Verata Documentation

Release latest

Sep 14, 2017

Contents

Contents:

CHAPTER 1

Installation

Requirements

- Python 2.7, 3.4, 3.5, or PyPy

Optional dependencies

- *lxml* library inside system if you consider using it as a parser

Installing

```
pip install verata
```

CHAPTER 2

Usage

Pattern matching for Pages

Why is it needed? Pattern matching checks URL and understands which page(s) logic we need to apply for concrete case.

Simplified wild cards are used for this task. Syntax is listed bellow:

Symbol	Translates to	Explanation
%	.*?	Matches N random symbols
*	.	Matches single random symbol

Path matching for data retrieval

It is used to extract data from page. Simplified xpath-alike expressions are used for this task.

General syntax looks like:

<TAG>[<ATTRIBUTE>=”<VALUE>”]/<TAG>[<ATTRIBUTE>=”<VALUE>”]/...

Each “/” symbolizes new child matching, part inside “[<...>]” is optional, it can be skipped if you have no intention to query inside these tags , eg.: if you want to get *all* links in the page.

For selecting concrete element from matches array, use “{#n}” syntax, eg. if we have :

```
<div class="menu">
  <a href=#1>Link1</a>
  <a href=#2>Link2</a>
  <a href=#3>Link3</a>
</div>
```

following query can be used to retrieve “Link2”: div[class=”menu”]/a{1} where “a{1}” indicates that we want to retrieve all links, and get 2nd link from that array

To select concrete data from element, attribute selector can be used. Notation starts with a '.' followed by attribute name (eg. href, class, src etc.) Retrieving concrete link would look like: div[class="menu"]/a{1}.href

CLI

Verata can operate in two modes:

1. as crawler - reading all links and traveling through whole page
2. as scraper - just getting data from concrete link

General params

Param	Explanation
–env	Path where your environment file lives
–log-level	Setup how much logging info you want to see
–debug	Shortcut to –log-level=DEBUG
–output	File where output will be put
–config	Config file

Operating as crawler

Param	Explanation
–paginate	Size of read chunks (default: 10)
–rest_interval	Time to rest after each chunk

Operating as scraper

Param	Explanation
–link	Link to page you want to scrape

CHAPTER 3

Configuration

Config

Value	Notes	Description
name		A user-friendly name for config
description		Description about config
site_root		Site url which we will be scrapping
start_page		Where to start our scrapping
cookies	(Optional, Array)	Add cookies to our requests
headers	(Optional, Array)	Add headers to our requests
proxies	(Optional, Array)	Add http proxies to our requests
pages	(Array)	Consists an array of *Page*
parser	(Optional)	Select parser for web pages.
auth	(Optional)	*Auth* object

```
name: Python Org scrapper
description: Just scrape it for testing
site_root: https://www.python.org
start_page: /blogs
cookies:
  authToken: abc1234
  remember: true
headers:
  "User-Agent": "Mozilla/5"
auth: ...
pages:
  - ...
```

Auth

```
url: /login
method: POST
params:
  user: {{ secret_user }}
  password: {{ secret_password }}
```

Value	Notes	Description
url		Login url
method	(Optional)	Default: POST, you can add any other http method
params	(Array)	Key-Value pairs for request

Page

Value	Notes	Description
name		Name of page
link_pattern		Pattern which allows to detect which page parser to use
mappings	(Array)	Array of *Mapping*

```
- name: Blog
  link_pattern: /blog%
  mappings:
    - ...
```

Mapping

Value	Notes	Description
name		Name of mapping
path		Path to element

```
- name: title
  path: h3[class="event-title"]/a
```