Grid

Maria Chiara Pievatolo

Università di Pisa

pievatolo@dsp.unipi.it

25 novembre 2021

Sommario

Grid può controllare due assi *contemporaneamente*

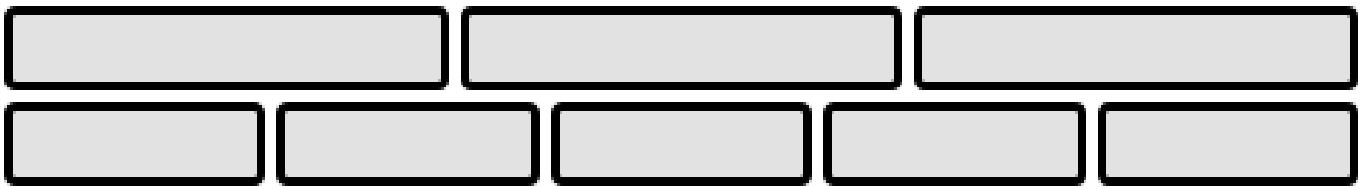


Figure 1 *Representative Flex Layout Example*

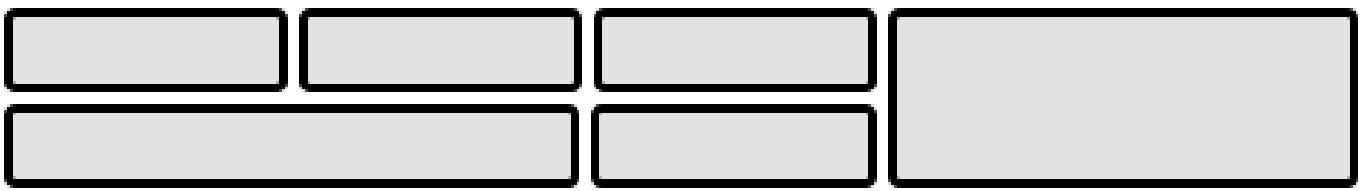


Figure 2 *Representative Grid Layout Example*

`https://www.w3.org/TR/css-grid-1/#intro`

Componenti e principi

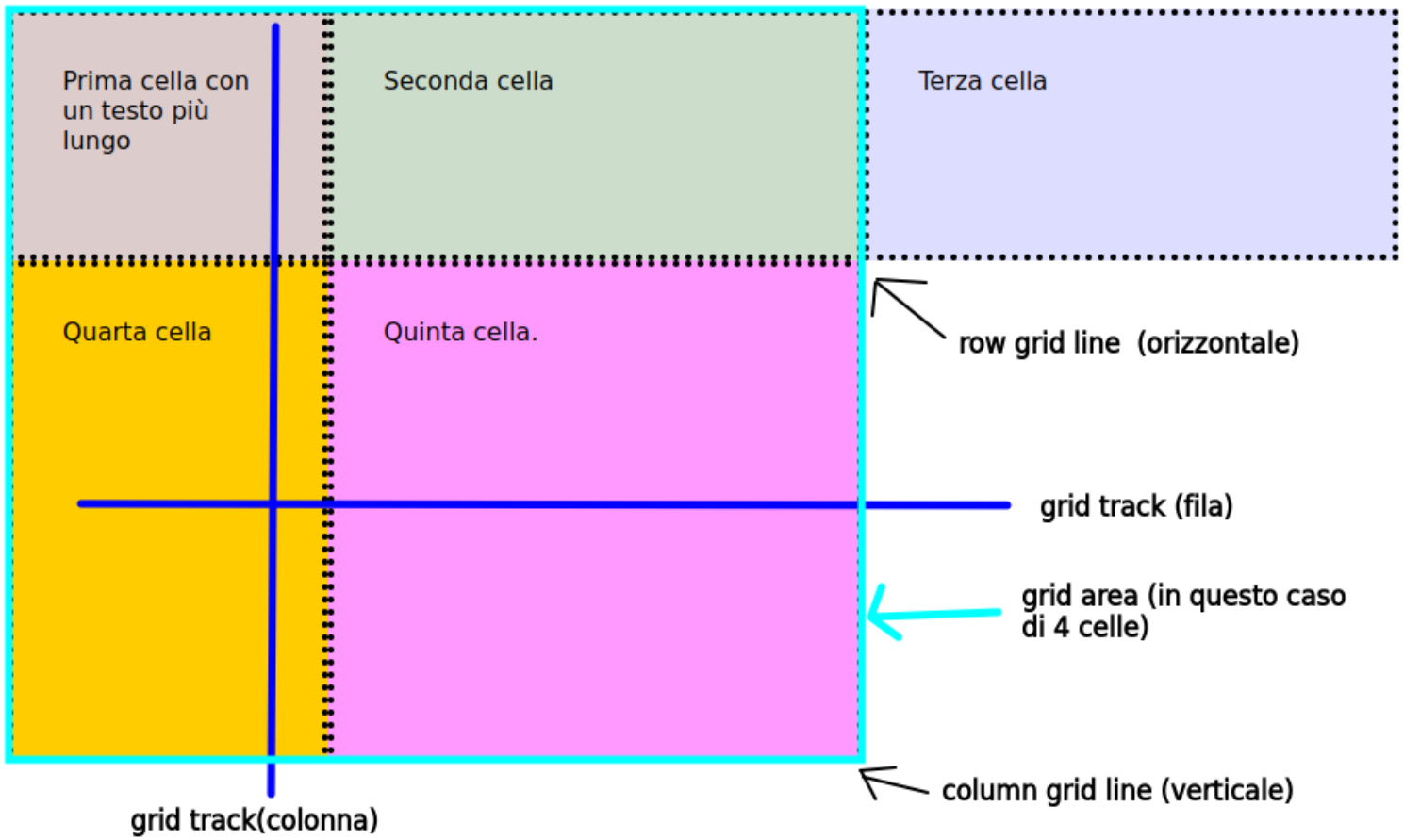
Componenti:

