

JRandO

0.0.11

Testdatengenerator

27.08.2009

© 2008-2009 Hans-Dieter Thiel Consulting limited

## Inhalt

Einleitung.....	4
Voraussetzungen.....	4
Installation.....	4
Anwendungsbereich.....	4
Komponenten.....	4
Interfaces und Implementierungen .....	4
IRandomizer .....	4
IntegerIndexSequence.....	4
IntegerRollingSequence.....	5
IntegerRandomizer.....	5
LongIndexSequence.....	5
LongRollingSequence .....	5
LongRandomizer.....	5
IValueProvider .....	5
IValueProviderCache .....	5
ISimpleValueGenerator .....	6
jrandBooleanGenerator .....	6
jrandEnumGenerator .....	6
jrandIntegerGenerator.....	6
jrandLongGenerator .....	6
jrandUuidValueGenerator.....	6
jrandObjectGenerator.....	6
SimpleDoubleGenerator.....	6
SimpleDateGenerator.....	6
SimpleFileGenerator.....	6
SimpleIntegerSequenceGenerator.....	6
Simple JDBCGenerator .....	6
Beispiele .....	6
Release Notes.....	7
Change Log .....	7
0.0.1.....	7
0.0.2.....	7
0.0.3.....	7
Lizenzen .....	8

Duales Lizenz Model.....	8
Open Source Lizenz .....	8
Kommerzielle Lizenz.....	8
Umfang der Nutzungsrechte .....	8
Gewährleistung / Haftung .....	8
Erfüllung der Gewährleistung.....	8
Keine Gewährleistung für bestimmte Verwendungszwecke .....	8
Keine Haftung für Folgeschäden .....	9
Urheberrecht .....	9
Anwendbares Recht .....	9
Sonstiges.....	9
Marken- und Produktnamen.....	9

## Einleitung

Wer testen will benötigt Testdaten. Die Generierung von Testdaten ist eine immer wiederkehrende Tätigkeit. Daher haben wir beschlossen ein Framework zur Testdatengenerierung zu entwickeln, welches die Implementierung vereinfacht und vereinheitlicht. Das Ergebnis ist JRandO, ein Testdatengenerator- oder besser ein Testobjektgeneratorframework. JRandO kann in JUnit Tests oder in Performance Tests (z.B. mit JMeter) benutzt werden. Es kann auch benutzt werden um Daten zu anonymisieren oder aber in einem Simulationsframework sinnvoll sein.

## Voraussetzungen

JDK 1.5 oder höher

## Installation

t.b.s.

## Anwendungsbereich

Das Framework ist in erster Linie dazu gedacht Java Objekte (POJOs) mit Zufallsdaten zu erzeugen. Diese können für Tests, Lasttests oder auch in der Simulation genutzt werden. Es ist nicht geplant damit direkt Datenbanktabellen zu füllen.

## Komponenten

Das Framework besteht aus den folgenden Java Komponenten: Randomizer, SimpleValueProvider, SimpleValueProviderCache, SimpleValueGenerator. Für diese Komponenten ist jeweils ein Interface definiert.

## Interfaces und Implementierungen

### IRandomizer

Ein Randomizer liefert Zufallszahlen zwischen einem gegebenen Minimum und Maximum. Durch den Seed können wiederholbare Zufallszahlenfolgen erzeugt werden.

```
public interface IRandomizer<R extends Number> {  
  
    public void setMax(R max);  
  
    public void setMin(R min);  
  
    public R getRandom();  
  
    public void setSeed(long seed);  
  
}
```

### IntegerIndexSequence

Liefert fortlaufende Integer ab einem gegebenen Minimum (default:0). Ein Maximum kann nicht angegeben werden.

### *IntegerRollingSequence*

Liefert fortlaufende Integer ab einem gegebenen Minimum (default:0). Ab einem gegebenen Maximum wird die Folge ab dem Minimum wiederholt.

### *IntegerRandomizer*

Liefert zufällige Integer zwischen einem gegebenen Minimum (default: Integer.MIN\_VALUE) und Maximum (default: Integer.MAX\_VALUE).

### *LongIndexSequence*

Liefert fortlaufende Long ab einem gegebenen Minimum (default:0). Ein Maximum kann nicht angegeben werden.

### *LongRollingSequence*

Liefert fortlaufende Long ab einem gegebenen Minimum (default:0). Ab einem gegebenen Maximum wird die Folge ab dem Minimum wiederholt.

