Introduzione a AJAX - Asynchronous Javascript And Xml

Sommario

- Motivazioni
- Cosa è AJAX?
- Vantaggi
- Esempi
- Funzionamento

Motivazioni (1)

- XHTML e HTTP sono strumenti “deboli”
  - Scarsa interattività
  - Aggiornamenti per “blocchi a grana grossa” (l’intera pagina web)
- Per usufruire dei servizi di applicazioni web, gli utenti preferiscono usare il browser anziché specifiche applicazioni

Motivazioni (2)

- Altre tecnologie browser-based hanno fallito
  - Java Applets non sono supportate universalmente (ad esempio non interagisce con Html)
  - Flash / Flex non sono supportate universalmente e hanno scarsa capacità
  - ...

Cos'è AJAX?

- Asynchronous Javascript And Xml (AJAX):
  - Approccio per sviluppare applicazioni Web
    - Aumenta la dinamicità di pagine web, grazie allo scambio di piccole quantità di dati
    - Permette alle pagine web di cambiare il proprio contenuto senza effettuare refresh dell'intera pagina
  - Tecnologia web indipendente dal software del web server

Approccio

- Le richieste del client sono elaborate in modo asincrono
- Viene aggiornata solo una piccola parte del documento correntemente presentato

Asynchronous Javascript And Xml (1)

- Asynchronous:
  - le richieste possono essere fatte asincronamente e sincronamente
  - in entrambi i casi le pagine web sono aggiornate senza refresh
- Javascript:
  - È usato tipicamente sul client
  - Qualsiasi linguaggio sul server

Asynchronous Javascript And Xml (2)

- Xml:
  - i messaggi di request/response possono contenere XML
  - in generale possono contenere qualsiasi testo

Vantaggi

- Migliora
  - la fruibilità della pagina
  - l'uso della banda utilizzata: sono ottenuti dal server solo I dati strettamente necessari

Esempi di Successo

- Google Maps (http://maps.google.com/)
- My Yahoo! (http://my.yahoo.com/)

Tecnologie di Base

- AJAX usa
  - XHTML e CSS per la presentazione
  - DOM aggiornato dinamicamente
  - XML con XSLT, oppure HTML preformattato, testo semplice, … per lo scambio di dati
  - un'istanza della classe XMLHttpRequest, che consente al browser di dialogare in modo asincrono con il server
  - JavaScript, che gestisce il tutto

Funzionamento (1)

- AJAX è eseguito all'interno del browser
- È basato sul protocollo HTTP
  - Trasferisce dati asincronamente tra il browser e il web server (HTTP requests)
- Le Http requests sono inviate da chiamate a script di JavaScript senza dover effettuare submit di form
- XML è comunemente usato come formato per ricevere dati dal server
  - Si può usare anche plain text

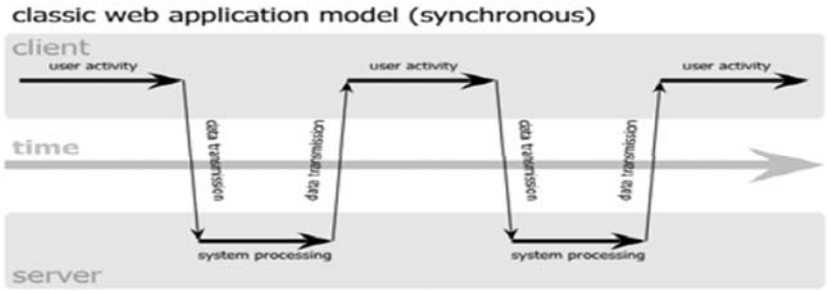
Funzionamento (2)

- Idea di base:
  - JavaScript
    - Definire un oggetto per poter inviare una opportuna HTTP request
    - Ottenere l'oggetto richiesto
    - Definire una opportuna funzione per gestire la response
    - Effettuare una request GET o POST
    - Inviare i dati
    - Gestire la response

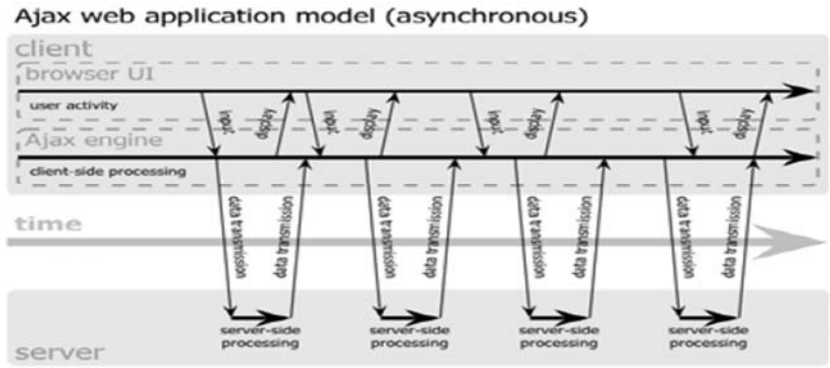
Funzionamento (3)

