

## Introducción a Oracle JET (Oracle JavaScript Extension Toolkit)

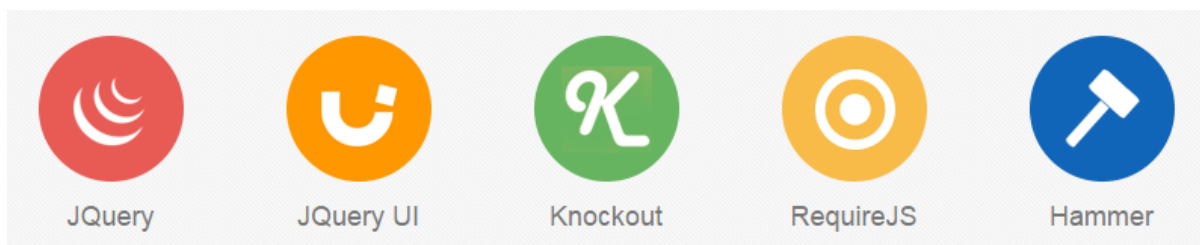
APPLICATION DEVELOPMENT 28/10/2015 - Rubén Rodríguez

El pasado viernes fue un día cargado de nuevas releases de productos ([ver post acerca de la nueva versión de Oracle ADF](#)) y como no podía ser menos, vamos a ver **Oracle JavaScript Extension Toolkit**, más conocido como **Oracle JET**, el nuevo Framework JavaScript que ha liberado Oracle.



***Oracle JET** está dirigido a desarrolladores JavaScript que trabajan en aplicaciones en el lado del cliente, y es una colección de librerías JavaScript de código abierto que junto a unas librerías propias de Oracle hacen que crear aplicaciones que se consumen e interactúan con otros productos Oracle sea muy simple y eficiente.*

Oracle JET es un framework para desarrollar aplicaciones basado en una serie de librerías *Open Source*:



- **JQuery**

Es una librería JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM y manejar eventos entre otras características.

- **JQuery UI**

Es una librería de componentes construidos utilizando JQuery, nos proviene de diferentes Widgets y nos permite fácilmente interactuar con los componentes (redimensionar, arrastras y soltar, etc) y añadir transiciones y efectos.

- **Knockout**

Es una librería JavaScript que permite crear interfaces de usuario responsive utilizando el patrón Model-View-ViewModel (MVVM).

- **RequireJS**

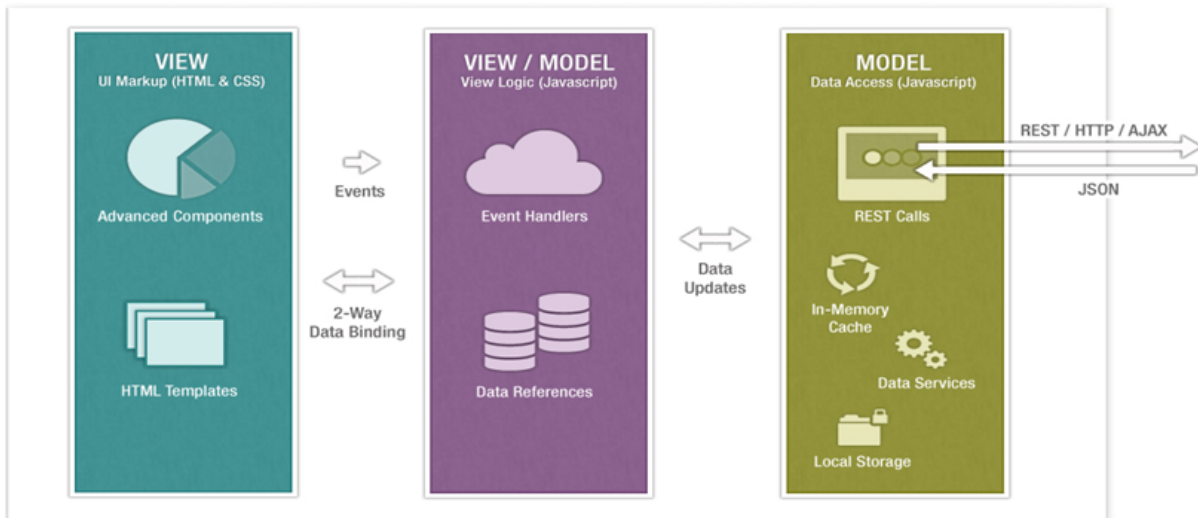
Es una librería JavaScript que nos permite separar en módulos los componentes de nuestra aplicación y resolver las dependencias de estos mismos.

