

3 Getting to know your Programming Environment



Objectives

At the end of the lesson, the student should be able to:

- Create a Java program using text editor and console in the Linux environment
- Differentiate between syntax-errors and runtime errors
- Create a Java program using NetBeans



Definitions

- Console
 - This is where you type in commands
 - Examples are Terminal (Linux), MSDOS Command Prompt (Windows)



Definitions

- Text Editor
 - Examples: Notepad, Wordpad, Vi



Definitions

- Integrated Development Environment or IDE
 - a programming environment integrated into a software application that provides a GUI builder, a text or code editor, a compiler and/or interpreter and a debugger.



My First Java Program

```
1 public class Hello {  
2  
3     /**  
4      * My first Java program  
5      */  
6     public static void main( String[] args ){  
7  
8         //prints the string "Hello world" on screen  
9         System.out.println("Hello world");  
10  
11     }  
12 }
```



Using Text Editor and Console

- NOTE:
 - This will be demonstrated by the teacher
 - Environment used: Ubuntu Dapper
 - For Windows Environment: Refer to Appendix B in your student Manual



Using Text Editor and Console

- Step 1: Start the Text Editor
 - To start the Text Editor in Linux, click on Menu-> Accessories-> Text Editor
- Step 2: Open Terminal
 - To open Terminal in Linux, click on Menu-> System Tools-> Terminal
- Step 3: Write your the source code of your Java program in the Text Editor



Using Text Editor and Console

- Step 4: Save your Java Program
 - Filename: Hello.java
 - Folder name: MYJAVAPROGRAMS
 - To open the Save dialog box, click on the File menu found on the menubar and then click on Save.
 - If the folder MYJAVAPROGRAMS does not exist yet, create the folder



Using Text Editor and Console

- Step 5: Compiling your program
 - Go to the Terminal window
 - Go to the folder MYJAVAPROGRAMS where you saved the program
 - To compile a Java program, we type in the command:
`javac [filename]`
 - So in this case, type in:
`javac Hello.java`

During compilation, javac adds a file to the disk called [filename].class, or in this case, Hello.class, which is the actual bytecode.



Using Text Editor and Console

- Step 6: Running the Program
 - To run your Java program, type in the command:
`java [filename without the extension]`
 - so in the case of our example, type in:
`java Hello`
 - You can see on the screen after running the program:
`"Hello world!"`

