

L'INCREDIBILE EM E I LAYOUT ELASTICI CON I CSS

Tue, 25th Sep 2007 Translated into Italian by [Nicola Pressi](#) [English version](#)→



Quasi un anno fa, Ty Gossman su [Stylegala](#) mi chiese di scrivere un articolo riguardante i layout elastici. Il meglio che ho potuto fare allora fu scrivere una veloce mail mentre lavoravo. Ho sempre voluto migliorare a questo proposito. Grazie per avermelo chiesto, mi scuso se ci ho messo così tanto.

Questo articolo vi guiderà attraverso la creazione di un semplice layout elastico; cosa sia esattamente un “em” e come calcolarlo, come usare gli em per creare layer elastici per contenuto con testo scalabile ed immagini, includendo un rudimentale ritmo verticale.

COS'È UN LAYOUT ELASTICO?

Un layout elastico si ridimensiona in base alla dimensione del testo dell'utente.

Più precisamente, un' *interfaccia* elastica si ridimensiona in base alla dimensione del testo del *browser*—solitamente di 16px—che l'utente può cambiare a piacimento. Alcune persone impostano delle dimensioni prefissate per ragioni di accessibilità, altre usano i [controlli UI](#) per incrementare la dimensione del testo a loro piacimento.

Il design elastico usa valori di *em* per *tutti* gli elementi. Gli em sono

una [misura relativa](#), scritta come: 1em, 0.5em, 1.5em ecc. Gli em possono essere specificati con tre cifre decimali, ad esempio: 1.063em. “Relativo” significa:

Sono calcolate in base alla dimensione del testo dell'elemento genitore. Per esempio, se un `<div>` ha una [dimensione di font calcolata](#) di 16px ogni elemento al suo interno—un figlio— [eredita](#) la stessa dimensione del font a meno che non venga cambiata. Se la dimensione del font del figlio viene cambiata in 0.75em allora la dimensione calcolata sarà $0.75 \times 16px = 12px$.

Se gli utenti incrementano (o diminuiscono) la dimensione del testo nel loro browser, l'intera interfaccia si stira(o si restringe.)

Guarda l' [esempio del layout elastico](#). Cambia la dimensione del testo per vederlo ridimensionarsi. Contiene tutto il codice CSS e HTML usato in questo tutorial.

Per altri esempi on-line, visita il sito di Dan Cederholm [elastic SimpleBits](#), o ridimensiona il testo su questo sito.

Le interfacce elastiche sono flessibili e accessibili per gli utenti, e possono essere precise quanto quelle al pixel. I layout possono essere creati in maniera veloce e accurata utilizzando i CSS una volta acquisiti i fondamenti delle relazioni tra la dimensione dei font, i pixel e gli em.

INTRODUZIONE AGLI EM, L'ELASTIGIRL DEI CSS

L'em è potente e flessibile quanto [Elastigirl](#), non le importa quanto sia la dimensione del font, se 12px, 16 o 60, lei sarà sempre uguale ad esso.

L'em è un'unità di misura in [tipografia](#). Cos'è e come mai questo nome? Ecco una piccola storia:

Un *em* era originariamente la misura del [blocchetto di metallo](#) usato per tagliare una singola lettera per uno specifico [font](#). Era grossomodo equivalente alla larghezza della lettera maiuscola “M”.

Altre letture:

[The amazing em unit](#)

[Em discussion on
Typophile](#)

Qui qualcuno potrebbe dire, “aspetta un momento, le lettere non occupano tutte lo stesso spazio.” Giusto. I computer usano il [kerning](#) per regolare lo spazio orizzontale che ogni lettera occupa sullo schermo, rendendole equidistanti e bilanciate. Originariamente, il blocchetto di metallo veniva tagliato o “kerned” (in inglese) per ottenere lo stesso spazio orizzontale tra la lettere.

Così, nei CSS, un em è effettivamente una misura verticale. Un em equivale allo spazio verticale necessario per ogni lettera in un font, lo stesso che occupa in orizzontale. Quindi:

Se la dimensione del font è 16px, allora 1em = 16px.

INIZIAMO

Prima di partire, però, dobbiamo sapere quale sia la dimensione del font di default per i browser. Fortunatamente, sappiamo che:

Tutti i più famosi browser hanno una dimensione dei font di default pari a 16px. Quindi, normalmente, 1em = 16px.

In Firefox, si può controllare la misura di default dei font tramite *Strumenti > Opzioni > Contenuto*.

Così, prima che un singolo selettore CSS sia scritto, il browser ha una dimensione dei font di default pari a 16px. Il tag `<body>` eredita questo valore se non gli viene assegnato altro tramite CSS. Pertanto 1em = 16px, 0.5em = 8px, 10em = 160px e così via. Ora possiamo specificare la dimensione di qualsiasi elemento usando

gli em!