- **Hammer**

Es una librería que añade el soporte de gestos táctiles en tablets o móviles y de eventos del ratón en aplicaciones web.

**Oracle JET tiene una arquitectura MVVM (Model-View-ViewModel).**

En una arquitectura MVVM, el modelo representa los datos de la aplicación, la View es la presentación de los datos y el ViewModel expone datos del modelo a la vista y mantiene el estado de la aplicación.



**El Oracle JET Common Model y la Collection API implementan la capa de modelo.** Esta API incluye los siguientes objetos JavaScript:

- **oj.Model** : Representa un solo registro de datos de un servicio de datos como puede ser un servicio RESTful.
- **oj.Collection** : Representa un conjunto de registros de datos y es una lista de objetos oj.model del mismo tipo.
- **oj.Events** : Ofrece métodos para manejar eventos.
- **oj.KnockoutUtils** : Contiene metodos para mapear atributos de un objeto oj.Model o un objeto oj.Collection con los objetos Knockout para que puedan ser utilizados por los modelos de los objetos.

Para implementar la capa **View**, **OracleJET ofrece una colección de componentes encapsulados como widgets de JQuery UI**, desde unos simples botones hasta componentes de visualización avanzados como gráficas.

Por último, **Knockout.js implementa la capa ViewModel** y ofrece la asociación de datos bidireccional entre las capas Model y View.

Aparte, Oracle JET entre otras muchas, características como éstas:

- Un framework de validación que ofrece validación a nivel de componente y conversión de datos.
- Una cache en la capa Model para optimizar el rendimiento de la paginación.
- Conexión a orígenes de datos a través de servicios web como por ejemplo REST o WebSocket.
- Autorización a través de OAuth 2.0 para modelos de datos obtenidos a través de servicios REST.

