**Data Generator**

**User’s Manual**

Table of Contents

[Introduction 3](#_Toc291182696)

[The Author 3](#_Toc291182697)

[Software License 3](#_Toc291182698)

[The Software 3](#_Toc291182699)

[Software Use 5](#_Toc291182700)

[Getting Started 5](#_Toc291182701)

[File Menu 5](#_Toc291182702)

[Edit Menu 6](#_Toc291182703)

[Database Menu 6](#_Toc291182704)

[Editing the Data Configuration 7](#_Toc291182705)

[Adding a Field 7](#_Toc291182706)

[Editing a Field 7](#_Toc291182707)

[Removing a Field 8](#_Toc291182708)

[Adding/Editing Fields 9](#_Toc291182709)

[Auto Increment 9](#_Toc291182710)

[Character String 9](#_Toc291182711)

[Custom Data 10](#_Toc291182712)

[Custom Weighted Data 10](#_Toc291182713)

[Location 10](#_Toc291182714)

[Name-Gender 12](#_Toc291182715)

[Double 12](#_Toc291182716)

[Integer 12](#_Toc291182717)

[Software Maintenance 13](#_Toc291182718)

[Preparing New Data for Import 13](#_Toc291182719)

[Location Data 13](#_Toc291182720)

[Name Data 14](#_Toc291182721)

[Importing Data 14](#_Toc291182722)

# Introduction

## The Author

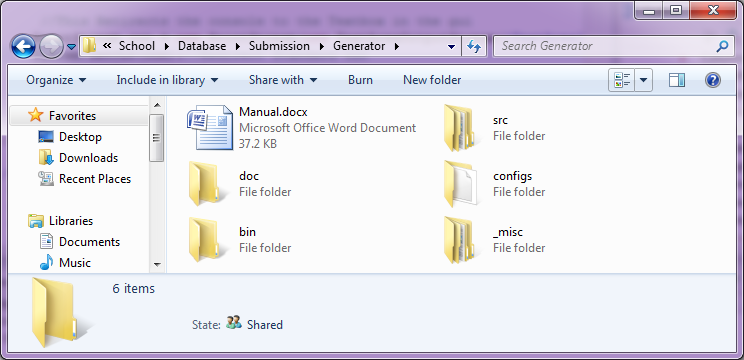
This software was written by Patrick McMorran for the purpose of generating random data sets. Patrick is a computer science major at Southern Connecticut State University. This project is open source, and made available under the Apache License V2.

## Software License

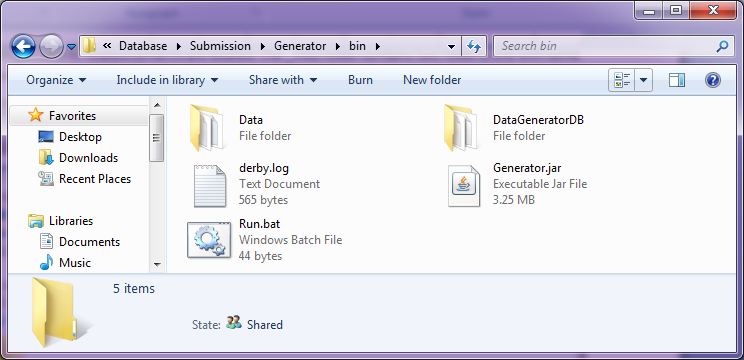
This software is released under the Apache License V2. This software also contains some Apache Software, such as the Apache Derby Database. More information on the project is available at <http://apache.org/>. A copy of the Apache License V2 is included in a text document with this software called LICENSE.txt, and is available online at <http://www.apache.org/licenses/LICENSE-2.0>.

## The Software

Below is a screen shot showing the files and folders inside of the Generator Directory. The src folder contains all of the source code attributed with the project, which can be used to extend the program. The doc folder contains the javadocs that are associated with the aforementioned source code. The configs folder contains some sample configuration files which can be used to load premade configurations into the program, which is used in the tutorials. The bin folder is where the runnable program (jar) and its related contents are located. The \_misc folder contains two folders, one with some sample output data and one with the raw information used for output.