SETTARE LA DIMENSIONE DEI FONT DEL BODY

Alcuni suggeriscono di settare la dimensione dei font del `<body>` in base alla dimensione del testo necessaria, oppure equivalente a 10px (0.625em o 62.5%) per poter calcolare la dimensione dei figli in maniera semplice. Entrambe hanno ragione, ma mi sembra più sensato lasciare le impostazioni di default del browser e cambiare solamente dove sia necessario.

Quindi, sappiamo che la dimensione dei font è pari a 16px di default. Sappiamo anche che se una persona ha cambiato i suoi settaggi, la nostra interfaccia elastica si regolerà in base a questi cambiamenti. Perfetto, così possiamo settare:

```
body{ font-size:1em; }
```

Purtroppo, IE ha un problema con gli em. Ridimensionando il testo da medium (default) a large in IE5/6 avremo un aumento inaspettato nella dimensione che va oltre quella prevista. Così ci vuole un altro selettore per fare in modo che anche su IE funzioni:

```
html{ font-size:100%; }
```

Come [Patrick H. Lauke](#) ha precedentemente [precisato](#), è una correzione per IE, non un hack— perché non sfrutta un bug di un browser per funzionare.

Diamo al nostro `<body>` qualche altro stile, e centriamo tutto nel nostro viewport (questo sarà importante successivamente per il content wrapper.) Il nostro CSS sarà come questo:

```
html{  
  font-size: 100%;  
}
```

```
body{
font-size: 1em;
font-family: georgia, serif;
text-align: center;
color: #444;
background: #e6e6e6;
padding: 0;
margin: 0;
}
```

LA FORMULA PER CONVERTIRE I PIXELS IN EM

Le prime volte che creerete pagine elastiche, vi troverete a fare *un sacco* di calcoli. Tenete una calcolatrice sotto mano.

Suggerimento: Trovare la dimensione dei font per ogni elemento con la brillante [Web developer toolbar](#) di [Chris Pederick](#) usando: Information > Display element information.

Non sono un mago della matematica, così ho bisogno di una semplice formula per ricordarmi come fare. Come designer, conosco intimamente i pixel, quindi partirò proprio da qui. Calcolo l'equivalente di 1px in em e moltiplico per la dimensione in pixel che mi serve. Questo calcolo mi fornisce il valore equivalente in em. Sappiamo che 1em è *sempre* equivalente alla dimensione del font dell'elemento genitore, quindi:

$$1 \div \text{dimensione del font dell'elemento genitore} \times \text{valore in pixel necessario} = \text{valore in em}$$

Per i vostri preferiti: [Pixel to ems conversion table](#) per la dimensione dei font.

Non lasciamo che il discorso sulla formula distolga l'attenzione dal nostro obiettivo. Le interfacce elastiche sono una gioia da costruire

quindi cerchiamo di fare pratica con la creazione di alcuni elementi della pagina.

COSTRUIRE UN CONTENITORE ELASTICO

Per creare un contenitore centrato come nell'[esempio](#), ci serve un po' di HTML. Con molta immaginazione gli daremo un ID chiamato "wrap":

```
<div id="wrap">
...
</div>
```

Vogliamo che abbia una larghezza di 740px perché possa stare comodamente in uno schermo con una risoluzione di 800×600px come nell'[esempio](#). Quant'è 740px in em? Scopriamolo:

Settiamo una larghezza di 740px in em per il nostro layer:

Sappiamo che l'elemento *genitore* (<body>) ha una dimensione dei font di 16px. Il nostro layer *figlio* (<div id="wrap">) erediterà lo stesso valore. Così, ora possiamo calcolare quanto vale 1px in em:

$$1\text{em} = 16\text{px}. \text{ Quindi, } 1\text{px} = 1 \div 16 = 0.0625\text{em}.$$

Quindi moltiplichiamo per 740 per ottenere gli em:

$$0.0625\text{em} \times 740 = 46.25\text{em}$$

Oppure possiamo fare un unico calcolo mediante la formula:

$$1 \div 16 \times 740 = 46.25\text{em}$$

$$(1 \div \text{dimensione del font dell'elemento genitore} \times \text{valore in pixel necessario} = \text{valore in em})$$

Potete utilizzare la stessa formula per calcolare qualsiasi valore di larghezza o altezza necessaria. Trovate quanto sia l'equivalente di 1px in em per quell'elemento. Moltiplicate quel valore per la

Gli em permettono solamente tre cifre decimali. Per i nostri calcoli vanno bene anche più di tre cifre decimali, ma prima di riportarle nel nostro CSS dobbiamo arrotondarle a tre.

dimensione in pixel che volete calcolare.

