Andy Gibson (http://www.andygibson.net/blog/)

Open Source Projects & Technical Writings

Generate Test Data with DataFactory

Feb 8th, 2011 (http://www.andygibson.net/blog/article/generate-test-data-with-datafactory/) in Articles (http://www.andygibson.net/blog/category/article/) by Andy Gibson (http://www.andygibson.net/blog/author/Andy Gibson/)

DataFactory is a project I just released which allows you to easily generate test data. It was primarily written for populating database for dev or test environments by providing values for names, addresses, email addresses, phone numbers, text, and dates.

To add DataFactory to your maven project, just add it as a dependency in your **`pom.xml`** file.

```
<dependency>
    <groupId>org.fluttercode.datafactory</groupId>
    <artifactId>datafactory</artifactId>
    <version>0.8</version>
    <type>jar</type>
</dependency>
```

Generating Test Data

Now you can create instances of the **`DataFactory`** class and create data :

```
public class Main {

    public static void main(String[] args) {
        DataFactory df = new DataFactory();
        for (int i = 0; i < 100; i++) {
            String name = df.getFirstName() + " "+ df.getLastName();
            System.out.println(name);
        }
    }
}
```

The produced output is :

```
Lindsey Craft
Erica Larsen
Ryan Levine
Erika Smith
Brooklyn Sloan
Karen Mayer
Eddie O'neill
Nancy Stevens
```

The **`DataFactory`** class can generate different types of values, from addresses to random text to random dates, to dates within a fixed time period. Addresses and business names can be created using the following code :

```
DataFactory df = new DataFactory();
for (int i = 0; i < 100; i++) {
    String address = df.getAddress()+","+df.getCity()+","+df.getNumberText(5);
    String business = df.getBusinessName();
    System.out.println(business + " located at " + address);
}
```

to produce :

```
Uvalda Signs located at 1383 Beam Way,Lyons,19316
Alma Accounting located at 1386 Countiss St,Nashville,14967
Fort Stewart Engineering located at 1753 Bethesda Rd,Springfield,26306
Sugar Hill Textiles located at 1141 Loudon Circle,Cordele,83937
Albany Engineering located at 1185 Grieves Avenue,Sugar Hill,36753
Poulan Insurance located at 816 Cohen Blvd,Lake City,74839
Crescent Services located at 1085 Cloveridge Boulevard,Bemiss,08769
```

Dates

There are a number of features to create dates, the first being creating a random date which is usually in a given sensible date range.

```
DataFactory df = new DataFactory();
Date minDate = df.getDate(2000, 1, 1);
Date maxDate = new Date();
for (int i = 0; i < 10; i++) {
    Date start = df.getDateBetween(minDate, maxDate);
    System.out.println("Date = "+start);
}
```

This produces a list of random dates between 1/1/2000 and the current date. Typically, a random date might be constrained by some other date, for example you can't have an end date that occurs before the start date. In this case, you would plug the start date in as the minimum date value :

```
DataFactory df = new DataFactory();
Date minDate = df.getDate(2000, 1, 1);
Date maxDate = new Date();

for (int i = 0; i < 10; i++) {
    Date start = df.getDateBetween(minDate, maxDate);
    Date end = df.getDateBetween(start, maxDate);
    System.out.println("Date range = " + dateToString(start) + " to " + dateToString(end));
}
```

The result is a list of dates where the second date is always later than the first :

```
Date range = 04/29/2005 to 07/16/2006
Date range = 08/07/2009 to 01/19/2010
Date range = 09/22/2000 to 12/15/2003
Date range = 07/31/2004 to 03/24/2009
Date range = 06/27/2003 to 01/10/2007
Date range = 07/10/2003 to 04/02/2008
Date range = 01/04/2003 to 01/12/2005
```

In many cases, you might want your end date to be only within a few days of the start date. For example, helpdesk support tickets or hotel stays don't last for years. To do this, you can specify the number of days from the base date you want to generate a result. In this case, we make the end date within 10 days of the begin date :

```
for (int i = 0; i < 10; i++) {
    Date start = df.getDateBetween(minDate, maxDate);
    Date end = df.getDate(start, 0, 10); //set end to within 10 days of the start
    System.out.println("Date range = " + dateToString(start) + " to " + dateToString(end));
}
```

