



Responsive web design

a cura di Fabio Pittarello, Università Ca' Foscari Venezia

Perchè abbiamo bisogno di siti responsive?

- ◆ I siti responsive rispondono all'esigenza di avere una presentazione flessibile in corrispondenza della tipologia di dispositivo utilizzata dall'utente, oggi molto più variabile di un tempo (desktop/laptop, smartphone, tablet, phablet, ecc.)
- ◆ Queste nuove esigenze hanno portato all'evoluzione delle specifiche dei linguaggi per il web e dei dispositivi in grado di interpretarle, dando vita al cosiddetto responsive web design
- ◆ La mancanza di un sito responsive diventa un problema non solo per l'accesso diretto a un sito, ma anche quando la mediazione avviene attraverso l'uso di una rete sociale (Facebook, Twitter, ecc.)

Accedere ad un sito attraverso Twitter

- ◆ Un numero significativo di utenti utilizza i social network su dispositivi mobili
- ◆ Ad esempio Twitter fornisce un'interfaccia adatta all'uso con dispositivi di piccole dimensioni



Accedere ad un sito attraverso Twitter

- ◆ Tuttavia non tutti i messaggi di Twitter sono autocontenuti



Accedere ad un sito attraverso Twitter

- ◆ Spesso i messaggi di Twitter contengono link a siti esterni progettati per l'uso con dispositivi desktop
- ◆ Anche se le interfacce degli smartphone ci permettono di visualizzare il testo zoomando, non si può certo dire che questo sia il formato ideale di presentazione



Accedere ad un sito attraverso Twitter

- ◆ Un altro esempio con i tweet dell'Ansa ...



Accedere ad un sito attraverso Twitter

- ◆ ... e la pagina collegata sul sito dell'ANSA, in formato di presentazione per desktop

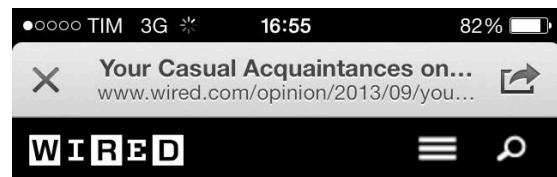


Accedere ad un sito attraverso Twitter

- ◆ Un diverso approccio per i tweets di Wired ...



Accedere ad un sito attraverso Twitter



- ◆ ... e per i collegamenti al sito di Wired, in versione mobile.



Perchè abbiamo bisogno di siti responsive?

- ◆ Le tecniche per ottenere questo risultato possono essere diverse: è possibile ad esempio attraverso l'uso di **tecniche di riconoscimento del browser** indirizzare l'utente verso una versione specifica per il tipo di dispositivo **oppure** è possibile (come succede nel responsive web design) fornire una **versione unica per i contenuti e forme di presentazione differenziate** che vengono **automaticamente selezionate dal browser** a seconda delle caratteristiche del browser stesso o del dispositivo

Come viene utilizzato il responsive web design?

- ◆ Le tecniche di responsive web design si possono utilizzare attraverso l'uso diretto di HTML, CSS e Javascript
- ◆ Un metodo alternativo consiste nell'utilizzare framework che semplificano l'uso dei linguaggi sopra menzionati e che incorporano tecniche di responsive web design
- ◆ Anche i moderni CMS (sistemi per la gestione dei contenuti) permettono di presentare i contenuti in modo appropriato per i diversi dispositivi attraverso l'uso di temi specifici che utilizzano tecniche di responsive web design

The screenshot shows the homepage of Less Framework 4. At the top, it says "Less Framework 4" and "An adaptive CSS grid system." Below this are three navigation buttons: "What It Is", "How It Works", and "Getting Started". The main visual is a collection of devices (a desktop monitor, two tablets, and two smartphones) all displaying the same blue and white striped grid pattern, demonstrating how the framework adapts to different screen sizes. Below this, there's a section titled "What It Is" with a paragraph of text. To the right of this text is a yellow sticky-note-like box containing a note: "Less Framework is a pretty old idea at this point. Check out its successor, Frameless, instead." At the bottom, there are two small diagrams labeled "Default Layout" and "Tablet Layout", each showing a horizontal grid with colored bars (blue, grey, yellow) representing the grid structure.

What It Is

Less Framework is a CSS grid system for designing adaptive websites. It contains 4 layouts and 3 sets of typography presets, all based on a single grid.

Less Framework is a pretty old idea at this point. Check out its successor, [Frameless](#), instead.

Default Layout Tablet Layout

Less, un framework responsive



Introducing Bootstrap.

Need reasons to love Bootstrap? Look no further.

Bootstrap, un framework responsive



Going Responsive

Responsive web design (RWD) is a design and technical approach that aims to adapt the layout and interaction of a site or app to work optimally across a wide range of device resolutions, screen densities and interaction modes with the same underlying codebase. The framework has a number of responsive widgets: [responsive grids](#), [reflow tables](#) and [column chooser tables](#), and [sliding panels](#).

[Jump to section](#) 

RWD Basics

RWD has three key elements:

- **CSS media queries**, used to target styles to specific device characteristics such as screen width/breakpoint or resolution.
- **A fluid grid**, that specifies elements and widgets in flexible units with the goal of making them flow to fill their containers.
- **Flexible images and media**, are also sized in relative units so they re-size to fit within their containers.

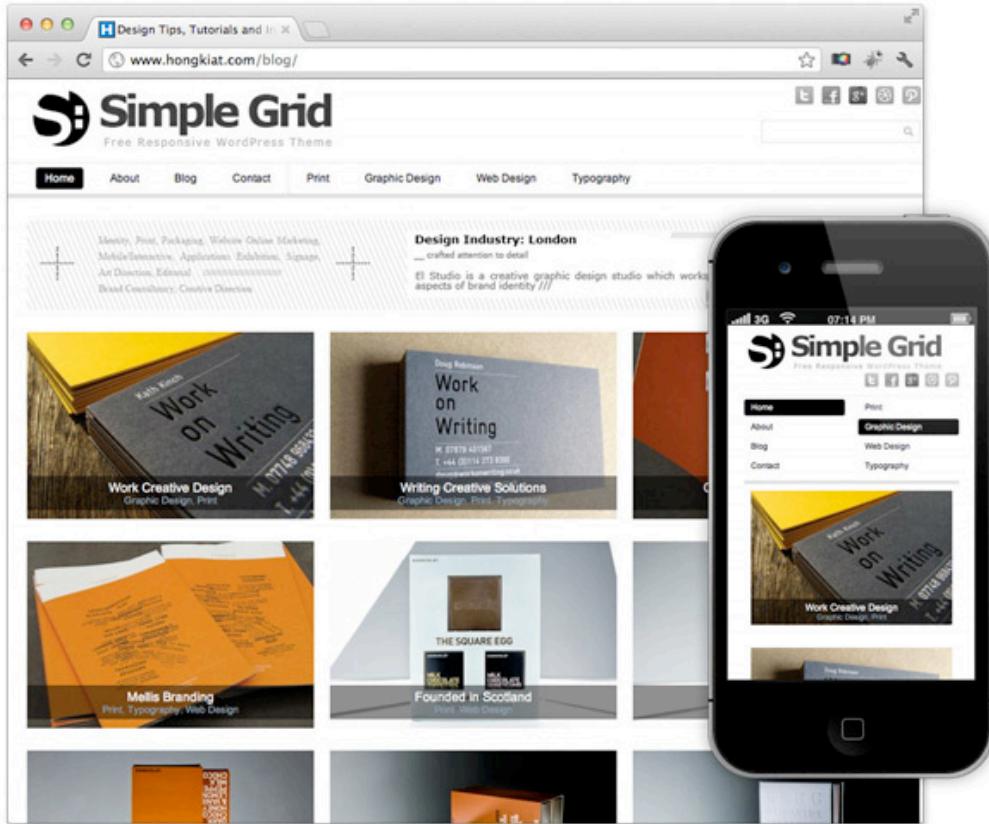
By creating all screen elements to be fluid and flexible, it allows the media queries to focus primarily on controlling layout rules for containers; the modules inside simply re-size to fit their containers.

A simple responsive example may be two stacked containers, each with flexible content or widgets inside. At greater widths, media queries are used to float both containers to create a two column layout to take better advantage of the wider screen.

Since the content inside each container is designed to re-flow to fit its parent, the media queries can focus just on the rules for making the columns stack or float, and to override or add styles only needed at greater widths.

[Return to top](#) 

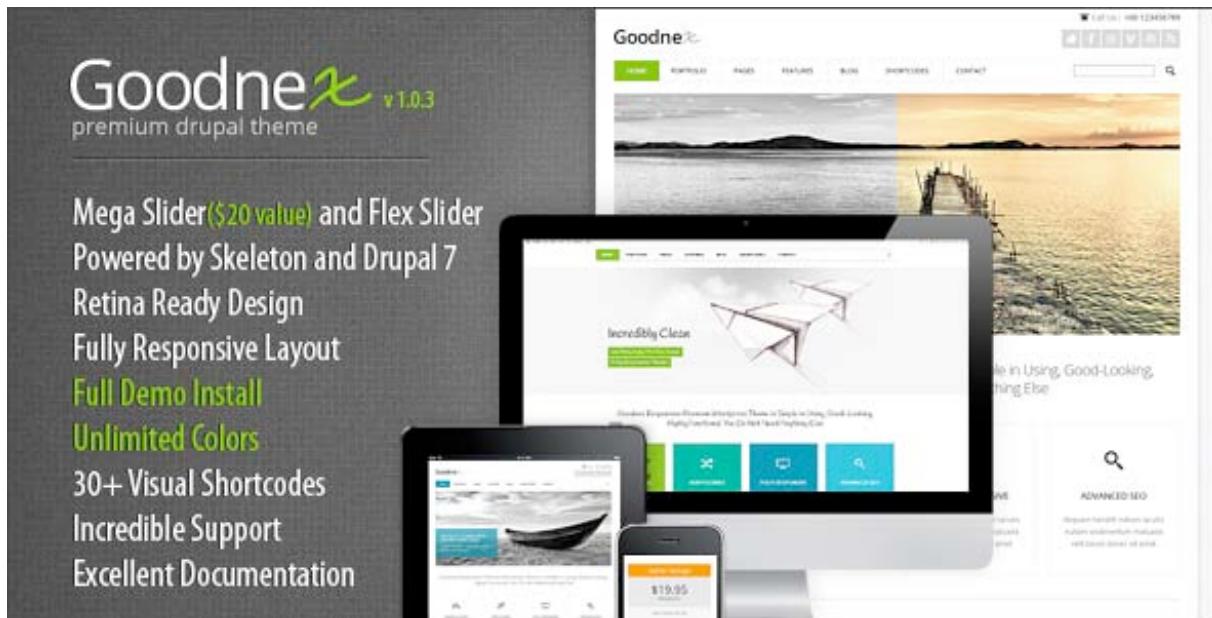
jQuery Mobile, un framework responsive



Un tema responsive per il CMS Wordpress

The screenshot shows the JA Elastica Joomla 2.5 template. The desktop version has a red header bar with the site name and a main content area featuring a large image of a computer monitor displaying the site's design. A sidebar on the right contains a Twitter feed, a section for 'RESPONSIVE WEB DESIGN', and a 'MAIN MENU' with links to Home, Explore, and About Joomla! 2.5. The mobile view on the right shows the site's responsive design on a smartphone, maintaining the red header and the main content grid.

Un tema responsive per il CMS Joomla



Un tema responsive per il CMS Drupal



Romney contro Obama



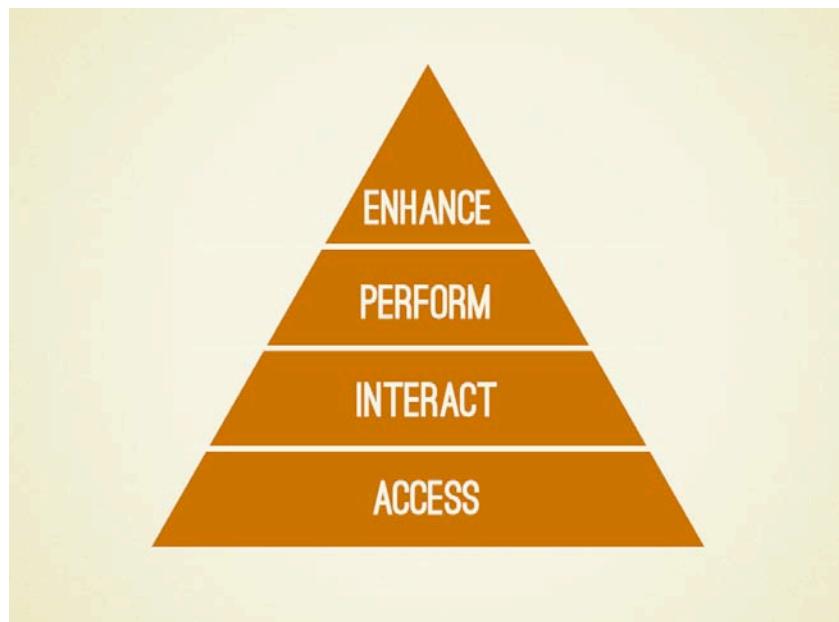
analisi dei siti web dei due candidati,
in versione mobile e desktop
un'occasione per verificare i benefici di un approccio
responsive, ma anche per identificarne le carenze



Nel 2012 più del 50% degli americani possiede uno smartphone.
Per il 28% degli americani il dispositivo mobile costituisce la modalità
primaria di accedere al web
Gli utenti americani arrivano al web mobile attraverso una molteplicità
di canali, in costante aumento (es. social apps, QR.codes, SMS)

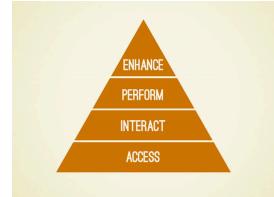
Che cosa vogliono i visitatori dei siti dei candidati alle elezioni?

- ◆ I visitatori cercano informazioni sui candidati oppure cercano strumenti per intraprendere azioni a loro supporto.
- ◆ **IL VISITATORE ORIENTATO ALL'INFORMAZIONE**
- ◆ Può cercare informazioni di base, come la biografia del candidato o le opinioni a proposito di temi diversi. Può essere interessato anche ad essere aggiornato sulle news, post del blog o smentite del candidato su affermazioni che lo discreditano (myth busting). Il sito ufficiale da al candidato la possibilità di fornire informazione senza alterazioni dovute alla mediazione dei giornalisti o alle inaccuracy del crowdsourcing.
- ◆ **IL VISITATORE ORIENTATO ALL'AZIONE**
- ◆ Cerca di supportare il candidato in modi diversi, donando denaro o tempo per fare propaganda e contribuendo agli eventi a supporto del candidato.



Kristofer Layon ha definito attraverso la metafora di una piramide gli aspetti basiliari di un'esperienza per gli utenti del web mobile, che comprende alla base l'accesso all'informazione e alla sommità gli aspetti più evoluti dell'esperienza, permessi dall'utilizzo di tecniche legate ad HTML5 (es. offline storage).