As can be seen below, this is the bin folder. The Generator.jar is a runnable java jar file, which is the actual program. This program uses an Apache Derby Database, which is stored and loaded from the DataGeneratorDB folder. The derby.log file is a log file used by the Derby server. The Data folder contains the data for “Built in” data types. Examples of this are location data and name data, which are used by the program to generate random names and locations. The Run.bat file in the directory is for windows users. Because this program can use a large amount of memory when generating larger datasets, the default virtual machine size is inadequate, and generally crashes. The bat file just sends some parameters to the virtual machine which extends the amount of memory available to it. For Mac users, running the jar with the parameters “-Xms128m -Xmx1024m” (without quotations) will run the program with enough memory available. If you do not have a gigabyte of memory, the program is generally runs under 512mb of memory and for most loads will find changing Xmx1024m to Xmx512m acceptable. This program requires read/write access to the local folder that it is in (the bin folder) because of the database it uses is for both temporary and permanent storage. This means that the program will not be able to function from a CD. Due to the nature of the program, this should ideally be run from a hard drive or USB 2.0 flash drive, otherwise the physical connection speed could become a bottleneck in the program.

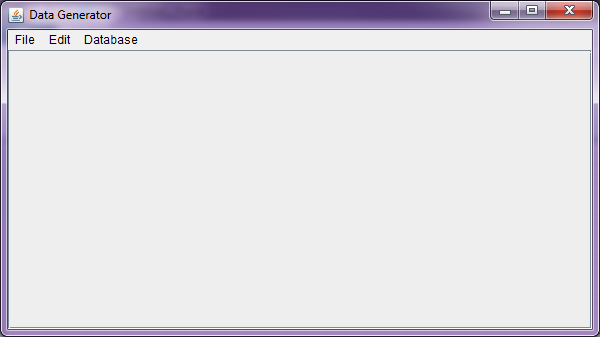


# Software Use

This Software was created for the purpose of generating random data sets for other applications. This application produces several different data types, but has been built with modularity in mind so that other data types can be added.

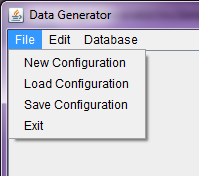
## Getting Started

First you should launch the runnable jar file, ideally with the parameters mentioned in the introduction. For windows users, just launch the Run.bat file to start the program. Below is an image showing the main screen of the data Generator. This screen offers three menus.



### File Menu

The File menu offers options for saving and loading different data configurations.



#### New Configuration

The New Configuration button unloads the currently open configuration, allowing for the user to then add fields to create new configurations.

#### Load Configuration

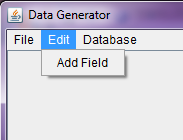
This opens a file dialog which allows the user to select previously saved dcfg files, the format of saved data configuration files.

#### Save Configuration

This opens a file dialog which allows the user to save their currently open configuration as a dcfg file for later use.

### Edit Menu

The Edit menu is used for modifying the current configuration, and offers the add field option.

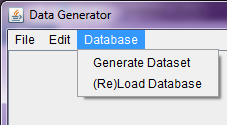


#### Add Field

This launches the Add Field Dialog, which gives a list of options available fields that can be added to the dataset configuration.

### Database Menu

This menu is used for running commands which interact with the database.



#### Generate Dataset

This option is selected when the user wants to begin the process of generating data that meets their configured criteria.

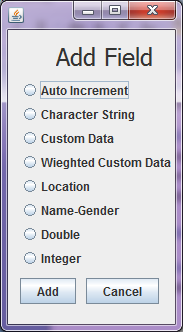
#### (Re)Load Database

This option is selected when the user wants to import new data into the database. For instance, the data currently imported is from the year 2000 census, and a user might wish to update this data with that of the 2010 census.

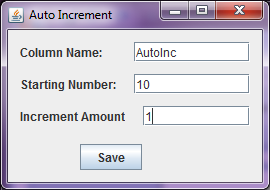
## Editing the Data Configuration

### Adding a Field

To add a Field, select Edit 🡪Add Field.



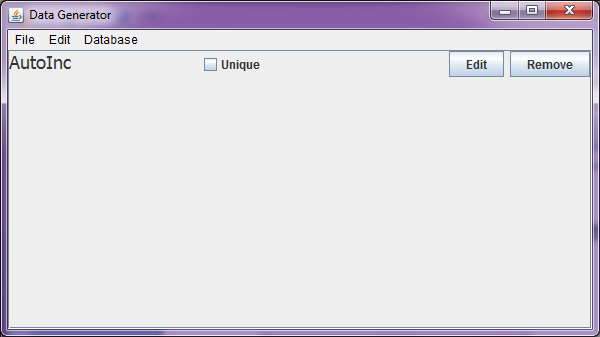
This dialog lists the various types of fields that can be added. The user should select their desired field and click the add button to add it to the configuration. If this dialog was launched mistakenly, clicking cancel or the “x” will close the dialog without affecting the current configuration.