And the result :

```
Date range = 04/29/2005 to 04/30/2005
Date range = 12/29/2003 to 12/30/2003
Date range = 06/25/2003 to 07/03/2003
Date range = 10/19/2009 to 10/19/2009
```

You can also specify a negative minimum days value that could return a date prior to the base date or a positive minimum date value to get a later date. Here's a more complex example that uses different date rules to come up with some complex test data.

```
for (int i = 0; i < 10; i++) {
     //generate an order date
    Date orderDate = df.getDateBetween(minDate, maxDate);

    //estimate delivery 4-10 days after ordering
    Date estimatedDeliveryDate = df.getDate(orderDate, 4, 10);

    //deliver between 2 days prior and 3 days after delivery estimate
    Date actualDeliveryDate = df.getDate(estimatedDeliveryDate, -2, 3);

    String msg =  "Ordered on "+dateToString(orderDate) +
        " deliver by = "+dateToString(estimatedDeliveryDate)+
        " delivered on " + dateToString(actualDeliveryDate);

    if (estimatedDeliveryDate.before(actualDeliveryDate)) {
        msg = msg + " - LATE";
    }
    if (estimatedDeliveryDate.after(actualDeliveryDate)) {
        msg = msg + " - EARLY";
    }
    System.out.println(msg);
}

```

Here we calculate an order date, and create a delivery date that is at least 4 days out but no more than 10, and then we created an actual delivery date that is between 2 days prior and 3 days after the expected delivery date.
Notice how we cherry picked the dates, the estimated delivery date is at least 4 days out from the order date, and the actual delivery date will only be at most 2 days prior to the estimated date. This means the actual delivery date is always at least 2 days out from the order date and we won't get a delivery date value that is before the item was ordered. This code produces the following values :

```
Ordered on 04/29/2005 deliver by = 05/06/2005 delivered on 05/06/2005
Ordered on 08/07/2009 deliver by = 08/13/2009 delivered on 08/13/2009
Ordered on 09/22/2000 deliver by = 09/27/2000 delivered on 09/25/2000 - EARLY
Ordered on 07/31/2004 deliver by = 08/07/2004 delivered on 08/09/2004 - LATE
Ordered on 06/27/2003 deliver by = 07/04/2003 delivered on 07/04/2003
Ordered on 07/10/2003 deliver by = 07/19/2003 delivered on 07/18/2003 - EARLY
Ordered on 01/04/2003 deliver by = 01/08/2003 delivered on 01/08/2003
```

Custom Random Values

If there is a set of values that is very specific to your application that you might want to generate data from, you can use methods on the **`DataFactory`** class to return values with the option for it to be randomly be a default value.

```
public static void main(String[] args) {
```

```
    DataFactory df = new DataFactory();

    //favorite animal
    String[] values = {"Cat","Dog","Goat","Horse","Sheep"};
    for (int i = 0; i < 100; i++) {
        System.out.println(df.getItem(values,80,"None"));
    }
}
```

This example uses the array of animals and returns a value with a 20% chance of being the default value of “None” to produce the following :

```
Sheep
None
Dog
Horse
```

Textual Data

Random text data comes in two forms, absolutely random data and text data made up of words. You can generate either using the following methods :

```
DataFactory df = new DataFactory();
System.out.println(df.getRandomText(20, 25));
System.out.println(df.getRandomChars(20));
System.out.println(df.getRandomWord(4, 10))
```

which produces

```
badly numbers good hot I
ywyypgqorighfawpftjq
demanded
```

All three of these methods can be passed a single length which returns a fixed length string, or a min/max length which produces a random string with a length somewhere between the min/max. For the single word method, if there are no words in the dictionary of suitable length, then a word is generated using random characters.

