HTML5 Mobile app: transizioni ed animazioni CSS3: parte1

Sicuramente avrai già sentito parlare di applicazioni mobile create completamente in **HTML5**, **CSS3** e **Javascript,** o magari abbinando questi tipi di linguaggi con quelli nativi, come nelle applicazioni "ibride".

Forse potrebbe interessarti l'ottimo post di Smashing Magazine *sulle applicazioni ibride:* Best Of Both Worlds: Mixing HTML5 And Native Code**.**

In questo articolo vorrei riflettere su quali strade poter percorrere per creare effetti accattivanti e controllare le animazioni in ambito mobile.

Sviluppare HTML5 mobile app con CSS3 e Javascript

Per facilitare lo sviluppo di applicazioni web sono nate numerose librerie e framework che consentono di ricreare, tramite HTML, interfacce pressoché identiche (o comunque molto simili) a quelle realizzate in maniera nativa. Tanto per citarne alcune, ecco una breve lista di mobile frameworks:

- App Framework – by Intel
- jQuery Mobile
- Sencha Touch
- APP UI
- ChocolateChip-UI
- Ratchet
- Fries
- jQ Touch

Una soluzione alternativa è quella di scrivere manualmente il codice **CSS** e **Javascript.**

Entrambe le metodologie hanno i loro pro e contro: se da una parte l'impiego dei vari **framework** rende più facile **gestire problemi comuni** e utilizzare animazioni standard o funzionalità javascript, dall'altra scrivere **personalmente** il codice offre molta più **libertà** di **personalizzazione.**

Durante le mie esperienze lavorative ho avuto la possibilità di realizzare diverse **mobile app,** tutte in **HTML5, Javascript** e **CSS,** alcune delle quali pubblicate nei vari **market stores** e altre usate come **versione mobile** di siti web (tra queste anche due progetti personali: Tint Weather App e Dieta SI o NO?).

Ognuno di quei progetti mi è servito a prendere una netta decisone: scrivere le mie applicazioni da zero.

Il motivo? **Performance** e **fluidità.**

E’ vero che i vari **framework** mettono a disposizione strumenti già pronti come liste, sliders, menu ecc., ma è anche vero che applicazioni create con tali librerie risultano il più delle volte pesanti, scattose e poco reattive.

Ma torniamo al punto.

Sia che si decida di optare per la scrittura manuale del codice che per l'utilizzo di framework appositi, la presenza di animazioni come slide-left, slide-right, pop o flip è essenziale per ricreare un'esperienza che sia **conforme** a quella di **applicazioni native.**

La metodologia con cui vengono realizzate incide notevolmente sulle prestazioni, soprattutto nel caso di dispositivi mobile, che per caratteristiche hardware sono meno potenti di quelli desktop.

Animazioni imperative ed animazioni dichiarative

Gli sviluppatori possono decidere di animare gli oggetti **DOM** tramite **Javascript** (**animazioni imperative**) e tramite **CSS** (**animazioni dichiarative**).

Animare usando Javascript permette un **controllo maggiore,** in quanto avviare, mettere in pausa, interrompere o cancellare un effetto risulta molto semplice.

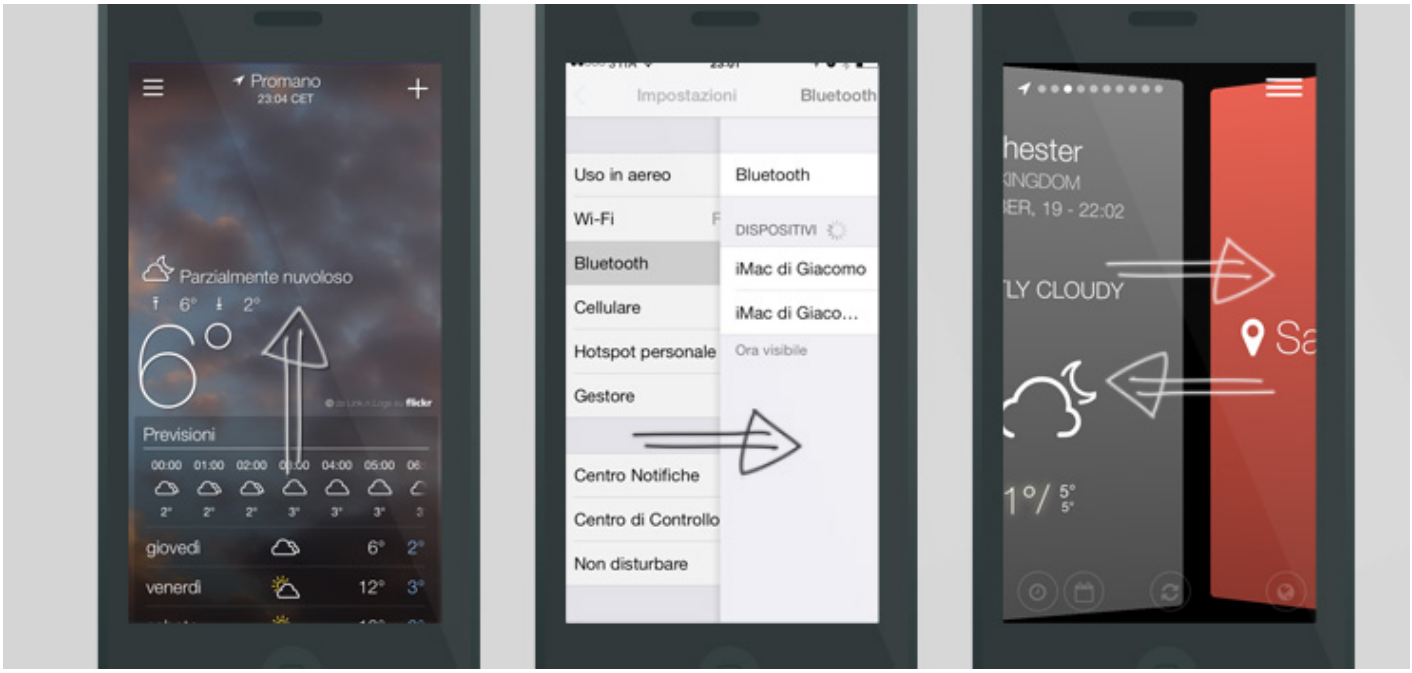
Tuttavia questo tipo di approccio presenta un **difetto** che non è possibile ignorare: le **animazioni** vengono eseguite nel **main thread** del **browser,** che è già occupato da altri javascript o dai processi di rendering degli elementi.

