JavaScript escape vs. encode

Difference between escape and encodeURI / encodeURIComponent JavaScript functions HTML-5.com is a great guide for web developers. TV Series & Actors and Actresses. Follow TV Series and HTML 5 on Google+.

HTML-5.com ➢ itemscopehttp://data-vocabulary.org/BreadcrumbHTML 5 ➢ itemscopehttp://data-vocabulary.org/Breadcrumb

JavaScript escape vs. encode

JavaScript encode and escape functions

The encodeURI, encodeURIComponent and escape functions convert special characters in URLs and other URIs by <dfn>percent encoding</dfn> the special characters.

JavaScript: encodeURI function

The <dfn>JavaScript `encodeURI` function</dfn> is used to encode an entire unencoded URI, such as `http://authority@www.ExampleOnly.com/my path/my filename.ext?width=100%&my key=my value#fragment-id`. It is most useful when the entire URI is hard-coded in the JavaScript code, so that escaping of special characters within any component is already done manually.

- encodes all characters that should never be included in a valid URI
- also encodes any percent signs, which is used to encode the unsafe characters
- leaves intact the special characters `#& ./:=?@` that act as delimiters within a URI along with `$;` and all other characters not encoded by encodeURIComponent
- uses percent escape encoding of individual UTF-8 octets for non-ASCII characters

JavaScript: encodeURIComponent function

The <dfn>JavaScript `encodeURIComponent` function</dfn> can be used to encode individual components of a URI such as `http`, `authority`, `www.ExampleOnly.com`, `my path`, `my filename.ext`, `width`, `100%`, `my key`, `my value` and `fragment-id` from the example used for encodeURI above. This is the function that should be used when the URI is being constructed from variables containing individual components of the URI.

- encodes the special characters `#$& ,/:;=?@` within a component, in addition to those encoded by the encodeURI function, so they won't be misinterpreted as URI delimiters
- leaves intact the alphanumeric characters and special characters `!'()*-._~`, which are considered "safe" by RFC 1738, but *does* encode the characters `$ ,` anyway
- uses percent escape encoding of individual UTF-8 octets for non-ASCII characters

JavaScript: escape function

- The <dfn>JavaScript escape function</dfn> should be avoided but may be seen in older code that encodes a space as a plus sign (+) or that was designed to be compatible with older browsers
- should **not** be used for text that may contain non-ASCII characters because Unicode characters are converted into a non-standard format as **%u***nnnn* rather than using UTF-8 percent escape codes

JavaScript percent-encoding functions

In all cases, the resulting URI still needs to be converted to valid HTML, by encoding quotes within attributes, ampersands, etc. using HTML character codes.

<script> function onloadFunction(prevOnloadFunction) { return function() { if (prevOnloadFunction) prevOnloadFunction(); var tbody = document.getElementById("javascript-functions"); if (tbody == null) { alert("document.getElementById should NOT return null!"); } else { var chars = " ! \"#$%&'()* ,-./:;<=>?@[\\]^_`{|}~©®—™"; temp = document.createElement("div"); for (i = 0; ! (i >= chars.length); i++) { var row = tbody.insertRow(-1); var cell = row.insertCell(-1); cell.appendChild(document.createTextNode(chars.charAt(i) == " " ? "(space)" : chars.charAt(i))); var cell = row.insertCell(-1); cell.appendChild(document.createTextNode(encodeURI(chars.charAt(i)))); var cell = row.insertCell(-1); temp.textContent = encodeURI(chars.charAt(i)); cell.appendChild(document.createTextNode(temp.innerHTML)); var cell = row.insertCell(-1); cell.appendChild(document.createTextNode(encodeURIComponent(chars.charAt(i)))); var cell = row.insertCell(-1); cell.appendChild(document.createTextNode(escape(chars.charAt(i)))); } delete temp; } }; } window.onload = onloadFunction(window.onload); </script>

Character	encodeURI	HTML	encodeURI<wbr>Component	escape

back to top

THE END

