Web Extract-1.930.1927

Need help?

Technical contact to tech@argos-labs.com

May you search all operations,

- **Actions**
- **Verifications**
- **System Calls**
- **Interactives**

Web Extract

Web Extract

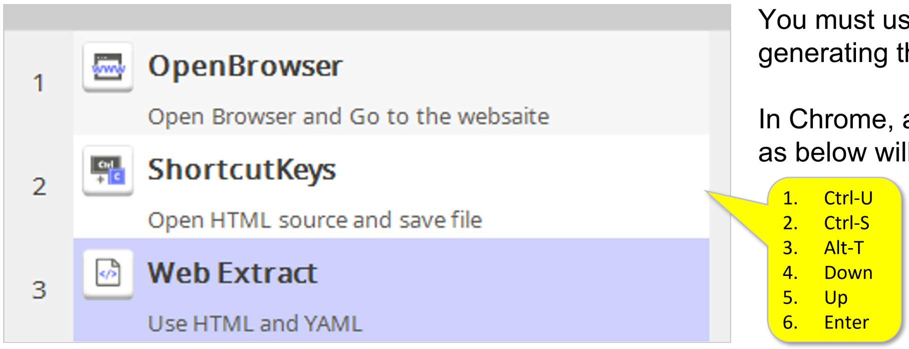
- You can build a bot to extract data from websites (Web Scraping) using this tool.
- In order to use this operation, you must have a knowledge about HTML and YAML.

Contents

1. This operation is used after extracting the HTML source file from your browser.

You must use Web Extract plugin after generating the HTML source file

In Chrome, a sequence of shortcut keys as below will generate the HTML source



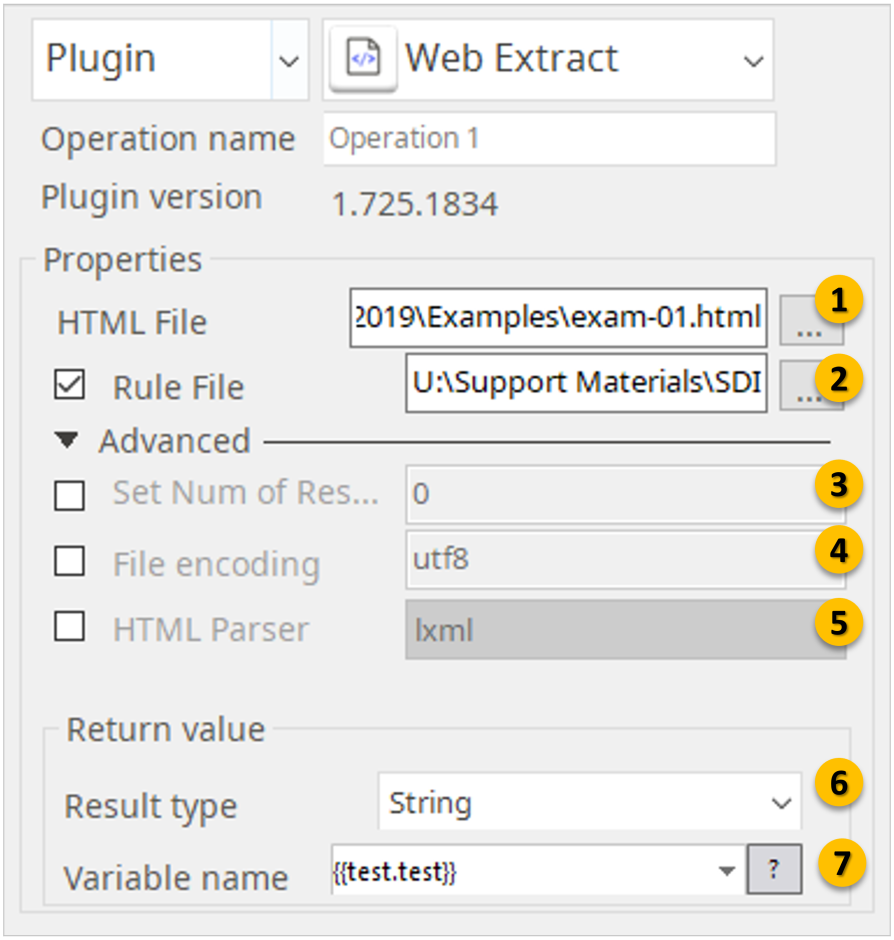
1 OpenBrowser
Open Browser and Go to the websaite