- un elemento contenitore (display:grid o display:inline-grid)
- degli elementi figli (grid items)

Principi:

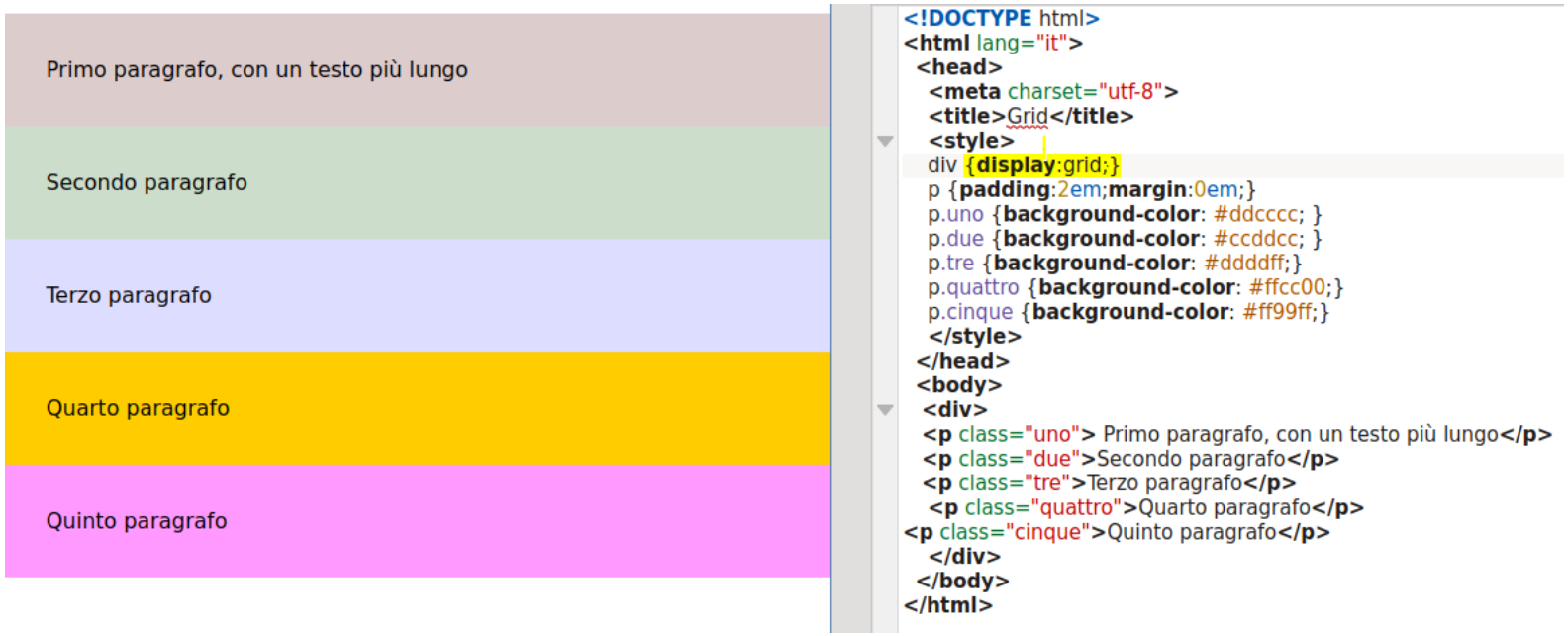
- orizzontalmente ogni cella di una fila è alta quanto la più alta della fila (salvo diversa indicazione)
- verticalmente ogni cella di una colonna è larga quanto la più larga (salvo diversa indicazione)

Nomenclatura



https://css-tricks.com/snippets/css/complete-guide-grid/#important-terminology

Display:grid



Il contenitore si comporta come un elemento a blocco

Display:inline-grid

Il contenitore si comporta come un elemento inline.

