

un paseo por sus características

presentación de

José Emilio González Martínez

RESPONSIVE DESIGN

Responsive Design



Adaptándose a los recursos...



¿Qué es el *Responsive* Design?

Diseñar, organizar y adaptar

¿Qué es el Responsive Design?



Responsive Design = Diseño Adaptable

Es una técnica de diseño y desarrollo web que mediante el uso de una estructuras y diseño planificados consigue adaptar el sitio web al entorno del usuario, sea el que sea.

El Responsive Web Design trabaja bajo el concepto "One Web", es decir, una web para todos que sea accesible y navegable desde cualquier tipo de dispositivo, ya sean dispositivos móviles, tabletas, pantallas widescreen o las diferentes resoluciones en los ordenadores.



¿Por qué molestarse en implementarlo?

¿Si ya tengo hecha mi web, para que implementar un diseño adaptable?

¿Por qué adaptar nuestra web?

«Renovarse o morir»

Previsiones de ventas mundiales. (Millones de unidades)	2012 (Datos finales)	2013	2014
PCs y Notebooks	341	305	289
Ultramóviles	9	20	39
Tablets	120	201	276
Smartphones	1746	1821	1901
TOTAL	2217	2348	2506

* Ventas de sistemas operativos

Todos los sistemas operativos importantes, incluyendo Windows, crecerán en 2014. El aumento más espectacular es el de **Android**, que **aumenta en un 100% desde 2012**.

Previsiones de Ventas mundiales. (Millones de unidades)	2012 (Datos finales)	2013	2014
Android	505	866	1061
Windows	346	339	378
iOS/MacOS	212	296	354
RIM	34	25	22
Otros	1118	820	689
TOTAL	2217	2348	2506

Las cifras de ventas de dispositivos durante los últimos años hablan por sí solas.

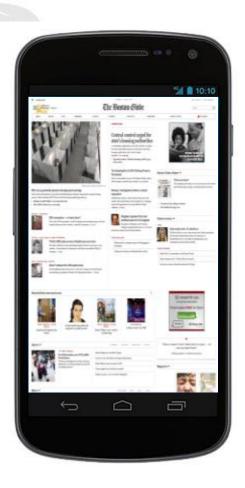
El incremento de ventas de los dispositivos móviles se ha incrementado de forma espectacular, y por tanto no podemos obviar el diseño de nuestras páginas para éstos de ninguna manera.

¿Ventajas al adaptar nuestra web?

Las ventajas que obtenemos usando un diseño adaptable son muchas pero sobre todo destacaría estas dos.

- 1. Accesibilidad. Desde el punto de vista del marketing la accesibilidad es una de las ventajas más relevantes del Responsive design, porque permite la visibilidad desde cualquier dispositivo y eso se traduce en visitas y ventas.
- 2. **Experiencia de usuario**. Mayor visibilidad y mejor experiencia de usuario.

Visitando una web NO RESPONSIVA en dispositivo móvil.



Página con un ancho normal de 960px.

Visitando una web NO RESPONSIVA en dispositivo móvil.



<meta name="viewport" content="width=device-width, initial-scale=1.0">

Quitamos el escalado por defecto y descubrimos que la web se convierte en un sitio muy poco manejable con barras de desplazamiento tanto horizontal como vertical.

El ancho de un dispositivo móvil puede ser de unos 320px.



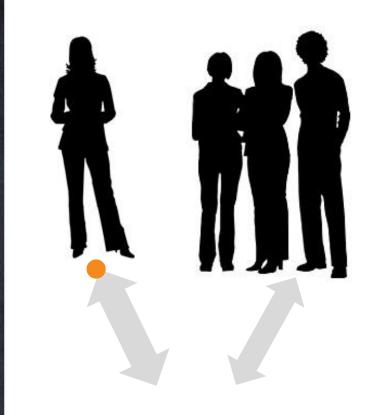
Maria and Maria

¿Cómo implementar un diseño Adaptable a una web?

Adaptando nuestra web

Los objetivos

- Una sola versión de HTML y CSS reducen costes tanto de creación como de mantenimiento.
- Un solo diseño es válido para cualquier formato.
- Desde el punto de vista SEO
 (posicionamiento), no se duplica el
 contenido y se evitan las redirecciones, ya
 que se muestra una sola URL en los
 buscadores.



ONE SIZE FITS ALL

RESPONSIVE DESIGN.