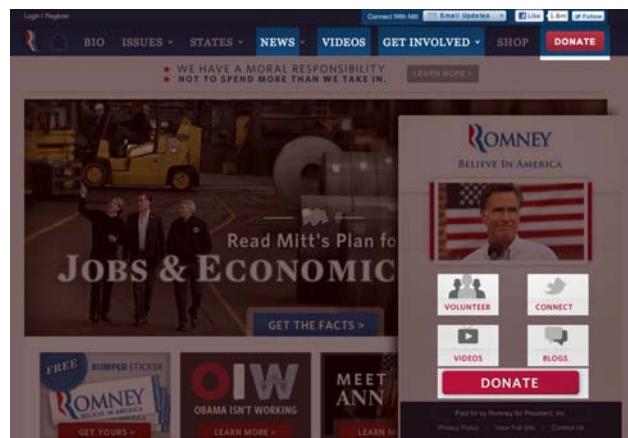
Accesso all'informazione



- ◆ Prima di tutto gli utenti necessitano di avere **accesso** all'esperienza. Storicamente il web mobile è stato visto come una specie di web lite, in cui gli utenti possono accedere solo ad un subset di contenuti e funzionalità.
- ◆ Nonostante questo il 28% dei residenti negli USA (dati 2012) utilizzano il dispositivo mobile come mezzo primario per accedere al web. Per questo motivo fornire agli utenti gli stessi contenuti (**content parity**) e garantire loro accesso ad un'esperienza completa è più importante che mai.
- ◆ Nelle slides seguenti vedremo come le soluzioni per i siti di supporto ai candidati delle Presidenziali americane del 2012 hanno risposto a questa esigenza.

Accesso: Romney

- ◆ Il sito di Mitt Romney's website utilizza il riconoscimento del dispositivo utilizzato (**device detection**) per indirizzare gli utenti con un dispositivo mobile ad un sito web separato.
- ◆ Questa redirezione permette di creare un'**esperienza diversificata**, rivolta in modo specifico verso il web mobile.
- ◆ Tuttavia il problema principale del sito mobile di Mitt Romney è che comprende **solo una frazione delle features include nel sito** in versione desktop.



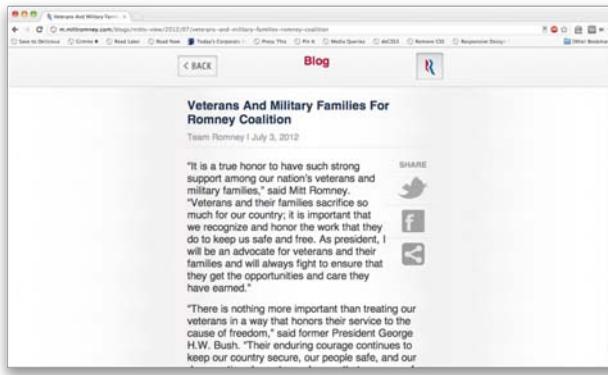
Accesso: Obama

- ◆ Il sito mobile di Barack Obama condivide lo stesso codice HTML del sito desktop. Questo **da la possibilità** di raggiungere l'obiettivo di **content parity**, lasciando comunque la possibilità al designer di decidere se tutti i contenuti devono essere visualizzati oppure se una parte di essi possono essere trascurati nell'esperienza mobile.
- ◆ Resta aperto il problema che comunque anche l'utente mobile deve scaricare tutti i contenuti, che siano visualizzati o no. Esistono comunque tecniche diverse che permettono di dare una risposta a questo problema

Accesso, considerazioni

- ◆ **E' un falso mito che gli utenti del web mobile non vogliano accedere a tutta l'informazione e funzionalità degli utenti desktop.** Ad esempio l'assenza di contenuti chiave nel sito di Romney lascia i potenziali elettori privi di risposte su alcune domande importanti, come "Quali sono i piani di Romney per l'economia?" oppure "Che cosa può fare Romney per il mio Stato?"

Accesso, considerazioni

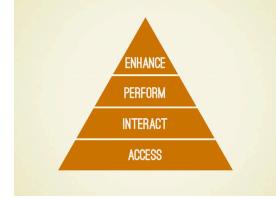


- ◆ Un altro problema è legato al cosiddetto **URL management**. Il riconoscimento del dispositivo è essenziale, dal momento che le versioni desktop e mobile del sito stanno su URL diversi. Sfortunatamente molti siti non applicano la redirezione dell'URL a tutti i livelli della gerarchia informativa e questo appare evidente quando un contenuto di una pagina esterna in versione mobile viene condiviso da una rete sociale e viene visualizzato anche dagli utenti desktop (vedi immagine) oppure viceversa.

Accesso, considerazioni

- ◆ Avere documenti HTML unici e specifiche di presentazione appropriate per tutte le situazioni di presentazione rende più semplice fornire ai visitatori il materiale informativo che stanno cercando nella modalità di presentazione appropriata.

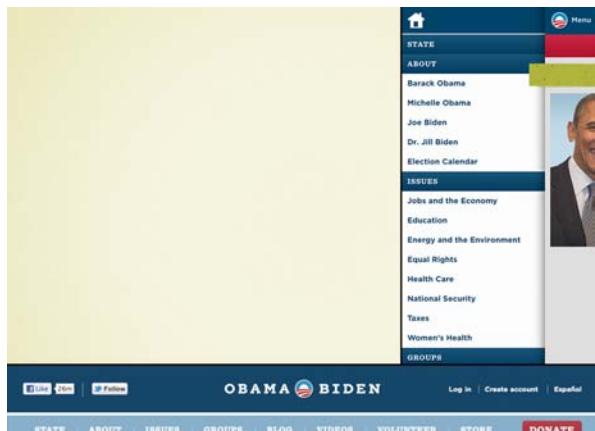
Interazione



- ◆ L'accesso ad un sito web è essenziale, ma poi il visitatore necessita di interagire con il contenuto e di spostarsi attraverso l'informazione. Consideriamo quindi uno dei più importanti meccanismi di interazione, la **navigazione**.
- ◆ NAVIGAZIONE: gli **strumenti** per la navigazione devono essere equilibrati, nel senso che **non devono essere eccessivi ma nemmeno scarsi** e difficili da raggiungere.

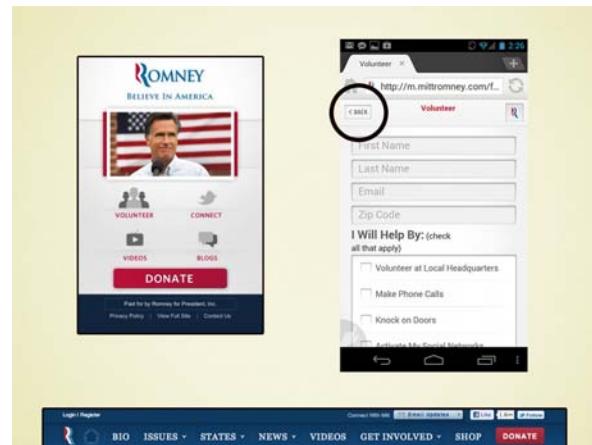
Interazione: Obama

- ◆ Il sito di Obama nella versione mobile è caratterizzato da un **menu fly-out** (o **drawer**) simile a quello in uso in Facebook. Richiede un'implementazione abbastanza complessa da un punto di vista tecnico e presenta problemi di funzionamento per molti dispositivi mobili, come rimarca Stephanie Rieger points in un suo post "A Plea for Progressive Enhancement", arrivando a non aprirsi proprio in alcuni casi.
- ◆ E' un buon esempio delle sfide che emergono nella creazione di esperienze che si adattano alle diverse classi di dispositivi.



Interazione: Romney

- ◆ La soluzione per la navigazione del sito mobile di Romney è relativamente semplice rispetto a quella del sito di Obama. Si presenta come una **dashboard** sulla home page, e tutte le pagine interne includono semplicemente un pulsante "Back" ed un logo che rimanda alla home page.
- ◆ Questa soluzione di navigazione evita la complessità, ma crea altri problemi. Il logo e il pulsante "Back" non si presentano con evidenza e non invitano ad essere cliccati. Oltre a questo il pulsante "Back" non funziona se si arriva alla pagina corrente da un altro sito; in sostanza il pulsante duplica le funzionalità del pulsante "Back" presente di default nei browser anzichè permettere di risalire la gerarchia informativa.



Interazione - scrolling

- ◆ Le pagine del sito mobile di Romney hanno una dimensione accettabile e l'utente non deve fare molto scrolling prima di trovare il contenuto.
- ◆ Le pagine del sito mobile di Barack Obama comprendono una grande quantità di contenuti, introducendo spesso sezioni nuove molto avanti nel flusso dei contenuti.
- ◆ Effettuare uno **scrolling tra molte sezioni diversificate di contenuti non corrisponde ad una buona esperienza di interazione**, perchè in questa situazione gli utenti possono anche non rendersi conto che determinati contenuti sono presenti. Al contrario è accettabile presentare un numero limitato di tipi di contenuto, come un articolo di blog ed una sidebar collegata.

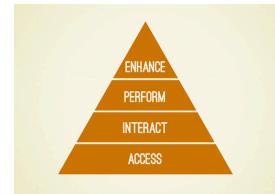


Interazione - scrolling

- ◆ Per rimediare all'eccesso di informazioni da caricare simultaneamente, possono essere usate tecniche come il caricamento condizionale (**conditional loading**) per fornire accesso a tutto il contenuto del sito senza che questo venga fornito tutto allo stesso momento.
- ◆ In aggiunta alla **sgradevolezza** di effettuare uno **scrolling attraverso una grande quantità di contenuto**, le **pagine estremamente lunghe sono caratterizzate anche da una terribile performance**.

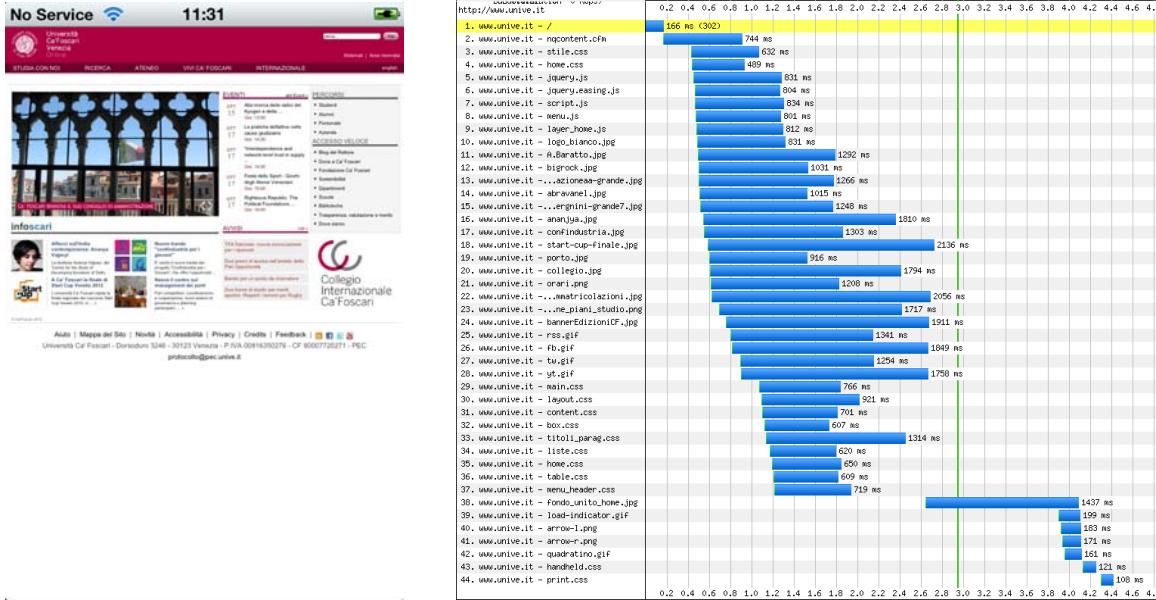


Performance



- ◆ Spesso i creatori di siti web fanno assunzioni esagerate rispetto al contesto d'uso dell'utente: ipotizzano connessioni estremamente veloci oppure macchine molto potenti. Il risultato è che le pagine web pesano mediamente 1 MB.
- ◆ 1 MB può non sembrare molto, ma da alcune indagini fatte risulta che il 71% degli utenti in mobilità si attende che i siti mobili si carichino con una velocità almeno uguale quella dei siti desktop e il 74% dei visitatori di siti mobili abbandonano un sito web se le pagine impiegano più di 5 secondi a caricarsi.
- ◆ Per valutare le performance di caricamento delle pagine web, è stato utilizzato il test fornito da Akamai (mobitest.akamai.com), che è accessibile da web e che si avvale di dispositivi reali per catturare i dati.

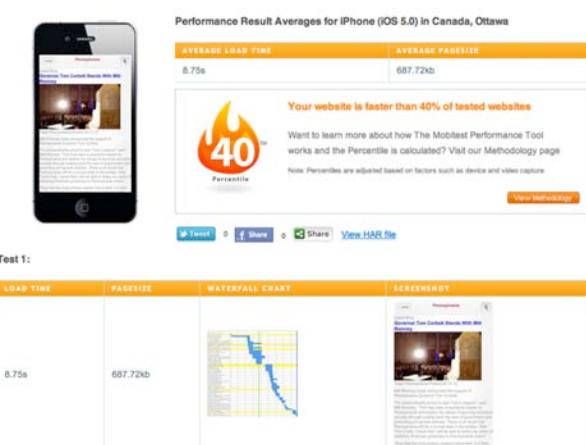
Mobile performance test



◆ <http://mobitest.akamai.com>

Performance: Romney

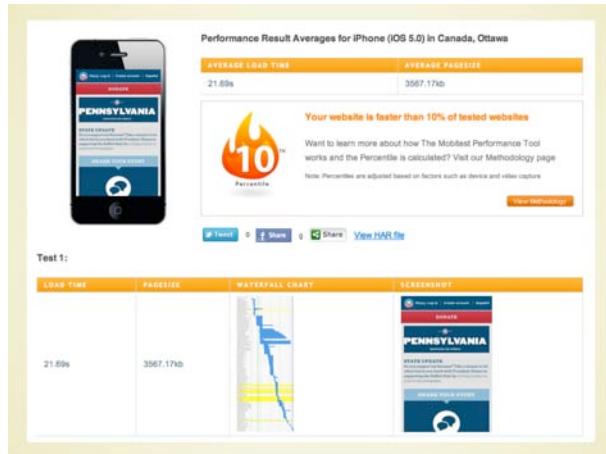
◆ Una pagina tipo del sito mobile di Romney pesa circa **687 KB** e ci mette circa 8.75 secondi ad essere caricata.



◆ Anche se questo valore è sopra al limite consigliato di 5 secondi, la pagina pesa di meno della media.

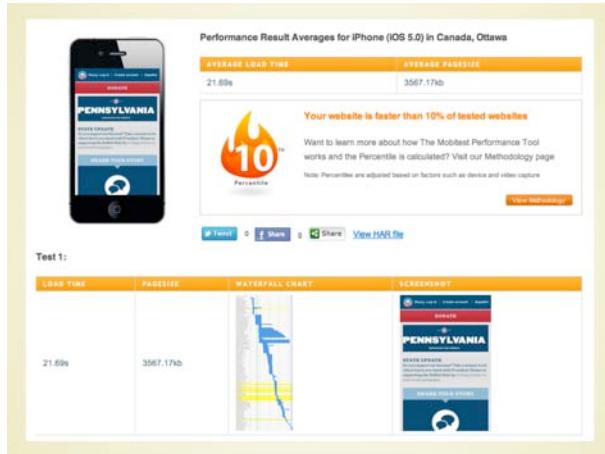
Performance: Obama

- ◆ Una pagina tipo del sito responsive di Obama pesa **4.2 MB** e richiede 25 secondi per il caricamento.
- ◆ A parte il fatto che solo i navigatori più pazienti sono disposti ad aspettare 25 secondi per il caricamento di una pagina, una quantità di dati così grande crea grossi problemi su alcuni dispositivi (es. alcuni BlackBerry) che non riescono nemmeno a renderizzare la pagina.

