DESIGN DOCUMENT FOR COURSE PROJECT “SISU”

COMP.CS.140 Programming 3: Interfaces and Techniques

Project Work Group
Gaurab Mahat, Paula Mosallaei, Peter Mwesigwa

Purpose of the Document

This document describes how the Sisu program (created by ‘Project Work Group’ team), is designed. It outlines the project functionality, contains a short user manual and the structure of the software, and describes the responsibilities of the key classes with their pre and post conditions. In addition, known bugs and missing features have been pointed out. Finally, the document presents the division of work.

Project Functionality

Sisu program is designed for student’s personal use to track the progress of their studies. Student can choose the Degree Program and Orientation (further called Option or Degree Option) they are completing and proceed to mark selected and completed courses. Student’s choices are then saved in a JSON file named with the student number. Student logs in to the application providing their student number, and the program checks whether a corresponding file exists. If it does, the program opens it and loads all data saved by the student in previous sessions, which can be further modified. If no corresponding file exists, all Degree Programs and data related to them are loaded from SisuAPI. A file with student’s data will be created once the student decides to save their changes.

GUI and short user manual

Graphical User Interface of Sisu program was fully implemented by the team in JavaFX. It launches with a dialog window asking user to log in with their student number consisting of either letters or digits. When the student number is entered and submitted with the button “Login”, the program checks in the background whether a file named with the provided number exists. Then there are two possible scenarios, which however look nearly the same for the user.

Scenario 1: If there is no file, a window is opened prompting the student to select their Degree Program and Degree Option. The lists are populated in the background with data retrieved from SisuAPI. In tab “Courses” of this window, the list of Degree Option’s constituent Modules and Courses is displayed. Here, student can select courses to be saved by first clicking the name of the Course from the list on the left and then clicking the button “Add Course”. Saved courses will appear in the list on the right. Student can also mark the selected courses as completed by first ticking them in the list on the right and then clicking the “Complete Course” button. All courses from a given Module or even from the entire Degree can be added by clicking on the Module name and clicking “Add Course” button. A Course can be deleted by ticking the Course in the list on the right and clicking “Remove Course” button. Clicking “Complete Course” will save the Course with a tick as completed and will also update the number of completed credits. Completed Courses cannot be removed. Removing and completing happens one Course at a time. When all wanted changes have been introduced, the window can be closed with “X” button. This will show a pop-up window, where student can choose if they want to save the changes or not. Saving will create a JSON file with student’s data and will make the student recognizable for the application. Not saving will discard all selections and changes and student will not be recognized the next time the program is launched.

Scenario 2: The file has been found by the program. It is read and the data is loaded to the program. A window with the dropdown lists is opened, however now there is only the Degree and Option saved previously by the student. Tab “Courses” looks the same as in Scenario 1 – left side shows all possible courses and right side shows student’s selected courses and completed courses. Completed Courses also have a small icon next to their name in the list on the left side. Completed Courses cannot be removed from the list on the right. More Courses can be selected and completed. The changes can be saved – the existing file will be overwritten. Saving happens the same way as in Scenario 1.

