

GABRIELE ROMANATO

Menu

CSS: il principio della modularità

Il principio alla base di un design CSS adattabile, scalabile e mantenibile è la modularità.

Bisogna concepire un foglio di stile come un insieme di moduli interdipendenti tra di loro, non come un unico blocco monolitico di codice. I moduli devono rispecchiare la struttura del sito a cui gli stili verranno applicati. Ma non basta: occorre anche prevedere le future caratteristiche che verranno aggiunte al sito e progettare gli stili CSS in modo da poter riutilizzare moduli esistenti.

Classi e ID

Gli ID andrebbero usati solo per la struttura di base del sito, ossia per quelle parti che difficilmente verranno cambiate. Le classi, invece, andrebbero usate per riutilizzare il nostro codice più volte e in diversi contesti.

Ecco un errore che si commette molto spesso con le classi CSS:

```
h1.post-title {  
    font-size: 2em;  
}
```

Non si dovrebbe mai legare una classe ad un solo tipo di elementi. Cosa succederebbe infatti se in una pagina la marcatura fosse diversa?

```
<h2 class="post-title"></h2>
```

Gli stili non verrebbero applicati. Invece, le classi CSS dovrebbero essere agnostiche circa il tipo di elementi a cui verranno applicati:

```
.post-title {  
    font-size: 2em;  
}
```

Ora quindi tutti i seguenti elementi riceveranno lo stesso stile:

```
<h1 class="post-title"></h1>  
<h2 class="post-title"></h2>
```

Un altro errore che si commette è quello di sovraccaricare le classi di stili. Invece bisognerebbe creare pochi stili generici per classe e lasciare che si combinino tra loro grazie alla cascata:

```
.aligncenter {  
    margin: 0 auto;  
}  
  
.sizefull {  
    display: block;  
    max-width: 100%;  
}
```

Esempi d'uso:

```
<p>  
      
</p>  
<div class="aligncenter sizefull"></div>
```

Una cosa da ricordare è che le classi possono essere contestualizzate con stili più specifici usando sempre la cascata:

```
p > img.aligncenter.sizefull:first-child {  
    padding: 5px;  
    border: 1px solid #ccc;  
}
```

Selettori avanzati

Non esitate ad utilizzare i nuovi selettori CSS3 e i selettori avanzati CSS 2.1 se questo vi semplifica il processo di sviluppo. Infatti è molto più semplice scrivere:

```
a[href$=".pdf"] {  
  
    background: url(pdf.png) no-repeat 0 100%;  
  
}
```

che usare una classe apposita. Inoltre i selettori CSS 2.1 che per troppo tempo sono stati trascurati sono già disponibili da Internet Explorer 7 in poi, quindi non ha senso continuare a non usarli.

Raggruppare i moduli secondo la funzione svolta

I moduli andrebbero raggruppati usando i commenti CSS in base alla funzione svolta nel layout:

```
/*= HTML Elements Reset */
```

```
html, body, h1, h2, h3, h4, h5, h6,  
p, blockquote, address, ul, ol, dl, dt,  
dd, form, pre {
```

```
    margin: 0;  
    padding: 0;  
    font-size: 100%;  
    font-weight: normal;  
    font-style: normal;  
    list-style: none;
```

```
}
```

```
/*= Main wrapper */
```

```
#site {  
    width: 90%;  
    max-width: 1000px;  
    margin: 2em auto;  
}
```

```
/*= Classes */
```

```
.aligncenter {  
    margin: 0 auto;  
}
```

```
.sizefull {  
    display: block;  
    max-width: 100%;  
}
```

Il carattere = nei commenti serve ad effettuare ricerche con il proprio editor in modo da individuare tutti i moduli del foglio di stile e spostarsi facilmente da un modulo all'altro.

Se usate una nomenclatura coerente nei vostri moduli potete riutilizzare il codice preso da altri progetti e importarlo nel vostro foglio di stile.

Riutilizzare il codice

Per riutilizzare il vostro codice dovete pensare a tutti i progetti ai quali avete lavorato e stilare un elenco delle caratteristiche che li accomunano a livello di design CSS.

Dopo aver individuato queste caratteristiche, create dei moduli nel modo più generico possibile astraendo tutti quegli stili che sono comuni ai vari progetti e che avete utilizzato più volte.

Vedrete che alla fine avrete creato un vostro framework con cui poter lavorare.