Impressory Documentation

Release 0.2-SNAPSHOT

William Billingsley

January 10, 2014

Contents

1 Contents: **3**

1.1 Courses . . . 3
1.2 Enrolling students . . . 4
1.3 Newsfeed . . . 4
1.4 Content viewer . . . 5
1.5 Interaction stream . . . 6
1.6 Tagging . . . 6
1.7 Kinds of Content . . . 7
1.8 Embedding Impressory elsewhere . . . 8
1.9 Setting up your own server . . . 8

2 Indices and tables **13**

Impressory is open source software for hosting interactive, social, connectivist teaching courses. It works in-class and on-line, and is designed to be easy and scalable.

The centre of a course is the newsfeed, where staff and students can post content and questions. Clicking on some content brings it up in the viewer. In this picture, the content being viewed is a video, but it's also part of a "sequence" containing other different kinds of content such as presentations, live polls, wiki pages, and embedded web pages and gadgets. In both pictures, the live *Interaction stream* is open. (You can toggle it open with the orange speech bubble).

A particularly effective way of using Impressory is to put your lectures in as content sequences. Then you can present them, with the interaction stream open, on the lecture screen allowing even large classes to interact with you. Questions on the interaction stream show up in orange to help highlight them. After the lecture, the class can continue the conversation from the newsfeed.

CHAPTER 1

Contents:

1.1 Courses

1.1.1 Creating a course

To create a course, click Create a Course on the front page. The form it brings up is very simple to get you started quickly, but you'll be able to edit many more options after you've initially created it.

1.1.2 Course Membership

Although courses are collaborative, not everyone is the same. If you're the teacher, you'd probably like to have a bit more control over your course than your students do. On the right hand side of this screenshot, you'll see the user has several roles on the course. The important ones are these:

Owner Given to the user who created the course

Administrator Can edit the course settings and invite people to the course

Moderator Can mark content as protected, and can edit protected content

Author Can add content to the course, and edit unprotected content

Reader Can read content in the course. (Note that if the course signup policy is open, you don't need this role to read content)

Chatter Can vote in polls, comment on posts, and chat in the interaction stream. (Note that if the course chat policy is open, you don't need this role to vote and chat)

Statistician Not used yet. Will let users view content analytics

Tutor (Unused)

1.1.3 Settings

Cover image A logo for your course. Typically 320px by 180px. Future versions will support more theming.

List publicly Tick this box if you want your course listed on the front page. Alternatively, leave it unticked if you want only your students to be able to find your course (because you'll give them the link). I find it can be useful to keep a course hidden, and allow anonymous chatting and voting. This means that your students don't even need to log in to participate.

Course signup policy (Not exposed on the form yet)

Course chat policy (Not exposed on the form yet)

1.2 Enrolling students

There are a number of different ways to enrol students and other staff into your course

1.2.1 Invites

You can create invitation codes that users can paste into the "Use invite" box on the course cover page. Invites can be created with a variety of roles, so the code you give your fellow course staff will normally be different than the code you give your students

1.2.2 LTI

(Not exposed yet)

1.3 Newsfeed

Most days, the first place students will visit is the newsfeed. This shows the most recent content that has been published in your course — including polls, videos, news posts, lecture presentations and wiki pages. Students can interact with much of the content directly from the newsfeed.

As you might expect, it looks fairly like a typical social media activity stream.

1.3.1 Filtering the feed

The newsfeed can be filtered, making it easier to find content on a particular topic, to find all the lectures, or content that's been added by staff rather than students.

Content in Impressory is tagged. For instance there may be a *Video* about *Version Control*. And students might want to hop from that to the *Tutorial* about *Version Control*. At the foot of each content entry, the tags describe what the content is about. In the newsfeed, clicking on a tag will filter the feed to show just content with that tag. For instance, click on a Lecture tag, and only the Lectures will be shown. For more information, see *Tagging*.