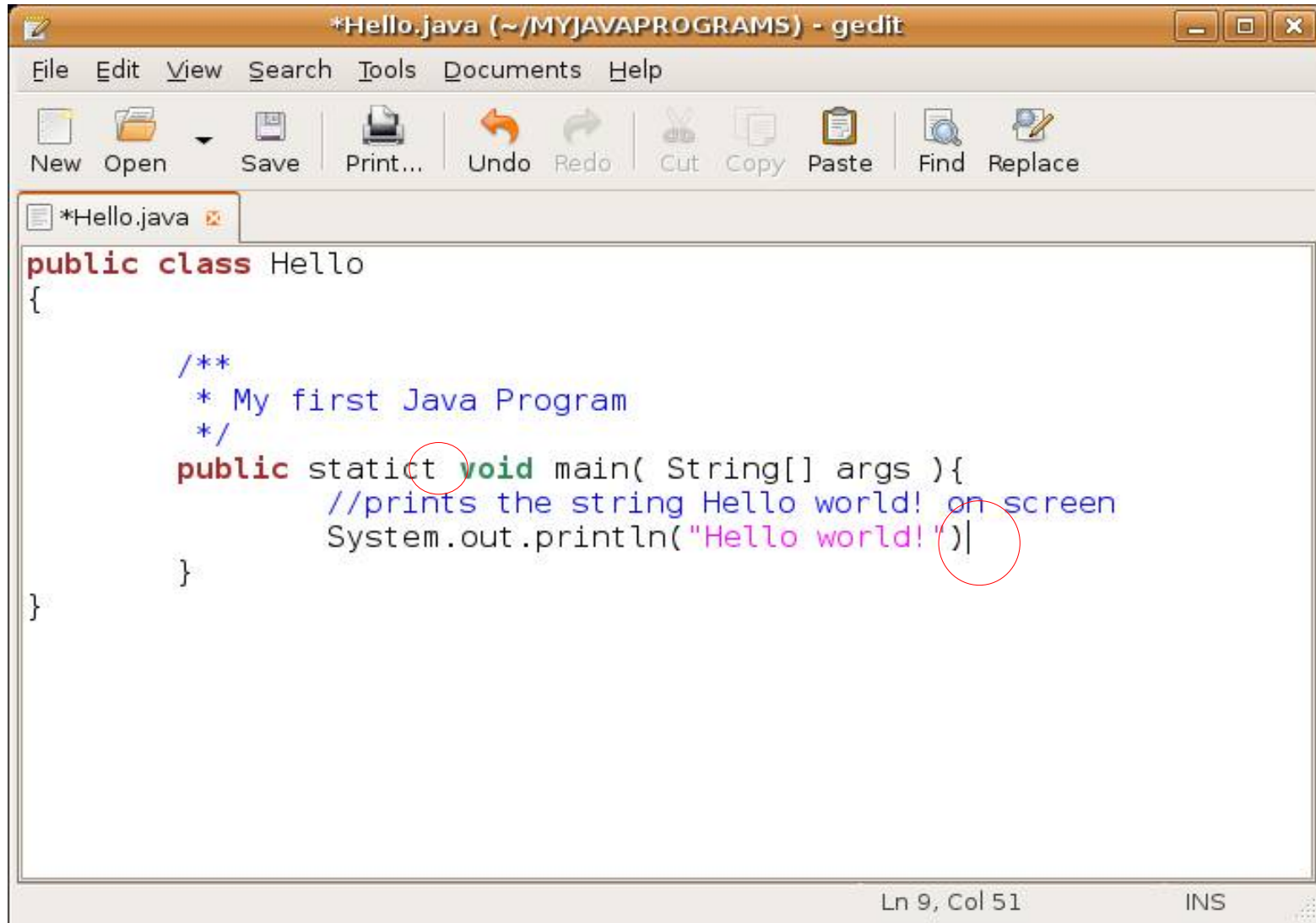


Errors : Syntax Errors

- Syntax Errors Syntax
 - errors are usually typing errors
- Common Syntax Errors:
 - misspelled a command in Java
 - forgot to write a semi-colon at the end of a statement



Example: Syntax Error



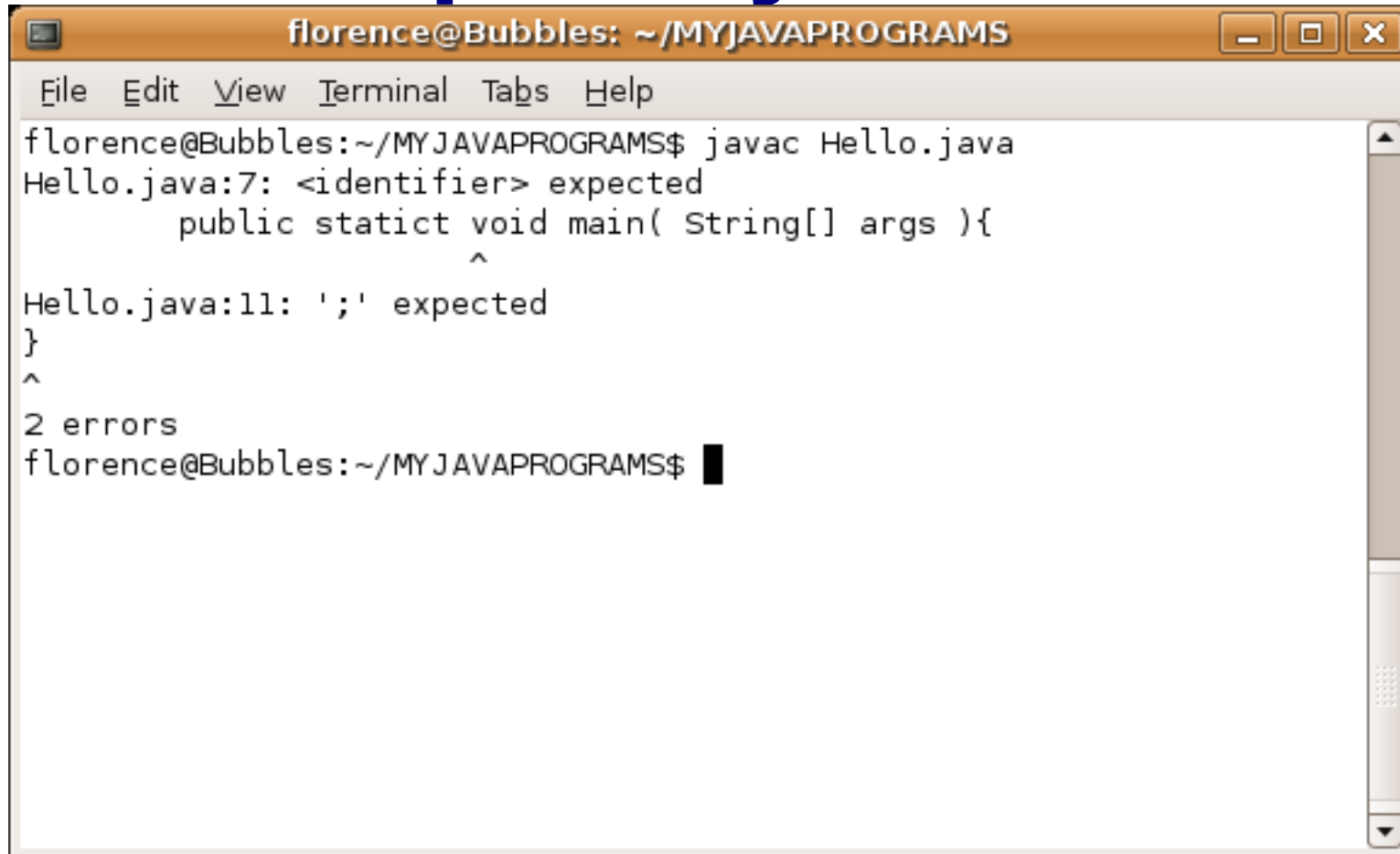
The screenshot shows a gedit editor window titled "*Hello.java (~ /MYJAVAPROGRAMS) - gedit". The menu bar includes File, Edit, View, Search, Tools, Documents, and Help. The toolbar contains icons for New, Open, Save, Print..., Undo, Redo, Cut, Copy, Paste, Find, and Replace. The code editor displays the following Java code:

```
public class Hello
{
    /**
     * My first Java Program
     */
    public static void main( String[] args ){
        //prints the string Hello world! on screen
        System.out.println("Hello world!")|
    }
}
```

