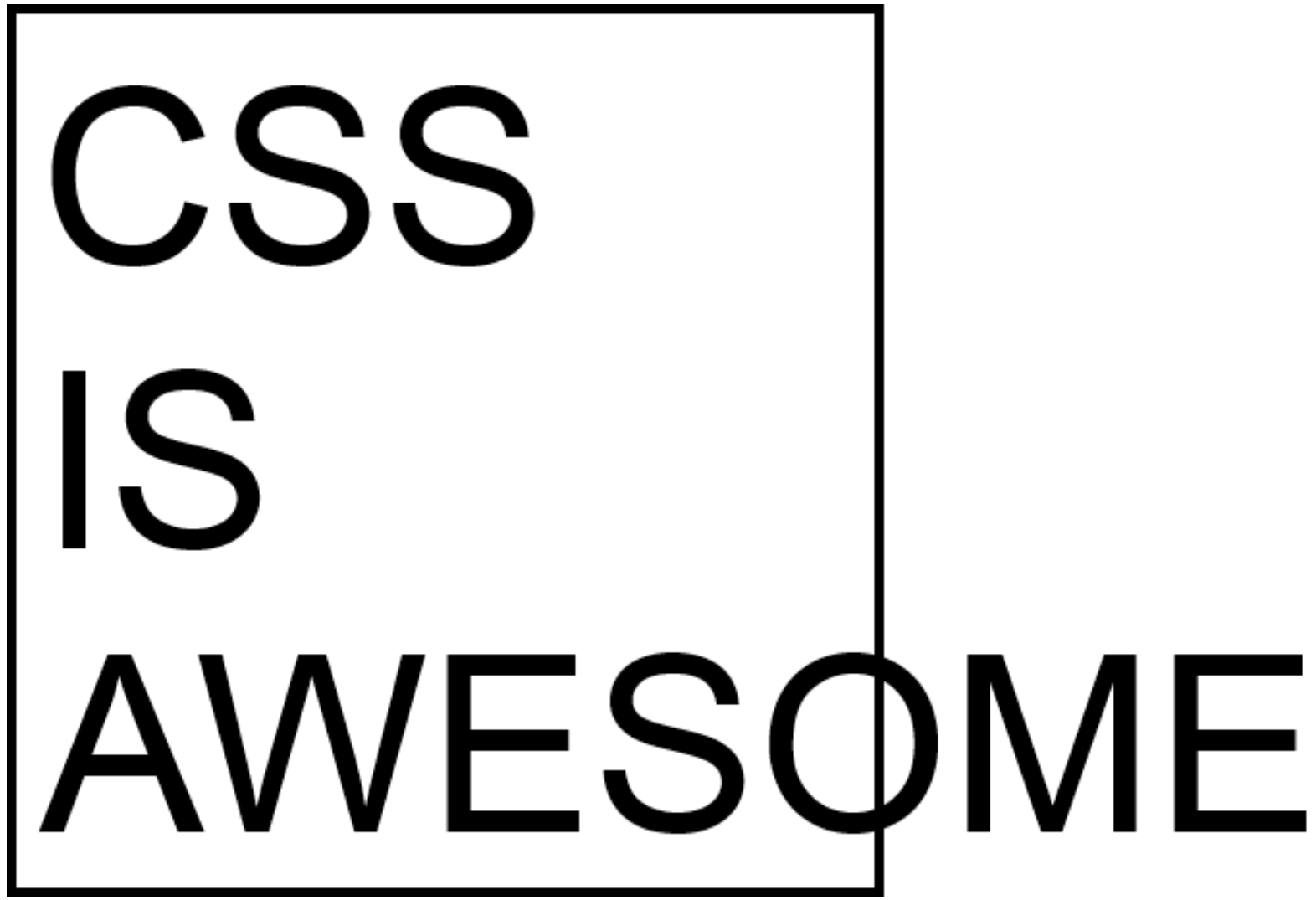
Responsive Web Apps

for the busy developer

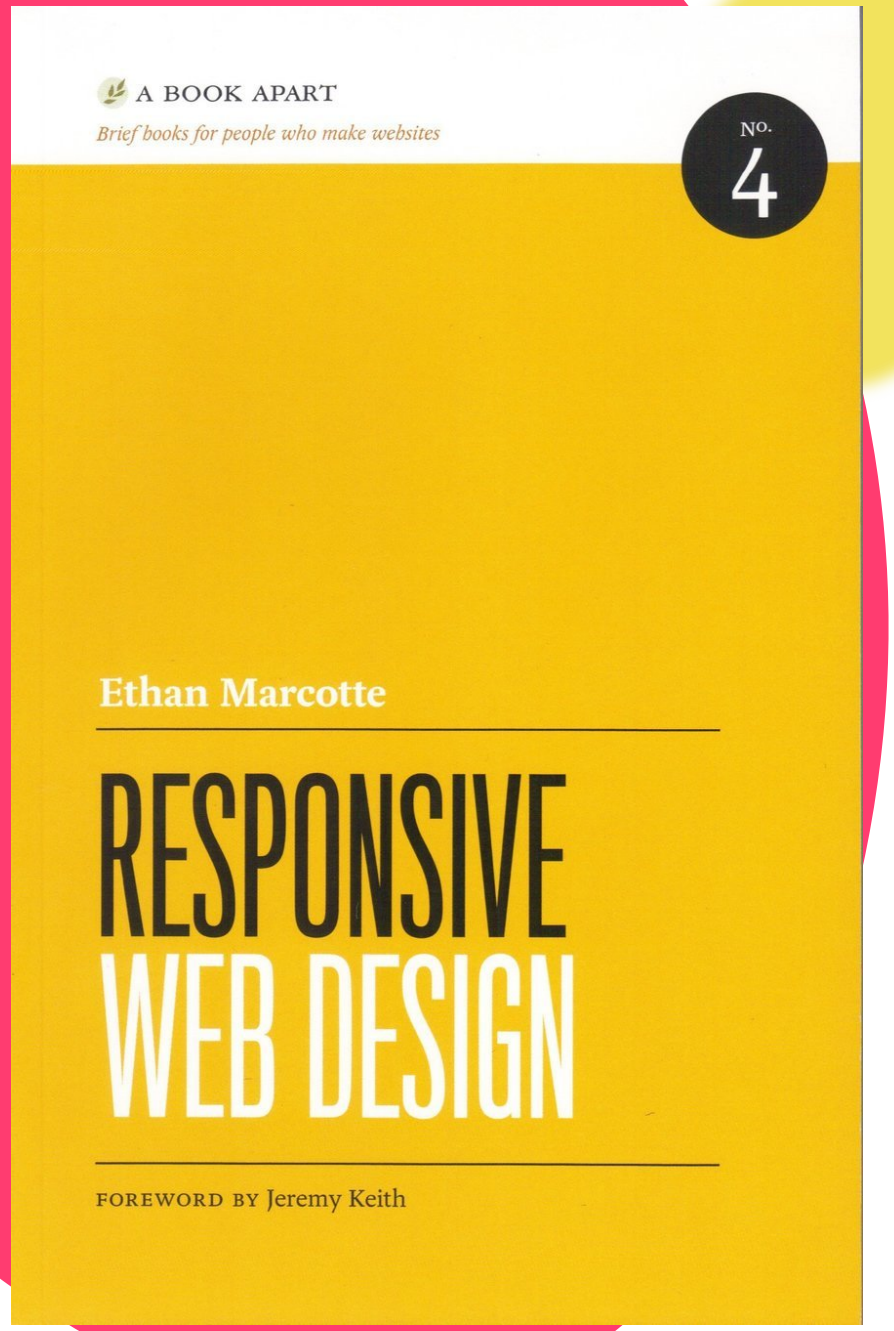
Simon MacDonald

@macdonst

What makes me the expert on Responsive Web Design?


CSS
IS
AWESOME

I read a book


A BOOK APART
Brief books for people who make websites
No. 4
Ethan Marcotte
RESPONSIVE WEB DESIGN
FOREWORD BY Jeremy Keith

What is responsive web design?

Responsive web design (**RWD**) or **responsive design** is an approach to web design that aims to make web pages render well on a variety of devices and window or screen sizes from minimum to maximum display size to ensure usability and satisfaction.

