SISU-LITE Project Documentation

COMP.CS.140

Roope Mantere
Ali Hussain
Vertti Nuotio

Contents

Features **1**

User Guide **2**

Login Window 2

Registration Window 2

Study Information Window 3

Course Detail Window 4

Known Bugs and Missing Features **5**

Division of Work **5**

Program Flow Class Diagram **5**

1. Features

This section lists all features that are either visible to the user or have a major impact on the flow of the program. It should be noted that the additional features listed here may not each correspond to an additional feature listed on Plussa. Some might be larger and some smaller in scope. Clear additional features from Plussa include course view, student settings, information entities from Sisu API and our own feature. The list of features is as follows:

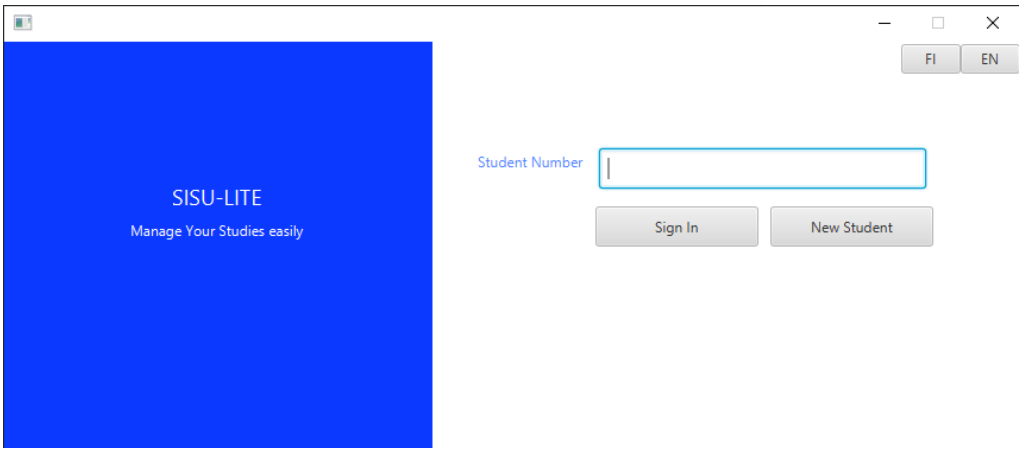
1) Language can be changed in all windows of the program. The options are Finnish (default) and English
2) User data is saved in a JSON and will be loaded when logging in if they have an existing account. User data includes name, student number, expected graduation date, completed courses and chosen degree programme
3) The user can log in with an existing student number, or register a new user account
4) When registering, the user can input their name, student number, chosen degree programme and a self-assessed graduation date estimate. The student number must not be in use by another account
5) The information for degree programmes, study units and courses is fetched from the Sisu API through an HTTP request. After the information has been fetched, it is saved into a temporary JSON file to speed up additional information requests during runtime
6) Study unit and degree programme information includes its name in all given languages and minimum required credits. Course information includes name, minimum credits, content, learning outcomes and a short description in all given languages. If searching for Finnish names and none can be found, the program returns the English version or a backup message if that can not be found either.
7) The degree programme is shown in a tree form in the main window
8) The user can preview other degree programmes within the main window
9) The user can switch their degree programme to another and reset it
10) Degrees and study units show the credits accumulated from completed courses directly and indirectly under them. Courses have their credit worth next to their name
11) Courses have tooltips attached which show a short description if given in Sisu when hovered over
12) Courses can be clicked to open a subwindow that shows more information about the course. If not yet completed, the user can mark the course as completed from a button
13) The user can save their progress whenever they want, or save and exit.

2. User Guide

This section offers a brief overview on how to use SISU-LITE software from a student user's perspective. The section contains images to aid with grasping the interface.

