



M E A N

WEB FULL STACK DEVELOPER

Germán Caballero Rodríguez
gcaballero@pronoide.es





jQuery

mobile



APACHE
CORDOVA™

ThemeRoller
For jQuery Mobile



INDICE

1) Componentes

Componentes

Roles

- Es un **Framework JavaScript** para el **desarrollo rápido y fácil** de sitios webs optimizados para teléfonos móviles. Con este framework, **aceleramos la velocidad de desarrollo** de aplicaciones, **encapsulando muchas tareas comunes** que se realizan cuando usamos el lenguaje JavaScript. **Agrega una capa más a JQuery** e intenta suplir algunas necesidades que los programadores de dispositivos móviles padecen

Componentes

Roles

- Parte principal del API.
- Definen que rol juega el elemento en la pagina Web.
- Se aplican con el atributo **data-role**

```
<div data-role="page"></div>
```

- Pueden ser entre otros
 - page
 - header
 - content
 - footer
 - button

Componentes

Páginas

- El atributo “data-role” con valor “page”, indica que el contenido dentro de la etiqueta con dicho atributo, corresponde a una pagina, por lo que en cada momento solo se muestra el contenido de dicha pagina.



Componentes

Páginas

- Para poder visualizar otra pagina, se ha de identificar la etiqueta con atributo **data-role="page"**, con un **id** y crear un vinculo hacia ella.

```
<div data-role="page" id="home">  
    <a href="#acercaDe">acerca de</a>  
</div>  
<div data-role="page" id="acerdaDe">  
    <a href="#home">Inicio</a>  
</div>
```

Componentes

Páginas

- En caso de que la página sea externa, se procede de forma similar, la diferencia está en el valor del atributo href, que en este caso será la URL.

```
<div data-role="page" id="home">  
    <a href="AcercaDe.html">acerca de</a>  
</div>
```

- Si no existe ningún elemento con data-role=page, el framework, envuelve todo el contenido dentro de un div con data-role=page.
- Aunque esta situación no es recomendable.

Componentes

Páginas

- Cuando se navega a una pagina externa, se puede modificar la url con la que se carga dicha pagina incluyendo en la etiqueta con `data-role="page"`, la etiqueta `data-url="/<nuevo path de pagina>"`.
- Se puede definir un titulo para la pagina con `data-title="<titulo de la pagina>"`.

```
<div data-role="page" id="pagina2" data-url="/nuevapagina/">
```

```
<div data-role="page" id="pagina2" data-title="pagina2">
```

Componentes

Páginas

- Se puede hacer que e la cabecera salga de forma automática un botón de vuelta, añadiendo a la pagina el atributo **data-add-back-btn="true"**.

```
<div data-role="page" id="pagina2" data-add-back-btn="true">
```

- Este botón es añadido a la cabecera de la pagina, siempre que exista y haya una pagina a la que retornar.
- Se puede definir también **data-back-btn-text="previous"** y **data-back-btn-theme="b"**.

Componentes

Paginas: AJAX

- No se recomienda realizar una navegación AJAX (la por defecto) a htmls, con multi-paginas, ya que al cargar solo la pagina principal y no todo el html, puede provoca errores de funcionamiento en diálogos o al navegar a las paginas internas.
- Para evitar este efecto, se ha de configurar la navegación sin AJAX, para ello se ha de definir o bien **rel="external"** para sitios externos en otro dominio, o **data-ajax="false"** para paginas internas sin comportamiento AJAX, o **target=""** cuando se carga el contenido en otra pestaña y por tanto no se emplea el mismo DOM.
- Por defecto el framework, siempre accede a los sitios externos sin emplear AJAX.
- Cuando se aplican estos atributos, se pierden las transiciones.

Componentes

Diálogos

- La forma recomendada de mostrar un dialogo, es aplicando a la pagina que conforma el dialogo, la propiedad **data-dialog="true"**

```
<div data-role="page" data-dialog="true" id="dialogo">...</div>
```

- Otra opción ya **deprecated**, es aplicar la propiedad **data-rel**, con valor **dialog** al enlace que abre el dialogo.

```
<a href="#dialogo" data-rel="dialog">abrir dialogo</a>
```

- Normalmente el contenido del dialogo, se define como una pagina interna.
- Los diálogos, no se almacenan como pagina, si en el momento de visualizar el dialogo se guarda en favoritos.

