AJAX

by

GaRaGeD

Maximiliano Valdéz

Historia

Listos para dormirse ?

Asynchronous JavaScript And XML

O lo que es lo mismo Transporte y Recepción asincrónica de datos por medio de JS usando XML (u otro formato a tu gusto)

Todo empezo en la era de la “guerra de los navegadores”, donde netscape (NS) y exploder (IE) competian por hacer una implementacion en JS para la importación y exportación de datos sin necesidad de recargar la página.

Para variar NS fue el primero en sacar una implementación, pero IE gano la guerra y ahora todos conocemos la parte de ellos y no la de NS.

Con la guerra ganada M$ dio el paso definitivo de la historia de AJAX usando la comoda ayuda de la comunidad. Originalmente el metodo era usado en outlook, y posteriormente en IE

IE implementó pues el objeto ActiveX e hizo posible la comunicación asincrona dando pauta a una nueva era en el diseño web (no tanto en realidad: iframes !).

XMLHTTP

A partir de ahi gecko hizo la implementación propia del mismo objeto y con la ayuda de la w3c ambos lados del oceano hay llegado a implementar un manejo del Document Object Model (DOM) que es aceptablemente compatible.

Pero que es AJAX ?

Es esencialmente un método de desarrollo web (2.0 ?) que nos permite darle al usuario final algunas comodidades con respecto a lo que se puede hacer con los metodos tradicionales.

Se basa en el uso del objeto XMLHTTP de JavaScript en conjunto con la manipulación de los documentos HTML por medio del DOM.

Parte de la propuesta es que se use lo mas posible XML como medio de transporte, pero hay muchas situaciones en las que no es lo mas conveniente y en su caso se puede utilizar HTML o notaciones de JS como JSON, RCP entre otras (si se te ocurre algo mejor, pruebalo !).

Ventajas de AJAX

- Actualizar secciones de una pagina, sin refrescarla, con datos provenientes del servidor.
- Agilizar ciertas actividades por medio de la minimizacion de los datos transferidos.
- Introducir técnicas avanzadas de JS que favorecen/facilitan el diseño de páginas vistosas.
- Permite hacer diseños complejos de manejo masivo de datos en una sola pagina (gmail, live.com).
- etc.

Desventajas

- JS obligatorio
- Nuevos conocimientos (JS ?, DOM, XML, ??)
- Boton de regreso
- Cuestiones de usabilidad miscelaneas

Introducción a AJAX

- Requerimientos:
  - JavaScript
  - DOM a nivel decente (No es necesario ser experto)
  - Sentido común
  - Usar una biblioteca adecuada (No reinventar la rueda)
- Usemos el XMLHTTPObject o como se le conozca en tu navegador de elección (y en todos los posibles)
- CSS y DHTML (Mas DOM)

este producto puede ser nocivo para tu salud mental

XMLHTTP: Compatibilidad

- Necesitamos compatibilidad con los diferentes navegadores
  - Hay que obtener el objeto de manera inteligente para unificar el trabajo en los diferentes navegadores
  - En pocas palabras busca el objeto en sus diferentes variantes y devuelvelo con un nombre comun (u-n-i-f-i-c-a-c-i-ó-n)
  - Seria inteligente empezar a usar una biblioteca de AJAX para no duplicar esfuerzos y en una de esas podemos empezar a ayudar!

Volvamos al objeto XMLHTTP

- Si recuerdan mencioné que diferentes browsers tienen implementaciones diferentes del objeto (M$ tiene 3 !)
- Dojo nos hace la vida facil proporcionandonos el objeto acueducto y aún mas lo esconde suficiente para que no tengamos que hacer demasiado con el (de hecho no lo tocamos, solo a la interfase que nos da dojo)
- Veamos que hariamos normalmente...

Old school

```
function createRequestObject() {
    var ro;
    var browser = navigator.appName;
    if(browser == "Microsoft Internet Explorer"){
        ro = new ActiveXObject("Microsoft.XMLHTTP");
    }else{
        ro = new XMLHttpRequest();
    }
    return ro;
}

var http = createRequestObject();
```

```
function sndReq(value) {
    http.open('get', 'test.php?var='+value);
    http.onreadystatechange = handleResponse;
    http.send(null);
}

function handleResponse() {
    if(http.readyState == 4){
        var response = http.responseText;
        var update = new Array();
        document.getElementById('somediv').innerHTML =
                response.innerText;

    }
}
```

Que con eso ?

Por si se me olvido decirlo en la página anterior !

