HTML5 Mobile app: transizioni e animazioni CSS3 (parte 2)

Nella prima parte di questa mini guida abbiamo parlato dei concetti di **animazioni imperative/animazioni dichiarative**, di come avviare o mettere in pausa un effetto **transition** e di come queste tecniche possano essere impiegate per la realizzazione di **interfacce web** e **HTML5 Mobile App**.

In questa seconda parte scopriremo come controllare la proprietà *animation* e come gestire, tramite Javascript, gli eventi associati a essa e alla proprietà transition, molto utili se dobbiamo eseguire degli effetti in sequenza o svolgere determinate operazioni all'inizio, durante o alla fine di un effetto.

Avviare e mettere in pausa la proprietà animation

Attivare un effetto realizzato con *animation* è un'operazione piuttosto semplice: una volta dichiarata la nostra animazione, basterà agire sulla proprietà CSS3 "***animation-name***" specificando come valore il nome dell'animazione appena creata.

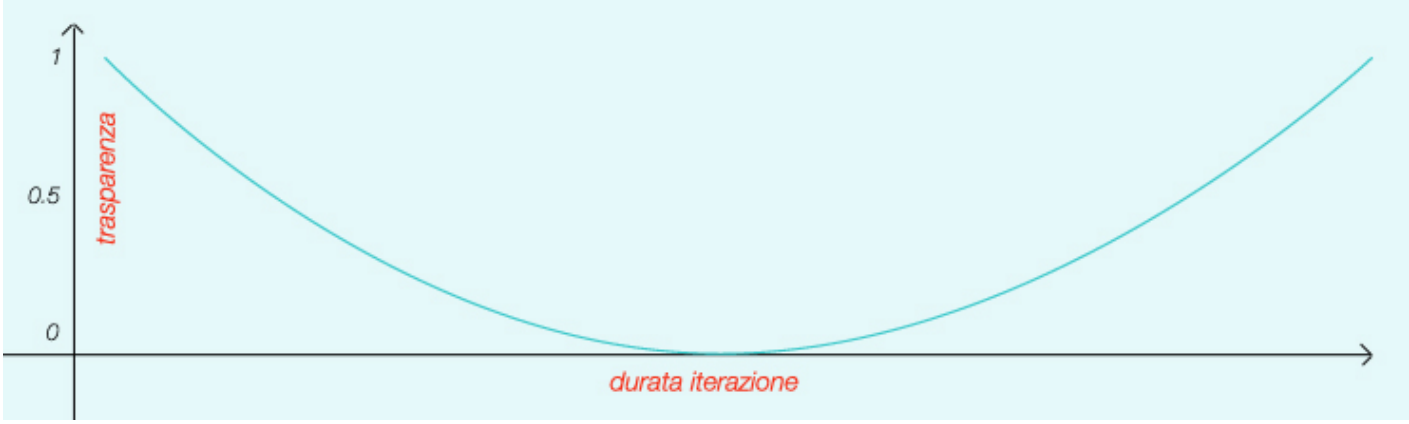
Vediamo l'esempio concreto di un ipotetico effetto "*flash*" sulla div con id "*#mydiv*".

Come prima cosa definiamo **dinamica** e il **nome** dell'effetto tramite la regola ***keyframe***, facendo attenzione nell'includere tutti i *prefix* per la compatibilità browser.

L'intento è quello di far scomparire in dissolvenza l'elemento DOM interessato quando l'animazione è giunta a metà della sua durata, per poi farlo ricomparire al termine.

Definiamo quindi gli stili dell'oggetto da animare e una classe aggiuntiva che specifica il tipo di animazione:

Nel momento in cui la classe "*animate*" viene aggiunta alla nostra
, essa scomparirà ed apparirà in dissolvenza per 20 volte, ognuna della durata di un secondo. Ogni ciclo viene chiamato ***iterazione****.*



Vediamo come attivare l'effetto tramite javascript:

Visualizza il risultato: DEMO

Per fermare un'animazione e/o riavviarla è invece necessario agire sulla proprietà ***animation-play-state***, che, come nel caso di un filmato, ne specifica lo stato ***play/pausa***. I possibili valori sono "*paused*" o "*running*": il secondo è quello predefinito.