Grid-template esplicito

Numero e dimensioni di righe e colonne

```
grid-template-columns: 30% 25% 30%;
/* lista con la larghezza di ciascuna colonna */
grid-template-rows: 10em 10em;
/* lista con l'altezza di ciascuna riga */
```

Grid-template esplicito e implicito

Queste dichiarazioni si applicano a un massimo di 6 elementi (grid-template esplicito):

grid-template-columns: 30% 25% 30%; vale per tre colonne

grid-template-rows: 10em 10em vale per due righe

Grid-template implicito

se gli elementi sono più di 6, continueranno ad allinearsi su 3 colonne e si estenderanno in ulteriori righe

Grid-template implicito


```
<meta charset="utf-8">
<title>Grid</title>
<style>
div {display:grid;
grid-template-columns: 10% 25% 30%;
grid-template-rows: 10em 10em; }
p {padding:2em;margin:0em;}
p.uno {background-color: #ddcccc; }
p.due {background-color: #ccddcc; }
p.tre {background-color: #ddddff;}
p.quattro {background-color: #ffcc00;}
p.cinque {background-color: #ff99ff;}
</style>
</head>
<body>
<div>
<p class="uno"> Primo paragrafo,
con un testo più lungo</p>
<p class="due">Secondo paragrafo</p>
<p class="tre">Terzo paragrafo</p>
<p class="quattro">Quarto paragrafo</p>
<p class="cinque">Quinto paragrafo</p>
<p class="quattro">Paragrafo aggiuntivo</p>
<p class="cinque">Paragrafo aggiuntivo</p>
<p class="uno">Paragrafo aggiuntivo</p>
</div> <!--altro testo esterno alla griglia-->
</body>
</html>
```

Grid-template implicito

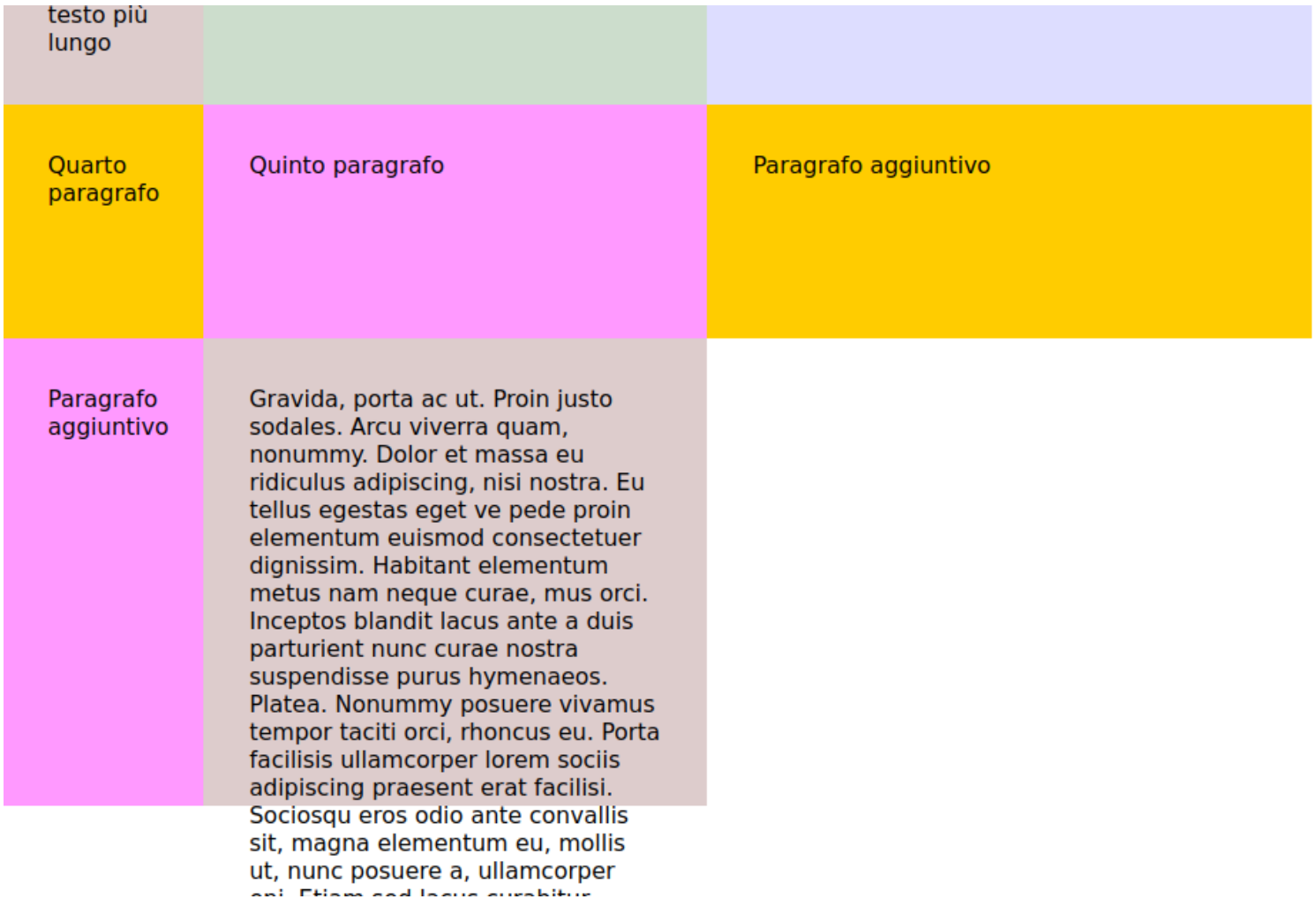
Regolare l'altezza delle righe aggiuntive

```
grid-auto-rows: 30em;
/*Un'altezza fissa espone al rischio di straripamenti
 ↪  del contenuto: per evitarli sostituisco la
 ↪  dichiarazione con questa:*/
grid-auto-rows: minmax( 30em, auto );
/*La fila è alta almeno 30em, ma può espandersi
 ↪  automaticamente, quando serve*/
```

