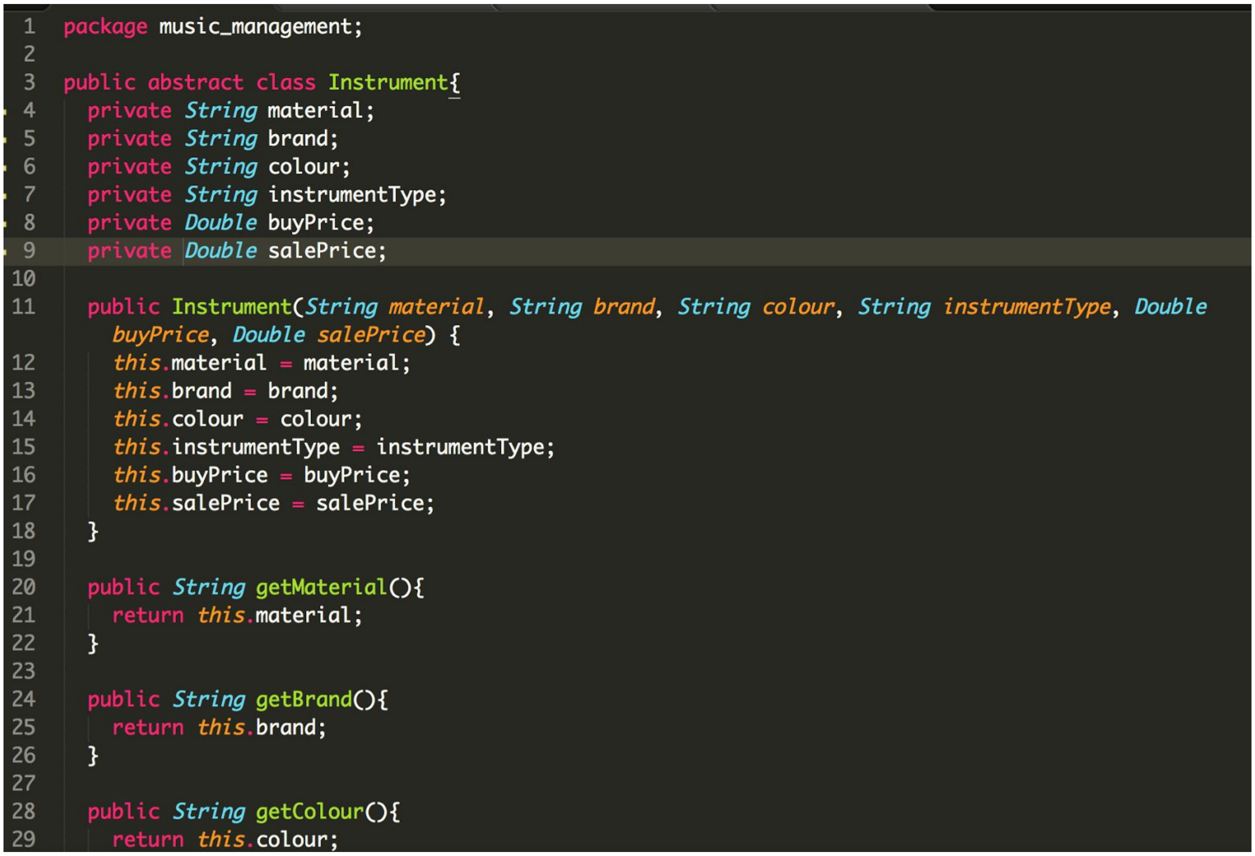
Stacey Napier PDA Evidence

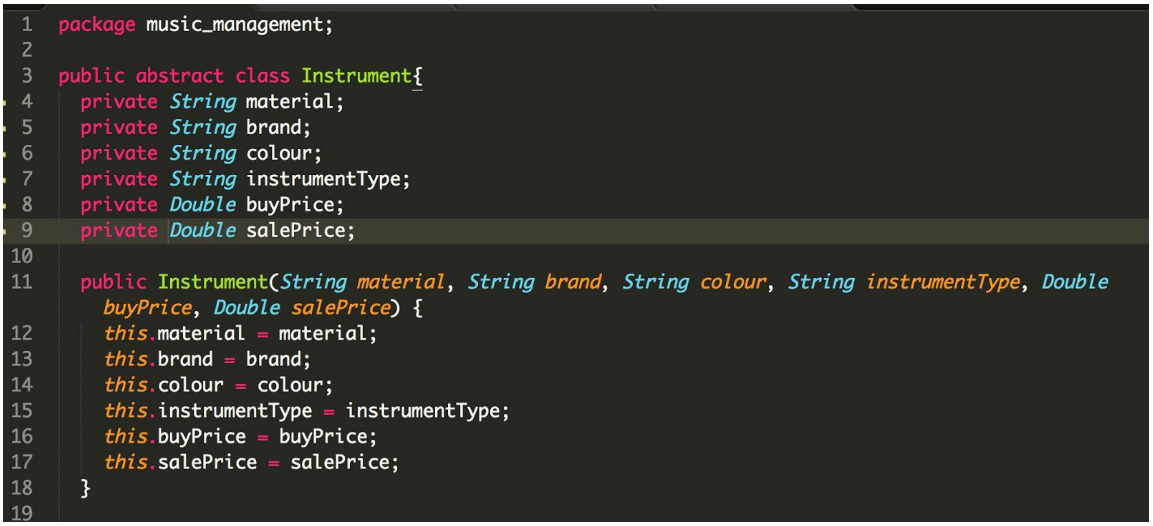
I.T 1 - Encapsulation



```java
package music_management;

public abstract class Instrument{
  private String material;
  private String brand;
  private String colour;
  private String instrumentType;
  private Double buyPrice;
  private Double salePrice;

  public Instrument(String material, String brand, String colour, String instrumentType, Double buyPrice, Double salePrice) {
    this.material = material;
    this.brand = brand;
    this.colour = colour;
    this.instrumentType = instrumentType;
    this.buyPrice = buyPrice;
    this.salePrice = salePrice;
  }

  public String getMaterial(){
    return this.material;
  }

  public String getBrand(){
    return this.brand;
  }

  public String getColour(){
    return this.colour;
```

