

# Elements & Typography

## Grid

For laying out content on a page, the core hub framework includes styles for a 12-column grid.

1

3

3

3

1

3

3

1

1

1

1

1

The grid supports up to 12 columns with `span#` and `offset#` classes.

Each column **must** have a .col class. The last column in a set must have the .omega class added for IE 7 to work properly. No clearing div is required.

For example, a four column grid would look like:

```
<div class="grid">
  <div class="col span3">
    ...
  </div>
  <div class="col span3">
    ...
  </div>
```

## ELEMENTS & TYPOGRAPHY

---

```
</div>
<div class="col span3">
  ...
</div>
<div class="col span3 omega">
  ...
</div>
</div>
```

Output:

...

...

...

...

## Spanning Columns

Columns can be spanned to easier portion content on the page. In the following example, we span the first 6 columns in a container, then follow with two, smaller 3 column containers for a 3-column layout where the first column takes up 50% of the space.

```
<div class="grid">
  <div class="col span6">
    ...
  </div>
  <div class="col span3">
    ...
  </div>
  <div class="col span3 omega">
    ...
  </div>
</div>
```

Output:

...  
...  
...

### Offsets

Columns may also be offset or 'pushed' over.

```
<div class="grid">
  <div class="col span3 offset3">
    ...
  </div>
  <div class="col span3">
    ...
  </div>
  <div class="col span3 omega">
    ...
  </div>
</div>
```

Output:

...  
...  
...

### Helper Classes

.span-quarter  
Span 3 columns. This is equivalent to .span3  
.span-third  
Span 4 columns. This is equivalent to .span4  
.span-half  
Span 6 columns. This is equivalent to .span6  
.span-two-thirds  
Span 8 columns. This is equivalent to .span8

## ELEMENTS & TYPOGRAPHY

---

### .span-three-quarters

Span 9 columns. This is equivalent to .span9

A four column grid with the helper classes:

```
<div class="grid">
  <div class="col span-quarter">
    ...
  </div>
  <div class="col span-quarter">
    ...
  </div>
  <div class="col span-quarter">
    ...
  </div>
  <div class="col span-quarter omega">
    ...
  </div>
</div>
```

There are equivalent .offset- classes as well:

### .offset-quarter

Offset 3 columns. This is equivalent to .offset3

### .offset-third

Offset 4 columns. This is equivalent to .offset4

### .offset-half

Offset 6 columns. This is equivalent to .offset6

### .offset-two-thirds

Offset 8 columns. This is equivalent to .offset8

### .offset-three-quarters

Offset 9 columns. This is equivalent to .offset9

Markup for a four column grid with the offset helper class:

```
<div class="grid">
  <div class="col span-quarter">
    ...
  </div>
  <div class="col offset-quarter span-quarter">
    ...
  </div>
  <div class="col span-quarter omega">
```

```
...
</div>
</div>
```

Output:

```
...
...
...
```

### Nesting Grids

The following is an example of a 3 column grid nested inside the first column of *another* 3 column grid.

```
<div class="grid">
  <div class="col span6">
    <div class="grid">
      <div class="col span4">
        ...
      </div>
      <div class="col span4">
        ...
      </div>
      <div class="col span4 omega">
        ...
      </div>
    </div>
  </div>
<div class="col span3">
  ...
</div>
<div class="col span3 omega">
  ...
</div>
</div>
```

Output:

...

...

...

...

...

## Notifications

The core framework provides some base styles for alter and notifications.

```
<p class="passed">Success message</p>
```

Success message

```
<p class="info">Info message</p>
```

Info message

```
<p class="help">Help message</p>
```

Help message

```
<p class="warning">Warning message</p>
```

### Warning message

```
<p class="error">Error message</p>
```

### Error message

## Sections & Asides

The majority of hub components have content laid out in a primary content column with secondary navigation or metadata in a smaller side column to the right. This is done by first wrapping the entire content in a div with a class of .section. The content intended for the side column is wrapped in a `<div class="aside">` tag. The primary content is wrapped in a `<div class="subject">` tag and immediately follows the .aside column.

**Note:** The .aside column must come first in order for the content to be positioned properly. If, unfortunately, this poses a semantic problem, we recommend using the grid system as a potential alternative.

Using aside & subject differs from the grid system in that the .aside column has a fixed width with the .subject column taking up the available left-over space. In the grid system, **every** column is flexible (uses a percentage of the screen) and cannot have a specified, fixed width.

Example usage:

```
<div class="section">
  <div class="aside">
    Side column content ...
  </div>
  <div class="subject">
    Primary content ...
  </div>
  <div class="clear"></div>
</div>
```