Grid-auto-rows


```
<meta charset="utf-8">
<title>Grid</title>
<style>
div {display:grid;
grid-template-columns: 10% 25% 30%;
grid-template-rows: 10em 10em; }
p {padding:2em;margin:0em;}
p.uno {background-color: #ddcccc; }
p.due {background-color: #ccddcc; }
p.tre {background-color: #ddddff;}
p.quattro {background-color: #ffcc00;}
p.cinque {background-color: #ff99ff;}
</style>
</head>
<body>
<div>
<p class="uno"> Primo paragrafo,
con un testo più lungo</p>
<p class="due">Secondo paragrafo</p>
<p class="tre">Terzo paragrafo</p>
<p class="quattro">Quarto paragrafo</p>
<p class="cinque">Quinto paragrafo</p>
<p class="quattro">Paragrafo aggiuntivo</p>
<p class="cinque">Paragrafo aggiuntivo</p>
<p class="uno">Paragrafo aggiuntivo</p>
</div> <!--altro testo esterno alla griglia-->
</body>
</html>
```

Grid-auto-rows: straripamento



Grid-auto-rows: minmax(20em, auto)

Quarto paragrafo	Quinto paragrafo	Paragrafo aggiuntivo
Paragrafo aggiuntivo	Gravida, porta ac ut. Proin justo sodales. Arcu viverra quam, nonummy. Dolor et massa eu ridiculus adipiscing, nisi nostra. Eu tellus egestas eget ve pede proin elementum euismod consectetuer dignissim. Habitant elementum metus nam neque curae, mus orci. Inceptos blandit lacus ante a duis parturient nunc curae nostra suspendisse purus hymenaeos. Platea. Nonummy posuere vivamus tempor taciti orci, rhoncus eu. Porta facilisis ullamcorper lorem sociis adipiscing praesent erat facilisi. Sociosqu eros odio ante convallis sit, magna elementum eu, mollis ut, nunc posuere a, ullamcorper eni. Etiam sed lacus curabitur suspendisse habitasse odio potenti ac, faucibus. Ultricies. Aliquam cubilia arcu mus platea vivamus, gravida.	