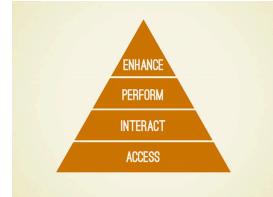


Performance, concludendo

- ◆ Sfortunatamente una cattiva performance caratterizza la maggior parte dei siti responsive. E' stato calcolato da Guy Podjarny, chief product architect di Akamai e CTO di Blaze.io, che solo il 3% delle versioni per piccolo schermo dei siti responsive sono significativamente meno pesanti delle versioni per schermi grande.
- ◆ Un ottimo motivo per focalizzarsi sulla performance come componente chiave di un web design adattivo.

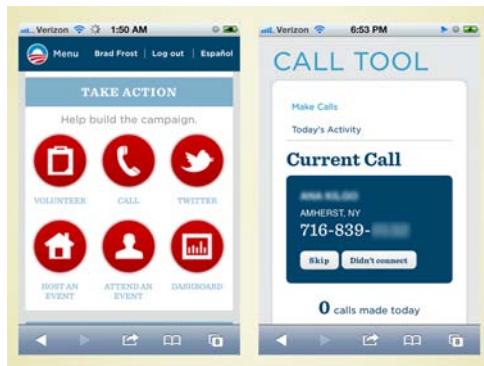


Enhance



- ◆ Il fatto che nelle slides precedenti si sia messo l'accento sull'importanza della **content parity** e sul fatto che sia fondamentale fornire accesso a informazioni e funzionalità a prescindere dal tipo di dispositivo e di configurazione **non significa** che si debba **fornire all'utente uno stesso tipo di esperienza**.
- ◆ Molti dispositivi e browser mobili hanno features che le controparti desktop non hanno e utilizzare queste potenzialità può permettere di portare l'esperienza in mobilità ad un nuovo livello.

Enhance: mobile communication



- ◆ I designer del sito di Obama hanno ad esempio implementato la possibilità, per gli utenti mobili che desiderano impegnarsi nella campagna, di fare chiamate da parte del candidato a potenziali elettori. Tendiamo a dimenticare che i dispositivi mobili permettono anche di telefonare e che i browser mobili possono far partire una chiamata semplicemente da un link tel presente nel codice HTML.

Enhance: form enhancements



- ◆ Le form per l'input dei dati spesso possono essere adattate con poco sforzo.
- ◆ Ad esempio un miglioramento reso possibile da HTML5 è la possibilità di specificare un determinato tipo di input (es. numerico); questa possibilità può essere utilizzata per attivare sul dispositivo la tastiera virtuale appropriata al momento dell'input.

Enhance: geolocation



- ◆ Rilevare la posizione geografica corrente è una possibilità importante per fornire agli utenti informazioni rilevanti nel contesto spaziale in cui ci si trova. Si può immaginare ad esempio di fornire all'utente post di blog, notizie ed eventi riferiti alla sua posizione. La rilevazione della posizione geografica potrebbe essere utilizzata dai siti dei candidati per identificare automaticamente notizie e proposte riguardanti lo Stato in cui si trova l'elettore.
- ◆ La geolocazione può aiutarci anche a risparmiare tempo nella compilazione di form e creare altre opportunità per l'implementazione di nuove features.

Riassumendo ...

- ◆ L'approccio responsive presenta numerosi vantaggi, ma devono essere valutati attentamente un insieme di fattori legati all'accesso, all'interazione, alla performance e al miglioramento dell'esperienza utente che potrebbero inficiare i vantaggi di questo approccio.

La soluzione migliore

- ◆ E' meglio un sito responsive oppure un sito mobile separato?



Responsive web design

◆ I tre ingredienti fondamentali

- ◆ Layout flessibile
- ◆ Immagini e media flessibili
- ◆ Media queries CSS

Griglia tipografica flessibile

- ◆ Anzichè partire da una dimensione della griglia tipografica espressa in pixel (ad es. 960 px) è possibile pensare ad una griglia tipografica che si riscalda, esprimendo le misure come percentuale della misura dell'elemento contenitore (il box associato al tag di livello superiore oppure la finestra, nel caso in cui il tag di livello superiore sia body), in modo che il design complessivo della pagina rimanga invariato al variare delle dimensioni della finestra.
- ◆ E' opportuno applicare la **stessa logica di flessibilità non solo** nello stabilire la **larghezza dei box** associati ai tag, **ma anche agli elementi correlati come il margine ed il padding**. Anche in questo caso il contesto a cui riferirli è la larghezza (o l'altezza) dell'oggetto contenitore.

Reset stylesheets

- ◆ I cosiddetti **reset stylesheets** sono collezioni di regole CSS associate alle pagine HTML, che vengono introdotte per avere un punto di partenza comune per la presentazione per tutti i browser, rispetto al quale specificare poi le regole di stile.
- ◆ In assenza di questi stylesheets e di ulteriori specifiche direttive per la presentazione da parte degli autori della pagina web, tutti i browser presenterebbero le informazioni secondo un formato di **default** che però **non è uguale per tutti i browser**. Ad esempio alcuni browser indentano le liste utilizzando il margine sinistro, mentre altri utilizzano il padding. Browser diversi utilizzano valori differenziati per i margini superiori e inferiori. Anche l'altezza della riga può variare a seconda del browser.
- ◆ Utilizzando le parole di Eric Meyer, possiamo dire che: “noi pensiamo che il nostro CSS modifichi l’aspetto di **default di un documento**, ma **con uno stylesheet di reset possiamo rendere questo aspetto più consistente tra i diversi browser**, impiegando meno tempo a combattere con i singoli default”.

Reset stylesheets

```
/* http://meyerweb.com/eric/tools/css/reset/
v2.0 | 20110126
License: none (public domain)
*/
html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre, a,
abbr, acronym, address, big, cite, code, del,
dfn, em, img, ins, kbd, q, s, samp, small,
strike, strong, sub, sup, tt, var, b, u, i,
center, dl, dt, dd, ol, ul, li, fieldset, form,
label, legend, table, caption, tbody, tfoot,
thead, tr, th, td, article, aside, canvas,
details, embed, figure, figcaption, footer,
header, hgroup, menu, nav, output, ruby,
section, summary,
time, mark, audio, video {
    margin: 0;
    padding: 0;
    border: 0;
    font-size: 100%;
    font: inherit;
    vertical-align: baseline;
}
/* HTML5 display-role reset for older browsers */
article, aside, details, figcaption, figure,
footer, header, hgroup, menu, nav, section {
    display: block;
}
body {
    line-height: 1;
}
ol, ul {
    list-style: none;
}
blockquote, q {
    quotes: none;
}
blockquote:before, blockquote:after,
q:before, q:after {
    content: '';
    content: none;
}
table {
    border-collapse: collapse;
    border-spacing: 0;
}
```

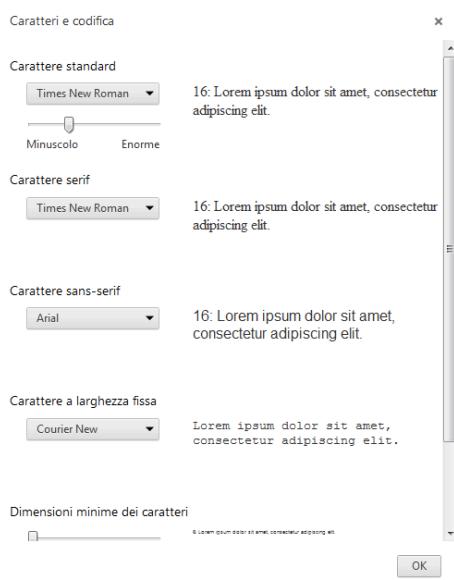
Caratteri tipografici flessibili

- ◆ Stabilire ad es. la seguente **regola di reset per i caratteri**

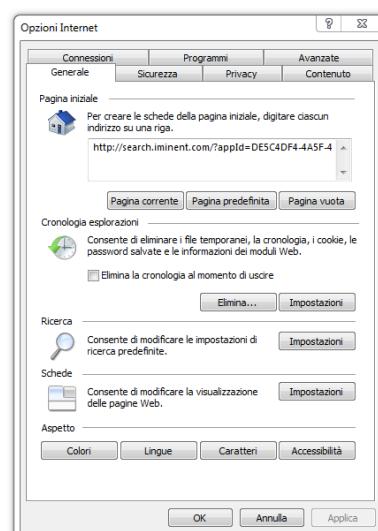
```
body {font: normal 100% Cambria, Georgia, serif;}
```

che fa coincidere la dimensione di base del carattere da utilizzare nelle diverse situazioni con la dimensione del carattere di default, corrisponde a definire una base comune per tutti i browser, dato che ormai la maggior parte utilizza i 16px come grandezza di default.

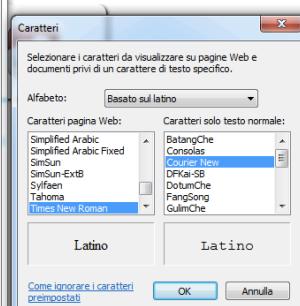
- ◆ Rispetto a questa regola è poi possibile definire poi le grandezze dei font nelle varie situazioni, utilizzando l'**unità di misura em**. Questo ci **consente di ottenere un'impostazione del carattere flessibile**, che potremo scalare a partire da quanto specificato nella regola CSS associata a body (ad esempio variando per body la percentuale che esprime la grandezza del carattere, automaticamente verrebbero scalate le grandezze di tutti gli altri caratteri).

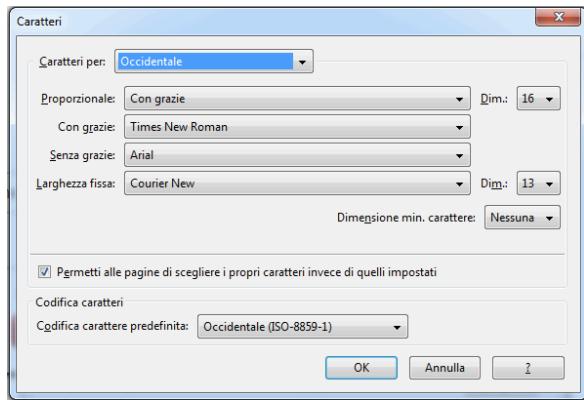


Selezione carattere tipografico
Chrome 21.0 Win

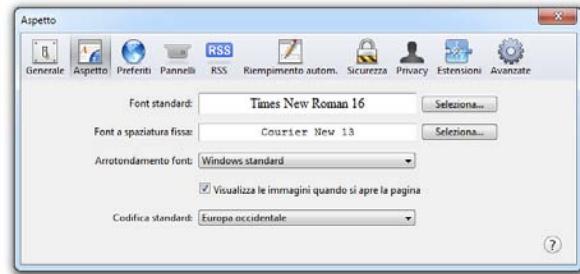


Selezione carattere tipografico
IE 9 Win

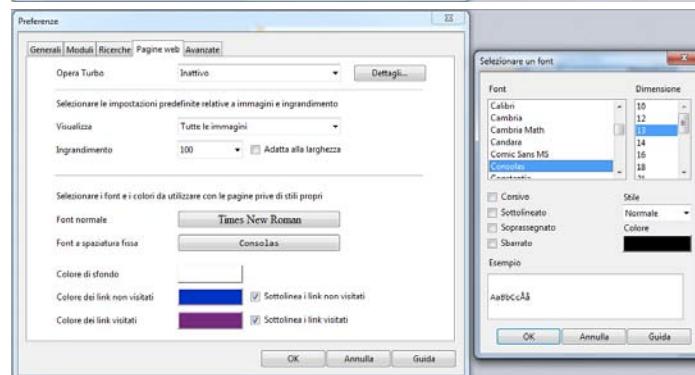
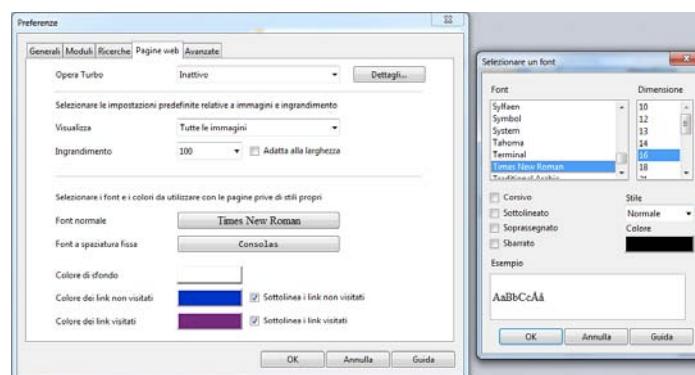