Componentes

Diálogos

- Se puede definir un botón de cerrar en la **cabecera** del dialogo, añadiendo **data-close-btn="right"**

```
<div data-role="page" data-close-btn="right" data-dialog="true">  
  <div data-role="header">...</div>  
</div>
```

Componentes

Diálogos

- Para crear un botón de cerrar personalizado, únicamente habrá que añadir un botón con **data-rel="back"**

```
<div data-role="page" data-close-btn="right" data-dialog="true">  
  <div data-role="header">...</div>  
  <div data-role="content">  
    <a href="#" data-role="button" data-rel="back">  
      Volver  
    </a>  
  </div>  
</div>
```

Componentes

Diálogos

- Para establecer el texto del botón de cerrar, se puede indicar con la propiedad **data-close-btn-text=""**
- O afectando a todos los botones de todos los diálogos, estableciendo la propiedad global

```
$.mobile.dialog.prototype.options.closeBtnText
```

- Se puede establecer un tema diferente para el fondo sobre el que aparece el dialogo con **data-overlay-theme** aplicado sobre la pagina del dialogo.

Componentes

Popup

- Para añadir un popup a la pagina, se ha de anotar un componente con **data-role="popup"** y el link que lo abra con **data-rel="popup"**.

```
<div data-role="popup"></div>
```

- Se pueden abrir de forma programática con el método popup, al que se le pueden pasar dos parámetros
 - "open"
 - "close"
- Cuando se abre, adicionalmente se puede definir una opciones de configuracion
 - **x**: Coordenada X.
 - **y**: Coordenada Y.
 - **transition**: Transición para mostrar el popup.
 - **positionTo**.

Componentes

Botones

- Se consiguen aplicando a
 - enlaces, etiquetas `<a>`
 - botones, etiquetas `<input type="button">`
 - botones, etiquetas `<input type="submit">`
- el rol **button**.

```
<a href="AcercaDe.html" data-role="button">acerca de</a>
```

```
<input type="button" data-role="button" value="Acerca De"/>
```

```
<input type="submit" data-role="button" value="Acerca De"/>
```

Componentes

Botones

- Alternativamente, se puede aplicar la clase CSS **ui-btn**
- Si además se desean aplicar redondeos, se aplica también la clase CSS **ui-corner-all**.

```
<a href="AcercaDe.html" class="ui-btn">acerca de</a>
```

- Si se quiere mostrar un icono se añade **ui-icon-delete**, y si se quiere quitar el texto **ui-btn-icon-notext**

```
<a href="AcercaDe.html" class="ui-btn ui-corner-all">acerca de</a>
```

- Para añadir sombras, se añade **ui-shadow**
- Para ver los botones en la misma línea **ui-btn-inline**

```
<a href="AcercaDe.html" class="ui-btn ui-corner-all ui-icon-delete ui-btn-icon-notext">acerca de</a>
```

Componentes

Botones

- Para aplicar un estilo diferente al de la pagina **ui-btn-<tema>**.
- Para ver el formato mini **ui-btn-mini**.
- Para posicionar los iconos **ui-btn-icon-left**.
- Para deshabilitar **ui-state-disabled**.

Componentes

Botones

- Para aplicar un estilo diferente al de la pagina **ui-btn-<tema>**.
- Para ver el formato mini **ui-btn-mini**.
- Para posicionar los iconos **ui-btn-icon-left**.
- Para deshabilitar **ui-state-disabled**.