Stenografia

grid-template: row track list / column track list

Invece di:

```
div {display:grid;
        grid-template-columns:1fr 2.5fr 3fr;
        grid-template-rows: 10em 10em;}
```

si può scrivere

```
div {display:grid; grid-template: 10em 10em / 1fr
↪  2.5fr 3fr ;}
```

fr: frazione che distribuisce lo spazio libero entro un track

In un track prima viene distribuito lo spazio riservato a ciascun elemento non flessibile; poi viene suddiviso lo spazio rimanente indicato dalla frazione o proporzione fr.

grid-template-columns: 1fr 2.5fr 3fr

Stenografia

Ripetere le colonne, in serie uguali e no

```
/*Tre colonne che si spartiscono ugualmente lo spazio
↪  libero*/
grid-template-columns: repeat( 3, 1fr );
/*Una coppia di colonne che si spartisce lo spazio
↪  libero in modo disuguale, ripetuta tre volte:
↪  quindi 6 colonne in tutto */
grid-template-columns: repeat( 3, 1fr,2fr );
```

grid-template-columns: repeat(2, 1fr 2fr)

Stenografia

Ripetere le colonne, con autofill o autofit

```
/*Le celle vuote sono eliminate */
grid-template-columns: repeat( autofit, minmax(30px
↪  1fr) ;
/* Le celle vuote residue rimangono visibili
↪  */
grid-template-columns: repeat( autofill, minmax(30px
↪  1fr);
```

Autofit riempie lo spazio

Autofill lascia libero lo spazio residuo

Colonne adattive senza media queries!

Una dichiarazione brevissima con autofill o autofit

```
display:grid;
grid-template-columns: repeat( auto-fill, minmax(
↪  240px, 1fr) );
grid-auto-rows: minmax( 20em, auto );
```

https://lab.sp.unipi.it/~chiara/html5/gridresponsive.html

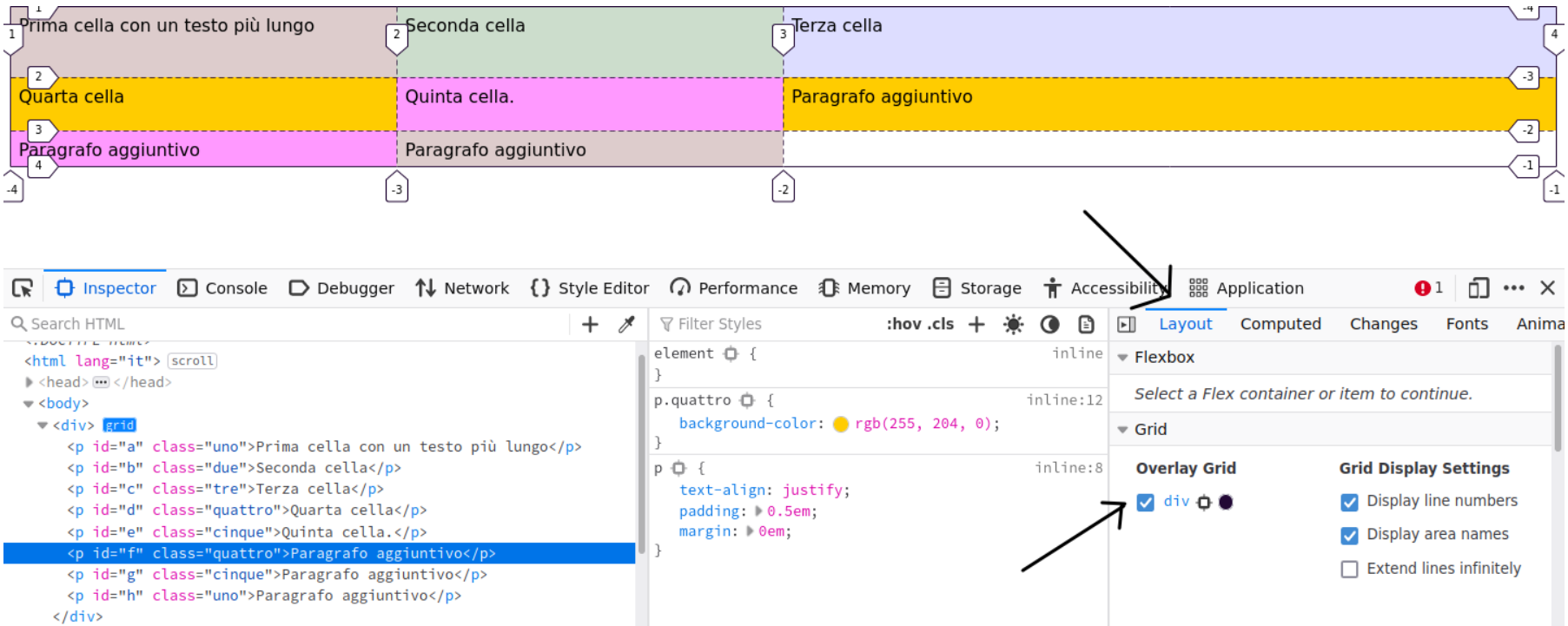
Disposizioni complesse

- È possibile usare grid per strutturare pagine con disposizioni complesse
- se siamo in grado di identificare ciascuna componente della griglia
- così da poterne cambiare l'estensione