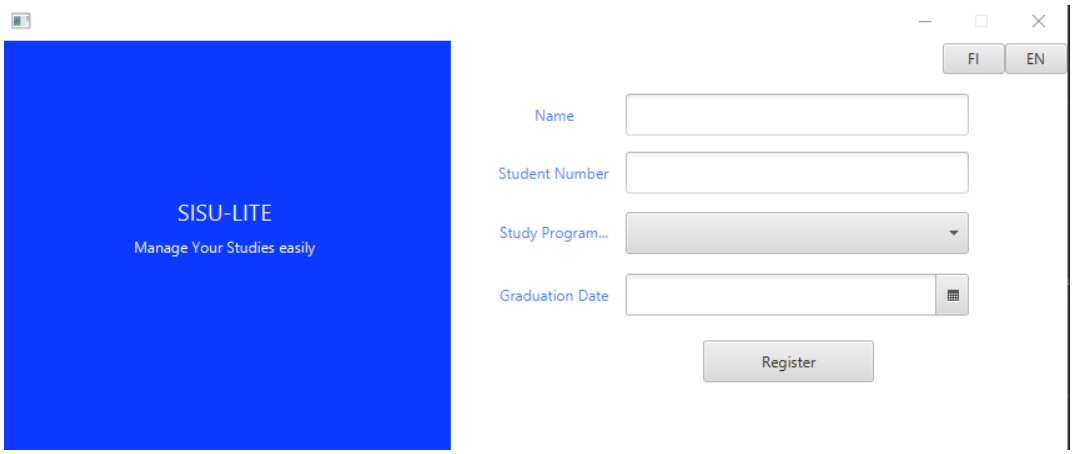
2.1. Login Window

When the user launches the software they are presented with a login screen where they may choose to login with their Student ID or register as a new student. Language can be changed from the top right corner, with the options being Finnish (FI, default) and English (EN).



2.2. Registration Window

If the user decides to register as a new student, they are presented with the registration page. The language can again be changed from the top right corner. Here the user can input their name, student number, and choose their study programme and expected graduation date. All fields must be filled and the student number must be unique before the user can proceed with registering. After this the user can register by pressing the register button, which opens the main view of the program.