Ci basterà quindi creare un'ulteriore classe "*paused*" e aggiungerla o rimuoverla dall'elemento al fine di fermare o riavviare l'effetto:

Codice CSS

Codice Javascript

Visualizza il risultato: DEMO

Ora che abbiamo visto come poter controllare la proprietà *animation*, vediamo come gestire gli eventi per sincronizzare tra loro diversi tipi di effetti.

La gestione degli eventi

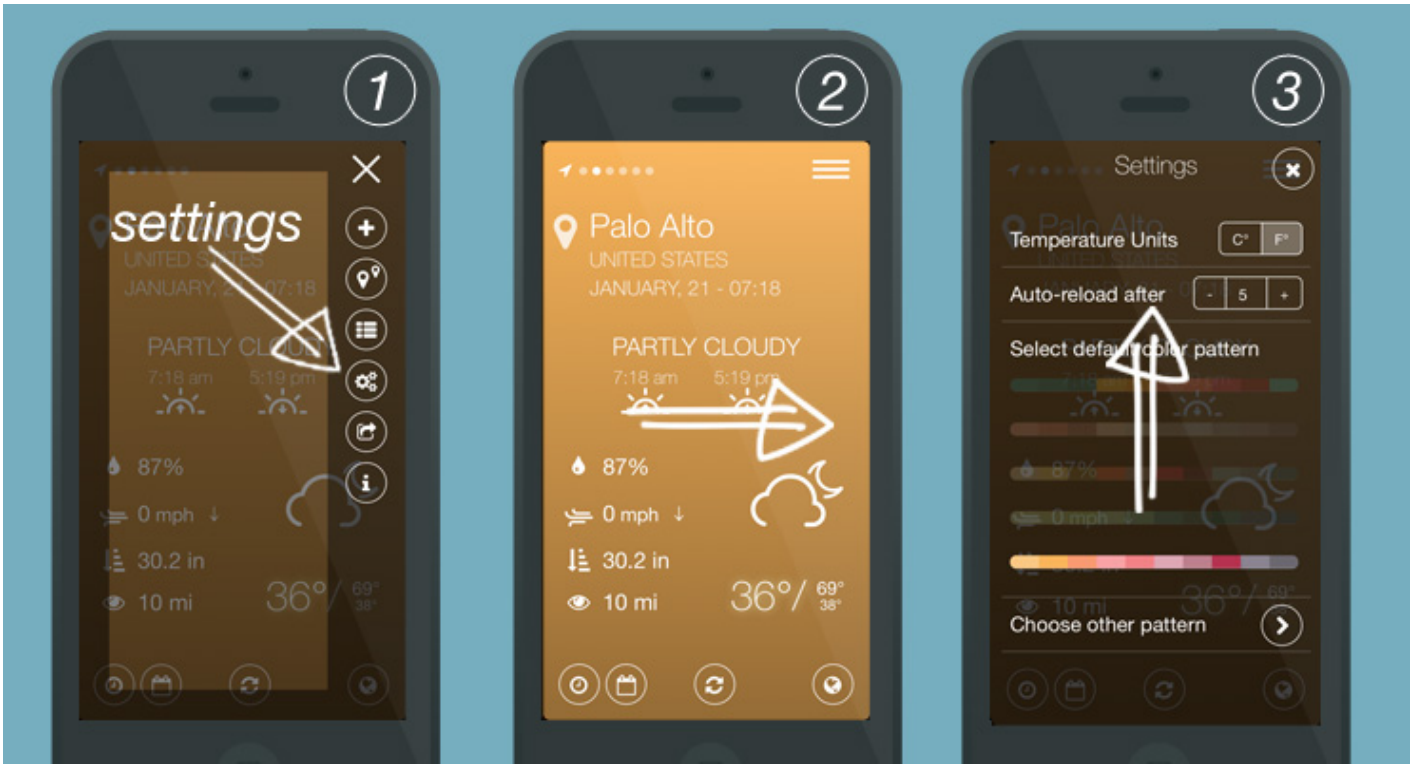
Sincronizzare in modo corretto gli effetti applicati a uno o più **elementi DOM** è indispensabile per poter realizzare delle **interfacce mobile in HTML5 e CSS3** che per performance si avvicinano il più possibile a quelle native.