Creare il CSS:

Applicate la larghezza in em, centrate il layer nel viewport usando i margini automatici a sinistra e a destra, assegnate un bordo grigio di 1px con un background bianco e allineate il testo a sinistra:

```
#wrap{
width: 46.25em;
margin: 1.5em auto;
border: 0.063em solid #ccc;
background: #fff;
text-align: left;
}
```

Ora abbiamo un **wrapper elastico per il nostro contenuto!**

DARE UNO STILE AL TESTO CON GLI EM

Inseriamo un<h1> e un<p> al nostro wrapper in questo modo:

```
<div id="wrap">

<h1> ... <h1>
<p> ... <p>

</div>
```

Lecture indispensabili:
*The Elements of
Typographic Style applied
to the Web* creato da
Richard Rutter basato sul
lavoro di Robert
Bringhurst.

A questo punto, potremmo applicare qualche ulteriore accorgimento estetico scegliendo un **basic leading** e aggiungendo un po' di **ritmo verticale**, tutto espresso in em. Un po' di storia in più:

Leading (pronunciato “led-ing”) era tradizionalmente l’inserimento di alcune righe vuote al di sotto delle righe di testo. Nei CSS è espresso come `line-height` ma al posto di inserire spazio sotto il testo, aumenta lo spazio *sopra e sotto* ogni linea di testo.

In [questo esempio](#), ho usato la stessa dimensione dei font che [Richard Rutter](#) ha usato nel suo eccezionale [chapter on vertical motion](#) per rendere la vostra lettura più coerente possibile. La dimensione del font del titolo sarà 18px. Il paragrafo sarà 12px con una altezza di linea pari a 18px. 18px sarà il nostro *basic leading*—il valore più importante della nostra interfaccia. Tutto il resto sarà proporzionale a questo valore.

Una nota sull’[ereditarietà nei CSS](#): Il nostro wrapper, `<div id="wrap">`, non ha una sua dimensione di font impostata, così erediterà la dimensione di 1em (16px) dall’elemento genitore, il `<body>`. Ogni elemento all’interno del nostro wrapper erediterà questa dimensione del font, a meno che non gli impostiamo un altro valore, come stiamo per fare.

Impostate una dimensione del font di 12px con una altezza di linea di 18px:

Sappiamo che i nostri paragrafi hanno ereditato una dimensione del testo di 1em (16px) dall’elemento genitore, `<div id="wrap">`. Dal nostro precedente calcolo, sappiamo che 1px is 0.0625em. Moltiplicheremo semplicemente per 12 per ottenere il valore in em di 12px:

$$0.0625 \times 12 = 0.750em$$

Oppure in un’unica formula:

$$1 \div 16 \times 12 = 0.750em$$

(1 ÷ dimensione del font dell’elemento genitore × valore in pixel necessario = valore in em)

Quindi applichiamo al CSS:

```
p{
font-size: 0.750em;
}
```

Margini, altezza della linea, padding ecc. in em sono tutti *relativi* alla dimensione del font dell'elemento a cui appartengono.

Per calcolare l'altezza della linea desiderata e i margini di 18px per il nostro *basic leading* procediamo come segue:

$$18 \div 12 = 1.5em$$

Dividendo l'altezza della linea desiderata(18px) per la dimensione del font dell'elemento (12px) otteniamo il valore in em per l'altezza della linea. In questo esempio, l'altezza della linea è una volta e mezza la dimensione del font: 1.5em.

Aggiungiamo l'altezza della linea e i margini al CSS:

```
p{
font-size: 0.750em;
line-height: 1.5em;
margin: 1.5em;
}
```

Ora il browser dirà a se stesso, "Oh, l'altezza della linea e il margine sono settati a 1.5em, quindi dovrebbe essere 1.5 volte la dimensione dei font. Quant'è la dimensione del font? 12px? OK, bene, settiamo l'altezza della linea e il margine 1.5 volte quel valore, quindi 18px."

Settiamo la dimensione del font a 18px per l'elemento `<h1>`:

l'elemento `<h1>` ha *ereditato* una dimensione del font di 1em (16px) dal suo genitore, `<div id="wrap">`. Quindi, sappiamo già che 1px è come prima: 0.0625em. Moltiplichiamo semplicemente questo per 18 per ottenere il valore di 18px:

$$0.0625 \times 18 = 1.125em$$

Oppure in un'unica formula:

$$1 \div 16 \times 18 = 1.125em$$

(1 ÷ dimensione del font dell'elemento genitore × valore in pixel necessario = valore in em)

Quindi applichiamo al CSS:

```
h1{
  font-size: 1.125em;
}
```

Per conservare il nostro ritmo verticale vogliamo settare un'altezza della riga e un margine a 18px. Facile: Se la dimensione del font è 18px allora 18px in em diventerà 1em! Aggiungiamo queste proprietà al CSS (e facciamo in modo di alleggerire il peso del font:)

```
h1{
  font-size: 1.125em;
  line-height: 1em;
  margin: 1em;
  font-weight: 300;
}
```

Ora abbiamo ritmo verticale! Vediamo le immagini.

