CSE332: Data Abstractions

Section 1

Hyeln Kim

Spring 2013

Section Agenda

- Introduction
- Generics
- Project 1
- Eclipse Tutorial

Introduction - Me

- HyeIn (Han) Kim
- From Korea
- 5th year Master's student
- BS in Biology & CSE at UW
- Teaching section AA & AB
  - Office hour: Thursday 12:30 ~ 01:30, CSE 216
  - Email: kainby87@uw.edu

Introduction - You

- Name
- Year
- Home Town
- Interesting Fact about yourself
- What you did over the break

Generics

- **What is generics?**

  - Technique of writing class/Interface without specifying type of data it uses

  - Idea: class/interface can have type parameter
    Usually denoted as T or E

Generics

- **Want a Bag class to store Items**

```
public class Bag {
        private String item;
        public void setItem(String x) {  item = x;  }
        public String getItem( )  {  return item;  }
}

public class Bag {
        private Book item;
        public void setItem(Book x) {  item = x;  }
        public Book getItem( )  {  return item;  }
   }
```

- Don’t want to make Bag class for all kind of fields.

Generics

- **Want a Bag class to store Items**

```
public class Bag<E> {
        private E item;
        public void setItem(E x) {  item = x;  }
        public E getItem( )  {  return item;  }
}


public interface List <E> {
          void add(E x);
          …
}
```

Generics

- **Want a Bag class to store Items**

  - Pre Java 5: Objects

```
public class Bag {
        private Object item;

        public void setItem(Object x ) {   item = x;   }
        public Object getItem() {  return item;  }
}

Bag b = new Bag();
b.setItem("How about that?”);
String contents = (String) b.getItem();
```

Generics

- **Why generics?**

```
Bag b = new Bag();
b.setItem( "How about that?" );
String contents = (String) b.getItem();        // Ok
double contents = (double) b.getItem();        // Error (Runtime)

Bag b = new Bag<String>();
b.setItem( "How about that?" );
String contents = b.getItem();                 // Ok
double contents = (double) b.getItem();        // Error (Compile time)
```

Generics

- **Why generics?** Type Safe Containers

  - Main advantage: compile-time type checking

  - Generics: Ensure correct type at compile time
    No need for cast or Type checking

* Note: Cannot create generic array!

```
E[] myArray = new E[INITIAL_SIZE];                  // Error
E[] myArray = (E[]) new Object[INITIAL_SIZE];       // Ok
```

Generics

- **More Generics: References**

  - Generics & Inheritance
  - Wild cards

http://www.cs.washington.edu/education/courses/cse332/12sp/section/week1/GenericsAndJUnit.pdf

http://www.cs.washington.edu/education/courses/cse332/12sp/section/week1/Bag.java

http://www.cs.washington.edu/education/courses/cse332/12sp/section/week1/Tuple.java

Textbook 1.4 ~ 1.5

Project 1

- **Sound Blaster!**

  - Part A: Due next Wednesday, 11pm
    Part B: Due Tuesday April 16th, 11pm

  - Personal Project: No partners!

Collaboration Policy
http://www.cs.washington.edu/education/courses/cse332/13wi/policies.shtml

Grading Policy
http://www.cs.washington.edu/education/courses/cse332/13wi/grading-policies.shtml

Programming Guidelines
http://www.cs.washington.edu/education/courses/cse332/13wi/programming-guidelines.shtml

Project 1

- **Part A**
  - Implement Stack ADT: Stores double
    - Implement interface DStack
    - Using Array (ArrayStack)
    - Using Linked List (ListStack)

- **Part B**
  - Implement Stack ADT: Use generic
    - Implement interface GStack
    - Using Array (GArrayStack)
    - Using Linked List (GListStack)

Project 1

- **Reverse.java**

  - Handles all music stuff
  - No need to edit for part A
  - Reverses in.dat file and writes it to out.dat
  - Accepts 4 command line parameters

    Stack Implementation: array or list
    Content type: double or generic
    Input file name: ex) in.dat
    Output file name: ex) out.dat