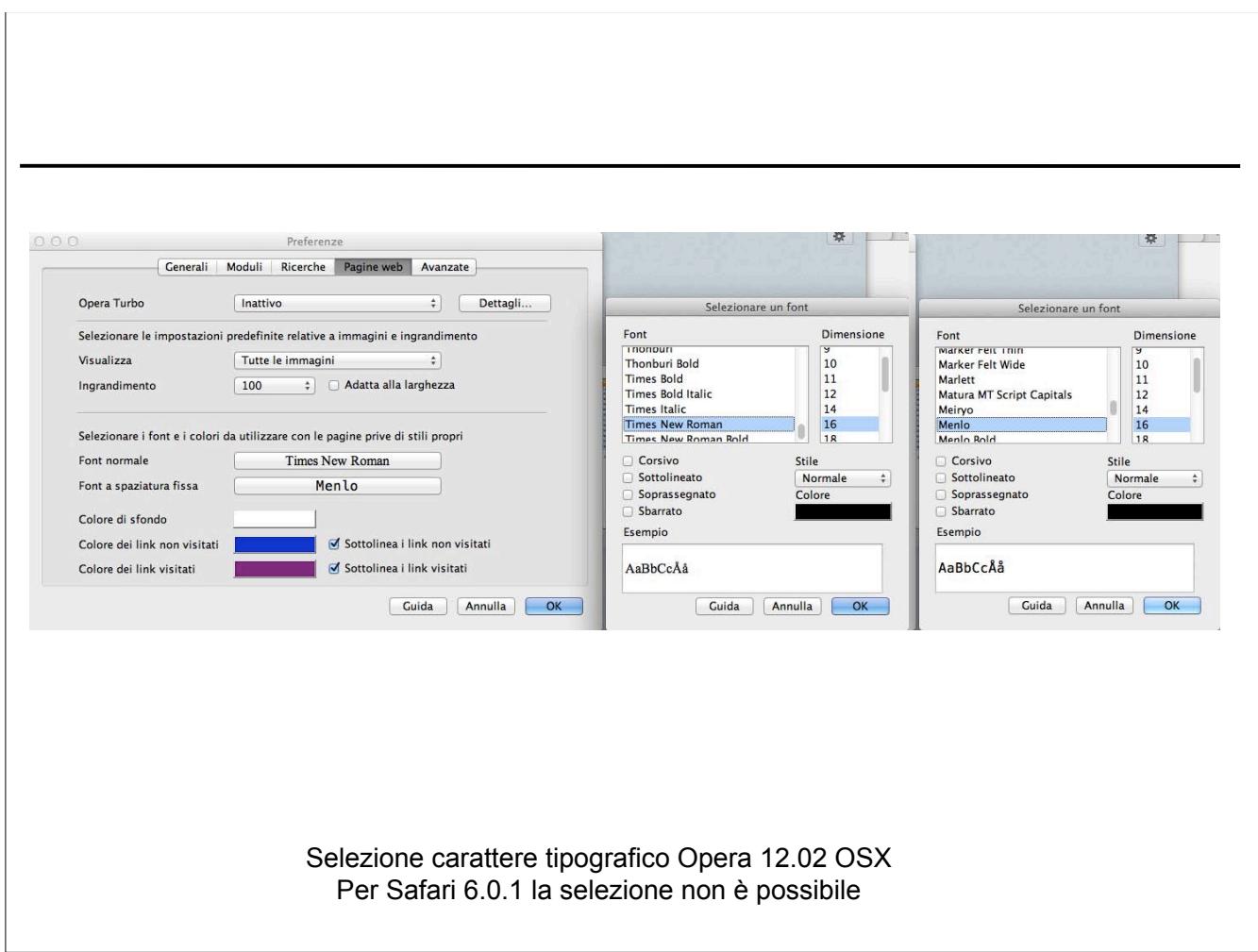
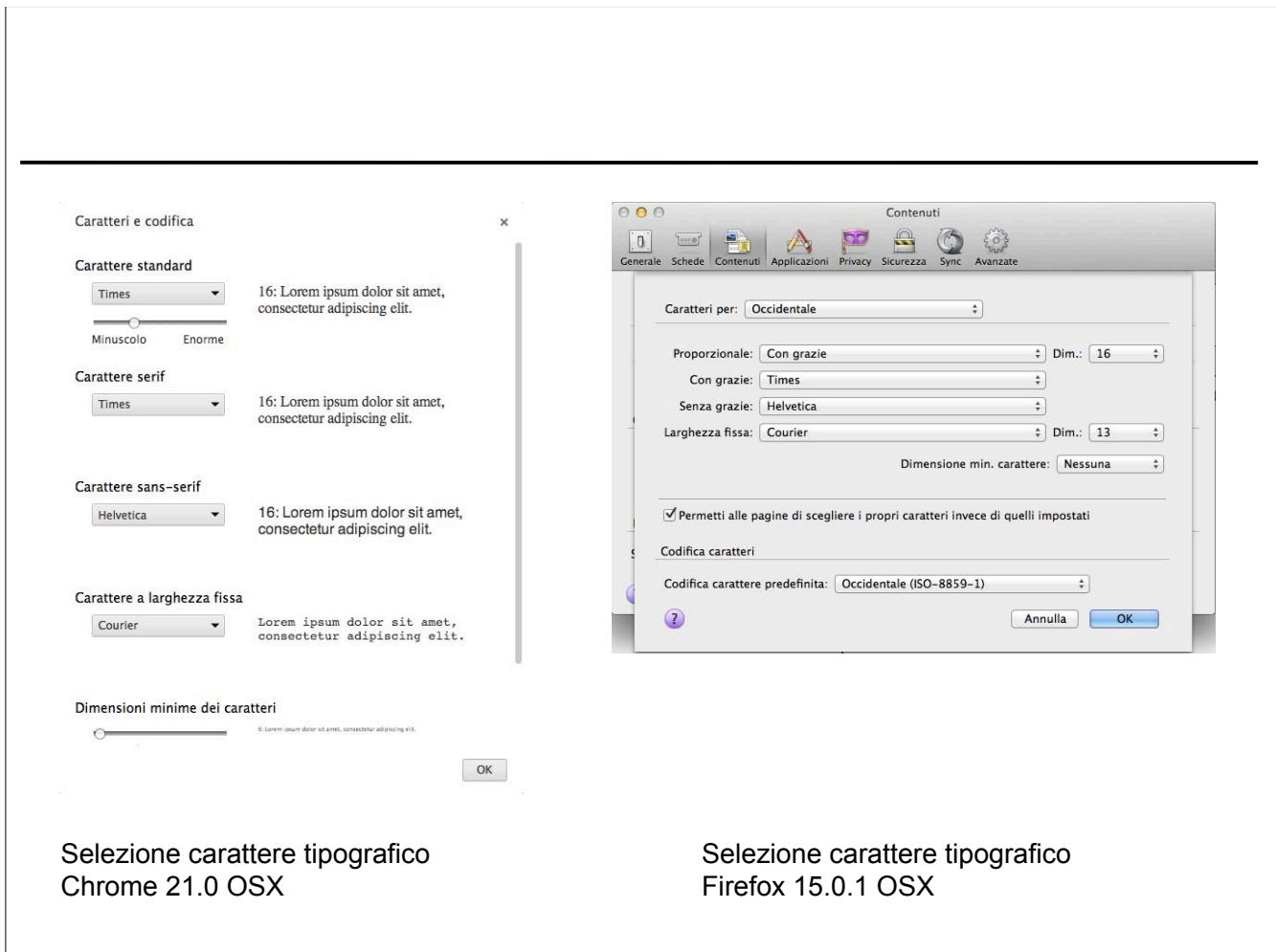
Selezione carattere tipografico
Firefox 11.0 Win



Selezione carattere tipografico
Safari 5.1.7 Win



Selezione carattere tipografico Opera 12.02 Win



Strumenti di navigazione e layout

- ◆ Anche la progettazione degli strumenti di navigazione dovrebbe tenere in considerazione la possibilità di fornire widget diversi a seconda della grandezza dello schermo a disposizione.
- ◆ Ad esempio esistono tecniche (vedi esempio trunk.js) che permettono di ottenere a partire dallo stesso codice HTML due modalità di presentazione diverse del sistema di navigazione principale, il primo come menu disposto orizzontalmente per gli schermi desktop, il secondo come drawer a scomparsa per gli schermi dello smartphone/tablet.

Immagini flessibili

- ◆ L'inserimento di un'immagine di grandezza fissa in un layout flessibile è problematica perché, nel caso in cui il contenitore dell'immagine abbia dimensioni inferiori a quella dell'immagine stessa, quest'ultima deborda dalle dimensioni stabilite per il contenitore, causando una presentazione non ottimale.
- ◆ Es.

```
<div class="figure">
  <p>
    
    <b class="figcaption">Lo, the robot walks</b>
  </p>
</div>
```

Immagini flessibili

- ◆ E' possibile fortunatamente **stabilire dei vincoli**

```
img {max-width: 100%;}  
(vedi http://bkaprt.com/rwd/11/)
```

che **stabiliscono una grandezza massima per l'immagine**. Nel caso in cui l'immagine sia più piccola essa verrà renderizzata utilizzando le sue dimensioni in pixel, ma nel caso in cui l'immagine sia più grande del contenitore verrà riscalata in modo da non superare i limiti del contenitore.

- ◆ A differenza di un tempo, i moderni browser garantiscono una riscalatura proporzionale di buona qualità dell'immagine.
- ◆ A margine si può aggiungere che la regola citata si può applicare alla maggior parte degli elementi a larghezza fissa, come il video o altri rich media

```
img, embed, object, video {max-width: 100%;}
```

Immagini flessibili e IE

- ◆ Purtroppo Internet Explorer fino alla versione 6 compresa non supportano la proprietà **max-width**
- ◆ Per risolvere questo problema ci sono diversi metodi, alcuni basati su Javascript. Un metodo alternativo basato su CSS prevede di utilizzare una regola di stile separata per IE 6 (utilizzando ad es. i commenti condizionali):

```
img, embed, object, video {width: 100%;}
```
- ◆ Nella maggior parte dei casi questo approccio (**width: 100%**) funziona bene, a parte le situazioni nelle quali un'immagine di piccole dimensioni deve essere scalata per occupare tutta l'area di un grande contenitore (perché l'immagine viene rascalata con un numero di pixel superiore a quello di partenza)
- ◆ Per ovviare a questo problema si possono utilizzare, anziché delle indicazioni riferite a tutte le immagini della pagina, dei selettori specifici (es. selettori contestuali o con specifica di classi) che limitino l'applicazione ad ambiti specifici

Immagini flessibili e IE

moment, but I am tired—say,
List Apart. As ever, it's a rare, h
nonsensical ravings into an artic
It's also something of a relief to h
a year ago during the W3C rade
about it for so long, it feels great
improved. So when you've a mo
hear your thoughts.

moment, but I am tired—say,
List Apart. As ever, it's a rare, h
nonsensical ravings into an artic
It's also something of a relief to h
a year ago during the W3C rade
about it for so long, it feels great
improved. So when you've a mo
hear your thoughts.

- ◆ Tuttavia i problemi non si limitano a questo, perchè su piattaforma Windows **alcuni browser** come Internet Explorer fino alla versione 7 o Firefox fino alla versione 2 **non scalano l'immagine con una buona qualità**, rendendone a volte problematica la comprensione.
- ◆ Per risolvere questo problema si può utilizzare **AlphaImageLoader**, un filtro CSS proprietario di Microsoft (<http://bkaprt.com/rwd/13/>) che veniva utilizzato generalmente per risolvere la mancanza di supporto dell'alpha channel delle immagini PNG in IE
- ◆ L'applicazione della **proprietà sizingMethod** di questo filtro con il valore **scale** ha come esito un incremento drastico della qualità del rendering in IE
- ◆ Per velocizzarne l'applicazione ne è stato predisposto uno script, reperibile in: <http://unstoppablerobotninja.com/entry/fluid-images>

Riassumendo ...

```
28| img {  
29|   max-width: 100%;  
30| }  
31| </style>  
32| <!--[if IE]><style type="text/css">  
33| img,  
34| p {  
35|   width: 100%;  
36| }  
37| </style><![endif]-->  
38|  
124| <body>  
125| <h1>Fluid is the new black.</h1>  
126| <p></p>  
127| <p><em>Boo-yah!</em> (a href="http://unstoppablerobotninja.com/entry/fluid-images/">Read the blog entry</a>.\)</p>  
128|  
129|  
130|  
131|  
132|</body>  
133|</html>
```

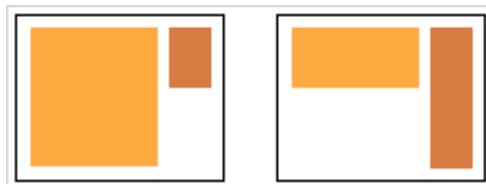
- ◆ L'esempio, scaricabile all'indirizzo web citato, è una sintesi delle tecniche esaminate:
 - ◆ il codice html fa riferimento ad un'immagine contenuta all'interno di un paragrafo;
 - ◆ le regole CSS indicano che l'immagine non deve oltrepassare la grandezza del contenitore; a favore di IE che non comprende la regola di stile precedente viene scritto il commento condizionale
- ◆ **Nota:** a rigore, dal momento che il problema si verifica solo per IE fino alla versione 6 si dovrebbe scrivere, in maniera più precisa, `<!--[if lt IE 6]>`

Riassumendo ...

```
40 | <script type="text/javascript" charset="utf-8">
41 | var imgSizer = {
42 |   Config : {
43 |     imgCache : []
44 |     ,spacer : "/path/to/your/spacer.gif"
45 |   }
46 |
47 |   ,collate : function(aScope) {
48 |     var isOldIE = (document.all && !window.opera && !window.XDomainRequest) ? 1 : 0;
49 |     if (isOldIE && document.getElementsByTagName) {
50 |       var c = imgSizer;
51 |       var imgCache = c.Config.imgCache;
52 |
53 |       addLoadEvent(function() {
54 |         imgSizer.collate();
55 |       });
56 |
57 |       function addLoadEvent(func) {
58 |         var oldonload = window.onload;
59 |         if (typeof window.onload != 'function') {
60 |           window.onload = func;
61 |         } else {
62 |           window.onload = function() {
63 |             if (oldonload) {
64 |               oldonload();
65 |             }
66 |             func();
67 |           }
68 |         }
69 |       }
70 |     }
71 |   }
72 | }
```

- ◆ Lo script (inserito nella sezione di testa) verrà eseguito automaticamente al caricamento della pagina e applicherà il fix a tutte le immagini della pagina

Immagini di background flessibili



- ◆ L'utilizzo di immagini di sfondo può essere utile per distinguere visivamente il fondo di due colonne di testo. La tecnica utilizzata, denominata delle false colonne, è stata introdotta qualche anno fa da Dan Cederholm, che aveva proposto questo escamotage per ovviare al fatto che con l'utilizzo dei tag <div> per definite le colonne di un layout non si riesce ad ottenere un background di eguale altezza (a meno che le due colonne non abbiano contenuti che occupano il medesimo spazio in altezza).
- ◆ La soluzione proposta consiste nell'utilizzare un background sull'elemento che contiene entrambe le colonne, simulando in questo modo l'esistenza di due colonne colorate.



Immagini di background flessibili

- ◆ L'immagine di Dan Cederholm era centrata sulla pagina e poi ripetuta verticalmente; presupponeva però che il layout avesse una dimensione fissa.
- ◆ Una proposta alternativa, che ne consente l'utilizzo con display flessibili, consiste nel **creare un'immagine più estesa della dimensione degli schermi utilizzati** finora (es. un'immagine larga 3000 pixel) nella quale vengano definite proporzionalmente le due partizioni delle colonne utilizzando un tool di manipolazione di immagini raster (es. Photoshop, Gimp).
- ◆ L'immagine viene poi posizionata come sfondo dell'oggetto contenitore delle due colonne, utilizzando una regola di stile nella quale viene indicata la sorgente, la ripetizione verticale e la posizione espressa in percentuale.
- ◆ Es. `.columncontainer {background: #F8F5F2 url("blog-bg.png") repeat-y 63.1111111111% 0;}`

Immagini di background flessibili

- ◆ Un concetto importante da capire, per quanto riguarda il posizionamento delle immagini è che:
 - ◆ se si usano **valori in pixel** questi indicano dove deve essere posizionato l'angolo superiore sinistro dell'immagine rispetto all'angolo superiore sinistro dell'oggetto contenitore
 - ◆ se si usano **valori percentuali** (il nostro caso), questi indicano sia la posizione in percentuale nell'immagine che la posizione in percentuale nell'oggetto contenitore; ad esempio se per un'immagine associata ad un determinato elemento si stabiliscono valori della proprietà **background-position** pari a 50% 50%, il punto dell'immagine che si trova al 50% della larghezza dell'immagine e al 50% della sua altezza verrà posizionato nel punto dell'elemento che si trova al 50% della larghezza e al 50% dell'altezza dell'elemento stesso. Nel caso specifico avremo un'immagine perfettamente centrata.

Immagini di background flessibili

- ◆ Quindi se se dobbiamo costruire uno sfondo per due colonne di colore rosso (la prima a sinistra) e verde (la seconda) tali che la prima debba avere una larghezza doppia della seconda, dovremo costruire un'immagine di sfondo nella quale la campitura rossa dell'immagine dovrà occupare il 66,6% dei pixel a partire da sinistra (il restante 33,3% dei pixel dovrà avere una campitura verde) e poi posizionarla con la seguente regola di stile:

```
.COLUMNCONTAINER  
{BACKGROUND: #F8F5F2 URL("BLOG-BG.PNG") REPEAT-Y 66.6%  
0;}
```

Immagini di background flessibili

- ◆ Si noti infine che le false colonne che abbiamo creato non sono realmente flessibili: **le dimensioni delle immagini non sono cambiate per nulla**, abbiamo solo utilizzato un metodo di posizionamento insieme ad un'immagine molto grande che da l'illusione di una scalatura dello sfondo.
- ◆ E' allo studio una proprietà CSS denominata **background-size**, che dovrebbe permettere di creare immagini di sfondo realmente flessibili, ma il supporto dei browser è al momento decisamente inconsistente.

