Web Application Engineering
Ajax & JSF

cristian lucchesi
IIT-CNR

Pescara, 15-16 Maggio 2007
Alei – Ud'A

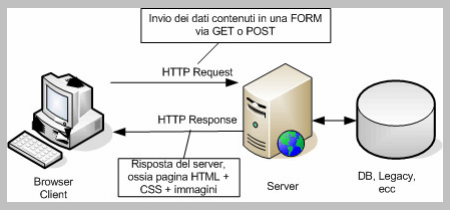
Desktop Application vs Web Application

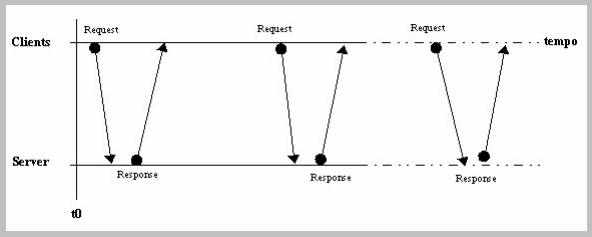
- oggi quando si scrive una applicazione software si hanno due scelte:
  - **applicazione desktop**
    - sono di solito rapide (girano sul proprio computer, non devono aspettare una connessione internet)
    - hanno una grande interfaccia grafica
    - sono incredibilmente dinamiche, puoi cliccare, inserire, trascinare aprire menu, sottomenu, etc, praticamente senza aspettare
  - **applicazione web**
    - sono (di solito) aggiornatissime
    - forniscono servizi che non sono pensabili sul proprio pc (si pensi a amazon.com, eBay, google
    - sono di solito lente, spesso si deve attendere per una risposta dal server
    - dopo un click si deve aspettare che la pagina si ricarichi aspettando la risposta dal server prima che la nuova pagina sia generata

Ajax

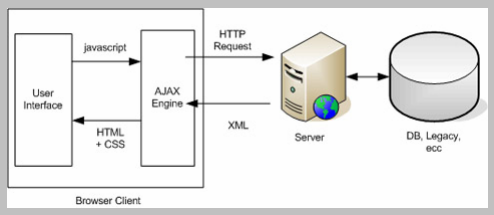
- molti hanno già sentito almeno parlare di AJAX (Asynchronous JavaScript And XML)
- con AJAX le applicazioni web acquistano i vantaggi delle applicazioni desktop
- gli scopi principali di AJAX sono:
  - inviare richieste ad un server in funzione degli eventi generati dall'utente
  - prelevare informazioni in modo asincrono da un server
  - aggiornare solo una parte della pagina, in modo da:
    - ricaricare solo la parte della pagina relativa alle informazioni prelevate
    - evitare di ricarica tutta la pagina
    - caricare inizialmente solo pochi dati e prelevare il resto in funzione delle eventi dell'utente

