



HO CHI MINH UNIVERSITY OF SCIENCE
FACULTY OF INFORMATION TECHNOLOGY
SOFTWARE ENGINEERING DEPARTMENT
ADVANCED PROGRAM IN COMPUTER SCIENCE
COURSE: **PROGRAMMING SYSTEMS**
LECTURER: Dr. ĐINH BÁ TIẾN

WEEK 02

CLASS IMPLEMENTATION

- ✚ MSc. TRƯƠNG PHƯỚC LỘC
- ✚ MSc. HỒ TUẤN THANH

HCMC, October 26, 2016

Contents

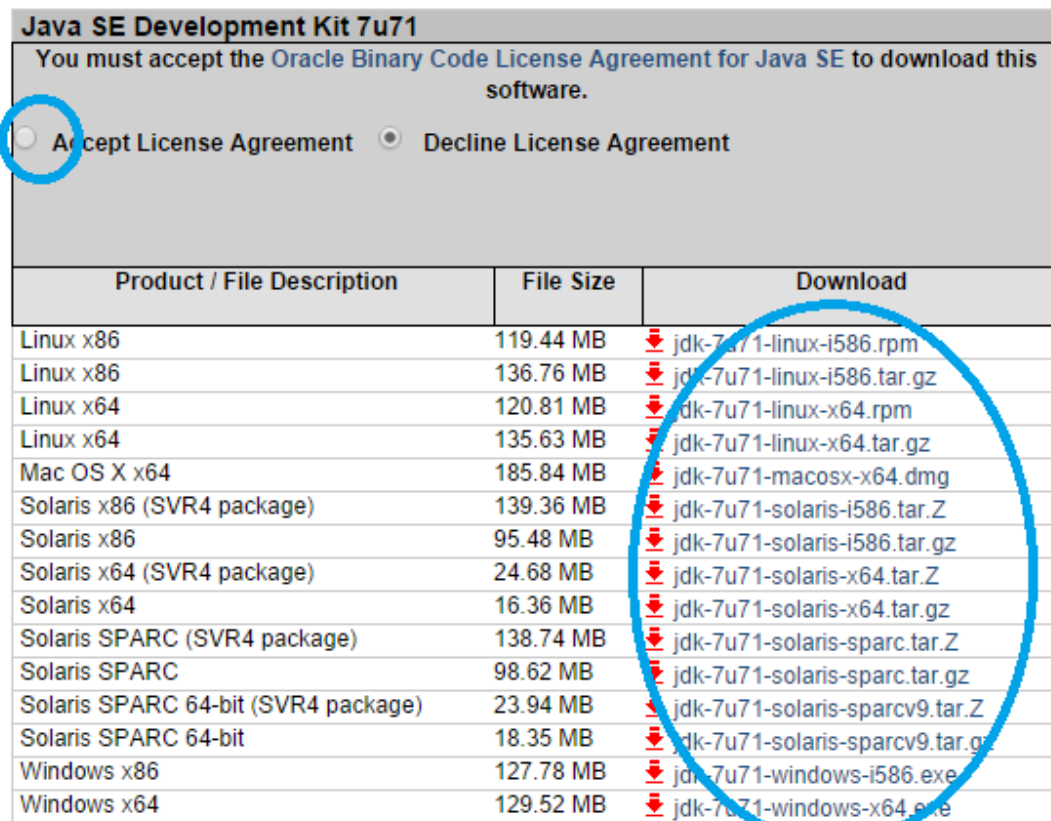
1	Tutorial for beginning java programming.....	3
1.1	Install JDK.....	3
1.2	Install Eclipse	3
1.3	Create java application with eclipse.....	4
1.3.1	Create project.....	4
1.3.2	Create the main class to contain main fuction:.....	6
1.4	Run application.....	8
1.5	Input & Output from console	8
1.5.1	Input.....	8
1.5.2	Output	8
1.5.3	Sample program.....	8
2	Exercises	9
3	Assignments (no diagrams).....	9
4	Homework.....	9

1 Tutorial for beginning java programming

For java programming, you need to install JDK and IDE (Eclipse, Netbean, ...)

