

Basic tests comparisons

In this section we provide a comparison of the different libraries on an instance of basic tests made on a class

Presentation of the class

We used the `Dwarf` class to make our tests :

This class has the following fields : `String name;` `private double size;` `private double weight;` `private boolean hasBeard;` `private boolean hasHangover;` `SongRepository knownSongs`

`SongRepository` is a helper class used to encapsulate a `Collection`. Its methods for modification are package protected and usable only by the `dwarf` whereas all methods that do not modify the collection are public.

▼ *Show Dwarf source code*

```
package org.examples.Dwarf;
import java.util.Random;

public class Dwarf {
    private String name;
    private double size;
    private double weight;
    private boolean hasBeard;
    private boolean hasHangover;

    public String getName() {
        return name;
    }

    public double getSize() {
        return size;
    }

    public void setSize(double size) {
        this.size = size;
    }

    public double getWeight() {
        return weight;
    }

    public void setWeight(double weight) {
```

```

        this.weight = weight;
    }

    public void shave() {
        this.hasBeard = false;
    }

    public void growBeard() {
        this.hasBeard = true;
    }

    public boolean isHungover() {
        return hasHangover;
    }

    private SongRepository knownSongs = new SongRepository();

    public Dwarf(String name, double size, double weight, boolean hasBeard) {
        this.name = name;
        this.size = size;
        this.weight = weight;
        this.hasBeard = hasBeard;
        this.hasHangover = false;
    }

    public void learnSong(String newSong) {
        this.knownSongs.add(newSong);
    }

    public void sleep() {
        this.hasHangover = false;
    }

    private String drink() {
        Random rand = new Random();
        int i = rand.nextInt(this.knownSongs.size() );
        this.hasHangover = true;
        return this.knownSongs.remove(i);
    }

    private void drinkWithoutKnownSongs() {
        this.hasHangover = true;
    }

    public boolean isBearded() {
        return this.hasBeard;
    }

    private void sing(String song) {
        System.out.println(song);
    }

```

```

public void goesToTavern() {
    if(knownSongs.isEmpty())
    {
        drinkWithoutKnownSongs();
    }else //if is not empty
    {
        sing(this.drink());
    }
}

public boolean isKnown(String song) {
    return this.knownSongs.contains(song);
}

public SongRepository getLearnedSongs() {
    return this.knownSongs;
}
}

```

▼ *Show SongRepository source code*

```

package org.examples.Dwarf;

import java.util.ArrayList;
public class SongRepository {

    private ArrayList<String> songs = new ArrayList<>();

    void add (String song) {
        this.songs.add(song);
    }

    public boolean contains (String song) {
        return this.songs.contains(song);
    }

    String remove (int i) {
        return songs.remove(i);
    }

    public int size () {
        return songs.size();
    }

    public boolean isEmpty(){return songs.isEmpty();}
}

```

Comparisons of the tests

testConstructor

This test is to ensure that the constructor properly sets the object's fields

Field	Apache	AssertJ	Guava	Truth	Atlanmod
Name	Validate.isTrue (Objects.equals (dwarf.getName(), "Jeremy"));	Column 3, row 1	Column 4, row 1	Column 5, row 1	Column 6, row 1
Size	Validate.isTrue (dwarf.getSize() == 80.4);	Column 3, row 2	Column 4, row 2	Column 5, row 2	Column 6, row 2
Weight	Validate.isTrue (dwarf.getWeight() == 90.3);	Column 3, row 3	Column 4, row 3	Column 5, row 3	Column 6, row 3
hasBeard	Validate.isTrue (!dwarf.isBearded());	Column 3, row 4	Column 4, row 4	Column 5, row 4	Column 6, row 4

testSize

This test ensures that `getSize()` and `setSize()` works as intended

testWeight

This test ensures that `getWeight()` and `setWeight()` works as intended

testBeard

This test ensures that `shave()`, `growBeard()` and `isBearded()` works as intended

testHungover

This test ensures that the hungover field is properly set by the dwarf going to the tavern and sleeping

testSongs

This test ensures that the learned songs get updated when a dwarf learns a song & goes to the tavern (which causes him to forget a song and sing it). This also ensures that the song is properly printed when sang.

Conclusion