

# dgMaster

A simple, free, extensible, open source data generator

(An ongoing project)

Released under the GPL license

© 2007 mmichalak.info

User's Guide

## Table of contents

1	Introduction.....	1
2	Current features.....	2
2.1	General features .....	2
2.2	Data generators .....	2
2.3	Output format.....	3
3	Using the application .....	4
3.1	Data generator panels.....	6
3.1.1	Boolean generator (BooleanRandomiser).....	7
3.1.2	Date generator (DateRandomiser, SQLDateRandomiser).....	8
3.1.3	First names generator (FirstnameRandomiser).....	10
3.1.4	E-mails generator (EmailRandomiser) .....	12
3.1.5	English text generator (EnglishTextRandomiser).....	13
3.1.6	English text generator (EnglishWordRandomiser).....	14
3.1.7	English full names (FullnameRandomiser) .....	15
3.1.8	English last names (LastNameRandomiser) .....	16
3.1.9	List items generator.....	17
3.1.10	List items sequencer.....	18
3.1.11	Long sequencer .....	19
3.1.12	Integer numbers (Byte, Short, Integer and Long randomisers) .....	20
3.1.13	Real numbers (Double and Float randomisers) .....	21
3.1.14	Timestamp (Timestamp Randomiser).....	22
3.1.15	SQLTime (SQLTimeRandomiser).....	23
3.1.16	Random strings (StringRandomiser).....	24
3.1.17	Unique random strings (UniqueRandomiser) .....	25
4	<i>Using the Generators</i> .....	26
4.1.1	Generating text data .....	26
4.1.2	Generating database data .....	29

# 1 Introduction

dgMaster is a simple, free and extensible data generator application that is capable of outputting high volumes of different types of data in different formats. A data generator application can be useful when it comes to testing an application, most typically, a database application.

Database applications are meant to cope with large volumes of data. During the implementation phase of such applications, a developer or team of developers will most probably be concerned with devising the algorithms that implement the required business logic and will be less concerned with populating the database with meaningful data. The data entry phase is part of the development cycle: Forms/web pages are built and tested, and part of this testing involves data entry.

Although data entry is part of testing the application, many times it is a necessity to have large volumes of data in order to do more thorough testing. Large volumes of data are necessary when it comes to testing report modules, running batch processes, or carrying out load-testing in an application.

*dgMaster can help the developer with populating his application with meaningful data (at least this is the goal/vision of this application). It is currently under development, and lacks quite many important features. This user guide explains how to use the existing features, features that are not so likely to change in the future versions of the software.*

## 2 Current features

This section outlines the most important features of the application. It also includes certain features that are *not* part of the application now, but they should be, (admittedly, in this sense, “Current features” is a misleading title, also see FAQ section of [dgmaster.sourceforge.net](http://dgmaster.sourceforge.net)).

### 2.1 General features

- dgMaster can easily be extended with new data generators.
- The application features a modern GUI and is layered and modular, so that it can also be used from the command line, (although for the moment being, a command line front-end is not part of the application).
- It makes use of a configurable logging mechanism, (log4j), so getting an idea of what is going on underneath the GUI at any given time is easier (good for debugging purposes and when developing new generators).
- Context sensitive help is desirable but is not currently implemented. After having completed most of the implementation work, it will be easier to start working on this.

### 2.2 Data generators

Currently, dgMaster supports a number of simple, single value data generators:

- Boolean,
- Date,
- String (random chars.),
- Numerical types (Integer, Long, Float, Double),
- SQL date, SQL time, timestamp,
- English first names, last names, and full names,
- Emails,
- English text,
- String values from user-defined lists (either retrieved sequentially, or randomly)
- Unique strings (for db keys),
- Numerical increment values.

Understandably, many types of different data generators are missing. The list will be enriched in the long run. Data generators that need to be developed include generators based on mask fields,

post codes of various countries, text and names of different languages, credit card numbers, social security numbers, and many more. However, the main feature of this application is that these generators can be easily implemented. Having as many generators as possible is nice and desirable, but is not a high priority as these can always be implemented anytime (*also see dgMaster - developer's guide*).

## **2.3 Output format**

At this moment, dgMaster is capable of producing random data in text format only (of course, support for database is high on the upgrades-list). Nevertheless, it offers the typical options such as aligning the fields, selecting the separator character or having fixed widths for each piece of data, etc.

### 3 Using the application

In order to be able to generate data, it is necessary to tweak the parameters of an existing data-generator definition and save these parameters for later use. Tweaking the parameters of a generator means providing special parameters for that specific generator. For example, for the boolean generator, the user is able to provide the percentage of true, false and null values. Similarly, for the integer data generator, the user can provide a custom distribution for the generated values; when it comes to the double data generator, the user can provide the precision and so on. Many times in this manual, as well as in the application, the terms randomiser and generator are used interchangeably.

Each data generator can be customised from the generator's panel. If a certain generator does not include the parameters you would like to use, you can develop your own data generator (*see dgMaster – Developer's guide*).

Existing data generators are accessible using the generators' browser, (JTree component, on the left side of the GUI, Fig. 1), under the node labeled “**Built-in generators**”. Expanding the specific node reveals the generators that are currently installed in the application (nodes with gear icon). Clicking on a certain generator name will bring up that generator's associated panel on the right. Figure 1 shows the nodes that represent the built-in definitions.