1.1 Install JDK

- Download JDK7 from [link](#) in accordance with your OS (Linux, Mac, Windows 64bit, windows 32bit)



Java SE Development Kit 7u71

You must accept the [Oracle Binary Code License Agreement for Java SE](#) to download this software.

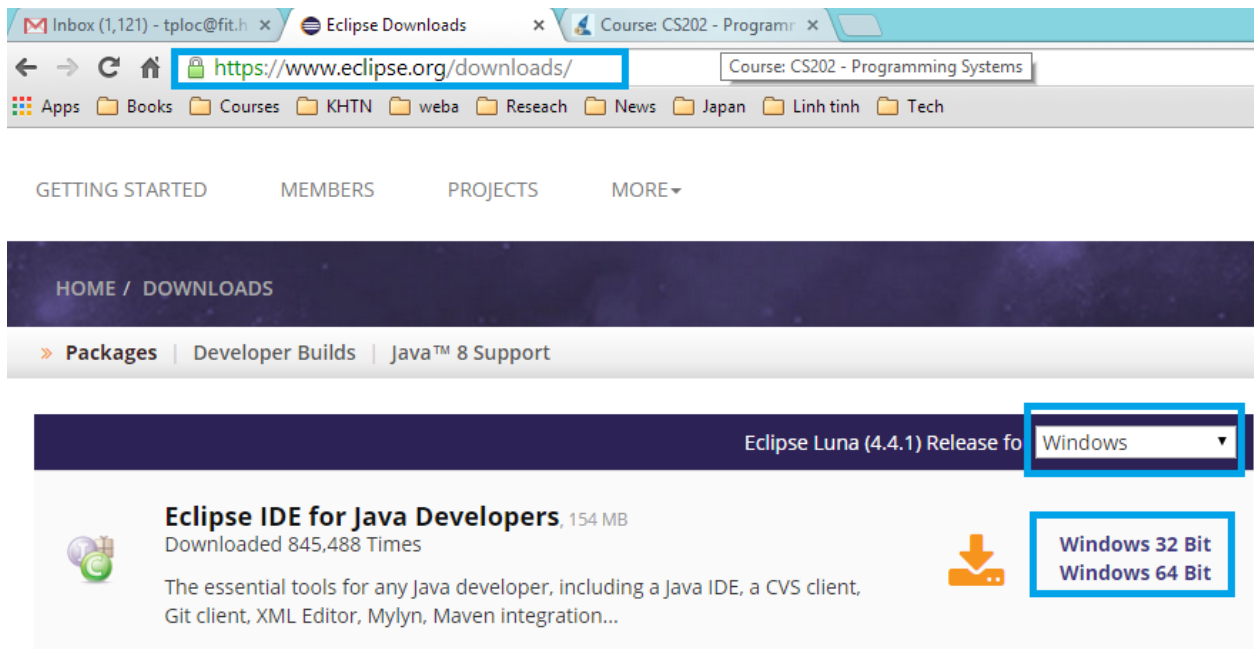
☒ Accept License Agreement ☐ Decline License Agreement