- Muchas funciones
- De hecho **createRequestObject()** no es tan “cross-browser” como quisieramos
- El manejador (**handleResponse()**) no maneja diferentes receptores para el HTML entrante
- Frecuentemente tendriamos que definir multiples funciones para quien hace la peticion y su manejador correspondiente

Usemos DOJO toolkit - http://dojotoolkit.org

- Dojo es una herramienta completa de JS para ajax, efectos, eventos, validacion y “widgets”
- Es extremandamente util para agilizar el trabajo de entrada/salida (IO) de datos entre otras muchas cosas
- Nos enfocaremos en la parte de IO lo que resta de la plática y si hay tiempo veremos otros detalles de dojo
- Una parte indispensable es la de los eventos, no podemos dejarla de lado y la iremos revisando junto con lo de IO

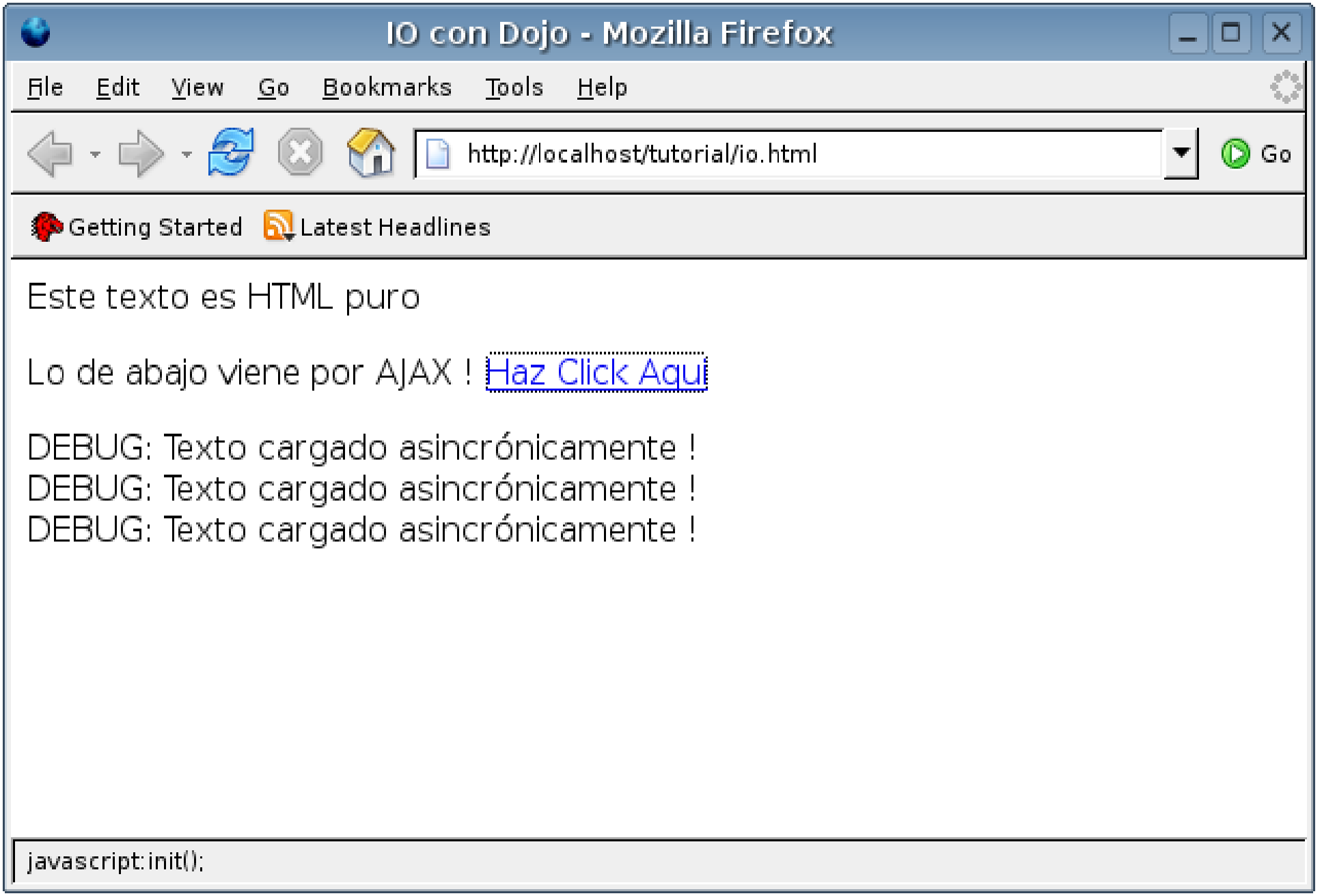
Cambio de Titulo

AJAX al modo fácil con DOJO

IO con Dojo

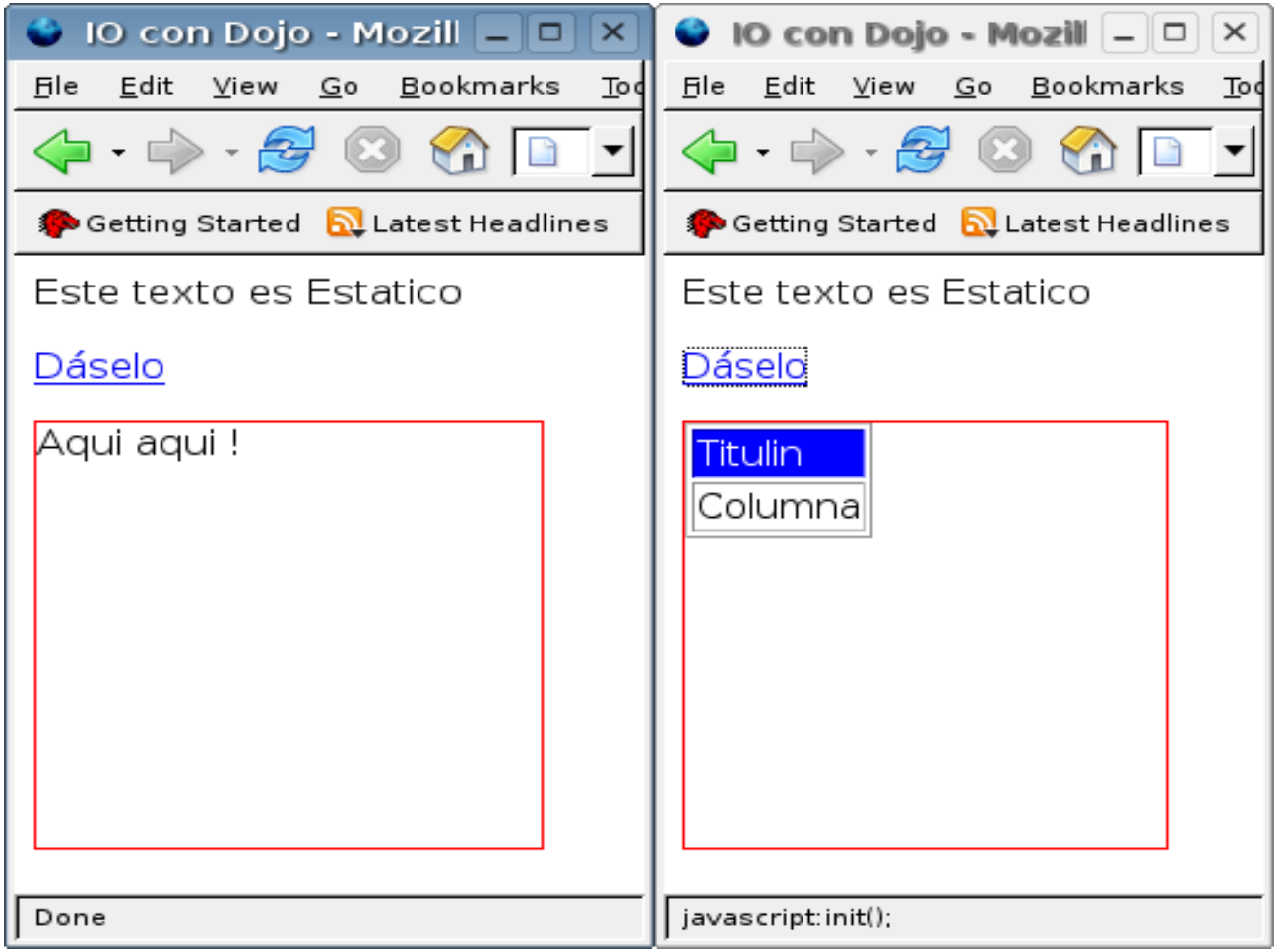
```
<html><head>
<script language="JavaScript" type="text/javascript">
djConfig = { isDebug: true };
</script>
<script language="JavaScript" type="text/javascript" src="../dojo/dojo.js"></script>
<script language="JavaScript" type="text/javascript">
dojo.require("dojo.io.*");
function init(){
    var bindArgs = {
        url: "io_test.txt",
        contentType: "text/plain",
        load: function(type, data, evt){
            dj_debug(data);
        }
    };
    var canBind = dojo.io.bind(bindArgs);
}
dojo.hostenv.modulesLoadedListeners.push(init);
</script>
</head>
<body>
<p>Este texto es HTML puro</p>
<p>Lo de abajo viene por AJAX ! <a href="javascript:init();">Haz Click Aqui</a></p>
</body>
</html>
```

Resultado


IO con Dojo - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/tutorial/io.html
Go
Getting Started
Latest Headlines
Este texto es HTML puro
Lo de abajo viene por AJAX ! Haz Click Aqui
DEBUG: Texto cargado asincrónicamente !
DEBUG: Texto cargado asincrónicamente !
DEBUG: Texto cargado asincrónicamente !
javascript:init();

Algo mas útil: importa HTML en ese cuadrito Rojo

```
var bindArgs = {
        url: "io_html.txt",
        contentType: "text/html",
        load: function(type, data, evt){
            dojo.byId('cont').innerHTML = data;
        }
    };
```



Ya nos estamos poniendo de acuerdo ?

