Random Generator

Publisher: Sagui Itay (saguiitay@hotmail.com)

Website: http://www.saguiitay.com

Overview

Random Generator is a utility library that allows you to easily add random names to your game.

The library includes a large variety of random generators, including:

- City Names
- Fantasy Names (for various races - dragonborn, dwarfs, elves, gnomes, goblins, halflings, human and orcs)
- Modern Names (for various nationalities - American, Dutch, French, German, Icelandic, Indian, Irish, Italian, Japanese, Russian, Spanish and Swedish)
- Planet / Constellation Names

The library is easily customizable:

- You can add move flavors (races, nationalities, etc.)
- Variations (First/Last name, etc.)
- Randomizers (Selector and Markov are already included)
- Customize existing generators (raw data is available in the source code)

How to use Random Generator

1. Add a Definition class to your scripts

Add the relevant Definition class to your script, along with a Random class:

```
private CityNameGenerator generator = new CityNameGenerator();

private System.Random m_random;
```

2. Generate random values

In your code, add the following snippet to generate a new random value:

```
generator.Generate(m_random);
```

FAQ

How do I generate specific values?

Each generator provides 2 properties – SupportedTypes and DefinitionFormats:

SupportedTypes
Supported types can be used to select a subset, or less generic value. For example, when generating the names of a person, the supported types might be Male or Female. When generating a name of a Fantasy character, you can choose the Race (e.g. Elf, Dwarf, Orc) of the character.

DefinitionFormats
DefinitionFormats can be used to select a specific structure for the value. For example, when generating a name, you can choose between “FirstName LastName” and just “FirstName”.

How can I create a custom Generator?
The simplest way to do so would be to start with an existing Generator, and customize it to your need. I suggest you start with the `CityNameGenerator` class - it has the following implementation details:

- Supports multiple folders (for different nationalities)
- Supports multiple DefinitionFormats
- Contains both Item and Markov PartDefinitions

What are PartDefinitions you ask?
Each generated value is created from one or more part (as defined in the selected DefinitionFormat). Each part is created from a PartDefinition.

A PartDefinition can be based on a list of predefined values, one of which will be randomly selected. Alternatively, a PartDefinition can be based on a Markov model – a prediction-based data structure that generate random values by sampling a list of known values.