Scrivere fogli di stile accessibili

Ormai avrete sentito spesso parlare di accessibilità e sarete abituati a considerarla come una serie di pratiche da adottare nella progettazione di un sito web. Saprete anche che un sito accessibile affida la formattazione dei suoi contenuti, quindi la presentazione, ai fogli di stile *CSS*.
Ebbene, anche questi possono essere resi più accessibili! Anzi, occorre fare attenzione a non vanificare un buon progetto dal punto di vista dell'accessibilità, compiendo cattive scelte nella scrittura dei CSS.

Lo stesso W3C, nella nota "CSS Techniques for Web Content Accessibility Guidelines 1.0" [http://www.w3.org/TR/WCAG10-CSS-TECHS/] descrive una serie di **tecniche e suggerimenti per sviluppare fogli di stile a cascata accessibili** e conformi alle Web Content Accessibility Guidelines 1.0 [http://www.w3.org/TR/WAI-WEBCONTENT/].

Nella nota del W3C vengono descritti **casi** in cui una priorità delle *WCAG* si ripercuote sul modo in cui utilizziamo i fogli di stile. Ma non spaventatevi, sebbene 17 punti siano molti, fortunatamente la nota è "abbordabile": ci sono brevi consigli dai titoli chiarificatori, esempi pratici corretti e altri "appositamente sbagliati".
In questo articolo vedremo quindi i più importanti di questi punti e vi fornirò alcuni consigli utili per il vostro foglio di stile.

Al punto *) "Diminuire la manutenzione ed incrementare la coerenza"* (la cui priorità è "3" dunque bassa considerando la scala della WCAG che va da "1" priorità massima ad appunto "3") si consiglia di usare regole di presentazione che possano valere per tutte le pagine del sito. Quindi si sconsiglia l'uso di CSS in linea o incorporati, meglio sarebbero fogli di stile esterni, nel numero minimo necessario. Se più di uno, utilizzare nei fogli lo stesso nome per la definizione di classi che rappresentano lo stesso concetto.
Credo che ciò ci trovi tutti d'accordo: regole di presentazione in linea appesantiscono la pagina e la leggibilità del codice (quelli incorporate un po' meno); troppi fogli di stile risultano difficilmente gestibili. Ne basta uno, accompagnato se necessario dal foglio di stile per la stampa. Si può importarlo o linkarlo, e definire i media a cui si riferisce (vedi mio articolo [http://www.christianfusi.com/cf/articoli/head.htm]).

Al punto 2) si parla anche della regola `!important`. La questione è complessa ma vi basti sapere che aggiungendo tale attributo ad una regola, questa prende il sopravvento su qualunque altra impostazione. Ma attenzione, questo deve valere per il foglio di stile dell'utente (se non lo sapete alcuni browser permettono di impostare

un foglio di stile personale per visualizzare le pagine web). Io utente posso volere sfondo nero e scritta bianca e usando `!important` prevalgo su di te, designer, che volevi lo sfondo giallo.

Il punto 3) è molto importante: *"Unità di misura"*. Dove possibile usate unità relative, non assolute [http://www.gdesign.it/pages/howto/articoli/cssdim/cssdim.php]! Questo perchè un browser come Internet Explorer non consente di ingrandire caratteri fissati in px/pt. Un vostro utente con problemi di vista potrebbe trovare faticoso o anche impossibile leggere quelle che scrivete "così in piccolo". Tra tutte le possibilità io vi consiglio questa: usate la percentuale per dare una dimensione generica ai font dal body, così: `body {font-size:80%}` mentre nelle altre regole usate gli `em`. Attenzione però che "em" è relativo al valore impostato dal selettore superiore a quelle che state definindo. Per cui, annidando i selettori, potreste finire co l'avere testo troppo piccolo (o grande):

primo selettore ad 1em

testo a 0.8em rispetto al div sopra (0.8 di 1em)

testo a 0.8em rispetto al div sopra (0.8 di 0.8em)

Al punto 4) si parla di generare contenuto per dare informazioni o aiuto supplementare all'utente. Ad esempio nell'HTML lo si fa definendo mediante gli `alt` un testo alternativo per le immagini nel caaso non venissero caricate. Nel CSS lo si potrebbe fare usando gli pseudo-elementi `:before` e `:after` insieme all'elemento `content`. Purtroppo su IE tutto ciò non funziona.* [#nota1]

I punti 5) 6) 7) 8) riguardano ancora la formattazione del testo. Superfluo dirvi che essa va fatta interamente via CSS. Vi ricordo solo un paio di regole: quando definite i font, mettete sempre per ultimo un font generico scegliendo tra: `serif` oppure `sans-serif`. I "sans-serif" sono i senza grazie, cioè i font bastoni cone Verdana, Arial, Monaco, Helvetica. I "serif" hanno invece i "trattini" (grazie), come il Times New Roman o il Garamond. Per esempio `body { font-family: Arial, Verdana, sans-serif; }`. Se sul vostro pc non ci sono installati i font da me specificati (arial e verdana), il browser userà il primo font che trova di quella famiglia definita dal font generico sans-serif.
Anche per il posizionamento del testo si posso usare regole CSS. Pertanto se dovete fare solo un rientro del testo, non usate a sproposito elementi strutturali (es. blockquote) ma meglio `margin, padding, text-indent` o per i più soffisticati `:first-letter` e altro ancora.

