Factory Pattern

Russell Ngo

Description

- Create objects in Java around a template
- Objects are made from a factory
- Factory calls concrete class
- Concrete class is made from a template
- Template is called an interface

Interface Requirements

- Is public
  - e.g.

```
public interface Car {}
```

- Needs all functions and variables concrete classes will share
  - e.g.

```
public interface Car {
    public int fuel = 0;
    string fuelCount();
    void refuel();
}
```

Concrete Class Requirements

- Needs “Implements ‘insert_interface_name’ ” when declaring class
  - e.g.

```
public class BMW implements Car {}
```

- Needs “@Override” to change properties of interface functions/variables
  - e.g.

```
@Override
public void refuel() {
    fuel += 100;
    System.out.println(“car refueled”)
}
```

- Any other functions/variables that are class specific
  - e.g. circle and triangle are both shapes, circle has diameter, triangle has base and height

Conclusion

- Factory creates objects using concrete classes
- Concrete classes are formed around an interface
- An interface holds the most common properties of the classes
- Concrete classes can have exclusive variables and/or functions