maintained once that the others mentioned above can not be used without Internet connection.

1.4 Architecture

The Indico architecture is called *Layer cake* and each layer is self-contained. As shown in the *Figure 1* the application is divided in request handlers (RH), Web pages (WP*), Components(W*) and Templates(*.tpl).

The *Figure 2* and *Figure 3* is shown the directory structure of Indico, emphasizing some folders:

- code/MaKac – the "heart" of Indico;
- code/htdocs – the low-level request handlers;
- code/tpl – where the template files are found;
- webinterface/rh – Indico request handlers;
- services/implementation – Asynchronous services;

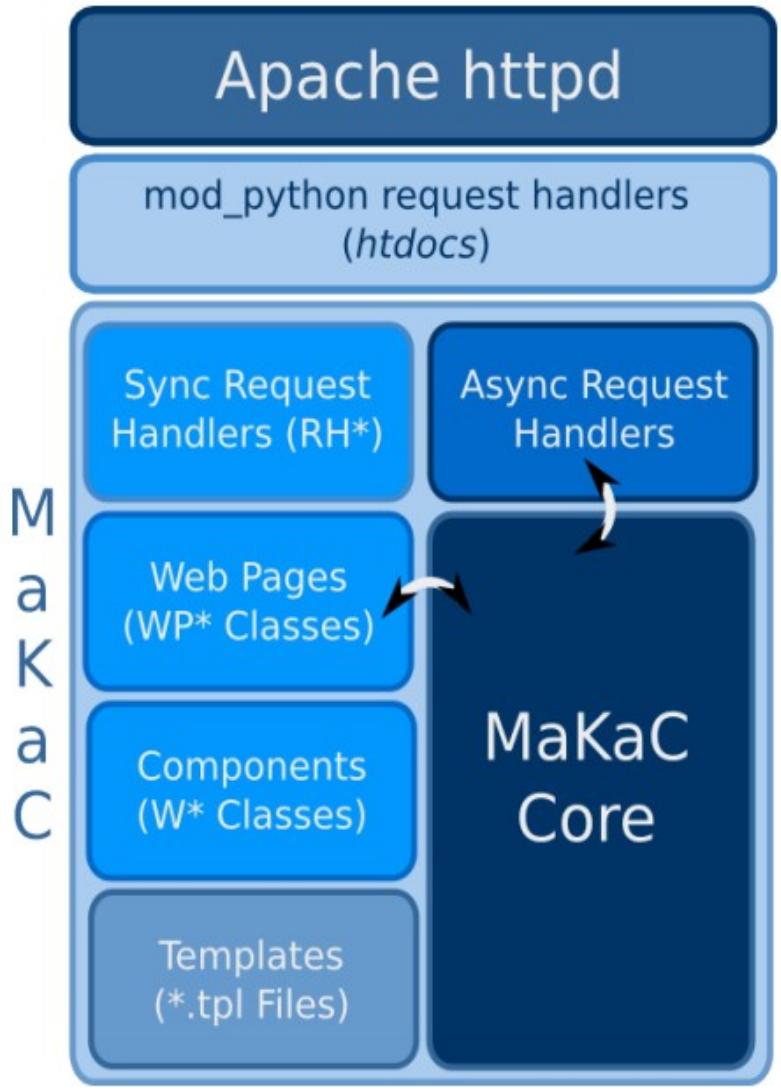

Figure 1: Layer Cake Architecture

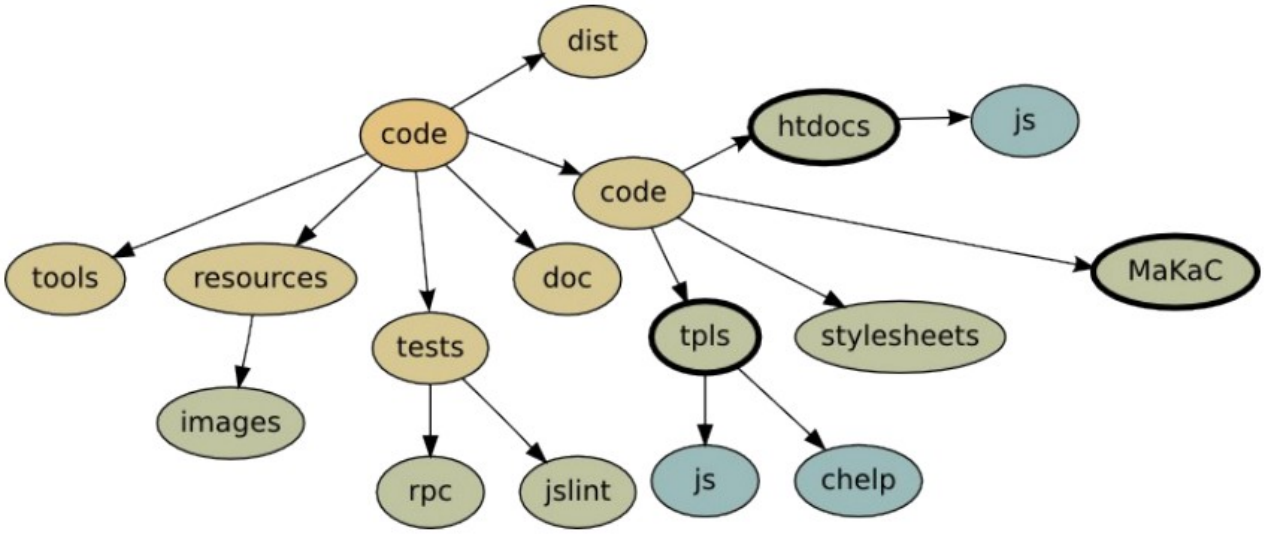

Figure 2: Directory Structure

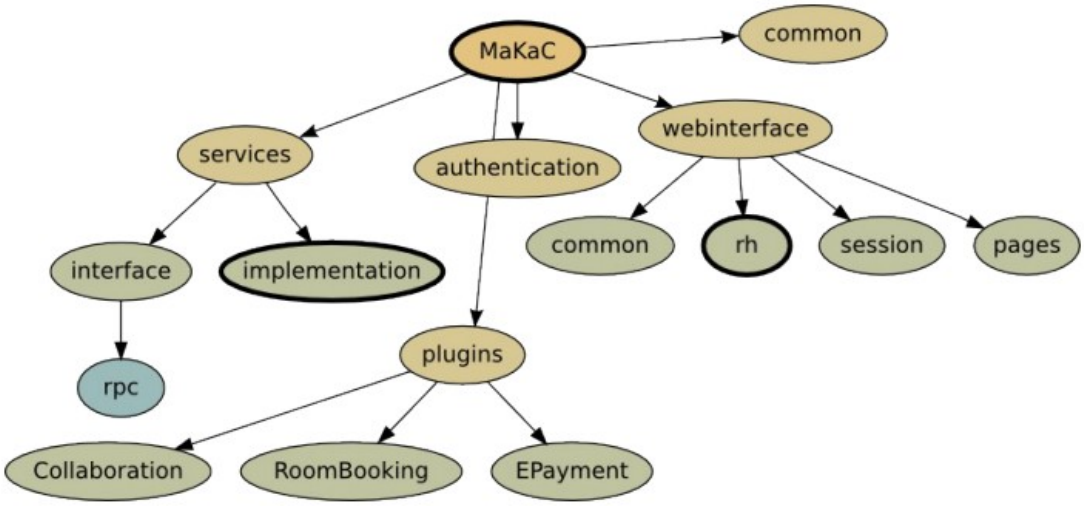

Figure 3: Directory Structure

2 Offline Website Creator

2.1 General idea

The general idea of providing offline events is to allow the user to view static pages with the same style as the online version, download materials (slides, minutes, etc) and navigate through the hyperlinks and pages.
The static contents are generated dinamically at the time that the Offline Website Creator is called. This component creates one ZIP file to storage every static file created, as well as stores all the images, javascript files and CSS files that are really used for the generated pages.
There are three type of events that can be hosted on Indico:

- Lecture: simple event to announce a talk;
- Meeting: an event that defines an agenda with many talks;
- Conference: is a complex event with features to manage the whole conference life cycle;

The Offline Website Creator is able to generate offline pages for each one of them. These three types of events pages share some characteristics as images, CSS, javascript and materials to be downloaded as also disablement of edition and configuration buttons and this is done by the default `OfflineEventCreator`.
Otherwise, Indico conferences needs more care. Some menu items that appear on the online conference page might not be shown (the complete conference menu can be viewed on *Figure 4*), there are: *Call for abstracts*, *My conference*, *Paper reviewing*, *Registration* and *Evaluation*. These specifics changes for conferences are implemented by `ConferenceOfflineCreator`.