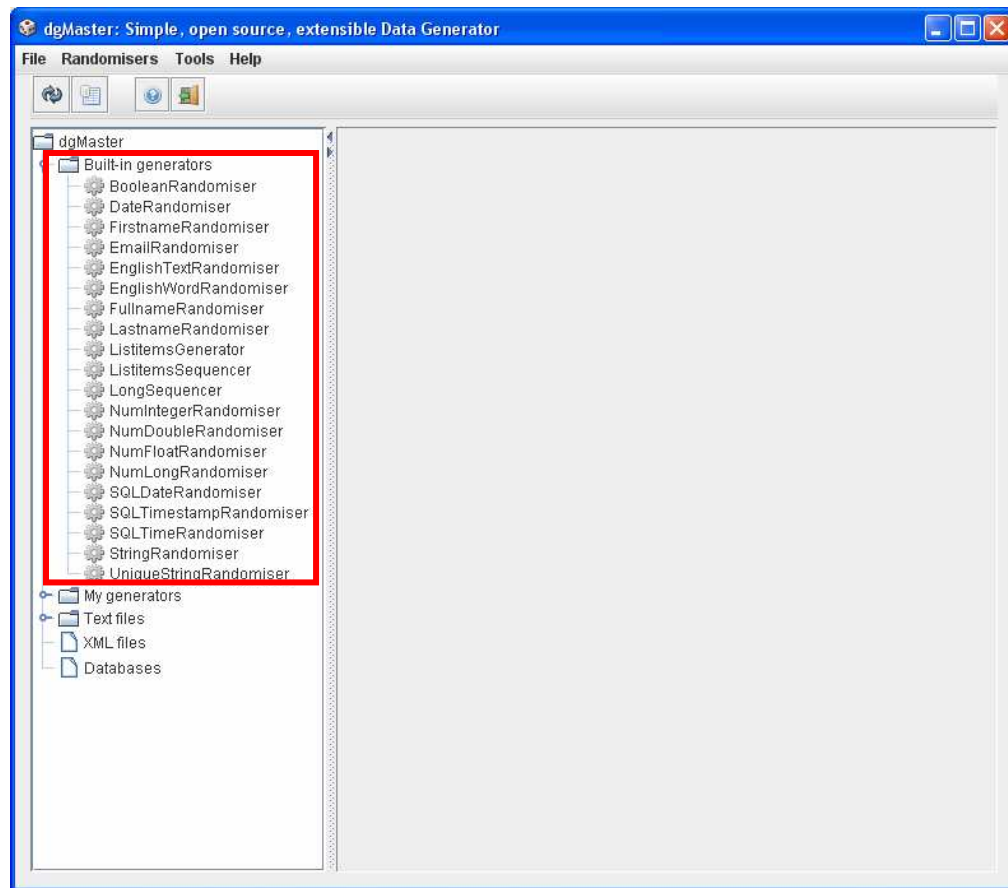
After having provided the required parameters for a certain data generator, the user needs to save the new parameterised data generator definition. User-defined generators are saved under the node “**My generators**”. Expanding this node shows the list of user-defined generators, which appear with a gear icon and a pencil.



*The idea of providing the parameters for a certain data generator and then saving these parameters is central to the application. It is not possible to use a data generator “on the fly”. The user selects a built-in data generator, fills in the required parameters, gives a new name to this generator, saves it, and then recalls the generator by its saved name when there is a need to use it.*

The next sub-sections explain what each data generator panel shows, and how the user can parameterise each data generator. There are a couple of tricky points with regards to certain

generators, for example dates. Part of this text will also be included in the application's help system in future versions.



**Figure 1.** Installed generators in the application; these need to be parameterised according to user's needs.

### 3.1 Data generator panels

Every single data generator panel includes two standard fields: *Name* and *description*. Filling the “name” field is mandatory; it essentially gives a name to your parameters set. For example, if you choose the integer data generator and provide a range of [0..140] for its generated range, you may want to call this parameters’ set “MilesPerHour”. The “description” field is optional. Regardless, of the data generator panel, a button on the left-bottom of the main window will allow you to save your definition. Figure 2, shows the boolean generator; the Save button appears regardless of which data generator panel is show.



User definitions are saved in the file “Repository.xml”. It is possible to remove a user definition by editing this file (currently, there is no “remove” button so as to remove a user-defined generator).

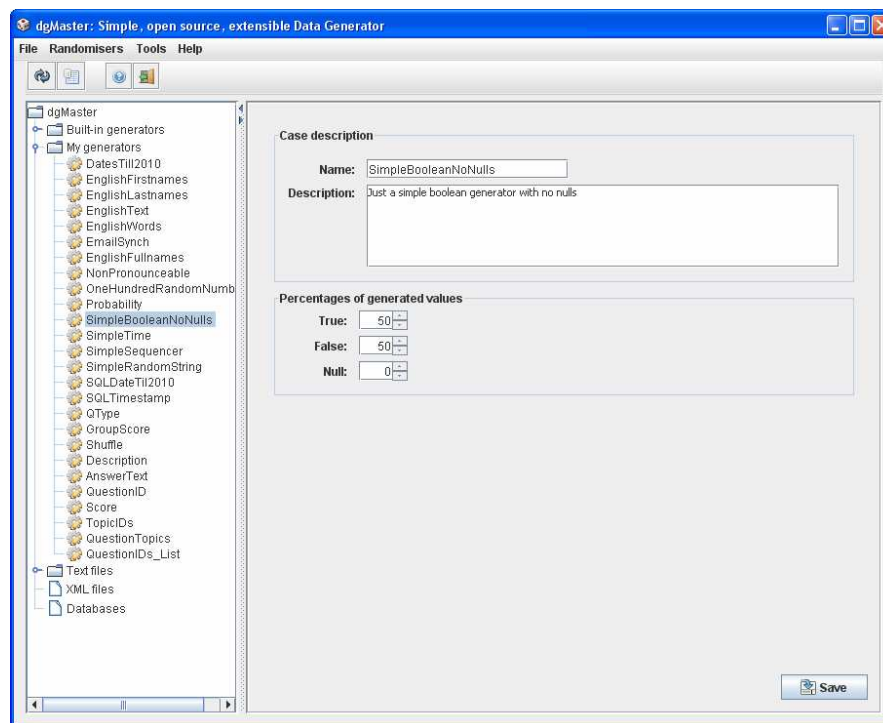


Figure 2, the boolean generator; on the left, user-defined parameters for each data generator, these appear with a gear and pencil icon.

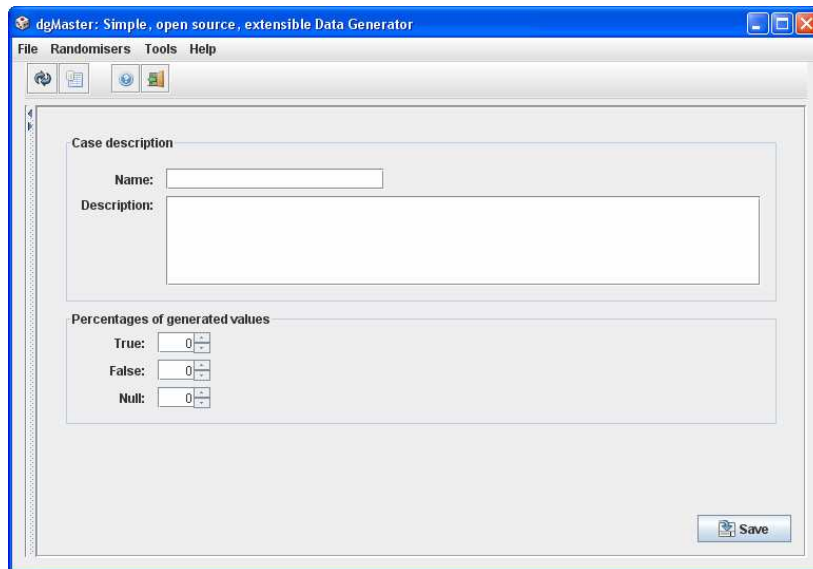


### 3.1.1 Boolean generator (BooleanRandomiser)

The boolean generator is quite simple. The user needs to provide the generator's name, and the percentages for the generated values.

Validation rules:

- Name is required.
- “True” and “False” values should sum up to 100, no negative values.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.