Un'alternativa per il background

- ◆ Un'alternativa all'uso delle immagini di background flessibili è quella dell'utilizzo della regola CSS **overflow:auto** oppure **overflow:hidden** applicata al contenitore delle due colonne.
- ◆ L'applicazione di questa regola - combinata con la definizione di un colore di sfondo per la colonna con il contenuto più lungo, permette di ottenere due colonne con campiture colorate differenziate.
- ◆ In questo caso viene assegnato dunque un colore di sfondo al contenitore delle due colonne e a alla colonna con il contenuto più lungo; la colonna nidificata con il contenuto più corto non viene colorata

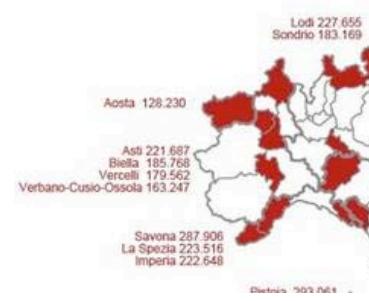
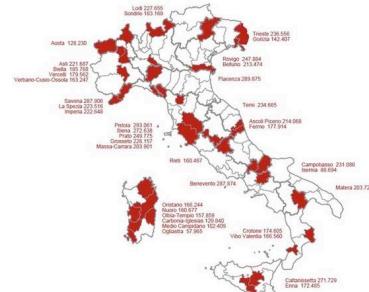
Un'alternativa per il background

- ◆ Il vantaggio di questa regola è che può essere applicata anche a layout con un numero di colonne superiore a due, con un modesto livello di aumento della struttura della pagina. Ad es. per gestire 3 colonne si potrà applicare (vedi codice a seguire) la regola di overflow a *contenitore3colonne*, *contenitore2colonne*; andrà poi applicato un background al div con contenuto più lungo (*colonna2* oppure *colonna3*)
- ◆

```
<div id="contenitore3colonne">
  <div id="colonna1">...</div>
  <div id="colontenitore2colonne">
    <div id="colonna2">...</div>
    <div id="colonna3">...</div>
  </div>
</div>
```

Overflow delle immagini: un'alternativa alla flessibilità

- ◆ Un'alternativa a mettere dei vincoli sulla grandezza di tutta l'immagine è di usare il **clipping**, visualizzando solo una parte dell'immagine stessa.
- ◆ Naturalmente questo è possibile solo nel caso in cui sia possibile rinunciare alle informazioni contenute nella parte dell'immagine che si intende tagliare.



Overflow delle immagini: un'alternativa alla flessibilità

- ◆ Dunque anzichè scrivere una regola di stile che comprima tutta l'informazione dell'immagine, come avevamo fatto precedentemente
 - ◆ `.feature img {max-width: 100%;}`
- ◆ vengono scritte due regole, la prima definita sul contenitore dell'immagine, la seconda sull'immagine stessa
 - ◆ `.feature {overflow: hidden;}`
 - ◆ `.feature img {display: block; max-width: auto;}`

Il terzo pilastro del responsive design: media queries

- ◆ La flessibilità nella presentazione del layout e delle immagini non mette al riparo dalle conseguenze di un restringimento o di un allargamento eccessivo della larghezza della finestra del browser. Le colonne di testo possono risultare eccessivamente corte o lunghe da leggere e possono sorgere problemi più gravi di presentazione.
- ◆ La specifica di CSS ha definito, nella versione 2, i **media type**, dando la possibilità di assegnare un set di regole specifiche ad un media specifico: **braille, embossed, handheld, print, projection, screen, speech, tty o tv**.

Il terzo pilastro del responsive design: media queries

- ◆ L'assegnazione delle regole di stile avveniva facendo riferimento ad uno stylesheet specifico, es.

```
<link rel="stylesheet" href="global.css" media="all" />
<link rel="stylesheet" href="main.css" media="screen" />
<link rel="stylesheet" href="paper.css" media="print" />
```
- ◆ oppure definendo sezioni specifiche all'interno di uno stesso foglio di stile

```
@media screen { body {font-size: 100%;}}
@media print { body {font-size: 15pt;}}
```

Il terzo pilastro del responsive design: media queries

- ◆ Purtroppo la definizione dei media type si è rivelata grossolana, in alcune situazioni particolari, come quella dei dispositivi mobili, che si sono sempre più differenziati per numero di pixel, capacità di calcolo e velocità di comunicazione via rete.
- ◆ Il **media handheld** si è rilevato pertanto insufficiente per gestire questa situazione. I progettisti di browser per smartphone (es. Safari per iOS), in una situazione caratterizzata anche dalla carenza di siti concepiti per il media handheld, hanno utilizzato le nuove caratteristiche dei dispositivi applicando le regole di presentazione associate al media screen.
- ◆ La soluzione di fare riferimento allo stesso media pensato per i dispositivi desktop non è tuttavia ideale perché, nonostante le nuove possibilità offerte dalle interfacce touch, richiede all'utente operazioni continue di zoom per riuscire a leggere i contenuti di proprio interesse.

Il terzo pilastro del responsive design: media queries

L'introduzione delle media queries nella specifica CSS3 permette di **rendere meno grossolana la suddivisione tra classi diverse di dispositivi**, consentendo di identificare un insieme di proprietà che caratterizzano il browser utilizzato dall'utente. Queste proprietà sono riassunte nella tabella a fianco.

proprietà	valore	min/max	descrizione
color	integer	si	numero di bit per componente di colore
color-index	integer	si	numero di componenti della color lookup table
device-aspect-ratio	integer/integer	si	rapporto larghezza/altezza della superficie di rendering
aspect-ratio	integer/integer	si	rapporto larghezza/altezza della display area
device-height	length	si	altezza della superficie di rendering del dispositivo
height	length	si	altezza della display area
device-width	length	si	larghezza della superficie di rendering del dispositivo
width	length	si	larghezza della display area
grid	integer	no	true per un disp. grid-based (es. telefono a font fisso)
monochrome	integer	si	numero di bit per pixel in un dispositivo monocromat.
resolution	resolution (es. dpi)	si	densità dei pixel nel device
scan	progressive o interlaced	no	tipo di scansione dello schermo per dispositivo tv
orientation	portrait o landscape	no	orientamento del dispositivo

Il terzo pilastro del responsive design: media queries

- ❖ A proposito delle features descritte è opportuno chiarire alcune definizioni.
- ❖ Per **superficie di rendering** del dispositivo si intende l'area dello schermo del dispositivo (perimetro rosso immagine a fianco)
- ❖ Per **display area** si intende lo spazio utile per la presentazione dei contenuti nella pagina web (perimetro ocra tratteggiato img. a fianco)
 - ❖ Per i **media continui** corrisponde alla **viewport**, cioè l'area attraverso la quale l'utente può visualizzare - eventualmente utilizzando un meccanismo di scrolling - il contenuto renderizzato sulla **canvas** (la canvas è l'area virtuale nella quale il contenuto di un documento html viene renderizzato)
 - ❖ Per i media non continui (**paged media**), che suddividono in contenuto in pagine, corrisponde al **page box** (cioè l'area della pagina + i margini)



Il terzo pilastro del responsive design: media queries

- ❖ Le media queries possono essere utilizzate in tutte le situazioni in cui vengono effettuati riferimenti ai media types, dall'ambito di un riferimento ad un foglio di stile esterno alla specifica di una sezione di regole embedded all'interno di un documento html.
 - ❖ <link rel="stylesheet" type="text/css" href="smallscreen.css" media="only screen and (max-width: 480px)" />
 - ❖ @import "smallscreen.css" only screen and (max-width: 480px);
 - ❖ <style type="text/css">
 @media only screen and (max-width: 480px){ /* rules defined inside */}
 </style>
- ❖ E' possibile concatenare query multiple in un'unica espressione:
- ❖ Es. @media screen and (min-device-width: 480px) and (orientation: landscape) { ... }
- ❖ nota: la keyword **only** può essere utilizzata in una media query per nascondere le regole di stile ai browser di vecchia generazione; per i nuovi browser scrivere only screen oppure screen è equivalente

Breakpoints

- ◆ Riassumendo, possiamo utilizzare le media queries per descrivere regole diverse di presentazione. Il primo passo sarà quello di definire dei **breakpoint** rispetto ai quali poi definire regole di presentazione che forniscono il risultato migliore nelle diverse situazioni.

breakpoint	descrizione
320 pixels	For small screen devices, like phones, held in portrait mode.
480 pixels	For small screen devices, like phones, held in landscape mode.
600 pixels	Smaller tablets, like the Amazon Kindle (6007800) and Barnes & Noble Nook (60071024), held in portrait mode.
768 pixels	Ten-inch tablets like the iPad (76871024) held in portrait mode.
1024 pixels	Tablets like the iPad (10247768) held in landscape mode, as well as certain laptop, netbook, and desktop displays.
1200 pixels	For widescreen displays, primarily laptop and desktop browsers.

Breakpoints

- ◆ Ad esempio, se volessimo distinguere 3 classi di regole di presentazione associate a dispositivi come smarphones, tablet e dispositivi laptop/desktop, potremmo dichiarare nelle diverse sezioni in un unico foglio di stile:
 - ◆

```
<style type="text/css">
  /* regole per desktop layout */
  @media screen and (max-width: 400px){/* regole per smartphone portrait */}
  @media screen and (max-width: 768px){/* regole per tablet portrait */}
</style>
```
- ◆ Vengono dunque definite un set di regole valide per la situazione standard, quella del layout multicolonnare per dispositivo desktop; successivamente vengono indicate regole specifiche (che potranno anche sovrascrivere le precedenti) nel caso in cui la finestra per la visualizzazione abbia una larghezza fino a 400 pixel o abbia una larghezza fino a 768 pixel. Si noti che **specificando max-width e non max-device-width le regole verranno applicate anche ai sistemi desktop, nel caso in cui la finestra del browser venga ridimensionata.**

Breakpoints

- ◆ A questo si può aggiungere una sezione caratterizzata dalla query **@media screen and (min-width: 1200px)**, per **evitare** che per schermi molto grandi il layout flessibile faccia diventare le **righe di contenuto molto estese e di difficile lettura**
 - ◆

```
<style type="text/css">
  /* regole per desktop layout */

  @media screen and (max-width: 400px){/* regole per smartphone portrait */}
  @media screen and (max-width: 768px){/* regole per tablet portrait */}
  @media screen and (min-width: 1200px){/* regole per desktop grandi */}
</style>
```
- ◆ Nella sezione riferita agli schermi molto grandi i box associati alle sezioni di testo avranno regole di stile che ne specificheranno la larghezza massima
 - ◆ es. `#contenuto {max-width: 800px;}`

Verso una filosofia Mobile first

- ◆ Questo approccio tuttavia non mette la mobilità al centro dell'attenzione. Infatti, se qualcosa dovesse andare storto, come una difettosa interpretazione delle media-queries o di una patch Javascript (da utilizzare per quei browser che hanno difficoltà ad interpretare le media queries), verrebbe effettuato un rendering che utilizza solo le regole CSS della prima sezione dello stylesheet, quelle riferite al desktop.
- ◆ La filosofia mobile first, descritta da Luke Wroblewski, ha fortissime affinità con quella del **progressive enhancement**, che veniva così descritta nel 2003 da Nick Finck e Steven Champeon:

Rather than hoping for graceful degradation, progressive enhancement builds documents for the least capable or differently capable devices first, then moves on to enhance those documents with separate logic for presentation, in ways that don't place an undue burden on baseline devices but which allow a richer experience for those users with modern graphical browser software.

Mobile first

- ◆ Applicando questa filosofia vengono prima descritte le regole di stile per una presentazione lineare, poi vengono redatte sezioni specifiche utilizzando la proprietà **min-width**; descrivendo via che cosa succede a mano a mano che la finestra si allarga:

```
◆ <style type="text/css">  
  /* regole per layout lineare */  
  @media screen and (min-width: 400px){/* rules inside */}  
  @media screen and (min-width: 768px){/* rules inside */}  
  @media screen and (min-width: 1200px){/* rules inside */}  
</style>
```

Supporto dei browser per le media queries

- ◆ Le media query godono di un buon livello di supporto da parte dei browser moderni. Opera le supporta fino dalla versione 9.5, Firefox dalla 4.5. Anche i browser basati su WebKit, come and Chrome, supportano le media queries. Anche Internet Explorer 9 le supporta. Per quanto riguarda i browser mobili, Safari e il browser di Android le supportano, così come Opera Mobile e Opera Mini e il browser di Mozilla.
- ◆ Sfortunatamente IE fino alla versione 8 compresa non le supportano.

Supporto dei browser per le media queries

- ◆ Ci sono tuttavia un certo numero di soluzioni basate sull'uso di Javascript:
 - css3-mediaqueries.js** (<http://bkaprt.com/rwd/37/>) e **respond.js** (<http://bkaprt.com/rwd/38/>)
 - ◆ css3-mediaqueries.js supporta quasi tutte le media queries
 - ◆ respond.js supporta solo min-width e max-width queries; richiede per essere utilizzata un piccolo hack, consistente nell'aggiungere alla fine di ogni media query un commento */mediaquery*/; ad esempio:
 - ◆ @media screen and (max-width: 768px) { }/*mediaquery*/
 - ◆ In entrambi i casi l'uso è semplice, si richiede solo di collegare al file HTML lo script; naturalmente il limite della soluzione è che l'interprete Javascript del browser deve essere attivato.

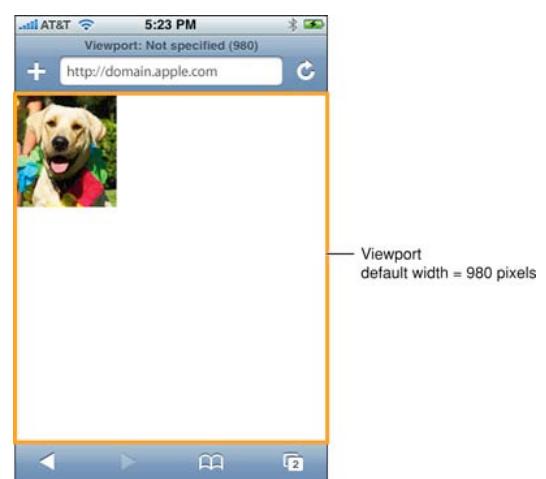
Concludendo ...

- ◆ Attraverso l'uso combinato di layout flessibili, media flessibili e media queries possiamo aderire ad uno stile di progettazione di tipo responsive, nel quale venga una presentazione adeguata a classi diverse di dispositivi.
- ◆ Le slides successive approfondiscono alcune problematiche particolari, per completare il quadro delle conoscenze necessarie.