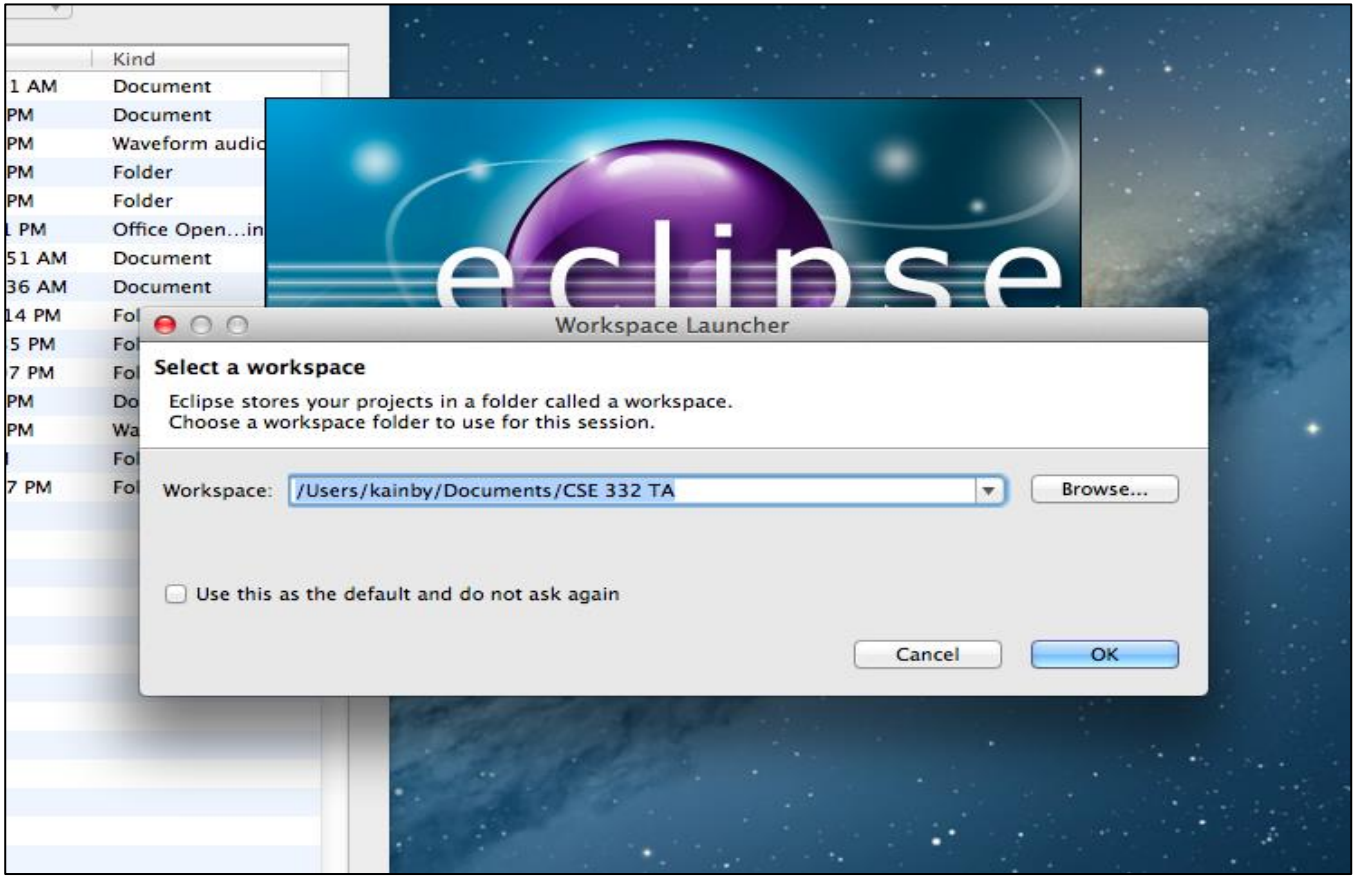
Project 1

- **Sound Exchange (SOX)**

  - Converts .wav file to .dat file & vice versa
    Reverse.java needs .dat file
    You need .wav file to play sound
  - Installed in lab machine
  - Use in command prompt / terminal
    ex) sox secret.wav secret.dat
  - Can also do what Reverse.java does
    ex) sox secret.wav secret_rev.wav reverse