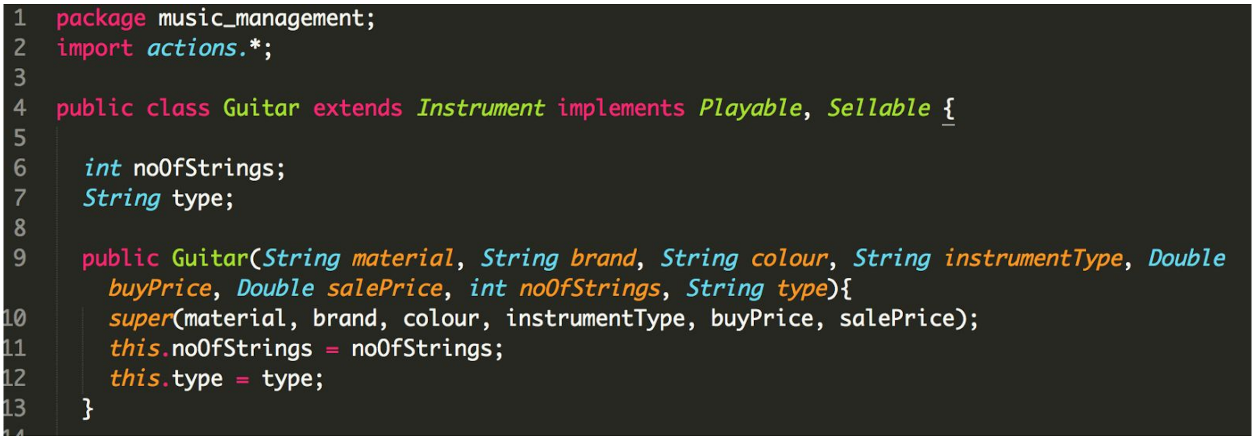
I.T 2 - Use of inheritance

- A Class



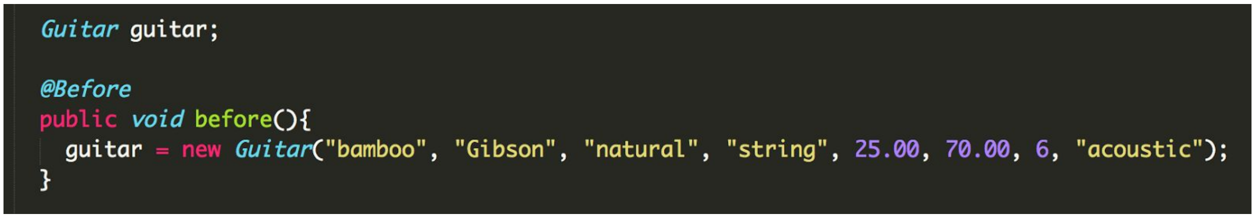
```java
package music_management;

public abstract class Instrument{
  private String material;
  private String brand;
  private String colour;
  private String instrumentType;
  private Double buyPrice;
  private Double salePrice;

  public Instrument(String material, String brand, String colour, String instrumentType, Double buyPrice, Double salePrice) {
    this.material = material;
    this.brand = brand;
    this.colour = colour;
    this.instrumentType = instrumentType;
    this.buyPrice = buyPrice;
    this.salePrice = salePrice;
  }

```

- A class that inherits from the previous class



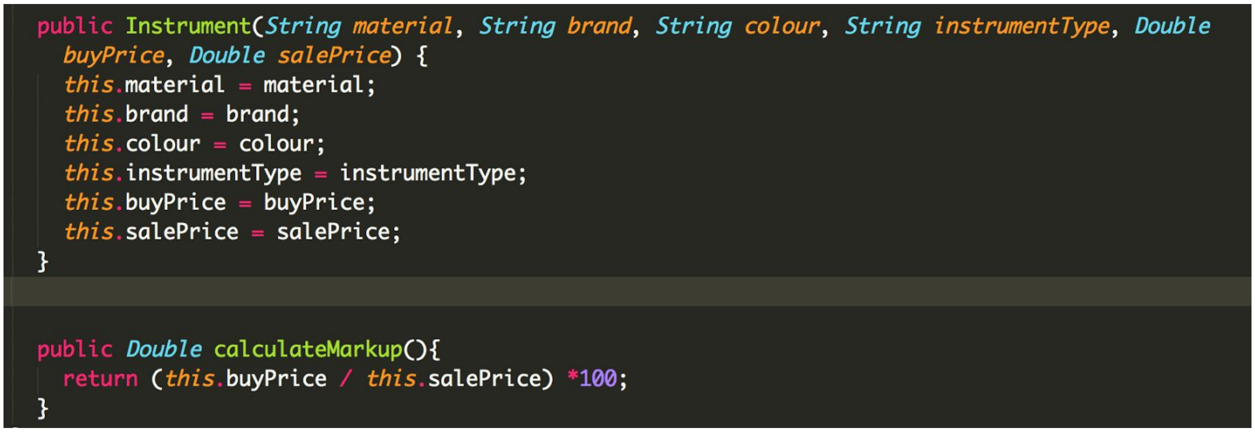
```java
package music_management;
import actions.*;

public class Guitar extends Instrument implements Playable, Sellable {

  int noOfStrings;
  String type;

  public Guitar(String material, String brand, String colour, String instrumentType, Double
    buyPrice, Double salePrice, int noOfStrings, String type){
    super(material, brand, colour, instrumentType, buyPrice, salePrice);
    this.noOfStrings = noOfStrings;
    this.type = type;
  }
```