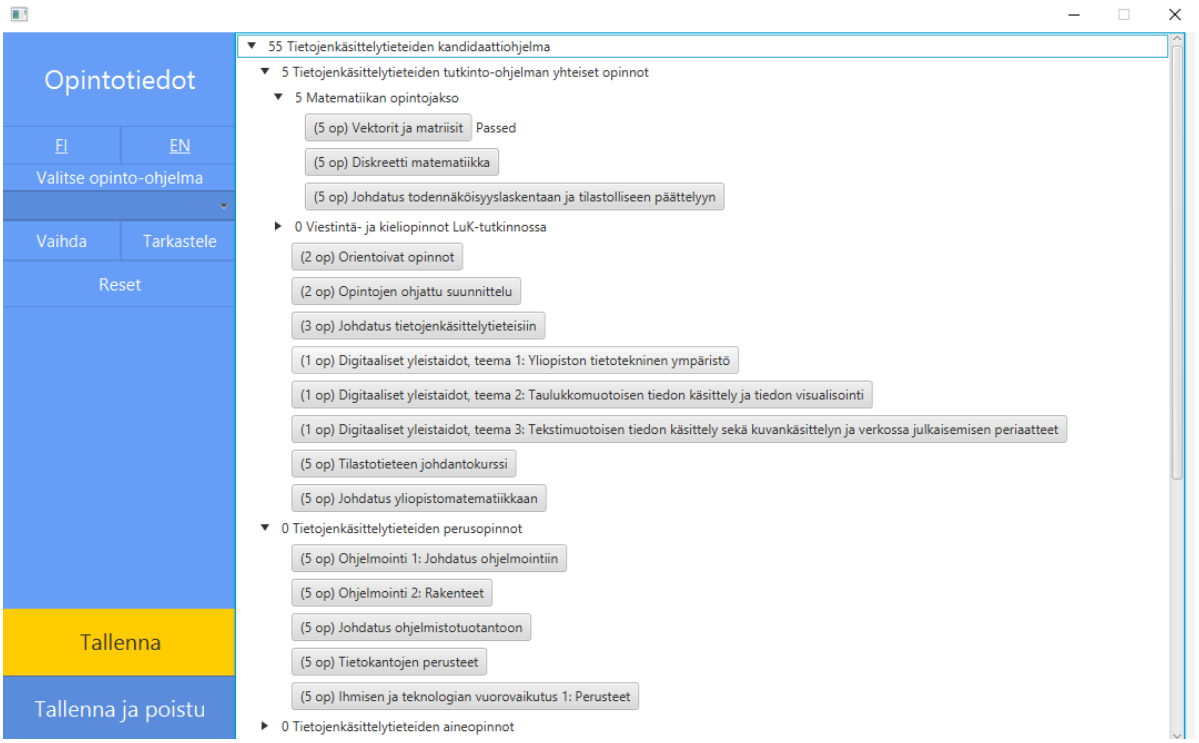


2.3. Study Information Window

After the login the user is presented with the main window where they can manage their passed courses and preview other study programmes. On the left panel the student may preview other study programmes and change their own study programme. Note that the change is permanent only after restarting the application, and before that the user can reset their degree programme to the original one through the “Reset” button. Language can also be changed from the “FI” and “EN” buttons on the left.

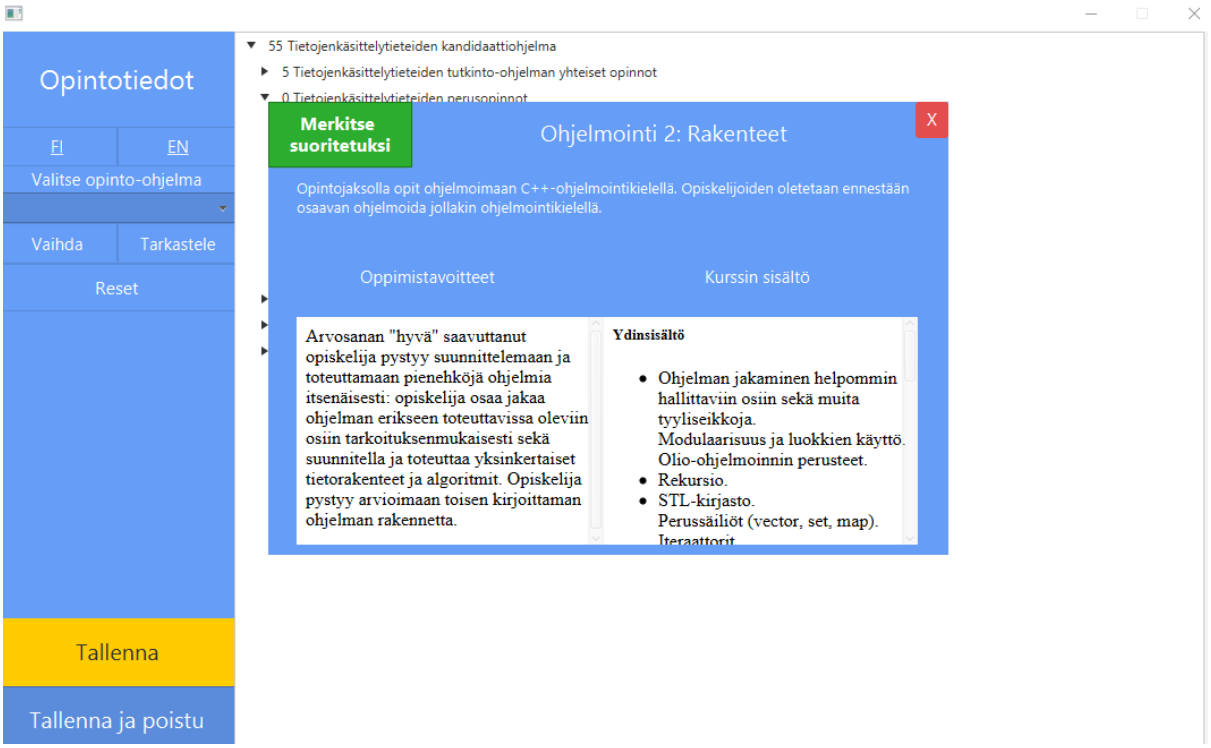
The “Save” button saves the current progress and degree programme to file and lets the user continue. The “Save and Exit” button saves the progress to disk and closes the entire program.

The study tree can be expanded by either clicking the arrow left of the item or double clicking it. Study units show the accumulated credit count below them. Courses can be clicked to open a more detailed window on top of the current window showcasing that course.



2.4. Course Detail Window

The course window contains more detailed information about the course. Details include name, completion status, short description, learning outcomes and contents. All details are in the language selected in the main (study information) window. The completion status is visible in the text on the button, which either reads “Completed” and is grayed out or reads “Mark as Completed” and is clickable. Clicking the button in the latter state marks the course as completed both in this window and the main window, but does not yet save the course in the student’s file.