Relazione tra viewport e dispositivo

Relazione tra viewport e dispositivo

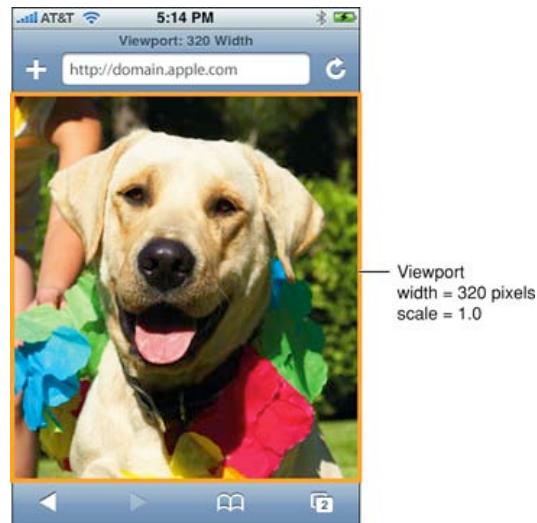
- ◆ La dimensione della viewport viene stabilita dallo user agent (in altre parole il browser).
- ◆ Non è detto tuttavia che la dimensione in pixel della viewport corrisponda alla dimensione in pixel hardware o in pixel CSS del dispositivo che mostrerà la pagina web.
- ◆ Ad esempio per **default Safari per iPhone** renderizza il contenuto su una **viewport di larghezza pari a 980 px** per poi restringerlo per la presentazione su uno **schermo** con una **larghezza effettiva pari a 320 px CSS**. Lo scopo è quello di presentare immediatamente all'utente una miniatura generalmente completa all'utente, che poi utilizzerà il meccanismo di zoom per leggere il contenuto di interesse.



Nota: vedi più avanti per la definizione di pixel CSS

Relazione tra viewport e dispositivo

- ◆ E' tuttavia modificare questa impostazione, agendo sui valori di un tag meta che è stato introdotto da Apple e che è diventato successivamente uno standard de facto. Questo tag permette di definire il valore della viewport attraverso l'attributo **content**. In sostanza viene dichiarato esplicitamente il valore della viewport.
 - ◆ Ad es. `<meta name="viewport" content="width=320" />` stabilisce che la dimensione della viewport deve essere 320 e non 980



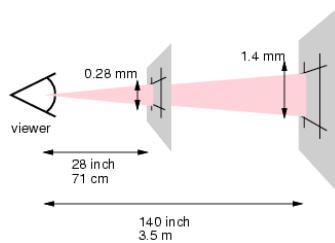
Relazione tra viewport e dispositivo

- ◆ In maniera più raffinata rispetto all'indicazione di un valore in pixel si può indicare che il livello di zoom iniziale con cui presentare il contenuto è 1, e che la larghezza da utilizzare deve corrispondere alla larghezza del dispositivo
 - ◆ `<meta name="viewport" content="initial-scale=1.0, width=device-width" />`
- ◆ In questo modo ad esempio è possibile utilizzare il diverso numero di pixel che sia hanno a disposizione del dispositivo a seconda che lo si utilizzi in modalità portrait o landscape, oltre che riuscire a utilizzare a pieno lo schermo di dispositivi che hanno un numero di pixel diversi (ad es. dispositivi Android).

Non ci sono più i pixel di una volta ...

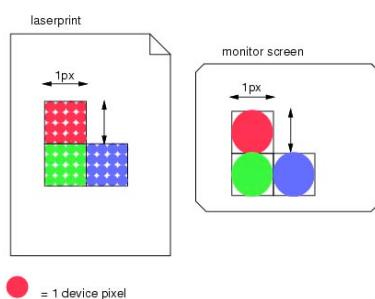
Non ci sono più i pixel di una volta

- ❖ Sebbene l'uso del termine pixel nell'ambito della specifica CSS porti a pensare ad una coincidenza con l'entità minima di rappresentazione dell'informazione su un dispositivo hardware, questo non è necessariamente vero.
- ❖ La specifica CSS indica, fin dalla versione 2.1, che l'unità di misura **pixel CSS** è ancorata al cosiddetto **pixel di riferimento** (reference pixel), rispetto al quale va definito il rapporto con i **pixel hardware**.
- ❖ Il pixel di riferimento è quello che viene visto da un occhio che osserva un pixel fisico di densità pari a 96 dpi ad una distanza di un braccio (28 inches).
- ❖ Il pixel di riferimento viene dunque definito attraverso un angolo!



Non ci sono più i pixel di una volta

- ◆ Si noti che nella definizione del pixel di riferimento vengono considerati **sia la densità della matrice dei pixel hardware sia la distanza a cui questa matrice viene vista dall'utente.**
- ◆ La scelta di questi parametri deriva dal fatto che molti dei contenuti web sono stati finora progettati per schermi desktop (quindi utilizzati ad una distanza di circa un braccio dall'occhio di un utente) con una densità di 96 dpi. Scegliere un riferimento diverso avrebbe significato il significato di uno strappo rispetto al lavoro di progettazione svolto finora.



Non ci sono più i pixel di una volta

- ◆ Riassumendo, se la densità della matrice fisica del dispositivo e la distanza di utilizzo dallo sguardo dell'utente corrispondono a quanto specificato nella slide precedente, il mapping tra pixel CSS e pixel hardware sarà di 1 a 1. Altrimenti, il costruttore di dispositivi dovrà tener presente il rapporto tra le due grandezze per utilizzarlo nel mapping tra specifica CSS e rendering della pagina, utilizzando i seguenti passaggi:



calcolo della densità equivalente, basato sulla conoscenza della densità di riferimento (96 dpi), della distanza di riferimento (28 pollici) e della distanza d'uso del nuovo dispositivo

densità equivalente =

(distanza di riferimento / distanza d'uso nuovo dispositivo) * densità di riferimento

si calcolerà poi la **devicepixelratio** (o rapporto tra pixel hardware e pixel CSS) sulla base della conoscenza del risultato precedente e della densità dei pixel hardware del nuovo dispositivo

rapporto pixel hardware / pixel CSS (**devicepixelratio**) = densità nuovo dispositivo effettiva / densità equivalente

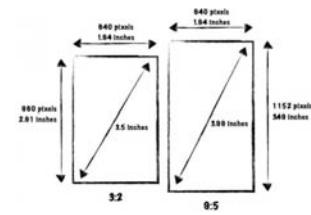
Non ci sono più i pixel di una volta

- ◆ Ad esempio effettuiamo questo calcolo per uno smartphone con schermo a 180 dpi e che stimiamo usato ad una distanza di 18 pollici:
 - ◆ densità equivalente = $(28/18)*96=150$ dpi circa
 - ◆ devicepixelratio = $180/150=1.2$
 - ◆ questo è il valore che il progettista del browser dovrà utilizzare quando il documento renderizzato in pixel CSS dovrà essere mappato sui pixel reali del dispositivo
 - ◆ ad esempio un box largo 100 px CSS dovrà essere renderizzato utilizzando 120 pixel fisici



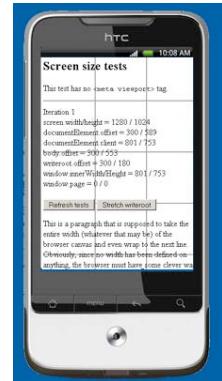
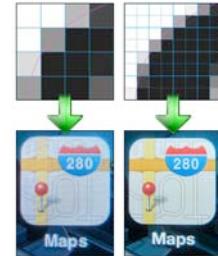
Non ci sono più i pixel di una volta

- ◆ Apple ha scelto per il suo iPhone di semplificare il mapping, ponendo pari a 1 la relazione tra pixel fisici e pixel CSS; successivamente ha mappato i dispositivi iPhone a densità doppia (iPhone4 e iPhone4S) ponendo il valore di questa relazione pari a 2.
 - ◆ iPhone fino alla versione 3 (480x320) devicepixelratio=1
 - ◆ iPhone 4 e 4s (960x640) devicepixelratio=2
 - ◆ iPhone 5 (1136x640) devicepixelratio=2
- ◆ Analogamente è stato stabilito un mapping 1:1 per iPad (1 e 2) ed un mapping 2:1 per gli iPad successivi a densità raddoppiata.
- ◆ Una situazione analoga si ritrova per i MacBookPro normali (mapping 1:1) e per quelli con retina display a densità doppia (mapping 2:1)



Non ci sono più i pixel di una volta

- ◆ Non tutti i vendor di dispositivi e di browser hanno scelto la stessa strada.
Per Android la situazione è abbastanza variegata:
 - ◆ Per il Nexus One + Webkit/Opera (480x800 pixel fisici) devicepixelratio=1.5
 - ◆ Per il galaxy Nexus + Chrome (720x1200) devicepixelratio=2
 - ◆ Per il galaxy Nexus + Opera (720x1200) devicepixelratio=2.25
 - ◆ Per il galaxy Note + Webkit/Chrome (800x1200) devicepixelratio=2
 - ◆ Per il galaxy Note + Opera (800x1200) devicepixelratio=2.25
- ◆ Il valore di devicepixelratio verrà utilizzato dal browser al momento di convertire le misure in pixel CSS a pixel hardware effettivi
- ◆ Si noti che in alcuni casi il valore dipende non solo dal dispositivo, ma anche dal browser



Non ci sono più i pixel di una volta

- ◆ Visualizzare sul proprio dispositivo mobile:
 - ◆ <http://www.quirksmode.org/m/tests/widthtest.html>
 - ◆ http://www.quirksmode.org/m/tests/widthtest_vpdevice.html
- ◆ Generare uno screenshot per ogni pagina
- ◆ Inviare i due screenshot a pitt@unive.it specificando nel subject della mail
webdesign-<nome dispositivo>
es. **webdesign-android_nexus_s**



Android Nexus 4

- ◆ pixel hardware
768x1280
- ◆ viewport
default=980
- ◆ devicepixelratio=2

The screenshot shows the "Screen size tests" section of the quirksmode.org/m/tests/widtht page. The page title is "Screen size tests". It displays a grid with three columns labeled "100px", "200px", and "300px" and three rows labeled "100px", "200px", and "300px". A note states: "This test has no <meta name='viewport' content='width = device-width'> tag." Below the grid, under "Iteration 1", it shows: screen.width/height = 384 / 592, documentElement.offset = 384 / 509, documentElement.client = 384 / 519, window.innerWidth/Height = 384 / 519, window.devicePixelRatio = 2, and window.page = 0 / 0. At the bottom, there are "Refresh tests" and "Stretch writeroot" buttons.

The screenshot shows the "Screen size tests" section of the quirksmode.org/m/tests/widtht page. The page title is "Screen size tests". It displays a grid with three columns labeled "100px", "200px", and "300px" and three rows labeled "100px", "200px", and "300px". A note states: "This test has no <meta name='viewport' tag.". Below the grid, under "Iteration 1", it shows: screen.width/height = 384 / 592, documentElement.offset = 980 / 1491, documentElement.client = 980 / 1324, window.innerWidth/Height = 980 / 1325, window.devicePixelRatio = 2, and window.page = 0 / 0. A note below says: "This is a paragraph that is supposed to take the entire width (whatever that may be) of the browser canvas and even wrap to the next line. Obviously, since no width has been defined on anything, the browser must have some clever way of handling this paragraph, or it'll be unreadable. Right now I feel that the different between browser experiences will mainly depend on handling this issue." At the bottom, there are "Refresh tests" and "Stretch writeroot" buttons.

Galaxy Nexus browser Chrome

- ◆ pixel hardware
720x1280
- ◆ viewport
default=980
- ◆ devicepixelratio=2

The screenshot shows the "Screen size tests" section of the quirksmode.org/m/tests/widtht page. The page title is "Screen size tests". It displays a grid with three columns labeled "100px", "200px", and "300px" and three rows labeled "100px", "200px", and "300px". A note states: "This test has no <meta name='viewport' tag.". Below the grid, under "Iteration 1", it shows: screen.width/height = 360 / 592, documentElement.offset = 980 / 1637, documentElement.client = 980 / 1412, window.innerWidth/Height = 992 / 1431, window.devicePixelRatio = 2, and window.page = 0 / 0. At the bottom, there are "Refresh tests" and "Stretch writeroot" buttons.

The screenshot shows the "Screen size tests" section of the quirksmode.org/m/tests/widtht page. The page title is "Screen size tests". It displays a grid with three columns labeled "100px", "200px", and "300px" and three rows labeled "100px", "200px", and "300px". A note states: "This test has no <meta name='viewport' content='width = device-width'> tag.". Below the grid, under "Iteration 1", it shows: screen.width/height = 360 / 592, documentElement.offset = 360 / 498, documentElement.client = 360 / 519, window.innerWidth/Height = 361 / 521, window.devicePixelRatio = 2, and window.page = 0 / 0. A note below says: "This is a paragraph that is supposed to take the entire width (whatever that may be) of the browser canvas and even wrap to the next line. Obviously, since no width has been defined on anything, the browser must have some clever way of handling this paragraph, or it'll be unreadable. Right now I feel that the different between browser experiences will mainly depend on handling this issue." At the bottom, there are "Refresh tests" and "Stretch writeroot" buttons.

Galaxy Nexus browser default

- ❖ pixel hardware
720x1280
- ❖ viewport
default=980
- ❖ devicepixelratio=2

The screenshots show the 'Screen size tests' page from quirksmode.org/m/. The left screenshot is for a browser without a viewport meta tag, and the right screenshot is for one with `<meta name="viewport" content="width=device-width">`. Both screenshots show a grid of 100px columns and 100px rows. The left screenshot has a warning message: "This test has no <meta viewport> tag." The right screenshot has a message: "This test has <meta name="viewport" content="width=device-width"> tag." Below the grid, there's a paragraph about handling spans and a variants section with two options: "Without viewport meta tag" and "With viewport width set to 380".

IOS - IPhone 4

- ❖ pixel hardware
640x960
- ❖ viewport
default=980
- ❖ devicepixelratio=2

The screenshots show the 'Screen size tests' page from quirksmode.org/m/. The left screenshot is for a browser without a viewport meta tag, and the right screenshot is for one with `<meta name="viewport" content="width=device-width">`. Both screenshots show a grid of 100px columns and 100px rows. The left screenshot has a warning message: "This test has no <meta viewport> tag." The right screenshot has a message: "This test has <meta name="viewport" content="width=device-width"> tag." Below the grid, there's a paragraph about handling spans and a variants section with three options: "Without viewport meta tag", "With viewport width set to 380", and "With hard-coded width: 1500px for the writeRoot".

IOS - IPhone 4 s

- ◆ pixel hardware
640x960
- ◆ viewport
default=980
- ◆ devicepixelratio=2

The screenshots show the 'Screen size tests' page from quirksmode.org. The left screenshot is from an iPhone 4s with a 640x960 pixel hardware and a default viewport of 980px. It displays a paragraph that spans the entire width of the screen, which is 320px. The right screenshot is from the same device but with a viewport meta tag set to 'width=device-width'. In this case, the paragraph is scaled down to fit the 320px width of the screen.

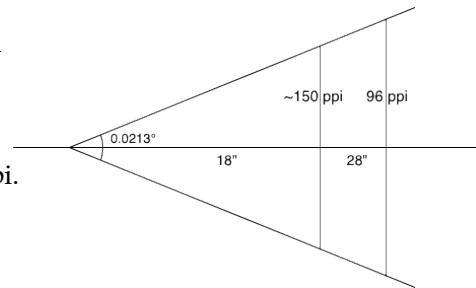
Wii U

- ◆ pixel hardware 854x480
- ◆ viewport default=980
- ◆ devicepixelratio=1

The screenshots show the 'Screen size tests' page from quirksmode.org. The top screenshot is from a browser window on a Wii U, showing the page without a viewport meta tag. The content spans the full width of the screen, which is 854px. The bottom screenshot is from the same browser window but with a viewport meta tag set to 'width=device-width'. In this case, the content is scaled down to fit the 854px width of the screen.