La gestione degli **eventi associati** a **transition** e **animation** è il modo più corretto per raggiungere questo obiettivo.

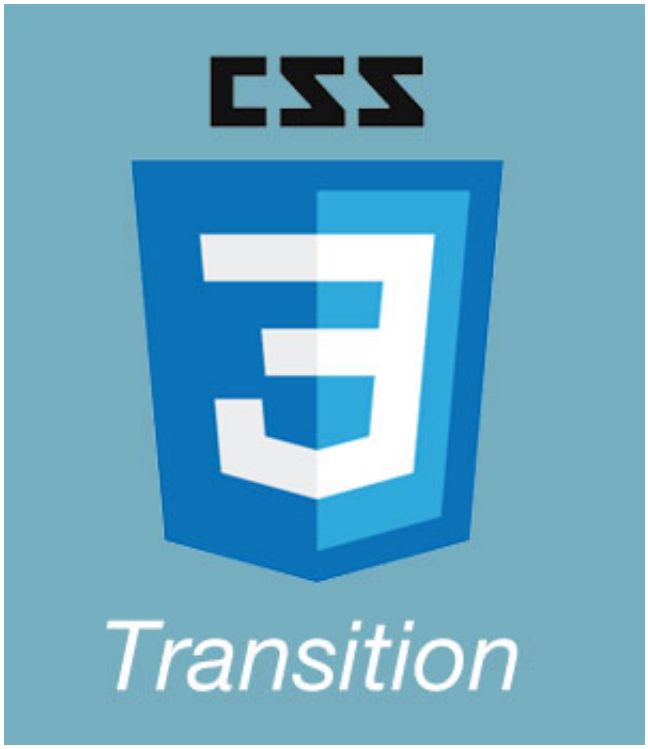
Per esempio, **eseguendo funzioni JavaScript** all'inizio, alla fine o durante un effetto, possiamo avviare un'animazione quando un'altra ha completato un'iterazione, oppure richiamare la ***transition*** su di un elemento DOM nel momento in cui è terminata quella precedente.

Prendiamo come esempio l'applicazione mobile Tint: quando l'utente sceglie la voce "*settings*" dal menu principale, viene lanciata una funzione javascript il cui compito è quello di **rimuovere la classe di stile** *"open"* dall'elemento che lo rappresenta, **attivando l'effetto CSS3 *transition* di chiusura.**

Appena quest'ultimo giunge al termine, viene lanciata una nuova funzione che aggiunge la classe "*view*" alla
utilizzata per la schermata delle impostazioni, in modo da avviare la relativa animazione di entrata.



