

Fake Data Library

Table of Contents

Screenshots	2
Fixtures and Integration Tests	4
How to configure/use	7
Classpath	7
Bootstrapping	7
API and Implementation	8
Known issues	10
Dependencies	11

This module (`isis-module-fakedata`) provides a domain service that generates fake random data. The random values generated can then be used within unit and integration tests.

The module consists of a single domain service `FakeDataDomainService`. This can be injected into fixtures and integration tests just like any other domain service.

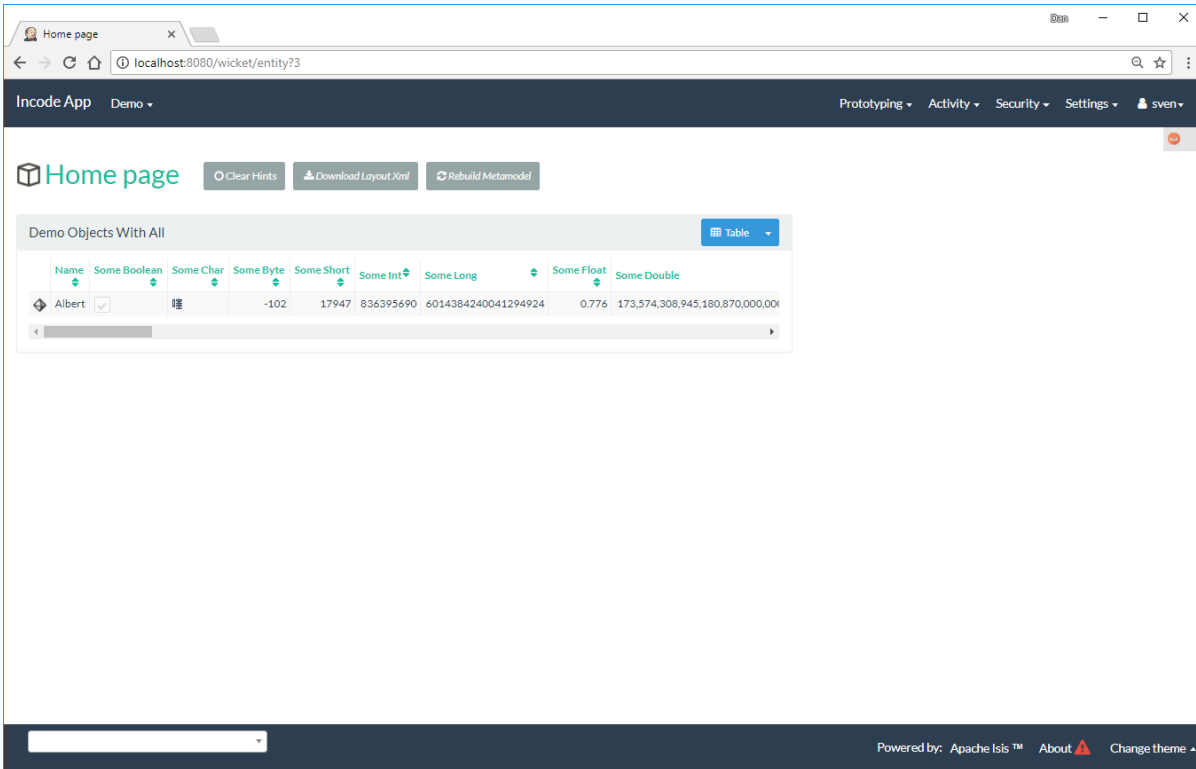
In addition, this module also acts as a useful regression suite for DataNucleus' persistence of value types (including our custom mappings of Isis' own value types).

Screenshots

The module's functionality can be explored by running the [quickstart with example usage](#) using the `org.incode.domainapp.example.app.modules.ExampleDomLibFakeDataAppManifest`.

The example app consists of a single domain entity that has a property for each of the value types supported by Isis.