DIMENSIONARE LE IMMAGINI IN EM

Per conservare il ritmo della [pagina di esempio](#), la dimensione di un'immagine dovrebbe essere moltiplicata per il *basic leading*.

La nostra immagine ha una larghezza e un'altezza di 90px (18px × 5.) Ha un margine destro e inferiore di 18px ed è posizionata con un float: left nel paragrafo di testo. Questo è l'HTML:

```
<p>
   Lorem...
```

```
</p>
```

L'immagine è un *figlio* del paragrafo—suo *padre*—quindi sappiamo che l'immagine *eredita* una dimensione di font di 12px. Quindi, per calcolare la dimensione di larghezza e altezza della nostra immagine utilizzeremo la formula nel seguente modo:

$$1 \div 12 \times 90 = 7.5em$$

(1 ÷ dimensione del font dell'elemento genitore × valore in pixel necessario = valore in em)

Quindi applichiamolo al CSS:

```
p img{
width: 7.5em;
height: 7.5em;
}
```

Sappiamo già quanto sono 18px in em per il paragrafo genitore, quindi andiamo ad aggiungere le proprietà dei margini(e del float) al CSS:

```
p img{
width: 7.5em;
height: 7.5em;
margin: 0 1.5em 1.5em 0;
float: left;
}
```

N.B. Non è necessario che l'immagine sia contenuta in un paragrafo affinché abbia significato semantico. E' stata usata in questo modo per fornire un esempio di come utilizzare l'eredità per calcolare il valore in em.

Ora abbiamo un contenitore elastico con un contenuto elastico ed un ritmo verticale come nell'[esempio](#). Con un po'di fortuna, vi farete subito l'occhio creando layout elastici, ed in particolare progettando con gli em. Ci siamo, siete l'[Edna Mode](#) dei designers! La [pagina elastica d'esempio](#) contiene tutto l'HTML e il CSS che vi serve. Visualizzate il sorgente per accaparrarvelo.

UTILIZZI DEL DESIGN ELASTICO

Alcuni benefici del design elastico per i designer sono precisione, controllo e accessibilità. Tuttavia, alcune persone sostengono che consentire al contenuto di espandersi oltre i limiti del viewport con l'incremento del testo potrebbe essere problematico. Inoltre, può accadere che il testo sconfini nel framework quando la dimensione del testo viene cambiata; un problema minore ma notevole.

Preoccupazioni sono state espresse anche riguardo la perdita di qualità delle immagini quando c'è un incremento della dimensione del testo. Speriamo che i browser migliorino il ridimensionamento, o forse in futuro avremo la possibilità di utilizzare [SVG](#) e [background images scalabili](#).

Implementare design elastici porta una profonda conoscenza degli em nei CSS. Senza dubbio è uno strumento che non può mancare nella cassetta degli attrezzi di ogni web designer o sviluppatore, da utilizzare sia da solo che combinato con altre tecniche per creare layout ibridi.

IN CONCLUSIONE...

Qualsiasi omissioni o errore presenti in questo articolo sono da ascrivere a me, quindi fatemi sapere se ho dimenticato qualcosa. Se conoscete altre fonti interessanti e pertinenti, vi prego di aggiungerle nei commenti o scrivetemi una e-mail e la includerò.

Mi prendo la piena responsabilità per tutti i riferimenti a [Gli Incredibili](#)—i miei 2 figli mi hanno dato il permesso di usare le loro figure per l'immagine del titolo—e spero che [Elastigirl](#) non si sia infastidita per essere stata comparata all'unità em.

Translated into Italian by [Nicola Pressi](#).

RIFERIMENTI E ALTRE LETTURE

Risorse:

- a. [Elastic layout example \(from the article\)](#)
- b. [Reference table of common pixel font sizes and em equivalents](#)

Altre letture:

- a. [The amazing em unit](#)
- b. [Em discussion on Typophile](#)
- c. [The Elements of Typographic Style applied to the Web — Richard Rutter](#)
- d. [CSS2: Font size](#)
- e. [CSS2: Assigning property values, Cascading, and Inheritance](#)
- f. [CSS Layouts: The Fixed. The Fluid. The Elastic. — Mike Cherim](#)
- g. [Fixed or fluid width? Elastic! — Roger Johansson](#)
- h. [Elastic Design — A List Apart](#)

[Comments \(from the original English version\)→](#)

CAN YOU TRANSLATE?

If you are bilingual and would like to translate this article for non English speakers, please download my vCard via the [about page](#) and contact me. Thanks!

***Jon Tan** fecit, 2007. Some rights reserved. vCard (via X2V).*