The screenshot shows a software window titled "dgMaster: Simple, open source, extensible Data Generator". The window has a menu bar with "File", "Randomisers", "Tools", and "Help". Below the menu bar is a toolbar with icons for file operations and a help icon. The main content area is divided into two sections. The top section, titled "Case description", contains a "Name:" label followed by a text input field, and a "Description:" label followed by a larger text area. The bottom section, titled "Percentages of generated values", contains three rows of input fields: "True:" with a value of 0, "False:" with a value of 0, and "Null:" with a value of 0. Each input field has small up and down arrow buttons. A "Save" button is located in the bottom right corner of the window.

Figure 3, boolean generator.

### 3.1.2 Date generator (DateRandomiser, SQLDateRandomiser)

The date randomiser generates Date objects (for java.sql.Date objects use the SQLDateRandomiser). The entry format for the dates is “yyyy-mm-dd”.

**There is a tricky issue at this point:** Although the user will be prompted when the entered date follows a different format, there is a slight problem as shown in the panel below. Java tries to convert date values that comply with the format to valid dates. This means that if the date 2040-31-12 is entered (which does not follow the yyyy-mm-dd format), Java converts this to 2042-07-12, as shown in the third row of the JTable (Fig. 4). An extra check could have prevented this in the code; for the moment being this is not happening, so the user should be cautious.

Figure 4 shows the DateRandomiser panel which allows the user to define the format of the returned date. The format follows the SimpleDateFormat class’s specifications (API doc follows):

Letter	Date or Time Component	Presentation	Examples
G	Era designator	<a href="#">Text</a>	AD
Y	Year	<a href="#">Year</a>	1996; 96
M	Month in year	<a href="#">Month</a>	July; Jul; 07
w	Week in year	<a href="#">Number</a>	27
W	Week in month	<a href="#">Number</a>	2
D	Day in year	<a href="#">Number</a>	189
d	Day in month	<a href="#">Number</a>	10
F	Day of week in month	<a href="#">Number</a>	2
E	Day in week	<a href="#">Text</a>	Tuesday; Tue
a	Am/pm marker	<a href="#">Text</a>	PM
H	Hour in day (0-23)	<a href="#">Number</a>	0
k	Hour in day (1-24)	<a href="#">Number</a>	24
K	Hour in am/pm (0-11)	<a href="#">Number</a>	0
h	Hour in am/pm (1-12)	<a href="#">Number</a>	12
m	Minute in hour	<a href="#">Number</a>	30
s	Second in minute	<a href="#">Number</a>	55
S	Millisecond	<a href="#">Number</a>	978
z	Time zone	<a href="#">General time zone</a>	Pacific Standard Time; PST; GMT-08:00
Z	Time zone	<a href="#">RFC 822 time zone</a>	-0800

*Notice however, that supplying a format is applicable only to the DateRandomiser and not the SQLDateRandomiser which returns a java.sql.Date instance and its format cannot be user-defined.*

## Validation rules:

- Name is required.
- Percentages should sum up to 100.
- “From” date should have a lower or equal value than the “To” date.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.
- It is not possible to exclude all the days. If the dates’ ranges and the excluding days are not in accordance, the generator will simply ignore the excluded days setting. For example, if 6 days are excluded and the generator is asked to generate dates that are 2 days apart that have days which are in the excluded range, the excluded days will be ignored.
- For the case of the DateRandomiser, a valid date format should also be entered; the yyyy-mm-dd is provided by default if the user does not enter anything.

dgMaster: Simple, open source, extensible Data Generator

File Randomisers Tools Help

Case description

Name:

Description:

Percentages of generated values

From:  To:  Percentage:

From	To	Percentage
2001-01-01	2020-12-31	20
1999-01-01	2040-12-31	20
2010-01-01	2042-07-12	20

Excluding days: ☐ Sun ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri

Date format:

Null:

Figure 4, DateRandomiser; notice SQLDateRandomiser, (not shown here), is slightly different.

### 3.1.3 First names generator (FirstnameRandomiser)

The first names generator is used to generate English first names. The “seed value” field offers a way to generate the same sequence of random items, each time the generator is run. This may not be so useful in the first place, but dgMaster uses this simple technique to “synchronise” different generators that may be complementary to each other. An example follows:

The first names and last names generators are complemented by the emails generator. This means, that if the user wants to use all the three generators, (first names, last names, emails generators), the email generator should not generate email addresses that are irrelevant to the generated first and last names. Behind the scene, the emails generator uses the same dictionaries with the first and last name generators. It uses two Math.Random objects for selecting the first and last name, which then concatenates in a random way to create a username; a random domain is also appended after the username generating an email address. If the two Math.Random generators are actually instantiated with seed values that correspond to the first names and last names generators, then the emails generator will actually be generating the similar sequence with those user-defined first name, last name generators. *Simply put, the seed value allows predictability in the randomness of the generated data.* In certain cases, as described here, this can be useful.

If there is a need to have the first names and emails generator in the same data set, then it is feasible to generate emails that have a username which is derived from the generated first name, by simply filling up the seed value. The seed value entered here should also be entered in the emails generator. Of course, this field is optional; if it is omitted, new sequences of first names will be generated each time the generator is run.

The file “en\_firstnames.txt” from which first names are read, is located under the “resources” directory.

Validation rules:

- Name is required.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

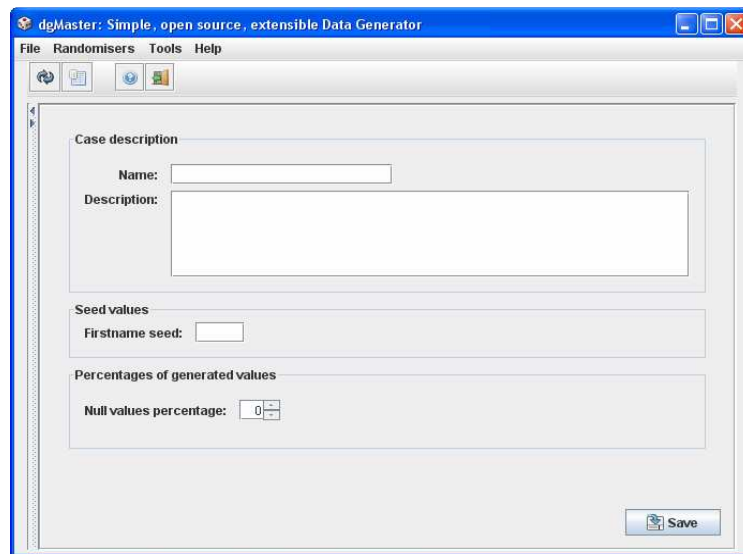


Figure 5, first names generator.