Two red circles highlight syntax errors: one around the word "static" and another around the closing parenthesis of the `println` statement. The status bar at the bottom right indicates "Ln 9, Col 51" and "INS".



Example: Syntax Error

A screenshot of a terminal window titled 'florence@Bubbles: ~/MYJAVAPROGRAMS'. The window has a menu bar with 'File', 'Edit', 'View', 'Terminal', 'Tabs', and 'Help'. The terminal content shows the command 'javac Hello.java' being executed. Two syntax errors are reported: 'Hello.java:7: <identifier> expected' pointing to 'statict' in 'public statict void main', and 'Hello.java:11: ';' expected' pointing to the closing brace '}' of the main method. The terminal concludes with '2 errors' and a new command prompt.

```
florence@Bubbles: ~/MYJAVAPROGRAMS
File Edit View Terminal Tabs Help
florence@Bubbles:~/MYJAVAPROGRAMS$ javac Hello.java
Hello.java:7: <identifier> expected
        public statict void main( String[] args ){
                ^
Hello.java:11: ';' expected
    }
    ^
2 errors
florence@Bubbles:~/MYJAVAPROGRAMS$
```



Errors: Runtime Errors

- Run-time Errors
 - errors that will not display until you run or execute your program
 - Even programs that compile successfully may display wrong answers if the programmer has not thought through the logical processes and structures of the program.
 - Examples:
 - You want your program to print 100 strings of “Hello world”, but it only printed 99.
 - Your program gets an input from the user, but the user inputted a character, and so your program crashes/terminates



Using NetBeans

- NOTE:
 - This will be demonstrated by the teacher
 - Environment used: Ubuntu Dapper
 - For Windows Environment: Refer to Appendix B in your student Manual



Using NetBeans

- Step 1: Run NetBeans
 - Two ways to run NetBeans:
 - Through command-line using terminal
 - By just clicking on the shortcut button found on the menu
 - To run NetBeans using command-line
 - Open terminal (see steps on how to run terminal in the previous discussion), and type: netbeans
 - The second way to run NetBeans
 - Click on shortcut on the desktop



Using NetBeans

- Step 2: Make a project
 - To make a project, click on File-> New Project
 - After doing this, a New Project dialog will appear
 - On the right pane of the project dialog, click on Java Application and click on the NEXT button
 - A New Application dialog will appear. Edit the Project Name part and type in "HelloApplication"



