plasmic Documentation

Release 0.1.0

The Meme Factory, Inc.

April 15, 2016

Contents

Plasmic parses a very Python-like mini-language to produce a dictionary. This allows Python programs to be configured using a Python-like language.

Here is a sample configuration:

```
{section1:
  {key1: "value1",
   key2: 2},
 section2:
  {key1: "othervalue",
   key3: 3.0
   subsection1:
    {key4: (1, 2, 3)
     key5: {"foo": "meta",
            "bar": "syntactic"}
    }
  }
}
```

The Plasmic language consists of dictionaries, tuples, and literal values. Plasmic dictionary keys must be literal values. Plasmic dictionary values can be literal values and, nested when desired, tuples and plasmic dictionaries.

Plasmic also does interpolation. This allows values to be designated once and used in multiple places within a configuration. Plasmic replaces reference to configuration keys with the key's value.

The recommended suffix for files written in the Plasmic language is: `.pcf`

CHAPTER 1

Contents

1.1 Why Yet Another Config Syntax?

The short answer is "Because there's nothing like Python's syntax for expressing Python data." Some syntaxes don't attempt to map to Python data types, your program must manually handle type conversion. Other syntaxes are serializations of data expressed in other languages. These are more expressive, both in data type and data structure complexity, but because they are syntaxes designed to express data it remains for an interpolation mechanism to be tacked on. Python syntax is the ideal choice for Python data representation and manipulation.

Plus, Python exposes it's own internals making it easy to write powerful Python-like parsers that are, at the same time, safe. Plasmic configuration files may look like Python but they cannot be abused to execute arbitrary code or, for that matter, any sort of code at all.

1.2 More About Syntax

The plasmic Python-like language contains almost no expression evaluation. The single, optional, expression allowed is a `string.Formmatter.format()` formatting syntax used for interpolation – the insertion of of data into values by reference to Plasmic dictionary keys.

As a convenience all dictionary entries keyed by symbols have their keys converted to the lower case string equivalent. The following 2 examples are equivalent:

Note: The following is here only as an example. It is bad practice in Plasmic to write symbols containing upper case letters.

A configuration with symbols for keys:

```
{section1:
  {key1: "value1",
   key2: 2},
 Section2:
  {Key1: "othervalue",
   KEY3: 3.0}
}
```

The identical configuration with strings for keys:

```
{"section1":
  {"key1": "value1",
```

```
   "key2": 2},
 "section2":
  {"key1": "othervalue",
   "key3": 3.0}
}
```

Although the symbols are written in mixed case the end result is a dictionary keyed with strings in lower case.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search