

**Developers Document**

**for**

**Library information appication**

**Version: 2.0**

**Date: 18/06/08**

**20143692 강경석**

**20144103 조유성**

**20145026 김상범**

**20141043 양준호**

**20146110 문태현**

**20143583 김기환**

[[Library information appliciation]](#_Toc480998875)

[Version 2.0](#_Toc480998877)

[June 10, 2018](#_Toc480998878)

[1. documentation for developers](#_Toc480998881)

[1.1 how to obtain the source code](#_Toc480998882)

[1.2 the layout of directory structure](#_Toc480998883)

[1.3 how to build the software](#_Toc480998884)

[1.4 how to test the software](#_Toc480998885)

[1.5 how to set up an automated daily build and test](#_Toc480998885)

[1.6 how to release a new version of software](#_Toc480998885)

[1.7 how to access the list of outstanding bugs and the list of resolved bugs, and how to resolve a bug](#_Toc480998885)

[1.8 source code example](#_Toc480998885)

# Documentation for developers

## How to obtain the (latest) source code

The developers will update and download source codes in the public storage which the project manager made by using web-hosting service “*GitHub*”.

This public storage will be used for the developers develop functions independently and integrate the functions together when complete.

As well as the public storages, which are a collection of the latest source codes, each developer may create a personal repository and access the repository of another worker even if the source code has problems in his or her repository.

“*Github*” also allows us to see the history of source code changes, compare to the previous version, move to the specific point of time, and alert us when uploading to a repository if the file you are trying to update conflicts with edits made by other developers. Using these features, we can handle problems made by lots of developers editing the source codes at the same time and manage the most up-to-date codes.

## The layout of your directory structure

The public storage of “*GitHub*” standard

|  |  |
| --- | --- |
| **A top-level directory layout** | **Purpose** |
| build | The compile files are stored inside the build folder |
| docs | Some reference data in to the project stored into the docs folder. |
| src | The actual source files of a software project are stored inside the src folder. |
| test | Automated tests are placed into the test folder. |
| tools | Supporting tools for this project are stored into the tools folder. |

|  |  |
| --- | --- |
| **Test Directory** | **Purpose** |
| test/benchmarks | Load and stress tests |
| test/integration | Integration tests |
| test/unit | Unit tests |
| test/whole | Whole tests |

## How to build the software

1) Language to use

we will use object-oriented language java to implement the functions of this program divided into class units.

2) Program development tool

To use the java language efficiently we will use eclipse as a program development tool.

Eclipse has a huge plug-in. In the eclipse Plug-in, we will use ‘junit’ to test our program. This test is mentioned in detail below “build and test”.

We use javafx for GUI. Using eclipse can be a great help in linking our program with javafx

And eclipse provede us many libraries, which will benefit program development

3) Functions

The functions of this program will be largely divided into user mode and administrator mode.

It is mentioned in SDS file and this File.

4) GUI

In order to improve user’s convenience and environment, our team will implement GUI using javajx.

There is an example of a GUI that we will implement in “how to use software”.

All the GUI necessary for the program as well as the example above will be implemented using javafx

5) Database

(1) Tool of database

Our team will create a local database with the text.txt file. And we will connect it with program.

When we run and test program, we will prove that our program can effectively manage the database. This means that even if we update the program and link the web database, the program can work efficiently.

(2) Components of database

\* User information database : The text file that stores the user database contains the user ID, password, name, birthday, phonenumber, rent\_booklist, rent\_time, rent\_fee

\* Book information database : The text file that stores the book database contains book status(whether it is rental or not), book name, writer, publisher, code(serial number) and book state.

## how to test the software

1. In unit testing, our team developers can test individual functions in individual modules.

Our library system has 6 modules(Library, Login, Administrator, User, User\_Data, Book\_Data). So we divided 6 stages to unit test.

In library module, there are seven functions (Library(), main(String[]), inputLogin(), signUp(), Login(), adminLogin(), userLogin()).