3. Known Bugs and Missing Features

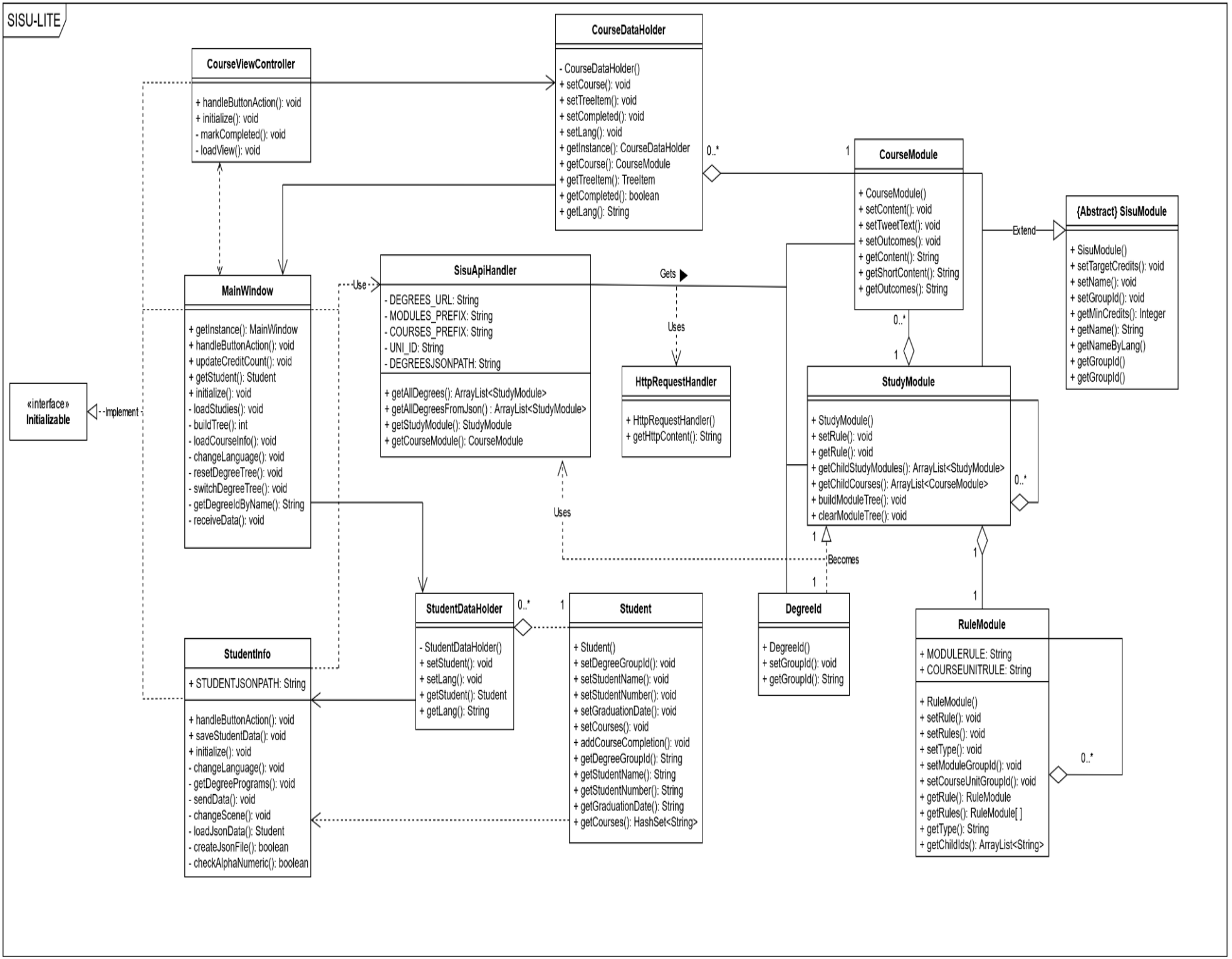
Worst case scenario exception handling is also not complete, but in extensive tests the program has been shown to never reach these worst cases. Tampering with the student JSON files can cause the program to crash, but this is not behavior a standard user can be expected to cause.