### 3.1.4 E-mails generator (EmailRandomiser)

The emails randomiser in dgMaster generates emails in which the part that comes before the “@” character is formed by English first and last names. These two items are concatenated in one of the following random ways:

- At least one character is selected from the first name.
- At least one character is selected from the last name.
- One of the following characters is inserted between the two generated substrings:”” (empty char.), “.”, “\_”.

If the emails generator is to be used together with the first and last names generators, the seed values in these three generators should be identical. If seed values are not entered, the emails generator simply generates random emails.

The file “TLDs.txt” from which top level domains is randomly chosen, is located under the “resources” directory.

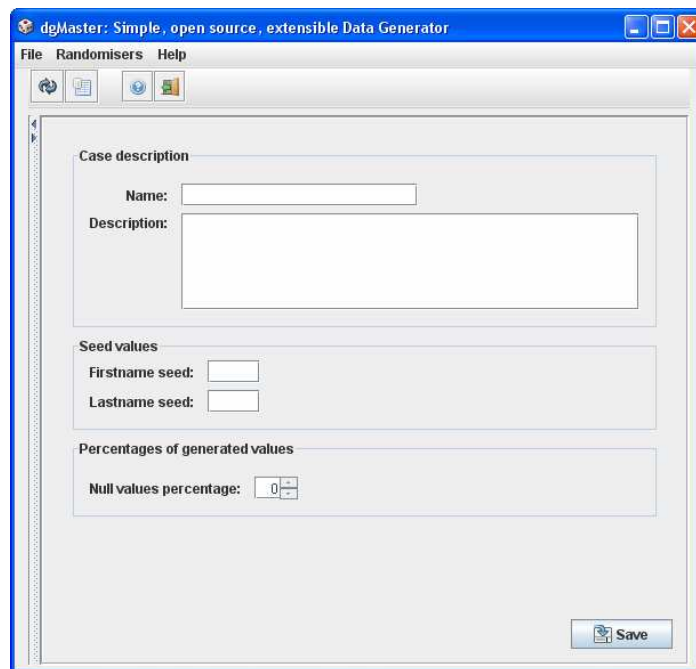


Figure 6, emails generator.

Validation rules:

- Name is required.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

### 3.1.5 English text generator (EnglishTextRandomiser)

The English text randomiser is used to generate English text by means of a dictionary. Consequently, the text is pronounceable, yet the generated sentences do not make any sense. (A Bayesian text generator has also been implemented, but is not currently included in dgMaster. The Bayesian generator can create random text –in any language– that can often make sense as an appropriate text passage.).

The punctuation allows for some punctuation symbols to be inserted. Text starting after the “.”, the “!”, and the “?” will have its first letter capitalised.

Validation rules:

- Name is required.
- Minimum and maximum lengths are required and they should be positive integers. Notice that if the maximum text length is reached the text is cut off, probably not making much sense as an English word.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

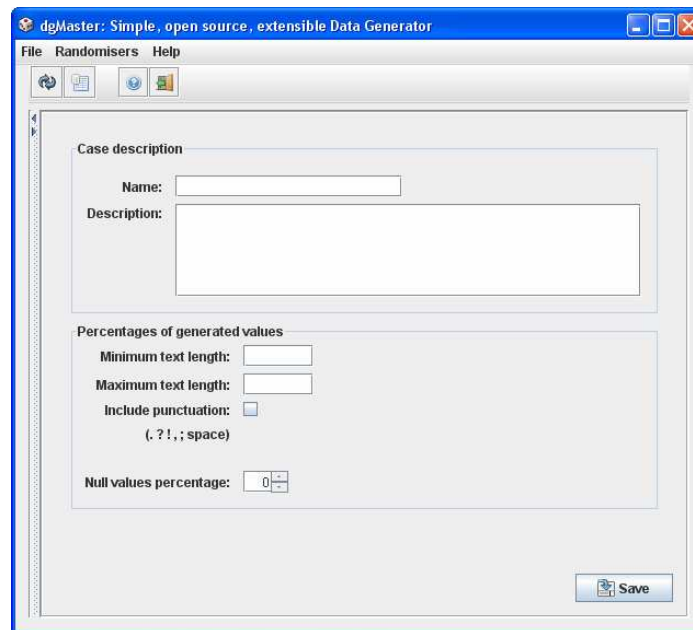


Figure 7, English text generator.

### 3.1.6 English text generator (EnglishWordRandomiser)

This generator is almost identical to the EnglishTextRandomiser; its only difference is that rather than specifying a maximum text length, the user specifies a maximum number of words. As a result, the text is never clipped off.

Validation rules:

- Name is required.
- Minimum and maximum number of words are required and they should be positive integers.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

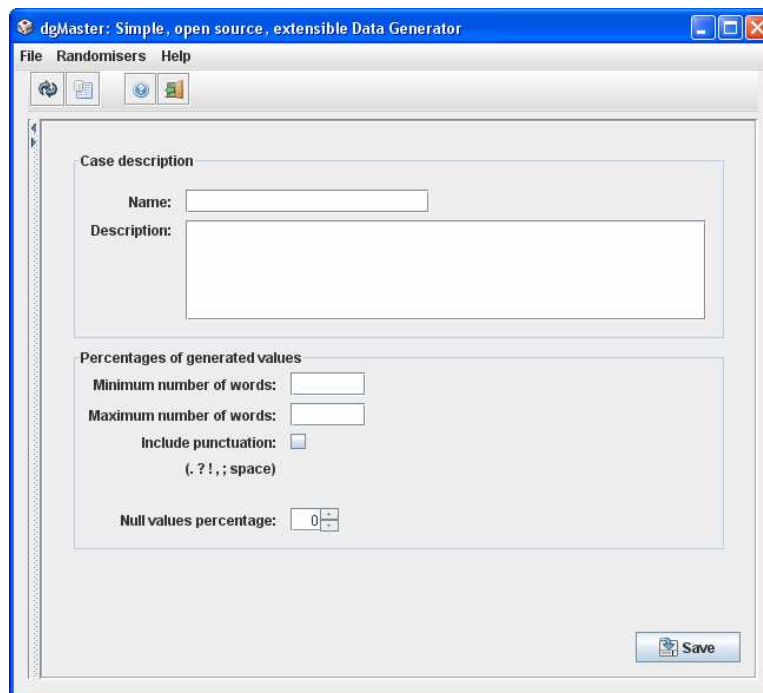
The image shows a screenshot of a software window titled "dgMaster: Simple, open source, extensible Data Generator". The window has a menu bar with "File", "Randomisers", and "Help". Below the menu bar is a toolbar with icons for file operations and a refresh button. The main content area is divided into two sections. The top section, titled "Case description", contains a "Name:" label followed by a text input field, and a "Description:" label followed by a larger text area. The bottom section, titled "Percentages of generated values", contains three input fields: "Minimum number of words:", "Maximum number of words:", and "Null values percentage:". Below the "Include punctuation:" checkbox is a small text label "(. ? ! , ; space)". A "Save" button is located in the bottom right corner of the main content area.