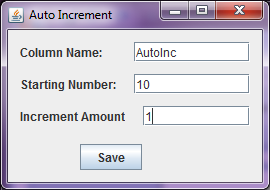
Upon selecting the add button, an edit dialog will pop up requesting that the user input parameters for the field. After filling out the required information, clicking save will add the field to the configuration.

### Editing a Field

Below shows the configuration after adding a field to the dataset configuration. If the user wants the field that is generated to be unique, (So that all information in that column does not repeat) checking the box that says Unique. To edit other parameters of the field, select the edit button.

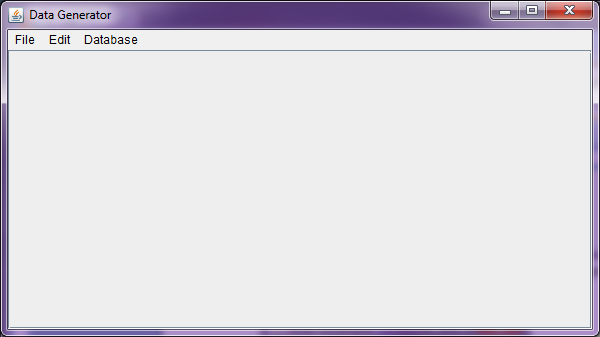


Doing so launches the configuration menu, where changes can be made. Clicking save applies the new settings to the field.



### Removing a Field

To remove a field, the user just needs to click the Remove button on the field, which will result in it being deleted from the configuration.

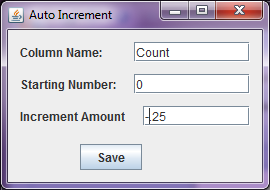


## Adding/Editing Fields

This program uses “Fields” to represent the different data types it can generate. Each data type has unique parameters to configure them, and some in common, such as the unique property and the field name. The Field name property is used to generate the Field (or column) title, which is printed in the first line of the data file. This Property is referred to as the Column Name, and is also used to help the user identify different fields in the configuration. Below each Field are a sample configuration and the resulting sample data. **WARNING:** When selecting the unique property on a field, ensure that the field has enough possibilities for the desired number of records you wish to produce. For instance, if you use a Custom Data Field with only 4 unique records, you will be unable to produce 10 records with it.

### Auto Increment

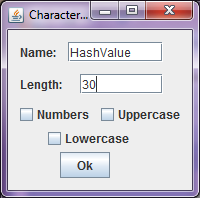
The Auto Increment Field starts at a given number and increments the number for each time a record is made by the amount configured. This Field has two unique configuration parameters. First is the Starting Number, which is the number (integer or double) which the field will start from. The Increment Amount is the number used to increment the starting number. Note that to make this an Auto Decrement field, just using a negative number in the increment amount does the trick.



|  |
| --- |
| Count |
| 0 |
| -0.25 |
| -0.5 |
| -0.75 |
| -1 |
| -1.25 |
| -1.5 |
| -1.75 |

### Character String

The Character String Field generates static length strings composed of the character types selected by the user. This has 4 unique fields. Length is the number of characters the string should be. The other three fields are checkboxes which are used to define what character types should be used in the String. Should no types be selected, the field will use all three types by default.

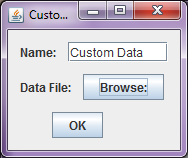


|  |
| --- |
| HashValue |
| dzczSuctBur9vtyJSvJvAWnRtqPtUj |
| udYnosxxHYpUunouqiaSY2jEeMKOUN |
| IOSvELrIG5pJti1Jtif6vIbqXSDof3 |
| MtEcVrnBxnIZcz9gAkESXzqhpgvVY6 |
| BmQeSXAwVavXpgbm8SP57zE3vt5ADA |
| NtuHMeap1Sh5duGDvr7YL6MtiVZVgM |
| tdmQvrwuDf6RwKNDFmtujIrJWYxRGT |
| QCTVEpHtSvQYUgsnhPhaxyRURmKrsV |

### Custom Data

The Custom Data Field uses the values from a defined Comma Spaced Value file. This Field is defined by selecting the browse button. In the case that the CSV has multiple columns, the program uses the first column of the file.

