

CSSE 220 Day 3

API Documentation, Unit Tests, and
Object References

Check out *JavadocsAndUnitTesting* from SVN

Questions?



Java Documentation

- » API Documentation, Docs in Eclipse, Writing your own Docs

Java API Documentation

- ▶ What's an API?

- Application Programming Interface

You need the 6 to get the current version of Java

- ▶ The Java API on-line

- Google for: **java api documentation 6**
- Or go to: <http://java.sun.com/javase/6/docs/api/>
- Also hopefully on your computer at

C:\Program Files\Java\jdk1.6.0_14\docs\api\index.html

- ▶ Find the documentation for the **String** class from one of the above links, as follows:

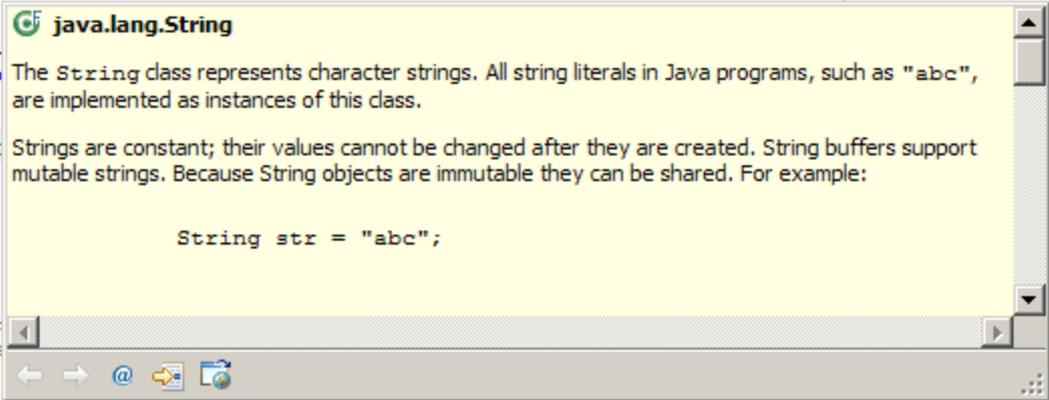
- Click **java.lang** in the top-left pane
- Then click **String** in the bottom-left pane

Alternative: Google for **java 6 String**

Java Documentation in Eclipse

- ▶ Setting up Java API documentation in Eclipse
 - Should be done already, but if the next steps don't work for you, we'll fix that
- ▶ Using the API documentation in Eclipse
 - Hover text
 - Open external documentation (Shift-F2)

```
main(String[] args) {  
    is a  
    JOpti  
    atln(  
    VG: Pr  
    ring:  
    Werldi  
    e's e
```



java.lang.String

The `String` class represents character strings. All string literals in Java programs, such as `"abc"`, are implemented as instances of this class.

Strings are constant; their values cannot be changed after they are created. String buffers support mutable strings. Because String objects are immutable they can be shared. For example:

```
String str = "abc";
```

Navigation icons: back, forward, search, external documentation (highlighted by a red arrow), and refresh.

Writing Javadocs

- ▶ Written in special comments: `/** ... */`
- ▶ Can come before:
 - Class declarations
 - Field declarations
 - Constructor declarations
 - Method declarations
- ▶ Eclipse is your friend!
 - It will generate Javadoc comments automatically
 - It will notice when you start typing a Javadoc comment

Example Javadoc for a Class

```
/**  
 * A Ball acts on its own (for example,  
 * moving on the screen) and can be moved  
 * externally, selected or killed.  
 *  
 * @author Curt Clifton.  
 * Created Sep 9, 2008.  
 */  
public class Ball { ... }
```

Description of
class

@author Tag
followed by author
name and date

Example Javadoc for a Method

Description of method, usually starts with a verb. Generally should say what the method accomplishes, NOT how it does so. BTW, saying “This is a method that ...” is not helpful; the reader knows that it is a method.

```
/**  
 * Returns the original String converted  
 * to a String representing shouting.  
 * Does not change the original String.  
 *  
 * @param input the original string  
 * @return input in ALL UPPER CASE  
 */  
public static String shout(String input) {  
    return input.toUpperCase();  
}
```

@param tag followed by parameter name and (optional) description. Repeat for each parameter.

@return tag followed by description of result. Omit for *void* methods.

Exercise

»» Add javadoc comments to
MoreWordGames

- Use Quick Fix!
(click on light bulb)

Javadocs: Key Points

- ▶ Don't try to memorize the Java libraries
 - Nearly 9000 classes and packages!
 - You'll learn them over time
- ▶ Get in the habit of writing the javadocs **before** implementing the methods
 - It will help you **think before doing**, a vital software development skill
 - This is called programming with ***documented stubs***
 - I'll try to model this. If I don't, call me on it!