A home page is displayed when the app is run:



will return an example demo object:

Albert

localhost:8080/wicket/entity/exampledemo.DemoObjectWithAll:0

Incode App Demo - Prototyping - Activity - Security - Settings - Sven -

Albert

Id

Id: 0

Version: 1

Primitives

Some Boolean* Update Some Boolean

Some Char* [a] Update Some Char

Some Byte* -102 Update Some Byte

Some Short* 17947 Update Some Short

Some Int* 836595690 Update Some Int

Some Long* 6014384240041294924 Update Some Long

Some Float* 0.776 Update Some Float

Wrappers

Some Boolean Wrapper Update Reset

Some Character Wrapper Update Reset

Some Byte Wrapper Update Reset

Some Short Wrapper Update Reset

Some Integer Wrapper Update Reset

Some Long Wrapper Update Reset

Some Float Wrapper Update Reset

Some Double Wrapper Update Reset

Dates and Times

Strings and Passwords

Some String Update Reset

Some Password Update Reset

Blobs and Clobs

Some Blob Update Reset

Some Clob Update Reset

Misc

Some Uri Update Reset

Some Uuid Update Reset

Some Enum OF3 Update Reset

Powered by: Apache Isis™ About Change theme

Fixtures and Integration Tests

Probably of more interest are the fixtures and integration tests that actually use the `FakeDataService`.

For example the `FakeDataDemoObjectUpdate` fixture script will update a demo object using the provided values (set as properties of the fixture script itself). However, if no value has been set by the calling test, then a random value, obtained from `FakeDataService`, will be used instead:

```
public class FakeDataDemoObjectUpdate extends DiscoverableFixtureScript {
```

```
    private FakeDataDemoObject fakeDataDemoObject;  
    public FakeDataDemoObject getFakeDataDemoObject() { ... }  
    public void setFakeDataDemoObject(final FakeDataDemoObject fakeDataDemoObject) { ... }
```

```
    ...  
    private Boolean someBoolean;  
    public Boolean getSomeBoolean() { ... }  
    public void setSomeBoolean(final Boolean someBoolean) { ... }
```

```
    private Character someChar;  
    public Character getSomeChar() { ... }  
    public void setSomeChar(final Character someChar) { ... }
```

```
    private Byte someByte;  
    public Byte getSomeByte() { ... }  
    public void setSomeByte(final Byte someByte) { ... }  
    ...
```

```
    protected void execute(final ExecutionContext executionContext) {
```

```
        ...  
        this.defaultParam("someBoolean", executionContext, fakeDataService.booleans().any());  
        this.defaultParam("someChar", executionContext, fakeDataService.chars().any());  
        this.defaultParam("someByte", executionContext, fakeDataService.bytes().any());  
        ...
```

```
        // updates  
        final FakeDataDemoObject fakeDataDemoObject = getFakeDataDemoObject();
```

```
...
wrap(fakeDataDemoObject).updateSomeBoolean(getSomeBoolean());
wrap(fakeDataDemoObject).updateSomeByte(getSomeByte());
wrap(fakeDataDemoObject).updateSomeShort(getSomeShort());
...
}
```

```
@javax.inject.Inject
private FakeDataService fakeDataService;
}
```

The `FakeDataService` can also be used within integration tests. For example, in `FakeDataDemoObjectsTest` a fake value is used to obtain a blob for update:

```
@Test
public void when_blob() throws Exception {
```

```
// given
Assertions.assertThat(fakeDataDemoObject.getSomeBlob()).isNull();
```

```
final Blob theBlob = fakeDataService.isisBlobs().anyPdf();
```

```
// when
updateScript.setFakeDataDemoObject(fakeDataDemoObject);
updateScript.setSomeBlob(theBlob);
```

```
fixtureScripts.runFixtureScript(updateScript, null);
```

```
nextTransaction();
```

```
// then
fakeDataDemoObject = wrap(fakeDataDemoObjects).listAll().get(0);
```

```
Assertions.assertThat(fakeDataDemoObject.getSomeBlob()).isNotNull();
Assertions.assertThat(fakeDataDemoObject.getSomeBlob().getMimeType().toString())
    .isEqualTo("application/pdf");
}
```

Note the use of `FakeDataService` in the "given" to obtain a PDF blob.