Modalità di identificazione

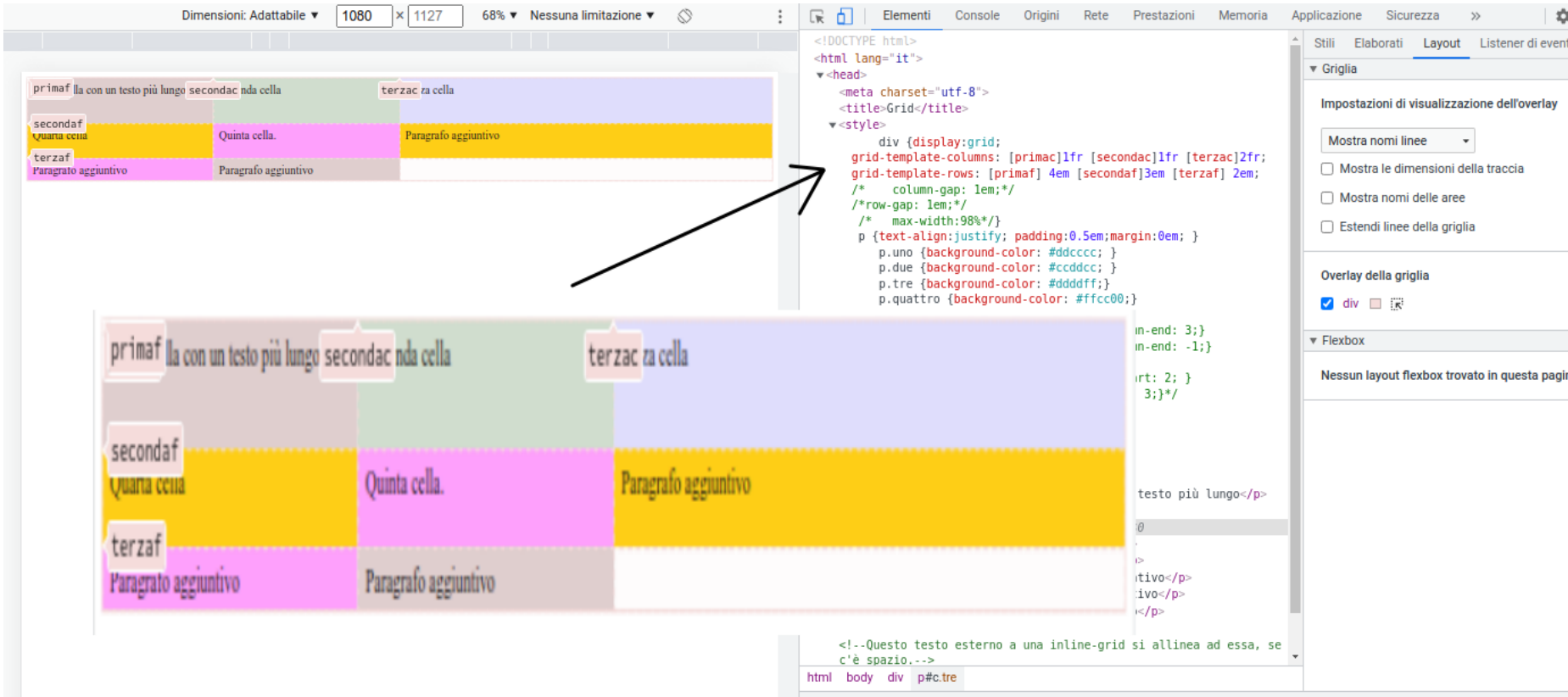
- Tramite numeri di riga (row grid line, column grid line)
- Tramite nomi di riga scelti dall'autore del CSS
- Tramite span (estensione)
- Tramite aree del template (grid-templates-areas)

Numeri di riga, visibili da browser con "Ispeziona"



I valori negativi indicano un ordine a ritroso, da fine riga o da fine colonna

Nomi di riga, visibili da Chromium con "Ispeziona"



I nomi sono assegnati ad arbitrio da chi scrive il CSS

Strumenti

Per indicare dove comincia e finisce una cella attribuisco come valore un numero o un nome di riga a:

- grid-column-start
- grid-column-end
- grid-row-start
- grid-row-end

Dato un documento HTML5 così strutturato

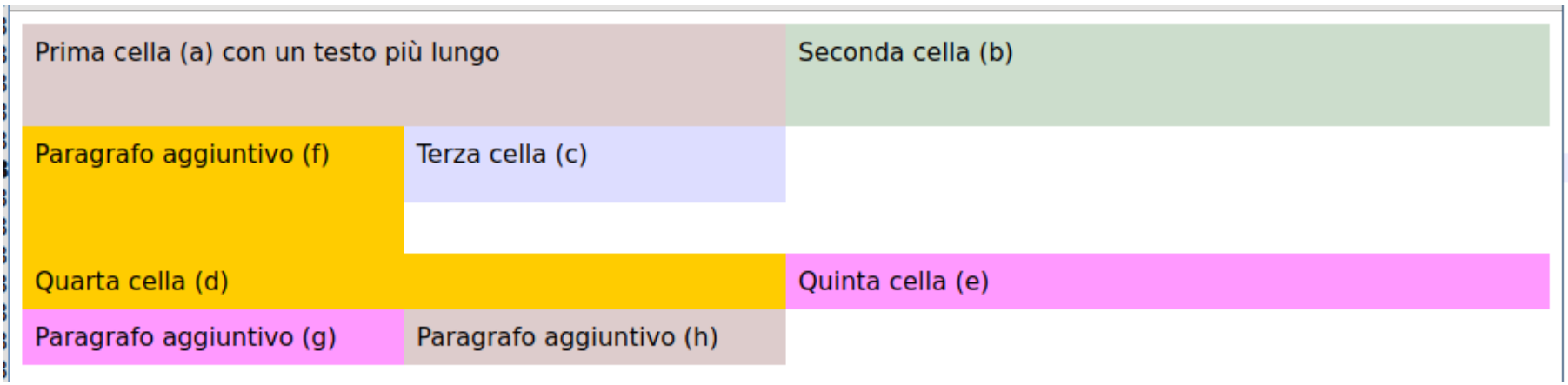
```
<div>
<p class="uno" id="a"> Prima cella
con un testo più lungo</p>
<p class="due" id="b">Seconda cella</p>
<p class="tre" id="c">Terza cella</p>
<p class="quattro" id="d">Quarta cella</p>
<p class="cinque" id="e">Quinta cella. </p>
<p class="quattro" id="f">Paragrafo aggiuntivo</p>
<p class="cinque" id="g">Paragrafo aggiuntivo</p>
<p class="uno" id ="h">Paragrafo aggiuntivo</p>
</div>
```

...cambio le estensioni delle celle così:

```
div {display:grid;
grid-template-columns: [primac]1fr [secondac]1fr
↪  [terzac]2fr;
grid-template-rows: [primaf] 4em [secondaf]3em
↪  [terzaf] 2em;}
#a{grid-column-start: /*1;*/ primac;grid-column-end:
↪  /*3;*/ terzac;}
#d{grid-column-start:/* -2*/ primac; grid-column-end:
↪  /*-4*/ terzac;}
#f {grid-row-start: secondaf; grid-row-end:
↪  4;}
```