Non ci sono più i pixel di una volta

- ◆ Nota: la definizione di **dip (device-independent pixel)**, introdotta per i devices Android (ma non parte della specifica CSS) corrisponde sostanzialmente alla definizione del pixel CSS, dal momento che un dip corrisponde - secondo la sua stessa definizione - ad un pixel fisico ad una densità di 160 dpi. Se consideriamo che la definizione è riferita ad uno smartphone che viene visto a circa 18 pollici di distanza dallo sguardo, vediamo che il valore è molto simile a quello ottenuto con il calcolo della slide precedente, riferito appunto ad uno smartphone.
- ◆ In definitiva, possiamo dire nella sostanza che **la definizione di un pixel CSS si sovrappone a quella di un dip**, o meglio, che il numero di dips è uguale al numero di pixel CSS ottimale per visualizzare un sito web al 100% di zoom.



Immagini High-DPI

Immagini High-DPI

- ◆ L'introduzione di nuovi dispositivi ad elevata densità di pixel hardware porta nuove opportunità per la rappresentazione delle immagini; al tempo stesso va garantita la rappresentazione delle immagini su dispositivi a densità più bassa.
- ◆ Al di là dell'opportunità non spesso attuabile di utilizzare formati di **immagini vettoriali** (es. SVG) ci sono un buon numero di tecniche per risolvere il problema di mostrare le **immagini raster** con la qualità migliore permessa dal dispositivo, che possiamo dividere in due grandi categorie:
 - ◆ tecniche per ottimizzazione le singole immagini
 - ◆ tecniche per ottimizzare la selezione tra immagini multiple

Immagini High-DPI

- ◆ Le tecniche che operano sulla singola immagine sacrificano la performance, dal momento che dovranno essere scaricata immagine High-DPI anche per i dispositivi a DPI più bassi. Gli approcci possibili sono i seguenti:
 - ◆ **utilizzo di immagini High-DPI fortemente compresse**, che permettono di ottenere un risultato migliore di quello ottenibile con le immagini non HiDPI (ma peggio naturalmente di un'immagine HiDPI poco compressa)



Immagini High-DPI

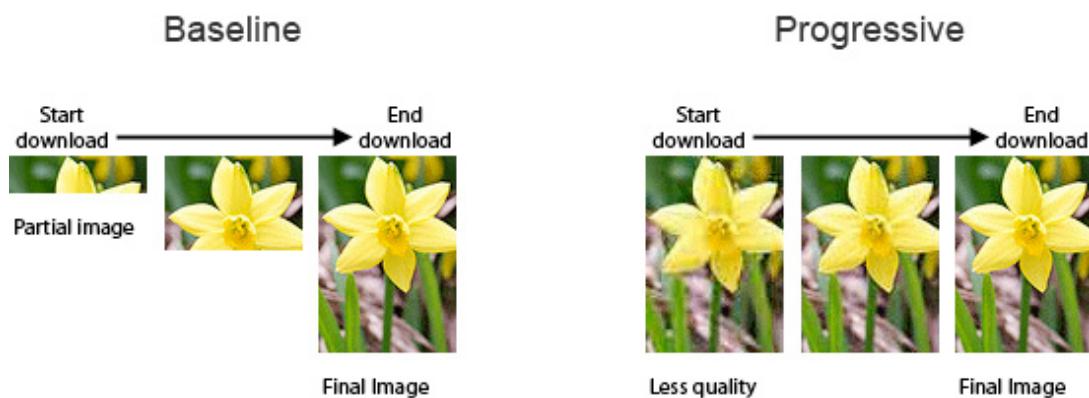
- ◆ **utilizzo di nuovi formati di immagine**, come il formato **Webp**, che comprime in maniera più efficiente ma non è ancora ampiamente supportato

webp

JPEG	WEBP
	
136780 bytes	122260 bytes (10.6%)

Immagini High-DPI

- ◆ **utilizzo di formati di immagine di tipo progressive**, insieme a tecniche che fermano il caricamento dell'immagine quando si raggiunge un livello di qualità sufficiente



Immagini High-DPI

- ❖ Le tecniche che operano per ottimizzare la **selezione tra immagini multiple** hanno in comune un **overhead nella creazione di versioni diverse della stessa immagine** e nella definizione di una strategia di selezione; le opzioni possibili includono:
 - ❖ **l'uso di JavaScript** per determinare il valore di **devicepixelratio**, le dimensioni del dispositivo ed altre feature; la controindicazione è un ritardo nel caricamento delle immagini, dato che nessuna immagine viene caricata prima che lo script sia stato eseguito
 - ❖ **la selezione server side**, che implica il riconoscimento dello user-agent (il browser) e la scrittura di handler per ogni immagini; una delle controindicazioni è che non è possibile rilevare dal lato server tutte le features del browser
 - ❖ **l'uso delle CSS media queries**
 - ❖ **l'utilizzo di regole CSS e attributi HTML nuovi**
(`image-set()` , ``)

Immagini High-DPI

- ❖ Le **media queries** possono essere utilizzate per definire l'immagine di sfondo di un contenitore; attraverso il loro uso si possono stabilire un insieme di breakpoint e fare riferimento a proprietà come **device-pixel-ratio**; il limite è che non possiamo utilizzare queste regole con immagini che fanno parte del contenuto
- ❖ Un esempio di utilizzo nel quale viene utilizzata una regola di fallback per i browser che non comprendono le media queries; vengono inoltre utilizzati un set di **vendor prefixes** per permettere il riferimento a implementazioni sperimentali dei vari costruttori di browser
- ❖

```
#my-image { background: (low.png); }

@media only screen and
(min--moz-device-pixel-ratio: 1.5),
(-o-min-device-pixel-ratio: 3/2),
(-webkit-min-device-pixel-ratio: 1.5),
(min-device-pixel-ratio: 1.5)
{
    #my-image { background: (high.png); }
}
```
- ❖ Per approfondimenti sui prefissi utilizzati dai vari progettisti di browser
http://www.quirksmode.org/blog/archives/2012/07/vendor_prefixes.html

Immagini High-DPI

- ◆ Una nuova proprietà CSS `image-set` è stata proposta da Apple ed è correntemente implementata da Safari e Chrome. Al momento deve essere utilizzata con il prefisso: `-webkit-image-set`.
Nella regola CSS vengono specificate un set di immagini, ognuna accompagnata da un'indicazione di ottimizzazione per un display `n x`.
Il browser sceglierà quale immagine caricare tra il set proposto, in base ad una varietà di fattori (che in futuro potrebbero includere anche la velocità della rete). Anzichè specificare `n x` è possibile specificare la densità dei pixel del dispositivo in dpi.
- ◆ Il browser rappresenterà le immagini renderizzandole con la stessa dimensione nella pagina, scalandole adeguatamente in modo da ottenere un risultato di qualità superiore per i display ad alta densità.

Immagini High-DPI

- ◆ Un esempio di utilizzo nel quale viene utilizzata all'inizio una regola di fallback per i browser che non comprendono la proprietà `image-set`; vengono utilizzati a seguire il vendor prefix per i browser basati su WebKit e la dichiarazione per le implementazioni definitive (al momento non disponibili)
- ◆

```
background-image: url(icon1x.jpg);
background-image: -webkit-image-set(
    url(icon1x.jpg) 1x,
    url(icon2x.jpg) 2x );
/* le dichiarazioni successive serviranno per la
versione finale delle implementazioni */
background-image: image-set(
    url(icon1x.jpg) 1x,
    url(icon2x.jpg) 2x
);
```

Immagini High-DPI

- ◆ Per l'attributo html **srcset** non ci sono ancora implementazioni di riferimento. La sintassi prevede di definire per ogni immagine un'indicazione di ottimizzazione per un display **nx**, seguita da una dichiarazione dei valori di larghezza e di altezza della viewport.
- ◆ Nell'esempio sottostante il browser utilizzerà l'immagine banner-phone.jpeg per i dispositivi con viewport sotto i 640px, banner-phone-HD.jpeg per i dispositivi sotto i 640px con schermo ad alta densità, banner-HD.jpeg per i dispositivi sopra i 640px con schermo ad alta densità, banner.jpeg a tutti gli altri dispositivi.
- ◆

```

```

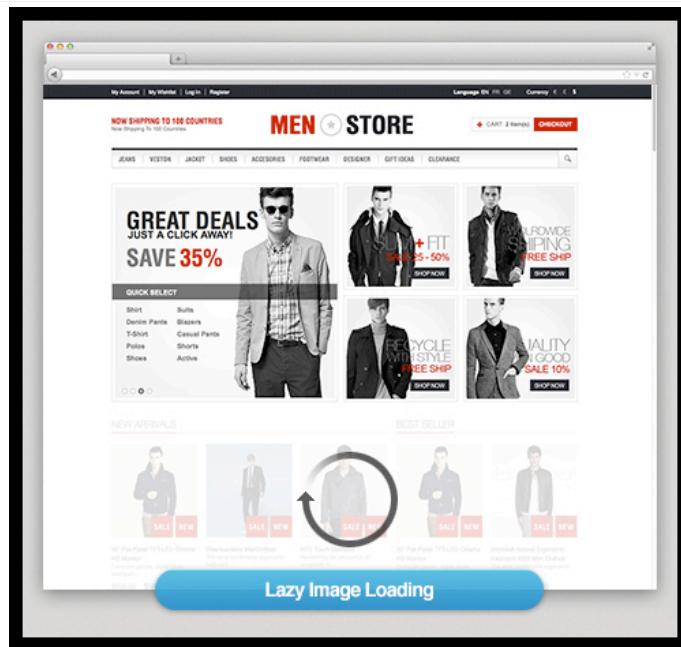
Immagini High-DPI

- ◆ Dal momento che l'attributo srcset non è ancora implementato nella maggior parte dei browser, si potrebbe essere tentati di sostituire le immagini con tag div e utilizzare l'approccio basato sulla proprietà **-webkit-image-set** CSS per lo sfondo. In questo modo si perde tuttavia il valore semantico legato al tag, importante per l'indicizzazione da parte dei motori di ricerca e l'accessibilità.
- ◆

```
<div id="my-content-image"
      style="content: -webkit-image-set(
      url(icon1x.jpg) 1x,
      url(icon2x.jpg) 2x);">
</div>
```

Immagini High-DPI

- ❖ Non c'è nessuna soluzione magica per risolvere il problema delle immagini HiDPI.
- ❖ Le soluzioni basate su Javascript, CSS e tecniche server-side hanno punti di forza e di debolezza.
- ❖ **L'approccio migliore è quello di utilizzare le nuove regole CSS e i nuovi marcatori**, fornendo adeguati fallbacks in modo che anche i browser meno recenti possano perlomeno visualizzare un'immagine.
- ❖ Concludendo, al momento le raccomandazioni sono le seguenti:
 - ❖ **per le immagini di sfondo** utilizzare la regola CSS image-set con i fallback appropriati;
 - ❖ **per le immagini che fanno parte del contenuto** utilizzare (verificata la disponibilità delle implementazioni) l'attributo HTML srcset con i fallback appropriati;
 - ❖ **per le situazioni in cui si è disposti a sacrificare la qualità ma si vuole evitare l'overhead di creare e gestire versioni multiple di una stessa immagine**, utilizzare immagini 2X fortemente compresse



Conditional and lazy loading

Conditional loading

- ◆ Le tecniche di caricamento condizionale (**conditional loading**) permettono di caricare in maniera selettiva sulla pagina parti di contenuto a seconda delle caratteristiche del dispositivo e delle features del browser dell'utente.
- ◆ Le tecniche di conditional loading utilizzano tipicamente Javascript per ottenere il risultato.
- ◆ Nelle pagine seguenti viene illustrato una di queste tecniche (<http://24ways.org/2011/conditional-loading-for-responsive-designs/>), che permette di visualizzare su dispositivi caratterizzati da un numero minimo di pixel sia il contenuto principale che il contenuto secondario (a colonna doppia o singola a seconda della dimensione della finestra del browser) e solo il contenuto principale nel caso di dispositivi con un numero di pixel più piccolo. In quest'ultimo caso però viene fornita la possibilità di visualizzare il contenuto (una selezione di news da Google News) attraverso un link.

Conditional loading

- ◆ questo è il risultato di visualizzazione per un dispositivo desktop con un browser con una finestra > 640 pixel

The screenshot shows a web browser window titled "Can't Hug Every Cat". The address bar displays the URL "media.24ways.org/2011/keith/cats--2.html". The browser interface includes standard buttons for back, forward, and search, along with links for "Add to Delicious", "Articles", "Botafumeiro - Wikipedia", "Elettrotreni... duct Details", "AKG K 271 ... Room Audio", and "Reader View". Below the toolbar, there are tabs labeled "Creat...", "Respo...", "Can't...", "Condit...", "Journa...", "Journa...", "CSS m...", and "more".

The main content area features a large title:

**A Metaphysical Reflection
Upon The Nature Of
Multitudinous Feline
Affection**

By Debbie the Online Dater

I love cats.
I love every kind of cat.
I just wanna hug all of them but I can't;
Can't hug every cat,
Can't hug every cat.

So anyway I am a cat lover
And I love to run.
I'm sorry I'm thinking about cats again.
I really love cats.

I'm thinking about cats again
And again, and again and again and again.
I think about how many don't have a home
And how I should have them.

More about cats...

Marica Pellegrinelli,
cats passion

La bella modella 24enne nata a Bergamo ha svelato la sua 'cats passion' via social. Su Twitter ha pubblicato un suo scatto in cui in primo piano c'è immortalata con il volto acqua e sapone mentre strige a sé due gatti (per ingrandire la foto clicca qui).

E-Cat: girano i primi falsi della fusione fredda di Rossi?

E-Cat. La fusione fredda di Andrea Rossi sembra non avere pace. Sul web circola la notizia che un utente, tal Antonio Marchionni, ha dichiarato di aver già

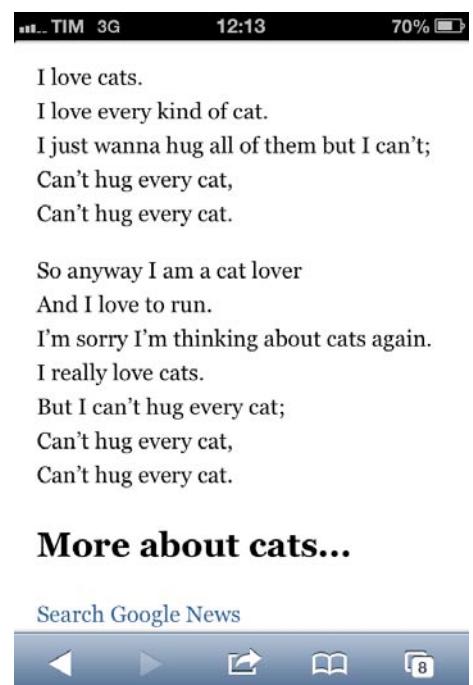
Conditional loading

- ◆ Questo è il risultato di visualizzazione della stessa pagina sullo stesso dispositivo desktop, ma con la finestra del browser rimpicciolita < 640 pixel



Conditional loading

- ◆ Questo è il risultato di visualizzazione della stessa pagina su un dispositivo con uno schermo piccolo < 640 pixel
- ◆ Si noti quindi che la visualizzazione su una finestra < 640 pixel su uno schermo grande (>640 pixel) produrrà un effetto diverso dalla visualizzazione su uno schermo piccolo (< o uguale a 640 pixel): nel primo caso verranno mostrate le news per esteso, nel secondo il link al servizio news.