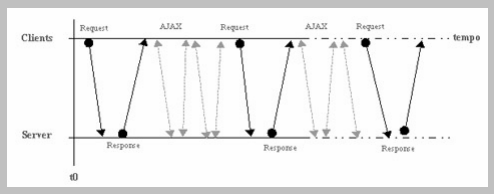
Modello classico





Modello Ajax

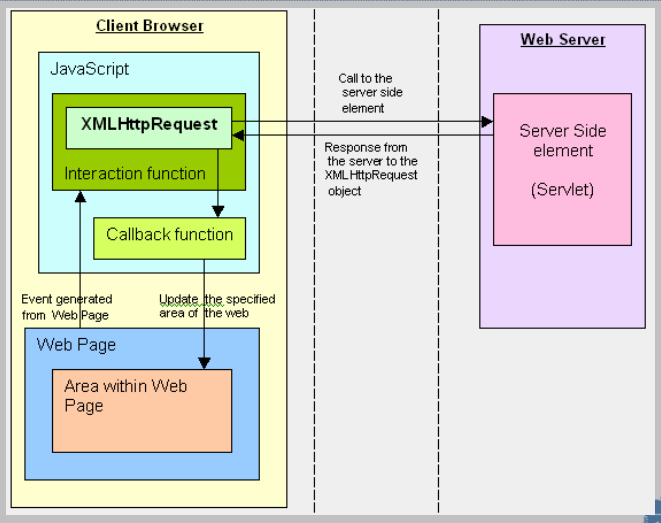




Vecchie tecnologie nuovi trucchi

- ci sono molte tecnologie che coinvolte nelle applicazioni Ajax
- **HTML** è utilizzato per construire le pagine web e identificare i campi per il successivo uso nel resto dell'applicazione
- **JavaScript** è il cuore del codice tramite cui funzionano le applicazioni AJAX ed è di supporto per la comunicazione con le applicazioni server
- **DHTML** o Dynamic HTML, aiuta nel aggiornare le pagine dinamicamente
- **DOM** è utilizzato (tramite Javascript) per manipolare sia la struttura della pagina HTML (in alcuni casi) sia per manipolare le risposte XML restituite dal server

Interazioni AJAX



XMLHttpRequest object

```
var httpRequest;

// per IE
if (window.ActiveXObject) {
  httpRequest = new ActiveXObject("Microsoft.XMLHTTP");
}
else if (window.XMLHttpRequest) {
  // per gli altri browser
  httpRequest = new XMLHttpRequest();
}



function callServer() {
  httpRequest.open("GET", url, true);
  ...
}
```

- il primo passo è creare e configurare un oggetto XMLHttpRequest object
- ci sono due implementazioni di questo oggetto, **ActiveXObject** per IE, e **XMLHttpRequest** per gli altri browser

- l'oggetto deve essere configurato specificando:
- il metodo HTTP, GET o POST
- la URI del metodo sul server con cui comunicaree
- come terzo parametro, specificare se si tratta di una interazione sincrona o asincrona (true vuol dire asincrona)

XMLHttpRequest object

cont.

```
function callServer() {
  ...
 httpRequest.onreadystatechange =
    processRequest;
  ...
}
```

- deve essere impostato il nome della funzione javascript che gestirà la chiamata di ritorno (callback) dal server, nell'esempio la funzione si chiama processRequest

```
function callServer() {
  ...
  httpRequest.send(null);
}
```

- infine si invia la richiesta, il parametro null indica che non si invia nessun parametro, tipicamente si può inviare un file xml

Gestire la risposta del server

```
function processRequest() {
  if (httpRequest.readyState == 4) {

    if(httpRequest.status == 200) {
      //process the response
    }
    else {
      alert("Error loading page\n" +
            httpRequest.status + ":" +
            httpRequest.statusText);
    }
  }
}
```

- la risposta del server arriva, in modo asincrono, in XML
- l'XML viene processato in Javascript dalla funzione dei callback specificata nell'oggetto httpRequest
- la funzione per prima cosa verifica lo stato dell'oggetto httpRequest, **4** significa che la risposta è completa
- successivamente controlla che la richiesta sia andata a buon fine controllando l'HTTP status code

Processare la risposta del server

```
//The method getElementsByTagName,
//gets the element defined by the given tag var

profileXML =
 httpRequest.responseXML.getElementsByTagName("Profile")[0];

//The node value will give you actual data
var profileText = profileXML.childNodes[0].nodeValue;
```

- la risposta è all'interno dell'attributo **responseText** dell'oggetto XMLHttpRequest
- la risposta è sempre in XML e l'attributo **responseXML** dell'oggetto XMLHttpRequest contiene la rappresentazione dell'oggetto in XML
- l'XML può essere analizzato dal Javascript utilizzando il DOM per ottenere i dati

Aggiornare l'HTML della pagina

```
//Create the Text Node with the data received
var profileBody = document.createTextNode(profileText);

//Get the reference of the DIV in the HTML DOM
//by passing the ID
var profileSection=
    document.getElementById("profileSection");

//Check if the TextNode already exist
if(profileSection.childNodes[0]) {
  //If yes then replace the existing node with the new one
  profileSection.replaceChild(profileBody,
                              profileSection.childNodes[0]);
}
else {
  //If not then append the new Text node
  profileSection.appendChild(profileBody);
}
```

- una volta ottenuti i dati, l'ultimo passo è aggiornate l'HTML tramite il DOM della pagina HTML
- in javascript si possono modificare (e ricaricare) gli elementi della pagina anche dopo che la pagina è stata caricata
- il metodo document.getElementById (id) è utilizzato per referenziare l'elemento della pagina con l'id specificato

Legare Ajax ad un evento

- per legare l'interazione Ajax alla pagina è necessario:
- identificare quella parte della pagina che deve essere aggiornata dinamicamente (tipicamente tramite l'utilizzo dell'attributo **id**)

   ...