First, we isolate all functions in each module. The reason is that our team developers should know whether each function in module work correctly or not.

Then, if error exists in functions in module, we can correct functions.

Through isolating functions in each module, it seems to increase our developing time and cost. However, by reducing the debugging time which takes up most of the development period, it enables the programming to be relaxed.

Also we can reshape code easily without changing the result (refactoring), because after the refactoring, our developers can be sure that our library system's each module works as intended by unit testing.

Our system is bottom up processing, so after unit testing is finished, we can easily integrate functions and modules because they are already tested through unit testing.

1. In component testing, where several individual functions are integrated to create each module, we focus on individual module's interfaces to integrate functions.

First, in Login module, there are functions that have roles of interfacing Administrator module and User module (adminLogin(), userLogin()).

Second, in User module, there are three functions that are role of interfacing between User module and User\_Data module(user\_Change(String), user\_Secession(), user\_Info(String)).

In User module, these functions update users' data so that User\_Data module is renewed.

Also, in User module, there are three functions that are role of interfacing between User module and Book\_Data. (book\_Search(String), overdue\_Calculate(), myBooklist())

In User module, these functions update user's booklist, and show user book's information such as writer, publisher, name, and code.

Third, in Administrator module, there are three functions that have roles of interfacing between Administrator module and User\_Data module(overdue\_Calculate(), user\_Search()), overdue\_Calculate() function should count overdue days of specific user, so this function roles interface between User\_Data module and Administrator module. user\_Search() function should know user's information such as user's book rent list, phone number and name.

Also, in Administrator module, there are five functions that have roles of interfacing between Administrator module and Book\_Data module(getBook\_List(), book\_ADD, book\_Setting(String), book\_Delete(String), book\_Search(String)).

These functions update book lists so, developers should know whether Book\_Data module changes. (Renewal of book lists)

1. After component testing, in release testing, our developer teams are divided into Administrator mode and User mode in our library system. Each team tests a complete version of Administrator mode and User mode before this program is distributed to the customers.
2. After release testing, in user testing, we will provide our complete version of library system to our potential users. Then we will get a feedback from our potential users. We will repeat this stage until our potential users can be satisfied with our library system.

## How to set up an automated daily (or more frequent) build and test

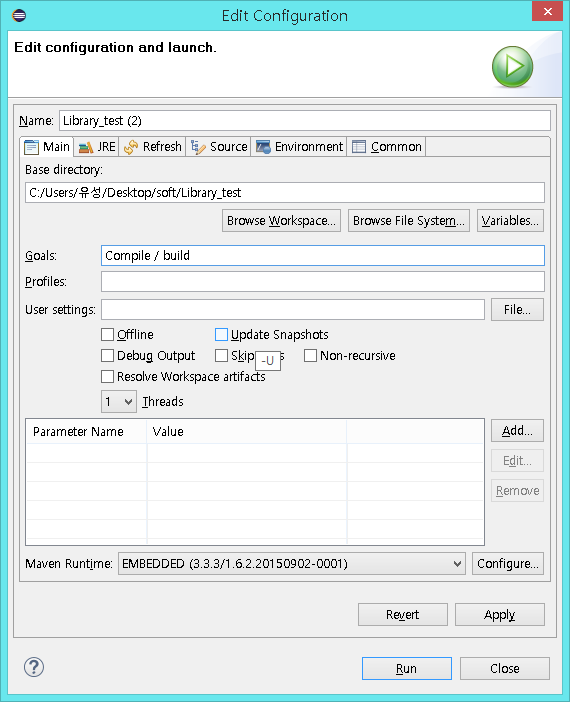
1. Our team manage source codes in GitHub. By interlocking GitHub and Jenkins, which is automated build tools, we will build automated daily build environment.

There are 3 things for preparations.

