

CSSE 220

Objects

Check out *SuperSimpleObjects* and *TeamGradebook* from SVN

Plan for today

- Introduce how to write your own classes
- Talk about object references and box and pointer diagrams
- Get started on TeamGradebook, your new assignment

Identifiers (Names) in Java

- The rules:
 - Start with letter or underscore (`_`)
 - Followed by letters, numbers, or underscores
- The conventions:
 - `variableNamesLikeThis`
 - `methodNameLikeThis (...)`
 - `ClassNamesLikeThis`
- You should follow the conventions!

Using Objects and Methods

“Who does what, with what?”

► Works just like Python:

- *object.method(argument, ...)*

Implicit
argument

Explicit
arguments

The dot notation is also used for *fields*

► Java Example:

```
String name = "Bob Forapples";  
PrintStream printer = System.out;
```

```
int nameLen = name.length();  
printer.printf("'%s' has %d characters", name, nameLen);
```

Class – What, When, Why, & How?

What:

- A blueprint for a custom **type**

When:

- Define a class when you're representing a concept (think nouns)
- When no other existing type can do what you want/need

Class – What, When, Why, & How?

Why:

- Keep similar concepts together
- Encapsulation (we'll get there in a bit)

How:

```
public class ClassName {  
    //fields  
    //methods  
}
```

Constructors – What, When, Why, How?

What:

- Special method called when a new instance of a class is created
- Initializes the new instance
- Like the `__init__` method in Python

When:

- Define a constructor when special initialization of a class is required
- Otherwise, Java implicitly creates a no-argument constructor if you don't add one

Constructors – What, When, Why, How?

Why:

- Allows you to ensure that a new instance of a class is a setup exactly how it needs to be before use of other methods/fields
- Puts it in a good state

How:

```
public class MyClass {  
    public MyClass() {  
        //initialization code  
    }  
    public MyClass(ParamType paramName) {  
        //initialization code  
    }  
}
```


new Keyword– What, When, Why, How?

What:

- Used to create a new instance of a class
- Calls the constructor in the class

When:

- Creating a new instance of a class
 - If the class definition is the blueprint for the house, a house that has been built is the “new instance” of the blueprint.

new Keyword– What, When, Why, How?

Why:

- To make a new instance

How:

- `MyClass instance = new MyClass();`
 - This will call the constructor with the matching parameters in `MyClass`
- Also used for arrays (as we've seen before):
 - `int[] arr = new int[5];`

Implementing classes

- Live coding with Bank Account object

Now code the StudentAssignments class yourself

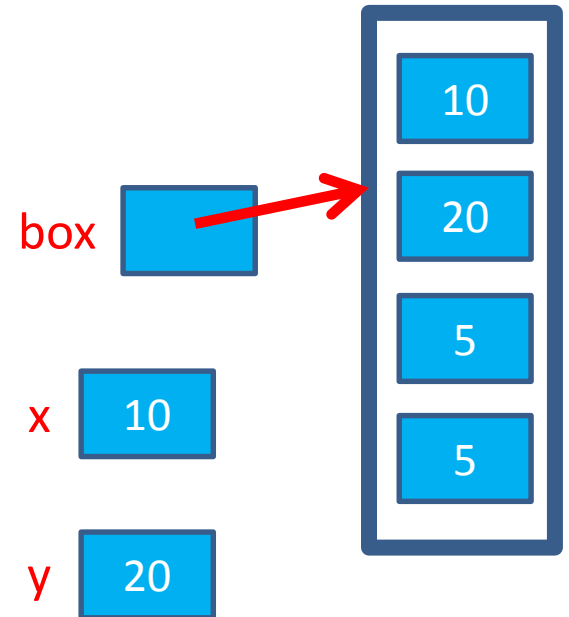
- Uncomment the stuff in StudentAssignmentsMain to see what the class ought to do
- Then create the class and add the constructors and methods you need
- If you finish early, add a function to compute the student's average grade

Differences between primitive types and object types in Java

OBJECT REFERENCES

What Do Variables Really Store?

- Variables of **primitive type** store *values*
- Variables of **class type** store *references*



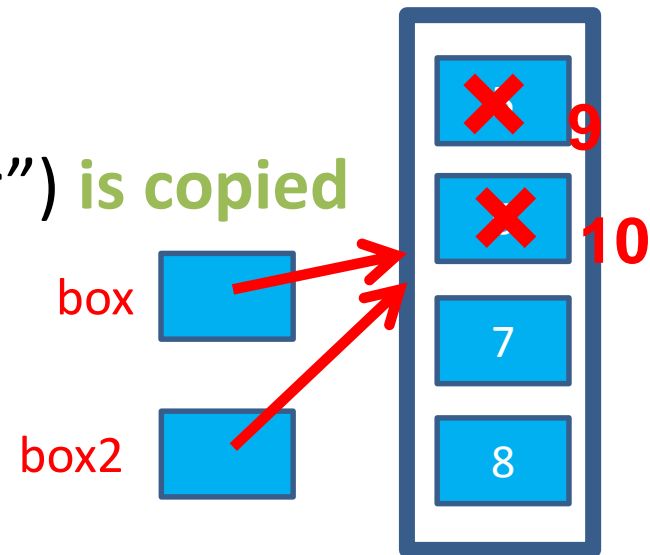
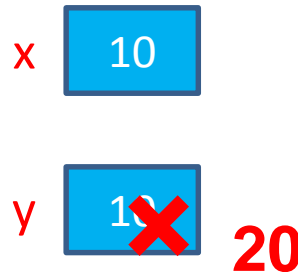
```
1. int x = 10;  
2. int y = 20;  
3. Rectangle box = new Rectangle(x, y, 5, 5);
```

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);
```

Boxes and lines exercise

Separating implementation details from how
an object is used

ENCAPSULATION

Encapsulation in Object-Oriented Software

- *Encapsulation*—separating implementation details from how an object is used
 - Client code sees a *black box* with a known *interface*

	Functions	Objects
Black box exposes	Function signature	Constructor and method signatures
Encapsulated inside the box	Operation implementation	<u>Data storage</u> and <u>operation implementation</u>

Start on TeamGradebook

- Try to finish the code for both add-student and get-names today
- If you are confused about what to do, get help!