Meeting summary 2017-04-20

Project course in computer science, 1DV508

**Deadline for submission to MyMoodle:** Thursday 2017-04-27 12:00

**Next seminar:** Friday 2017-04-28 10:00 in B2034

**Next meeting:** Monday 2017-04-24 (time and location yet to be decided)

# Topics

## Implementation Details

* We decided to implement the following use cases in this iteration: Create a timeline (TUC-1), Save timeline (TUC-2) and Load timeline (TUC-3)
* We need to implement the IO component to be able to implement the save and load timeline use cases. We decided that it would be enough with one class, FileHandler, which has methods to read (save) a timeline to a XML file and write (load) a timeline to a XML file.
* We decided that the communication between the MenuController, the FileHandler and TimelineContainer should work like this:

**Saving a timeline**

* 1. The MenuController gets notified by MenuView that the save button has been pressed. MenuView sends the file the user selected with the file chooser as a parameter.
  2. The MenuController fetches the selected timeline from TimelineContainer.
  3. The MenuController asks the FileHandler to save the Timeline fetched from TimelineContainer to the file.

**Loading a timeline**

1. The MenuController gets notified by MenuView that the load button has been pressed. MenuView sends the file the user selected with the file chooser as a parameter.
2. The MenuController asks the FileHandler to load the contents of the file and gets a Timeline as a response.
3. The MenuController asks the TimelineContainer to add the loaded Timeline to its list of timelines and set the loaded Timeline as the selected one.

* We decided to use the Java Architecture for XML Binding (JAXB) library to implement reading and writing of timelines to XML.
* We decided that the goal of this implementation was to be able to create a new timeline and have its name appear in the drop-down menu where the user selects which timeline to work with. The name should also appear when a timeline is loaded.
* We need to add a save and load button to the menu view so we can test saving/loading.
* To implement saving and loading of timelines, we need to also create the classes for Timeline and Event. For now, the Timeline class only needs a list of Events, a variable for its name as well as a getter and a setter for the name. The Event does not need to have any variables or methods for this iteration.

## Product Specifications

* We talked about having a popup window appear to warn the user if it tries to close the program without having saved all opened timelines.
* We decided that when the user presses the save button, it should only save the currently selected timeline.
* It was suggested that the unsaved timelines could have a symbol next to their names to indicate to the user that they have unsaved changes (Similar to how Eclipse displays an \* next to the name of tabs when it has unsaved changes).

## Team Organization

* Since Tomas has decided to drop the course, we need to reassign the teams. This will be done on the next meeting on Monday. The time for this meeting has not been decided yet.
* The meeting on Monday will also be a progress checkup meeting.

## Documentation

* An implementation plan needs to be created that shows what tasks have been assigned to whom and which test are related to that task.
* The communication between MenuController, FileHandler and TimelineContainer when saving/loading timelines should be shown in a class diagram.
* The sequence of events when adding and saving/loading a timeline should be shown with pseudocode.
* Test cases for adding and saving/loading a timeline should be written.