Conditional loading

- ◆ questa è la prima parte del file html, in cui vengono dichiarate le regole CSS di presentazione *responsive* dei contenuti ed il contenuto principale `<div role="main">`

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  <meta charset="utf-8">
5  <meta name="viewport" content="width=device-width, initial-scale=1.0">
6  <title>Can't Hug Every Cat</title>
7  <style>
8  body {
9      font: 1em/1.5 Cambria, Georgia, serif;
10     margin: 0 5%;
11 }
12 a {
13     color: #369;
14     text-decoration: none;
15 }
16 @media all and (min-width: 640px) {
17     [role="main"] {
18         width: 60%;
19         float: left;
20     }
21     [role="complementary"] {
22         width: 30%;
23         float: right;
24     }
25 }
26 </style>
27 </head>
28 <body>
29
30 <div role="main">
31
32 <h1>A Metaphysical Reflection Upon The Nature Of Multitudinous Feline Affection</h1>
33 <footer>By <a href="http://www.youtube.com/watch?v=sP4NMoJcFd4">Debbie the Online Dater</a></footer>
34
35 <p>I love cats.<br>
36 I love every kind of cat.<br>
37 I just wanna hug all of them but I can't;<br>
38 Can't hug every cat,<br>
39 Can't hug every cat.</p>
40
41 <p>So anyway I am a cat lover<br>
42 And I love to run.<br>
43 I'm sorry I'm thinking about cats again.<br>
44 I really love cats.</p>
45
```

Conditional loading

- ◆ questa è la seconda parte del file html, in cui viene dichiarato il contenuto secondario `<div id="newsresults">` e lo script che sostituisce all'interno di esso il link a Google News con i contenuti esplicativi, nel caso in cui la grandezza dello schermo del dispositivo sia > 640 pixel.

```
81 <div role="complementary">
82
83 <aside>
84 <h1>More about cats...</h1>
85 <div id="newsresults">
86 <a href="http://www.google.com/search?q=cats&tbm=nws">Search Google News</a>
87 </div>
88 </aside>
89
90 </div><!-- /@complementary -->
91
92 <script>
93
94 var searchNews = function(searchterm) {
95     var elem = document.createElement('script');
96     elem.src = 'http://ajax.googleapis.com/ajax/services/search/news?v=1.0&q=' + searchterm + '&callback=displayNews';
97     document.getElementsByTagName('head')[0].appendChild(elem);
98 };
99
100 var displayNews = function(news) {
101     var html = '',
102         items = newsresponseData.results,
103         total = items.length;
104     if (total>0) {
105         for (var i=0; i<total; i++) {
106             var item = items[i];
107             html+= '<article>';
108             html+= '<a href="'+item.unescapedUrl+'>';
109             html+= '<h1>' + item.titleNoFormatting + '</h1>';
110             html+= '</a>';
111             html+= '<p>';
112             html+= item.content;
113             html+= '</p>';
114             html+= '</article>';
115         }
116         document.getElementById('newsresults').innerHTML = html;
117     }
118 };
119
120 if (document.documentElement.clientWidth > 640) {
121     searchNews('cats');
122 }
123 </script>
```

Conditional loading

- ◆ All'atto del caricamento della pagina viene verificata la larghezza del dispositivo:

```
if (document.documentElement.clientWidth > 640)
{searchNews('cats');}
```
- ◆ se la grandezza è maggiore di 640, allora viene invocata la funzione **searchNews** che prende il risultato di una richiesta Ajax asincrona al sito di Google News e lo inserisce come contenuto html del contenitore div che ha come id newsresults.

Lazy loading

- ◆ Il **lazy loading** si distingue dal conditional loading perché anzichè omettere il caricamento di parti del contenuto in maniera definitiva, ne ritarda il caricamento fino al momento in cui effettivamente serve.
- ◆ Ad esempio il caricamento potrà avvenire nel momento in cui l'utente effettuando lo scrolling della pagina manifesterà l'intenzione di proseguire l'esplorazione dei contenuti.
- ◆ In questo modo verrà caricato effettivamente solo il contenuto che serve, risparmiando quantità anche considerevoli di traffico dati.

Responsive tables

Responsive tables

- ◆ Le tabelle di dati costituiscono una delle entità meno facilmente adattabili nel contesto di un design responsive, dal momento che la rappresentazione dei dati è fortemente vincolata dalla struttura bidimensionale della tabella.



Responsive tables

- ◆ Una soluzione possibile (vedi <http://css-tricks.com/responsive-data-tables/>) è quella di utilizzare regole CSS per linearizzare le strutture tabellari, ottenendo il risultato visibile a fianco



Responsive tables

- ◆ Il codice html descrive una tabella standard.
- ◆ Nota: nell'immagine a fianco vengono solo rappresentate 3 colonne anzichè le 10 dell'esempio completo.

```
HTML


| First Name | Last Name | Job Title            |
|------------|-----------|----------------------|
| James      | Matman    | Chief Sandwich Eater |
| The        | Tick      | Crimefighter Sorta   |


```

Responsive tables

- ◆ Le prime regole CSS descrivono i consueti parametri che riguardano la grandezza, i bordi, il colore di sfondo e la caratteristica del testo.
- ◆ Si noti il selettore **tr:nth-of-type(odd)** che permette di stabilire un colore differenziato per lo sfondo delle righe dispari.

```
/*
Generic Styling, for Desktops/Laptops
*/
table {
    width: 100%;
    border-collapse: collapse;
}
/* Zebra striping */
tr:nth-of-type(odd) {
    background: #eee;
}
th {
    background: #333;
    color: white;
    font-weight: bold;
}
td, th {
    padding: 6px;
    border: 1px solid #ccc;
    text-align: left;
}
```

Responsive tables

- ◆ La seconda parte delle regole CSS verrà applicata nel caso in cui la finestra del browser venga ristretta sotto il limite dei 760 punti oppure quando la larghezza del dispositivo è compresa tra 768 e 1024 (ad es. un iPad).
- ◆ L'intestazione della tabella (`thead`) viene tolta dal flusso e posizionata fuori vista
- ◆ Le celle della tabella vengono considerati come **block-level** (e quindi vengono posizionati in verticale); viene assegnato un **padding ampio sulla sinistra**, in modo da **riservare spazio a** un gruppo di **etichette** che descrivono i dati; queste etichette vengono inserite grazie alla pseudoclasse **:before**

```
@media
only screen and (max-width: 760px),
(min-device-width: 768px) and (max-device-width: 1024px) {

    /* Force table to not be like tables anymore */
    table, thead, tbody, th, td, tr {
        display: block;
    }

    /* Hide table headers (but not display: none;, for accessibility) */
    thead tr {
        position: absolute;
        top: -9999px;
        left: -9999px;
    }

    tr { border: 1px solid #ccc; }

    td {
        /* Behave like a "row" */
        border: none;
        border-bottom: 1px solid #eee;
        position: relative;
        padding-left: 50%;
    }

    td:before {
        /* Now like a table header */
        position: absolute;
        /* Top/left values mimic padding */
        top: 6px;
        left: 6px;
        width: 45%;
        padding-right: 10px;
        white-space: nowrap;
    }
}
```

Responsive tables

```
/*
Label the data
*/
td:nth-of-type(1):before { content: "First Name"; }
td:nth-of-type(2):before { content: "Last Name"; }
td:nth-of-type(3):before { content: "Job Title"; }
td:nth-of-type(4):before { content: "Favorite Color"; }
td:nth-of-type(5):before { content: "Wars of Trek?"; }
td:nth-of-type(6):before { content: "Porn Name"; }
td:nth-of-type(7):before { content: "Date of Birth"; }
td:nth-of-type(8):before { content: "Dream Vacation City"; }
td:nth-of-type(9):before { content: "GPA"; }
td:nth-of-type(10):before { content: "Arbitrary Data"; }
```

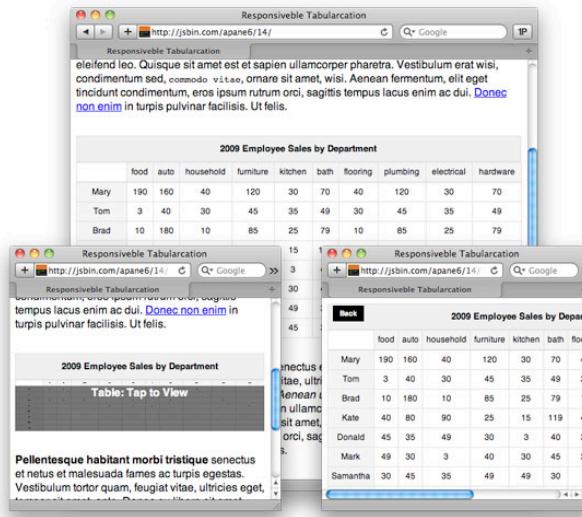
- ◆ I selettori CSS visualizzati qui sopra permettono di anteporre un testo descrittivo ai contenuti di ogni cella, specificando in maniera puntuale a quale cella della riga ci si riferisce

Responsive tables

- ◆ Purtroppo IE non comprendono le regole di stile che permettono di linearizzare la tabella e renderizzano male l'output. Al momento l'unica soluzione identificata consiste nel dichiarare queste regole all'interno di commenti condizionali.

```
<!--[if !IE]><!-->
<style>
/* table-related media query stuff only */
</style>
/* Or an external stylesheet or whatever */
<!--<![endif]-->
```

Responsive tables



◆ <http://jsbin.com/apane6/14>

Responsive tables

The image shows two desktop browser windows side-by-side, both displaying the same responsive table. The table has columns for Index, Value, Change, Change %, Year High, Year Low, Daily Low, Daily High, and Turnover €(Mil.). The left window shows the full table with all columns clearly visible. The right window shows a compressed view where some columns are hidden, and the table's width is reduced to fit the available space.

Index	Value	Change	Change %	Year High	Year Low	Daily Low	Daily High	Turnover €(Mil.)
ISEQ® Overall	2,725.99	-15.30	-0.56%	3,037.89	2,333.35	2,712.84	2,743.31	24.00
ISEQ® Financial	130.77	-3.24	-2.42%	493.83	101.54	130.43	136.14	2.76
ISEQ® General	3,751.79	-17.49	-0.46%	4,146.84	3,188.68	3,731.15	3,770.88	21.24
ISEQ® Small Cap.	1,661.94	3.76	0.23%	2,175.60	1,633.21	1,643.92	1,661.94	0.20

◆ <http://www.irishstu.com/stublog/wp-content/uploads/2011/12/table-childs.html>

Responsive tables

when the browser is resized to 640 px the table will respond and change its layout

Employee No.	User Name	Real Name	Age	Profession	Contact
2311	Lorem	Ipsum dolor	22	Web designer	1234567890
4542	Viverra	Nam non viverra	22	Web developer	2345678901
5643	Sed	Donec sed volutpat	25	Editor	3456789012
5484	Orci	Phasellus eu orci nunc	32	User Interface	4567890123

when the browser is resized to 640 px the table will respond and change its layout

Employee No.
2311

User Name
Lorem

Real Name
Ipsum dolor

Age
22

Profession
Web designer

◆ <http://www.mobifreaks.com/wp-content/demos/Responsive-and-SEO-Friendly-Data-Tables/>

Responsive tables

2009 Employee Sales by Department

	food	auto	household	furniture	kitchen	bath	flooring	plumbing	electrical	hardware
Mary	190	160	40	120	30	70	40	120	30	70
Tom	3	40	30	45	35	49	30	45	35	49
Brad	10	180	10	85	25	79	10	85	25	79
Kate	40	80	90	25	15	119	40	80	90	25
Donald	45	35	49	30	3	40	30	45	35	49
Mark	49	30	3	40	30	45	35	49	45	35
Samantha	30	45	35	49	49	30	3	40	45	35
Harold	30	30	3	40	45	35	45	35	49	49

2009 Employee Sales by Department

Department	Percentage
Mary	22.5%
Tom	15.21%
Brad	9.34%
Kate	9.34%
Donald	9.34%
Mark	9.34%
Harold	9.34%

◆ <http://jsbin.com/emexa4>

Responsive tables e JQuery Mobile

- ◆ Alcuni framework come JQueryMobile permettono di ottenere facilmente una tabella responsive, utilizzando sia tecniche di linearizzazione che di visualizzazione selettiva delle colonne che compongono la tabella.

JQuery Mobile: linearizzazione di tabelle

Rank	Movie Title	Year	Rating	Reviews
1	Citizen Kane	1941	100%	74
2	Casablanca	1942	97%	64
3	The Godfather	1972	97%	87
4	Gone with the Wind	1939	96%	87
5	Lawrence of Arabia	1962	94%	87
6	Dr. Strangelove Or How I Learned to Stop Worrying and Love the Bomb	1964	92%	74
7	The Graduate	1967	91%	122
8	The Wizard of Oz	1939	90%	72
9	Singin' in the Rain	1952	89%	85
10	Inception	2010	84%	78

Rank	1
Movie Title	Citizen Kane
Year	1941
Rating	100%
Reviews	74
Rank	2
Movie Title	Casablanca
Year	1942
Rating	97%
Reviews	64
Rank	3
Movie Title	The Godfather
Year	1972
Rating	97%

- ◆ <http://view.jquerymobile.com/1.3.2/dist/demos/widgets/table-reflow/>

JQuery Mobile: visualizzazione selettiva di colonne

Rank	Movie Title	Year	Rating	Reviews
1	Citizen Kane	1941	97%	97%
2	Casablanca	1942	96%	96%
3	The Godfather	1972	94%	94%
4	Gone with the Wind	1939	92%	92%
5	Lawrence of Arabia	1962	90%	90%
6	Dr. Strangelove Or How I Learned to Stop Worrying and Love the Bomb	1964	89%	89%
7	The Graduate	1967	88%	88%
8	The Wizard of Oz	1939	87%	87%
9	Singin' in the Rain	1952	86%	86%
10	Inception	2010	84%	84%

◆ <http://view.jquerymobile.com/1.3.2/dist/demos/widgets/table-column-toggle/>

JQuery Mobile: visualizzazione selettiva di colonne 2

Rank	Movie Title	Year	Rating	Reviews
1	Citizen Kane	1941	97%	97%
2	Casablanca	1942	96%	96%
3	The Godfather	1972	94%	94%
4	Gone with the Wind	1939	92%	92%
5	Lawrence of Arabia	1962	90%	90%
6	Dr. Strangelove Or How I Learned to Stop Worrying and Love the Bomb	1964	89%	89%
7	The Graduate	1967	88%	88%
8	The Wizard of Oz	1939	87%	87%
9	Singin' in the Rain	1952	86%	86%
10	Inception	2010	84%	84%

◆ <http://view.jquerymobile.com/1.3.2/dist/demos/widgets/table-column-toggle/>