Per cui capita spesso che lo **spazio disponibile** nel thread venga **conteso,** aumentando di conseguenza la possibilità che alcuni frame che compongono **l'animazione** siano completamente **ignorati** (producendo fastidiosi **sfarfallii** o movimenti **scattosi).**

Animando tramite CSS3 abbiamo invece il **vantaggio** di lasciare al **browser l'ottimizzazione** delle **prestazioni;** esso infatti organizzerà da solo **l'esecuzione** dei **frames,** svolgendo inoltre parte delle operazioni all'infuori del main thread.

Contrariamente alle animazioni imperative, per quelle dichiarative il controllo risulta molto difficoltoso. Per cui diventa più difficile sincronizzare tra loro più animazioni, ed è quindi più facile commettere errori.

Sempre tenendo in considerazione il loro impiego in ambito mobile, decidiamo di puntare sulle performance, per cui la scelta ricade sulle **animazioni dichiarative.**



Alcuni esempi di animazioni facilmente riproducibili con i CSS3

Forse può interessarti questo post su come ottimizzare animazioni CSS3 per le performance.

Come controllare le animazioni CSS3

Agendo da CSS abbiamo a disposizione due tipi di proprietà: la ***transition*** e ***l'animation.*** La prima permette di applicare un **effetto graduale** su di un elemento **DOM** nel momento in cui viene modificata una **proprietà.**

La seconda invece agisce in modo diverso: il suo compito è quello di **applicare all'elemento** interessato **un'animazione precedentemente dichiarata** e **formata** anche da più di due stati. La proprietà ***animation*** offre un controllo più **dinamico,** in quanto è possibile controllarne i vari **step.**

In questa prima parte ci limiteremo a vedere le possibili strade per controllare effetti creati con la proprietà ***transition,*** mentre nella seconda parte vedremo come controllare la proprietà ***animation.***

Avviare e mettere in pausa un effetto *transition*

Per dare inizio ad una transizione basterà modificare l'attributo CSS "scatenante". Prendiamo in considerazione un'ipotetica animazione che mostra in dissolvenza la *div* **“#mydiv”**:

La proprietà scatenante in questo caso è ***l'opacity,*** e quando sarà modificata, l'effetto avrà inizio e durerà 5 secondi. Per poter avviare l'animazione basterà semplicemente aggiungere la classe **"show"** alla nostra div:

Ecco il risultato: demo

Se vogliamo invece mettere in pausa un effetto ***transition*** per poi riavviarlo in un secondo momento, dovremo usare il metodo ***getComputedStyle*** nel caso di **javascript** classico, o il metodo ***.css*** nel caso di **jQuery:**

In poche parole per l'avvio o la ripresa dell'effetto aggiungiamo la classe **"show"**, mentre per metterlo in pausa riprendiamo il valore della proprietà "scatenante" nel momento in cui vogliamo fermare l'animazione; poi rimuoviamo la classe "show" ed aggiorniamo quella proprietà con il dato appena acquisito.

Risultato: Demo

Conclusioni

Controllare la proprietà ***transition*** e ***animation*** rappresenta un vantaggio molto grande se si vuole rendere più piacevole l'esperienza **utente** dei nostri prodotti, soprattutto se il nostro intento è quello di realizzare una mobile app in HTML5, CSS3 e Javascript.

Utilizzando questa tecnica unita a molte altre ho creato per i miei clienti oltre 10 applicazioni mobile, tra cui due progetti personali: Tint Weather App e Dieta SI o NO?.

Se anche tu sei un web designer e vuoi realizzare la tua applicazione mobile da distribuire nei vari stores, continua a seguirci!

Arriveranno presto nuovi articoli di approfondimento sull'argomento, **per rendere la tua mobile app fluida e reattiva come quelle native.**