Figure 8, English words generator.



### 3.1.7 English full names (FullnameRandomiser)

This generator was built as a guide for developing a new data generator. This is the reason why it does not have seed value fields. The absence of seed values means that this generator cannot be synchronised with the emails generator. The FullnameRandomiser can be used to generate English full names which may also include a title, and a middle initial name. For example, it is possible to generate values such as “Dr Nick J. Mitchell” or “J. Simon”.

In order to make the data more realistic, the title “Dr” only appears with a probability of 0.3. Similarly, even if the user selects the middle initial, there will be a 0.25 probability that this setting will be ignored. Although there is no way to change these probabilities in the panel, they can still be adjusted by editing the source code (FullnameRandomiser.java).

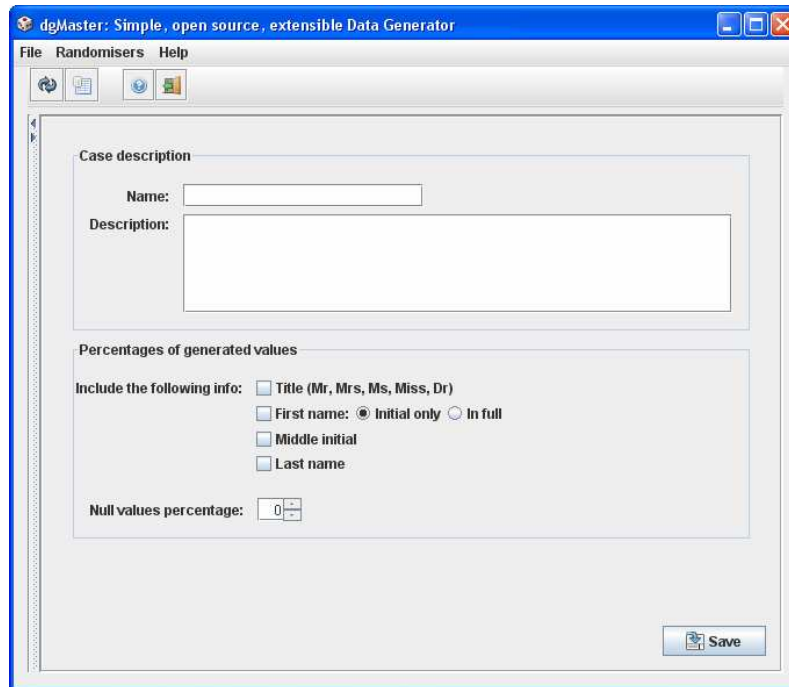


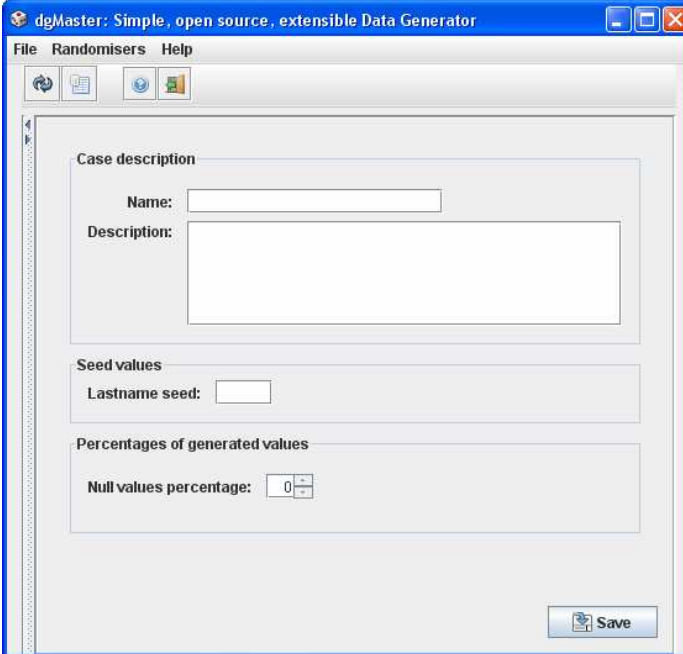
Figure 9, English full names generator.

Validation rules:

- At least one check box must be selected.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

### 3.1.8 English last names (LastNameRandomiser)

This generator can be used to generate English last names. As the form below shows a seed value can be used so as to make a “predictable” random sequence. The seed value entered here can be used in the emails generator, (which uses the same dictionary with the English last name generator), and as a result, the two generators will produce data that are related to each other.



The screenshot shows the dgMaster application window. The title bar reads "dgMaster: Simple, open source, extensible Data Generator". The menu bar includes "File", "Randomisers", and "Help". The toolbar contains icons for refresh, save, and help. The main content area is divided into three sections: "Case description" with "Name:" and "Description:" labels and corresponding input fields; "Seed values" with a "Lastname seed:" label and an input field; and "Percentages of generated values" with a "Null values percentage:" label and a spinner control set to 0. A "Save" button is located at the bottom right of the window.

**Figure 10, English last names generator.**

Validation rules:

- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

### 3.1.9 List items generator

The list items generator can return string values from a user-specific list, (with user-specific distributions), or a certain text file (without user-specific distributions in this case).

Validation rules:

- Name is required.
- Percentages should sum up to 100.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

dgMaster: Simple, open source, extensible Data Generator

File Randomisers Help

Case description

Name: departments

Description:

Percentages of generated values

List data: ☐ Use file  ☒ Use list

New Item: Computer Science

Percent.(optional): 20

Item	Percent.
Engineering	30
Linguistics	30
Law	20
Computer Science	20

Null: 0

Figure 11, list items generator.

### 3.1.10 List items sequencer

The interface of this generator is almost identical to the previous one, (the percentages are not included in the JTable), however, the returned values are not random; as the name suggests they are drawn sequentially from the table or the file (the file format assumes one line per item). If the end of the list is reached while generating data, the array index is initialised and the generator starts generating items from the 1<sup>st</sup> element of the array again.