Structure of the Software

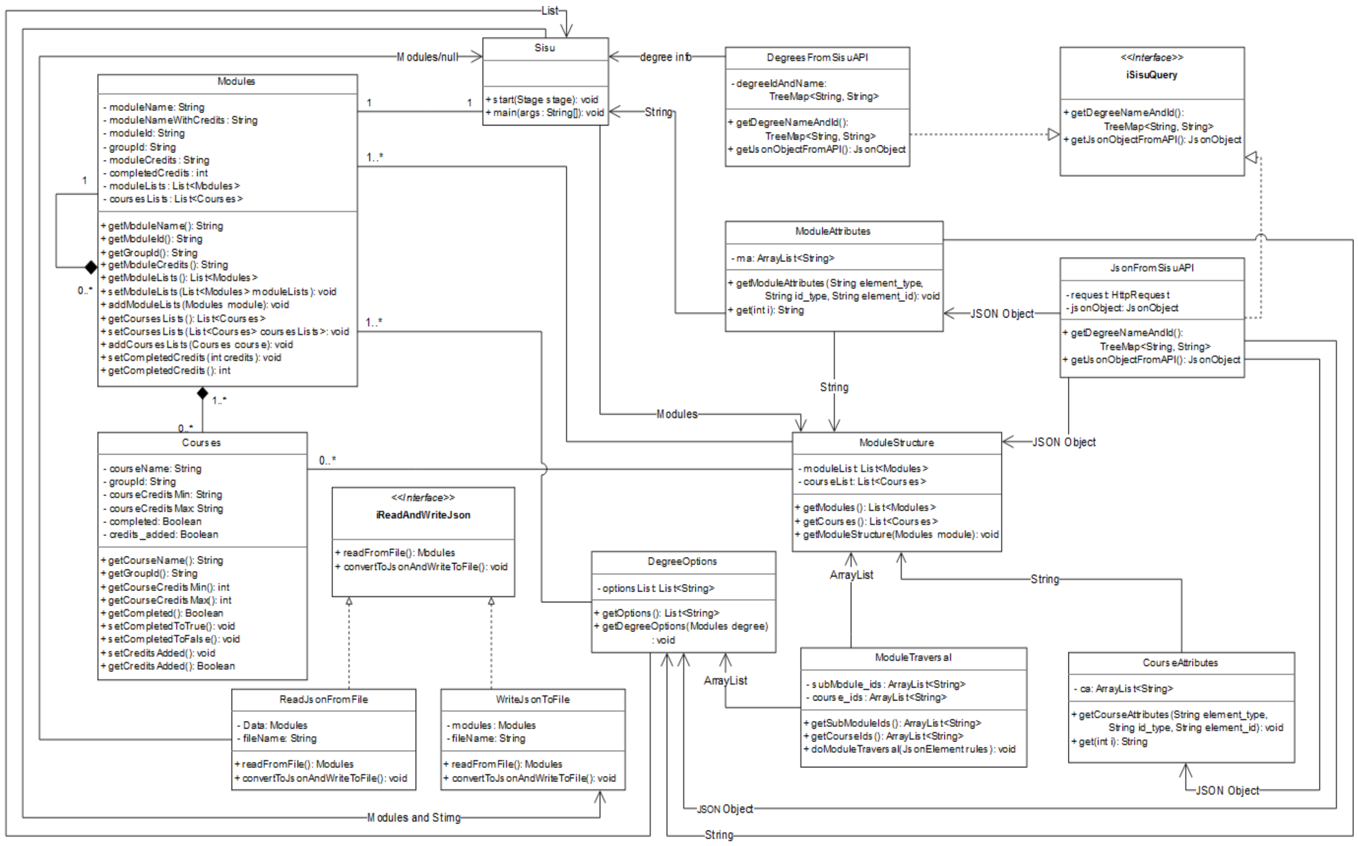
Sisu
+ start(Stage stage): void
+ main(args : String[]): void
Modules
- moduleName: String
- moduleNameWithCredits : String
- moduleId: String
- groupId: String
- moduleCredits : String
- completedCredits : int
- moduleLists : List<Modules>
- coursesLists : List<Courses>
+ getModuleName(): String
+ getModuleId(): String
+ getGroupId(): String
+ getModuleCredits (): String
+ getModuleLists (): List<Modules>
+ setModuleLists (List<Modules> moduleLists): void
+ addModuleLists (Modules module): void
+ getCoursesLists (): List<Courses>
+ setCoursesLists (List<Courses> coursesLists>: void
+ addCoursesLists (Courses course): void
+ setCompletedCredits (int credits): void
+ getCompletedCredits (): int
Courses
- courseName: String
- groupId: String
- courseCreditsMin: String
- courseCreditsMax: String
- completed: Boolean
- credits_added: Boolean
+ getCourseName(): String
+ getGroupId(): String
+ getCourseCreditsMin(): int
+ getCourseCreditsMax(): int
+ getCompleted(): Boolean
+ setCompletedToTrue(): void
+ setCompletedToFalse(): void
+ setCreditsAdded(): void
+ getCreditsAdded(): Boolean
DegreesFromSisuAPI
- degreeIdAndName: TreeMap<String, String>
+ getDegreeNameAndId(): TreeMap<String, String>
+ getJsonObjectFromAPI(): JsonObject
<<Interface>>
iSisuQuery
+ getDegreeNameAndId(): TreeMap<String, String>
+ getJsonObjectFromAPI(): JsonObject
ModuleAttributes
- ma: ArrayList<String>
+ getModuleAttributes (String element_type, String id_type, String element_id): void
+ get(int i): String
JsonFromSisuAPI
- request: HttpRequest
- jsonObject: JsonObject
+ getDegreeNameAndId(): TreeMap<String, String>
+ getJsonObjectFromAPI(): JsonObject
ModuleStructure
- moduleList: List<Modules>
- courseList: List<Courses>
+ getModules (): List<Modules>
+ getCourses (): List<Courses>
+ getModuleStructure(Modules module): void
<<Interface>>
iReadAndWriteJson
+ readFromFile(): Modules
+ convertToJsonAndWriteToFile(): void
DegreeOptions
- optionsList: List<String>
+ getOptions (): List<String>
+ getDegreeOptions(Modules degree): void
ModuleTraversal
- subModule_ids: ArrayList<String>
- course_ids: ArrayList<String>
+ getSubModuleIds (): ArrayList<String>
+ getCourseIds (): ArrayList<String>
+ doModuleTraversal(JsonElement rules): void
CourseAttributes
- ca: ArrayList<String>
+ getCourseAttributes (String element_type, String id_type, String element_id): void
+ get(int i): String
ReadJsonFromFile
- Data: Modules
- fileName: String
+ readFromFile(): Modules
+ convertToJsonAndWriteToFile(): void
WriteJsonToFile
- modules: Modules
- fileName: String
+ readFromFile(): Modules
+ convertToJsonAndWriteToFile(): void
List
Modules/null
degree info
String
Modules
JSON Object
ArrayList
Modules and String
1
1..*
0..*
1..*

