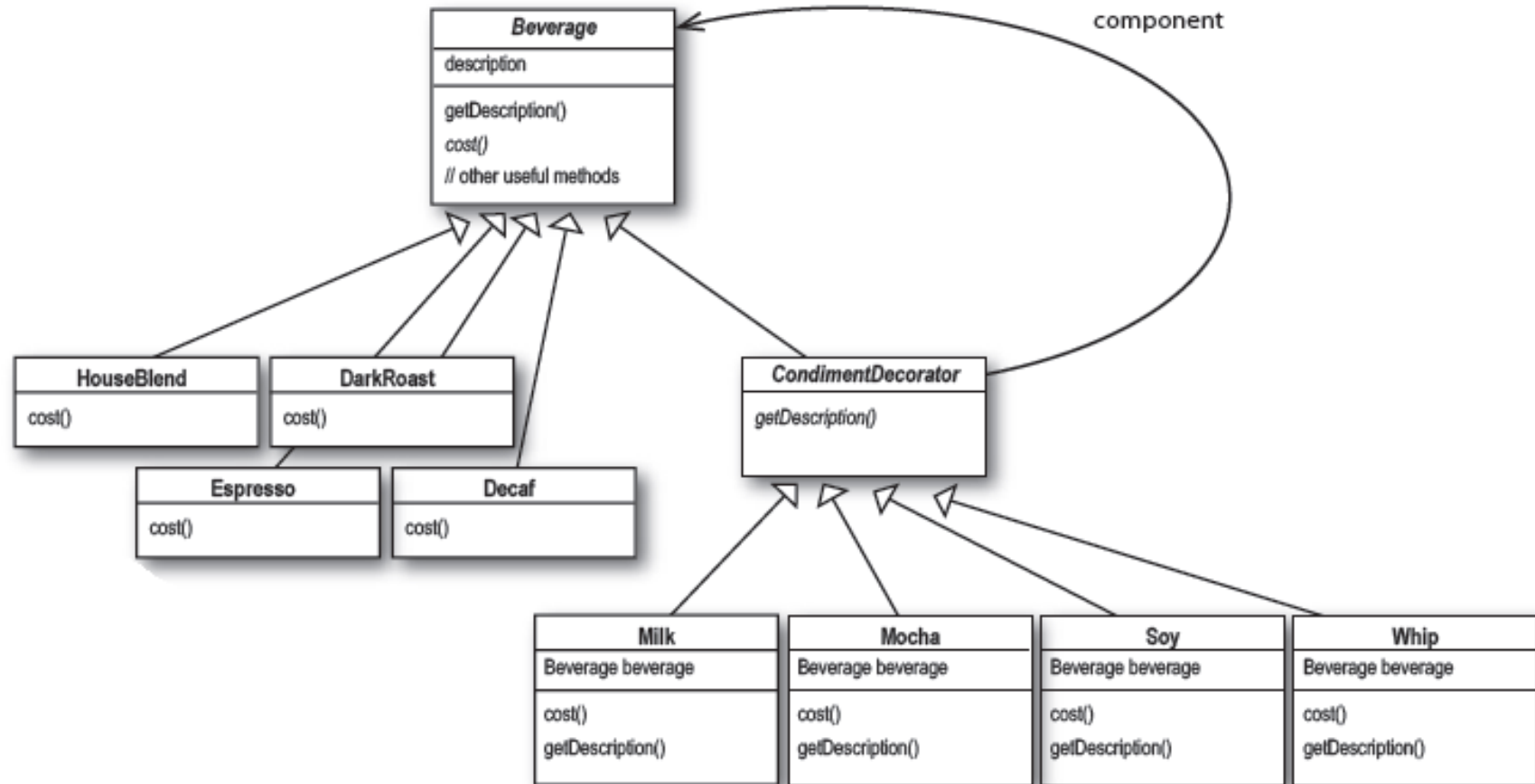


iCoffee Decorator Pattern



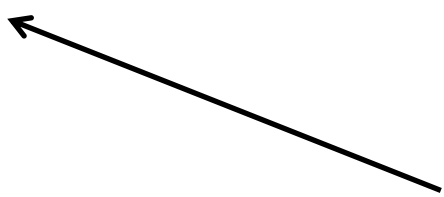
Code iCoffee

```
public abstract class Beverage {  
    String description = "Unknown Beverage";  
  
    public String getDescription() {  
        return description;  
    }  
  
    public abstract double cost();  
}
```

Code iCoffee - DECORATOR

```
public abstract class Beverage {  
    String description = "Unknown Beverage";  
  
    public String getDescription() {  
        return description;  
    }  
  
    public abstract double cost();  
}
```

```
public abstract class CondimentDecorator extends Beverage {  
    public abstract String getDescription();  
}
```



Code iCoffee– CONCRETE Component Espresso

```
public abstract class Beverage {  
    String description = "Unknown Beverage";  
  
    public String getDescription() {  
        return description;  
    }  
  
    public abstract double cost();  
}
```

```
public class Espresso extends Beverage {  
  
    public Espresso() {  
        description = "Espresso";  
    }  
  
    public double cost() {  
        return 1.99;  
    }  
}
```

```
public abstract class CondimentDecorator extends Beverage {  
    public abstract String getDescription();  
}
```

