Cookies

Ingo Köster

Diplom Informatiker (FH)

Was sind Cookies?

- Eine Website kann in Cookies Informationen beim Client ablegen und diese später wieder auslesen
- Der Browser speichert die Informationen in einer externen Datei
- Wenn ein Browser eine Webseite von einem Server anfordert, werden Cookies, die zu der Seite gehören, mit an die Anfrage gehängt
- Auf diese Weise erhält der Server die notwendigen Daten um sich an einzelne Nutzer zu "erinnern"

Was sind Cookies?

› Das Cookie wird im HTTP-Header übertragen

Cookie „API“

- Keine Methoden zum Verwenden von Cookies
- Cookies werden über das Lesen und Schreiben der `cookie`-Eigenschaft des `document`-Objektes gesetzt & gelesen
- Cookies bestehen aus formatierten Strings
- Über diese Strings wird auch Geltungsbereich und Lebensdauer gesetzt

Cookies speichern

- Cookies werden als Name-Werte-Paare gespeichert
  - `document.cookie = 'name=wert';`
- Die Name-Werte-Paare sollten keine Semikola, Komma oder Leerzeichen (whitespaces) enthalten
  - Nach RFC 2109 können in Cookies auch Semikola, Kommata und Leerzeichen gespeichert werden
    - Diese müssen in Anführungszeichen gesetzt werden
- Semikola werden zum Trennen der Attribute wie `path`, `domain`, `max-age`, etc. verwendet

Cookies speichern

› Beispiel für Cookie im Header

```
HTTP/1.0 200 OK
Set-Cookie: letzteSuche=Y29va2llIGF1ZmJhdQ==;
            expires=Tue, 29-Mar-2014 19:30:42 GMT;
            Max-Age=2592000;
            Path=/cgi/suche.py
```

› Hinweis: Zum Speichern eines Cookie muss ein Dokument geladen sein (in einem leerem Browserfenster können keine Cookies gespeichert werden)

encodeURIComponent & decodeURIComponent

- Codieren bzw. Decodieren Strings als gültige Komponente eines URIs (Uniform Resource Identifier)
- Beispiel:

```
var uriEncode = encodeURIComponent
("www.Not a URL.com");
var uriDecode = decodeURIComponent(uriEncode);
```

- Inhalt von `uriEncode`

```
www.Not%20a%20URL.com
```

- Inhalt von `uriDecode`

```
www.Not a URL.com
```

Cookies lesen

- Alle Cookies eines Dokumentes werden ebenfalls über die `document.cookie`-Eigenschaft ausgelesen

- Enthält keines der Attribute die ggf. gesetzt wurden (z.B. `max-age`)

- Um zu prüfen, ob ein Cookie gespeichert wurde, eine if-Bedingung verwenden
  - `if (document.cookie.length > 0) { … }`
  - `if (document.cookie != "") { … }`

Cookies lesen

› Cookie-String beispielsweise mittels `split()` am Gleichzeichen zerlegen

› Falls `encodeURIComponent` verwendet wurde mittels `decodeURIComponent` zurückwandeln

› Ggf. können die Cookie-Daten auch mittels `JSON.parse()` verarbeitet werden

Cookies löschen

› Das Ablaufdatum des betreffenden Cookies in die Vergangenheit setzen

› `document.cookie = 'name=wert;expires=Thu, 01-Jan-70 00:00:01 GMT;';`

› Der Cookie wird anschließend vom Browser gelöscht

Lebensdauer

- Die Lebensdauer von Cookies ist eher kurz
- Im Standardfall ist ein Cookie nur für eine Session gültig
- Im Unterschied zu `sessionStorage` sind diese jedoch nicht von der Lebensdauer eines Fensters, sondern von der Lebensdauer des Browser-Prozesses abhängig
- Die Spanne der Lebensdauer kann mit Hilfe des Attributes `max-age` (in Sekunden) gesetzt werden
- Alternativ kann das Ablaufdatum in UTC oder GMT mit dem Attribut `expires` gesetzt werden

Beispielfunktion zum Setzen eines Cookie

```
function setCookie(cname, cvalue, exdays)
{
    var d = new Date();
    d.setTime(d.getTime() + (exdays*24*60*60*1000));
    var expires = "expires=" + d.toGMTString();
    document.cookie = cname + "=" + cvalue + "; " + expires;
}
```

Cookie Attribut `secure`

- Ist `secure` gesetzt, so darf der Cookie nur dann gelesen werden, wenn eine HTTPS-Verbindung verwendet wird

- Diese Eigenschaft wird ohne Wert angegeben
  - `document.cookie = "name=wert;secure";`

Geltungsbereich

- Ähnlich wie bei `sessionStorage` und `localStorage` ist die Sichtbarkeit auf die Same Origin Policy (SOP) beschränkt
  - `http://www.beispiel.de`
  - `https://www.beispiel.de`
  - `http://shop.beispiel.de`
  - `http://www.beispiel.de:8080`

- Dazu kommt bei Cookies der Pfad des Dokumentes
  - http://www.beispiel.de/shop/index.html — Erstellt Cookie
  - http://www.beispiel.de/shop/warenkorb.html — Sichtbar
  - http://www.beispiel.de/shop/data/data.html — Sichtbar
  - http://www.beispiel.de/hilfe.html — Nicht Sichtbar!

Geltungsbereich

- Wird mit den Attributen `path` und `domain` konfiguriert
- Mittels `path` kann das Cookie so gesetzt werden, dass es für alle Seiten eines URL gelesen werden kann
- Path `"/"` für **www.beispiel.de**, bedeutet das alle Seiten den Cookie lesen können
- Mittels `domain` können Subdomains angegeben werden
- Mit **".beispiel.de"** können z.B. **shop.beispiel.de** und **forum.beispiel.de** den Cookie lesen
- Wird `domain` nicht gesetzt, ist der Standardwert der Hostname des Servers

Cookies im Browser aktiviert?

› Überprüfen ob Cookies im Browser aktiviert sind

› Mittels der booleschen Eigenschaft `navigator.cookieEnabled`