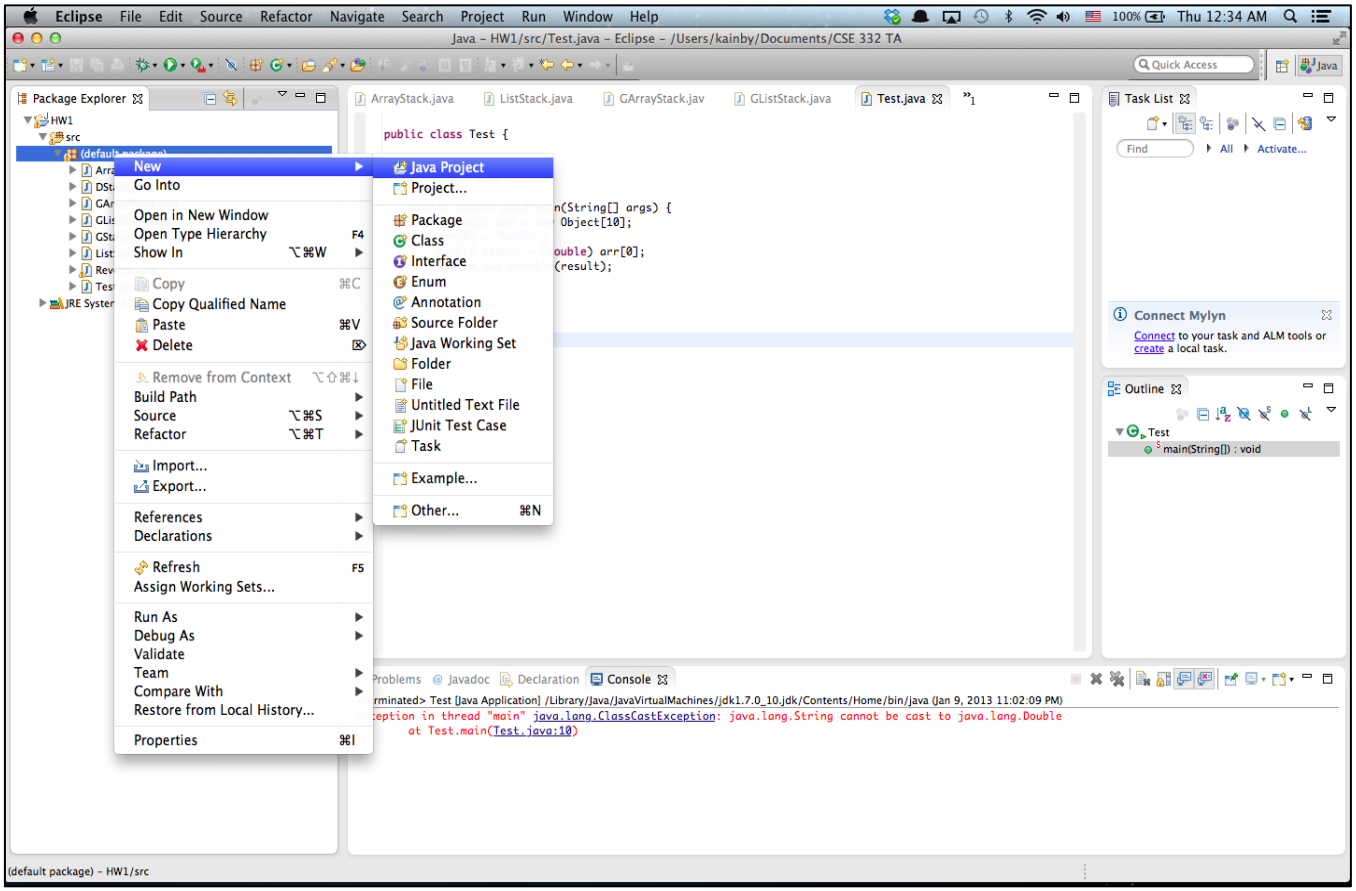
Eclipse Tutorial

- **Select WorkSpace**



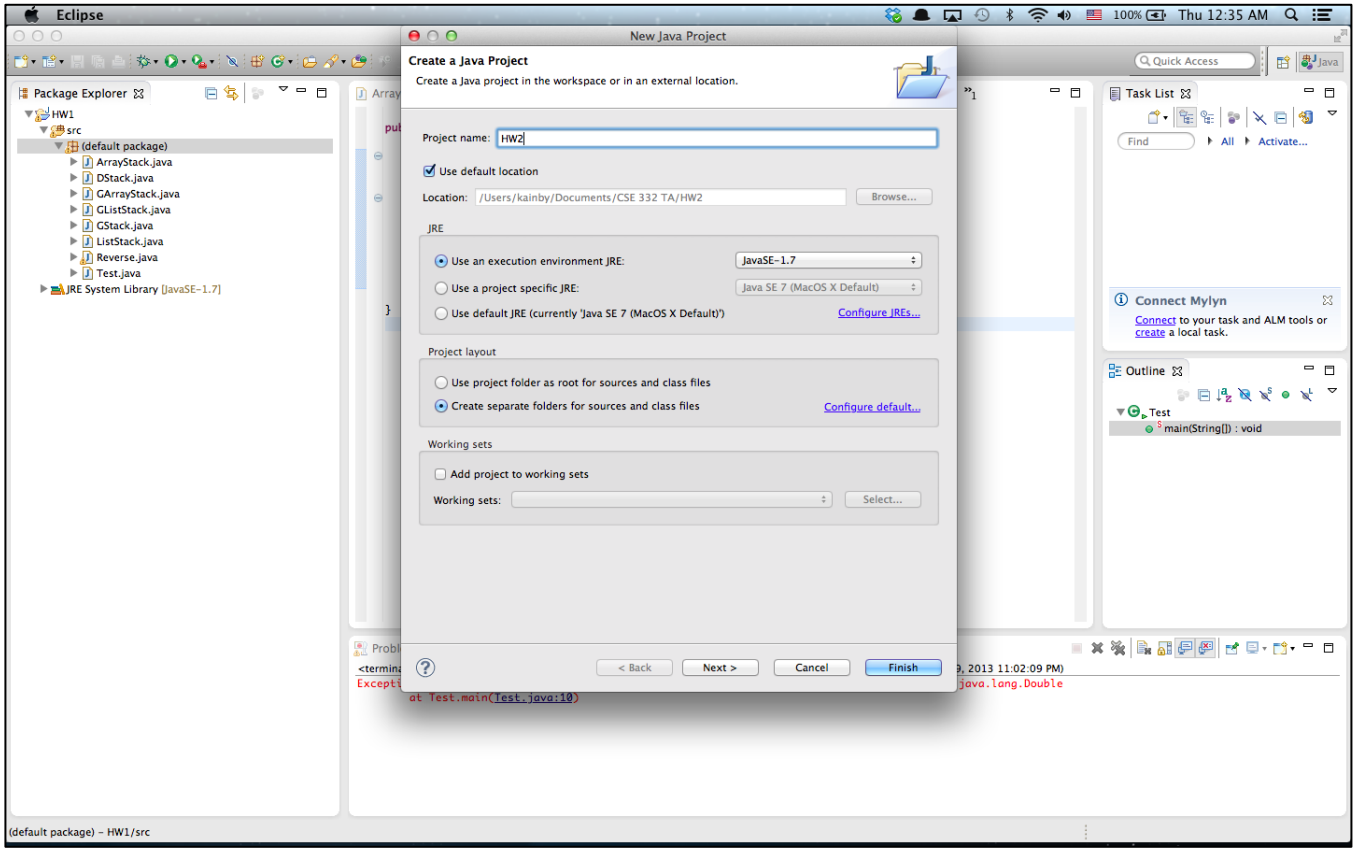
Eclipse Tutorial

- **Create Project**