### *LongRandomizer*

Liefert zufällige Long zwischen einem gegebenen Minimum (default: Long.MIN\_VALUE) und Maximum (default: Long.MAX\_VALUE).

### **IValueProvider**

Ein SimpleValueProvider enthält eine Menge von Werten. Der Zugriff auf die einzelnen Werte erfolgt über einen fortlaufenden Index von MinIndex bis MaxIndex. Der SimpleValueProvider kann einen SimpleValueProviderCache benutzen um die Werte zwischenspeichern. Als Quelle für die Werte kommen z.B. eine Datei (SimpleFileValueProvider) oder eine Datenbank (SimpleJDBCValueProvider) in Frage.

```
public interface IValueProvider<V,I extends Number> {  
    public V getValue(I index);  
    public I getMaxIndex();  
    public I getMinIndex();  
    public void init();  
}
```

### **IValueProviderCache**

Der SimpleValueProviderCache speichert alle Werte eines ValueProviders zwischen. Dies kann im Hauptspeicher (InMemoryValueProviderCache) oder in einer InMemoryDatenbank (InMemoryDatabaseValueProviderCache) passieren.

```
public interface IValueProviderCache<V,I> {  
    public void put(I index, V value);  
    public V get(I index);  
}
```

## **ISimpleValueGenerator**

Ein SimpleValueGenerator liefert zufällige Werte für eine gegebene Klasse. Er nutzt dazu einen ValueProvider in Verbindung mit einem Randomizer.

```
public interface ISimpleValueGenerator<V> {  
    public V next();  
}
```

## ***jrandoBooleanGenerator***

Generiert Booleans (true, false). Benutzt jrandoIntegerRandomizer und jrandoBooleanValueProvider.

## ***jrandoEnumGenerator***

Generiert Enum Werte anhand einer gegebenen Enum Klasse. Benutzt jrandoIntegerRandomizer und jrandoEnumValueProvider.

## ***jrandoIntegerGenerator***

Generiert Integer Werte. Benutzt jrandoIntegerValueProvider und jrandoIntegerRandomizer.

## ***jrandoLongGenerator***

Generiert Long Werte. Benutzt jrandoLongValueProvider und jrandoLongRandomizer.

## ***jrandoUuidValueGenerator***

Generiert UUIDs.

## ***jrandoObjectGenerator***

Generiert Java Objekte POJOs.

## ***SimpleDoubleGenerator***

Generiert Double Werte

## ***SimpleDateGenerator***

Generiert Datumswerte

## ***SimpleFileGenerator***

Generierte String Werte, welche in einer Datei abgelegt sind. Die Werte werden in einem Cache gespeichert.

## ***SimpleIntegerSequenceGenerator***

Generiert eine Folge von aufsteigenden Integer Werten.

## ***SimpleJDBCGenerator***

Generiert String Werte, welche in einer Datenbank abgelegt sind. Die Werte können in einem Cache gespeichert werden.

## **Beispiele**

Beispiele für die Anwendung des Frameworks finden Sie in JRandoSample.zip!

## Release Notes

### Change Log

#### 0.0.1

Erster Entwurf.

#### 0.0.2

BugFixes, IntegerSequence, LongSequence, SimpleBooleanProvider ergänzt.

#### 0.0.3

BugFixes, Collection Generatoren

## Lizenzen

### Duales Lizenz Model

Für Open Source Projekte JRandO ist verfügbar unter der GNU GPL v3 + FLOSS Exception. Für die kommerzielle Nutzung gibt es eine kommerzielle Lizenz.

#### Open Source Lizenz

JRandO ist verfügbar unter der [GNU GPL v3](#). Für Open Source Projekte, welche nicht unter die GPL v3 Lizenz fallen gibt es zwei FLOSS Exceptions.

[Open Source License Exception for Applications](#)

[Open Source License Exception for Development](#)

#### Kommerzielle Lizenz

##### *Umfang der Nutzungsrechte*

Hans-Dieter Thiel Consulting limited gewährt Ihnen eine Lizenz für die Software. Die Lizenz gibt Ihnen die Berechtigung, die Software auf einem oder mehreren Computern, die Sie privat oder beruflich in branchenüblicher Weise nutzen, zu installieren. Die Nutzungsrechte sind weltweit gültig, nicht ausschließlich und nicht übertragbar. Die Software wird mit den Funktionen und Eigenschaften überlassen, die sie zum Zeitpunkt der Installation enthält.