2 ShortcutKeys
Open HTML source and save file

3 Web Extract
Use HTML and YAML

1. Ctrl-U
2. Ctrl-S
3. Alt-T
4. Down
5. Up
6. Enter

2. The Parameters.



Plugin: Web Extract

Operation name: Operation 1

Plugin version: 1.725.1834

Properties

HTML File: 2019\Examples\exam-01.html

☑ Rule File: U:\Support Materials\SDI

Advanced

☐ Set Num of Res...: 0

☐ File encoding: utf8

☐ HTML Parser: lxml

Return value

Result type: String

Variable name: {{test.test}}

1) Specify your HTML Source file here.

2) Specify your Rule file (YAML) here --- always check the check-box --- this file is mandatory.

3) If your data has many occurrences, you can limit the # of data to be extracted by setting the number here (0 means no limitation = default).

4) Define preferred encoding standard of your HTML file here – if your choice does not work Web Extract will go to auto-detect mode.

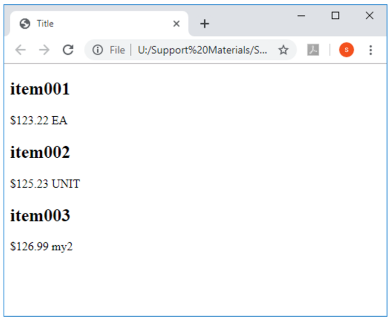
5) Define the HTML parsing standard here or leave it unchecked for auto detect mode.

6) Choose your output format (String, CSV, or File).

7) You must set your variable at Settings menu in the Main menu.

3. A simple example below should help you build the web scraping bot.

- Below is your target website page.
- And then below is the HTML source file.



item001

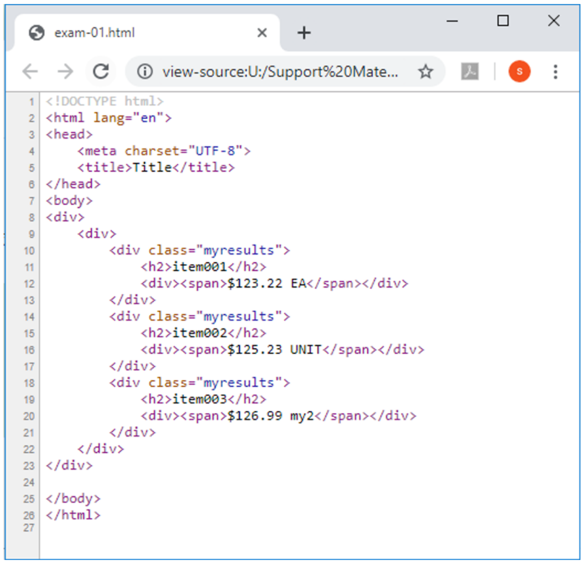
$123.22 EA

item002

$125.23 UNIT

item003

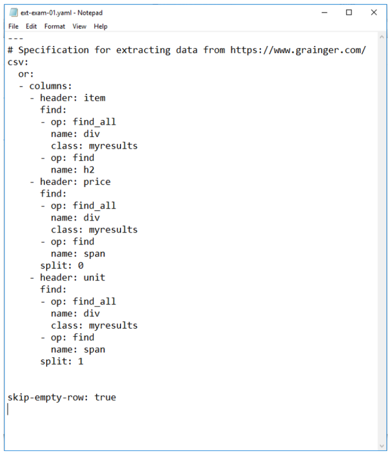
$126.99 my2



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
<div>
    <div>
        <div class="myresults">
            <h2>item001</h2>
            <div><span>$123.22 EA</span></div>
        </div>
        <div class="myresults">
            <h2>item002</h2>
            <div><span>$125.23 UNIT</span></div>
        </div>
        <div class="myresults">
            <h2>item003</h2>
            <div><span>$126.99 my2</span></div>
        </div>
    </div>
</div>

</body>
</html>
```

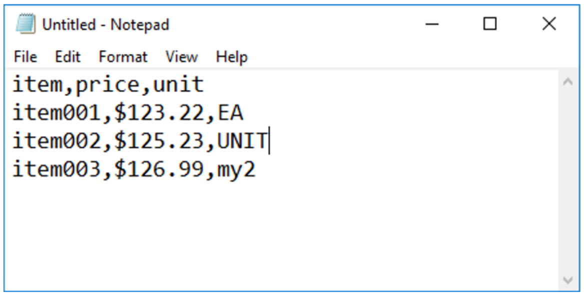
- Below is the Rule file (YAML).



