

CSS GRID

&

RESPONSIVE DESIGN

## ASIDE: CSS VARIABLES

You can set up variables that can be used in your CSS to make changing values easier.

```
:root {  
    --main-bg-color: gray;  
}  
  
body {  
    background-color: var(--main-bg-color);  
}
```

# INTRODUCING...

## CSS GRID

# GRID CONTAINERS

- Usually used to divide a page into sections

<https://gridlayout.io/demos/holy-grail.html>

- Set up using a **Grid Container**

# GRID CONTAINERS

- `display: grid;`
- `grid-template-columns`

```
grid-template-columns: 20px 100px 50px;  
grid-template-columns: 100px 1fr 1fr;
```

- `grid-template-areas`

```
grid-template-areas:  
    "name1 name2 name3"  
    ". name4 name 5"  
    "name6 name6 name6";
```

# GRID CONTAINERS

- `grid-gap: 10px;`
- `grid-area`

```
header {  
    grid-area: header;  
}
```

- Used in `grid-template-areas`

```
grid-template-areas:  
    "header header header"  
    ". name4 name 5"  
    "name6 name6 name6";
```

# GRID CONTAINERS

- `grid-template-rows`

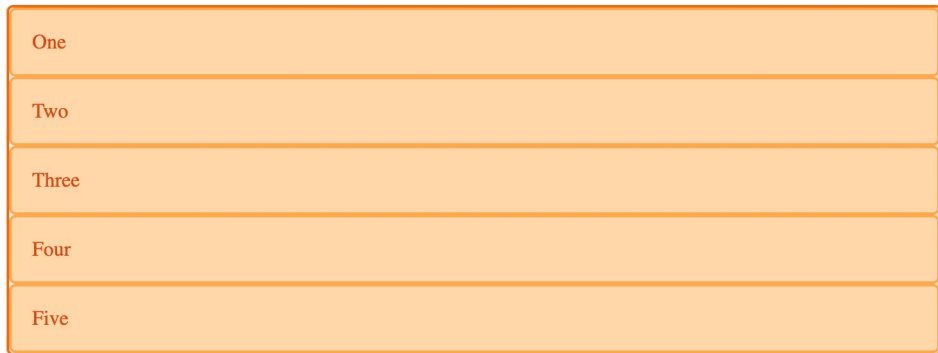
```
grid-template-rows: 100px 1fr 100px;
```

- Grid items are **direct children** of the grid container in the HTML.

# DIRECT CHILDREN AS GRID ITEMS

```
<div class="wrapper">  
  <div>One</div>  
  <div>Two</div>  
  <div>Three</div>  
  <div>Four</div>  
  <div>Five</div>  
</div>
```

```
.wrapper {  
  display: grid;  
}
```





# GRID ALIGNMENT

- **justify-items**
  - aligns items along the row axis
    - start
    - end
    - center
    - stretch

# GRID ALIGNMENT

- **align-items**

- aligns items along the column axis

- start

- end

- center

- stretch

# GRID ALIGNMENT

- **justify-self**
  - overrides row alignment for element.
- **align-self**
  - overrides column alignment for element.

RESPONSIVE DESIGN

# RESPONSIVE DESIGN

A page should respond to the user's environment based on screen size, platform, and orientation (landscape / portrait).

Consider:

- Flexible grid layout
  - Use percentages, not pixels
- Responsive images
  - Use percentages, not pixels

```
img {  
    width: 100%;  
    height: auto;  
}
```

- Change layout entirely based on screen characteristics

# MEDIA QUERIES

Selectively apply CSS based on screen/environment characteristics.

- **@media** rule
  - [https://www.w3schools.com/cssref/css3\\_pr\\_mediaquery.asp](https://www.w3schools.com/cssref/css3_pr_mediaquery.asp)
- Apply CSS conditionally based on **@media** query

```
@media screen and (max-width: 768px) {  
    .container {  
        // override properties  
    }  
}:
```

# BREAKPOINTS

- The width where "something changes" in the layout
- Breakpoints are your choice. You may have zero, or many

@media queries  
used in Bootstrap for  
reference

```
// Extra small devices (portrait phones, less than 576px)
// No media query since this is the default in Bootstrap

// Small devices (landscape phones, 576px and up)
@media (min-width: 576px) { ... }

// Medium devices (tablets, 768px and up)
@media (min-width: 768px) { ... }

// Large devices (desktops, 992px and up)
@media (min-width: 992px) { ... }

// Extra large devices (large desktops, 1200px and up)
@media (min-width: 1200px) { ... }
```

# MOBILE FIRST

- **Mobile devices**, mobile screens are now the **primary means** in which **users interact** with your applications.
- The term **mobile-first** was coined to indicate that **applications should be developed with a "mobile-first mindset."**
- Instead of adding breakpoints into the design as the width of the screen gets smaller, you should **create breakpoints in the design when the width of the screen gets larger.**
- Start with the small screen first, then expand until it looks bad. At this point, it's time to insert a new breakpoint.