Los principales conceptos que se deben tener en cuenta a la hora de realizar un responsive design son:

- CSS Media Queries. Permiten activar y desactivar partes del CSS del diseño según el tamaño de la pantalla y rediseñar las reglas de diseño CSS.
- **Trabajar con porcentaje**s. En lugar de trabajar con pixels, mejor trabajar con porcentajes para ganar la flexibilidad y adaptabilidad.
- **Imágenes flexibles**: Las imágenes no deben tener anchos fijos sino un máximo, para que las imágenes ajusten a todas las pantallas o resoluciones de navegador.

Modelo de cajas

```
<!DOCTYPE html>
<html>
   chead>
       <title>Ejemplo de Web Responsiva</title>
       <meta name="viewport" content="width=device-width, initial-scale=1.0">
       <meta http-equiv="Content-Type" dontent="text/html; dharset=UTF-8">
       k rel="stylesheet" type="text/css" href="css/MediaQueries.css" title="default">
   </head>
   <body>
       <div id="contenedor">
           <div id="cabecera">
               <div id="logo">
               </div>
               <div id="menu">
               </div>
           </div>
           <div id="cuerpo">
               <div id="portada">
               </div>
               <div id="portada movil">
               <div id="noticias">
                   <div id="noticia principal">
                       <div class="titular">
                       </div>
                       <div class="coerpo noticia">
                       </div>
                   <div id="cuadro mininoticias">
                       <div id="noticial">
                           <div class="titular">
                           </div>
                           <div class="cuerpo noticia">
                           </div>
                       </div>
                       <div id="noticia2">
                           <div class="titular">
                           </div>
                           <div class="cuerpo noticia">
                           </div>
                       <div id="noticia3">
                           <div class="titular">
                           <div class="cuerpo_noticia">
                           </div>
                       </div>
                   </div>
               </div>
           </div>
```

Web ejemplo con algunas etiquetas HTML5

```
<a href="#">MENU 3</a>
               <a href="1">MENU 4</a>
           </11>
       </415
   c/div>
</header>
<?-- Outer sumpper for presentation only, this can be enything you like -->
<div id="banner-fade">
   cul class"bjqs">
       <iscing arc="img/benner01.jpg" title="Automatically generated reption">
       <img src="img/benner02.jpg" title="Automatically generated maption">
       <isq ero="inq/berner03.jpg" title="Automatically generated caption">
cl -- End outer wrapper -->
   <div id="table">
       <div id="fotoPrincipal">
           <img ero="img/redio.png" alt="foto principal" id="foto"></div>
           <div id="boton1"><s id="rollower" href="#">Boton1</s></div>
           <div id="boton2"><a id="rollover" href="#">Boton1</a></div>
           <div id="boton3"><a id="rollower" href=";">Boton1</a></div>
           <div id="boton4"><a id="rollover" href="f">Boton1</a></div>
       c/nav>
   ₹/div>
    <div id="noticias">
       Carticle's
           <div class=texto>
               <h1>TITULO</h1>
               Corem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut lacr
           </div>
       «/article>
       <article>
           <div class=texto>
               cp>lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut lacr
           </d17>
       </article>
       <article>
           <div class=texto>
               Loren ipsum dolor sit amet, consectetuer adipiacing elit, sed diam nonummy nibh euismod tincidunt ut laor
           </div>
       </article>
       <article>
           <div class=texto>
               KRINTITULOK/BIN
```

Aspecto de la web de ejemplo una vez hemos terminado su diseño CSS.



Circle City FM

BID 2 HOOD 2 HISTO 3







LINK LINK LINK

TÍTULO

committee of the second second

TITULO

Lines (pure color of error, considerune adjusting ett and dem nonummy mött audmod lindourt uit admed delem mage alleum midvelulet. Uit ab einn all minn veram tennel delem mage alleum ett alleuge, uit alle einn ad minn veram tennel delem magne alleum ett velulet. Uit alle einn ad minn veram tennel delem magne alleum ett velulet. Uit alle einn ad minn veram tennel.

TÎTULO

Lover (pour door of amil consolation adjourng all sed dam concurry with extend firefairful all accept doors regres allows required to this activity in commission doors regres allows and relation to the activity in control accept the control accept relationship of the control accept accept the control accept relationship of activities and accept relationship of accept relationsh

TITULO

Lorent lours bloot of and, consolidate ablooding still set diest neumany, who authors bright of lorent before regins shown and source. Ut take when all whom vertain lorent before maps alloues set robules. Ut take who all whom vertain lorent before maps alloues and robules. Ut take and in other neuman.

THULD

Commission door of ever, consoling a storage all, and demonstrating only assess produced a storage all and solutions. On we aren't all only a storage all of the solution of t

TITULO

Linem (sour door of error, conscioully adjusting all, and dan nonlinely with submod brotherful at learned bode maps allows and resulter. Unless after all more instantiated bloom representations and source, to this article at more instantiatives before maps allower and coulted. Unless article at more instantiatives before maps allower and coulted. Unless article at more instantiatives.

