jQuery, REST y otros frameworks

Adolfo Sanz De Diego

Octubre 2012

# Acerca de

## Pronoide



Pronoide

* Pronoide consolida sus servicios de formación superando las **22.000 horas impartidas** en más de 500 cursos (Diciembre 2011)
* En la vorágine de **tecnologías y marcos de trabajo existentes para la plataforma Java**, una empresa dedica demasiado esfuerzo en analizar, comparar y finalmente decidir cuáles son los pilares sobre los que construir sus proyectos.
* Nuestros Servicios de Formación Java permiten ayudarle en esta tarea, transfiriéndoles nuestra **experiencia real de más de 10 años**.

## Autor

* **Adolfo Sanz De Diego**
  + Correo: [asanzdiego@gmail.com](mailto:asanzdiego@gmail.com)
  + Twitter: [@asanzdiego](http://twitter.com/asanzdiego)
  + Linkedin: <http://www.linkedin.com/in/asanzdiego>
  + Blog: <http://asanzdiego.blogspot.com.es>

## Licencia

* **Este obra está bajo una licencia:**
  + [Creative Commons Reconocimiento-CompartirIgual 3.0](http://creativecommons.org/licenses/by-sa/3.0/es/)
* **El código fuente de los programas están bajo una licencia:**
  + [GPL 3.0](http://www.viti.es/gnu/licenses/gpl.html)

# jQuery

## ¿Qué es jQuery?

* Es una librería JavaScript que **simplifica el manejo del DOM** del HTML.
* Funciona seleccionando uno o varios elementos y ejecutando una acción sobre ellos.
* "**Write less, do more.**"
* La mejor API que he visto [jQuery Quick API Reference](http://oscarotero.com/jquery/)

## Selectores

* Utiliza **los mismos que CSS**, y alguno más propio.

## Eventos

* jQuery está pensado para **recoger y/o lanzar** eventos.
* Estos eventos normalmente son **eventos de ratón, de teclado**.
* También maneja los **eventos de cambio de estado** de algún elemento del DOM.

## CSS

* Podemos cambiar tanto el **atributo style**, como las **clases** de un elemento.

## Efectos

* **Ocultar, mostrar, desvanecer** elementos.
* También podemos hacer **animaciones** cambiando el CSS.
* Ver también [CSS3 transitions jQuery plugin](http://ricostacruz.com/jquery.transit/)

## HTML

* Podemos **cambiar el DOM** del HTML:
  + añadiendo texto tanto al principio como al final de un elemento,
  + cambiando el texto de un elemento,
  + añadiendo un elemento tanto antes como después de un elemento,
  + eliminando elementos.

## AJAX

* **Simplifica las peticiones AJAX**, pudiendo manejar su estado.

## Plugins

* Existe un **gran catálogo** de plugins.
* Los plugins **se crean de una forma muy sencilla**.
* Ver [Tutorial Oficial](http://docs.jquery.com/Tutorials:Getting_Started_with_jQuery)

# jQuery UI

## ¿Qué es jQuery UI?

* Son un conjunto de **componentes visuales**, con [temas personalizables](http://jqueryui.com/themeroller/).
* Las clases CSS se pueden reutilizar. Ver [The jQuery UI CSS Framework](http://docs.jquery.com/UI/Theming/API)
* Las demos están muy bien, con muchos y muy buenos ejemplos [jQuery UI Demos](http://jqueryui.com/demos/)

## Componentes

* **Interactions**: draggable, droppable, resizable, selectable, sortable
* **Widgets**: accordion, autocomplete, button, datepicker, dialog, menu, progressbar, slider, spinner, tabs, tooltip
* **Efects**: blind, bounce, clip, drop, explode, fade, fold, highlight, pulsate, scale, shake, slide, transfer

# jQuery Mobile

## ¿Qué es jQuery Mobile?

* Son componentes pensados para **aplicaciones móviles**.
* Están pensados para el manejo de **eventos táctiles**.

## Ejemplos

* Toda la documentación del site está hecha con el propio framework.
* Ver [jQuery Mobile](http://jquerymobile.com/)
* Ver [jQuery Mobile Documentation](http://jquerymobile.com/demos/)

# REST

## ¿Qué es REST?

* REST (Representational State Transfer) es una técnica de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web.
* Es decir, **una URL (Uniform Resource Locator) representa un recurso al que se puede acceder o modificar mediante los métodos del protocolo HTTP (POST, GET, PUT, DELETE)**.
* Ver [Artículos de REST de Enrique Amodeo Rubio (@eamodeorubio)](https://eamodeorubio.wordpress.com/category/webservices/rest/)

## ¿Por qué REST?

* Es **más sencillo** (tanto la API como la implementación).
* Es **más rápido** (peticiones más lijeras que se puede cachear).
* Es **multiformato** (HTML, XML, JSON, etc.).
* Se complementa muy bien con **AJAX**.

## Ejemplo API

|  |  |  |
| --- | --- | --- |
| GET | -> http://myhost.com/person | -> Devuelve todas las personas |
| POST | -> http://myhost.com/person | -> Crear una nueva persona |
| GET | -> http://myhost.com/person/123 | -> Devuelve la persona con id=123 |
| PUT | -> http://myhost.com/person/123 | -> Actualiza la persona con id=123 |
| DELETE | -> http://myhost.com/person/123 | -> Borra la persona con id=123 |

## Manejo de errores

* **Se pueden utilizar los errores del protocolo HTTP**:
  + 200 OK Standard response for successful HTTP requests
  + 201 Created
  + 202 Accepted
  + 301 Moved Permanently
  + 400 Bad Request
  + 401 Unauthorised
  + 402 Payment Required
  + 403 Forbidden
  + 404 Not Found
  + 405 Method Not Allowed
  + 500 Internal Server Error
  + 501 Not Implemented

# Otros frameworks

## backbone.js

* [backbone.js](http://backbonejs.org/) implementa el **paradigma MVC en cliente**.
* Permite crear un **modelo que se actualiza con el servidor (mediante REST y AJAX)**, **y que refresca las vistas que tenga asociadas en el cliente**.
* Ver [Tutorials, blog posts and example sites](https://github.com/documentcloud/backbone/wiki/Tutorials%2C-blog-posts-and-example-sites)
* Ver [Tutplus Backbone Tutorial](http://net.tutsplus.com/sessions/build-a-contacts-manager-using-backbone-js/)

## require.js

* [require.js](http://requirejs.org/) permite la **gestión de dependencias** entre módulos.
* Los **módulos se cargan de forma asíncrona** (según se vayan necesitando).
* Ver [Organizing your application using Modules with require.js](http://backbonetutorials.com/organizing-backbone-using-modules/)

## dust.js

* [dust.js](http://akdubya.github.com/dustjs) es un **motor de plantillas** en JavaScript.

## lesscss.js

* [lesscss.js](http://lesscss.org/) permite crear **archivos CSS de forma programática**.
* Se pueden usar archivos .less y la librería JavaScript (desarrollo) o "compilar" los archivos .less en archivos .css (producción)
* Aquí hay [un buen tutorial sobre less](http://ivanmendoza.net/diseno-web-2/introduccion-less-css)

## modernizr.js

* [modernizr.js](http://modernizr.com/) **detecta las características CSS3 y HTML5 que soporta el navegador**.