Code iCoffee– CONCRETE Component HouseBlend

```
public abstract class Beverage {  
    String description = "Unknown Beverage";  
  
    public String getDescription() {  
        return description;  
    }  
  
    public abstract double cost();  
}
```

```
public class HouseBlend extends Beverage {  
    public HouseBlend() {  
        description = "House Blend Coffee";  
    }  
  
    public double cost() {  
        return .89;  
    }  
}
```

```
public abstract class CondimentDecorator extends Beverage {  
    public abstract String getDescription();  
}
```

Code iCoffee– CONCRETE Component DarkRoast

```
public abstract class Beverage {  
    String description = "Unknown Beverage";  
  
    public String getDescription() {  
        return description;  
    }  
  
    public abstract double cost();  
}
```

```
public class DarkRoast extends Beverage {  
    public DarkRoast() {  
        description = "Dark Roast Coffee";  
    }  
  
    public double cost() {  
        return .99;  
    }  
}
```

```
public abstract class CondimentDecorator extends Beverage {  
    public abstract String getDescription();  
}
```

Code iCoffee– CONCRETE Component

Decaf

```
public abstract class Beverage {  
    String description = "Unknown Beverage";  
  
    public String getDescription() {  
        return description;  
    }  
  
    public abstract double cost();  
}
```

```
public class Decaf extends Beverage {  
    public Decaf() {  
        description = "Decaf Coffee";  
    }  
  
    public double cost() {  
        return 1.05;  
    }  
}
```

```
public abstract class CondimentDecorator extends Beverage {  
    public abstract String getDescription();  
}
```

Code iCoffee– DECORATOR Mocha

```
public abstract class Beverage {  
    String description = "Unknown Beverage";  
  
    public String getDescription() {  
        return description;  
    }  
  
    public abstract double cost();  
}
```

```
public class Decaf extends Beverage {  
    public Decaf() {  
        description = "Decaf Coffee";  
    }  
  
    public double cost() {  
        return 1.05;  
    }  
}
```

```
public abstract class CondimentDecorator extends Beverage {  
    public abstract String getDescription();  
}  
  
public class Mocha extends CondimentDecorator {  
    Beverage beverage;  
  
    public Mocha(Beverage beverage) {  
        this.beverage = beverage;  
    }  
  
    public String getDescription() {  
        return beverage.getDescription() + ", Mocha";  
    }  
  
    public double cost() {  
        return .20 + beverage.cost();  
    }  
}
```


Code iCoffee– DECORATOR Whip

```
public abstract class Beverage {  
    String description = "Unknown Beverage";  
  
    public String getDescription() {  
        return description;  
    }  
  
    public abstract double cost();  
}
```

```
public class Decaf extends Beverage {  
    public Decaf() {  
        description = "Decaf Coffee";  
    }  
  
    public double cost() {  
        return 1.05;  
    }  
}
```

```
public abstract class CondimentDecorator extends Beverage {  
    public abstract String getDescription();  
}
```

```
public class Whip extends CondimentDecorator {  
    Beverage beverage;  
  
    public Whip(Beverage beverage) {  
        this.beverage = beverage;  
    }  
  
    public String getDescription() {  
        return beverage.getDescription() + ", Whip";  
    }  
  
    public double cost() {  
        return .10 + beverage.cost();  
    }  
}
```

Code iCoffee– DECORATOR Soy

```
public abstract class Beverage {  
    String description = "Unknown Beverage";  
  
    public String getDescription() {  
        return description;  
    }  
  
    public abstract double cost();  
}
```

```
public class Decaf extends Beverage {  
    public Decaf() {  
        description = "Decaf Coffee";  
    }  
  
    public double cost() {  
        return 1.05;  
    }  
}
```

```
public abstract class CondimentDecorator extends Beverage {  
    public abstract String getDescription();  
}  
  
public class Soy extends CondimentDecorator {  
    Beverage beverage;  
  
    public Soy(Beverage beverage) {  
        this.beverage = beverage;  
    }  
  
    public String getDescription() {  
        return beverage.getDescription() + ", Soy";  
    }  
  
    public double cost() {  
        return .15 + beverage.cost();  
    }  
}
```

Test iCoffee

```
public class StarbuzzCoffee {  
  
    public static void main(String args[]) {  
        Beverage beverage = new Espresso();  
        System.out.println(beverage.getDescription()  
                            + " $" + beverage.cost());  
  
        Beverage beverage2 = new DarkRoast();  
        beverage2 = new Mocha(beverage2);  
        beverage2 = new Mocha(beverage2);  
        beverage2 = new Whip(beverage2);  
        System.out.println(beverage2.getDescription()  
                            + " $" + beverage2.cost());  
  
        Beverage beverage3 = new HouseBlend();  
        beverage3 = new Soy(beverage3);  
        beverage3 = new Mocha(beverage3);  
        beverage3 = new Whip(beverage3);  
        System.out.println(beverage3.getDescription()  
                            + " $" + beverage3.cost());  
    }  
}
```

```
Espresso $1.99  
Dark Roast Coffee, Mocha, Mocha, Whip $1.49  
House Blend Coffee, Soy, Mocha, Whip $1.34  
%
```

Homework

- The company now has sizes for coffees: small, medium and large,
 - The cost of the ingredients varies from their current value
 - Small does not vary
 - Medium increases 50%
 - Large increases 100%
 - Nota that the abstract class is going to require set and get methods for size