- An object of the inherited class



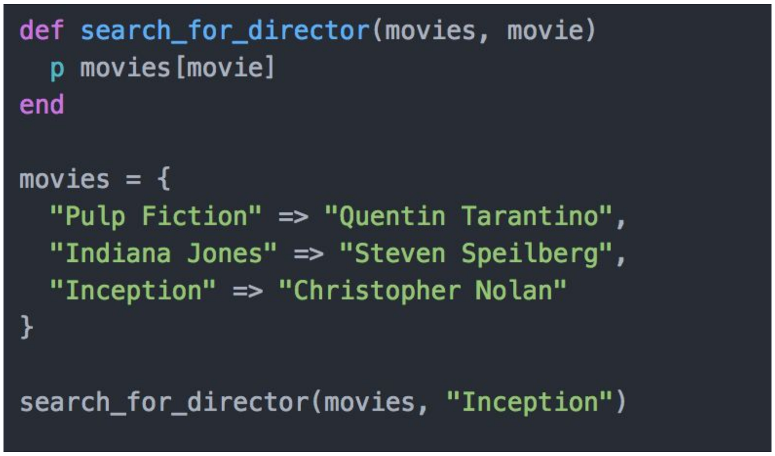
```java
Guitar guitar;

@Before
public void before(){
  guitar = new Guitar("bamboo", "Gibson", "natural", "string", 25.00, 70.00, 6, "acoustic");
}
```

- A method that uses the information inherited from another class

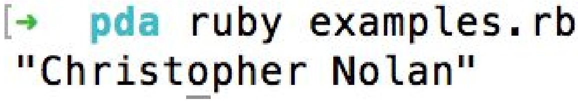


```java
public Instrument(String material, String brand, String colour, String instrumentType, Double
  buyPrice, Double salePrice) {
  this.material = material;
  this.brand = brand;
  this.colour = colour;
  this.instrumentType = instrumentType;
  this.buyPrice = buyPrice;
  this.salePrice = salePrice;
}


public Double calculateMarkup(){
  return (this.buyPrice / this.salePrice) *100;
}
```

I.T 3 Searching data

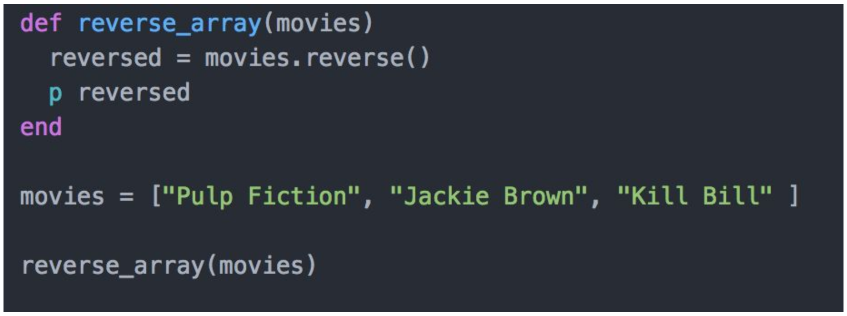


```ruby
def search_for_director(movies, movie)
  p movies[movie]
end

movies = {
  "Pulp Fiction" => "Quentin Tarantino",
  "Indiana Jones" => "Steven Speilberg",
  "Inception" => "Christopher Nolan"
}

search_for_director(movies, "Inception")
```

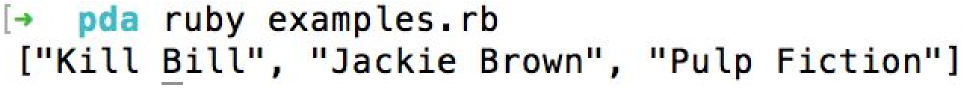


```
[→  pda ruby examples.rb
"Christopher Nolan"
```