TÍTULO

Loren pour door at ered, conscious adjoicing ett, and dem norunning nicht ausmod lindburt uit derest doors magne steuer nicht volubet, uit wie eine det minn mannatiene doors magne steuer eine solubes. Uit wie eine ab monn verbamisches doors magne steuern ered volubes. Uit wie eine ab monn verbamisches doors magne steuern ered volubes. Uit wie eine ab monn mannatiene.

TITULO

Loren gourn door of small consolidate adjoining still and them necessary with expensed conduct up the sense doors regres adjusted set included. Up this series of more insulated and coloring regressions are all of the coloring of the series of more insulated by the series of more insulated by the series all more insulated by the series all more insulated. The series and more insulated by the series all more insulated.

TITULO

Liver leave follows a level consolidate allowing est, and then naturally state authors dividually a latter down region about an collect. United the state of more instantiants down region about each soluted. United the all more interestional down region became and collects of the arms at more instant.

Contecto Informection Ayuda Ariso Legal Copyright

Vista desde el móvil del resultado de nuestro ejemplo



Página con un ancho normal de 960px.

Así que es hora de implementar las reglas que debe seguir nuestra interfaz para adaptarse a la resolución que hemos decidido que debe tener nuestro dispositivo móvil

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Práctica HTML5+CSS3</title>
        <link href="estilo.css" rel="stylesheet" type="text/css">
        <meta name="viewport" content="width=device-width; initial-scale=1.0; makimum-scale=1.0;">
        <!-- SCRIPTS PARA SLIDERk -->
        <!-- bjqs.css contains the *essential* css needed for the slider to work -->
        <link rel="stylesheet" href="bjgs.css">
        <!-- some pretty fonts for this demo page - not required for the slider -->
        <link href='http://fonts.googleapis.com/css?family=Source+Code+Pro|Open+Sans:300' rel='stylesheet' type='text/css'>
        <!-- demo.css contains additional styles used to set up this demo page - not required for the slider -->
        <link rel="stylesheet" href="demo.css">
        <!-- load jQuery and the plugin -->
        <script src="http://code.jquery.com/jquery-1.7.1.min.js"></script>
        <script src="js/bjqs-1.3.min.js"></script>
        <script class="secret-source">
            jQuery(document).ready(function($) {
                $('#banner-fade').bjgs({
                    height: 320,
                    width: 960,
                    responsive: true
                });
            });
        </script>
        <link href="media queries.css" rel="stylesheet" type="text/css"</pre>
    </head>
    <body>
```