Sie dürfen die Software weder vermieten noch verleasen oder verleihen. Allerdings dürfen Sie die überlassenen Lizenzen vollständig und auf Dauer an einen Dritten übertragen, vorausgesetzt, dass der Empfänger sich mit diesen Lizenzbedingungen einverstanden erklärt.

##### *Gewährleistung / Haftung*

Hans-Dieter Thiel Consulting limited beschränkt die Haftung auf Schäden, die durch Vorsatz verursacht wurden. Im übrigen gelten die gesetzlichen Vorschriften. Die Gewährleistungsdauer beträgt 12 Monate ab Beginn der gesetzlichen Gewährleistungspflicht. Ist der Kunde ein Verbraucher im Sinn des Bürgerlichen Gesetzbuches, so beträgt die Gewährleistungsfrist zwei Jahre.

##### *Erfüllung der Gewährleistung*

Gewährleistungsansprüche werden durch Reparatur oder den Ersatz fehlerhafter Software erfüllt. Fehlerhafte Software ist an Hans-Dieter Thiel Consulting limited zurückzugeben.

Ein Gewährleistungsanspruch besteht nicht, wenn die Fehlerhaftigkeit der Software durch einen Unfall, durch Missbrauch oder fehlerhafte Anwendung verursacht wird.

Für eine Ersatz-Software übernimmt Hans-Dieter Thiel Consulting limited eine Gewährleistung für den Rest der ursprünglichen Gewährleistungszeit, mindestens jedoch für weitere 30 Tage, gerechnet vom Datum der Entgegennahme der Ersatzsoftware an.

##### *Keine Gewährleistung für bestimmte Verwendungszwecke*

Hans-Dieter Thiel Consulting limited übernimmt keine Gewähr dafür, dass die Software für die von Ihnen bestimmten Zwecke, für die Sie die Software einsetzen wollen, tauglich ist oder mit anderer, von Ihnen gewählter Software kompatibel ist. Sie tragen die alleinige Verantwortung für Auswahl, Installation und Nutzung sowie für die damit beabsichtigten Ergebnisse.

### ***Keine Haftung für Folgeschäden***

Mit Ausnahme von vorsätzlich verursachten Schäden haftet Hans-Dieter Thiel Consulting limited nicht für irgendeinen Schaden, der durch die Verwendung oder die Unmöglichkeit der Verwendung der Software verursacht worden ist. Dies gilt ohne Ausnahme auch für entgangenen Geschäftsgewinn, Betriebsunterbrechungen, entgangene Geschäftsinformation oder anderen wirtschaftlichen Verlust, auch wenn Hans-Dieter Thiel Consulting limited vorher auf die Möglichkeit eines solchen Schadens hingewiesen wurde.

Die Beschränkung der Haftung gilt nicht für Schäden, die auf der Verletzung des Körpers, Lebens oder der Gesundheit beruhen.

### ***Urheberrecht***

Hans-Dieter Thiel Consulting limited hat die alleinigen, ausschließlichen Rechte an der Software. Die Überlassung einer Lizenz gewährt Ihnen allein das Recht zur Nutzung der Software in dem Umfang, der durch diese Lizenzbedingungen eingeräumt wird; alle Urheberrechte verbleiben vollständig bei Hans-Dieter Thiel Consulting limited.

### ***Anwendbares Recht***

Die Überlassung der Software unterliegt dem deutschen Recht. Verweist dieses Recht auf ausländische Rechtsordnungen, sind solche Verweisungen unwirksam. Die Anwendung des UN-Kaufrechts (UNCITRAL) wird ausdrücklich ausgeschlossen. Der Vertragsort und der Gerichtsstand sind für Vollkaufleute Südlohn, sonst gilt die gesetzliche Regelung. Die Überlassung gilt zu dem Zeitpunkt als vereinbart, zu dem Sie sich die Software durch einen Download beschafft haben, spätestens jedoch mit der Installation der Software.

### ***Sonstiges***

Ihre Allgemeinen Geschäfts- und Einkaufsbedingungen finden keine Anwendung.

## **Marken- und Produktnamen**

Alle erwähnten Marken- und Produktnamen sind Warenzeichen der jeweiligen Rechtsinhaber und werden hiermit anerkannt. Das Fehlen einer entsprechenden Kennzeichnung in der Software und der Dokumentation bedeutet nicht, dass es sich um einen freien Namen im Sinne der Waren und Markenzeichengesetzgebung handelt.