

Recitation 2

Brandon Rozek
rozekb@rpi.edu

Rensselaer Polytechnic Institute, Troy, NY, USA

January 2022

Outline

Mainly two things:

- Review for Quiz on Friday
- Homework Help, etc.

As always, this recitation is for you. Therefore, feel free to steer this discussion.

Language Paradigm

Java is an object-oriented language

- Object-oriented is a style in which the majority of data is stored in objects along with the code to manipulate them.
- Fields are variables stored in an object.
- Methods are functions/operations stored in an object.

Variable Models

Variable models determine what an assignment to a variable performs:

- Value Model: Copy value from the r.h.s to the l.h.s.
- Reference Model: Make the l.h.s reference the r.h.s.

Java uses a *mixed* reference model for variables. Value model for primitives, and reference model for objects.

Practice:

- 1 Is String a primitive?
- 2 What are the primitives in Java?
- 3 Which keyword do you need to create an object in Java?

Stack vs Heap

- A stack is a data structure that is used to keep track of *primitives* and *references* associated with variables.
- A stack frame is used to associate variables with a particular context such as a function/method execution.

For example, declaring and assigning an `int` in a method will store it in its stack frame until the method exits.

Stack vs Heap

- A heap is a data structure that manages dynamic memory.
- In Java, objects are stored in the heap.
- Memory allocation for the heap needs to be requested from the operating system.

Luckily Java does this for you with the `new` operator. It even deallocates objects whose references are no longer in the stack for you! (Garbage Collection)

Equality Checking

There are two ways of checking equality:

- `a == b` checks if `a` and `b` refer to the same object in the heap.
- `a.equals(b)` executes `a`'s `equals` method either from its own class or the closest parent class that overwrote that method.
- If the class and none of its ancestors overwrote the `equals` method, then it acts the same as `==`.

Practice:

- 1 Where are references to objects stored? Stack or Heap?
- 2 How do we check if two strings are made up of the same sequence of characters?

Java Access Modifiers

	Class	Package	Subclass	World
public	Y	Y	Y	Y
protected	Y	Y	Y	N
No modifier	Y	Y	N	N
private	Y	N	N	N

Inheritance and Interfaces

- A class can only inherit a single class
- However, it can inherit multiple interfaces.

```
class Dog extends Animal implements Bark, Fetch {  
    public void bark() {  
        // ...  
    }  
    // ...  
}
```

Interfaces vs Abstract Classes

- Interfaces do not contain any implementation, only method signatures. They cannot be *instantiated*.
- Meanwhile, abstract classes can contain implementations for methods. However, the class itself cannot be *instantiated*.

Code Snippets

```
public abstract class Animal {  
    abstract boolean isAlive();  
}
```

```
interface Fetch {  
    public Frisbee fetch(double angle, double velocity);  
}
```

Mutability

Mutability denotes whether or not an object can change.

- An object which can change is called *mutable*.
- Otherwise if it can't change, then it is called *immutable*.
- Primitives are immutable.

Why have Immutable objects?

With multiple references to an object, it makes it so that others don't change the object from under you.

```
String greeting = "Hello";  
String typicalOutput = greeting + ", World";  
System.out.println(typicalOutput);  
System.out.println(greeting);
```

What's the output?

Practice:

Is the following code snippet possible? Why?

```
String greeting = "Hello";  
// Whoops, need it to be lowercase  
greeting[0] = 'h';
```

Are Java arrays mutable?

Strong vs Weak Typing

- Strong typing is generally associated with the language enforcing *type safety*. Often by requiring each variable and method to be annotated with a *type signature*.
- Weak typing is the opposite and generally gets conflated with implicit conversion. For example Python allows the expression `"=" * 5`.

Static vs Dynamic Typing

The main difference is where the majority of the type checks happen:

- For static typing this happens at compile time
- For dynamic typing this happens at runtime.

Practice:

- 1 Does Java use Strong or Weak Typing?
- 2 Does Java use Static or Dynamic Typing?

Any Questions?