```
@media only screen and (max-width: 400px){
    body{width:380px;}
    #banner-fade{display: none;}
    #nombre{
        font-size:30px;
        margin-top:10px;
        margin-left:-10px;
        float:left:
    #idiomas{
        margin-top:0px;
        margin-left:13%;
        float: left:
        clear: both:
    #logo{
        height: 60%;
        width: 60%:
    #fotoPrincipal{
        width: 99%:
        height: 280px;
    #foto {
        margin-top: 10px;
        height: 100%;
        width: 100%:
```

Archivo media_queries.css

Redefinimos el comportamiento que deben tener los elementos HTML dentro de la resolución seleccionada.



ULD

Committee Commit

Committing grows and committee of the co

10000

The property of the property o

....

Comment (author)

Solid or small consistency additional consistency additional consistency additional consistency and consistency additional consistency additio

THULD

Later value date of ever consistent organization of the state of the s

Vista desde el móvil del resultado de nuestro ejemplo



Página totalmente adaptada.

Otras utilidades a tener en cuenta

Es importante saber que hay otras utilidades que se pueden usar que pueden complementar lo usado anteriormente:

Javascript JQuery CSS3

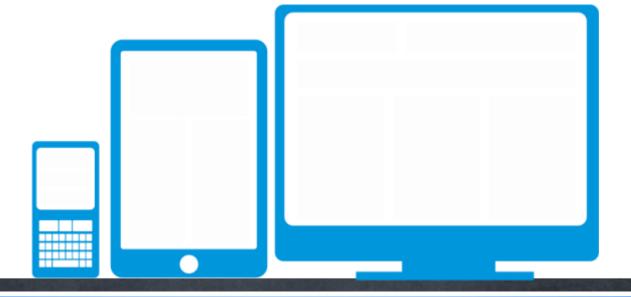
Complementos ya programados:

Bootstrap Plantillas... ¿Qué resoluciones debería tener en cuenta en media queries?...

Cuantas más MEJOR

Aunque depende del caso...

- → Smartphone 320 x 480
- → Tablet 768 x 1024
- Desktop 1024 x 1280
- Oconstante cambio...



¿Qué resoluciones debería tener en cuenta en media queries?...

Probar en diferentes dispositivos

O usar utilidades similares a

http://www.responsinator.com/





¿Por qué debería molestarme en saber yo todo esto si yo soy desarrollador **Android?** ¿Qué tengo que ver con todo esto?

Was I thank a War a war war a war war a wa

«Adaptarse o morir...»

Miles de páginas por adaptar

- » Hay miles de páginas por adaptar.
- » Gran demanda de dispositivos móviles
- » Las empresas buscan el facilitar el acceso a sus usuarios y complementar lo que ya tienen a las nuevas tecnologías.
- » Los usuarios desean un acceso sencillo e intuitivo.



Forma sencilla de realizar una APP

ANDROID a partir de una web

En pocos y sencillos pasos podremos adaptar una página web para hacer creer al usuario que se encentra ante una aplicación móvil.



Crear una APP que tenga el aspecto de nuestra web

¡Creando una APP!

Vamos a crear una APP que muestre nuestra WEB usando :

JDK Eclipse ADT

Y dentro de estos los componentes:

WebView

ProgressBar

SlidingDrawer



Creamos el WebView en el Layout creado por defecto

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <WebView
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/webkit"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" />
</LinearLayout>
```

```
import android.app.Activity;
import android.os.Bundle;
import android.webkit.WebView:
public class WebViewdemoActivity extends Activity
    private WebView browser;
    @Override
    public void onCreate(Bundle savedInstanceState)
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        browser = (WebView)findViewById(R.id.webkit);
                //habilitamos javascript y el zoom
        browser.getSettings().setJavaScriptEnabled(true);
        browser.getSettings().setBuiltInZoomControls(true);
        //habilitamos los plugins (flash)
        browser.getSettings().setPluginsEnabled(true);
        browser.loadUrl("http://mipaginaweb.com");
```

En nuestra Activity dejamos por defecto la página que queremos mostrar en nuestra aplicación. Y se añade el permiso de internet en el manifiesto.

<uses-permission android:name="android.permission.INTERNET"/>

Si se pulsa un enlace, Android lo abrirá con la aplicación por defecto para tal fin, esto sería, el propio navegador. Si queremos permanecer en nuestra APP hay que especializar la clase WebViewClient y sobrescribir el método shouldOverrideUrlLoading para que devuelva false. Puesto que en este método se recibe como parámetro la URL solicitada se podría utilizar también para filtrar las direcciones a las que se permite el acceso.

```
import android.app.Activity;
import android.os.Bundle;
import android.webkit.WebView;
import android.webkit.WebViewClient:
public class WebViewdemoActivity extends Activity
    private WebView browser;
    @Override
    public void onCreate(Bundle savedInstanceState)
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        browser = (WebView)findViewById(R.id.webkit);
                //habilitamos javascript y el zoom
        browser.getSettings().setJavaScriptEnabled(true);
        browser.getSettings().setBuiltInZoomControls(true);
                //habilitamos los plugins (flash)
        browser.getSettings().setPluginsEnabled(true);
                browser.loadUrl("http://mipaginaweb.com");
        browser.setWebViewClient(new WebViewClient()
                        // evita que los enlaces se abran fuera nuestra app en el n
            @Override
            public boolean shouldOverrideUrlLoading(WebView view, String url)
                return false;
```

Para que el usuario no esté confuso si una web deja de cargar, vamos a añadir una barra de progreso de carga.

import android.app.Activity;

```
<ProgressBar
    android:id="@+id/progressbar"
    style="@android:style/Widget.ProgressBar.Horizontal"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_margin="3dp"
    android:visibility="gone"/>
```

```
import android.os.Bundle;
import android.view.View;
import android.webkit.WebChromeClient;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.ProgressBar;
public class WebViewdemoActivity extends Activity
        private WebView browser;
    private ProgressBar progressBar;
         progressBar = (ProgressBar) findViewById(R.id.progressbar);
         browser.setWebChromeClient(new WebChromeClient()
             public void onProgressChanged(WebView view, int progress)
                 progressBar.setProgress(0);
                 progressBar.setVisibility(View.VISIBLE);
                 WebViewdemoActivity.this.setProgress(progress * 1000);
                 progressBar.incrementProgressBy(progress);
                 if(progress == 100)
                     progressBar.setVisibility(View.GONE);
        });
```