– Ethan Marcotte

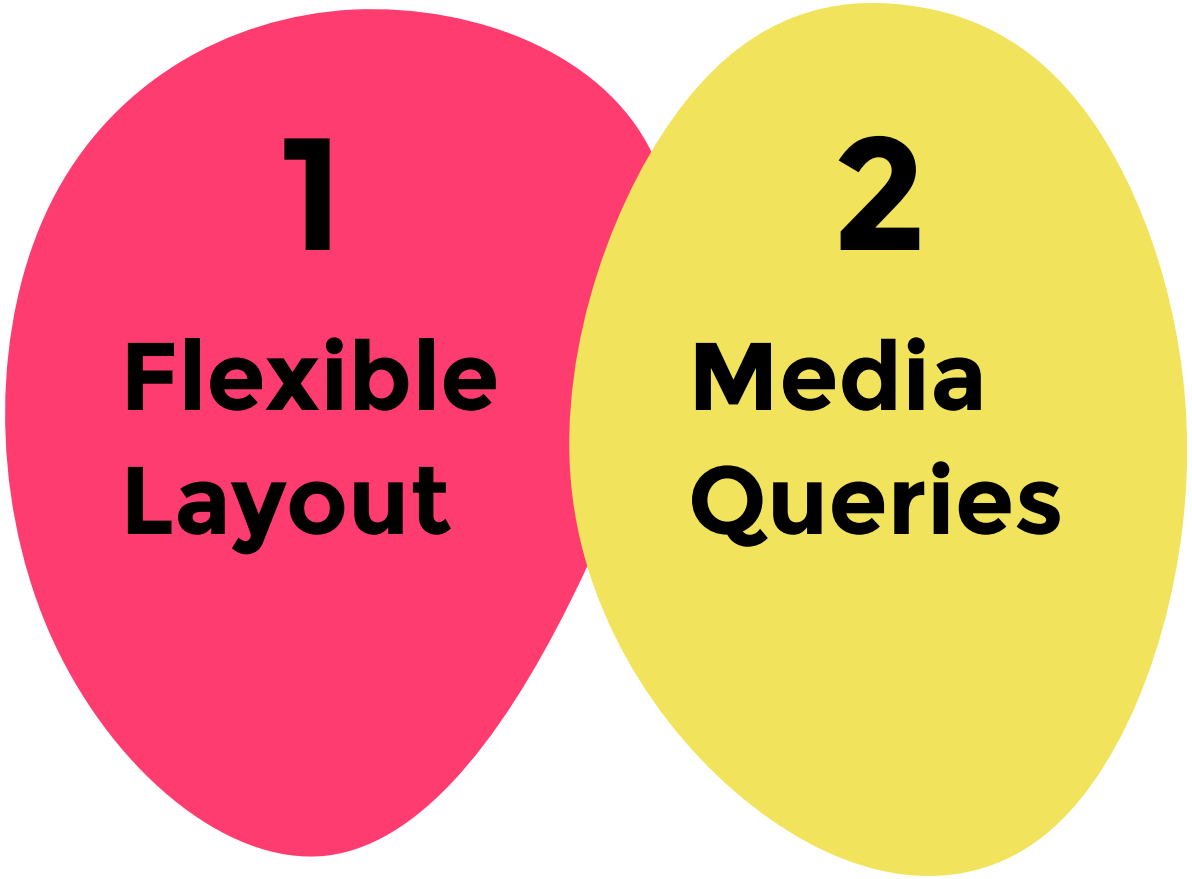
Pillars of RWD

Pillars of RWD

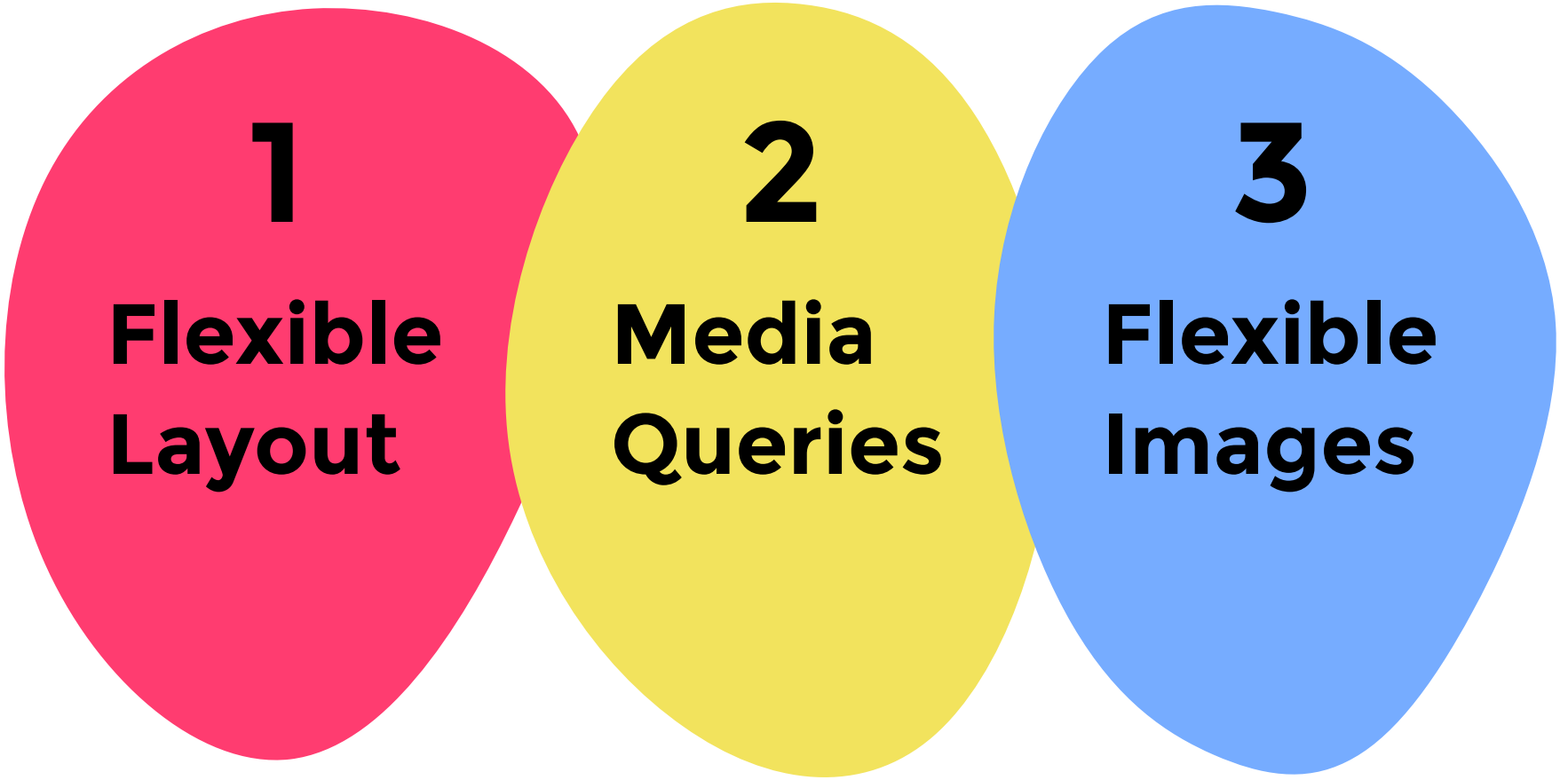
1

Flexible Layout

Pillars of RWD

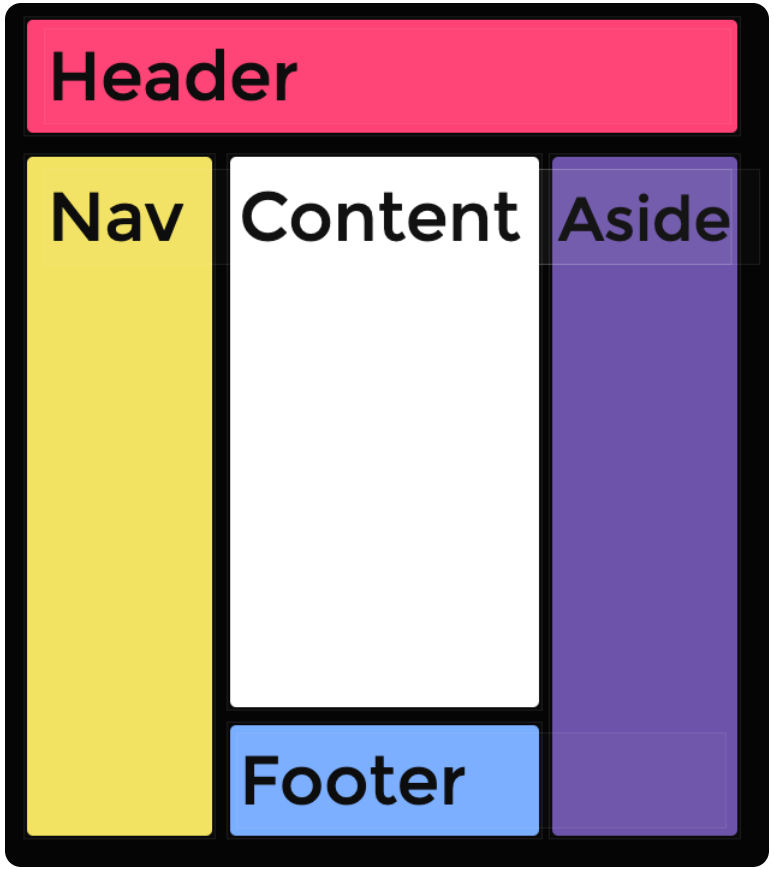


Pillars of RWD



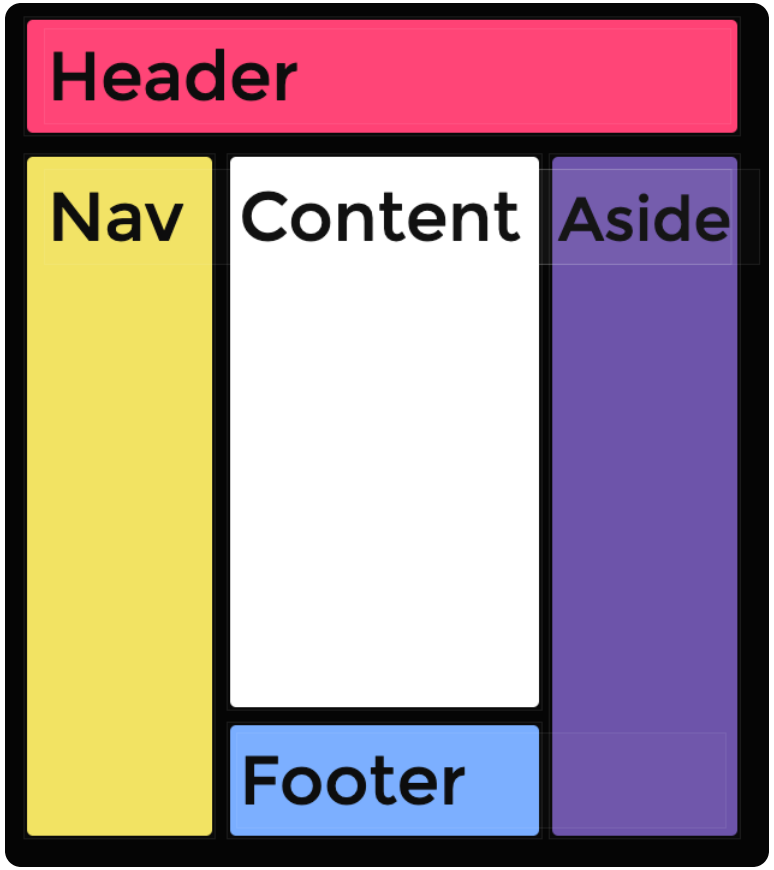
Flexible Layout

Legacy Layouts - Tables




```
<html>
  <body>
    <table>
      <tr>
        <td colspan="3">Header</td>
      </tr>
      <tr>
        <td rowspan="2"">Navigation</td>
        <td>Content</td>
        <td rowspan="2">Navigation</td>
      </tr>
      <tr>
        <td>Footer</td>
      </tr>
    </table>
  </body>
</html>
```

