



Interfacce a tab con CSS e Javascript

Le tab sono uno degli elementi di navigazione più comuni. Abbiamo esplorato alcune possibilità di personalizzazione con i CSS in diversi articoli: a partire dalle tab minimali e senza grafica di Un menu a tab con i CSS (<http://css.html.it/articoli/leggi/418/un-menu-a-tabs-con-i-css/>), passando alle Tab grafiche con i CSS (<http://css.html.it/articoli/leggi/1738/menu-con-tab-grafiche/>) fino alle Tab centrate con i CSS (<http://css.html.it/articoli/leggi/2248/tab-grafiche-centrate/>).

Le tab sono comuni anche nelle interfacce web e sono usate estensivamente in diversi siti (ad esempio qui su HTML.it (<http://www.html.it/>) o su Yahoo! (<http://it.yahoo.com/>)) per ottenere informazioni facilmente consultabili in uno spazio ristretto. Vedremo in questo appuntamento una possibile implementazione di interfacce a tab con un pizzico di Javascript, partendo dal markup necessario fino ad arrivare alla personalizzazione con grafica e CSS. Ecco una piccola anteprima dell'esempio (<http://www.html.it/articoli/2447/tabEsempio.html>) e il suo screenshot:

Figura 1 - Screenshot dell'esempio



La scelta dello script

Per realizzare una soluzione come quella dell'esempio (<http://www.html.it/articoli/2447/tabEsempio.html>) una delle prime cose da fare è la scelta dello script. Le soluzioni a questo punto sono diverse:

1. Disponendo di sufficienti conoscenze di Javascript, si può pensare di realizzare lo script a partire da zero
2. Ci si può affidare ad uno dei tanti plugin dei principali framework: JQuery (<http://jquery.com/>), MooTools (<http://mootools.net>) o altri
3. Si può usare uno dei tanti script già pronti disponibili in rete che non abbiano dipendenze da librerie

In quanto ai plugin, rimando agli articoli Tab semplici e accessibili con JQuery (<http://javascript.html.it/articoli/leggi/2255/tab-semplici-e-accessibili-con-jquery/>) e Un'interfaccia a tab con il plugin JQuery.tabs (<http://javascript.html.it/articoli/leggi/2301/uninterfaccia-a-tab-con-il-plugin-jquerytabs/>) nella sezione Javascript.

Per questo articolo ho optato invece per una soluzione non basata su librerie che sia facilmente pronta all'uso. Per mettere in pratica in casi reali una soluzione come quella dell'esempio (<http://www.html.it/articoli/2447/tabEsempio.html>) che accompagna l'articolo non saranno necessarie conoscenze di Javascript. Vedremo poi come, una volta capito come impostare il markup, la personalizzazione del CSS sia piuttosto semplice e consenta molta libertà.

Tra i molti script disponibili in rete ho scelto Yetii (<http://www.kminek.pl/lab/yetii/>), uno script semplice e leggero (circa 2kb) con diverse funzionalità interessanti: è possibile scegliere la tab inizialmente aperta e anche avere tab temporizzate. Da evidenziare che lo script è totalmente non intrusivo, dato che non necessita di aggiunta di eventi in linea nel markup e gli esempi degradano bene con Javascript disabilitato.

Nota bene: lo script nella versione originale ha problemi di compatibilità con IE 5.0 e IE5.5: con una piccola modifica, ho esteso la compatibilità anche a questi due browser.

Il markup

Il markup di base per lo script ha un formato piuttosto standard, ma consente comunque una buona flessibilità. Per prima cosa sarà necessario un contenitore generico con id specificato che contenga il tutto. Nel caso dell'esempio (<http://www.html.it/articoli/2447/tabEsempio.html>), questo id è **tabcont1**. La lista di navigazione a tab dovrà contenere lo stesso id, a cui verrà aggiunto il suffisso **"-nav"**: per l'esempio sarà quindi **tabcont1-nav**. I vari pannelli dovranno avere infine la classe **"tab"**.

Per comodità di personalizzazione con i CSS e soprattutto per poter consentire diverse tab nella stessa pagina, ho aggiunto le classi *tabpanel* al contenitore generico e *tabnav* alla navigazione. Se vorrete adattare l'esempio alle vostre pagine si tratterà quindi di:

- assegnare un id a piacere al contenitore generale e la classe "tabpanel";
- assegnare un id (ottenuto aggiungendo "-nav" all'id del punto uno) alla lista di navigazione e la classe "tabnav";
- assegnare la classe "tab" a ciascun pannello, oltre agli id necessari affinché vengano usati nella navigazione come ancore di pagina.

Questi i passi necessari per la struttura generale dell'esempio (<http://www.html.it/articoli/2447/tabEsempio.html>), di cui riporto la parte essenziale di codice HTML:

```
<div id="tabcont1" class="tabpanel">
<ul id="tabcont1-nav" class="tabnav">
<!-- contenuto lista di navigazione -->
</ul>
<div id="tab1" class="tab">
<!-- contenuto primo pannello --> </div>
<div id="tab2" class="tab">
<!-- contenuto secondo pannello -->
</div>
</div>
```

Aggiungere lo script

Una volta stabilite una o più interfacce a tab nella pagina, aggiungere lo script è piuttosto semplice. Per prima cosa andrà linkato nella sezione `head` di pagina:

```
<script type="text/javascript" src="yetii.js"></script>
```

L'altra aggiunta da fare è per attivare lo script, inserendo un paio di righe prima della chiusura dell'elemento `body` a fondo pagina:

```
<!-- ... fine pagina -->
<script type="text/javascript">
var tabber=new Yetii('tabcont1');
tabber.init();
</script>
</body>
```

In grassetto noterete che è stato indicato l'id dell'elemento che contiene l'interfaccia a tab: è questa l'unica modifica che dovrete apportare nelle righe appena viste per adattare lo script alle vostre pagine.