Gestire la fine di una transizione: l'evento transitionEnd



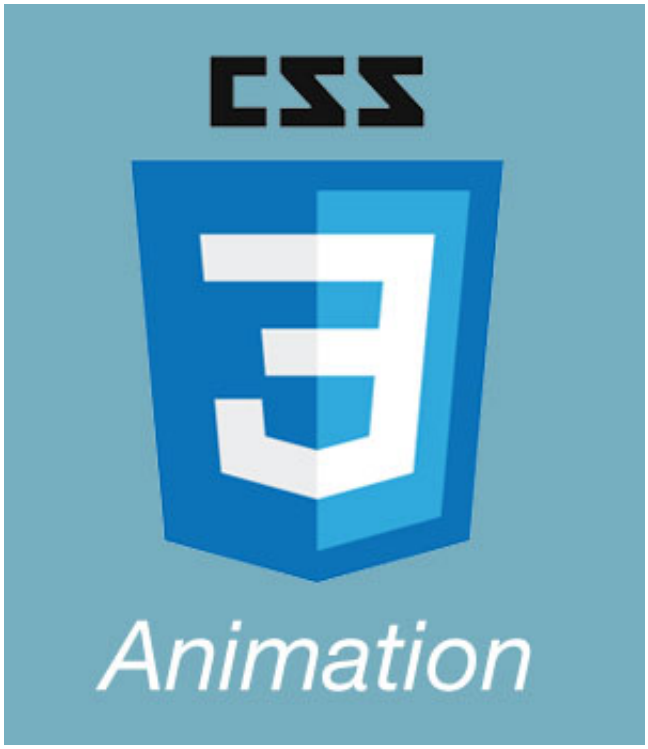
Per quanto riguarda la proprietà *transition*, l'unico evento che possiamo gestire è quello che scatta quando una transizione viene completata; esso è denominato ***transitionEnd***.

Vediamo il codice javascript per eseguire una funzione al termine di una transizione su un elemento DOM:

Alla funzione viene restituito un **oggetto *event*** che possiamo utilizzare per eseguire dei controlli in modo da capire quale tipo di transizione è appena terminata:

Ecco un esempio che dimostra la gestione dell'evento: DEMO

Gli eventi della proprietà animation



La proprietà *animation* mette a disposizione tre tipi di eventi: **animationStart**, **animationIteration** e **animationEnd**:

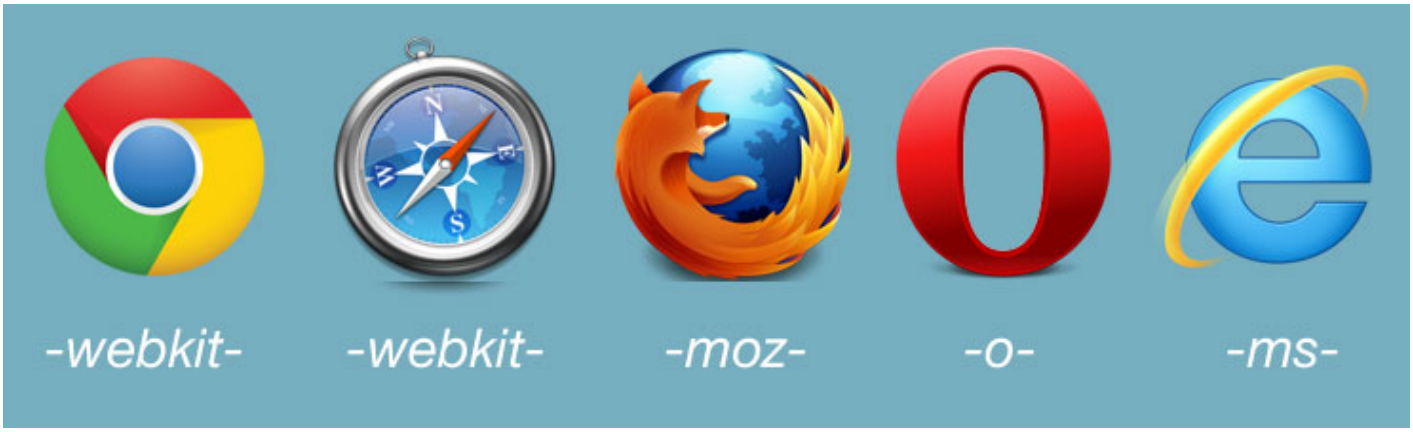
- ***animationstart***: viene generato la prima volta che un'animazione ha inizio. Se viene richiamata un'animazione specificando più **iterazioni** (come nell'esempio riportato in precedenza), allora verrà generato solo all'inizio della prima iterazione.
- ***animationiteration***: viene richiamato all'inizio di ogni iterazione, eccetto la prima. Se l'animazione prevede una sola iterazione l'evento non viene generato.
- ***animationend***: viene generato quanto l'ultima iterazione di un'animazione giunge al termine (nel caso precedente al termine della 20° iterazione).

