PDA: Software Development

Implementation and Testing
Level 8

Iain Paterson
Cohort E15

I.T 1

Screenshot of Encapsulation

```java
public abstract class Competitors {

    private NationalCountry country;
    private ArrayList<Medal> medals;
    private SportPlayed sport;

    public Competitors(NationalCountry country, SportPlayed sport) {
        this.country = country;
        this.sport = sport;
        this.medals = new ArrayList<>();
    }

    public NationalCountry getCountry() { return country; }

    public ArrayList<Medal> getMedals() { return medals; }

    public SportPlayed getSport() { return sport; }

    public void addMedal(Medal medal) { this.medals.add(medal); }
}
```

I.T 2

Screenshot of the use of Inheritance

```java
package com.example.user.raysmusicstore;


public abstract class Instrument {

    private String brand;
    private String colour;
    private String family;


    public Instrument(String brand, String colour, String family) {
        this.brand = brand;
        this.colour = colour;
        this.family = family;
    }

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

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

    public String getFamily(){
        return this.family;
    }

}

```

A class that inherits from the previous class

```java
package com.example.user.raysmusicstore;


public class KettleDrum extends Instrument implements Playable  {

     String skinType;

    public KettleDrum( String brand, String colour,String family, String skinType ){
        super(brand, colour, family);
        this.skinType = skinType;
    }

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

    public String getSkin() {
        return skinType;
    }


}
```

An object in the inherited class

```java
public class KettleDrumTest {

    KettleDrum kettleDrum;

    @Before
    public void before(){
        kettleDrum = new KettleDrum( "Timpani", "Bronze", "Percussion", "Cow Hide" );
    }

    @Test
    public void testKettleDrumSkin() { assertEquals( "Cow Hide", kettleDrum.getSkin()); }

    @Test
    public void testKettleDrumMakesNoise() { assertEquals( "BOOM BOOM BOOM", kettleDrum.play()); }

    @Test
    public void testKettleDrumColour() { assertEquals( "Bronze", kettleDrum.getColour()); }

    @Test
    public void testKettleDrumBrand(){
        assertEquals( "Timpani", kettleDrum.getBrand());
    }

    @Test
    public void testKettleDrumFamily() { assertEquals( "Percussion", kettleDrum.getFamily()); }


}
```

A method that uses the information from the inherited class

```java
    @Test
    public void testKettleDrumBrand(){
        assertEquals( "Timpani", kettleDrum.getBrand());
    }
```

```
Run KettleDrumTest.testKettleDrumBrand
1 test passed - 1ms
KettleDrumTest (com.exampl 1ms
    testKettleDrumBrand 1ms
"/Applications/Android Studio.app/Contents/jre/jdk/Contents/Home/bin/java" ...

Process finished with exit code 0
```

I.T 3

Demonstration of searching data in a program

```ruby
def self.find( id )
  sql = "SELECT * FROM animals
  WHERE id = $1;"
  values = [id]
  found = SqlRunner.run( sql, values )
  return found.map{ |animal_type| Animal.new( animal_type )}[0]
end
```

Searched for data

```
[1] pry(main)> Animal.find(13)
=> #<Animal:0x007fad2236d310
 @age=4,
 @date_entered="2017-05-12",
 @id=13,
 @name="Rover",
 @photo="dog.jpg",
 @type="dog",
 @vet_id=5>
```

```
[2] pry(main)> Animal.find(16)
=> #<Animal:0x007fad23099e58
 @age=7,
 @date_entered="2017-04-09",
 @id=16,
 @name="Willow",
 @photo="cat.jpg",
 @type="cat",
 @vet_id=6>
```

I.T 4

Demonstration of sorting data in a program

Array to be sorted

```
movie_titles = [
    'Matrix',
    'Stand By Me',
    'Conspiracy Theory',
    'Avatar',
    'Robocop',
    'Beverly Hills Cop'
]
```

Function that uses the array

```ruby
def get_movies(array)
  return array.sort
end

puts get_movies(movie_titles)
```

Sorted array in alphabetical order

```
psql                    ruby
[→  pda_example ruby array_sort.rb
Avatar
Beverly Hills Cop
Conspiracy Theory
Matrix
Robocop
Stand By Me
```

I.T 5

Demonstrate the use of a Hash

```
i_t_5.rb                                i_t_6.rb
teams_stadiums = { :rangers => 'Ibrox',
                   :juventus => 'Juventus Stadium',
                   :manchester_United => 'Old Trafford',
                   :bayern_Munich => 'Allianz Arena',
                   :barcelona => 'Camp Nou',
                   :benfica => 'Estadio da Luz'}
```

Function that uses hash

```
def rangers_stadium(hash)
  return hash[:rangers]
end


puts rangers_stadium(teams_stadiums)
```

Result of function

```
ruby    ruby app...    psql
[→  week_2 ruby i_t_6.rb
Ibrox
→  week_2
```

I.T 6

Demonstration the use of an Array

```
i_t_5.rb                                i_t_6.rb
football_teams = [ 'Rangers', 'Juventus', 'Manchester United',
                   'Bayern Munich', 'Barcelona', 'Benfica' ]
```

Function that uses array

```ruby
def get_team(array)
 return array.reverse
end


puts get_team(football_teams)
```

Result of function

```
ruby    ruby app...    psql
[→  week_2 ruby i_t_5.rb
Benfica
Barcelona
Bayern Munich
Manchester United
Juventus
Rangers
→  week_2
```

I.T 7

The use of polymorphism

The Guitar Class which implements the Playable interface and play method

```java
package com.example.user.raysmusicstore;


public class Guitar extends Instrument implements Playable {

    int numberOfStrings;

    public Guitar( String brand, String colour, String family, int numberOfStrings){
        super(brand, colour, family);
        this.numberOfStrings = numberOfStrings;
    }

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

    public int getStrings() { return numberOfStrings; }

}

```

The playable interface with the play method

```java
package com.example.user.raysmusicstore;



public interface Playable {

    String play();
}
```

The play method being tested in the guitar test

```java
package com.example.user.raysmusicstore;

import static org.junit.Assert.*;
import org.junit.*;

public class GuitarTest {

    Guitar guitar;

    @Before
    public void before() { guitar = new Guitar( "Fender", "Red", "Percussion",  6 ); }

    @Test
    public void numberOfStings() { assertEquals( 6, guitar.getStrings()); }

    @Test
    public void testGuitarMakesSound(){
        assertEquals( "Strum Strum Strum", guitar.play());
    }

    @Test
    public void testGuitarColour() { assertEquals( "Red", guitar.getColour()); }

    @Test
    public void testGuitarBrand() { assertEquals( "Fender", guitar.getBrand()); }

    @Test
    public void testGuitarFamily() { assertEquals( "Percussion", guitar.getFamily()); }


}
```

Result of the play method being tested

```java
package com.example.user.raysmusicstore;

import static org.junit.Assert.*;
import org.junit.*;

public class GuitarTest {

    Guitar guitar;

    @Before
    public void before() { guitar = new Guitar( "Fender", "Red", "Percussion",  6 ); }

    @Test
    public void numberOfStings() { assertEquals( 6, guitar.getStrings()); }

    @Test
    public void testGuitarMakesSound(){
        assertEquals( "Strum Strum Strum", guitar.play());
    }
```

Run GuitarTest.testGuitarMakesSound

1 test passed – 1ms

GuitarTest (com.example.use 1ms

testGuitarMakesSound 1ms

```
"/Applications/Android Studio.app/Contents/jre/jdk/Contents/Home/bin/java" ...

Process finished with exit code 0
```