I.T 4 Sorting data

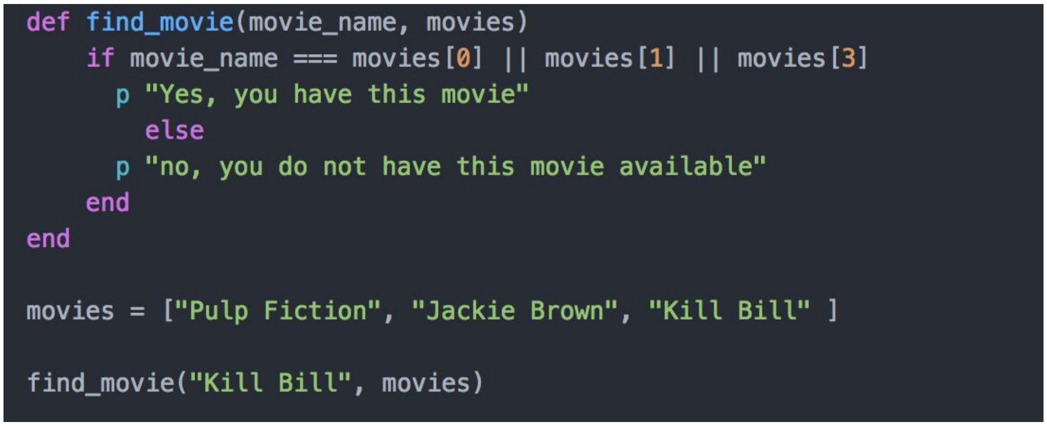


```ruby
def reverse_array(movies)
  reversed = movies.reverse()
  p reversed
end

movies = ["Pulp Fiction", "Jackie Brown", "Kill Bill" ]

reverse_array(movies)
```

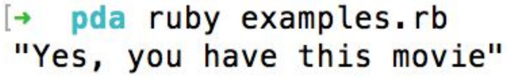


```
[→  pda ruby examples.rb
["Kill Bill", "Jackie Brown", "Pulp Fiction"]
```

I.T 5 Array

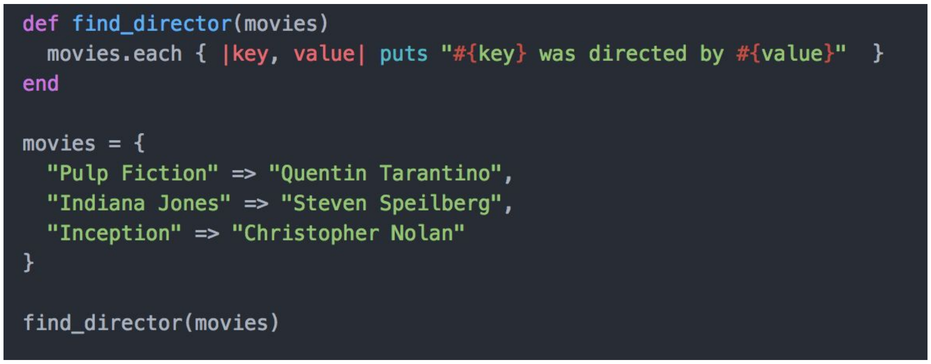


```ruby
def find_movie(movie_name, movies)
    if movie_name === movies[0] || movies[1] || movies[3]
      p "Yes, you have this movie"
        else
      p "no, you do not have this movie available"
    end
end

movies = ["Pulp Fiction", "Jackie Brown", "Kill Bill" ]

find_movie("Kill Bill", movies)
```

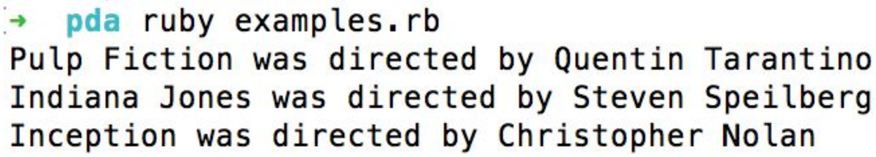


```
[→  pda ruby examples.rb
"Yes, you have this movie"
```

