Assignment 1

Due: 30 Jan 2017 at 11:59pm

Implement a scanner for the programming language with the following lexical structure.

```
comment ::=   /*   NOT(*/)*  */
token ::= ident  | keyword | frame_op_keyword | filter_op_keyword | image_op_keyword | boolean_literal
          | int_literal  | separator  | operator
ident ::= ident_start  ident_part*    (but not reserved)
ident_start ::=  A .. Z | a .. z | $ | _
ident_part ::= ident_start | ( 0 .. 9 )
int_literal ::= 0  |  (1..9) (0..9)*
keyword ::= integer | boolean | image | url | file | frame | while | if | sleep | screenheight | screenwidth
filter_op_keyword ::= gray | convolve | blur | scale
image_op_keyword ::= width | height
frame_op_keyword ::= xloc | yloc | hide | show | move
boolean_literal ::= true | false
separator ::=     ;  | ,  |  (  |  )  | { | }
operator ::=      |  | &  |  ==  | !=  | < |  > | <= | >= | +  |  -  |  *   |  /   |  %  | !  | -> |  |-> | <-
```

- Use the provided Scanner.java and ScannerTest.java as starting points.
- If an illegal character is encountered, your scanner should throw an IllegalCharException. The message should contain useful information about the error. The contents of the message will not be graded, but you will appreciate it later if they are helpful.
- If an integer literal is provided that is out of the range of a Java int, then your scanner should throw an IllegalNumberException. The contents of the message will not be graded, but you will appreciate it later if they are helpful.

Turn in a jar file containing the source code Scanner.java and ScannerTest.java.

Your ScannerTest will not be graded, but may be looked at in case of academic honesty issues. We will subject your scanner to our set of junit tests and your grade will be determined solely by how many tests are passed. Name your jar file in the following format:
firstname_lastname_ufid_hw1.jar

Additional requirements:

- This code must remain in package cop5556sp17(case sensitive): do not create additional packages.
- Names (of classes, method, variables, etc.) in starter code must not be changed.
- Unless otherwise specified, your code should not import any classes other than those from the standard Java distribution.

Submission Checklist

- **Make sure that sources are included in the jar file**. Many IDEs (including Eclipse) do not do this by default.
    - A quick reference for how to export jar file from eclipse
    - If you are not using Eclipse, check Creating a JAR file
- To ensure that we will be able to compile and run your submission: upload your jar file to one of the uf cise server, e.g. storm.cise.ufl.edu, uncompress it and run from the command line. Instructions:
    - Copy/upload your file to cise server. If your OS is windows, try to install some unix client like putty for this step. Or you can use some ftp client(e.g. Filezilla) and skip this step. Suppose your cise id is *username*, the following instruction will upload the *HW1.jar* to your home folder on cise server:
      ```
      scp /my/path/to/HW1.jar username@storm.cise.ufl.edu:~/
      ```
    - Uncompress file:
      ```
      jar xf HW1.jar
      ```
        - If you packaged everything correctly, your uncompressed project directory structure will looks like following:
          ```
          cop5556fa17
                  |--Scanner.java
                  |--ScannerTest.java
                  |-- *all the other files*
                  |-- ...
          ```
    - Compile:
      ```
      javac -cp .:/usr/share/java/junit4.jar:/usr/share/java/hamcrest-core.jar
      cop5556fa17/*.java
      ```
    - Run junit test from command line:
      ```
      java -cp .:/usr/share/java/junit4.jar:/usr/share/java/hamcrest-core.jar
      org.junit.runner.JUnitCore cop5556fa17.ScannerTest
      ```

Hints: Please make sure that your jar file has the same directory structure with the original one that you downloaded from Canvas. Otherwise you will not be able to pass this test following the instructions above. **No matter how your program runs on your own machine, if it fails to compile/run on the cise server(storm or thunder), your homework 1 will get a zero grade, and there is no regrade. So double check before your submission.**