- Como podemos ver es bastante sencillo hacer un sitio web 2.0
- En realidad esta pasando mucho por detras que no hemos visto, y definitivamente no vamos a verlo todo aqui, pero podemos habler un poco en el minuto que me queda de presentación (si es necesario !)
- Levanten la mano los que quieran que hable otro minuto mas

Lo que Dojo hizo por nosotros

```
function init(){
    var bindArgs = {                          // Arreglo de datos
        url: "io_test.txt",                   // URL a traer
        contentType: "text/plain",            // Tipo de respuesta
        load: function(type, data, evt){ // Handler
            dj_debug(data);                   // Génio de la lámpara
        }
    };
    var canBind = dojo.io.bind(bindArgs); // Peticion Real
}
```

Todavía falta mucho

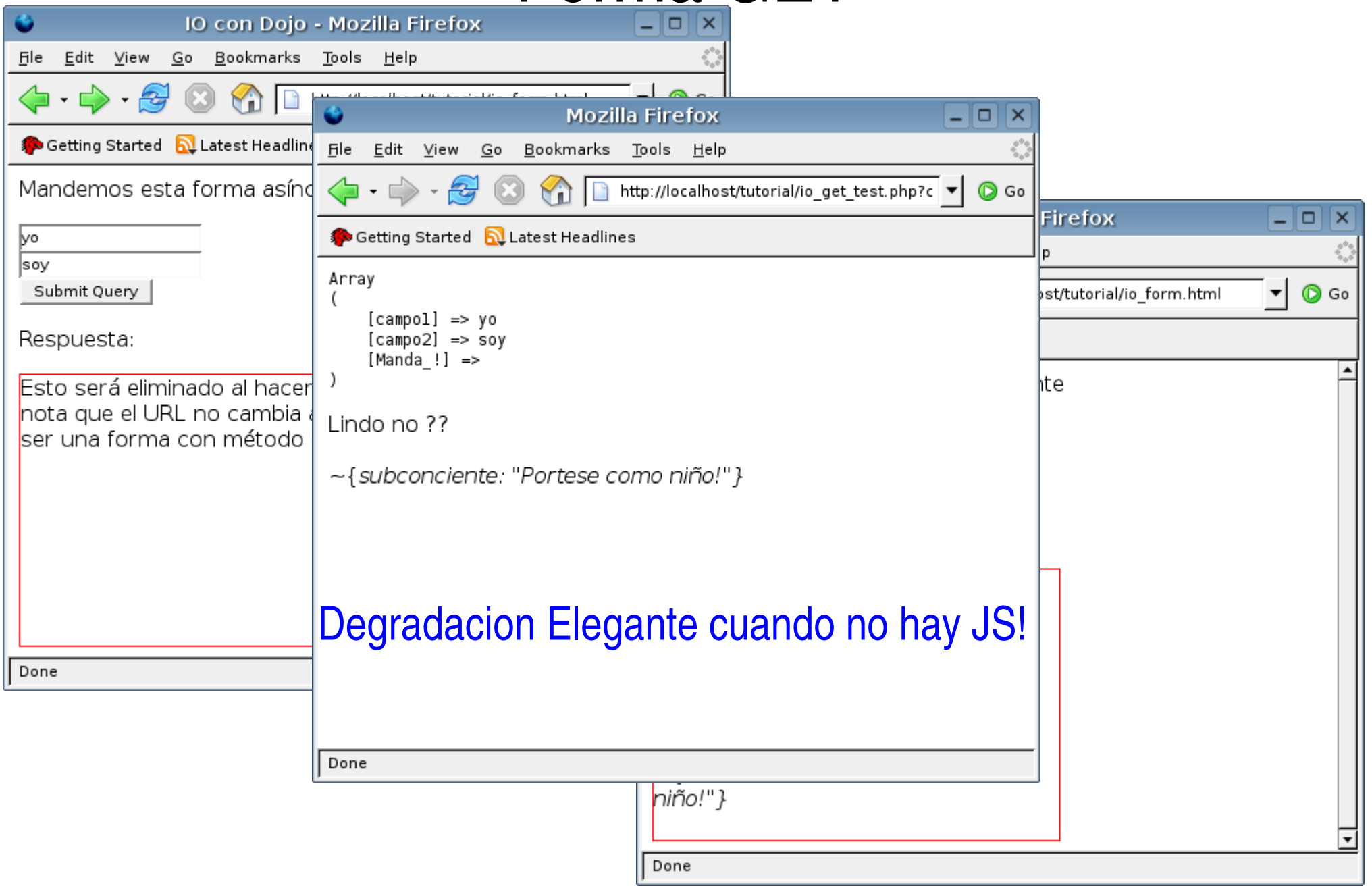
```
var bindArgs = {
    formNode: form,
    load: function(type, data, evt){
        dojo.byId('cont').innerHTML = data;
    }
 };
.............
<form name="formulario" method="get"
action="io_get_test.php"
onsubmit="init(this);return false;">
```