1.3.2 Comments and voting

Comments and voting are, of course, shown directly under each item on the newsfeed. We include downvoting as well as upvoting, as it's as important to hear what students struggled with — a strongly downvoted item is a good hint that there may be a lot of valuable feedback in the comments.

1.3.3 Coming soon: Q&A

If a content item is marked as being a question, then students can reply to it. Both the question and the replies can be commented on and voted on. This produces a conversation that is very much like the conversations on Q&A sites like Quora or StackOverflow or that you would find in many MOOC platforms. But unlike those platforms, in Impressory this discussion is not hived away in some separate "discussion" area that many students might never visit, but takes

place front-and-center on the newsfeed. Discussion is not a second-class citizen in the course, but a core part of the interaction alongside the content.

It also means you can include video questions (or video replies).

1.3.4 Coming soon: Settings

On the right hand side of each content entry, there are a set of controls. These allow you to edit many of the settings for the content entry directly from the newsfeed. For instance, editing its tags, setting it to "show first" (be the first page returned when a topic is looked up), and whether it is marked to appear in the newsfeed and the index.

1.3.5 Coming soon: Pushing the feed to Facebook

Very often students will set up a Facebook page for a course – a place where they can converse away from the prying eyes of the teaching staff. However, if sharing happens only on the Facebook page, it is less useful for the course. Facebook does not provide learning analytics, or convenient linking between content that might be on the same topic.

Our planned solution for this is to integrate with Facebook, so that when a user adds content, then if they have linked their Facebook account, Impressory will post it as a story onto their wall or a Facebook page. This allows students to converse and commune on Facebook, while still bringing them back to Impressory to interact with the content. That way the course still gets the analytics and feedback, and the students get the convenience of the topic-tagging when it comes to revision.

1.3.6 Coming soon: Pushing the feed to Twitter

If a student links their Facebook account, when an item of content is added, it'll be possible to tweet it over Twitter with a course hashtag.

1.4 Content viewer

The content viewer is typically how you show content in a lecture theatre, and how students watch a live lecture online.

1.4.1 Live interaction on the main screen

In the lecture theatre, you can open the chat stream on the right hand side, so that students can comment and ask questions during the lecture. In Software Studio, we would often get twenty to thirty comments and questions per lecture – far more than the number of students who would be willing to stick their hand up and ask. But that depends on your class size, room shape, and the ambience of the lecture.

1.4.2 Content sequencing and topic-

And of course you can push polls to the chat stream, sequence different pieces of content after each other for the lecture, comment, vote, and easily navigate using the topic tags.

1.4.3 Scalable virtual classrooms

For large remote classes, such as a MOOC, we also think this is a better way of interacting than virtual classrooms. Add the broadcast (for instance using the YouTube URL of a Google Hangout-On-Air) as a content item, and use the polls and chat to allow your class to interact with you. It's easier for students to skim-read text comments coming up from a large class than to wait while each other speaks, or try to skim video snippets.

1.5 Interaction stream

Imperssory courses have a live interaction stream. We'd call it a chat stream, but you can do more than just chat with it. It can be opened whether your looking at the newsfeed, content in the viewer, or on its own if you're using a small mobile device.

This means that you can have a live interaction with your class on the main screen in the lecture theatre, that then continues when your students are back at home or in the library, chatting from the newsfeed.

If your content is a live video broadcast (such as YouTube Live or Google Hangouts), the interaction stream gives you a return channel where many students can chat, ask questions, and respond to polls in real-time, helping your class to be live and interactive even if the class size is in the thousands.

Figure 1.1: The interaction (chat) stream next to the newsfeed, as a student might use it from their laptop at home

Figure 1.2: The interaction stream next to the viewer, as you might show it on the main lecture screen in class

Figure 1.3: The interaction stream on its own in a small browser (such as on a smartphone)

1.5.1 Asking questions

If a chat message starts "Q:", it is highlighted in orange. This is designed for use in large classes, as there can be a lively enough discussion that the questions need to be highlighted so that the teacher can see easily which comments they need to respond to.