Legacy Layouts - Grid Systems




```
<div class="container">
  <div class="row">
    <div class="twelve columns">Header</div>
  </div>
  <div class="row">
    <div class="two columns">Nav</div>
    <div class="eight columns">
      <div class="container">
        <div class="row">
          <div class="twelve columns">Content</div>
        </div>
        <div class="row">
          <div class="twelve columns">Footer</div>
        </div>
      </div>
    </div>
    <div class="two columns">Aside</div>
  </div>
</div>
```


CSS
GRID

Can I Use CSS Grid?

CSS Grid Layout (level 1) - CR

Method of using a grid concept to lay out content, providing a mechanism for authors to divide available space for layout into columns and rows using a set of predictable sizing behaviors. Includes support for all `grid-*` properties and the `fr` unit.

Usage % of all users ?

Global 95.81% + 0.64% = 96.45%

unprefixed: 95.81%

Current aligned | Usage relative | Date relative | Filtered | All

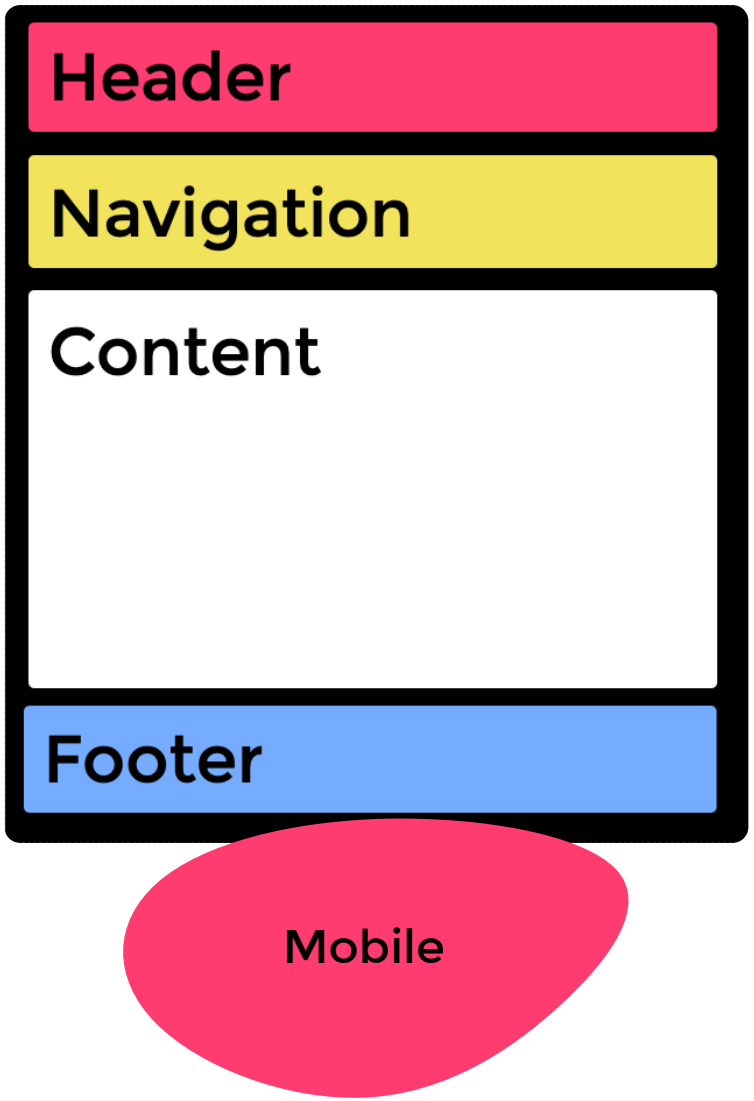
IE	Edge *	Firefox	Chrome	Safari	Opera	Safari on iOS *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baidu Browser	KaiOS Browser
		2-39	4-28													
		3 40-51	1 29-56		10-27											
6-9	2 12-15	4 52-53	4 57	3.1-10	1 28-43	3.2-10.2							4-5.4			
2 10	16-97	54-96	58-97	10.1-15.1	44-82	10.3-15.1		2.1-4.4.4	12-12.1				6.2-15.0			
2 11	98	97	98	15.3	83	15.3	all	98	64	98	96	12.12	16.0	10.4	7.12	2.5
		98-99	99-101	15.4-TP		15.4										

“ If you are having code problems I feel bad for you son.

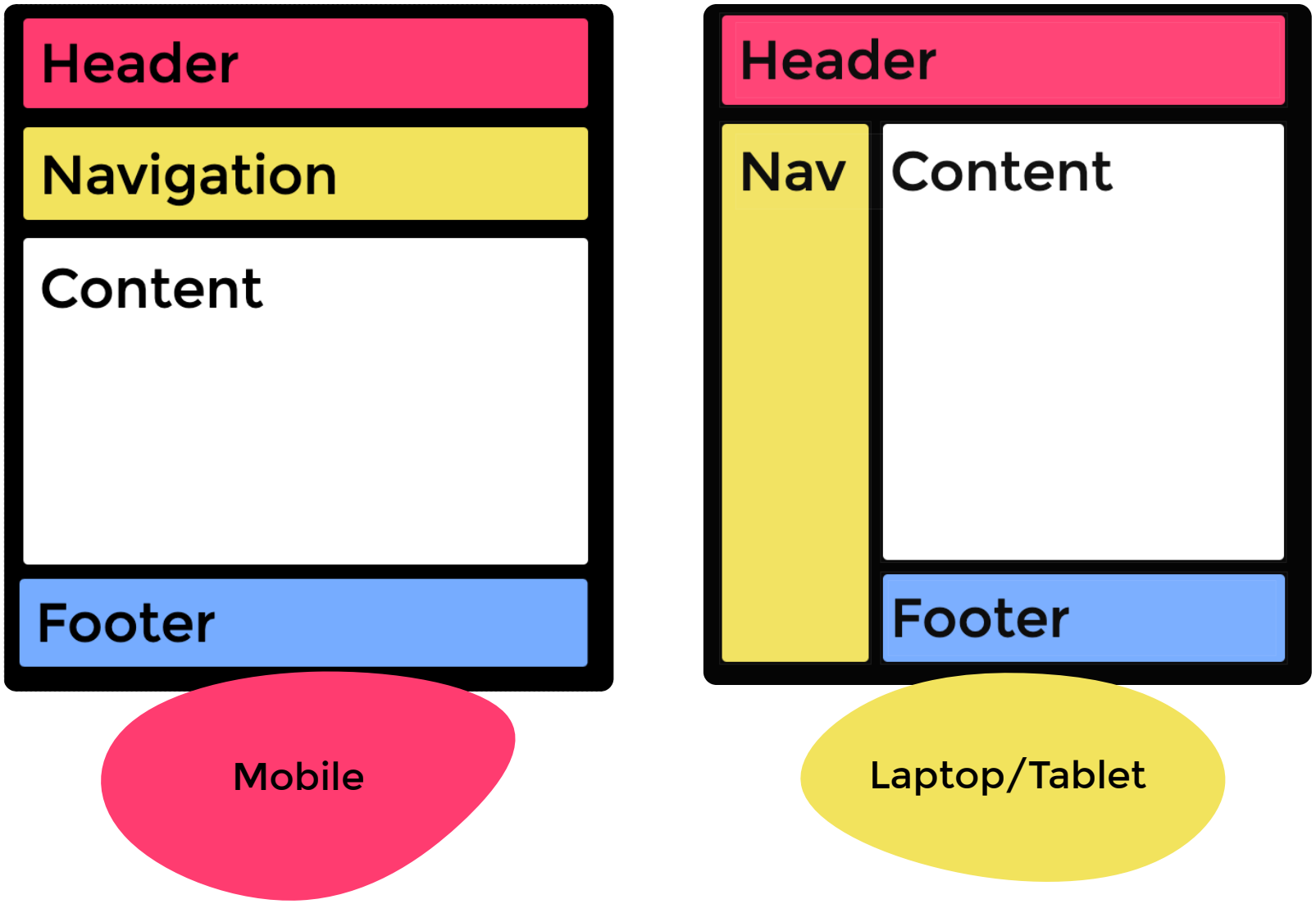
“I got 99 problems but IE ain't one.