Podemos mandar una forma transparentemente, y con degradación elegante simplemente asignando el elemento formNode con la forma como valor.

Aqui mandamos la forma a la función que la maneja y deshabilitamos la respuesta por defecto.

Esto nos permite funcionar correctamente si hay o no hay JS

Forma GET


IO con Dojo - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Getting Started
Mandemos esta forma asínc
yo
soy
Submit Query
Respuesta:
Esto será eliminado al hacer
nota que el URL no cambia
ser una forma con método
Done
Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/tutorial/io_get_test.php?c
Go
Getting Started
Latest Headlines
Array
(
[campo1] => yo
[campo2] => soy
[Manda_!] =>
)
Lindo no ??
~{subconciente: "Portese como niño!"}
Degradacion Elegante cuando no hay JS!
Done
Firefox
Go
Done

Espectativas Reales

AJAX por si solo es una simple herramienta de entrada y salida de datos, el verdadero encanto de la tecnología WEB 2.0 (a.k.a WEB0.2) está en producir aplicaciones accesibles con maxima funcionalidad y un mínimo de codigo:

- En la práctica queremos poder recibir datos y código, los datos se presentan, pero nuevo código nos hace posible incrementar las características de la aplicación dependiendo de los requerimientos del usuario final
- No queremos repetir código, y necesitamos que sea mantenible y entendible
- Incluso queremos ser cool y manejar POO

Aplicaciones chidas con AJAX how-to

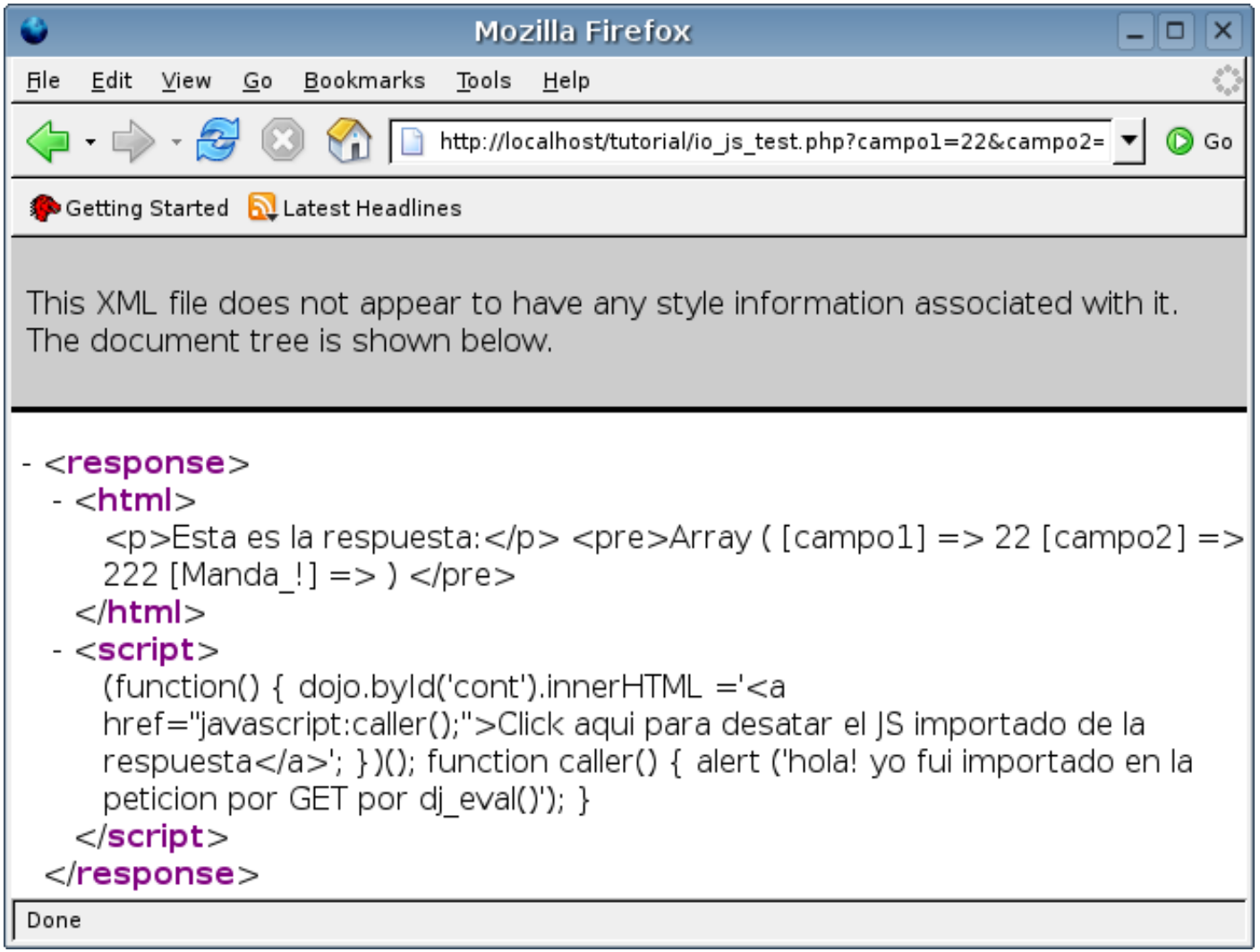
- Degradación elegante (Si es posible)
- Soporte para el botón de regreso
- Código limpio y compacto
- Dinamismo en las características
- Buenas ideas
- ...
- Profit ! (haste rico pues !)