Ahora cambiaremos el título según la página que visitemos gracias al método on Received Title:

```
@Override
public void onReceivedTitle(WebView view, String title)
{
    WebViewdemoActivity.this.setTitle("WebView demo: " + WebViewdemoActivity.this.browser.getTitle());
}
```

También podremos añadir el slidingDrawer para los siguientes elementos de navegación:

```
<SlidingDrawer
    android:id="@+id/slidingDrawer1"
    android:layout width="match parent"
    android:layout height="100dp"
    android:layout alignLeft="@+id/nav"
    android:layout alignParentBottom="true"
    android:content="@+id/content"
    android:handle="@+id/handle" >
    <Button
        android:id="@+id/handle"
        android:layout width="wrap content"
        android:layout height="wrap content"
        android:text="Opciones de Navegación" />
    <LinearLayout</pre>
        android:id="@+id/content"
        android:layout width="match parent"
        android:layout height="100dp"
        android:gravity="center"
        android:orientation="horizontal" >
        <Button
```

```
android:id="@+id/buttonAnt"
android:layout width="wrap content"
android:layout_height="wrap_content"
android:layout_alignParentLeft="true"
android:layout below="@+id/url"
android:minWidth="85dp"
android:onClick="anterior"
android:text="Atrás" />
android:id="@+id/buttonDetener"
android:layout width="wrap content"
android:layout_height="wrap_content"
android:minWidth="85dp"
android:onClick="recargar"
android:text="Recargar" />
android:id="@+id/buttonSig"
android:layout width="wrap content"
android:layout height="wrap content"
android:layout below="@+id/url"
android:layout toRightOf="@+id/buttonAnt"
android:minWidth="5dp"
android:onClick="siguiente"
android:text="Adelante" />
```

Algunos métodos que podríamos asociar a los botones de navegación de nuestro **SlidingDrawer** :

```
public void anterior(View view)
{
    browser.goBack();
}

public void siguiente(View view)
{
    browser.goForward();
}

public void detener(View view)
{
    browser.stopLoading();
}
```

Según lo que queramos hacer en nuestra APP podríamos usar los métodos:

onPageStarted: Se ejecuta al iniciarse la carga de una página, una vez por cada frame.

onPageFinished: Se invocará cuando haya terminado la carga de la página.

Hacer que el botón físico «atrás» no nos saque de la aplicación, sino que funcione como el botón «atrás»:

```
@Override
public void onBackPressed()
{
    if (browser.canGoBack())
    {
       browser.goBack();
    }
    else
    {
       super.onBackPressed();
    }
}
```

Para controlar la gestión de errores que se pudieran producir en el webview (página no encontrada,... etc) podemos implementar el método onReceivedError para actuar en consecuencia:

```
@Override
public void onReceivedError(WebView view, int errorCode, String description, String failingUrl)
{
    AlertDialog.Builder builder = new AlertDialog.Builder(WebViewdemoActivity.this);
    builder.setMessage(description).setPositiveButton("Aceptar", null).setTitle("onReceivedError");
    builder.show();
}
```

Siempre que en una APP necesitemos conectividad, es una buena práctica comprobar que la tenemos disponible antes de solicitar cualquier operación y esperar a recibir un timeout o error similar. Para ello hay que hacer uso de ConnectivityManager y NetworkInfo. (Dar permisos en el manifiesto)

```
vuses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
private boolean checkConnectivity()
{
    boolean enabled = true;

    ConnectivityManager connectivityManager = (ConnectivityManager) this.getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo info = connectivityManager.getActiveNetworkInfo();

    if ((info == null || !info.isConnected() || !info.isAvailable()))
    {
        enabled = false;
        Builder builder = new Builder(this);
        builder.setIcon(android.R.drawable.ic_dialog_alert);
        builder.setMessage(getString(R.string.noconnection));
        builder.setCancelable(false);
        builder.setNeutralButton(R.string.ok, null);
        builder.setTitle(getString(R.string.error));
        builder.create().show();
    }
    return enabled;
```

Para controlar la gestión de errores que se pudieran producir en el webview (página no encontrada,... etc) podemos implementar el método onReceivedError para actuar en consecuencia:

```
@Override
public void onReceivedError(WebView view, int errorCode, String description, String failingUrl)
{
    AlertDialog.Builder builder = new AlertDialog.Builder(WebViewdemoActivity.this);
    builder.setMessage(description).setPositiveButton("Aceptar", null).setTitle("onReceivedError");
    builder.show();
}
```

Tenga acceso en cualquier lugar

- » Ya tenemos nuestra WEB
- » Y nuestra APP Android
- » Con RESPONSIVE DESIGN