- identificare quale evento aggiornerà l'area prescelta
  - cliccare su un link (onclick="javascript:callServer()")
  - passare il mouse su un'immagine (onmouseover="javascript:callServer())
  - ogni volta che si inserisce un carattere in un input (onkeyup)
  - ...

Ajax Framework

- Un framework aiuta il lavoro del programmatore sia per la scrittura del codice lato client (per spedire le richieste e ricevere le risposte) che lato server per processare le richieste e fornire le risposte al browser

- Ajax component frameworks
  - offrono componenti pre-built, come tabbed panel che automaticamente creano e gestiscono l'html, autocompletion, drag and drop, ...
  - offrono API per la customizzazione
  - offrono skinning facilities
  - supportano il controllo programmatico dei componenti
- Server-driver Ajax Framework
  - i componenti sono di solito creati e manipolati dal server utilizzando un linguaggio di programmazione o di template
  - le pagine sono generate da una combinazione di elementi HTML generati sia lato server che lato client
  - le azioni dell'utente sono comunicate al server tramite le tecniche Ajax, il codice lato server manipola le richieste e i cambiamenti sul modello sono riflessi sul client automaticamente

Ajax4Jsf

- Ajax4jsf è un server-driver Ajax framework
- Ajax4jsf estende le JSF aggiungendo la gestione di AJAX nelle applicazioni JSF senza preoccuparsi di scrivere nemmeno una riga di Javascript

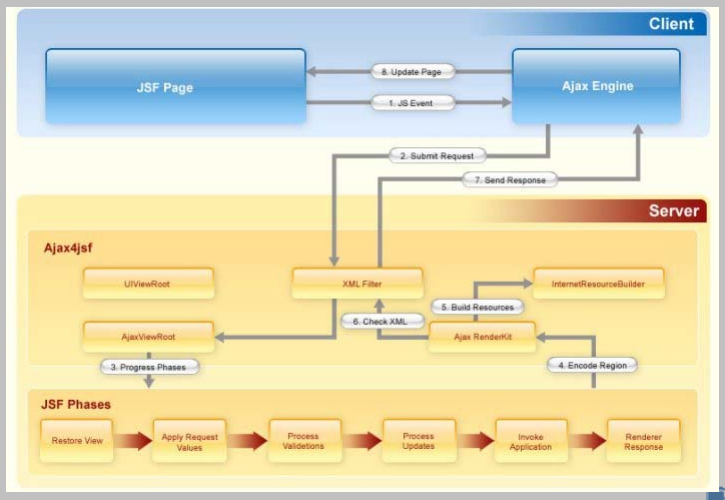
```
<f:view>
   <h:form>
      <h:panelGrid columns="2">
         <h:outputText value="Type the Text:" />
         <h:inputText value="#{bean.text}">
            <a4j:support event="onkeyup" reRender="repeater" />
         </h:inputText>
         <h:outputText value="Text in the AJAX Response:" />
         <h:outputText id="repeater" value="#{bean.text}" />
      </h:panelGrid>
   </h:form>
</f:view>
```

Fires AJAX request on this event

a4j:support adds AJAX capability to existing components

Points to component(s) to be re-rendered

Ajax4jsf e JSF life-cycle



Riferimenti

- [Jesse James Garrett] Ajax: A New Approach to Web Applications **http://www.adaptivepath.com/publications/essays/archives/000385.php**
- Implementing simple AJAX interaction in your Web Application using XMLHttpRequest object: **http://www.javareference.com/jrexamples/viewexample.jsp?id=111**
- Mastering Ajax, Part 1: Introduction to Ajax: **http://www-128.ibm.com/developerworks/web/library/wa-ajaxintro1.html**
- ajax-tutorials.com: **http://www.ajax-tutorials.com/tutorial-list/**
- JBoss Ajax4jsf **http://labs.jboss.com/jbossajax4jsf/**

grazie per l'attenzione

cristian.lucchesi@iit.cnr.it