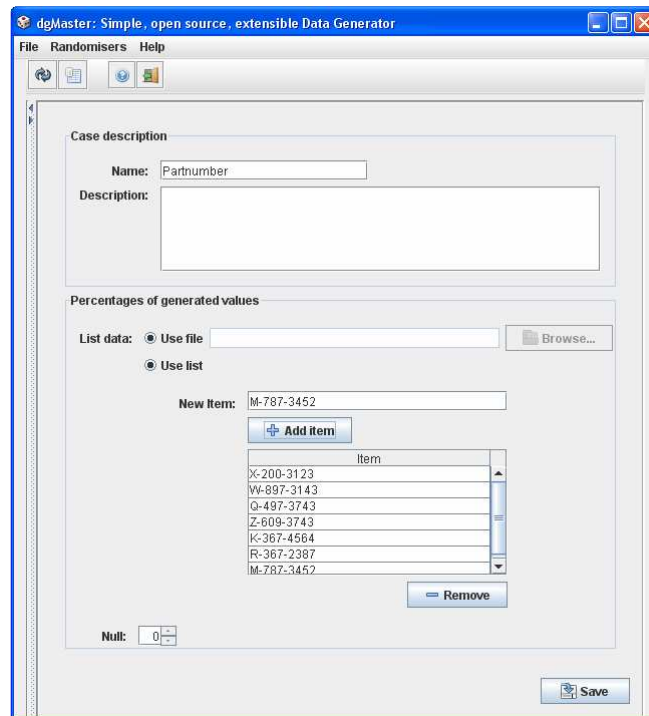


Figure 12, list items sequencer.

Validation rules:

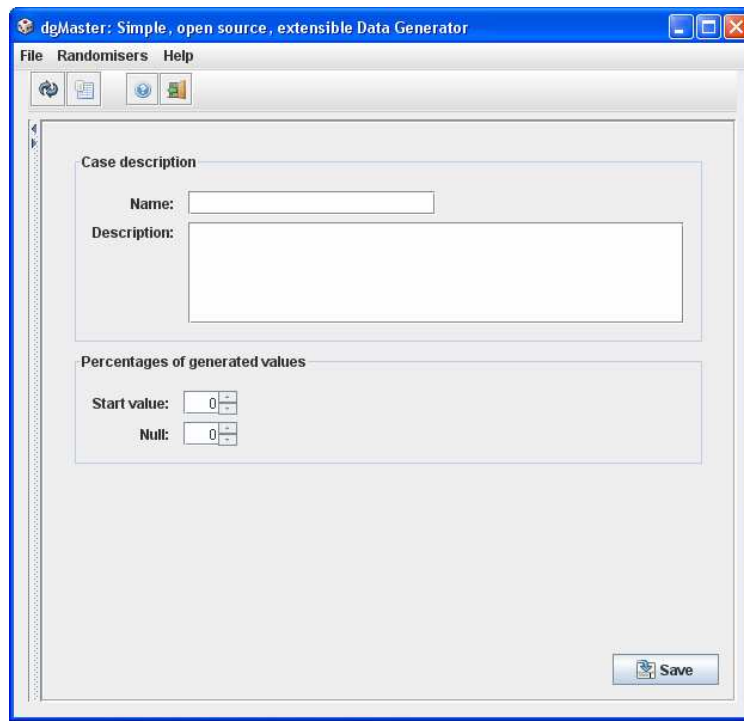
- Name is required.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

### 3.1.11 Long sequencer

As with the previous generator, this one does not generate random values but can rather be used to generate a sequence of numbers, starting from a certain value.

Validation rules:

- Name is required.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.
- Start value should be numeric.



The screenshot shows the 'dgMaster: Simple, open source, extensible Data Generator' application window. The window has a menu bar with 'File', 'Randomisers', and 'Help'. Below the menu bar is a toolbar with icons for file operations and a refresh button. The main area is divided into two sections. The first section, 'Case description', contains a 'Name:' label followed by a text input field, and a 'Description:' label followed by a larger text area. The second section, 'Percentages of generated values', contains two rows: 'Start value:' with a numeric input field set to '0', and 'Null:' with a numeric input field set to '0'. A 'Save' button is located in the bottom right corner of the main area.

Figure 13, long sequencer.

### 3.1.12 Integer numbers (Byte, Short, Integer and Long randomisers)

Byte, short, integer and long numbers can be generated from the ByteRandomiser, ShortRandomiser, IntegerRandomiser and LongRandomiser respectively. These generators feature a custom distribution over the desired values range, as figure 14 shows.

Validation rules:

- Name is required.
- From, to values should be in the each time range of the specific generator.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

The screenshot shows the 'dgMaster: Simple, open source, extensible Data Generator' window. It has a menu bar with 'File', 'Randomisers', and 'Help'. Below the menu is a toolbar with icons for file operations and a help icon. The main area is divided into two sections. The top section, 'Case description', has a 'Name:' field with 'CustomerAge' and a 'Description:' text area. The bottom section, 'Percentages of generated values', contains input fields for 'From:' (91), 'To:' (110), and 'Percentage:' (1). There is an 'Add range' button. To the right is a table with columns 'From', 'To', and 'Percentage'. The table contains the following data:

From	To	Percentage
22	40	50
41	50	30
51	90	19
91	110	1

Below the table is a 'Remove Sele...' button. At the bottom left, there is a 'Null:' field with a value of 0 and a spinner. At the bottom right, there is a 'Save' button.

Figure 14, generic number generators.

### 3.1.13 Real numbers (Double and Float randomisers)

Double and float randomisers are used to generate real numbers with a user-specific distribution. The panel is almost similar to the previous one; however, there is one extra field, the one which is related to the precision of the generated number.

Validation rules:

- Name is required.
- From, to values should be in the each time range of the specific generator.
- Precision should be a positive integer.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

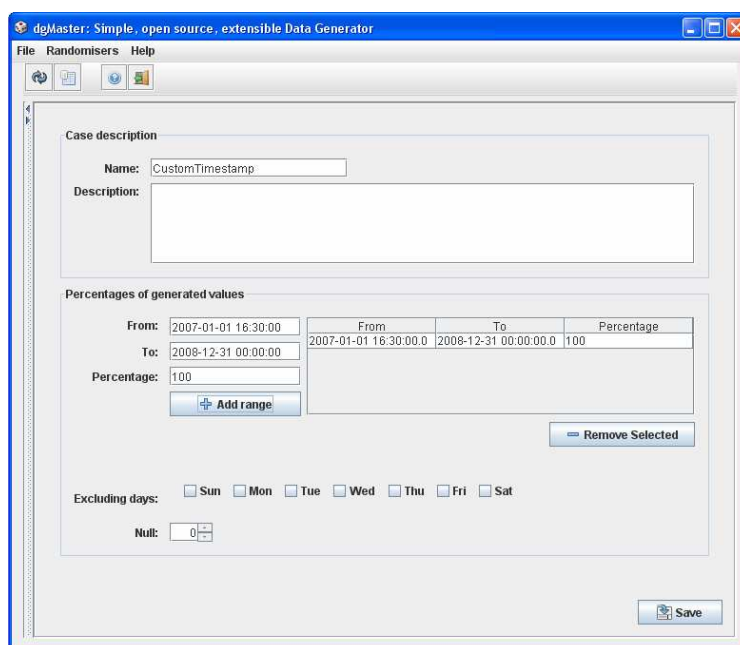
From	To	Percent
0.0	0.5	50.0
0.51	0.6	10.0
0.61	0.7	10.0
0.71	0.8	10.0
0.81	1.0	20.0

Figure 15, real number's generator.