Componentes

Botones: Iconos












































- Se puede definir un icono en el botón, con el atributo **data-icon**

```
<a href="AcercaDe.html" data-role="button" data-icon="plus">acerca de</a>
```

- Algunos de los iconos definidos
 - delete (x)
 - arrow-l (<)
 - arrow-r (>)
 - arrow-u (^)
 - arrow-d (v)
 - plus (+)
 - minus (-)
 - check (v)
 - gear (*)

Componentes

Botones: Iconos

 action	 alert	 arrow-d
 arrow-d-l	 arrow-d-r	 arrow-l
 arrow-r	 arrow-u	 arrow-u-l
 arrow-u-r	 audio	 back
 bars	 bullets	 calendar
 camera	 carat-d	 carat-l
 carat-r	 carat-u	 check
 clock	 cloud	 comment
 delete	 edit	 eye
 forbidden	 forward	 gear
 grid	 heart	 home
 info	 location	 lock
 mail	 minus	 navigation
 phone	 plus	 power
 recycle	 refresh	 search
 shop	 star	 tag
 user	 video	

Componentes

Botones: Iconos

- Se pueden definir nuevos iconos, generando una clase css, que tenga el siguiente formato **ui-icon-<nombre icono>**, que defina una imagen de 18x18 como imagen de fondo

```
.ui-icon-custom-icon {  
    background-image: url("custom-icon.png");  
}
```

Componentes

Botones: Iconos

- Por defecto los iconos se sitúan en la parte izquierda.
- Se puede modificar su posición con **data-iconpos**, que puede tomar valores
 - right
 - left
 - top
 - bottom
 - notext

```
<input type="button"  
      data-icon="delete"  
      data-iconpos="notext"  
      value="Icon only"/>
```


Componentes

Botones: Agrupación

- Para visualizar agrupados los botones, se emplea **data-role="controlgroup"**.

```
<div data-role="controlgroup">  
  <a href="#nuevo">Nuevo</a>  
  <a href="#borrar">Borrar</a>  
  <a href="#editar">Editar</a>  
</div>
```

- Por defecto la distribución es en vertical, para verlo en horizontal, se aplica la propiedad **data-type="horizontal"**

```
<div data-role="controlgroup" data-type="horizontal" >  
  <a href="#nuevo">Nuevo</a>  
  <a href="#borrar">Borrar</a>  
  <a href="#editar">Editar</a>  
</div>
```

Componentes

Transiciones

- Se obtiene con el atributo **data-transition**.
- El atributo se aplica sobre el elemento que dispara la navegación, por ejemplo el `<a>`.

```
<a href="AcercaDe.html" data-transition="slideup">acerca de</a>
```

- Puede tener los siguientes valores.
 - slideup
 - Slidedown
 - slide (movimiento hacia un lado)
 - fade (desaparece)
 - flip (giro de pantalla)
 - pop (aparición desde el centro de la pantalla)

Componentes

Transiciones

- Se obtiene con el atributo **data-transition**.
- El atributo se aplica sobre el elemento que dispara la navegación, por ejemplo el <a>.

```
<a href="AcercaDe.html" data-transition="slideup">acerca de</a>
```

- Puede tener los siguientes valores.
 - slideup
 - slide
 - fader
- Para cuando se quiere que los enlaces que vuelven a una pagina anterior, realicen el efecto contrario de transición, se emplea **data-direction="reverse"**

```
<a href="AcercaDe.html" data-direction="reverse">acerca de</a>
```

Componentes

Transiciones

- También se puede emplear para volver a la pagina anterior **data-rel="back"**, que hará lo mismo que el botón atrás del navegador, deshacer la navegación, y por defecto aplicara la transición inversa a la aplicada al llegar a la pagina actual.

```
<a href="AcercaDe.html" data-rel="back">acerca de</a>
```

Componentes

Cabecera

- Aunque cuando se inserta en una cabecera una etiqueta `<a>`, se crea por defecto un botón y no es necesario aplicar el rol **`data-role="button"`** o los estilos CSS, a partir de 1.4 se desaconseja.
- Aunque los botones se van colocando pegados uno a la izquierda y el siguiente a la derecha, repitiendo en otra fila el mismo proceso si hay mas de dos botones, este comportamiento esta desaconsejado, por lo que se han de incluir los estilos.
- Los elementos `<h1>`, los coloca centrados por defecto.
- Si se quiere anular los estilos, colocación por defecto, se puede emplear **`data-role="none"`**.