4. Division of Work

The project was organized through Trello and was to use agile methods with weekly meetings. The team stuck to this schedule in the beginning, with work getting done. The arrival of Wappu slowed down progress considerably , and work only picked up again after it, in the second week of May. The team mostly stuck to the agreed work division, with Ali creating the unit tests, student data readers/writers and refactoring code as a senior developer, Roope developing UI related parts (FXML + Controllers), and Vertti developing Sisu API related handlers and models.

5. Program Flow Class Diagram

The image on the next page depicts the flow of the program as a class diagram.The diagram follows standard UML notation and contains all classes and interactions deemed important for the flow of the program. The image is also available in the Documentation folder.


SISU-LITE
CourseDataHolder
- CourseDataHolder()
+ setCourse(): void
+ setTreeItem(): void
+ setCompleted(): void
+ setLang(): void
+ getInstance(): CourseDataHolder
+ getCourse(): CourseModule
+ getTreeItem(): TreeItem
+ getCompleted(): boolean
+ getLang(): String
CourseViewController
+ handleButtonAction(): void
+ initialize(): void
- markCompleted(): void
- loadView(): void
0..*
1
CourseModule
+ CourseModule()
+ setContent(): void
+ setTweetText(): void
+ setOutcomes(): void
+ getContent(): String
+ getShortContent(): String
+ getOutcomes(): String
Extend
{Abstract} SisuModule
+ SisuModule()
+ setTargetCredits(): void
+ setName(): void
+ setGroupId(): void
+ getMinCredits(): Integer
+ getName(): String
+ getNameByLang()
+ getGroupId()
+ getGroupId()
SisuApiHandler
- DEGREES_URL: String
- MODULES_PREFIX: String
- COURSES_PREFIX: String
- UNI_ID: String
- DEGREESJSONPATH: String
+ getAllDegrees(): ArrayList<StudyModule>
+ getAllDegreesFromJson() : ArrayList<StudyModule>
+ getStudyModule(): StudyModule
+ getCourseModule(): CourseModule
Gets
Use
Uses
HttpRequestHandler
+ HttpRequestHandler()
+ getHttpContent(): String
MainWindow
+ getInstance(): MainWindow
+ handleButtonAction(): void
+ updateCreditCount(): void
+ getStudent(): Student
+ initialize(): void
- loadStudies(): void
- buildTree(): int
- loadCourseInfo(): void
- changeLanguage(): void
- resetDegreeTree(): void
- switchDegreeTree(): void
- getDegreeIdByName(): String
- receiveData(): void
«interface»
Initializable
Implement
StudyModule
+ StudyModule()
+ setRule(): void
+ getRule(): void
+ getChildStudyModules(): ArrayList<StudyModule>
+ getChildCourses(): ArrayList<CourseModule>
+ buildModuleTree(): void
+ clearModuleTree(): void
0..*
1
Becomes
StudentDataHolder
- StudentDataHolder()
+ setStudent(): void
+ setLang(): void
+ getStudent(): Student
+ getLang(): String
0..*
1
Student
+ Student()
+ setDegreeGroupId(): void
+ setStudentName(): void
+ setStudentNumber(): void
+ setGraduationDate(): void
+ setCourses(): void
+ addCourseCompletion(): void
+ getDegreeGroupId(): String
+ getStudentName(): String
+ getStudentNumber(): String
+ getGraduationDate(): String
+ getCourses(): HashSet<String>
DegreeId
+ DegreeId()
+ setGroupId(): void
+ getGroupId(): String
RuleModule
+ MODULERULE: String
+ COURSEUNITRULE: String
+ RuleModule()
+ setRule(): void
+ setRules(): void
+ setType(): void
+ setModuleGroupId(): void
+ setCourseUnitGroupId(): void
+ getRule(): RuleModule
+ getRules(): RuleModule[ ]
+ getType(): String
+ getChildIds(): ArrayList<String>
StudentInfo
+ STUDENTJSONPATH: String
+ handleButtonAction(): void
+ saveStudentData(): void
+ initialize(): void
- changeLanguage(): void
- getDegreePrograms(): void
- sendData(): void
- changeScene(): void
- loadJsonData(): Student
- createJsonFile(): boolean
- checkAlphaNumeric(): boolean