I.T 6 Hash

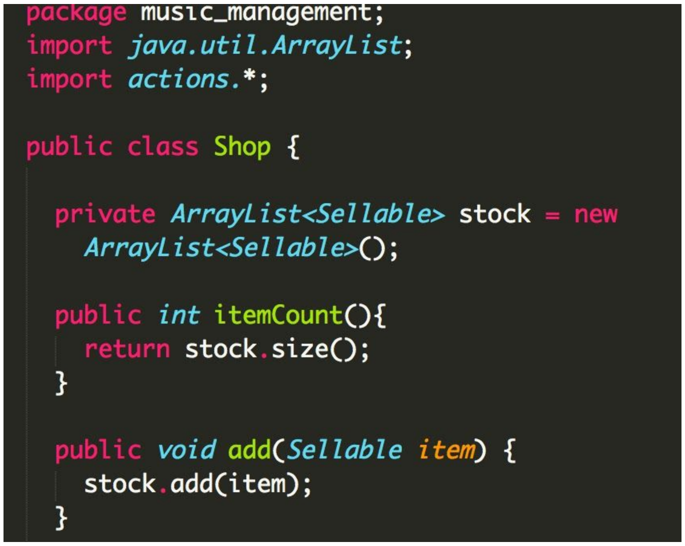


```ruby
def find_director(movies)
  movies.each { |key, value| puts "#{key} was directed by #{value}"  }
end

movies = {
  "Pulp Fiction" => "Quentin Tarantino",
  "Indiana Jones" => "Steven Speilberg",
  "Inception" => "Christopher Nolan"
}

find_director(movies)
```

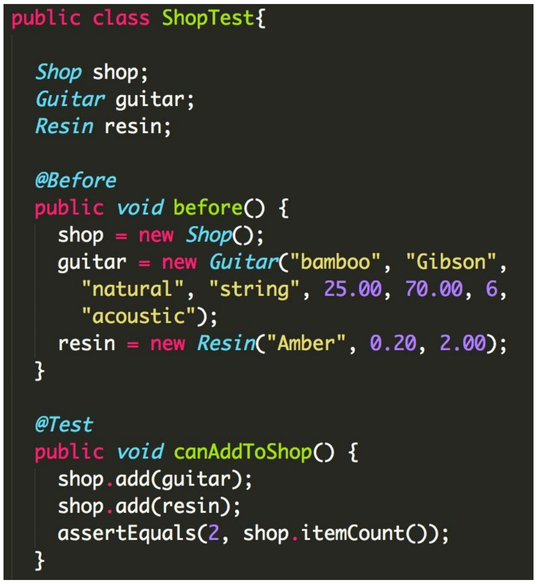


```
→  pda ruby examples.rb
Pulp Fiction was directed by Quentin Tarantino
Indiana Jones was directed by Steven Speilberg
Inception was directed by Christopher Nolan
```

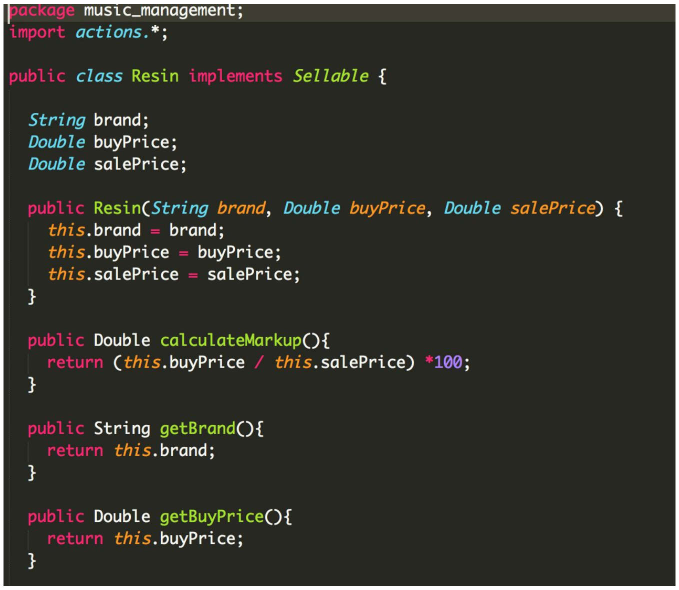
IT 7 Polymorphism



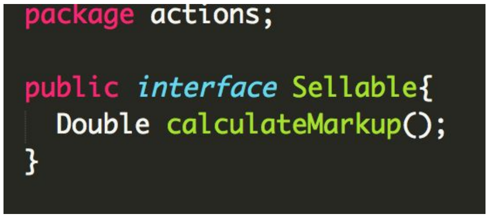
```java
package music_management;
import java.util.ArrayList;
import actions.*;

public class Shop {

  private ArrayList<Sellable> stock = new
    ArrayList<Sellable>();

  public int itemCount(){
    return stock.size();
  }

  public void add(Sellable item) {
    stock.add(item);
  }
```