Necesitamos funciones dinámicas en nuestra aplicación

- La respuesta es importar funciones de JS segun se necesite
- Hay diferentes maneras de hacerlo
  - XML
  - JSON
  - RPC

Primer paso

- Queremos importar funciones nuevas !!
- El siguiente XML sirve de PoC



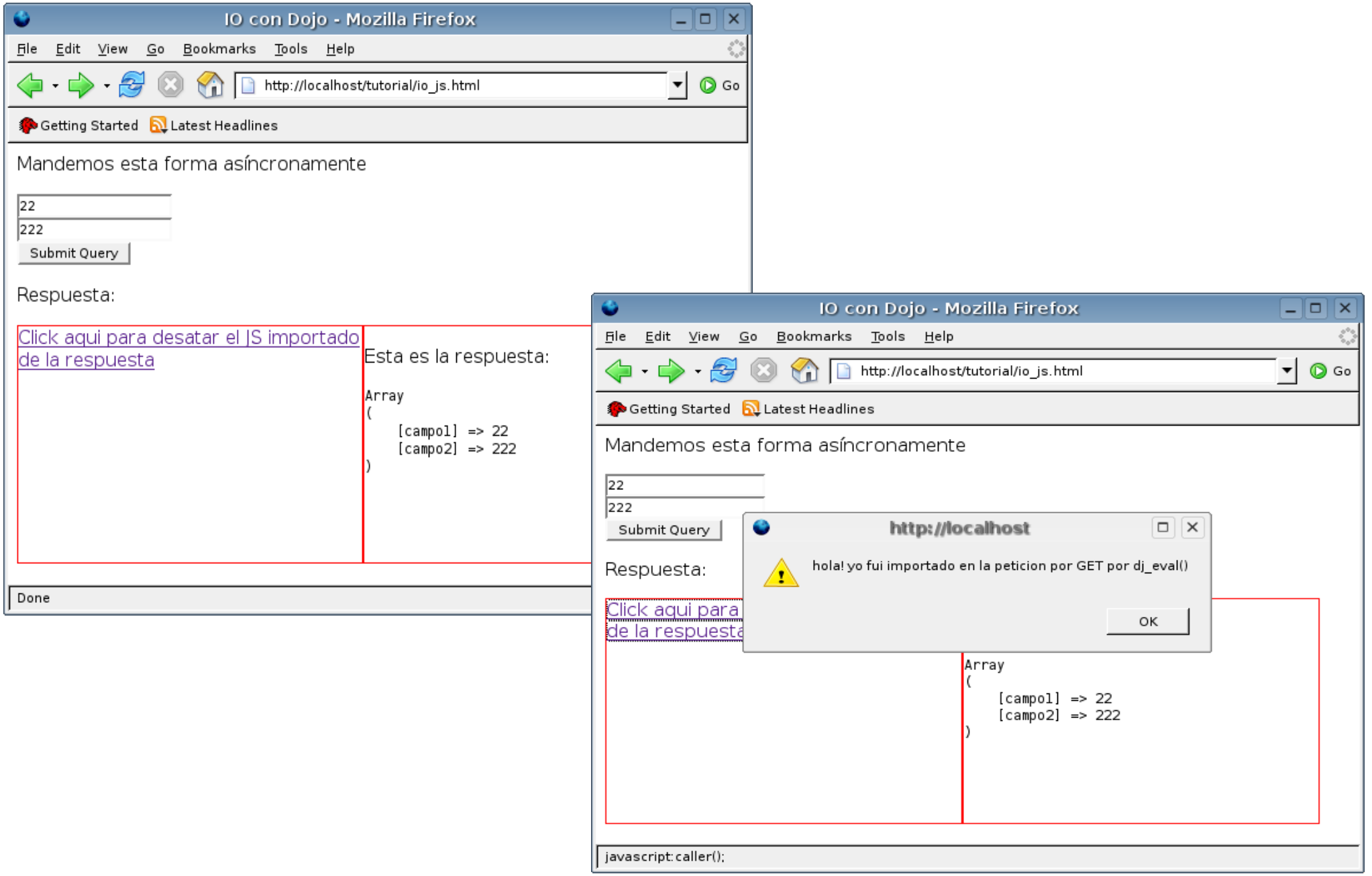
JS ház tuyo el XML !

```
var bindArgs = {
        formNode: form,
        mimetype: "text/xml",
        load: function(type, xml, evt){
            var html = xml.getElementsByTagName('html').item(0)
            dojo.byId('cont2').innerHTML = html.firstChild.nodeValue;
            var script = xml.getElementsByTagName("script").item(0);
            dj_eval(script.firstChild.nodeValue); // Executa el JS
        }
};
```