Let's build some layouts

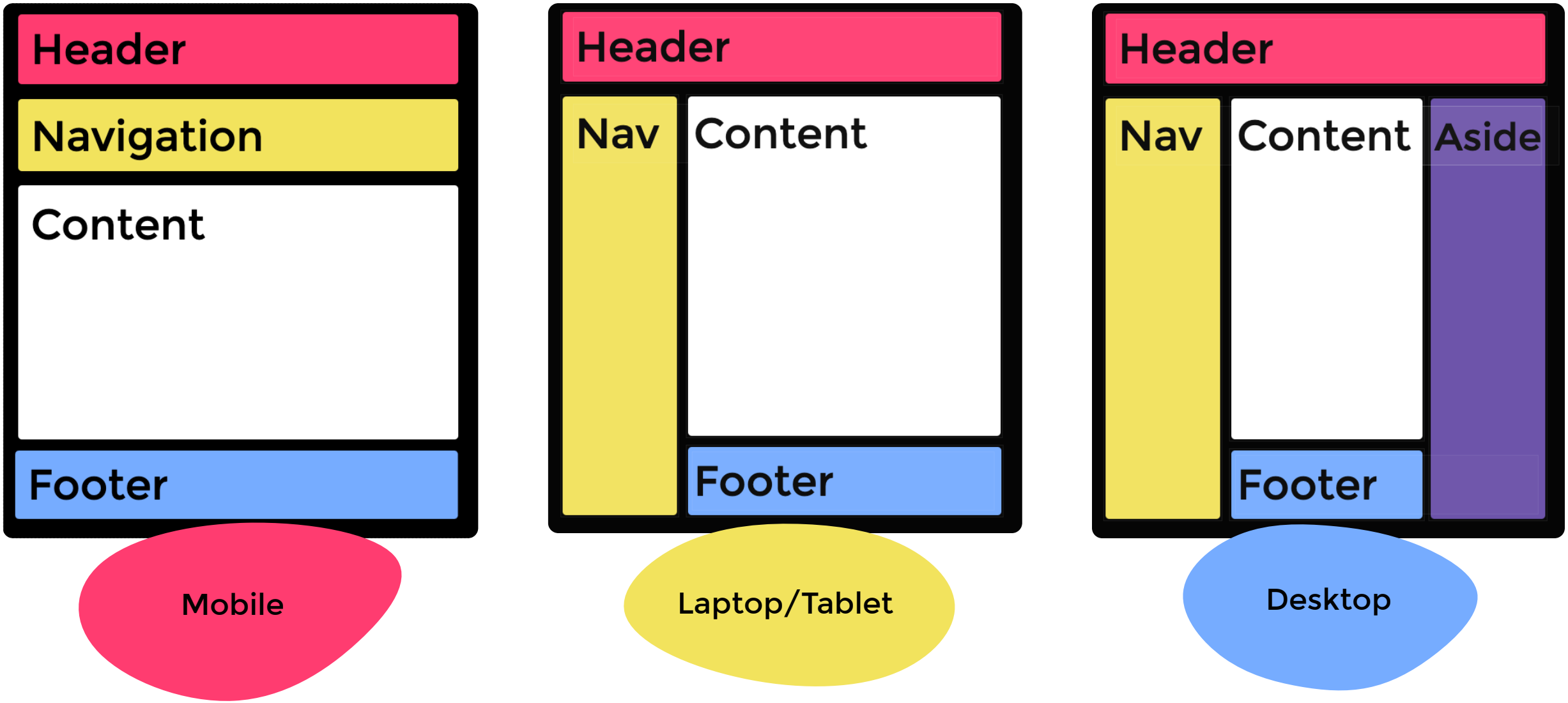
Let's build some layouts



Let's build some layouts



Let's build some layouts



Header

Navigation

Content

Footer

Block Layout

```css
body {
  display: grid;
  gap: 1em;
}
```

```html
<html>
  <body>
    <header>Header</header>
    <nav>Navigation</nav>
    <main>Content</main>
    <footer>Footer</footer>
  </body>
</html>
```

Header

Navigation

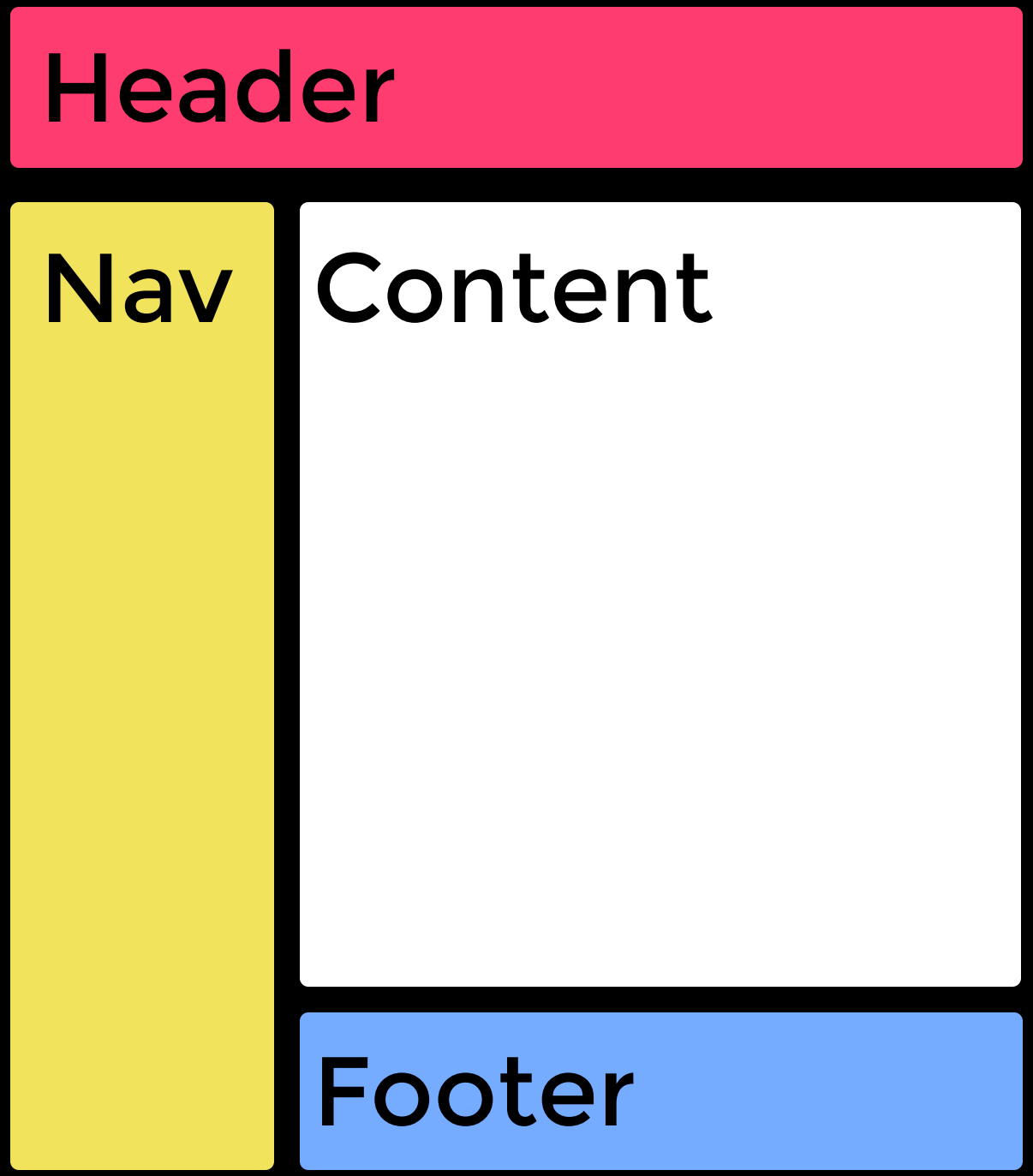
Content

Footer

Mobile Layout

```css
body {
  display: grid;
  gap: 1em;
  grid-template-rows: auto auto 1fr auto;
  min-height: 100vh;
}
```

