Guida HTML5: i tag audio e video - parte 2

Nello scorso articolo abbiamo parlato degli enormi progressi che ha compiuto l'HTML5 per consentire un **maggiore supporto** agli **elementi multimediali** e abbiamo introdotto il tag , dando spazio anche ad alcune considerazioni sui formati e sulla compatibilità che ti invito a rileggere, data l'attinenza con questo capitolo dedicato all'elemento .

Una volta ripassata la scorsa lezione, siamo pronti ad iniziare!

Video

Il **tag** dedito a gestire i **video** è molto simile a quello che abbiamo utilizzato per i file **audio.** Iniziamo subito con un esempio pratico, inserendo un filmato nella nostra pagina:

Il quale apparirà, in Google Chrome, così:

Anche gli attributi legati al tag sono, in parte, gli stessi usati con i tag :

- ***controls*** - indica al browser di visualizzare o meno il player con i relativi comandi predefiniti
- ***loop*** - riproduce in modo continuativo il video
- ***autoplay*** - riproduce il file automaticamente senza bisogno d'interazione da parte dell'utente
- ***preload*** - configurabile con i tre valori *auto, metadata* e *none.* Il primo inizia il prima possibile il preload, il secondo carica immediatamente solo i dati estraibili dal file, come la durata totale, e il terzo inibisce il preload fino al click dell'utente.

A questi si aggiungono *height* e *width* che, come possiamo intuire, specificano rispettivamente l'altezza e la larghezza, espresse in pixel, della finestra video.

Generalmente questi valori dovrebbero corrispondere proporzionalmente all'altezza e alla larghezza naturale del video, ma in alcuni casi potrebbe essere conveniente specificarli per evitare fastidiosi errori di posizionamento nel layout.

Per finire, *poster,* ci permette di **specificare l'immagine d'anteprima** che verrà mostrata prima dell'avvio del filmato. Questo attributo verrà utilizzato soprattutto dai browser nelle seguenti situazioni: quando abbiamo **disabilitato** il ***preload*** (quindi settato su *none),* quando il **primo frame** ancora **non è stato caricato** oppure quando la sorgente video non è stata trovata.

Il codice, in pratica, risulterà:

Proprio come con l'elemento , anche in questo caso, per **motivi di compatibilità** tra dispositivi e browser, necessitiamo di usare almeno due formati differenti per coprire la maggior parte dei possibili fruitori del sito (vedi sezione 'formati e compatibilità' del capitolo precedente).

L'elemento ci permette, anche stavolta, di specificare diverse sorgenti:

Nasce, a questo punto, spontanea la riflessione: quante sorgenti dobbiamo specificare per ottenere un livello accettabile di compatibilità?

Diciamo che per ritenerci parzialmente soddisfatti dovremmo utilizzare, come nell'esempio,

un **mp4** affiancato da un **ogv** (in alternativa puoi usare anche il **webm** in sostituzione **dell'ogv** tenendo conto, però, che non risulterà compatibile con alcune vecchie versioni di Firefox e Opera) e da un cosiddetto **flash fallback.**

Fallback

Letteralmente il termine inglese fallback vuol dire **ripiego, scorta** ed è con questo vocabolo che si indica la **porzione di codice** a cui i **browser** che non supportano l'HTML5, **fanno riferimento.** Nel nostro caso, continuando a parlare del tag **,** potrebbe essere utile inserire un messaggio che avvisi l'utente di aggiornare il proprio browser.

Analizzando questo codice, capiamo che i browser che supportano il tag adopereranno i file **mp4** e **ogv** contrariamente a chi, non riuscendo a interpretare il tag, visualizzerà il messaggio racchiuso tra il tag

.

Esistono anche altri tipi di fallback, decisamente più complessi ed efficaci, come quello in **Flash.** Proprio come poco fa, anche in questo caso solo i browser che non riescono a caricare il player HTML5 faranno affidamento su un **player alternativo** in Flash.

Ad esempio utilizzando questo codice:

Non abbiamo fatto altro che inserire, sempre all'interno del tag , il tag che andrà a caricare la **versione swf** del nostro video qualora i tag .

Infine, potremmo voler considerare anche la **fascia di utenti** che non possono usufruire né del video in **HTML5** né di **Flash,** aggiungendo al **fallback** un immagine ***poster,*** un messaggio e dei link che rimandano al download del filmato.

All'interno del tag , quindi, andremo a scrivere:

Conclusione

Ti ricordo che puoi scaricare l'archivio zip di questa lezione contenente il codice completo.

Nel corso del prossimo articolo vedremo come arricchire, modificare e controllare, grazie ad alcune righe di JavaScript, il nostro player in HTML5.

E tu? Cosa ne pensi di questa nuova modalità per riprodurre elementi multimediali? Quali pensi siano le migliorie più incisive rispetto a Flash?

GUIDA HTML5: GLI ARTICOLI

1) Guida HTML5:Introduzione
2) Guida HTML5: la prima pagina
3) Guida HTML5: la struttura
4) Guida HTML5: Immagini e outlines
5) Guida HTML5: nuovi elementi semantici
6) Guida HTML5: i form – Parte 1
7) Guida HTML5: i form – Parte 2
8) Guida HTML5: i form – Parte 3
9) Guida HTML5: i form – Parte 4
10) Guida HTML5:i tag audio e video – parte 1
11) Guida HTML5:i tag audio e video – parte 2
12) Guida HTML5: I player video
13) HTML5: Il Canvas – Parte 1
14) Guida HTML5: Il Canvas - Parte 2
15) HTML5: Il Canvas - Parte 3
16) HTML5: Il Canvas - Parte 4
17) HTML5: Web storage