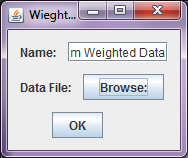
|  |
| --- |
| Custom Data |
| Babby |
| Pappy |
| MILPY |
| Mappy |
| Mappy |
| Pappy |



### Custom Weighted Data

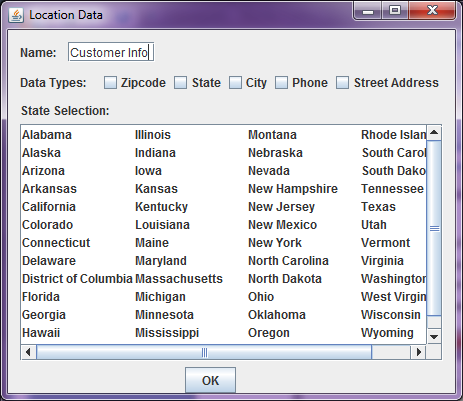
The Custom Weighted Data Field is similar to the Custom Data Field, but it also reads in the corresponding weight values of each value in the CSV and returns values reflecting the weight distribution. The first column of the CSV file should be the values, and the second column should be the corresponding weights of each.

|  |
| --- |
| Custom Weighted Data |
| Very Likely |
| Very Likely |
| Very Likely |
| Very Likely |
| Very Likely |
| Very Likely |
| Kinda Likely |



### Location

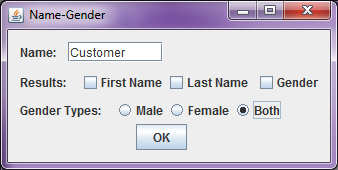
The Location Data Field is likely the most complex data fields offered by the generator. This field can generate up to 7 different columns of data at once, all of which are related to each other. That is to say that the zipcode will be in the correct town and state, with a street name that exists within the town and a phone number who’s area code and prefix also correspond to the zipcode. The number of records generated for each zipcode is distributed by population, which results in a population distribution that reflects the actual United States. The Check boxes are to define which columns the user would like the generator to create. The Unique property is special on this, is unique applies to the field as a whole, not each individual property. For example, if it is supposed to generate zipcode, state and address, then there will never be a record generated that has all three of those columns the same. However, a record could have a zipcode and state that matches other records, but their addresses would be different, ect. The Location field can generate data of the entire USA or of specific states. If the user wishes to select multiple states, they should just hold the CTRL key on the keyboard, and click each state they desire to highlight them. Clicking a highlighted state will deselect it. Because distribution is based off of population, records of each state will be relative to each other. For instance, if the user selected both Connecticut and Texas, then Texas would have more records due to its greater population. If the user does not define which columns the Field should produce, with will print them all by default. If the user does not select any states to use for the location’s scope, then it selects all of them.



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Customer Info-zipcode | Customer Info-state | Customer Info-city | Customer Info-areacode | Customer Info-prefix | Customer Info-extension | Customer Info-street |
| 8109 | New Jersey | MERCHANTVILLE | 609 | 792 | 4702 | 4 STARR |
| 68933 | Nebraska | CLAY CENTER | 402 | 762 | 2544 | 1 CENTRAL |
| 99587 | Alaska | GIRDWOOD | 907 | 754 | 8405 | 129 GENERAL DELIVERY |
| 36703 | Alabama | Selma | 334 | 876 | 2095 | 3 WATER |
| 97801 | Oregon | Pendleton | 541 | 379 | 3685 | 45 OBRIEN |
| 75169 | Texas | WILLS POINT | 903 | 560 | 2547 | 11 VZ COUNTY ROAD 3728 |
| 38632 | Mississippi | HERNANDO | 662 | 449 | 2919 | 12 PINE |
| 85543 | Arizona | PIMA | 928 | 485 | 8683 | 37 MANER |
| 92504 | California | Riverside | 951 | 776 | 7354 | 59 ACAPULCO |
| 2891 | Rhode Island | WEEKAPAUG | 401 | 596 | 3597 | 4 MARY LOU |

### Name-Gender

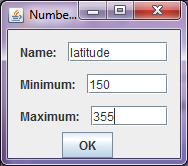
The Name-Gender Field produces up to 3 columns, First Name, Last Name, and Gender. This Field ensures that the First Name and Gender columns correspond so that males receive male names, ect. If the user does not select any of the columns it will generate them all by default. The user can also configure what gender it produces, so that a single gender is produced instead of both.



|  |  |  |
| --- | --- | --- |
| Customer-First | Customer-Last | Customer-Gender |
| ARTHUR | Rivera | Male |
| DAVID | Turner | Male |
| JODY | Phillips | Female |
| HUNG | Miller | Male |
| JOSEPH | Baptiste | Male |
| BRENT | Boggs | Male |

### Double

The Double Field produces random double values within a given range. The minimum field is the lowest double the generator should produce and the Maximum is the highest number the field should produce.