1.5.2 Polls

When a poll is open in the viewer, click "push to stream" to push the poll to the interaction stream. This is particularly helpful if you are giving a lecture and have a poll as part of your content sequence – you can push the poll so that students can easily vote on it, and whenever you choose click the "results" tab on the poll to see the live results streaming in.

(Typically you'd leave just the question up for a while, before showing the live results, so that the students don't all just follow the crowd and vote for the most popular option.)

1.6 Tagging

When you add content to Impressory, you are encouraged to give it a topic. This helps, for instance, navigate from the presentation on version control, to the video, to the interactive exercise. The tagging format for impressory is:

Figure 1.4: The results of a live poll being shown on the lecture screen, while the poll has also been pushed to the interaction stream.

adjective noun about *topic*.

So for instance we might have a *Lecture* about *Kirchoff's Current Law*, and we might have an *Exercise* about *Kirchoff's curernt Law*. If you want to use the adjectives then you might have an *Examinable Exercise* about *Kirchoff's Current Law*.

Content can have more than one topic, and it can have more than one noun. Often, the noun will already have been added for you – Impressory knows for instance that YouTube videos are videos.

The reason for this tag format is that down the track we're going to want to do some more interesting things with your course. For instance, build a map of how the topics in your course relate to each other. Or statistics on the usage of the different kinds (nouns) of content (video, lecture, exercises, etc).

1.7 Kinds of Content

There are many different kinds of content entry that Impressory supports: Markdown pages, polls, web pages, YouTube videos, Google Slides, and more being added all the time.

1.7.1 Posts and Wiki pages (Markdown)

Wiki pages in Impressory use Markdown notation. This is a popular and simple notation, because it is both easy-to-read and easy-to-write. This is how this section might look in Markdown:

```
Posts and Wiki pages (Markdown)
-------------------------------

Wiki pages in Impressory use [Markdown][MD] notation.
This is a popular and simple notation, because it is both easy-to-read and easy-to-write.
This is how this section might look in Markdown

[MD]: http://daringfireball.net/projects/markdown/syntax
```

To include a link to another piece of content, just use its ID as the link:

```
Here is my text with a link to [another content item](52c3a0eee1000096077b06e0)
```

1.7.2 Video

Video is supported through YouTube. Just paste the YouTube URL or embed code into the "share something" box on the newsfeed. Video renders both in the viewer and in the newsfeed.

This also means you can use Impressory to add interaction around a live broadcast. Embed the YouTube URL of a Google Hangout to broadcast the video, and use the chat stream (including [polls](polls.html) and all the other goodies) to interact with your online audience.

1.7.3 Embed the Web

Web pages (including all those online interactive exercises that are out there) can be added as content.

1.7.4 Presentations

Presentations are supported through Google Slides. Shortly we'll also add support for presentations written locally, using Reveal.js.

1.7.5 Content sequences

During a lecture, you might want to progress seemlessly from one piece of content to another. For instance, if you have a couple of presentations you want to queue after each other. Content sequences allow you to do this.

1.8 Embedding Impressory elsewhere

1.8.1 Embedding content items

Content items from within Impressory can be embedded onto other websites, such as blogs. For instance you may wish to post one of the live polls to a blog outside the course, allowing the public to vote on it and your students to see the results.

1.8.2 Embedding interaction

(Coming soon)

1.9 Setting up your own server

Contents:

1.9.1 Using the distribution zip

If you want to run your own server, you can download a distribution zip that has been pre-built

To run the server, you'll need two pieces of software:

- MongoDB, the database that Impressory uses. (This can be either local or remote)
- A Java Virtual Machine. Impressory is written in Scala, which runs on the JVM.