Writing Code to Test Your Code

- »» Test-driven Development,
unit testing and JUnit

Unit Testing

▶ From Wikipedia:

“Unit testing is a software verification and validation method in which a programmer tests if individual units of source code are fit for use.

- A unit is the smallest testable part of an application.
 - ❖ In procedural programming a unit may be an individual function or procedure
 - ❖ [and in object oriented programming, a unit may be a method or a class].”

▶ Hence *writing code to test other code*

- Focused on testing individual pieces of code (units) in isolation
 - Individual methods
 - Individual classes

▶ Why would software engineers do unit testing?

Unit Testing With JUnit

- ▶ JUnit is a unit testing *framework*
 - A *framework* is a collection of classes to be used in another program.
 - Does much of the work for us!
- ▶ JUnit was written by
 - Erich Gamma
 - Kent Beck
- ▶ Open-source software
- ▶ Now used by **millions** of Java developers

JUnit Example

- ▶ `MoveTester` in Big Java shows how to write tests in plain Java
- ▶ Look at `JUnitMoveTester` in today's repository
 - Shows the same test in JUnit
 - Let's look at the comments and code together...

Interesting Tests

Important Slide: Use this as a reference!

- ▶ Test “boundary conditions”
 - Intersection points: $-40^{\circ}\text{C} == -40^{\circ}\text{F}$
 - Zero values: $0^{\circ}\text{C} == 32^{\circ}\text{F}$
 - Empty strings
- ▶ Test known values: $100^{\circ}\text{C} == 212^{\circ}\text{F}$
 - But not too many
- ▶ Tests things that might go wrong
 - Unexpected user input: “zero” when 0 is expected
- ▶ Vary things that are “important” to the code
 - String length if method depends on it
 - String case if method manipulates that

Exercise



Walk through creating unit tests for shout in MoreWordGames

- Name your test case class MoreWordGamesTest

Then write unit tests for whisper and holleWerld.

Remember, the goal is to write tests that cover “interesting” cases.

Object References

- » Differences between primitive types and object types in Java

What Do Variables Really Store?

- ▶ Variables of number type store *values*
- ▶ Variables of class type store *references*

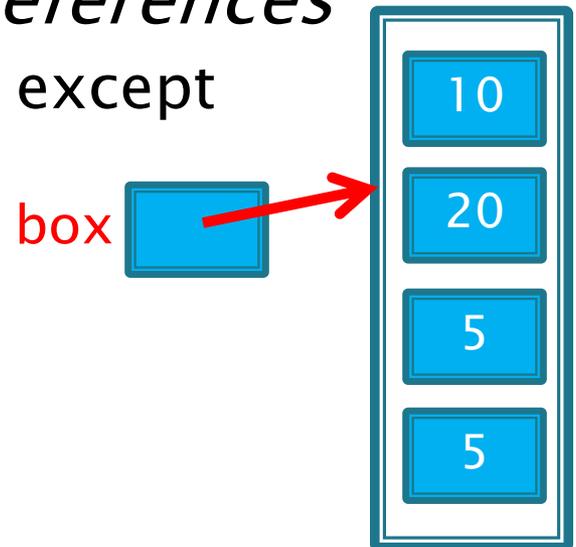
- A reference is like a pointer in C, except
 - Java keeps us from screwing up
 - No `&` and `*` to worry about (and the people say, “Amen”)

- ▶ Consider:

1. `int x = 10;`

2. `int y = 20;`

3. `Rectangle box = new Rectangle(x, y, 5, 5);`



Assignment Copies Values

- ▶ Actual value for number types
- ▶ **Reference** value for object types
 - The actual **object is not copied**
 - The **reference value** (“the pointer”) **is copied**

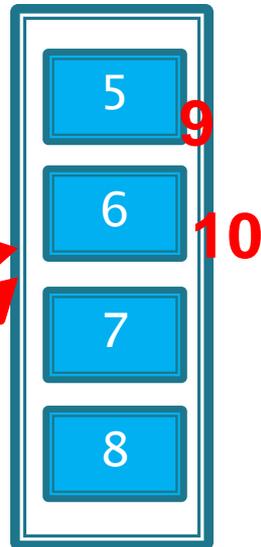
▶ Consider:

1. `int x = 10;`



2. `int y = x;`

3. `y = 20;`



4. `Rectangle box = new Rectangle(5, 6, 7, 8);`

5. `Rectangle box2 = box;`

6. `box2.translate(4, 4);`

Exercise

- »» Begin the Written Exercise from Homework 3