Product / File Description	File Size	Download
Linux x86	119.44 MB	jdk-7u71-linux-i586.rpm
Linux x86	136.76 MB	jdk-7u71-linux-i586.tar.gz
Linux x64	120.81 MB	jdk-7u71-linux-x64.rpm
Linux x64	135.63 MB	jdk-7u71-linux-x64.tar.gz
Mac OS X x64	185.84 MB	jdk-7u71-macosx-x64.dmg
Solaris x86 (SVR4 package)	139.36 MB	jdk-7u71-solaris-i586.tar.Z
Solaris x86	95.48 MB	jdk-7u71-solaris-i586.tar.gz
Solaris x64 (SVR4 package)	24.68 MB	jdk-7u71-solaris-x64.tar.Z
Solaris x64	16.36 MB	jdk-7u71-solaris-x64.tar.gz
Solaris SPARC (SVR4 package)	138.74 MB	jdk-7u71-solaris-sparc.tar.Z
Solaris SPARC	98.62 MB	jdk-7u71-solaris-sparc.tar.gz
Solaris SPARC 64-bit (SVR4 package)	23.94 MB	jdk-7u71-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	18.35 MB	jdk-7u71-solaris-sparcv9.tar.gz
Windows x86	127.78 MB	jdk-7u71-windows-i586.exe
Windows x64	129.52 MB	jdk-7u71-windows-x64.exe

- Run the file and follow the guide

1.2 Install Eclipse

- Download Eclipse from [link](#)

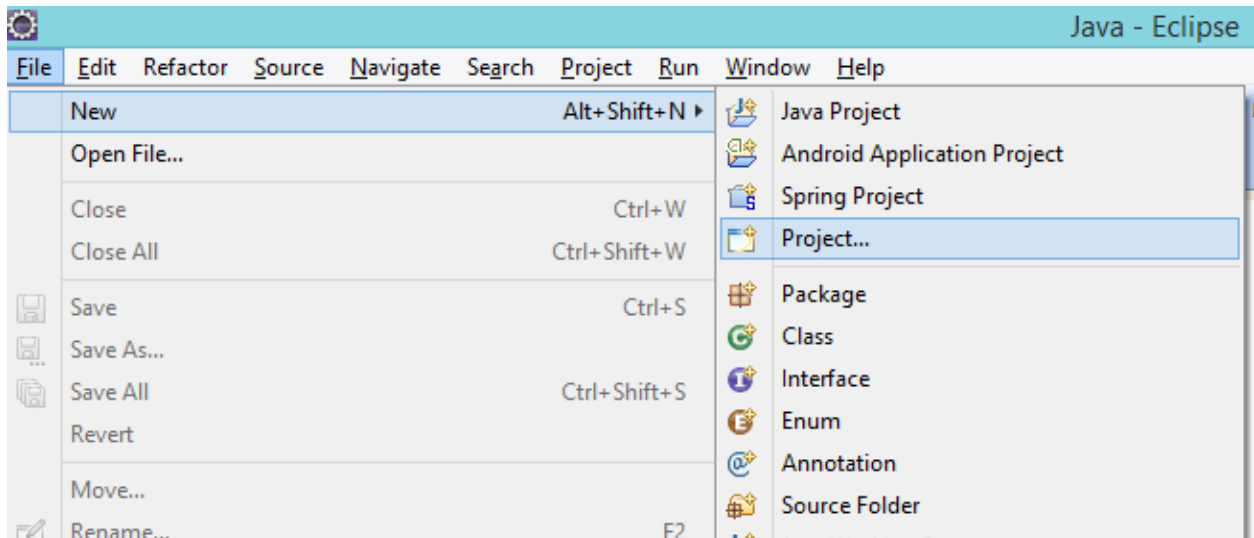


- Unzip to the folder
- Run **eclipse** file.