```
---
# Specification for extracting data from https://www.grainger.com/
csv:
  or:
  - columns:
    - header: item
      find:
      - op: find_all
        name: div
        class: myresults
      - op: find
        name: h2
    - header: price
      find:
      - op: find_all
        name: div
        class: myresults
      - op: find
        name: span
      split: 0
    - header: unit
      find:
      - op: find_all
        name: div
        class: myresults
      - op: find
        name: span
      split: 1


skip-empty-row: true
```

- And finally, the output file with extracted data.



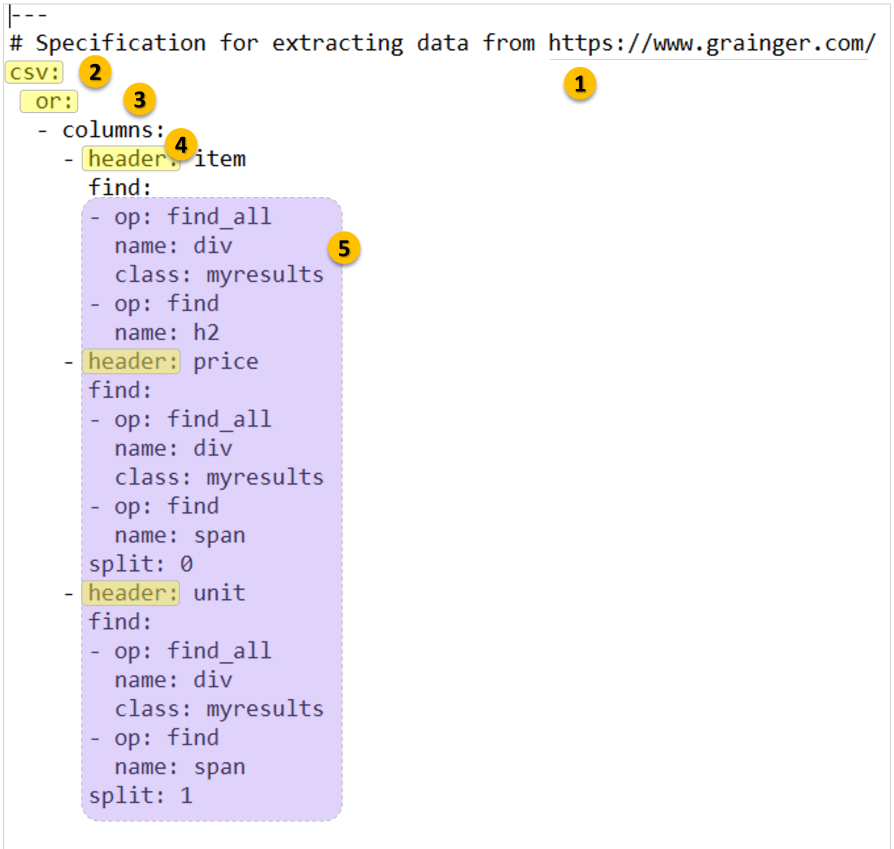
```
item,price,unit
item001,$123.22,EA
item002,$125.23,UNIT
item003,$126.99,my2
```

- The Rule file structure guide

"Web Extract" Rule file structure

Keys and Hierachy						Value type	Occurences	Description	example
csv							Only Once	Web Extract will give .csv as its initial output then you may change to other formats	
	or						0 or 1	"or" can take two or more "columns". It gets the result from the first "columns" and if the result was not found then it goes to the next "columns" and so on.	
		columns					1 or many	if parent is "csv" then there must be only 1 "columns". If parent is "or" then you can have 2 or more "columns"	
			header				Only Once	Name of header in CSV. It occurs only once in "columns"	item
			find				Only Once	This marks the start of your extraction rules. It occurs only once in "columns"	
				op: select_one			0 or many	This is the parent key+value for xpath	op: select_one
					xpath	string	Only Once	xpath must follow "op: select_one" key+value	xpath: /html/body/table/tbody/tr[4]/td/
				op: find_all				Use find_all when the result is a list of repetitive data	op: find_all
				op: find				Use find when the result is a single specific data from a list of repetitive data	op: find
					name	string	Only Once	Name of tag in HTML	name: div
					class	string	0 or many	Find a tag which have the class attribute	class: priceContainer
					key:value	string	0 or many	Additional attribute can be added to help specify target. Find [key="value"] attribute format in a tag	data-automated-test: brand
					key:true	bool	0 or many	Find only the [key] part of the attribute in a tag	myclass: true
			split: n			int	0 or 1	Split with white space and get n-th result (0 is the first)	split: 0
			split				0 or 1	This is the parent key for "separator" and "index"	
				separator		string	Only Once	Split with this separator	separator: "\n"
				index		int	Only Once	Split with this index, n-th result (0 is the first)	index: 1
			re-replace				0 or 1	This is the parent key for "from" and "to"	
				from		string	Only Once	Regular Expression to match	"\\s+"
				to		string	Only Once	Target string to be replaced	" "
no-result						string	0 or 1	If there is no result then print this message. If omitted "No Result" is printed out.	no-result: There is no Result
skip-empty-row						bool	0 or 1	If the result row has a empry row (for example, ",,,") then suppress this row	skip-empty-row: true

4. Below are the explanations of the Rule file construction (syntax).