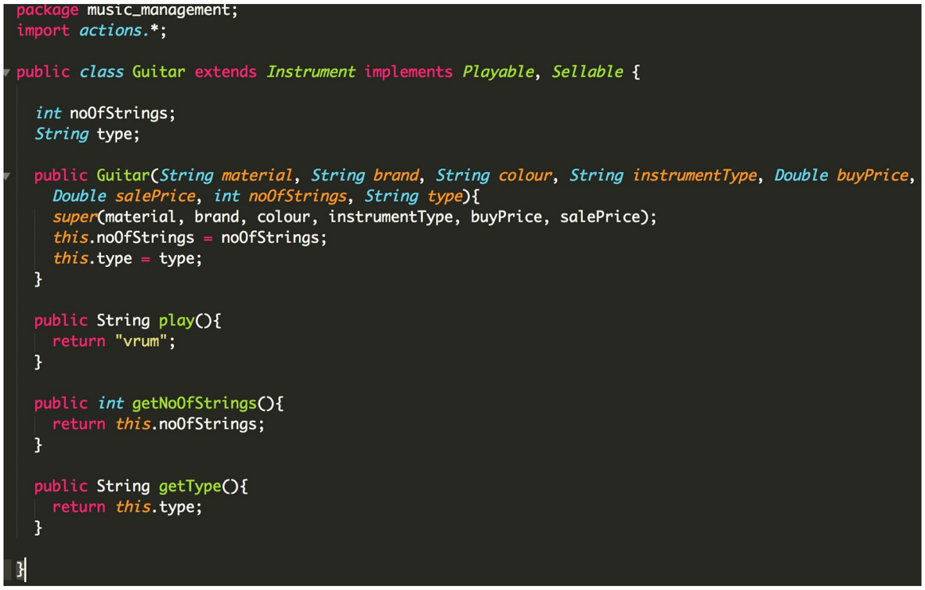
```java
public class ShopTest{

  Shop shop;
  Guitar guitar;
  Resin resin;

  @Before
  public void before() {
    shop = new Shop();
    guitar = new Guitar("bamboo", "Gibson",
      "natural", "string", 25.00, 70.00, 6,
      "acoustic");
    resin = new Resin("Amber", 0.20, 2.00);
  }

  @Test
  public void canAddToShop() {
    shop.add(guitar);
    shop.add(resin);
    assertEquals(2, shop.itemCount());
  }
```



```java
package music_management;
import actions.*;

public class Resin implements Sellable {

  String brand;
  Double buyPrice;
  Double salePrice;

  public Resin(String brand, Double buyPrice, Double salePrice) {
    this.brand = brand;
    this.buyPrice = buyPrice;
    this.salePrice = salePrice;
  }

  public Double calculateMarkup(){
    return (this.buyPrice / this.salePrice) *100;
  }

  public String getBrand(){
    return this.brand;
  }

  public Double getBuyPrice(){
    return this.buyPrice;
  }
```



```java
package actions;

public interface Sellable{
  Double calculateMarkup();
}
```



```java
package music_management;
import actions.*;

public class Guitar extends Instrument implements Playable, Sellable {

  int noOfStrings;
  String type;

  public Guitar(String material, String brand, String colour, String instrumentType, Double buyPrice,
    Double salePrice, int noOfStrings, String type){
    super(material, brand, colour, instrumentType, buyPrice, salePrice);
    this.noOfStrings = noOfStrings;
    this.type = type;
  }

  public String play(){
    return "vrum";
  }

  public int getNoOfStrings(){
    return this.noOfStrings;
  }

  public String getType(){
    return this.type;
  }

}
```