Componentes

Cabecera

- Si se quiere situar el botón a la derecha, se puede emplear la clase CSS **class="ui-btn-right"** y para la izquierda **ui-btn-left**
- Se puede aplicar tanto a cabecera como pie de pagina, que se oculten al clickar sobre la pagina con **data-fullscreen="true"**.
- Sino se incluye una titulo en la cabecera, etiquetas H, se ha de incluir un elemento **span**, con el estilo CSS **class="ui-title"**, para que se pinte correctamente los botones dentro de la cabecera.

```
<span class="ui-title" />
```

Componentes

Pie de pagina

- Al igual que con la cabecera, no es necesario poner el rol **data-role="button"** a la etiqueta `<a>` para mostrar el estilo de botón.
- Los botones los coloca in-line.
- Los botones, quedan ajustados a la zona dedicada al pie de pagina, se puede aplicar un estilo css, que deja unos márgenes con la zona de Pie de Pagina, este es **ui-bar**.

```
<div data-role="footer" class="ui-bar">  
  <a href="#nuevo" >nuevo</a>  
  <a href="#borrar" >borrar</a>  
  <a href="#editar" >editar</a>  
</div>
```

Componentes

Pie de pagina

- Al igual que con la cabecera y el pie, para definir el contenido de una pagina, se tiene **data-role="content"**.

```
<div data-role="content" ></div>
```

- A partir de la versión 1.4, se sustituye por el atributo HTML5 **role="main"**, en conjunto con las clase CSS **class="ui-content"**.

```
<div role="main" class="ui-content"></div>
```


Componentes

NavBar

- Se pueden aplicar tanto a Heder como a Footer.
- Se aplica con el rol **navbar**.
- Dentro del elemento con el rol **navbar**, se crea una lista ordenada o no, etiqueta `` o ``.

```
<div data-role="header" class="ui-bar">  
  <div data-role="navbar">  
    <ul>  
      <li><a href="#nuevo" >nuevo</a></li>  
      <li><a href="#borrar" >borrar</a></li>  
      <li><a href="#editar" >editar</a></li>  
    </ul>  
  </div>  
</div>
```

Componentes

NavBar

- En una misma línea, se pueden poner máximo 5 botones, de pone mas, se reorganiza en columnas.
- El uso típico, es el de navegación por pestañas, para que la primera pestaña este seleccionada, se puede emplear el estilo CSS **class="ui-btn-active"**.

Componentes

NavBar

- En una misma línea, se pueden poner máximo 5 botones, de pone mas, se reorganiza en columnas.
- El uso típico, es el de navegación por pestañas, para que la primera pestaña este seleccionada, se puede emplear el estilo CSS **class="ui-btn-active"**.

Componentes

Accordion

- Se pueden agrupar elementos ocultables, con el rol **collapsible-set**.

```
<div data-role="collapsible-set">
  <div data-role="collapsible" data-collapsed="false">
    <h3>Section 1</h3>
    <p>I'm the collapsible set content for section 1.</p>
  </div>
  <div data-role="collapsible">
    <h3>Section 2</h3>
    <p>I'm the collapsible set content for section 2.</p>
  </div>
</div>
```

Componentes

Grids

- Para poder distribuir de forma uniforme los contenidos en la pantalla, se recurre a unos estilos CSS predefinidos.
- Así se ha de definir un bloque html, típicamente un div, que tenga como class CSS, alguna de las siguientes.
 - **ui-grid-solo**: Una columna.
 - **ui-grid-a**: Dos columnas
 - **ui-grid-b**: Tres columnas
 - **ui-grid-c**: Cuatro columnas.
 - **ui-grid-d**: Cinco columnas

Componentes

Grids

- Dentro de esta etiqueta contenedora, se han de definir otros bloques html contenedores, con estilos CSS como
 - **ui-block-a**: Primera celda.
 - **ui-block-b**: Segunda celda.
 - **ui-block-c**: Tercera celda.
 - **ui-block-d**: Cuarta celda.
 - **ui-block-e**: Quinta celda.
- Para definir una nueva fila, se ha de repetir la misma secuencia, es los elementos de la primera columna, son siempre **ui-block-a**.

