

Scenario Editor Store Target and Post-processing

You can use the Store tab in the scenario editor to define extractors that can "listen" for a particular Step and Request and extract data of its response.

When creating a Store entry or an Assert entry in the Scenario Editor, you need to configure the Target depending on the type of response.

Also, you might require to apply some post-processing to the response. In this document, you will find examples of the possible values that can be used for the Target and the Post-processing inputs.

PostProcessing

Here is a list of examples of all the possible functions that can be used for post-processing entry. value() returns the extracted value.

```
// substring(value(), 0, 2)
// substring(URLEncode(value()), 2)
// URLEncode(value())
// URLDecode(value())
// append(value(), "-abc123")
// length(value())
// URLParam(value(), 'id')
```

Target

In the same way, the TARGET can have a hint text that should be dynamic according to the selected Target.

The // characters indicate a comment and is not part of the target, with the exception of the HTML target that is an XPath.

DEFAULT Targets

```
“ duration          // returns the request duration
  responseCode      // returns the response status code
  size              // returns de size in bytes of the response
  md5               // returns checksum md4 value of the response
```

HTTP Targets

Given the response is the following string: "**data=45, data=68**" then

```
“ raw/data[0]          // returns 'data=45' and "raw/data[1]" returns 'data=68'
  #data                // returns 45
  Header:Accept        // returns de value of the header Accept
  CookieValue:CSRF-TOKEN // returns de value of the cookie CSRF-TOKEN
```

HTML Targets

Use normal XPathS to extract values from HTML. Note that in this case the double slash at the beginning is needed in the target.

```
“ //input[@name='__RequestVerificationToken']/@value
```

JSON Targets

Use JSON paths to extract values from JSON responses, like

```
“ employees[0].address.city
  [2].carousel.images[1].path
```

PDF Targets

```
“ document              // gets all text in document except forms text
  page[2]              // returns al text in page 2 except forms text
  documentforms        // gets a json with contents of the forms in the d
  regex:Once upon a .* ever after. // returns text that matches the regular expressio
```

STRING Targets

Given the response is the following string: "**The tested processed returned: ID=ABCDEF.**" then

```
“ 11,20"          // returns "processed"
   ID=,false      // returns "ABCDEF."
   ID=,true       // returns "ID=ABCDEF."
   ID=,38,false   // returns "ABCDE"
   ID=38,true     // returns "ID=ABCDE"
```

TEXT Targets

The target should be specified as a regular expression, like:

```
“ (?<=\"__RequestVerificationToken\" type=\"hidden\" value=\")[a-zA-Z0-9\\-_]+
   id=.*
```

CSV Targets

Given that you have a response formatted as a CSV with the following data

id	name	state
1	john	California
2	barry	Nevada
3	stef	Texas
4	karla	New York

Then:

```
“ 2,2              // returns barry
   last(),"name"    // returns karla
   ,"name"          // returns a string array {"name","john","barry","stef","karla"}
   3,last()         // returns Texas
   id=3, 2          // returns stef
```

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

Revision #2

Created 2026-07-31 15:39:47 UTC by Admin

Updated 2026-07-31 15:54:17 UTC by Admin