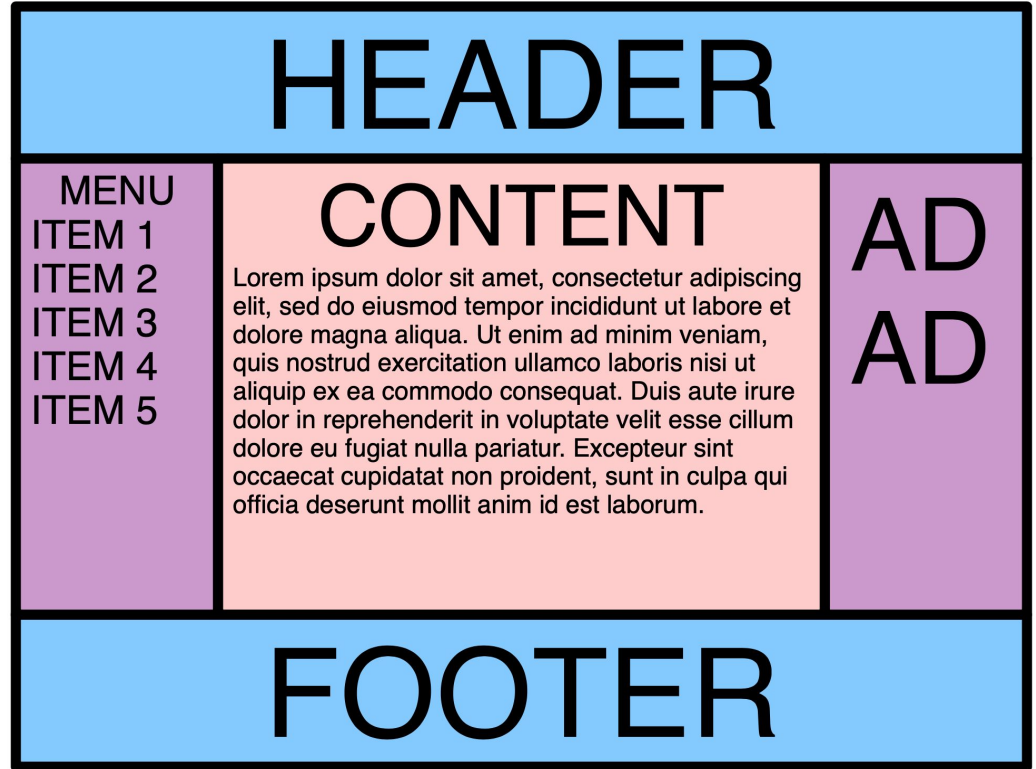


# CODE WALKTHROUGH

The "Holy Grail" Layout:

The sidebars have a fixed width, with the center column adjusting in size to fill the window (fluid or liquid layout).

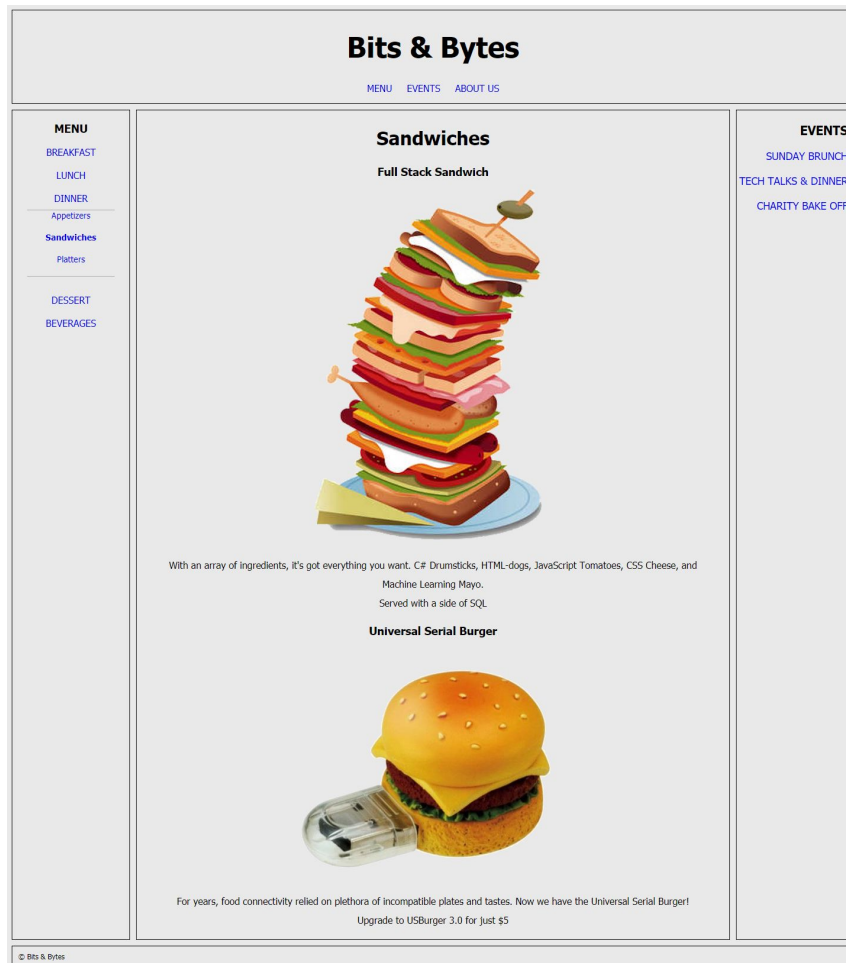
Another common requirement is that, when a page does not contain enough content to fill the screen, the footer should drop to the bottom of the browser window instead of leaving blank space underneath.



By David Lark - Own work, CC BY-SA 4.0,  
<https://commons.wikimedia.org/w/index.php?curid=42413988>

# DESKTOP LAYOUT

This is the layout that should be applied for desktop screen sizes.



# MOBILE LAYOUT

This is the layout that should be applied for screen sizes smaller than 768px.