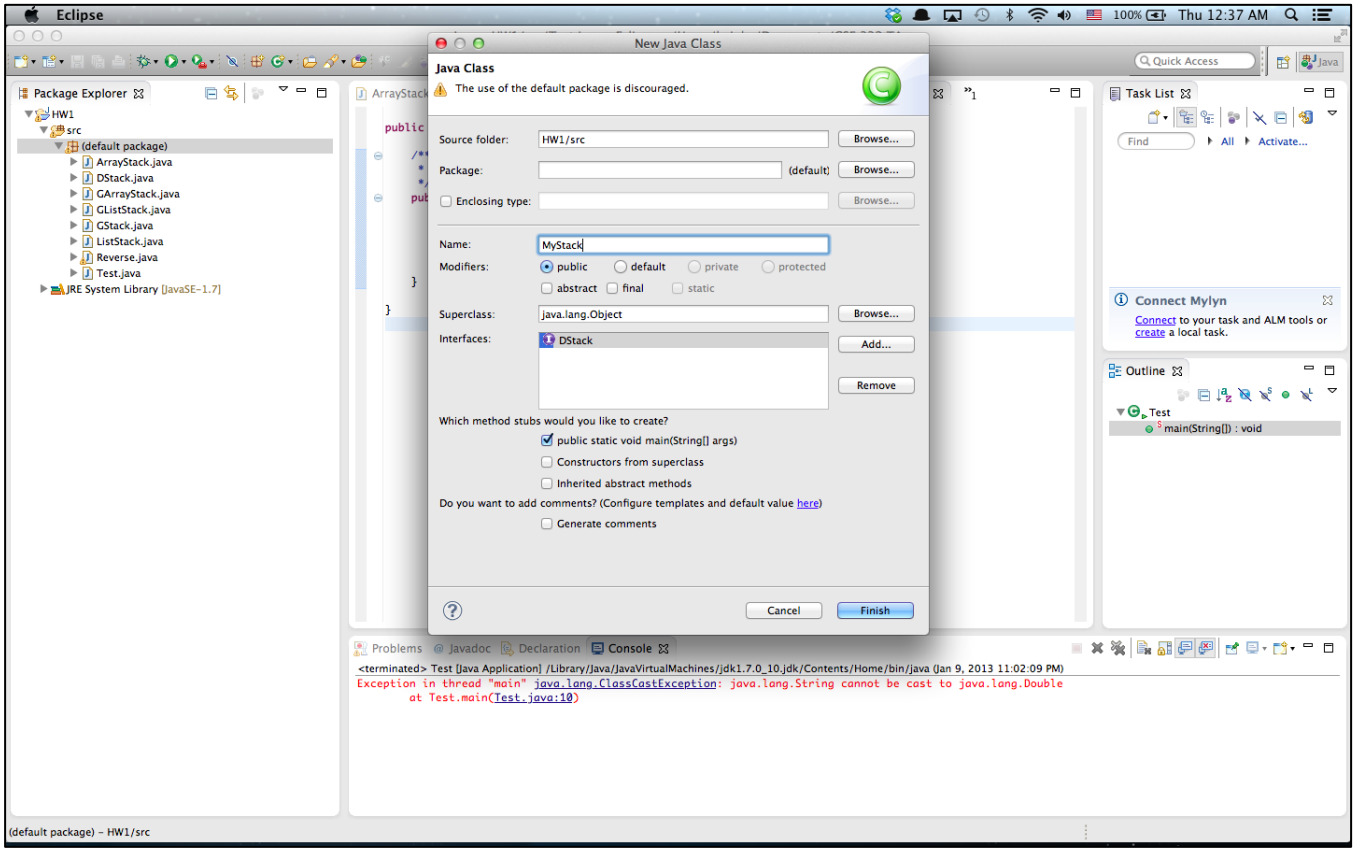
Eclipse Tutorial

- **Create Project**



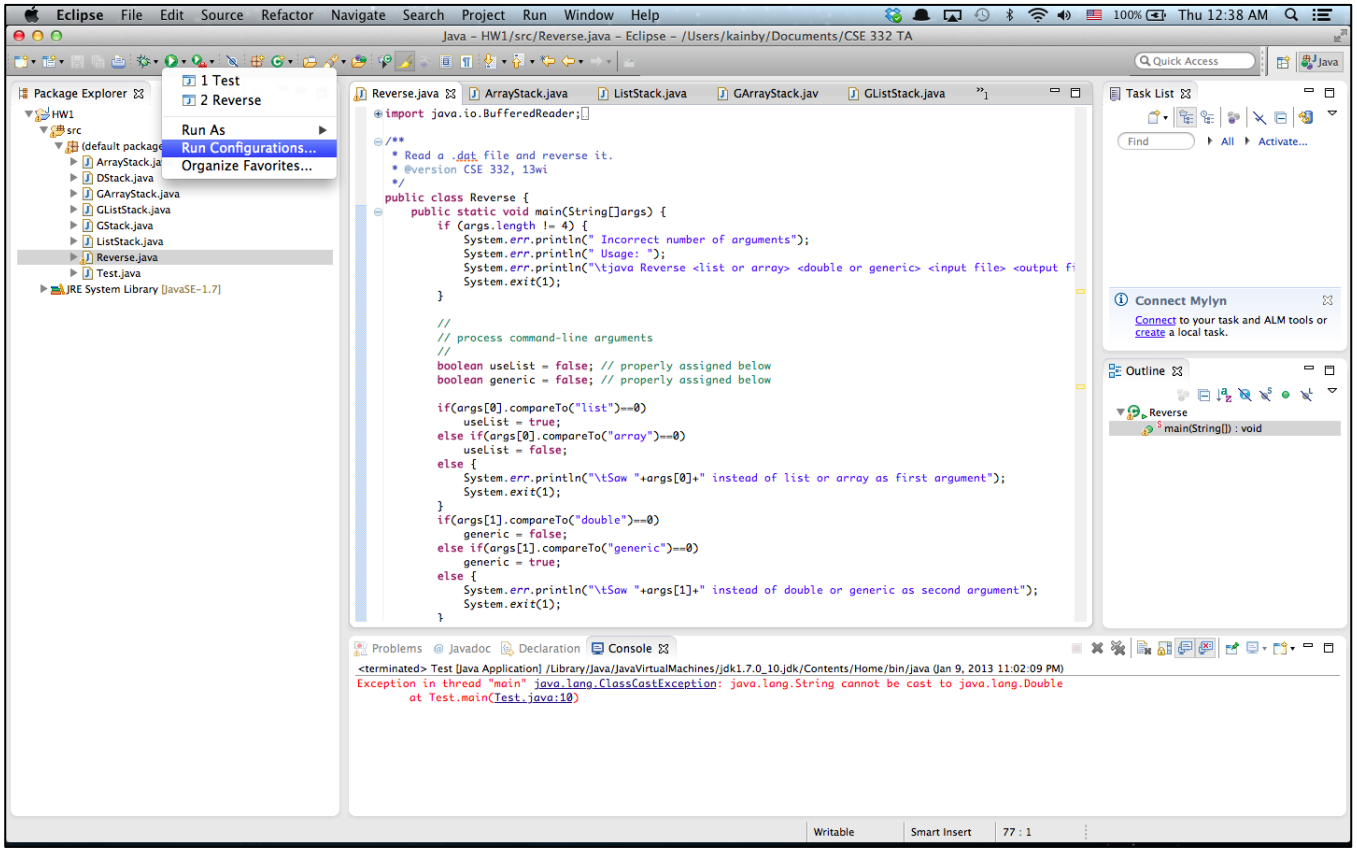
Eclipse Tutorial

- **Create Class**



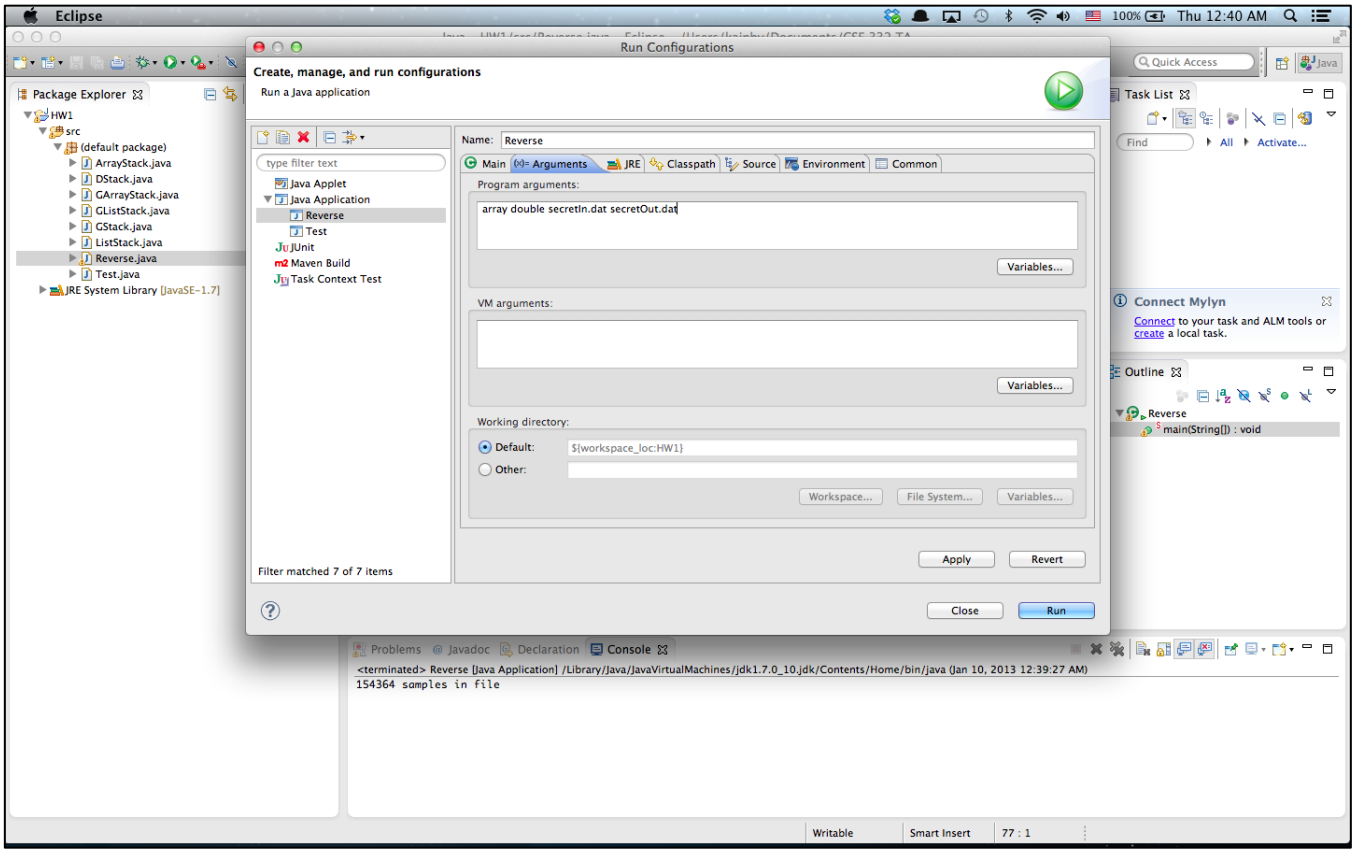