```
---
# Specification for extracting data from https://www.grainger.com/
csv:
 or:
 - columns:
  - header: item
   find:
   - op: find_all
    name: div
    class: myresults
   - op: find
    name: h2
  - header: price
   find:
   - op: find_all
    name: div
    class: myresults
   - op: find
    name: span
   split: 0
  - header: unit
   find:
   - op: find_all
    name: div
    class: myresults
   - op: find
    name: span
   split: 1
```

1) Give explanations of the Rule file as comments.

2) Regardless of the desired final output format, always start with [csv].

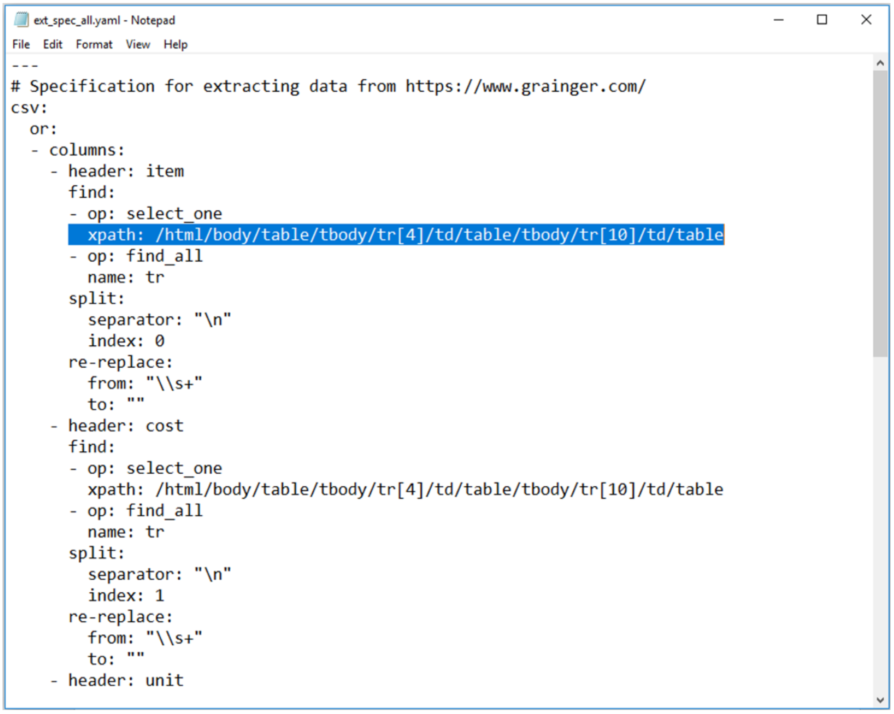
3) [or] is used when you have more than just one type of HTML source returned from the website. It is optional.

4) [header] defines the labels of your output data table.

5) Rest of the YAML is to specify the data to be extracted. Use combinations of tag (name) and attribute (key+value) to identify the data.

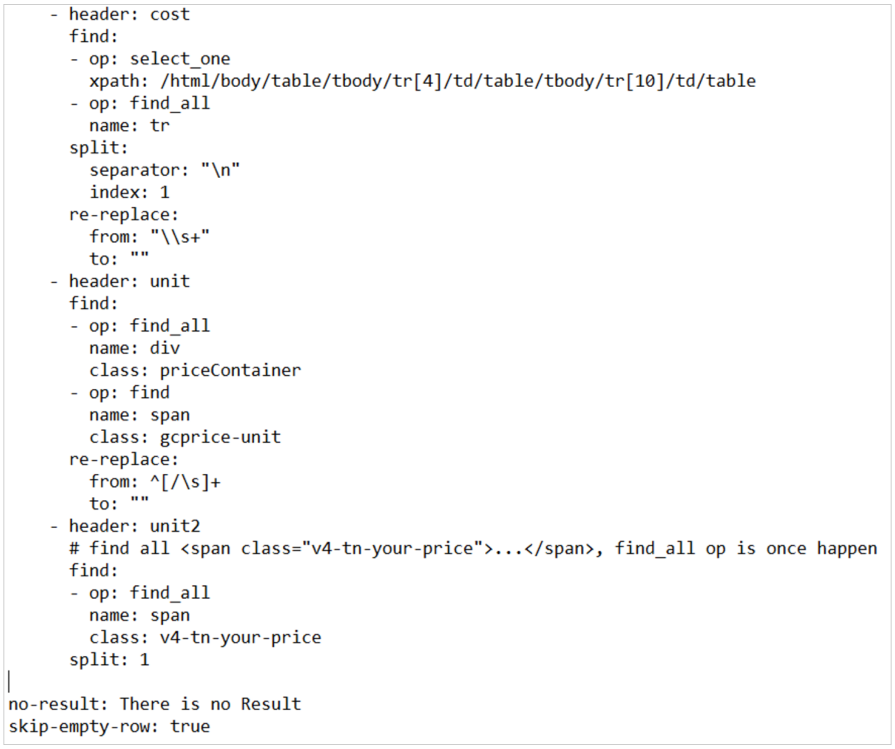
You may use multiple attributes if needed. Please note that the Rule file also includes “split” and “re-replace” for correcting the data.

5. Use of xpath is also possible to specify the target area in the HTML source file like in an example below.



