ICS 141 Programming with Objects

Jessica Maistrovich

Metropolitan State University

Class – Part 3

Review parts of a “basic” Class

- variables to hold the current state of the object
- constructor(s) to help build the object
- getters/setters as needed to access information about the object’s current state
- toString to describe the object in a way humans understand
- equals to decide if two objects of the same type are equal

toString()

toString

- What happens when you print a primitive data type? What happens when you try to print an object?
- Remember that the name of object is a reference
- What do you want it to print when you are printing the object?
  - If you had to write one sentence to describe your thing, what would it say?
  - Design decision

```
public String toString(){
    String output = "";
    output += "The manager of store number " + storeNumber;
    output += " is " + managerName;
    return output;
}
```

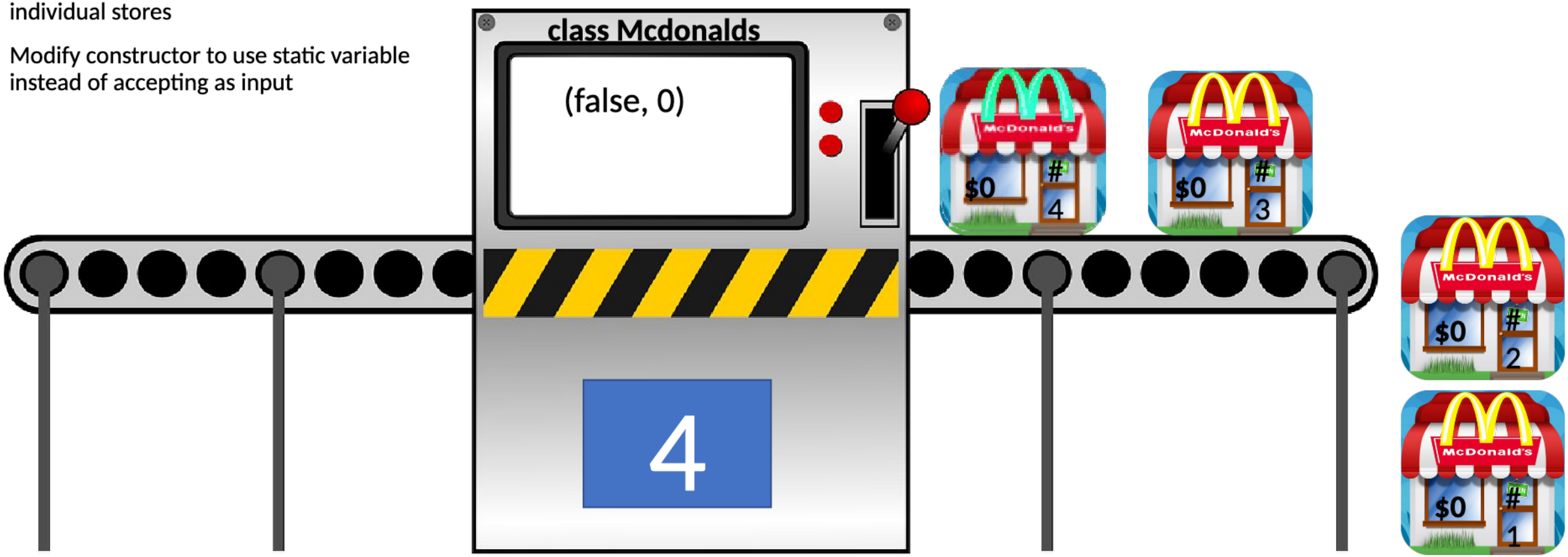
Try it!

- Create a toString method for your Dog class. What sentence will you use to describe the dog?

static modifier

Static

- Want to keep track of number of stores (for store number)
- Belongs to the machine, not the individual stores
- Modify constructor to use static variable instead of accepting as input



Static Method Calls vs Method Calls

- Static methods belong to the machine, so have to ask the machine (class)
- Methods belong to individual objects, so have to ask an individual object
- Example:

```
double randomNumber;
```

```
randomNumber = Math.random()
        vs
Random rand = new Random()
randomNumber = rand.nextDouble()
```

```java
public class McDonalds {
    private boolean hasGoldenArches;
    private int storeNumber;
    private double annualEarnings;
    private String managerName;
    private static int numStores = 0;

    public McDonalds(double earnings) {
        hasGoldenArches = true;
        numStores = numStores + 1;
        storeNumber = numStores;
        annualEarnings = earnings;
        managerName = new String("Jess");
    }

    public McDonalds(boolean golden, double earnings, String name) {
        hasGoldenArches = golden;
        storeNumber = ++numStores;
        annualEarnings = earnings;
        managerName = name;
    }
```

Try it!

- Add a static counter to the dog class.
- Create an variable to hold the dog’s id.
- Update the constructor(s) to use the static variable to assign a unique id to every dog that is created.

equals

How do you know if two things are equal?

- Depends on what type of object you are talking about:
  - Primitive data types
    - 2, ‘a’,
    - double (iffy)
    - ==
  - Objects:
    - McDonalds
    - Dog
    - String
    - .equals(Object other)
- Programmer gets to decide what makes two of their objects equal
  - One of the objects is always the thing that called the method, the other is sent as a parameter.
  - When the call is made, essentially asking “am I equal to this other thing”
    - Returns true if yes
    - Returns false if no

McDonalds had a unique id – storeNumber

```
public class McDonalds {
    private boolean hasGoldenArches;
    private int storeNumber;
    private double annualEarnings;
    private static int numStores = 0;


    public boolean equals(McDonalds other){
        return this.storeNumber == other.storeNumber;
    }
```

Try it!

- Add an equals method to the dog class. Two dogs should be considered equal if they have the same id.