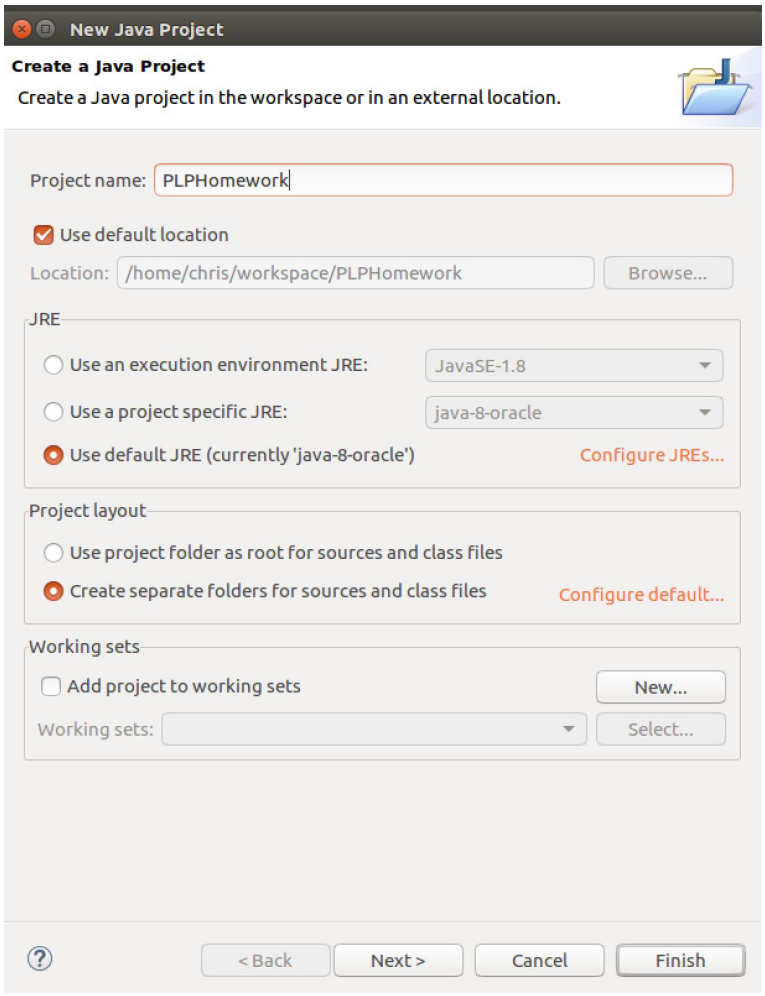
Comments and suggestions:

- The given scanner should compile correctly with the junit test. When executed, only one test will pass, but all should pass in your completed scanner.
- Work incrementally: add a capability along with a junit test to exercise it incrementally
- You will probably want to develop some methods to encapsulate checks to make it easier to write JUnit test cases.
- If you use Integer.parseInt to get the value of a numeric literal, it will throw a NumberFormatException if the value is too large. This is useful functionality, but the exception is not the same one as specified. You need to catch it and throw a Scanner.LexicalException with a useful message.
- If you use Eclipse to work with the assignments, it is suggested to create a project and import the jar files (eg.HW1.jar) provided by each assignment into the project. After completing your work on the source files (keep all source files within the package cop5556fa17), you can export the package cop5556fa17 as a jar file for submission (remember to select the option of including source files in the jar package), so that it may have the same directory structure with the original jar file, and pass the above test on storm.

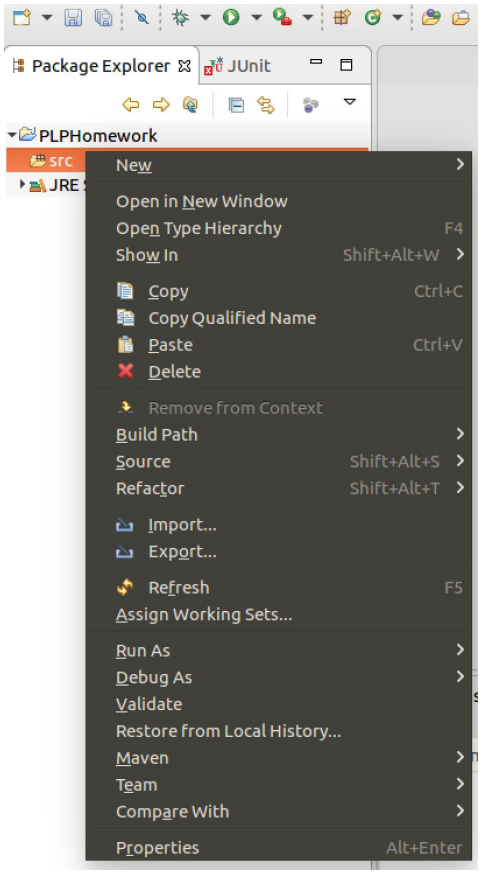
A Quick Tutorial on How to Start Homework 1 in Eclipse:

1. Create a project (e.g. PLPHomework)

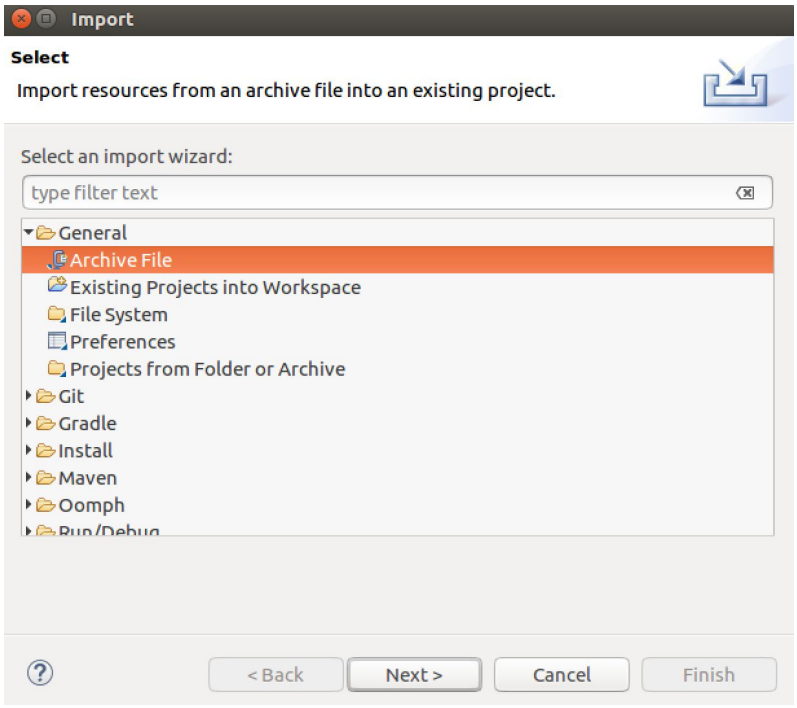
   `File->New->Java Project`



2. After project created, right click on the src folder in the left sidebar, choose `Import…`

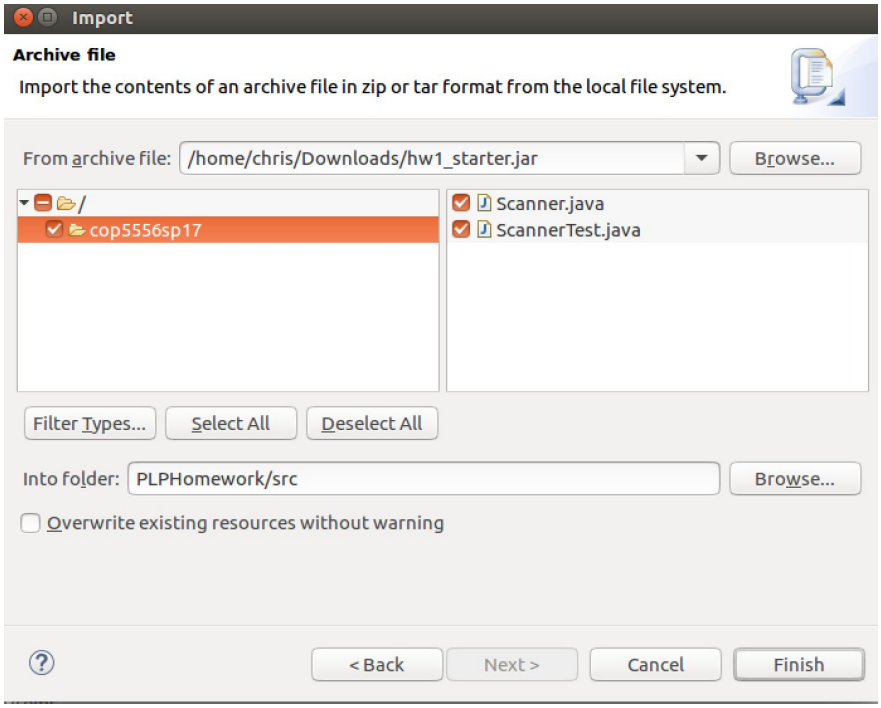


Select `General->Archive File`

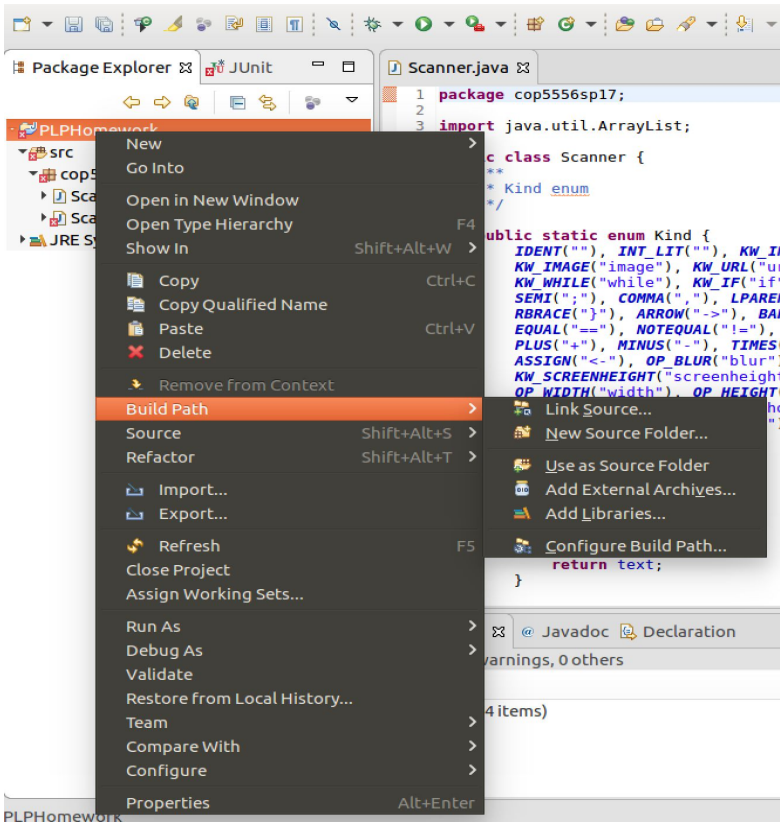