### 3.1.14 Timestamp (Timestamp Randomiser)

This generator can be used to generate items of data type `java.sql.Timestamp`. Values in the “from”, “to” fields should comply with the format “YYYY-MM-DD HH:MM:SS” (year-month, day of month, hour, minute, seconds). Notice it is not possible to change this format, the time stamp is always returned with the specific format. If a different format is required, use the `DateRandomiser`.

*Also pay attention to the fact that entering certain incorrect values may not always trigger an error. Instead of an error, dgMaster will try to convert the given timestamp to a valid timestamp, which may not be what the user had in mind. This is an identical situation to that one explained in section 3.1.1.*



The screenshot shows the 'dgMaster: Simple, open source, extensible Data Generator' window. The 'Randomisers' tab is active. Under 'Case description', the 'Name' field contains 'CustomTimestamp' and the 'Description' field is empty. The 'Percentages of generated values' section contains a table with one row: 'From' is '2007-01-01 16:30:00', 'To' is '2008-12-31 00:00:00', and 'Percentage' is '100'. Below the table are 'Add range' and 'Remove Selected' buttons. The 'Excluding days' section has checkboxes for Sun, Mon, Tue, Wed, Thu, Fri, and Sat, all of which are unchecked. A 'Null' field with a spinner is set to '0'. A 'Save' button is at the bottom right.

From	To	Percentage
2007-01-01 16:30:00	2008-12-31 00:00:00	100

Figure 16, SQL timestamp generator.

Validation rules:

- Name is required.
- Percentages should sum up to 100.
- “From” date should have a lower or equal value than the “To” date.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.
- It is not possible to exclude all the days. If the dates’ ranges and the excluding days are not in accordance, the generator, will simply ignore the excluded days setting. For example, if 6 days are excluded and the generator is asked to generate dates that are 2 days apart that have days which are in the excluded range, these days will be ignored.



### 3.1.15 SQLTime (SQLTimeRandomiser)

This generator is used to generate data of type java.sql.Time. Data in the “from”, “to” values are entered in the 24 hours format (HH:MM:SS).

*Again, the user should be careful when entering the ranges as if values are entered in incorrect format, then the result may not be the desired one. The user can of course check the result by looking at the table. The next figure demonstrates this issue. As it can be seen the time entered is not a valid one, yet in the table, it has been properly converted to a valid time..*

From	To	Percentage
00:00:00	23:59:00	50
11:10:20	11:20:20	50

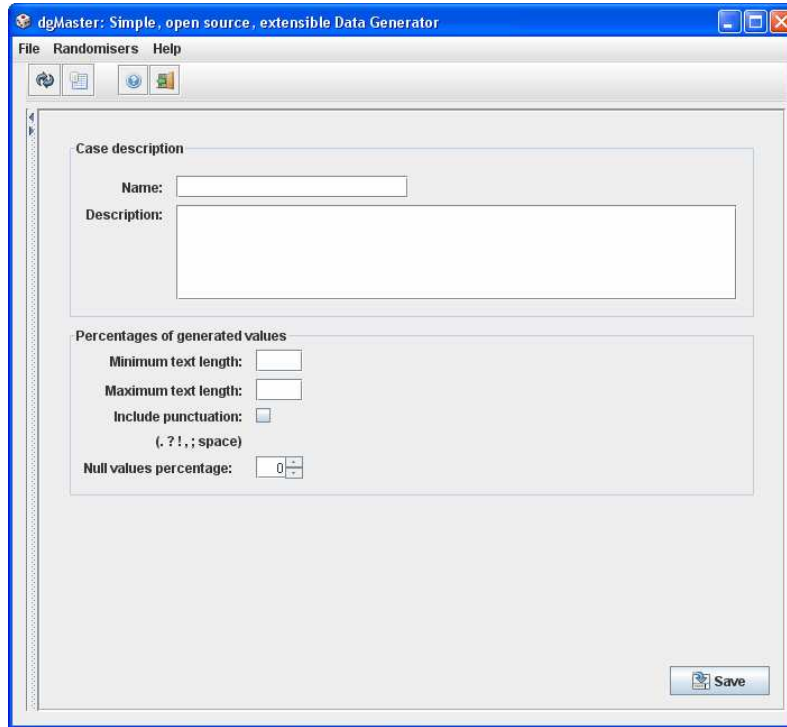
Figure 17, SQL time generator.

Validation rules:

- Name is required.
- Percentages should sum up to 100.
- “From” time should have a lower or equal value than the “To” time.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

### 3.1.16 Random strings (StringRandomiser)

This randomiser generates random (random, as in non-pronounceable) text. If punctuation is included, then the first letter generated after the symbols “.”, “!”, “?”, will be a capital one.



**Figure 18, Random string generator.**

Validation rules:

- Name is required.
- Minimum text length should be less than the maximum text length.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

### 3.1.17 Unique random strings (UniqueRandomiser)

This randomiser generates random strings which are unique. There is no configuration involved apart from the null percentage. The randomiser forms strings with the following format:

*Current time in milliseconds + Capital Character + Integer Counter*

The capital character starts from “A” and in every generation cycle it moves on to the next letter. When “Z” is reached, the next generated item will start all over again and will use “A”. The integer counter starts with 0 and goes on indefinitely. ***When the integer limit is reached, negative values will be generated.***

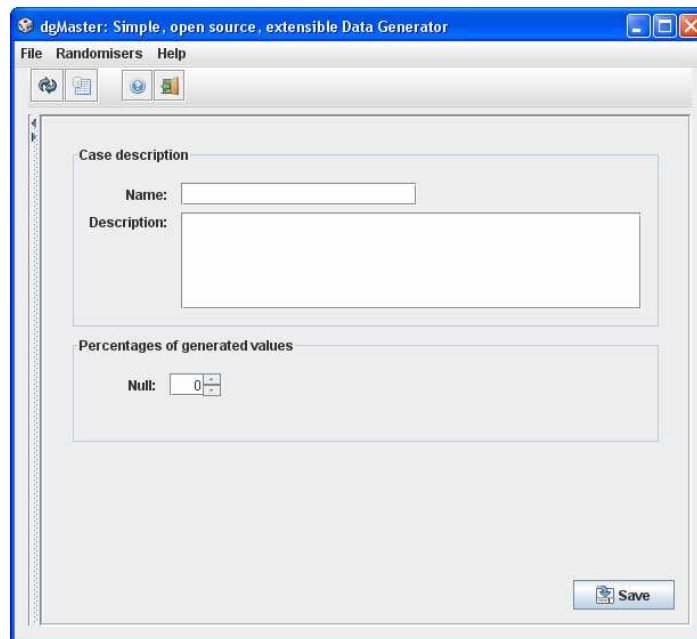


Figure 19, Unique string generator.