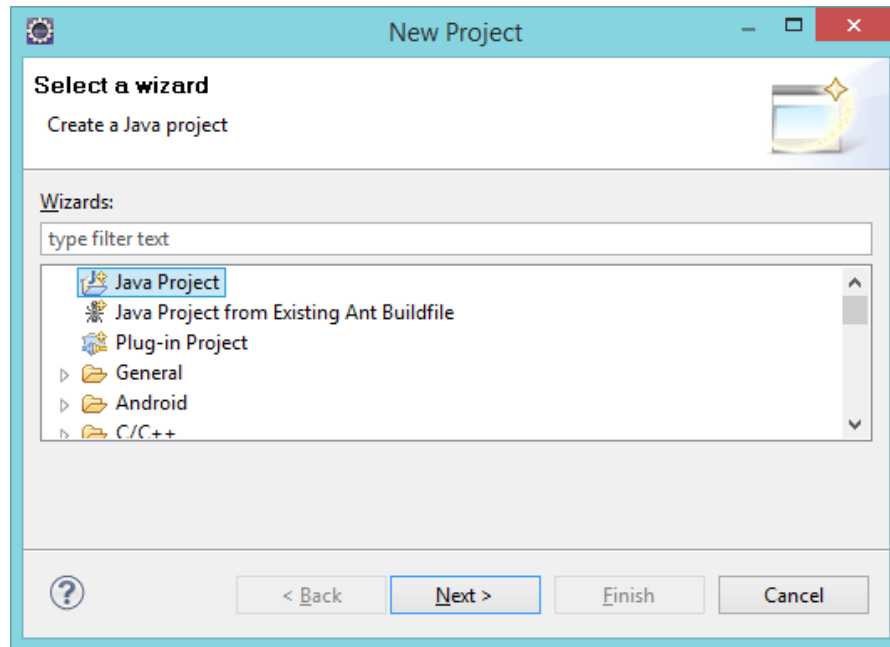
1.3 Create java application with eclipse