Punto 9) *"I colori"*. Come vuole l'accessibilità, state attenti affinchè il testo sia adeguatamente contrastato rispetto allo sfondo e provvedete a non fornire istruzioni che si basino esclusivamente sul colore (ad esempio "clicca il link rosso"). Nel css i colori vanno in esadecimale, se sono websafe bastano tre caratteri, così il rosso sarà `#f00` . Per contrasti e indicazioni su come vedono i colori persone che soffrono di disturbi visivi, esistono diversi tool online molto utili [indicati negli approfondimenti di quest'articolo].

Dal punto 10) in poi si ricorda di usare dove possibile testo aggiuntivo e fornire informazioni semantiche : liste, immagini, tabelle. Escludendo le liste (la cui soluzione offerta mi sembra inattuabile), il resto è roba nota. Attenti a come si leggerà il vostro sito senza i CSS, per controllare vi basta un secondo, fatelo!
Per questi motivi, oltre ad essere semanticamente più corretto, organizzare sempre un menù come una lista (usando i tag adeguati) è una buona abitudine, seguitela.

Al punto 14) viene consigliato di non operare dinamicamente sul documento, cioè non mostrare/nasconderne parti o cambiarne i colori "al volo". Non sono del tutto d'accordo. Penso che questa dinamicità alle pagine web, se ben studiata, possa giovare all'usabilità. Per l'accessibilità basta A) usare codice standard e seguire il *DOM* B) se si usa javascript far si che i contenuti siano accessibili anche ad utenti che l'hanno disabilitato (es. gli screenreader).

Infine le ultime precisazioni: nella nota 15) si parla dei fogli di stile uditivi. Lasciate perdere; purtroppo nessun browser li ha mai supportati, tanto che nelle specifiche dei CSS3 mi par di capire siano stati cambiati.
Allo stesso modo se predisponete fogli di style per altri device, prendete in cosiderazione le stampanti e basta.

Suggerimenti:

Spesso mi son chiesto se esiste un modo ottimale di "scrivere", non rispetto ai contenuti, ma alla forma, un CSS complesso. Con un po' d'esperienza son arrivato a queste conclusioni:

1. Usare molto i commenti per separare le varie parti del foglio.
2. Iniziare con classi generiche e poi definire le regole in base alla sezioni della pagina (header, main menu, content, footer).
3. Indentare il codice.

Qualcuno vi dirà che così cresce il peso del file css, è vero, ma una soluzione c'è: le shorthands. In una sola volta si possono dare molte indicazioni, risparmiando tantissimi byte.

Questo:	diventa:
`p { margin-top: 1px; margin-right: 5px; margin-bottom: 3px; margin-left: 5px; }`	`p { margin: 1px 5px 3px }` I valori vengono definiti in ordine orario, partendo da top. Se un valore è uguale a quello del suo opposto (top/bottom e left/right) basta definirlo una volta sola.
`p { border-color: #000; border-width: 5px; border-style: solid; }`	`p {border: 5px solid #000;}`
`p { font-family: verdana; font-size: 14px; line-height: 160%; font-style: italic; font-weight: bold; }`	`p {font: bold italic 14px/160% Verdana}`

Note:

* nonostante con IE questi pseudo-elementi non funzionino, gli altri browser li gestiscono. Quindi perchè non usarli per aggiungere micro-icone [http://www.kryogenix.org/days/external] o scrivere per esteso i link in stampa [http://www.constile.org/tutorial/stampare_con_i_css/]?

Approfondimenti (link a siti esterni):

- Comprimere i CSS con PHP. [http://www.fiftyfoureleven.com/sandbox/weblog/2004/jun/the-definitive-css-gzip-method/]
- CSS shorthands: [http://www.sorehead.org/css/shorthand.html] e [http://home.no.net/junjun/html/shorthand.html].
- Link utili sui CSS. [http://dezwozhere.com/links.html]
- Tutorial sui CSS: [http://www.w3schools.com/css/default.asp], [http://www.brainjar.com/css/using/].
- Naming delle sezioni di un layout. [http://www.stuffandnonsense.co.uk/archives/naming_conventions_table.html]
- Non usare il trattino basso nei nomi. [http://devedge.netscape.com/viewsource/2001/css-underscores/]
- I colori come vengono visti da chi ha disturbi visivi. [http://www.pixy.cz/apps/barvy/index-en.html]

- Calcolare il contrasto tra colori.
  [http://www.juicystudio.com/services/colourcontrast.asp]

Christian Fusi | luglio 2004