Validation rules:

- Name is required.
- “Null” value not required, but if a number is entered it should be non-negative, up to 100.

## 4 Using the Generators

### 4.1.1 Generating text data

Generating data in text format is achieved by selecting “Define text file...” from the “Randomisers” menu (main menu bar), (see next figure).

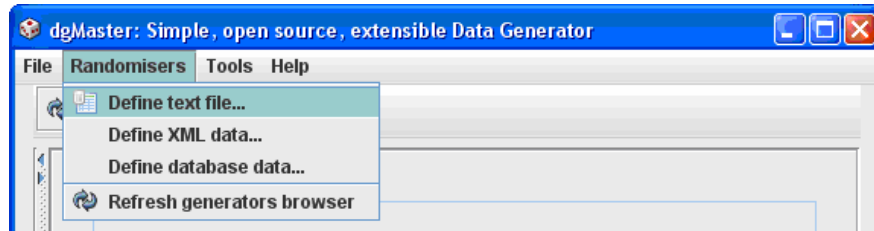


Figure 20, selecting the generate text option.

This option brings up the panel shown in the next figure.

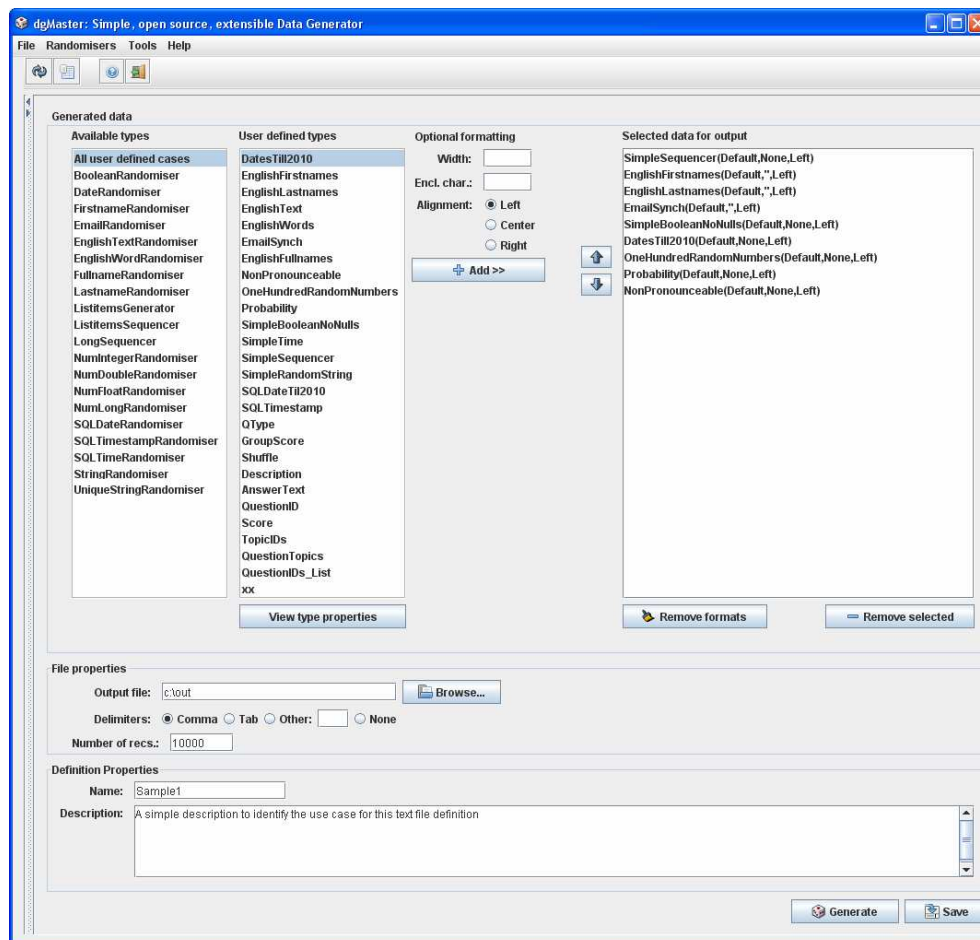


Figure 21, text file options panel.

As fig. 21 shows there are three available list boxes. The leftmost one (*Available types*) shows the installed data types. Clicking on each of those data types populates the next list box (*User defined types*) with user definitions that belong to this specific data type.



***In dgMaster, the user needs to have previously and appropriately set-up a desired set of generators before proceeding with the data generation. Of course, once these generators have been set-up it is possible to use them many times.***

For example, when clicking on the IntegerRandomiser, the middle list box may be populated with values such as “Age”, “NumberOfChildren”, “NoClaimBonus”. Similarly, clicking on the DoubleRandomiser, will populate the list box with values such as “Probability”, “Temperature”, etc. Of course, these user-defined data types, should have been previously defined in the dgMaster database, as explained in section 3.

The user proceeds by selecting the appropriate value from the middle list box and adding it to the right-most one by clicking the “Add” button. It is also possible to provide the width, the enclosing character and the alignment of the field. If the specified width is less than the actual width of the generated value, the user-specific will be ignored, (that is, values are not cut-off). When the field is added, this optional information is added in the description:

Field ( Width, enclosing character, Alignment ), where values are as follows:

- Field : the user defined data type
- Width: Default or a user specific numerical value.
- Enclosing character: None, or a user specific character
- Alignment: Left, Center, or Right.

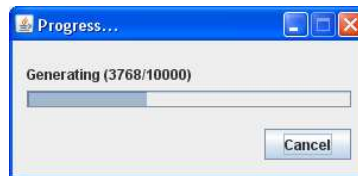
It is possible to change the order of the fields, to remove the formatting of a specific field, (meaning to return its format state to “Default, None, Left”), or to remove entirely a selected field.

The File Properties panel allows the user to define where the output will be saved as well as the number of records to generate. A mandatory name for the data file definition should be entered in the Definition Properties panel; an optional description can also be included here.

When the file definition is completed, the user needs to save this “specification”, by clicking on the “Save” button. Data file definitions are saved in the file “OutputFiles.xml” and appear under the node “Text Files” in the tree browser of the left panel. Saved definitions can be recalled from there by simply clicking on a name.

Data are generated by clicking on the “Generate” button. Initialising the randomisers may take a couple of seconds depending on the amount of work that needs to be done in the constructors of the objects which are loaded dynamically. A small dialog box (similar to the one in figure 22), shows the progress of the generated data and the user can cancel the process of data generation at any time.

If there are any errors during the process, the user is appropriately informed and the data generation process stops.



**Figure 22, progress indicator when generating data.**

THE FOLLOWING SECTIONS ARE NOT YET APPLICABLE  
THIS IS WORK UNDER PROGRESS

#### 4.1.2 Generating database data