XML revisited

```
<response><html><![CDATA[
<p>Esta es la respuesta:</p>
<pre>Array
(
    [campo1] => 22
    [campo2] => 222
    [Manda_!] =>
)
</pre>]]>
</html>
<script><![CDATA[
(function() {
    dojo.byId('cont').innerHTML ='<a href="javascript:caller();">
        Click aqui para desatar el JS importado de la respuesta</a>';
})();
function caller() {
    alert ('hola! yo fui importado en la peticion por GET por dj_eval()');
}]]>
</script></response>
```

Show me the money



Formas con Archivos

Esta parte es truculenta.

El Objeto XMLHTTP no soporta envio de archivos, asi que una vieja técnica nunca reconocida como AJAX aunque lo merece sale al quite:

Uso de Iframes !!

Cómo lo hacemos con dojo ??

```
dojo.require("dojo.io.IframeIO");
```

Eso es todo

Eventos con Dojo

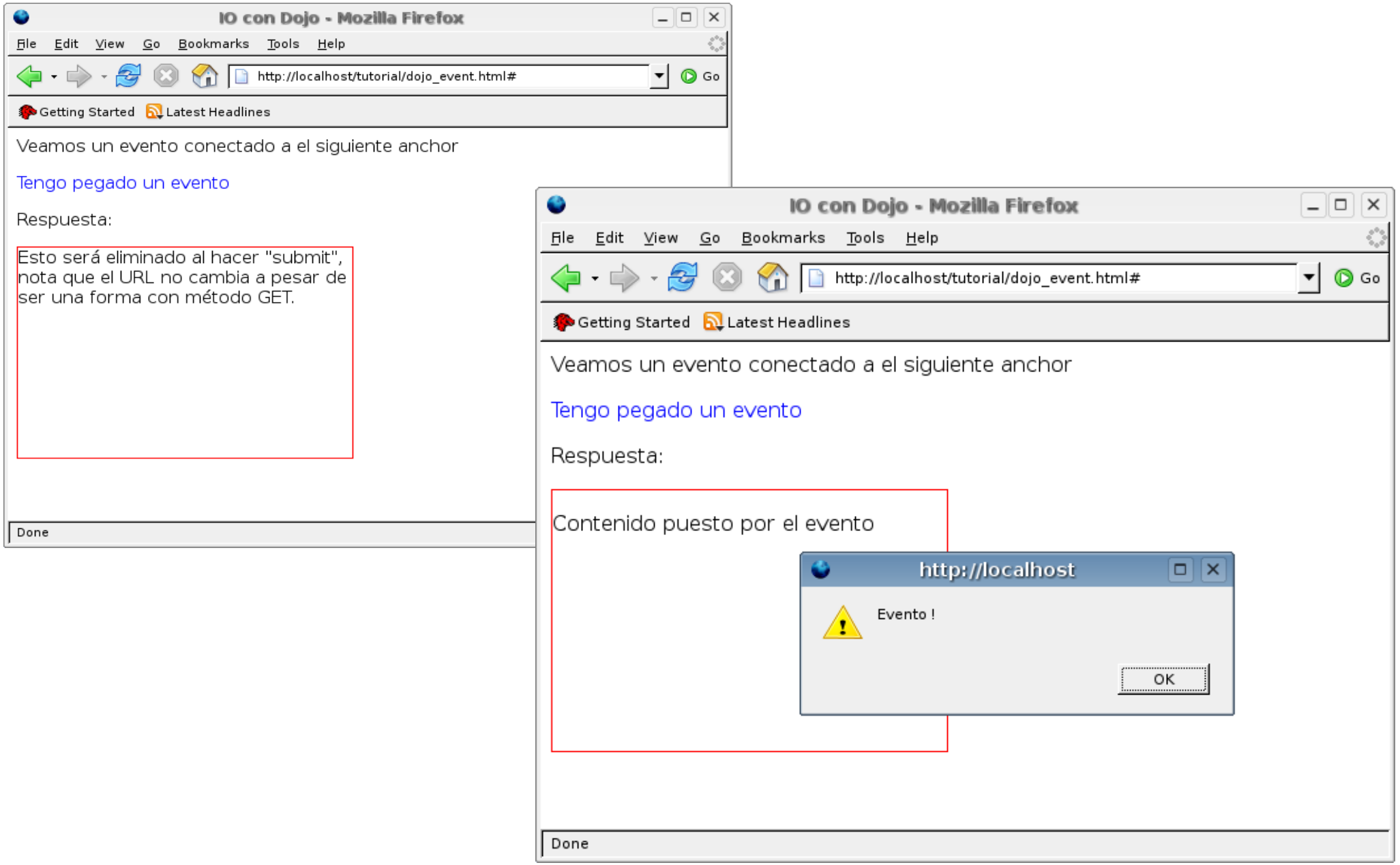
- Esta es una de las partes mas importantes, y tambien mas truculentas de una aplicacion con AJAX
- Mucho de lo que queremos hacer depende de tener “listeners” que puedan actuar dependiendo de algun evento.
- dojo.require("dojo.event.*"); es la solucion a casi cualquier cosa relacionada con eventos

Veamos el siguiente ejemplo:

```
function responde(){
    alert("Evento !");
}
function listen(){
    dojo.event.connect(dojo.byId('click'),"onclick",responde);
}
dojo.event.connect(dojo, "loaded", "listen");
```

Lo que estamos haciendo es llamar a la funcion listen() inmediatamente después de cargar todo el sistema Dojo. Esta funcion conecta al hijo (DOM) que tenga el ID “click” y lo hace correr la funcion responde(), la cual hace una tonteria.

Evento


IO con Dojo - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/tutorial/dojo_event.html#
Go
Getting Started
Latest Headlines
Veamos un evento conectado a el siguiente anchor