Eclipse Tutorial

- **Run Configuration (Command line Args)**



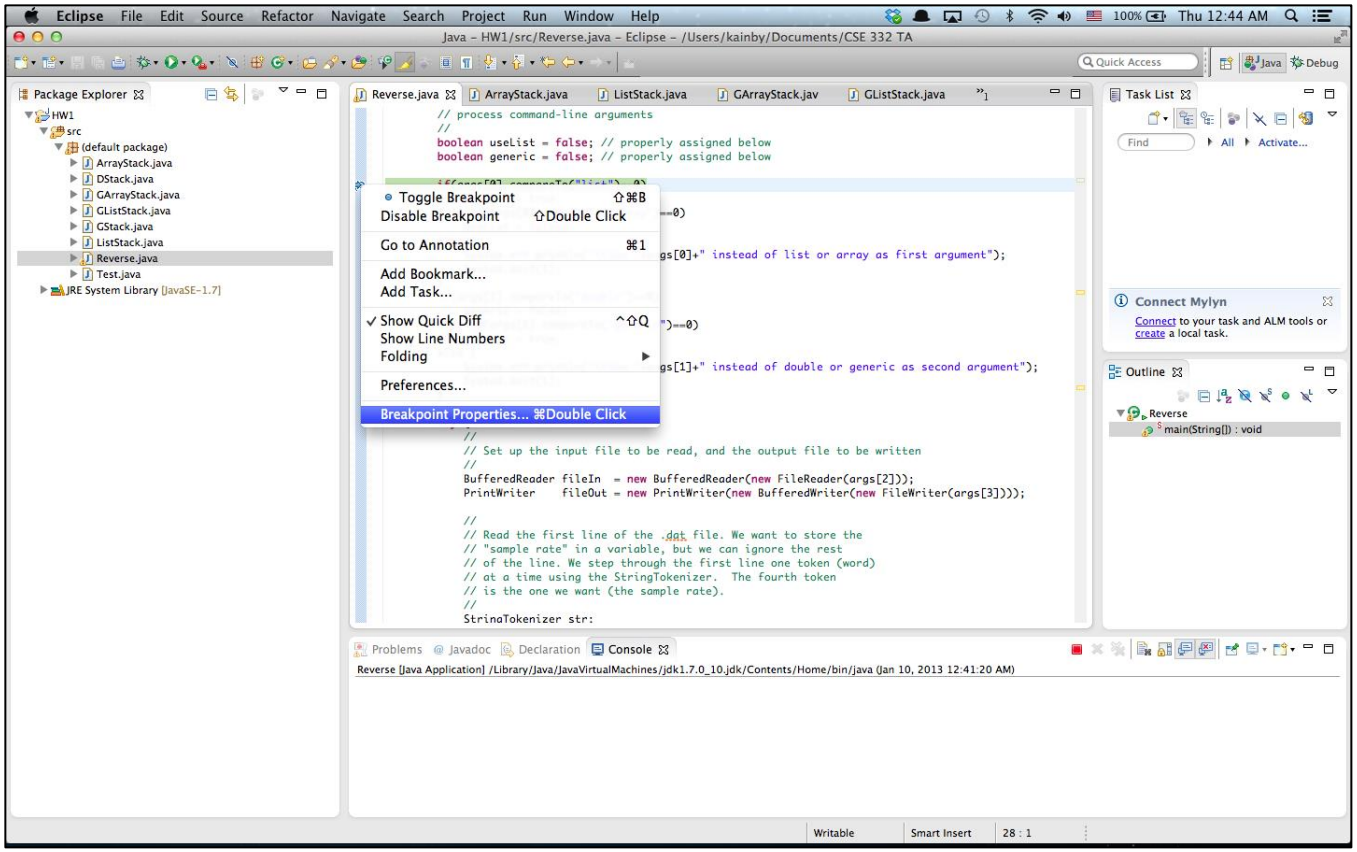
Eclipse Tutorial

- **Run Configuration (Command line Args)**