```
---
# Specification for extracting data from https://www.grainger.com/
csv:
  or:
  - columns:
    - header: item
      find:
      - op: select_one
        xpath: /html/body/table/tbody/tr[4]/td/table/tbody/tr[10]/td/table
      - op: find_all
        name: tr
      split:
        separator: "\n"
        index: 0
      re-replace:
        from: "\\s+"
        to: ""
    - header: cost
      find:
      - op: select_one
        xpath: /html/body/table/tbody/tr[4]/td/table/tbody/tr[10]/td/table
      - op: find_all
        name: tr
      split:
        separator: "\n"
        index: 1
      re-replace:
        from: "\\s+"
        to: ""
    - header: unit
```

Additional explanations are provided below.



```
    - header: cost
      find:
      - op: select_one
        xpath: /html/body/table/tbody/tr[4]/td/table/tbody/tr[10]/td/table
      - op: find_all
        name: tr
      split:
        separator: "\n"
        index: 1
      re-replace:
        from: "\\s+"
        to: ""
    - header: unit
      find:
      - op: find_all
        name: div
        class: priceContainer
      - op: find
        name: span
        class: gcprice-unit
      re-replace:
        from: ^[/\s]+
        to: ""
    - header: unit2
      # find all <span class="v4-tn-your-price">...</span>, find_all op is once happen
      find:
      - op: find_all
        name: span
        class: v4-tn-your-price
      split: 1

no-result: There is no Result
skip-empty-row: true
```

`split`

The Split command can take integer, or you can define separate as shown in this example.

`re-replace`

The re-replace command will replace the “from” value (regular expression) to “to” value (string).

`no-result`

Global options can be added at the bottom of the Rule file.

In this example, it shows that when there is no result that data says “There is no Result (default is “No Result”) and skip-empty-row can take true /false parameter.