Browse and choose your downloaded `h1_starter.jar`, make sure both

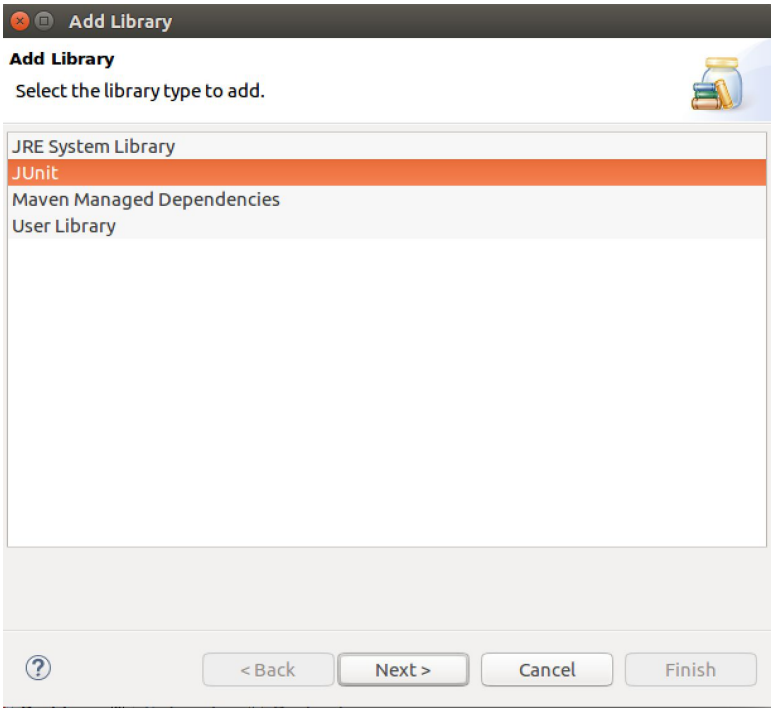
`Scanner.java` and `ScannerTest.java` have been checked



3. Add Junit Library to Build Path.
   Right Click on project, select `Build Path->Add Libraries…`



In the list, choose JUnit



4. To Run the unit tests