Il CSS dell'esempio

Visti brevemente il markup e il Javascript, non ci resta che procedere con la parte centrale dell'esempio (<http://www.html.it/articoli/2447/tabEsempio.html>), ovvero le tab. Per buona parte mi sono basato sul lavoro risultante dall'articolo Menu con tab grafiche (<http://css.html.it/articoli/leggi/1738/menu-con-tab-grafiche/>). Da notare che per aggiungere flessibilità alle tab, sono necessari nel markup elementi `span` all'interno dei link:

```
<ul id="tabcont1-nav" class="tabnav">
<li><a href="#tab1"><span>News</span></a></li>
<li><a href="#tab2"><span>Articoli</span></a></li>
<li><a href="#tab3"><span>Video</span></a></li>
<li><a href="#tab4"><span>Download</span></a></li>
</ul>
```

Per quanto riguarda il CSS dell'esempio (<http://www.html.it/articoli/2447/tabEsempio.html>), rimando all'articolo sulle tab grafiche (<http://css.html.it/articoli/leggi/1738/menu-con-tab-grafiche/>) per i dettagli implementativi. C'è però una piccola differenza rispetto alla versione originale, ovvero *il bordo mancante sulla tab aperta*: si tratta una caratteristica piuttosto comune nelle tab, oltre che molto funzionale.

Per ottenere la tab attiva aperta in maniera cross-browser, il procedimento non è stato dei più semplici: è bastato però alla fine imitare il bordo inferiore del menu attraverso una piccola linea di sfondo, su cui i diversi bordi delle tab poggiano. La tab aperta ha un bordo inferiore bianco, che copre quindi l'immagine di sfondo della lista.

Da notare infine che lo script aggiunge la classe "active" al link corrispondente alla tab attiva e si occupa in maniera totalmente trasparente di nascondere e mostrare i vari pannelli. Non ci resta quindi che vedere il CSS dell'esempio per intero:

```
ul.tabnav{width: 100%;overflow:hidden;list-style: none;
margin: 0;padding:0;background:url(line.png) repeat-x bottom}
ul.tabnav li{float: left;margin: 0 0 0 0.5em;padding: 0}
ul.tabnav a{float: left;padding: 0 0 0 0.8em;
background: url(tab.png) no-repeat top left;
text-decoration: none;color: #222;
border-bottom: 1px solid #D7D7D7}
```

```
ul.tabnav span{float: left;padding: 0.6em 0.8em 0.6em 0;
background: url(tab.png) no-repeat top right;cursor: pointer}
ul.tabnav a.active,ul.tabnav a:hover{
background: url(tab2.png) no-repeat top left;
border-bottom:1px solid #FFF}
ul.tabnav a.active span,ul.tabnav a:hover span{
background: url(tab2.png) no-repeat top right;color: #184D8A}
```

Secondo esempio

Ho preparato anche un secondo esempio (<http://www.html.it/articoli/2447/tabEsempio2.html>) che mostra alcune possibilità avanzate dello script. Il primo pannello a tab infatti mostra la seconda tab inizialmente aperta, mentre la seconda interfaccia ha le tab circolari ogni tre secondi. I due contenitori generici hanno rispettivamente **id="tabcont1"** e **id="tabcont2"**, mentre per quanto riguarda lo script, ecco la parte prima della chiusura dell'elemento body:

```
<script type="text/javascript">
var tabber=new Yetii('tabcont1',2); //la seconda tab inizialmente aperta
var tabber2=new Yetii('tabcont2');
tabber.init();tabber2.init(3); //tab circolari ogni tre secondi
</script>
```

Per il primo pannello, dopo l'id del div su cui applicare le tab, andrà specificato dopo la virgola il numero che indica la tab da aprire all'avvio della pagina; mentre nel secondo caso, dentro le parentesi della chiamata `init` andrà indicato il numero di secondi di temporizzazione delle tab.

Conclusioni

Abbiamo visto in questo articolo una possibile implementazione delle tab, focalizzandoci sulla configurazione dello script e il markup necessario. Per quanto riguarda la personalizzazione con grafica e CSS, gli esempi qui presentati dovrebbero poter fornire al lettore un buon punto di partenza.

La compatibilità dei due esempi è decisamente buona: la resa visiva e la funzionalità dell'interfaccia sono state testate con successo su Internet Explorer dalla versione 5 alla 7, oltre che sulle ultime versioni di Opera, Firefox e Safari. Codice ed esempi sono disponibili per il download (http://www.html.it/articoli/2447/tabjs_dl.zip). Alla prossima.

Versione originale: <http://css.html.it/articoli/leggi/2447/interfacce-a-tab-con-css-e-javascript/>

© 1997-2006 [HTML.it](http://www.html.it)

La vendita, il noleggio, il prestito e la diffusione del contenuto di questa pagina sono vietate, tranne nei casi specificati nella pagina <http://www.html.it/info/note-legali.php>