Runtime time string replacement and encoding

Adhese contains a number of scripts that can be added to any kind of template and will be interpreted at runtime. This allows you to create complex templates where variables can be transformed or encoded via the adheseScript tags.

The scripts are useful for URL encoding, JavaScript escaping, string replacement and other similar tasks.

Tags in the format `[adheseScript...]` have a corresponding `[/adheseScript]`. Each tag applies some modifications to the text in between these tags.

Note that it is possible to nest these tags to apply multiple operations to the same text. The tags are then applied one at a time, with the inner ones being applied first and then the outer ones.

Example of nested tags:

```
[adheseScriptBase64Decode]
  [adheseScriptReplace:-:l]he--o wor-d[/adheseScript]
[/adheseScript]
```

Let's look at some tags in more detail.

Non-escaping script tags

adheseScriptReplace

Searches the enclosed input text for the first parameter and replaces it with the second parameter. Search and replace is done with literal text, not with regex.

```
[adheseScriptReplace:-:l]he--o wor-d[/adheseScript]
```

This results in

```
hello world
```

adheseScriptLower

Puts the enclosed input text in lowercase.

```
[adheseScriptLower]FOOBAR[/adheseScript]
```

results in

```
foobar
```

adheseScriptBase64Decode

Applies Base64 decoding to the enclosed input text.

```
[adheseScriptBase64Decode]What a nice string of characters[/adheseScript]
```

results in

```
V2hhdCBhIG5pY2Ugc3RyaW5nIG9mIGNoYXJhY3RlcnM=
```

adheseScriptUriEncode

Applies UriEncoding to the enclosed input text.

```
[adheseScriptUriEncode]What a nice string of characters[/adheseScript]
```

results in

```
What%20a%20nice%20string%20of%20characters
```

Escaping script tags

By default, some escaping is applied to any `[adhese...]` tags. If multiple tags are nested inside each other, the default escaping will only be applied once, after the final tag is executed.

The default escaping does the following:

```
' ? \'
" ? \"
\ ? \\
]]> ? ]]]]><![CDATA[>
\b \t \n \r \f
unicode characters outside the range 32, 0x7f
```

This should be sufficient for the majority of use cases. However, there may be edge cases where additional control over escaping is required. This can be achieved by utilising one of the tags outlined below.

For example:

- When used within XML without CDATA tags (although we recommend the use of CDATA tags instead of more complex escaping techniques)
- When multiple payloads are nested within each other, a single round of escaping is not enough. The default escaping process would protect against the injection of malicious code into the JavaScript, but an attacker would be able to inject malicious code into the inner HTML. To ensure security, this tag must be escaped twice.

  ```
  var foo = "<img src=\"[adhese...]\" >";
  ```

- When `[adhese...]` tags are placed directly inside HTML without being enclosed in `'` or `"` By default `<` and `>` are not escaped, so an attacker could insert any HTML they want.

  ```
  <h1>[adhese...]</h1>
  ```

If the outermost tags apply to escape (as in one of the tags listed below, except for adheseScriptEscape), then the default escaping is disabled.

If JSON is requested, additional JSON escaping will be applied. This additional escaping is the same as the default escaping described above, with the exception of single quotes and `]]>`, which are not escaped. Additional escaping is applied even when the default escaping is turned off.

adheseScriptEscapeXMLCDATA

```
]]> ? ]]]]><![CDATA[>
```

Useful in XML CDATA tags when the default escaping breaks things by escaping too much.

adheseScriptEscapeXML

Useful for escaping inside XML that does not use CDATA tags.

See https://commons.apache.org/proper/commons-text/javadocs/api-release/org/apache/commons/text/StringEscapeUtils.html#escapeXml11(java.lang.String) for details.

adheseScriptEscapeJSStringWFS

Equivalent to the default escaper but doesn't escape `]]>`

Useful for js. Doesn't escape `/` so don't use it directly in a js regex replace

adheseScriptEscapeJSString

Equivalent to adheseScriptEscapeJSStringWFS but it also escapes `/`

see also: https://commons.apache.org/proper/commons-text/javadocs/api-release/org/apache/commons/text/StringEscapeUtils.html#escapeEcmaScript(java.lang.String)

adheseScriptEscapeHTML

applies https://commons.apache.org/proper/commons-text/javadocs/api-release/org/apache/commons/text/StringEscapeUtils.html#escapeHtml4(java.lang.String)

but also escapes `' ? &#39;`

adheseScriptEscapeNone

Escapes nothing, which is useful for JSON fields that are not embedded into something else (as the separate JSON escaping is enough there).

Revision #5
Created 3 June 2024 10:14:53 by Casper Steuperaert
Updated 25 March 2025 09:03:39 by Casper Steuperaert