- Maven tools

- GitHub account

- Project to build in GitHub





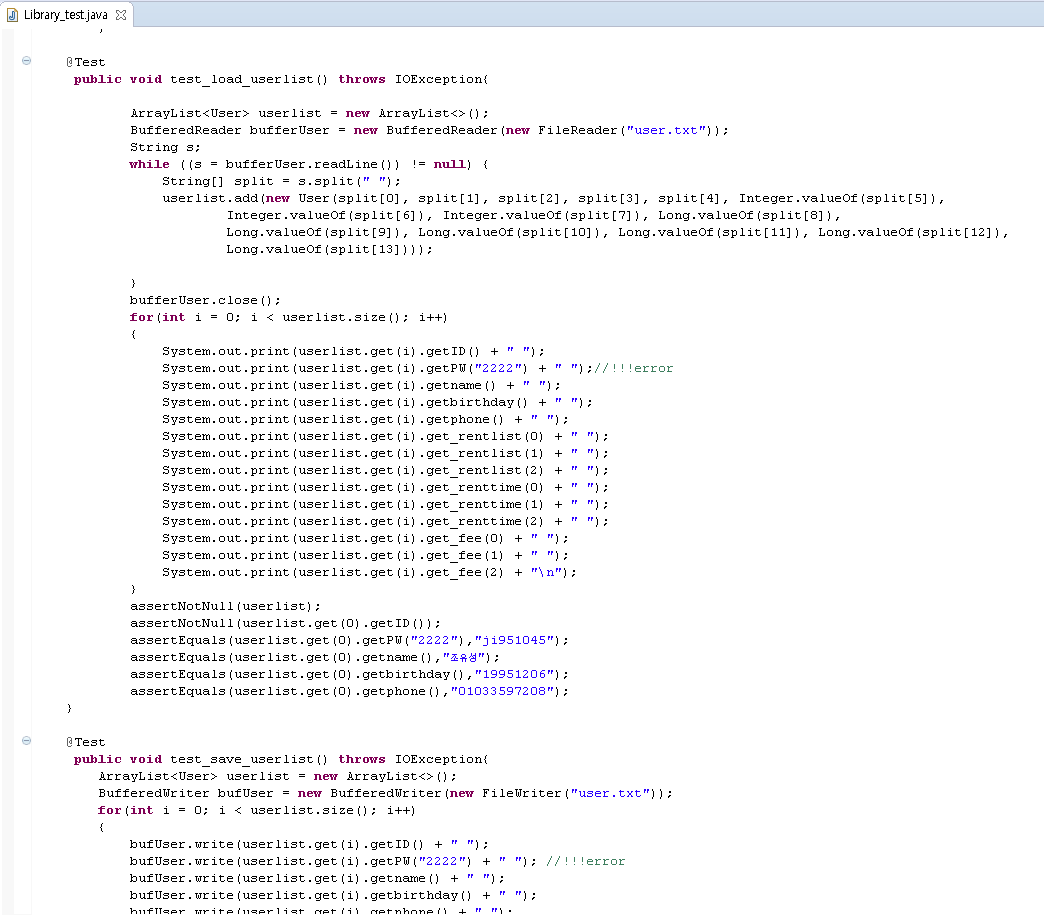
<Maven auto build>

2. We will use JUnit to implement automated test.

JUnit is a unit testing tool. We do not have to debug inconveniently for using system.out by writing external test program(case). It reduces time for program test and it is in Eclipse for plugin form.

The test result does not remain in just a text file, but remains in Test class so that it can produce test methods and the history of class to the developers. Also, simply uninherited Test may be classified as a unit test.

For test classification implement, the settings are like below.



## How to release a new version of your software

When we think we need improvement in our program or when we get feedbacks from users, our project team will review the content and after the discussion about the improvement of the program, we will design a new version. The designed content will be implemented by the developers using “*GitHub*” mentioned above. The completed source code is tested through a series of testing, and after the testing process, The new version will be uploaded to the project manager Taehyun Moon’s GitHub <https://github.com/xogus1107/librarymanagement>

## How to access the list of outstanding bugs and the list of resolved bugs, and how to resolve a bug

To access the list of outstanding bugs, we decided to notice bug lists in github

There is a issues categories in our github