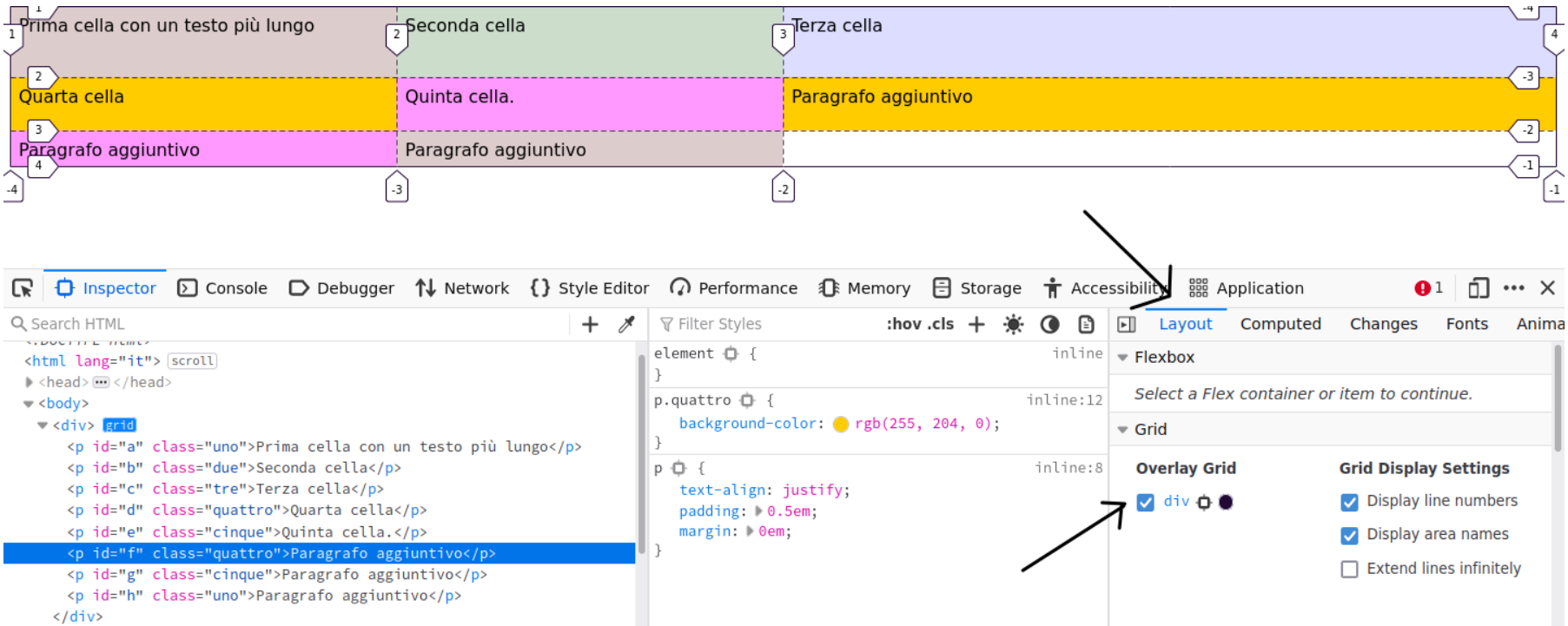
Posso usare in alternativa i valori commentati

Risultato



https://lab.sp.unipi.it/~chiara/html5/gridcomplessa.html

Span



Anziché i numeri di riga, posso estendere le righe e le colonne

Esempio

```
/* Estendi il paragrafo a orizzontalmente di due
 ↪  celle e verticalmente di tre*/
#a{grid-column-start: span 2; grid-row-start: span
 ↪  3;}
#c {grid-row-start: span 2;}
#d{grid-column-start: span 3;}
#f {grid-row-start: span 2;}
```

Risultato

https://lab.sp.unipi.it/~chiara/html5/gridspan.html

Dato un documento HTML5 così strutturato

```
<body>
<header>Intestazione</header>
<article>Articolo</article>
<nav>Barra di navigazione</nav>
<footer>Piè di pagina</footer>
</body>
```

...definisco nel CSS le aree del template

```
        body {
display: grid;
grid-template-rows: repeat(2, auto);
grid-template-columns: 4fr 1fr;
grid-template-areas: "pagehead pagehead"
"mains navigation"
"pagefoot pagefoot"; }
```

...e le associo agli elementi HTML contenuti in body

```
header {grid-area: pagehead;}
article {grid-area: mains;}
nav {grid-area:navigation:}
footer { grid-area: pagefoot;}
```

Spiegazione

Nell'esempio ho una griglia a due colonne...

- Riservo le due celle della prima fila per l'intestazione (pagehead pagehead)
- Distribuisco le due celle della seconda fila a contenuto principale e menu di navigazione (mains navigation)
- Riservo le due celle dell'ultima fila per il piè di pagina (pagefoot pagefoot)
- Associo infine gli elementi HTML alle aree così definite

Risultato

https://lab.sp.unipi.it/~chiara/html5/area.html

Mind the gap! I

Aggiungo spazio fra le caselle di una griglia

```
column-gap: 1em;
row-gap: 1em;
 /* Stenografia per righe e colonne */
gap: 1em
```

Mind the gap! II

Attribuisco un margine agli elementi di una griglia

```
p {margin:1em; }
```

Come verrebbe visualizzato p se non fosse compreso in grid?

Avvertenze

- le proprietà justify-..., align-... valgono anche per grid
- se una cella contiene un'immagine non flessibile grid cessa di essere responsive.

Ricordiamoci di attribuire a img "max-width:100%;"


Chris House (2021)
A Complete Guide to Grid
https://css-tricks.com/snippets/css/complete-guide-grid/


FreeCodeCamp (2021)
Css Grid
https://www.freecodecamp.org/learn/responsive-web-design/#css-flexbox