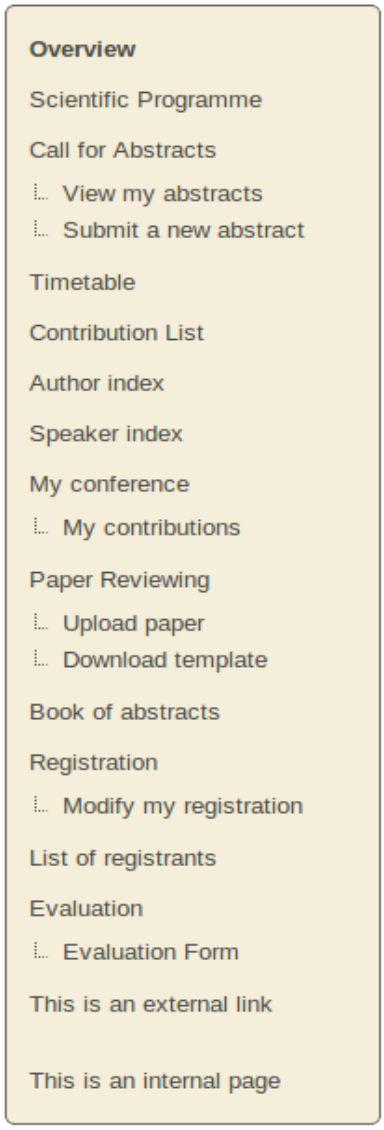

Figure 4: Complete menu items

2.2 Implementation

The main part of the implementation of the new module Offline Website Creator is presented below.

```
class OfflineEvent:

    def __init__(self, rh, conf, eventType, html):
        self._rh = rh
        self._conf = conf
        self._eventType = eventType
        self._html = html

    def create(self):
        if self._eventType in ("simple_event", "meeting"):
```

```
            websiteCreator = OfflineEventCreator(self._rh, self._conf,
self._html)
        elif self._eventType == "conference":
            websiteCreator = ConferenceOfflineCreator(self._rh, self._conf,
self._html)
        return websiteCreator.create()
```

```
class OfflineEventCreator:

    def __init__(self, rh, conf, html):
        self._rh = rh
        self._conf = conf
        self._html = html
        self._outputFileName = ""
        self._fileHandler = None
        self._mainPath = ""
        self._staticPath = ""

    def create(self):
        try:
            self._fileHandler = ZIPFileHandler()
            # Create main and static folders
            self._mainPath = "OfflineWebsite-%s" %
self._normalisePath(self._conf.getTitle())
            self._fileHandler.addDir(self._mainPath)
            self._staticPath = os.path.join(self._mainPath, "static")
            self._fileHandler.addDir(self._staticPath)

            # Getting all materials, static files (css, images, js and
vars.js.tpl)
            self._getAllMaterial()
            self._html = self._getStaticFiles(self._html)
            # Specific changes
            self._create()

            fname = os.path.join(self._mainPath,
urlHandlers.UHConferenceDisplay.getStaticURL())
            self._fileHandler.addNewFile(fname, self._html)
            self._fileHandler.close()
            self._outputFileName =
self._generateZipFile(self._fileHandler.getPath())
        except Exception, e:
            Logger.get('offline-website').error(str(e))
        self._fileHandler = None
        return self._outputFileName
```

```
class ConferenceOfflineCreator(OfflineEventCreator):

    # Menu itens allowed to be shown in offline mode
    _menu_offline_itens = {"overview" : None, \
                           "programme" : None, \
                           "timetable" : None, \
                           "authorIndex" : None, \
```

```
                            "speakerIndex" : None, \
                            "contributionList" : None, \
                            "registrants" : None,
                            "abstractsBook" : None}

    def _create(self):
        self._initializeMenuItensComponents()
        # Getting conference logo
        self._addLogo()
        # Getting all menu itens
        self._getMenuItens()
        # Getting specific pages for contributions
        for cont in self._conf.getContributionList():
            self._getContrib(cont)
```