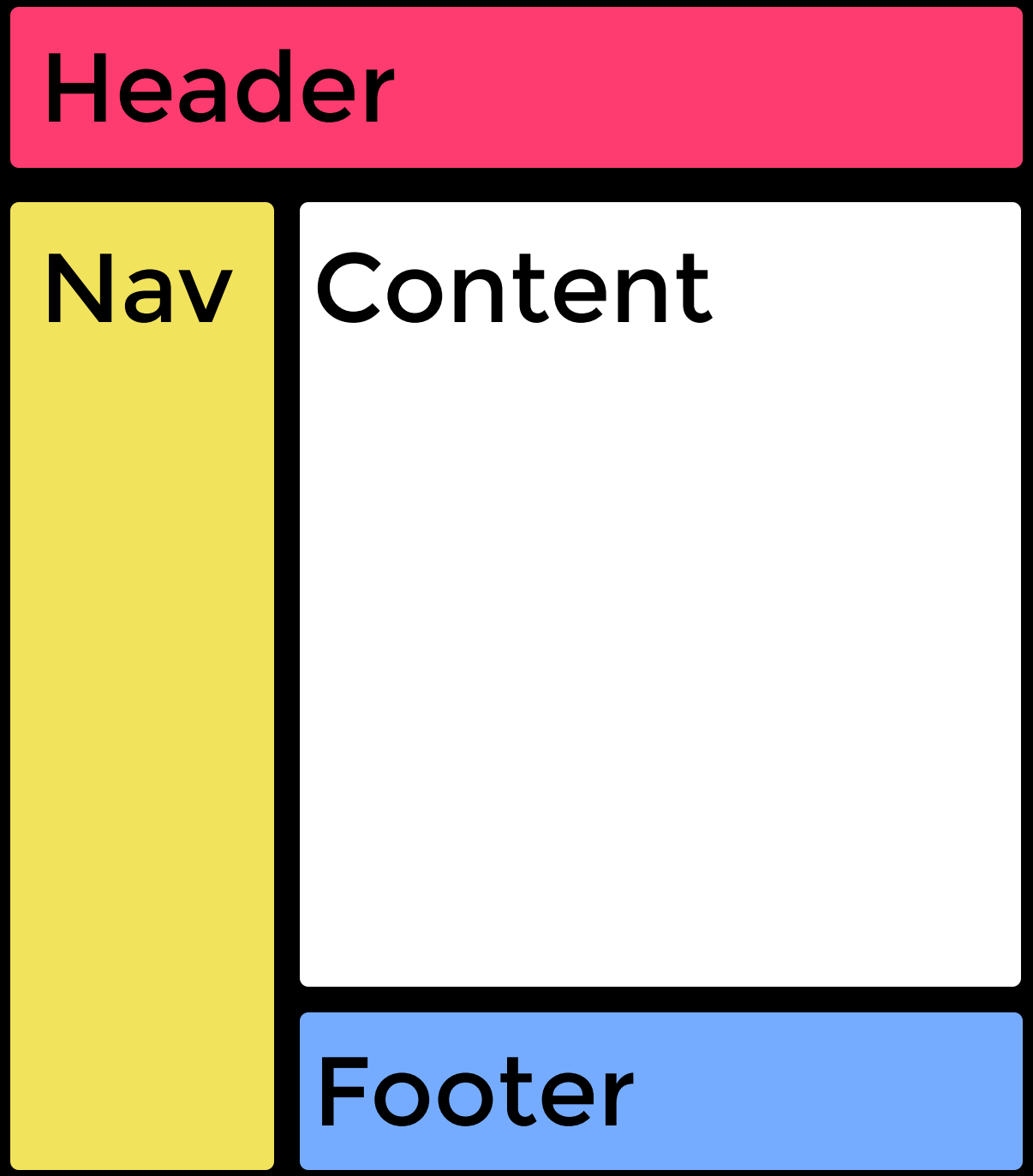
```html
<html>
  <body>
    <header>Header</header>
    <nav>Navigation</nav>
    <main>Content</main>
    <footer>Footer</footer>
  </body>
</html>
```



Laptop Layout

```css
body {
  display: grid;
  gap: 1em;
  grid-template-columns: 12em auto;
  grid-template-rows: auto 1fr auto;
}

header {
  grid-column: span 3;
}

nav {
  grid-row: span 2;
}
```

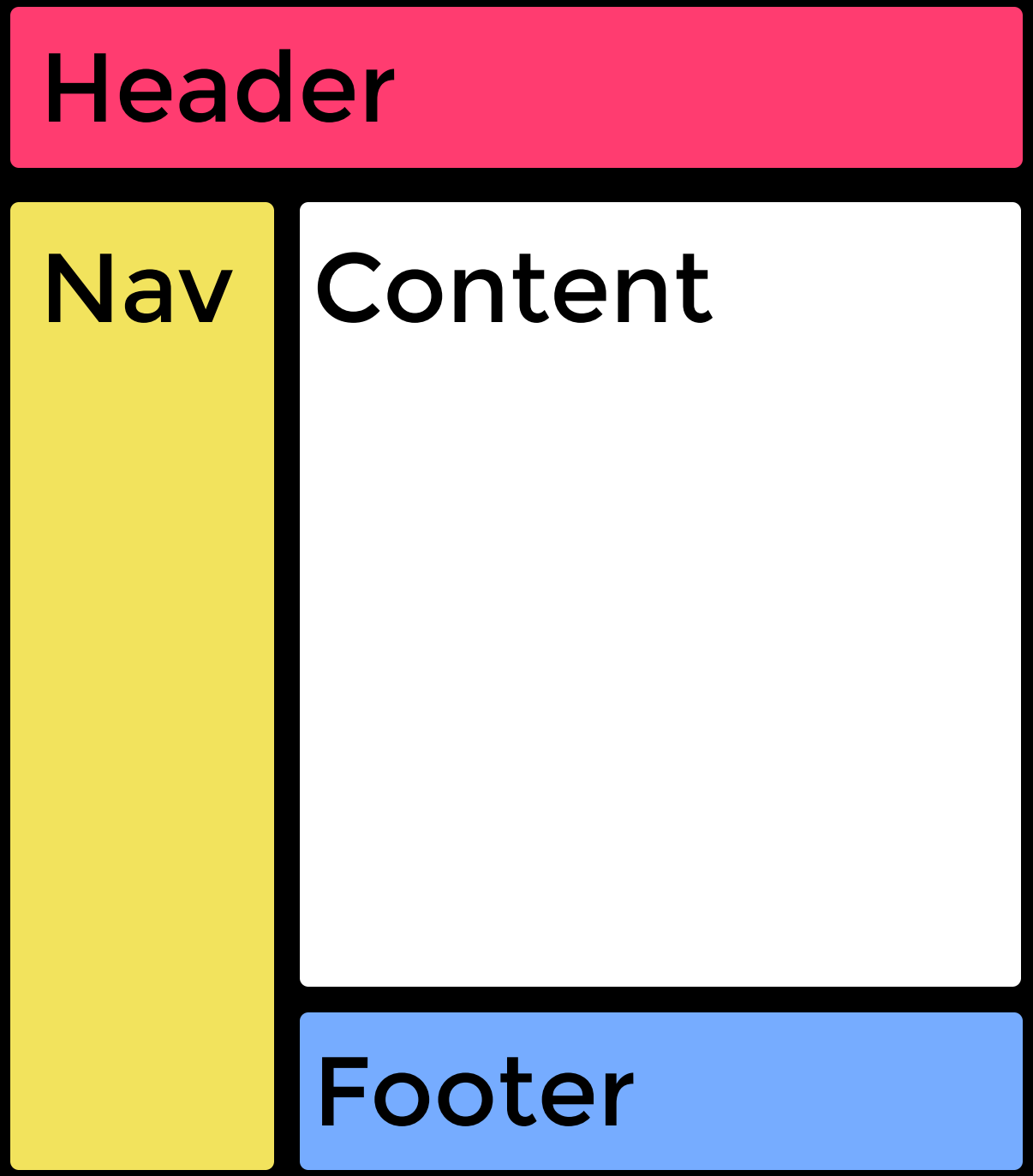
```html
<html>
  <body>
    <header>Header</header>
    <nav>Navigation</nav>
    <main>Content</main>
    <footer>Footer</footer>
  </body>
</html>
```



Laptop Layout

```css
body {
  display: grid;
  gap: 1em;
  grid-template-columns: 12em auto;
  grid-template-rows: auto 1fr auto;
}

header {
  grid-column: span 3;
}

nav {
  grid-row: span 2;
}
```

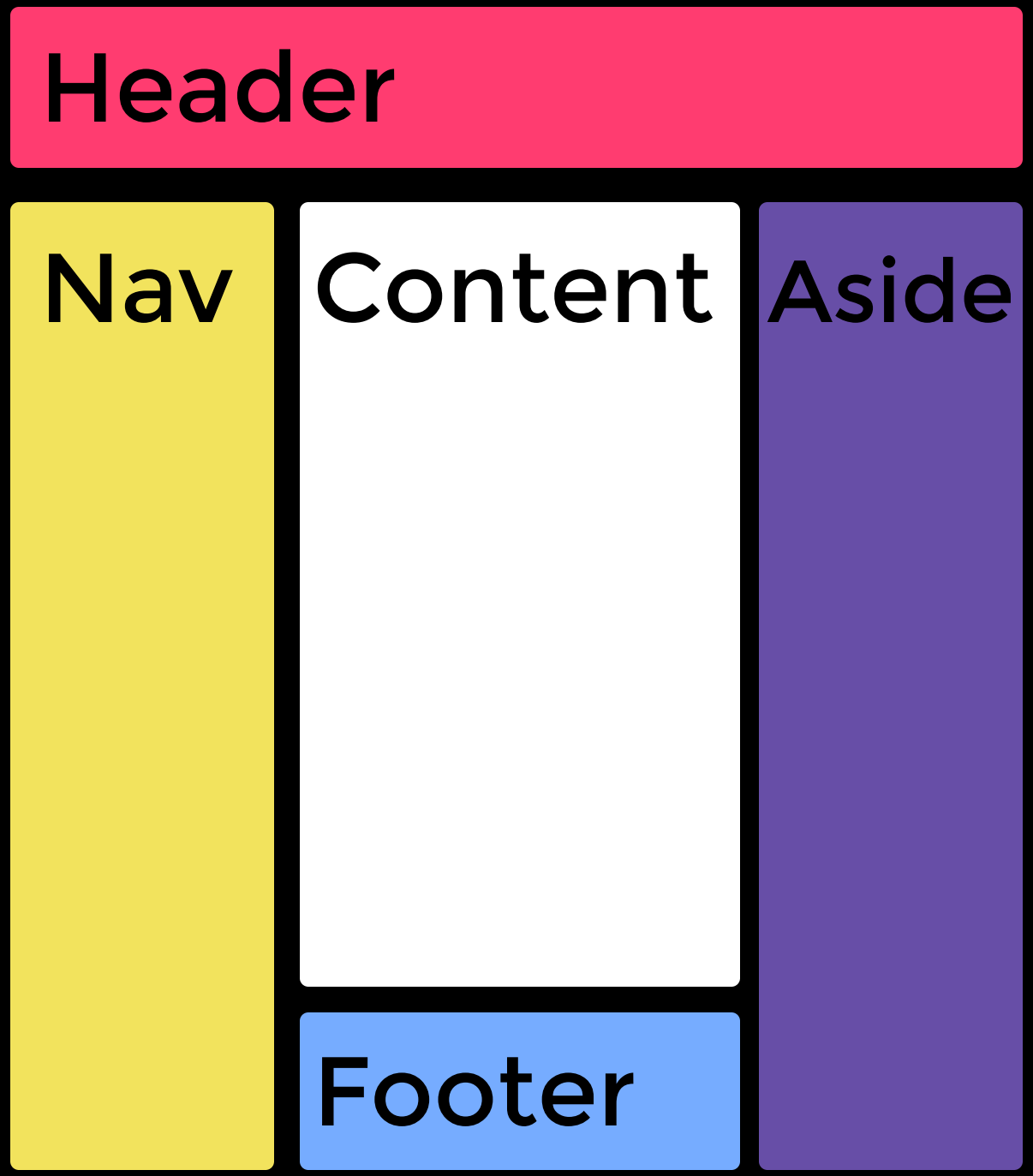
```html
<html>
  <body>
    <header>Header</header>
    <nav>Navigation</nav>
    <main>Content</main>
    <footer>Footer</footer>
  </body>
</html>
```