1.3.1 Create project

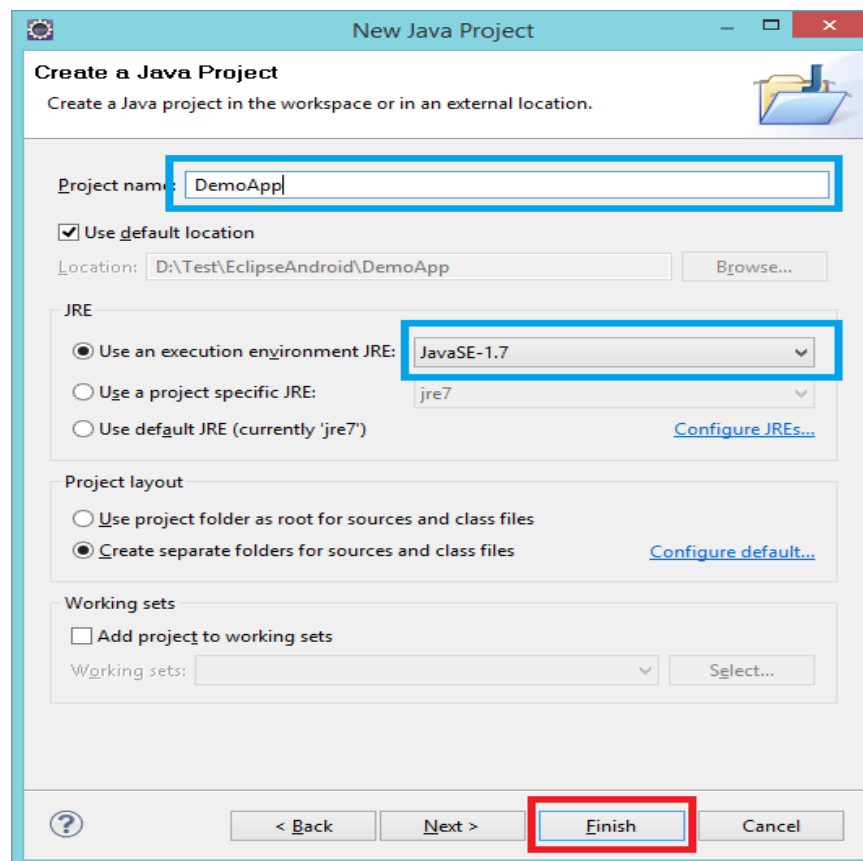
- Menu File => New => Project



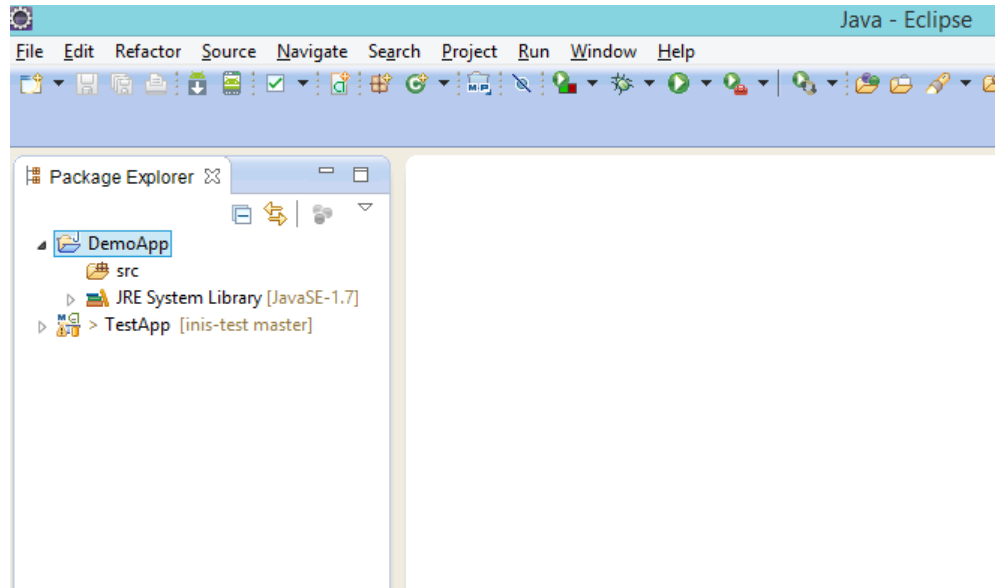
- In “New project” dialog => choose Java Project => Next



- In “New Java Project”:
 - Project Name: ...
 - JRE: ...
 - ⇒ **Finish**

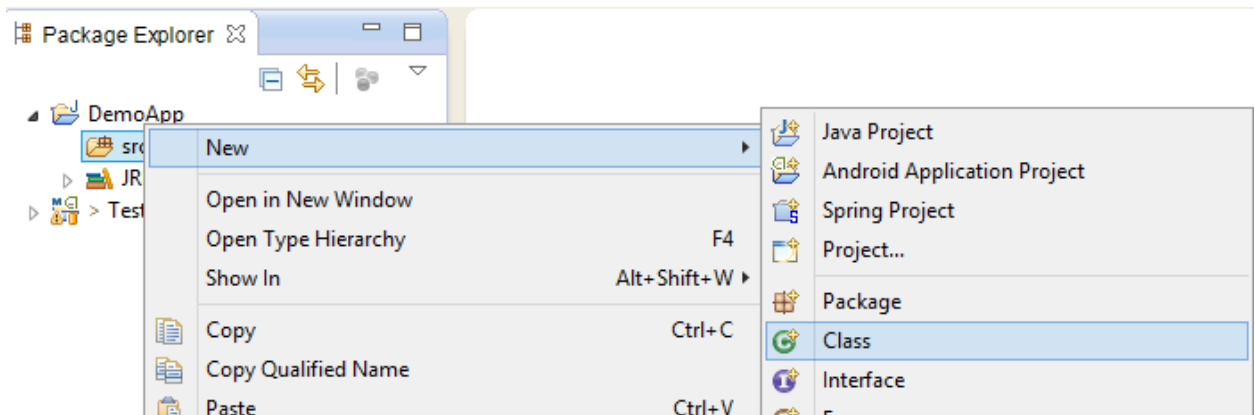


- The result



1.3.2 Create the main class to contain **main function**:

- Right click on src => New => class



- In “New Class Java” dialog
 - o Package: ...
 - o Name: ...
 - o Public static void main (String [] args): **only check when you create main function**
 - ⇒ Finish

Java Class
Create a new Java class.

Source folder: DemoApp/src Browse...

Package: app Browse...

☐ Enclosing type: Browse...