2.3 Modifications

Most of the modifications made in the code are in fact made in new files dedicated specially for the generation and creation of static pages, the new file is called *common/offlineWebsiteCreator.py*. Some others changes:

- Creation of static components to take off the Header and Footer of the Templates and to do some other needed specifics changes in each original component;

```
class WPStaticEventBase:

    def _getHeader(self):
        return ""

    def _getFooter(self):
        return ""

    def getJSFiles(self):
        return self._asset_env['base_js'].urls()


class WPStaticConferenceTimeTable(WPStaticEventBase, WPConferenceTimeTable):

    def getJSFiles(self):
        return WPStaticEventBase.getJSFiles(self) +
self._includeJSPackage('Timetable')

class WPStaticConferenceProgram(WPStaticEventBase, WPConferenceProgram):
    pass

class WPStaticContributionList(WPStaticEventBase, WPContributionList):

    def _getBody(self, params):
        from MaKaC.webinterface.rh.conferenceDisplay import RHContributionList
        # Getting an contribution list empty filter
```

```
        filterCriteria = RHContributionList.create_filter(self, self._conf,
params)
        wc = WConfContributionList(self._getAW(), self._conf, filterCriteria,
"")
        return wc.getHTML()
```

- Adding a staticURL in each used RequestHandler

```
@classmethod
    def getURL( cls, target=None, **params ):
        """Gives the full URL for the corresponding request handler. In case
            the target parameter is specified it will append to the URL the
            the necessary parameters to make the target be specified in the
            url.

            Parameters:
                target - (Locable) Target object which must be uniquely
                    specified in the URL so the destination request handler
                    is able to retrieve it.
                params - (Dict) parameters to be added to the URL.
        """
        if ContextManager.get('offlineMode', False):
            url = URL(cls.getStaticURL(target), **params)
        else:
            url = cls._getURL(**params)
            if target is not None:
                url.addParams( target.getLocator() )
        return url

@classmethod
    def getStaticURL(cls, target=None):
        url = cls._relativeURL.replace("py", "html")
        return url

class UHConferenceEmail(URLHandler):
    _relativeURL = "EMail.py"

    @classmethod
    def getStaticURL(self, target):
        if target is not None:
            return "mailto:%s" % str(target.getEmail())
        return self._getURL()

class UHConferenceLogo( URLHandler ):
    _relativeURL = "conferenceDisplay.py/getLogo"

    @classmethod
    def getStaticURL(self, target):
        url = os.path.join(Config.getInstance().getImagesBaseURL(), "logo",
str(target.getLogo()))
        return url
```

3 Conclusion

The core of the Offline website Generation it's done, working for meetings and lectures as well. The offline conference is created but some menu items pages, like: *Author Index*, *Speaker Index* and *List of Registrants* are not completely useful because it needs the improvement to use Javascript on filtering. That will be done by Indico team in the near future.

3.1 What is next?

The next steps to deploy this new module are:

- Provide a way on Indico so the user could retrieve an offline version of his/her event. So far it's done via HTTP request;
- Make sure that protected events can be also generated as offline events;
- Test a variety of different ways to generate the pages;

4 Bibliography

- "Indico" <http://indico.cern.ch/> [accessed 15th August 2012]
- "Indico Software Wiki" <http://indico-software.org> [accessed 15th August 2012]
- "Indico – Developer's Guide", 14th July 2008, <http://indico.cern.ch/getFile.py/access?resId=0&materialId=slides&confId=37937> [accessed 15th August 2012]

5 Summary

This report is part of the Openlab Summer Student that takes place in Indico Project, IT-CIS Group at CERN. During the two months of the project a new Indico module responsible for generating offline pages and provide the user a way to create Offline Events was developed.