- XHTML
  - Caricare JavaScript
  - Definire il control che svolge la request
  - Identificare gli elementi di input che saranno letti dallo script

Applicazione Web Classica



Applicazione Web AJAX



XMLHttpRequest object (1)

- Un elemento di una pagina chiama una funzione JavaScript
- La funzione deve creare un oggetto XMLHttpRequest per contattare il server
- Si deve determinare il client
  - IE, Firefox, Safari, …
- Se IE allora

  http = new ActiveXObject("Microsoft.XMLHTTP")
- Se Mozilla allora

  http = new XMLHttpRequest()

XMLHttpRequest object (2)

```
var request;
function getRequestObject() {
    if (window.ActiveXObject) {
        return(new ActiveXObject("Microsoft.XMLHTTP"));
    } else if (window.XMLHttpRequest) {
        return(new XMLHttpRequest());
    } else {
        return(null);
    }
}
```


Invio della Richiesta (1)

- Una volta creato, l'oggetto XMLHttpRequest deve essere impostato per poter chiamare il server

```
http.open("GET", serverurl, true);
http.onreadystatechange =
   jsMethodToHandleResponse;
http.send(null);
```

- L'oggetto XMLHttpRequest è usato per contattare il server e ricevere I dati da questo forniti

Invio della Richiesta (2)

- Ottenuta la response, il metodo JavaScript jsMethodToHandleResponse può aggiornare la pagina

Initiate Request

```
function sendRequest() {
   request = getRequestObject();
   request.onreadystatechange = handleResponse;
   request.open("GET", "message-data.html",   true);
   request.send(null);
}
```

Response handler function name

URL of server-side resource

Don't wait for response
(Send request asynchronously)

POST data
(always null for GET)

Gestione della Response

- È necessario implementare un metodo JavaScript per gestire la response (Event Handler)

```
function jsMethodToHandleResponse(str)
{
//simply take the response returned an update an html
   element with the returned value from the server
    document.getElementById("result").innerHTML = str;
}
```

- La pagina ha comunicato con il server senza dover effettuare un refresh completo

Handle Response

```
function handleResponse() {
   if (request.readyState == 4) {
       alert(request.responseText);
   }
}
```

Response is returned from server
(handler gets invoked multiple times)

Pop up dialog box

Text of server response

Proprietà readyState

- La proprietà readyState di XMLHttpRequest definisce lo stato corrente dell'oggetto XMLHttpRequest
- Possibili valori di readyState

State	Description
0	The request is not initialized
1	The request has been setup
2	The request has been submitted
3	The request is in process
4	The request is completed

Esempio Completo

```
var request;
function getRequestObject() {
    if (window.ActiveXObject) {
        return(new ActiveXObject("Microsoft.XMLHTTP"));
    } else if (window.XMLHttpRequest) {
        return(new XMLHttpRequest());
    } else
        return(null);
}
function sendRequest() {
    request = getRequestObject();
    request.onreadystatechange = handleResponse;
    request.open("GET", "message-data.html", true);
    request.send(null);
}
function handleResponse() {
    if (request.readyState == 4) {
        alert(request.responseText);
    }
}
```


Lato XHTML (1)

- Usare XHTML, non HTML 4
  - Per il DOM

  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN“

  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

  <html xmlns="http://www.w3.org/1999/xhtml">...</html>
- A causa di difetti di IE NON usare l’header XML prima del DOCTYPE

Lato XHTML (2)

- Caricare il file JavaScript

  <script src="*relative-url-of-JavaScript-file"*

  type="text/javascript"></script>
- Usare il tag di fine </script> separato
- Definire il control per iniziare la request

  <input type="button" value="*button label"*

  onclick="*mainFunction()"*/>

Problemi con IE

- IE non manipola correttamente l’header XML
  - Ogni doc XML deve iniziare con l’header:

    <?xml version="1.0" encoding="UTF-8"?>
  - Ma IE potrebbe non comportarsi correttamente

References

- http://en.wikipedia.org/wiki/Ajax_%28programming%29
- http://www.w3schools.com/ajax/default.asp