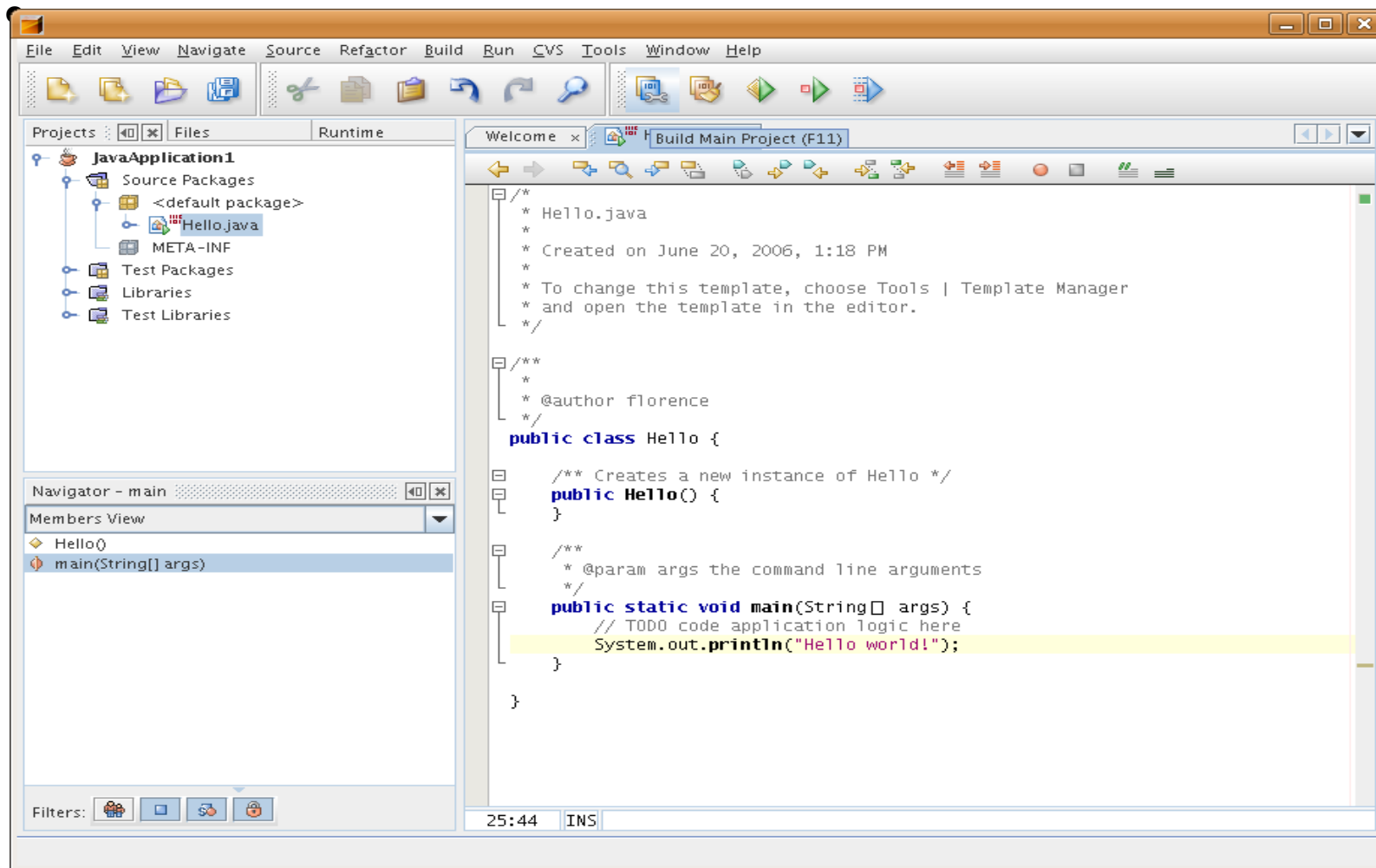
Using NetBeans

- Step 2: Make a project (continuation)
 - Now try to change the Application Location, by clicking on the BROWSE button
 - A Select Project Location dialog will then appear. Double-click on the root folder.
 - The contents of the root folder are then displayed. Now double-click on the MYJAVAPROGRAMS folder and click on the OPEN button
 - Finally, on the Create Main Class textfield, type in Hello as the main class' name, and then click on the FINISH button



Using NetBeans

- Step 3: Type in your program



Using NetBeans

- Now, try to modify the code generated by NetBeans.
- Ignore the other parts of the program for now, as we will explain the details of the code later.

- Insert the code

```
System.out.println("Hello world!");
```

after the statement,

```
//TODO code application logic here.
```



Using NetBeans

- Step 4: Compile your program
 - To compile your program, just click on Build -> Build Main Project
 - If there are no errors in your program, you will see a build successful message on the output window
- Step 5: Run your program
 - To run your program, click on Run-> Run Main Project
 - The output of your program is displayed in the output window



Summary

- My First Java Program
- Using a Text Editor and Console
 - Write program
 - Compile program
 - Run program
- Errors
 - Syntax Errors
 - Runtime Errors
- Using NetBeans
 - Write program
 - Compile program
 - Run program