Componentes

Grids

- Ejemplo

```
<div class="ui-grid-a">
  <div class="ui-block-a">
    <div class="ui-bar ui-bar-a" style="height:60px">
      Block A
    </div>
  </div>
  <div class="ui-block-b">
    <div class="ui-bar ui-bar-a" style="height:60px">
      Block B
    </div>
  </div>
</div>
```

Componentes

Panel

- Son los elementos que permiten crear menús, columnas plegables, ... Es decir aquellos elementos que se salen de la distribución típica de la página.
- Se añaden con **data-role="panel"**.
- Han de ser hijos de **data-role="page"**, pero has de estar definidos antes o después de todos los elementos de **header**, **footer** o **content**.
- Los paneles llevarán asociado un **id**, que será empleado por los enlaces, para mostrarlos, de forma análoga a como se hace con las páginas.
- Se puede definir la propiedad **data-position**, con valores **right** y **left**, que indica por donde aparecerá el panel.

Componentes

Panel

- Se puede definir **data-display**, que define la interacción del panel con la pagina, puede tomar valores
 - **reveal**: Da la sensación de que el panel esta bajo la pagina y esta lo muestra mientras se desliza.
 - **push**: Da la sensación de que el panel esta pegado a la pagina, y se muestra en el grado que desaparece la pagina, moviéndose los dos a la vez.
 - **overlay**. Se muestra sobre la pagina tapando lo que haya debajo
- Se puede activar o desactivar la animación con **data-animate="false"**.

Componentes

Panel

- Para mostrar el contenido del panel, que se haya añadido de forma dinámica, hay que lanzar.

```
$( "#mypanel" ).trigger( "updatelayout" );
```

- Para abrir un panel

```
$( "#idofpanel" ).panel( "open" , optionsHash );
```

- Para cerrarlo

```
$( "#idofpanel" ).panel( "close" );
```

- Para cambiar el estado

```
$( "#idofpanel" ).panel( "toggle" );
```

Componentes

Panel

- El objeto panel, lanza eventos
 - panelbeforeclose
 - panelbeforeopen
 - panelclose
 - panelcreate
 - panelopen

Componentes

Formularios

- Se obliga para una correcta renderización, que todos los elementos de formulario vayan acompañados de su correspondiente etiqueta `<label>`.

```
<label for="username" >Username:</label>  
<input type="text" name="username" id="username" value=""/>
```

- Si se desea compactar la representación, se puede emplear el estilo **`class="ui-hidden-accessible"`**, para el **label** y definir el **placeholder** en el campo input.

```
<label for="username" class="ui-hidden-accessible">Username:</label>  
<input type="text" name="username" id="username" value=""  
placeholder="Username"/>
```

Componentes

Formularios

- Para deshabilitar los componentes, con el atributo estándar **disabled**.
- Todos los pares de **label** e **input**, deben ir rodeados de un elemento con el rol **data-role="fieldcontain"**.
- Para agrupar componentes, se emplea la etiqueta **fieldset**, sobre la que se aplica **data-role="controlgroup"**.

```
<fieldset data-role="controlgroup">  
  <input type="checkbox" name="one" id="one" />  
  <label for="one" >One</label>  
  <input type="checkbox" name="two" id="two" />  
  <label for="two" >Two</label>  
</fieldset>
```

Componentes

Formularios

- Para los **checkbox** y **radio**, se puede establecer la posición del icono con **data-iconpos**.

```
<fieldset data-role="controlgroup" data-iconpos="right">  
  <input type="checkbox" name="one" id="one" />  
  <label for="one" >One</label>  
  <input type="checkbox" name="two" id="two" />  
  <label for="two" >Two</label>  
</fieldset>
```

- También se puede establece el tema con **data-theme** sobre el **fieldset**.
- La etiqueta **Legend** dentro del **fieldset**, se toma como titulo del formulario.

Componentes

Formularios

- Para añadir de forma dinámica componentes a un **data-role="controlgroup"**, se ha de ejecutar el evento **create**, después de haber añadido los componentes.

```
<JS>
var count = 0;
$(document).ready(function() {
    $("#cargar").on("click", function(){
        $("#ctrlGroup").append(crearInput(count));
        $("#ctrlGroup").trigger('create');
        count++;
    });
});
<HTML>
<a id="cargar">cargar</a>
<fieldset data-role="controlgroup" id="ctrlGroup"></fieldset>
```