La metodologia di utilizzo per tutti e tre gli eventi è analoga a quella dell'evento *transitionEnd*:

Ecco un esempio che mostra il funzionamento dei tre eventi: DEMO

Anche in questo caso viene restituito un **oggetto *event*** su cui possiamo effettuare dei controlli per determinare qual è il nome dell'animazione appena iniziata, terminata, o della quale è stata appena completata un'iterazione:

Come gestire i prefix per la compatibilità



Purtroppo, come per molte funzionalità CSS3, anche qui abbiamo dei problemi per quanto riguarda la compatibilità: devono essere infatti specificati dei prefissi diversi a seconda del browser che viene utilizzato.
Ecco una tabella comparativa che mostra quali sono i prefissi utilizzati dai vari browser:

W3C standard	Firefox	webkit	Opera	IE10
transitionend	transitionend	webkitTransitionEnd	oTransitionEnd	MSTransitionEnd
animationstart	animationstart	webkitAnimationStart	oAnimationStart	MSAnimationStart
animationiteration	animationiteration	webkitAnimationIteration	oAnimationIteration	MSAnimationIteration
animationend	animationend	webkitAnimationEnd	oAnimationEnd	MSAnimationEnd

NOTA: Se si sta sviluppando una HTML5 Mobile App potrebbe bastare, oltre a quello standard, il solo l'utilizzo del prefix -webkit-. Tuttavia risulterebbe in tutti casi scomodo il fatto di dover associare la stessa funzione a più di un tipo di evento, come nel codice seguente:

Per fortuna ci sono numerose tecniche per gestire più velocemente tali eventi e scrivere meno righe di codice; uno di quelli più efficaci è sicuramente l'utilizzo di Modernizr (molto probabilmente avrai già notato Modernizr nelle demo che ti ho appena mostrato).

Modernizr è una **libreria** nata per rilevare le funzionalità HTML5 e CSS3 compatibili con il browser dell'utente. Fra le molteplici funzioni di utilità che mette a disposizione possiamo trovare il metodo **.prefixed(str)**.

Passando come parametro il nome della funzionalità che ci interessa sottoforma di stringa, il metodo restituisce come parametro il nome della variante di quella funzionalità con il prefisso corretto.

Per poter associare una funzione javascript ad un evento *transition* e *animation* senza preoccuparci dei prefix di ogni browser, dovremo in primo luogo definire i nomi degli eventi:

Poi, ogni volta che abbiamo bisogno di associare delle funzioni apposite basterà proseguire cosi facendo:

In questo modo potremo "**centralizzare**" la definizione del nome corretto di ogni evento in una variabile da utilizzare quando ne abbiamo bisogno.

Forse ti potrebbe interessare un tutorial in cui mi sono servito di Modernizr per gestire l'evento transitionEnd legato all'animazione di chiusura di un mobile slide menu:

- Come realizzare un HTML Mobile Slide Menu con Animazioni CSS3

Conclusioni

Transition e *animation* rappresentano uno strumento molto utile per poter realizzare **applicazioni con interfacce fluide e performanti**, anche dovendo lavorare su markup HTML abbastanza complessi.

Grazie ad esse ho sviluppato diversi progetti per dispositivi mobile e posso affermare che le prestazioni raggiunte si avvicinano molto a quelle delle applicazioni native.

Se sei un web designer e vorresti realizzare **una tua HTML5 Mobile App**, forse potrebbe interessarti l'**E-book** che sto scrivendo: HTML Mobile Accelerato**.**