

Objects

An Object is a “class” of thing (Can be as specific as you want)

- ▶ Furniture, Seat, Chair
- ▶ Vehicle, Motorized Vehicle, Car, Hybrid Car
- ▶ LifeForm, Animal, Human, Employee

Properties

Objects/Classes have properties

- ▶ Furniture *has* materials, dimensions
- ▶ Seat *has* materials, dimensions, armrestsPresent
- ▶ Vehicle *has* fuelType, isManned
- ▶ and Integer *has* a maximum value (MAX_INT)

Actions/Methods

An object or class can “do” stuff (actions)

- ▶ Animal → eats, sleeps
- ▶ HybridCar → starts, stops, changeGears, reportStatus
- ▶ LifeForms → fight, eat, moves, etc.
- ▶ String → can compare themselves to other Strings (equals)

Instances

An instance of a class is a specific “thing” that belongs to that class

- ▶ “my Prius” is an instance of “Hybrid Cars”
- ▶ “John” is an instance of Human
- ▶ “Hello World” is an instance of String
- ▶ `String a = new String();` is an instance of String

In Code

```
public class Car // class/Object
{
    public int fuelType; // property

    public static void printProperties() //method.
    {
        System.out.println(fuelType);
    }
}
```

Instantiating A class

```
Car c = new Car();
```

Code

- ▶ Creating a class (Car)
- ▶ Put some properties (member variables)
- ▶ Put some methods (NOT STATIC!!! yay!)
- ▶ Instanciate your class in **main**
- ▶ Save your class with the same name .java

Using this

- ▶ `this` represents the current instance of a class
- ▶ add `this` to the `Car` class

static vs. instance

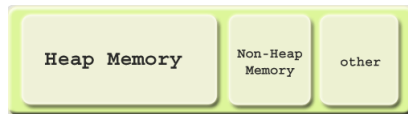
- ▶ `static` is shared among all instances of a class
- ▶ `non static`'s scope is only the current instance
- ▶ Example: counting Cars.

Constructor

- ▶ Initialize variables
- ▶ Can have many constructors

A word on object pointers

JVM divides memory:



- ▶ All objects reside in memory
 - ▶ Instance variables reside in Heap space
 - ▶ Methods and meta data reside in non-heap
- ▶ JVM stuff resides in Other

A note on == and !=

- ▶ == and != compare object locations or primitive values
- ▶ `byte`, `short`, `int`, `long`, `float`, `double`, `boolean`, `char` are **datatype primitives** and are set to a default value (not location) when declared;
- ▶ An object's default value (a.k. location) is `null`

Methods to compare

- ▶ can you implement `equals` for a `Car` class?

Classes with a life of their own

- ▶ can you implement DOPs and DUPs as classes with their own `move`, `fight`, `eat`, `reproduce` behavior?