Changing the test data values produced

The data used to generate the values come from classes that can be replaced with other versions. For example, the name values can be changed by providing the **`DataFactory`** instance with an object that implements the **`NameDataValues`** interface. Here is a simple class that does that to return Scandinavian first names and delegates to the the default implementation to return all the other values.

```
public class ScandinavianNames  implements NameDataValues {

    //first name values to use
    String[] firstNames = {"Anders","Freydís","Gerlach","Sigdis"};

    //delegate to the default implementation for the other values
    NameDataValues defaults = new DefaultNameDataValues();

    public String[] getFirstNames() {
        //return our custom list of names
        return firstNames;
    }

    //for the other values, just use the defaults
    public String[] getLastNames() {
        return defaults.getLastNames();
    }

    public String[] getPrefixes() {
        return defaults.getPrefixes();
    }

    public String[] getSuffixes() {
        return defaults.getSuffixes();
    }

}

```

Obviously, to use all your own names you would add and return values for last name and the suffix/prefix values. To use this new implementation, just create an instance of the data provider and pass it to the instance of the data factory.

```
public static void main(String[] args) {
    DataFactory df = new DataFactory();
    df.setNameDataValues(new ScandinavianNames());
    for (int i = 0; i < 10; i++) {
        System.out.println(df.getName());
    }
}
```

Our results are

```
Sigdis Craft
Gerlach Larsen
Sigdis Levine
Sigdis Smith
Freydís Sloan
Gerlach Mayer
```

```
6
```

You can always start working with the default implementation and use a more locale specific implementation if you need it later.

The different pieces that can be replaced are as follows :

- **`NameDataValues`** – Generates names and suffix/prefixes
- **`ContentDataValues.java`** – Generates words, business types, email domain names and top level domain values
- **`AddressDataValues`** – Generates city names, street names and address suffixes

Note that if you intend on replacing the component that generates words, you should have a good collection of words of various lengths from 2 up to say 8 or more characters.

Hopefully this will give you a head start in generating data in development and test environments for new projects. Now I have DataFactory in the Central Maven Repository I plan on using this in the Knappsack archetypes rather than hard coding the data which was in fact generated from an earlier DataFactory implementation.

DataFactory (http://www.andygibson.net/blog/tag/datafactory/)

← Shrinkwrap DSLs Makes Building Config Files Easy (http://www.andygibson.net/blog/article/shrinkwrap-dsls-makes-building-config-files-easy/)

Double the speed of (some of) your JSF pages (and dodge bugs too) → (http://www.andygibson.net/blog/article/double-the-speed-of-some-of-your-jsf-pages-and-dodge-bugs-too/)

3 thoughts on “Generate Test Data with DataFactory”

Adrian Mitev

February 10, 2011 (http://www.andygibson.net/blog/article/generate-test-data-with-datafactory/comment-page-1/#comment-16024)

Have you tried benerator?

Job de Noo (http://www.denoo.info)

February 11, 2011 (http://www.andygibson.net/blog/article/generate-test-data-with-datafactory/comment-page-1/#comment-16025)

Nice library, would be even nicer if you could compile it against java 5 instead of 6. Or do you use some java 6 features?

Otherwise a random BigDecimal value would also be appreciated.

swung

February 14, 2011 (http://www.andygibson.net/blog/article/generate-test-data-with-datafactory/comment-page-1/#comment-16027)

Thanks Adrian for your job!

I am trying to provide I18N support to this library. https://github.com/swung/datafactory (https://github.com/swung/datafactory)

About

Location : Manchester, UK
Available for work : July 2015
Contact Me (http://www.andygibson.net/blog/contact-me/)

I've been developing software for around 20 years, mostly in Java. This site is a home for my open source projects, writings, articles and tutorials mainly focusing on Java and Java EE.

 (http://twitter.com/andy_gibson_)  (http://www.andygibson.net/blog/feed/)  (http://www.linkedin.com/in/andygibsonprogrammer) (https://www.ohloh.net/accounts/79307?ref=Tiny)

Recent Posts

- Using Facelets Parameters with Templates in JSF (http://www.andygibson.net/blog/programming/using-facelets-parameters-with-templates-in-jsf/)
- JSF Page Templates With Facelets (http://www.andygibson.net/blog/programming/jsf-page-templates-with-facelets/)
- Microservices implement old principles in new technology (http://www.andygibson.net/blog/post/microservices-implement-old-principles-in-new-technology/)
- Exonerating the JSF Lifecycle (http://www.andygibson.net/blog/article/exonerating-the-jsf-lifecycle/)
- Comparing Constants Safely (http://www.andygibson.net/blog/quickbyte/comparing-constants-safely/)
- Microsoft Open Sources .Net But Is It Enough? (http://www.andygibson.net/blog/news/microsoft-open-sources-net-but-is-it-enough/)
- Immutability Through Interfaces (http://www.andygibson.net/blog/programming/immutability-through-interfaces/)
- Java 8 features key to adoption (http://www.andygibson.net/blog/news/java-8-features-key-to-adoption/)
- Back Home (http://www.andygibson.net/blog/personal/back-home/)
- Auditing Entities With JPA Events (http://www.andygibson.net/blog/article/auditing-entities-with-jpa-events/)