To use the distribution zip

1. Download the zip
2. Unzip it wherever you'd like to run it from:

   ```
   $ unzip impressory-0.2-SNAPSHOT.zip
   ```

4. To run the server, you now just run the start-up script. The instructions below are for Linux (using nohup so that when you close the terminal the program doesn't also close):

   ```
   $ nohup impressory-0.2-SNAPSHOT/bin/impressory > log.out 2> err.out &
   ```

The server should now be up and running, listening on port 9000. Open a web browser and visit it. The first user to register is given administrative rights, so you should make sure that's you!

Although that's compiled and run the server, there are probably a few things you'll still want to configure:

- Database environment variables
- Social media logins
- HTTPS
- Port redirection

1.9.2 Building Impressory from source

Impressory is an open source project. This means that one way to run the server is to compile and run the source code. These instructions assume you intend to run the server on Linux or Mac. (We'll develop Windows instructions in due course.)

First, we need a few pieces of software:

- MongoDB, the database that Impressory uses.
- A Java Virtual Machine. Impressory is written in Scala, which runs on the JVM.
- sbt, the Scala Build Tool. (It's not very big.)
- git, to get the source code so we can build it.

Then we can build the software

1. Clone the sourcecode repository from GitHub:

   ```
   $ git clone https://github.com/impressory/impressory.git
   ```

2. Ask sbt to build the distribution zip file. This builds a zip file containing everything you need to run the server:

   ```
   $ sbt dist
   ```

   This builds the distribution zip in *target/universal/impressory-0.2-SNAPSHOT.zip* You can copy this file wherever you'd like to run the server from.

3. Unzip the distribution file it made:

   ```
   $ unzip impressory-0.2-SNAPSHOT.zip
   ```

4. To run the server, you now just run the start-up script. The instructions below are for Linux (using nohup so that when you close the terminal the program doesn't also close):

   ```
   $ nohup impressory-0.2-SNAPSHOT/bin/impressory > log.out 2> err.out &
   ```

The server should now be up and running, listening on port 9000. Open a web browser and visit it. The first user to register is given administrative rights, so you should make sure that's you!

Although that's compiled and run the server, there are probably a few things you'll still want to configure:

- Database environment variables
- Social media logins
- HTTPS
- Port redirection

1.9.3 Configuration

An Impressory server can be configured either by setting environment variables, or by passing arguments to the start-up script

Database

The following environment variables configure the MongoDB connection:

```
$IMPRESSORY_MONGO_URL
$IMPRESSORY_MONGO_DBNAME
$IMPRESSORY_MONGO_DBUSER
$IMPRESSORY_MONGO_DBPWD
```

These can also be set by passing them as arguments to the start script:

```
-Dmongo.connection=
-Dmongo.dbname=
-Dmongo.dbuser=
-Dmongo.dbpwd=
```

Social media logins

Students can log in using GitHub or Twitter accounts.

For this to work, you'll need to set an OAuth client key and secret that is issued to you by GitHub or Twitter.

GitHub:

1. Register an application on GitHub

   The authorisation callback URL is {your server URL} + /oauth/github/callback

2. Set the Client ID and Client Secret before starting Assessory. This can be done by setting these environment variables:

   ```
   $IMPRESSORY_AUTH_GITHUB_CKEY
   $IMPRESSORY_AUTH_GITHUB_CSECRET
   ```

   or by passing them as arguments to the start script:

   ```
   -Dauth.github.ckey=
   -Dauth.github.csecret=
   ```

Twitter:

1. Register an application on Twitter

   The authorisation callback URL is {your server URL} + /oauth/twitter/callback

2. Set the Client ID and Client Secret before starting Assessory. This can be done by setting these environment variables:

   ```
   $IMPRESSORY_AUTH_TWITTER_CKEY
   $IMPRESSORY_AUTH_TWITTER_CSECRET
   ```

   or by passing them as arguments to the start script:

   ```
   -Dauth.twitter.ckey=
   -Dauth.twitter.csecret=
   ```

HTTPS

To make Impressory listen using HTTPS, you'll need to pass a few more parameters to the start script, as described here

CHAPTER 2

Indices and tables

- *genindex*
- *modindex*
- *search*