Ajax libraries - communication

Andrea Ferracani
andreaferracani@gmail.com

Ajax non era una novità quando Jesse James Garret coniò il nome (asynchronous javascript and xml).

E' un merging di diverse tecnologie e ne eredita tutti i problemi e le limitazioni

E' diventato diffuso grazie ad alcune brilanti applicazioni come Google suggest o Google maps

Ajax è il merging di queste tecnologie:

- Semantic xhtml markup
- Il document object model
- Javascript
- XML

XHTML e DOM

Con ajax funziona qualsiasi pagina html, ma sicuramente un markup semantico con tag e id appropriati rende la manipolazione degli elementi molto più facile

XMLHttpRequest

E' il core di ajax, l'oggetto che serve ad interagire con il server tramite chiamate asincrone. C'è un draft del W3C con le specifiche ma non è implementato da tutti i browser (http://www.w3.org/TR/XMLHttpRequest).
Esiste già da IE 5.0, in cui è un active-x plugin, cioè un componente che poteva essere istanziato così:

```
var request = new ActiveXObject("Microsoft.XMLHTTP");
```

Adesso è stato implementato da tutti (anche explorer 7) in questa forma:

```
var req = new XMLHttpRequest();
```

I metodi tipici di questo oggetto sono:

- open(method, URL[, asynchronous[, userName[, password]]])
- setRequestHeader(label, value), va chiamato dopo open e prima di send
- send(content) trasmette la richiesta, e content opzionale se per es. il metodo è post

- abort(), per fermare la request corrente
- getAllResponseHeaders(), ritorna tutti gli headers come stringa
- getResponseHeader(label), ritorna l'header con una label specifica

FARE UNA REQUEST

```
function stateChangeListener() {
//some code
}
var request = false;
if(window.XMLHttpRequest) {
var request = new window.XMLHttpRequest();
} else if (window.ActiveXObject) {
var request = new window.ActiveXObject('Microsoft.
XMLHTTP');
}
```

```
if(request) {
request.onreadystatechange = stateChangeListener;
request.open('GET', '/your/script/?
var=value&var2=value', true);
request.send(null);
}
```

Nel caso si usi il metodo post la request sarebbe:

```
request.open('POST', '/your/script/', true);
request.send('var=value&var2=value');
```

!! Javascript non può accedere a file system o da file input, limitazione dei browser

Il metodo onreadystatechange, per l'esattezza un gestore d'evento, sarà chiamato diverse volte durante la request registrando lo stato di una certa proprietà dell'oggetto.

Le proprietà di XMLHttpRequest sono:

- readyState è un intero che regista gli stati della request:
  - 0 uninitialized
  - 1 loading
  - 2 loaded
  - 3 interactive
  - 4 complete

- responseText se il server risponde una stringa
- responseXML un document object che rappresenta l'XML di risposta se c'è l'header giusto nel documento
- status, lo stato della richiesta appunto 404 not found oppure 200 ok per es. Vedi (http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html)
- statusText
- onreadystatechange che deve contenere le funzioni da eseguire relative allo stato della richiesta

es.

```
function stateChangeListener() {
// Switch functionality depending on the state of
the request
switch (request.readyState) {
case 1:
// Loading
break;
case 2:
// Loaded
break;
```

```
case 3:
// Interactive
break;
case 4:
// Complete
if (request.status == 200) {
// Do something with request.responseText
// or request.responseXML
} else {
// There was a possible error code in request.status
// with a message as reported in request.statusText
}break;}}
```