ValidatingConfigParser Documentation

Release 0.0.1

Markus Juenemann

October 14, 2012

CONTENTS

CHAPTER

ONE

INTRODUCTION

The *validatingconfigparser* module provides the class *ValidatingMixIn* which is meant to be used as a mix-in class for the *RawConfigParser*, *ConfigParser* and *SafeConfigParser* classes found in the Python Standard Library. It extends the *get()*, *getint()*, *getfloat()*, *getboolean()* and *set()* methods by an additional keyword argument *validator*. This *validator* can be used to ensure that the *value* returned or given as argument to above methods passes a validation test. The *validator* must be an instance of a class that provides the same methods as the validators of the *Formencode* project. In fact, *ValidatingMixIn* was specifically designed to use the *Formencode.validators*.

```
from ConfigParser import ConfigParser
from validatingconfigparser import ValidatingMixIn
from formencode.validators.import OneOf

# Ensure that the value is either 1, 2 or 3.
validator = OneOf([1, 2, 3])

# Create a new parser with the ValidatingMixin.
class ValidatingConfigParser(ValidatingMixIn, ConfigParser):
    pass

parser = ValidatingConfigParser()
parse.read("settings.conf")

# Below will raise formencode.Invalid if validation fails.
parser.get("name", "section", validator=validator)
```

As it may be tedious to create your own validating *ConfigParser* sub-class as shown above, the *validatingconfigparser* module already provides validating variants of the original parsers for convenience. These parsers can be used as direct replacements of the original parsers as their API is compatible.

- *validatingconfigparser.RawConfigParser*
- *validatingconfigparser.ConfigParser*
- *validatingconfigparser.SafeConfigParser*

In addition the *__init_()* methods of the validating parser listed above accepts a *schema* keyword argument which must be an instance of a *Formencode.Schema* class.

Contents:

CHAPTER

TWO

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*