Tengo pegado un evento
Respuesta:
Esto será eliminado al hacer "submit", nota que el URL no cambia a pesar de ser una forma con método GET.
Done
IO con Dojo - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/tutorial/dojo_event.html#
Go
Getting Started
Latest Headlines
Veamos un evento conectado a el siguiente anchor
Tengo pegado un evento
Respuesta:
Contenido puesto por el evento
http://localhost
Evento !
OK
Done

Qué falta ?

- La mayor parte de una aplicación es la usabilidad
- El diseño atrae usuarios
- Las caracteristicas los hace quedarse
- La evolución nos lleva al cielo :-P

CSS y DOM

- Lo mas importante es saber usar CSS, necesitas crear una aplicación que luzca igual en por lo menos los 3 browsers mas populares
- Saber CSS implica que tienes nociones buenas de DOM
- Además DOM nos permite la manipulacion de la aplicación evitando refrescos de página
- La lectuda de XML reside mucho en DOM tambien
- DHTML es el resumen de todo esto, y XMLHTTP lo convierte en AJAX

Conclusiones

- AJAX provee un simple método de IO
- Utiliza DOM para hacer posible un dinamismo no posible con los métodos tradicionales de diseño web
- XML provee un medio de transporte flexible y robusto
- La imaginación es el límite
- Si usas AJAX eres cool (google, yahoo, m$...)
- Además no cualquier güevmaster le mueve a eso del AJAX

Conclusiones

- Aun cuando AJAX es tentador, fácilmente se puede caer en excesos, recuerda siempre que una aplicación flexible es mejor, si sigue funcionando correctamente sin JS puedes sentirte contento contigo mismo
- La mayoria de las aplicaciones web se pueden mejorar con un poco de AJAX, eso es una realidad
- Si tienes a alguien que diseña tan mal como yo no vas a llegar muy lejos

Preguntas Jóvenes

Gracias fue un placer!