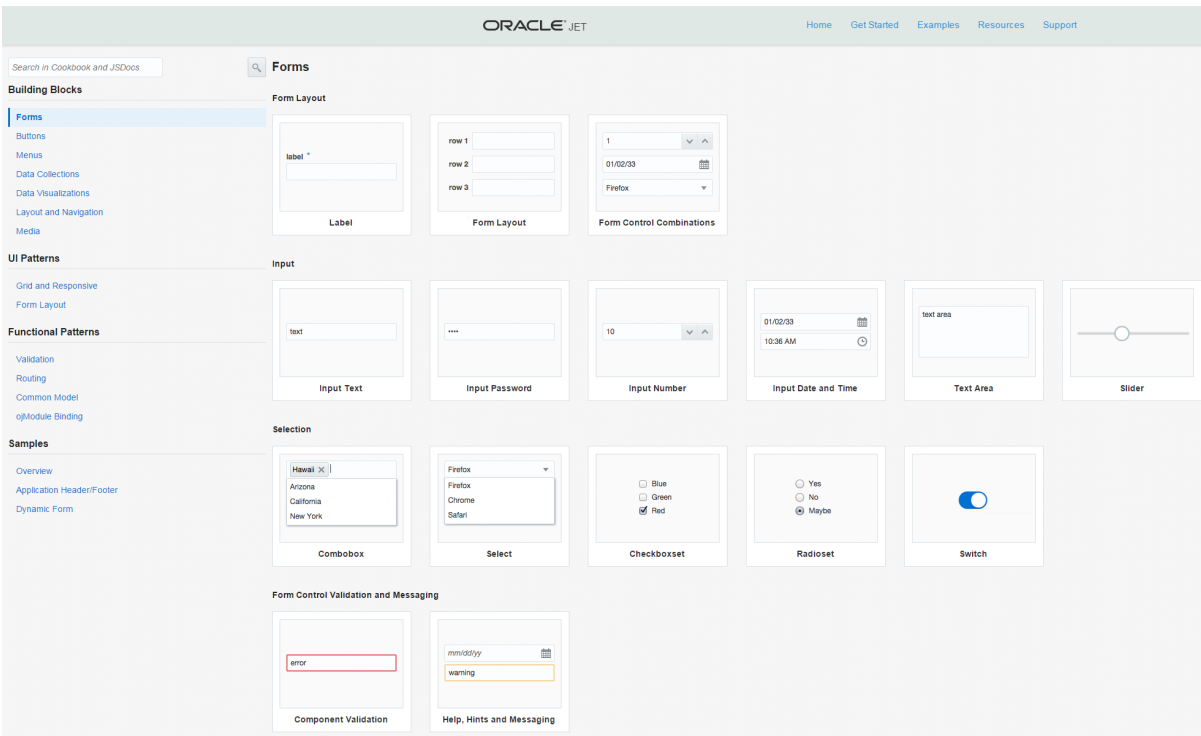
Para empezar a utilizar Oracle JET, Oracle ha puesto a disposición de todo el que esté interesado los siguientes recursos:

- **Una demo de una aplicación Oracle JET**



- **Un Cookbook**

Podremos ver los diferentes componentes Oracle JET y el código JavaScript y HTML necesario para que funcione.



También es posible editar al momento el código HTML y JavaScript y ver el resultado en la misma página.

Toggle Button: Single Checkbox
API Documentation: [ojButtonset](#)

Basic Recipe

Developer's Guide

This demo features a single checkbox-based JET Button, both standalone and in a JET Buttonset.

Standalone (No Buttonset)

Toggle

Utilizing Buttonset's "checked" binding

Advanced mode

User is: a beginner

Recipe

- Using the `ojComponent` binding, create a toggle button from a `<label>` and `<input type="checkbox">`. In that order.
- If a checked binding is needed for the button, wrap it in a JET Buttonset. See the [Checkbox Set demo](#) and [Radio Group demo](#) for details on creating Buttonsets.

HTML Editor

Apply Changes

```

1<div id="buttons-container">
2  <br><!-- ----- -->
3
4  <h3>Standalone (No Buttonset)</h3><br>
5
6  <label for="standalone">Toggle</label>
7  <input type="checkbox" id="standalone" data-bind="ojComponent: {component: 'ojButton'}"/>
8  <br></br>><br><!-- ----- -->
9
10 <h3>Utilizing Buttonset's "checked" binding</h3><br>
11
12 <div id="advancedWrapper"
13   data-bind="ojComponent: {component: 'ojButtonset', checked: isAdvanced}">
14   <label for="advanced">Advanced mode</label>
15   <input type="checkbox" id="advanced" value="advanced"/>
16 </div>
17
18 <br><br>
19 <p class="bold" id="userText" data-bind="text: userText"/>
20
21</div>

```

JS Editor

Apply Changes

```

1require(['ojs/ojcore', 'knockout', 'jquery', 'ojs/ojknockout', 'ojs/ojbutton'],
2function(oj, ko, $)
3{
4
5  function ButtonModel() {
6    this.isAdvanced = ko.observableArray([]);
7    this.userText = ko.computed(function() {
8      return User.is: " " + (this.isAdvanced().length ? "an expert" : "a beginner");
9    }, this);
10  }
11
12  $(function() {
13    ko.applyBindings(new ButtonModel(), document.getElementById("buttons-container"));
14  });
15
16  });
17

```

Para más información y descargas hay que dirigirse a