Overview

- Escaping characters in HTML.
- Escaping characters in URL query strings.

Need for Escaping Characters

- In certain contexts we need to change the meaning of certain characters.
- Need to have some convention about how to escape the traditional meaning.
- Example of escapes are escape sequences introduced using \ character in strings in languages like JavaScript derived from C: `"hello\tworld\n`

  .
- Will cover two escape contexts:
  1. Need to escape <, " and & characters when needed as part of content or attribute value in HTML.
  2. Need to escape special characters when they appear as part of query string in a URL.

HTML Escaping

Minimally, need to escape markup characters within HTML.

- **Less-than character** < used for starting HTML tags.
- **Double-quote character** " used for delimiting attribute values.
- **Ampersand character** & used for introducing escape sequences.
- Also allows representing characters difficult to type on keyboard.
- Mustache escapes all . . . content within {{...}}.

Numeric Character Entity References in HTML

Decimal References `&#`*dddd*`;` represents the character having Unicode code point *dddd* decimal.

Hexadecimal References `&#x`*hhhh*`;` represents the character having Unicode code point *hhhh* hexadecimal.

Examples: `&#931;` `&#0931;`, `&#x3A3;`, `&#x3a3;`, `&#x03A3;` are all numeric character references for the character **Greek Capital Letter Sigma** Σ.

Named Character Entity References in HTML

Named character entity references `&`*name*`;` represents character with name *name*.

- XML allows only 5 named entities: `&quot;`, `&amp;`, `&apos`, `&lt;`, `&gt;` represent the characters *quotation-mark* `"`, *ampersand* `&`, *apostrophe* `'`, *less-than-sign* < and *greater-than-sign* > respectively.
- HTML allows a *huge additional set* of named entities.

URI Encoding

- Encode characters which may have reserved meanings within a URL.
- RFC 3986 reserves special characters like `/`, `?` and `&`.
- Special characters need to be escaped using %*hh* where *hh* is the ASCII code for the character.
  - Slash `/` represented as `%2F`.
  - Question-mark `?` represented as `%3F`.
  - Ampersand `&` represented as `%26`.
- Alphanumerics, hyphen `-`, underscore `_`, period `.` and tilde `~` never need to be escaped.
- Characters need not be URI-escaped if used within a context where they are not special; for example, `/` does not need to be escaped within a query string.

JavaScript Encode URI Functions

`encodeURI(`*string*`)` Will encode only those special characters which do not have special use within a URI. So it will not escape characters like `/`, `?`, `#`. Use to encode entire URI which does not contain special characters within contexts where they have special meaning. Decode using `decodeURI()`.

`encodeURIComponent(`*string*`)` Will encode all characters except `-`, `_`, `.`, `!`, `~`, `*`, `'`, `(` and `)`. Hence safe to use only on URI component. Decode using `decodeURIComponent()`. Can also be used to encode cookie values.

JavaScript Encode URI Functions Examples

```
> uri = ’http://www.example.com?q=encode url’
’http://www.example.com?q=encode url’
> encodeURI(uri)
’http://www.example.com?q=encode%20url’
> encodeURIComponent(uri)
’http%3A%2F%2Fwww.example.com%3Fq%3Dencode%20url’
> decodeURI(encodeURI(uri))
’http://www.example.com?q=encode url’
>
```