How to configure/use

Classpath

Update your classpath by adding this dependency in your project's `dom` module's `pom.xml`:

```
<dependency>
  <groupId>org.isisaddons.module.fakedata</groupId>
  <artifactId>isis-module-fakedata-dom</artifactId>
  <version>1.13.0</version>
</dependency>
```

Check for later releases by searching [Maven Central Repo](<http://search.maven.org/#search|ga|1|isis-module-fakedata-dom>).

For instructions on how to use the latest `-SNAPSHOT`, see the [contributors guide](#).

Bootstrapping

In the `AppManifest`, update its `getModules()` method, eg:

```
@Override
public List<Class<?>> getModules() {
    return Arrays.asList(
        ...
        org.isisaddons.module.fakedata.FakeDataModule.class,
        ...
    );
}
```

API and Implementation

The `FakeDataService` defines the following API:

```
public interface FakeDataService {  
  
    public Names name() { ... }  
    public Comms comms() { ... }  
    public Lorem lorem() { ... }  
    public Addresses addresses() { ... }  
    public CreditCards creditCard() { ... }  
    public Books books() { ... }  
  
    public Bytes bytes() { ... }  
    public Shorts shorts() { ... }  
    public Integers ints() { ... }  
    public Longs longs() { ... }  
    public Floats floats() { ... }  
    public Doubles doubles() { ... }  
    public Chars chars() { ... }  
    public Booleans booleans() { ... }  
  
    public Strings strings() { ... }  
  
    public Collections collections() { ... }  
    public Enums enums() { ... }  
  
    public JavaUtilDates javaUtilDates() { ... }  
    public JavaSqlDates javaSqlDates() { ... }  
    public JavaSqlTimestamps javaSqlTimestamps() { ... }  
    public JodaLocalDates jodaLocalDates() { ... }  
    public JodaDateTimes jodaDateTimes() { ... }  
    public JodaPeriods jodaPeriods() { ... }  
  
    public BigDecimals bigDecimals() { ... }  
    public BigIntegers bigIntegers() { ... }  
  
    public Urls urls() { ... }  
    public Uuids uuids() { ... }  
  
    public IsisPasswords isisPasswords() { ... }  
    public IsisMoneys isisMoneys() { ... }  
    public IsisBlobs isisBlobs() { ... }  
    public IsisClobs isisClobs() { ... }  
  
}
```

where each of the returned classes then provides suitable methods for obtaining values within that domain of values.

For example, `Names` provides:

```
public class Names ... {
    public String fullName() { ... }
    public String firstName() { ... }
    public String lastName() { ... }
    public String prefix() { ... }
    public String suffix() { ... }
}
```

and `IsisBlobs` provides:

```
public class IsisBlobs ... {
    public Blob any() { ... }
    public Blob anyJpg() { ... }
    public Blob anyPdf() { ... }
}
```

and `Collections` API includes:

```
public class Collections ... {
    public <T> T anyOf(final Collection<T> collection) { ... }
    public <T> T anyOfExcept(final Collection<T> collection, final Predicate<T>
except) { ... }
    public <T> T anyOf(final T... elements) { ... }
    public <T> T anyOfExcept(final T[] elements, final Predicate<T> except) { ... }
    ...
    public <E extends Enum<E>> E anyEnum(final Class<E> enumType) { ... }
    public <E extends Enum<E>> E anyEnumExcept(final Class<E> enumType, final
Predicate<E> except) { ... }
    public <T> T anyBounded(final Class<T> cls) { ... }
    public <T> T anyBoundedExcept(final Class<T> cls, final Predicate<T> except) { ... }
}
}
```

with similar methods for all the primitives

Known issues

None known at this time.

Dependencies

Maven can report modules dependencies using:

```
mvn dependency:list -o -pl modules/lib/fakedata/impl -D excludeTransitive=true
```

which, excluding Apache Isis itself, returns these compile/runtime dependencies:

```
com.github.javafaker:javafaker:jar:0.5
```

For further details on 3rd-party dependencies, see:

- [DiUS/java-faker](#)