Laptop Layout

```css
body {
  display: grid;
  gap: 1em;
  grid-template-columns: 12em auto;
  grid-template-rows: auto 1fr auto;
}

header {
  grid-column: span 3;
}

nav {
  grid-row: span 2;
}
```

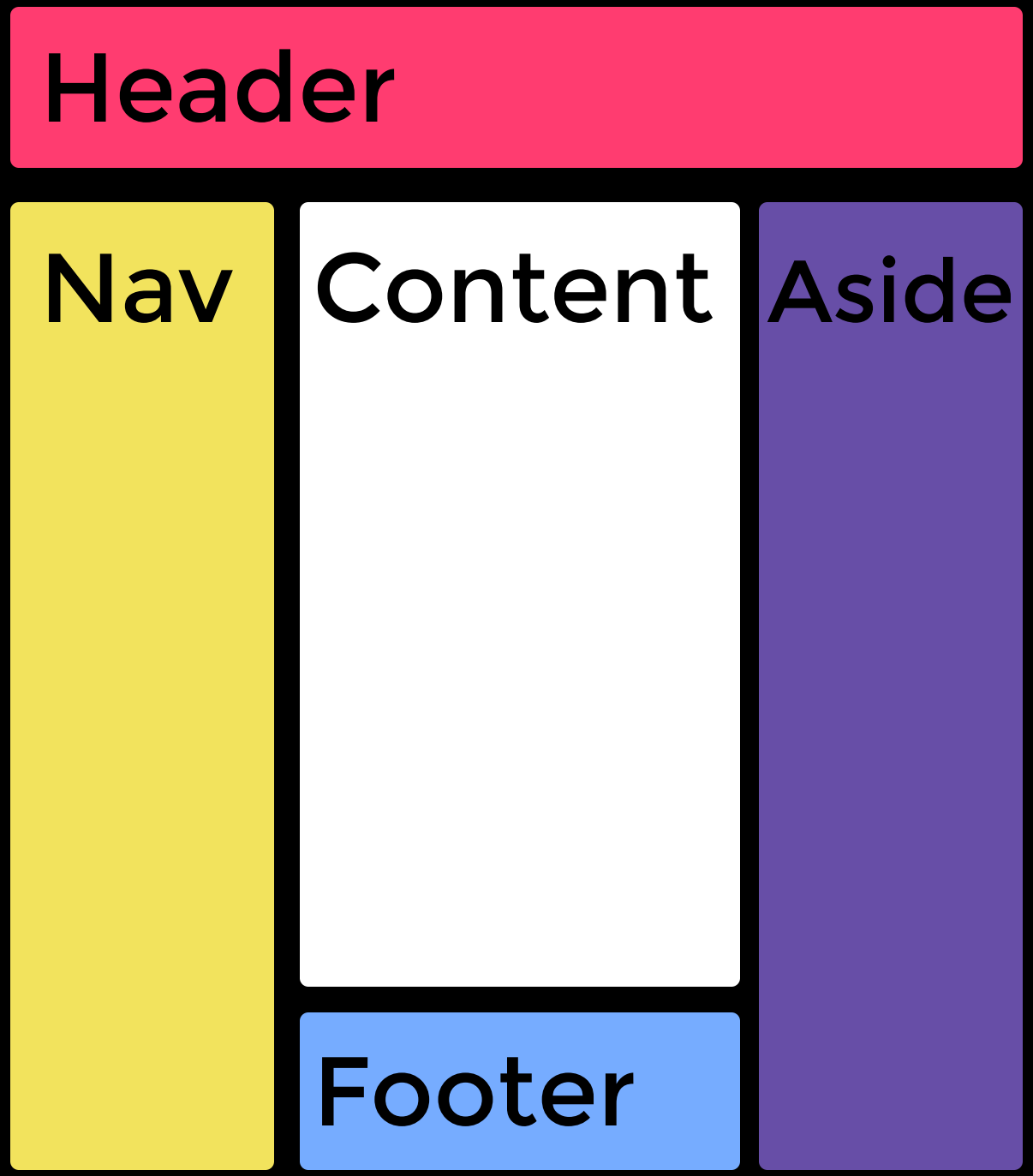
```html
<html>
  <body>
    <header>Header</header>
    <nav>Navigation</nav>
    <main>Content</main>
    <footer>Footer</footer>
  </body>
</html>
```



Desktop Layout

```css
body {
  display: grid;
  gap: 1em;
  grid-template-columns: 12em auto 12em;
  grid-template-rows: auto 1fr auto;
}

header {
  grid-column: span 3;
}

nav, aside {
  grid-row: span 2;
}
```

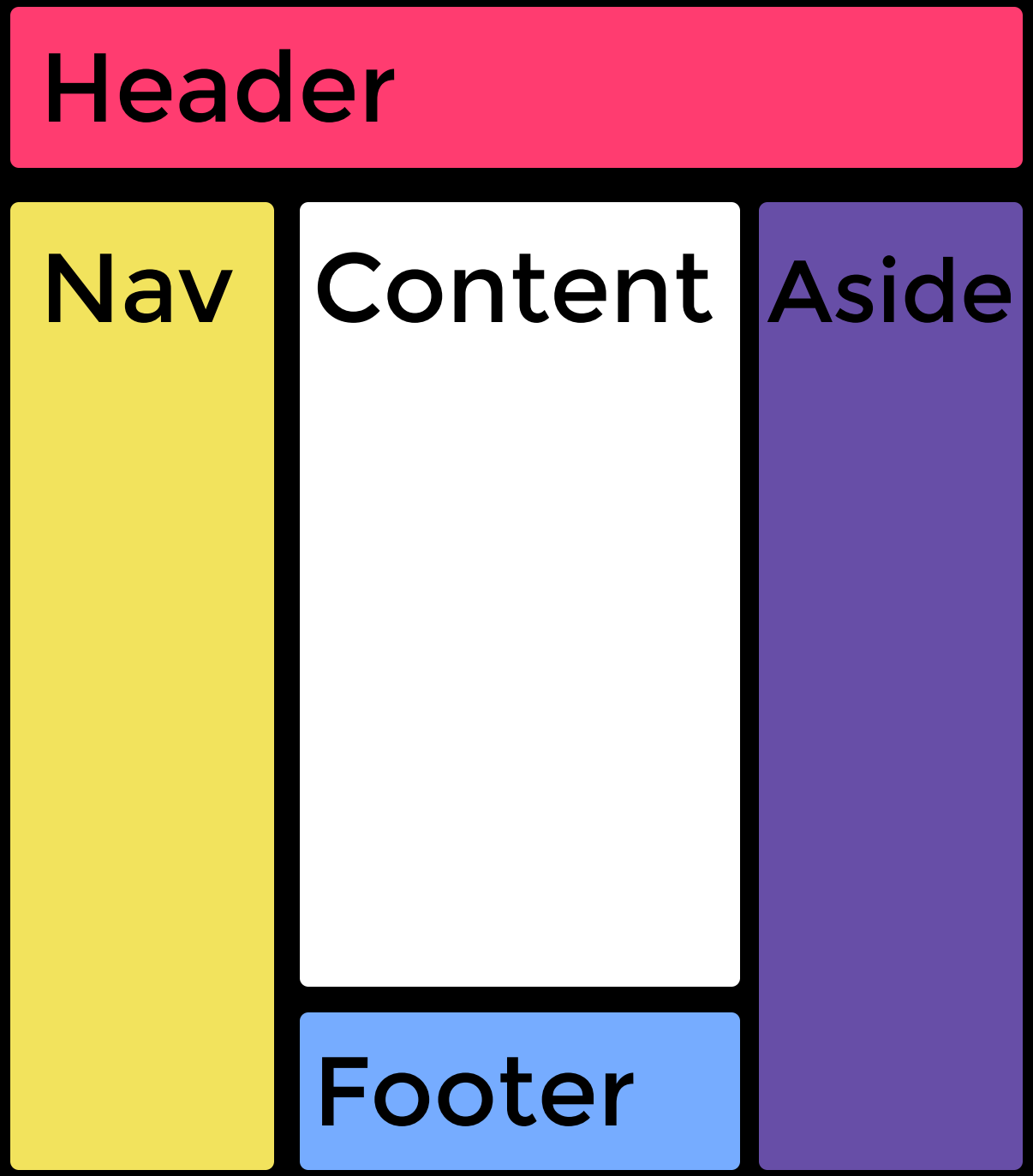
```html
<html>
  <body>
    <header>Header</header>
    <nav>Navigation</nav>
    <main>Content</main>
    <aside>Aside</aside>
    <footer>Footer</footer>
  </body>
</html>
```