Recent Comments

- Kieron Wilkinson on Comparing Constants Safely (http://www.andygibson.net/blog/quickbyte/comparing-constants-safely/comment-page-1/#comment-16238)
- Andy Gibson (http://www.andygibson.net) on Immutability Through Interfaces (http://www.andygibson.net/blog/programming/immutability-through-interfaces/comment-page-1/#comment-16237)
- cblin on Immutability Through Interfaces (http://www.andygibson.net/blog/programming/immutability-through-interfaces/comment-page-1/#comment-16236)
- Lucaster on Back Home (http://www.andygibson.net/blog/personal/back-home/comment-page-1/#comment-16234)
- Andy Gibson (http://www.andygibson.net) on Back Home (http://www.andygibson.net/blog/personal/back-home/comment-page-1/#comment-16233)

Tags

Apprentice (http://www.andygibson.net/blog/tag/apprentice/) Archetypes (http://www.andygibson.net/blog/tag/archetypes/) Arquillian (http://www.andygibson.net/blog/tag/arquillian/) Articles (http://www.andygibson.net/blog/tag/articles/) Beginner (http://www.andygibson.net/blog/tag/beginner/) Blog (http://www.andygibson.net/blog/tag/blog/) CDI (http://www.andygibson.net/blog/tag/cdi/) Conversation (http://www.andygibson.net/blog/tag/conversation/) DataValve (http://www.andygibson.net/blog/tag/datavalve/) Delphi (http://www.andygibson.net/blog/tag/delphi/) Design (http://www.andygibson.net/blog/tag/design/) Eclipse (http://www.andygibson.net/blog/tag/eclipse/) EJB (http://www.andygibson.net/blog/tag/ejb/) Facelets (http://www.andygibson.net/blog/tag/facelets/) Food (http://www.andygibson.net/blog/tag/food/) Frameworks (http://www.andygibson.net/blog/tag/frameworks/) Games (http://www.andygibson.net/blog/tag/games/) Geek (http://www.andygibson.net/blog/tag/geek/) Glassfish (http://www.andygibson.net/blog/tag/glassfish/) Graphics (http://www.andygibson.net/blog/tag/graphics/) Java (http://www.andygibson.net/blog/tag/java/) Java EE (http://www.andygibson.net/blog/tag/java-ee/) JBoss (http://www.andygibson.net/blog/tag/jboss/) JEE (http://www.andygibson.net/blog/tag/jee/) Journeyman (http://www.andygibson.net/blog/tag/journeyman/) JPA (http://www.andygibson.net/blog/tag/jpa/) JSF (http://www.andygibson.net/blog/tag/jsf/) JTexgen (http://www.andygibson.net/blog/tag/jtexgen/) Knappsack (http://www.andygibson.net/blog/tag/knappsack/) Maven (http://www.andygibson.net/blog/tag/maven/) Microsoft (http://www.andygibson.net/blog/tag/microsoft/) Modeling (http://www.andygibson.net/blog/tag/modeling/) Movies (http://www.andygibson.net/blog/tag/movies/) Netbeans (http://www.andygibson.net/blog/tag/netbeans/) Open Source (http://www.andygibson.net/blog/tag/open-source/) Personal (http://www.andygibson.net/blog/tag/personal/) Podcast (http://www.andygibson.net/blog/tag/podcast/) Programming (http://www.andygibson.net/blog/tag/programming/) Project Kenai (http://www.andygibson.net/blog/tag/project-kenai/) Seam (http://www.andygibson.net/blog/tag/seam/) Spigot (http://www.andygibson.net/blog/tag/spigot/) Spring (http://www.andygibson.net/blog/tag/spring/) Testing (http://www.andygibson.net/blog/tag/testing/) Weld (http://www.andygibson.net/blog/tag/weld/) Wicket (http://www.andygibson.net/blog/tag/wicket/)

Archives

Select Month