Figure 1: Class Diagram

Key Classes and their Responsibilities

The central classes of the program are Modules and Courses.

Modules correspond to StudyModule in SisuAPI, but Degree Program, the root level of the Degree structure, is handled as Modules instance as well. To create an instance of this class the following attributes are needed: the name of the Module, both its ID and GroupID, and credits. Each Modules instance stores a list of its direct children Modules and Courses. All class attributes can be retrieved from the Modules instance using corresponding methods.

Attributes of Modules needed to create an instance are retrieved directly from SisuAPI using class **ModuleAttributes**. The IDs of Modules' children, which are needed to obtain data from SisuAPI, are found by class **ModuleTraversal**. Class **ModuleStructure** uses both classes to retrieve the entire structure of a Degree and saves it in the main Modules instance, named Degree. However, Module Structure is only used after the user selects the Degree Option, so that only one structure is traversed and saved. Degree Options are retrieved with the help of class **DegreeOptions**, which takes Degree as its parameter. A separate class for obtaining Options is needed, because the tree structures of Degree Programs are not consistent in SisuAPI.

Courses correspond to CourseUnits in SisuAPI. To create an instance of Courses, the following attributes are needed: the course name, course min and max credits, and course GroupID. Similarly to ModuleAttributes class, **CourseAttributes** class retrieves from SisuAPI all attributes needed to create a Courses instance.

Sisu is the main class of the program. It generates all windows of GUI, handles their functionality, and stores global variables, particularly the Modules instance Degree. Degree, being an instance of Modules class, contains the structure of a chosen Degree Program, and in case of recognized user, Degree structure saved in previous sessions.