Desktop Layout

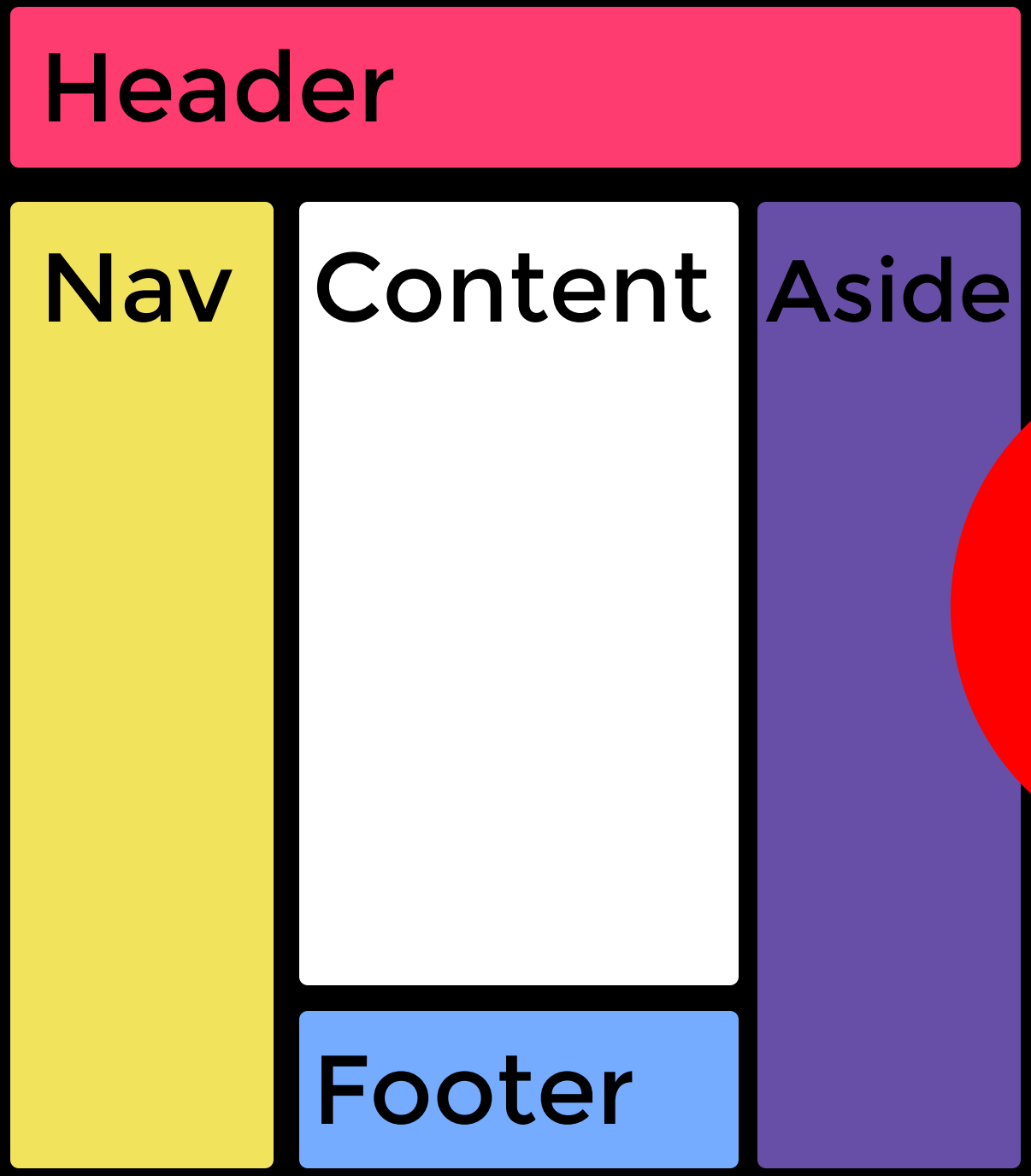
```css
body {
  display: grid;
  gap: 1em;
  grid-template-columns: 12em auto 12em;
  grid-template-rows: auto 1fr auto;
}

header {
  grid-column: span 3;
}

nav, aside {
  grid-row: span 2;
}
```

```html
<html>
  <body>
    <header>Header</header>
    <nav>Navigation</nav>
    <main>Content</main>
    <aside>Aside</aside>
    <footer>Footer</footer>
  </body>
</html>
```



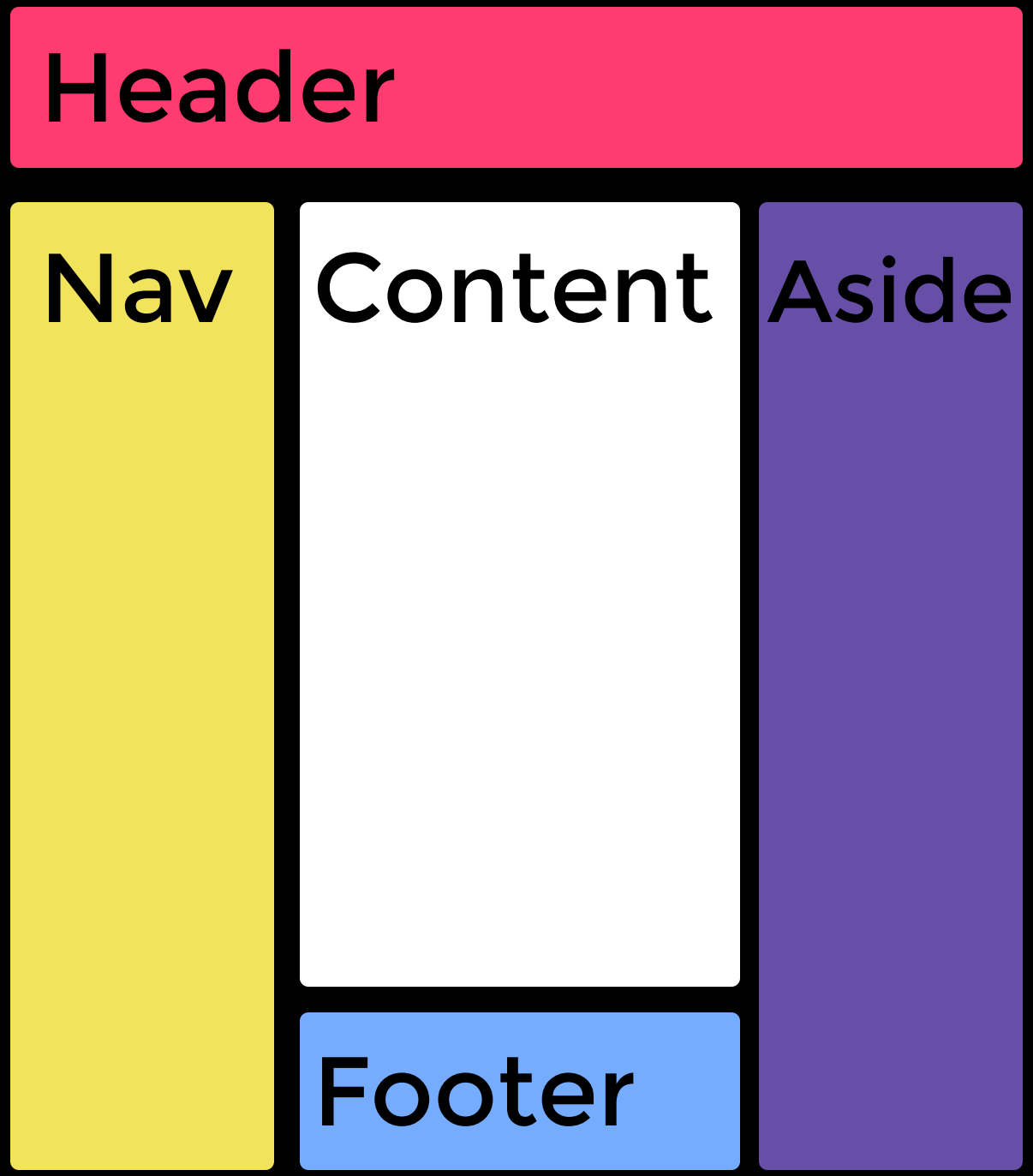
Line Based

```
1  body {
2    display: grid;
3    gap: 1em;
4    grid-template-columns: 12em auto 12em;
5    grid-template-rows: auto 1fr auto;
6  }
7
8  header {
9    grid-column: 1 / 4;
10   grid-row: 1;
11 }
12 nav {
13   grid-column: 1 / 2;
14   grid-row: 2 / 4;
15 }
16 main {
17   grid-column: 2 / 3;
18   grid-row: 2;
19 }
20 aside {
21   grid-column: 3 / 4;
22   grid-row: 2 / 4;
23 }
24 footer {
25   grid-column: 2 / 3;
26   grid-row: 3;
27 }
```