Name: Main

Modifiers: ☒ public ☐ default ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add...

Which method stubs would you like to create?

☒ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

? Finish Cancel

- The result

Package Explorer

- DemoApp
 - src
 - app
 - Main.java
- JRE System Library [JavaSE-1.7]
- TestApp [inis-test master]

Main.java

```
package app;

public class Main {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
    }

}
```

1.4 Run application

- Right click to Project => Run As => Java Application

1.5 Input & Output from console

1.5.1 Input

- Create **object** of **Scanner** class in **java.util.Scanner**
- Call **nextLine()** function

```
String name;  
Scanner in = new Scanner(System.in);  
  
name = in.nextLine();
```

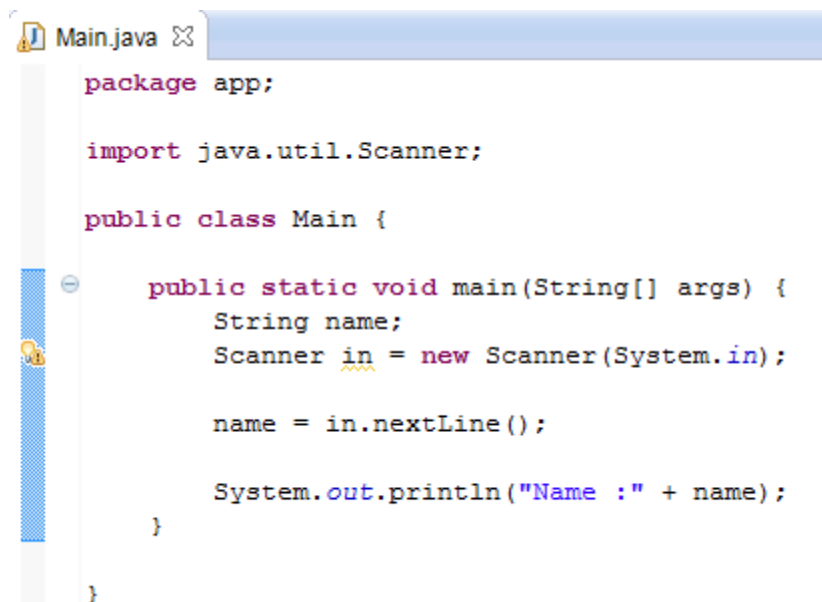
1.5.2 Output

- Use **System.out.print(...)** or **System.out.println("...")**

```
System.out.println("Name :" + name);
```

1.5.3 Sample program

- User input name from screen, and program output the name



```
Main.java  
package app;  
  
import java.util.Scanner;  
  
public class Main {  
  
    public static void main(String[] args) {  
        String name;  
        Scanner in = new Scanner(System.in);  
  
        name = in.nextLine();  
  
        System.out.println("Name :" + name);  
    }  
}
```


2 Exercises

For each exercise below, draw a class diagram and implement it in C++ and Java (**only input from screen**).

1. Write a console application to load an array of fractions from a text file. Find the sum of them. Find the maximum and the minimum of them. Sort the array ascending.
2. Write a console application to load an array of points (Oxy plane) from a text file. Allow user to input a point P. Find an element in the array that the distance from it to P is longest.
3. Write a console application to load an array of triangles from file. Remove all invalid triangles. How many invalid triangles did you have to remove? How many equilateral triangles? Right isosceles triangle? Right triangles? Acute triangles? Obtuse triangles? are there in the array?
4. Write a console application to input a date (day, month, and year). Output to screen the date before (yesterday), the date after (tomorrow).
5. Write a console application to input a time (hour, minute, second). Increase the time to one second and output to screen. What is the result if we decrease the time to one second?
6. Write a console application to load an array of students (id, full name (first name and last name), address, date of birth, mark) from text file. Sort them by their last name and write the array to and XML file.

3 Assignments (no diagrams)

1, 2

4 Homework

1 – 6 (full)