Finally, there are two packages created for the needs of the program: SisuQuery and ConvertJson. **SisuQuery** comprises of classes that are responsible for fetching JsonElements from SisuAPI, which are later used by other classes to populate Degree structure. These classes need the following parameters to be able to form a query: type of item data is fetched for (Modules or Courses), type of iD supplied (ID or GroupID) and the ID itself.

ConvertJson package contains classes responsible for saving student data into JSON files and for reading JSON files with student data into the program. The classes need Degree object (an instance of Modules classes) and/or student number to save/read a file.

Implemented Functionalities, Known Bugs and Missing Features

All basic and nearly all the intermediate features required in the project specification have been implemented, and they are listed below. From intermediate features the only one missing is that not all classes have test files. The program has additional features, which are marked as "ADDITIONAL".

- The initial dialog window
- GUI implemented fully by the team in JavaFX
- Structures of degrees retrieved from SisuAPI
- Selected study program displayed in the main window
- Users can select courses, mark them as completed and save them
- ADDITIONAL: Users cannot login without typing in their student number, if the user tries to login with an empty string, a message is shown to the user
- ADDITIONAL: Pop-up window asking the user if changes should be saved
- ADDITIONAL: Completed credits are shown/updated in real-time
- ADDITIONAL: An icon shows up next to a completed course
- ADDITIONAL: Users cannot save their data without selecting both the Degree program and Orientation, otherwise an alert message is shown to the user

- Unit tests for some classes have been implemented
- Users are recognized by the program
- User data is imported by the program when the user is recognized
- Documentation: JavaDocs, pdf file

Known bugs:

- java.lang.IndexOutOfBoundsException: caused by ObservableArrayList program_modules when a different Degree Program is selected after the Option has been already selected in the second dropdown list; most probably it is thrown because the Observable list is being modified while being in use by EventHandler; the error does not break the program and after it pops up the functionality does not change.
- The program lags after the Degree Option have been selected, especially when the option contains a lot of submodules, but it does not crash.
- When a Course is marked as completed, and it is in the bottom of the tree structure, it is possible that the Credits of its higher-level parent Modules are not updated. Usually, the direct parent Module’s credits are updated.
- In Scenario 1, if some Courses have been selected and displayed in the right panel, and then user comes back to the Degree selection tab and will choose another Degree or Option, the program will not work anymore.

Division of work

The team members met and communicated on a regular basis and responsibilities were assigned on the go, according to the general progress of work and members’ individual progress. Usually, newly assigned tasks were related to the previous tasks completed by a member. The table below roughly outlines the tasks completed by each group member. During the meeting all members worked together to fix bugs, improve the code and fix merging conflicts. The final cleaning of the code was also done together.

Team member	Tasks
Gaurab	classes in package fi.tuni.prog3.sisu.ConvertJson (reading and writing student data from/to JSON file
	classes in package fi.tuni.prog3.sisu.SisuQuery (fetching JsonObjects and JsonArrays from SisuAPI)
	partly classes Courses and Modules
	implementing the dialog window (Scene One)
	implementing the window for recognized user (Scene Three)
	pop-up window showing up when the program is closed (all scenes)
	class diagram
	JavaDoc comments for some of the classes
	test files for classes Modules, Courses, and JsonFromSisuAPI
Paula	classes in package fi.tuni.prog3.sisu (except Sisu.java and classes Modules and Courses only partly; responsible for finding relevant data in JsonObjects and JsonArrays and populating Degree object with the degree structure)
	populating the second drop-down list in the main window (Scene Two)
	populating left panel with selected Degree Option structure (Scene Two)

	partly populating right panel with selected courses (Scene Two)
	instructions box (Scene Two)
	JavaDoc comments for some of the classes
	documentation
Peter	implementing the main window of the GUI (Scene Two), which gave start to all parts of GUI and all functionalities
	populating right panel with selected courses (Scene Two)
	functionalities of add, remove, and complete buttons (Scene Two, but used in Scene Three as well)
	updating credits of completed courses in the Courses and Modules and in the left panel
	partly Modules and Courses classes
	populating the first drop-down list in the main window (Scene Two)