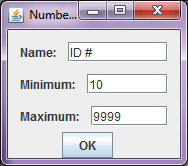
latitude,  
237.8118751734037  
336.53443643182845  
350.8332309114435  
283.8096914928432  
353.9992357723195  
242.7945352630614  
260.0052193921041

,

242.18008963127065,

### Integer

The Integer Field produces random integer values within a given range. The minimum field is the lowest integer the generator should produce and the Maximum is the highest number the field should produce.



|  |
| --- |
| ID # |
| 7409 |
| 8383 |
| 7399 |
| 9444 |
| 4264 |
| 3902 |
| 5022 |

# Software Maintenance

This section has some information that can be used to maintain the data generator, such as updating the built-in information.

## Preparing New Data for Import

Data that will be used by the generator has to be formatted and prepared ahead of time, before it is imported. The import is a simple loading of tables that have been exported to CSV files.

### Location Data

Location data is split between two tables (or files). The first table builtin\_location.csv is a table that contains records with Zipcode, Population, State, City, Phone Area Code and Phone Prefix in that order. The CSV is formatted in that order. There should be a full record for each Phone prefix. Below is a sample section of how the CSV file is formatted.

"01001",16576,"Massachusetts","AGAWAM","413","530"  
"01001",16576,"Massachusetts","AGAWAM","413","237"  
"01001",16576,"Massachusetts","AGAWAM","413","822"  
"01001",16576,"Massachusetts","AGAWAM","413","821"  
"01001",16576,"Massachusetts","AGAWAM","413","789"  
"01001",16576,"Massachusetts","AGAWAM","413","786"  
"01001",16576,"Massachusetts","AGAWAM","413","636"  
"01001",16576,"Massachusetts","AGAWAM","413","575"  
"01001",16576,"Massachusetts","AGAWAM","413","531"  
"01001",16576,"Massachusetts","AGAWAM","413","896"

The second file is a list of Zipcodes and their corresponding street names. The first column is the Zipcode, and the second is a Street name located within the Zipcode. There should be a full record for every Street Name. Below is a sample of how the CSV is formatted.

“01001”,“WORTHINGTON BROOK”  
“01001”,“WRENWOOD”  
“01001”,“WRIGHT”  
“01001”,“WYMAN”  
“01001”,“YARMOUTH”  
“01001”,“YORKSHIRE”  
“01001”,“ZACHARY”

“01002”,“ADAMS”

“01002”,“ALLEN”

“01002”,”ALLEN MILL”

“01002”,“ALPINE”

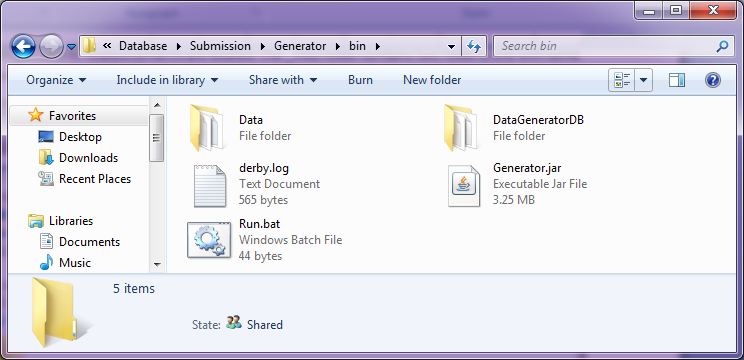
It should be noted that these two tables should be checked to ensure that both Zipcode columns do not contain keys that do not exist in the other table. If the builtin\_location table contains a Zipcode that has no records in the other table, an error will occur causing the generation will fail.

### Name Data

The Name data is split between three files, builtin\_femalenames.csv, builtin\_lastnames.csv, builtin\_malenames.csv. Each file is formatted to contain a name in the first column, and the name’s respective weight in the second column. Below is a sample of how the CSV files are formatted.

"JOHN",3.27  
"ROBERT",3.14  
"MICHAEL",2.62  
"WILLIAM",2.45  
"DAVID",2.36  
"RICHARD",1.70  
"CHARLES",1.52  
"JOSEPH",1.40  
"THOMAS",1.38  
"CHRISTOPHER",1.03

## Importing Data

Once the data is prepared with the correct format and file names, inside the bin folder (as seen below) there is the Data folder. Place the data into the Data folder, replacing any data that is there. 

Then launch the application as noted in the introduction, and select Database🡪Generate Dataset. This will launch the data loader application, which will delete the database and start creating a new one. If this process is canceled, the database will be incomplete and the generator’s functionality will be hindered in terms of location data and name data.