jambit
WHERE INNOVATION WORKS

ToiletPaper #5

RandomBeans

By Gernot Pointner

✖ Problem

Writing setup code for tests involving **complex POJOs** is cumbersome and **error prone**.

This often results in **shared test setup code** used by several unit tests, which in turn leads to **strong coupling** between several tests classes. In addition, it is almost **impossible to create all** possible POJO **combinations** and run into all **corner cases**.

✔ Solution

Use the **RandomBeans** library to create random instances for your test cases.

It provides a **fluent API** for specifying value ranges, fixed values, or a set of values for your created test instances. It also allows for the creation of **more complex POJOs** (e.g. recursive object graphs). Using RandomBeans, one can **create different test instances for every test run.** If a test fails, it is possible to replay the exact same test scenario by using the same seed.

Some **possible use cases** include:

- Populating a test DB with random entities
- Generating random request or responses for your RESTful API
- Creating random input for your validators

➔ Example

```
 // example User POJO – setters/getters omitted for the sake of brevity
public class User {
       public String name;
       public List<String> hobbies;
       public LocalDate birthday;
       public URL website;
}
// configure a EnhancedRandom for creating User instances
// * generate UTF-8 String with 10 to 15 characters
// * created LocalDates are within a range from 50 years ago till now
// * collections should contain from 1 up to 20 elements
// * the field “website” of our User should have a valid URL
EnhancedRandom er = EnhancedRandomBuilder.aNewEnhancedRandomBuilder()
  .charset(Charset.forName("UTF-8"))
  .stringLengthRange(10, 15)
  .dateRange(now().minusYears(50), now())
  .collectionSizeRange(1, 20)
  .randomize(field().named("website").ofType(URL.class).inClass(User.class).get(),
     UrlRandomizer.aNewUrlRandomizer())
  .build();

// create test object using our previously configured EnhancedRandom
User user = er.nextObject(User.class);
```

✚ Links

- https://github.com/benas/random-beans