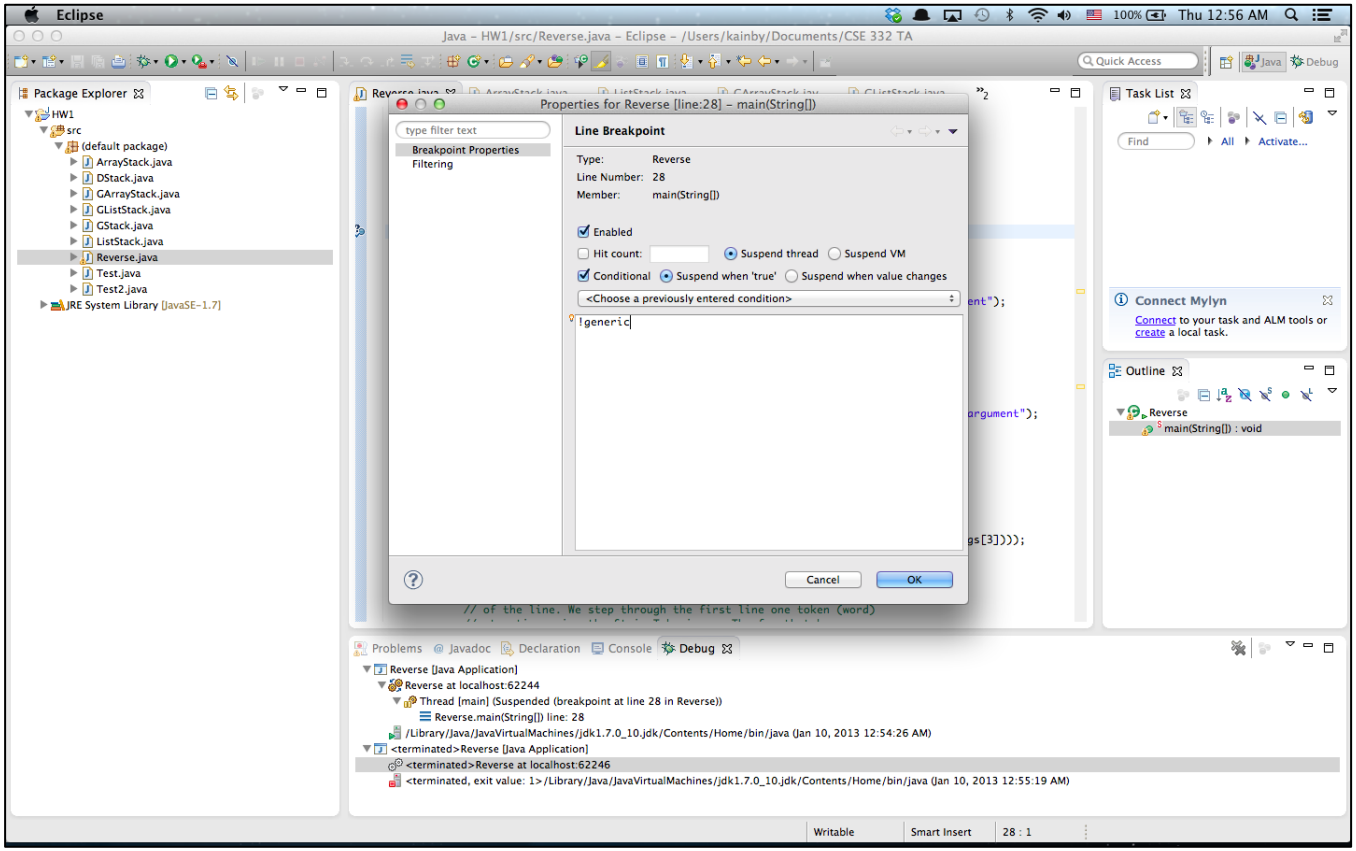
Eclipse Tutorial

- **Conditional Debugging**



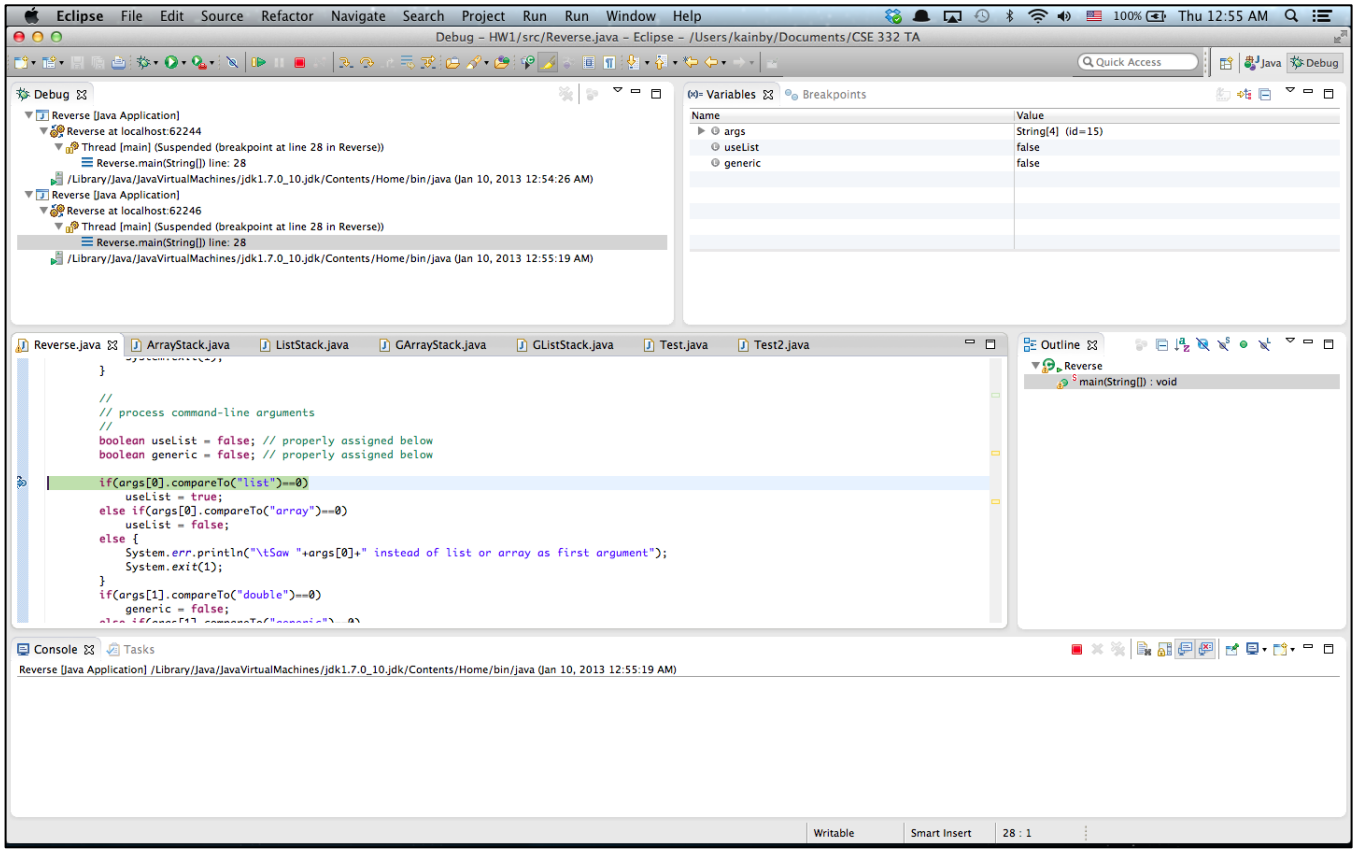
Eclipse Tutorial

- **Conditional Debugging**



Eclipse Tutorial

- **Conditional Debugging**



Eclipse Tutorial

- **More Tutorials**

  - Written Tutorial

    http://www.vogella.com/articles/Eclipse/article.html

  - Video Tutorial

    http://eclipsetutorial.sourceforge.net/totalbeginner.html

  - Eclipse Shortkeys

    http://www.rossenstoyanchev.org/write/prog/eclipse/eclipse3.html