Line Based

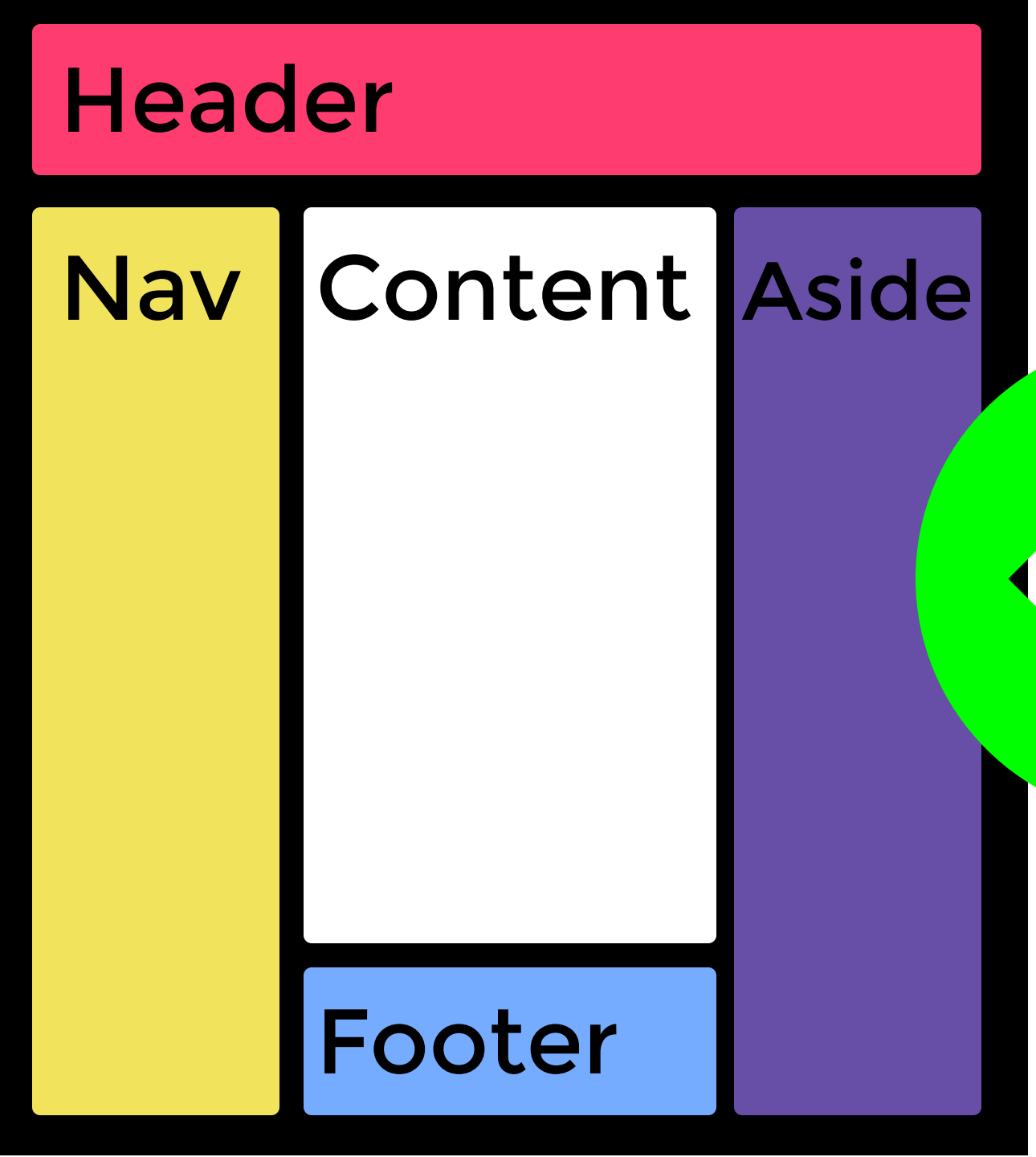
```
1  body {
2    display: grid;
3    gap: 1em;
4    grid-template-columns: 12em auto 12em;
     grid-template-rows: auto 1fr auto;
8
9      grid-column: 1 / 4;
10     grid-row: 1;
11   }
12
13     grid-column: 1 / 2;
14     grid-row: 2 / 4;
15   }
16
17     grid-column: 2 / 3;
18     grid-row: 2;
19
20 aside {
21   grid-column: 3 / 4;
22   grid-row: 2 / 4;
23 }
24 footer {
25   grid-column: 2 / 3;
26   grid-row: 3;
27 }
```



Grid Areas

```css
body {
  display: grid;
  gap: 1em;
  grid-template-columns: 12em auto 12em;
  grid-template-rows: auto 1fr auto;
  grid-template-areas:
    'header header header'
    'nav main aside'
    'nav footer aside';
}

header { grid-area: header; }

nav { grid-area: nav; }

main { grid-area: main; }

aside { grid-area: aside; }

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

Grid Areas




```
1  body {
2    display: grid;
3    gap: 1em;
     grid-template-columns: 12em auto 12em;
     id-template-rows: auto 1fr auto;
     template-areas:
     er header header'
     ain aside'
     oter aside';
   }
11
12 d-area: header; }
13
   area: nav; }

   id-area: main; }

   { grid-area: aside; }

20 footer { grid-area: footer; }
```

Media Queries

What is a media query?

Media queries allow you to adapt your site or app depending on the presence of various device characteristics such as:

Media queries allow you to adapt your site or app depending on the presence of various device characteristics such as:

- Screen resolution

Media queries allow you to adapt your site or app depending on the presence of various device characteristics such as:

- Screen resolution
- Screen orientation

Media queries allow you to adapt your site or app depending on the presence of various device characteristics such as:

- Screen resolution
- Screen orientation
- User preferences like color scheme or reduced motion

Media queries allow you to adapt your site or app depending on the presence of various device characteristics such as:

- Screen resolution
- Screen orientation
- User preferences like color scheme or reduced motion
- etc.

Can I Use Media Queries?

CSS3 Media Queries - REC

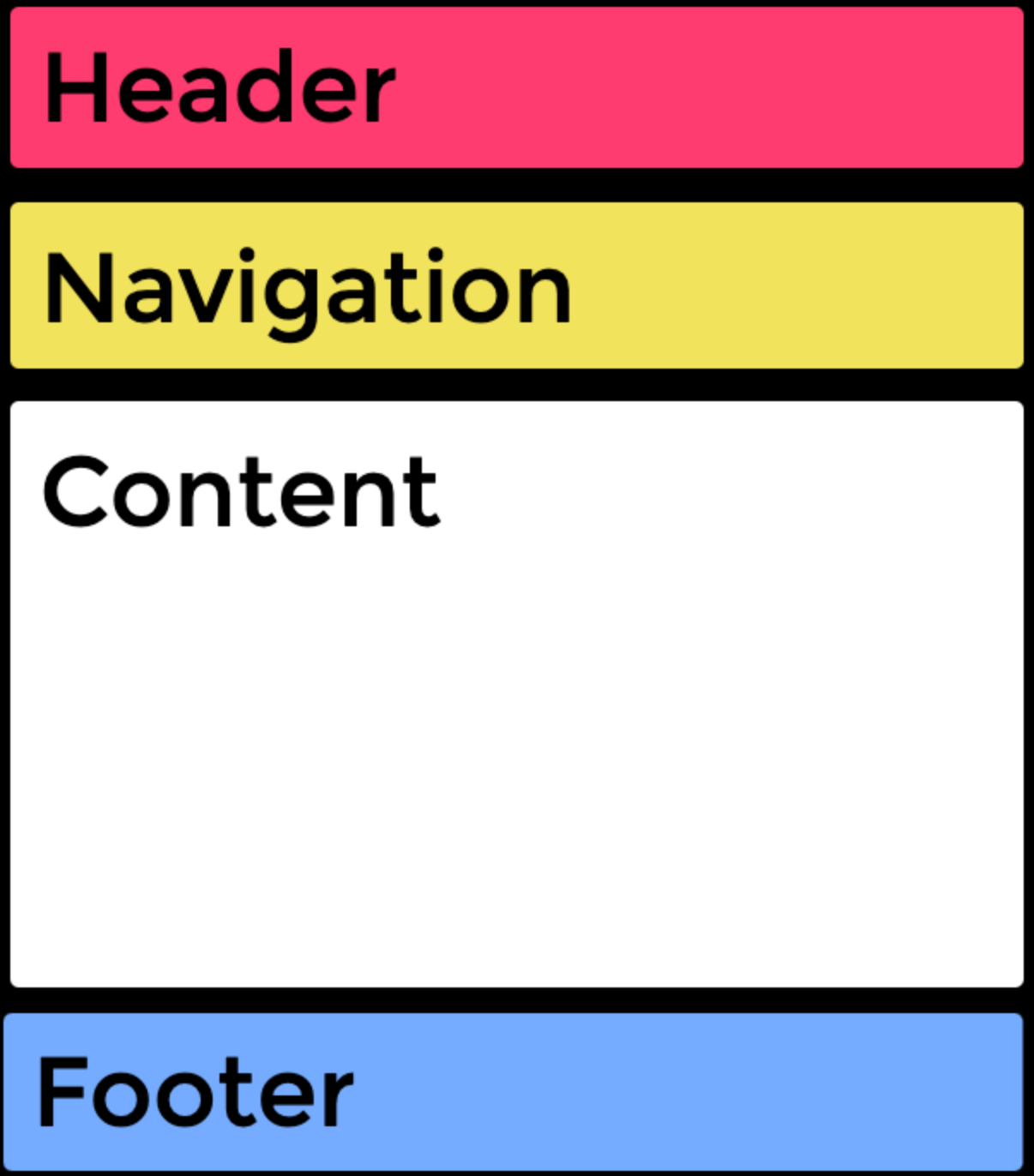
Method of applying styles based on media information. Includes things like page and device dimensions

Usage: % of all users ?

Global: 99.52% + 0.01% = 99.53%

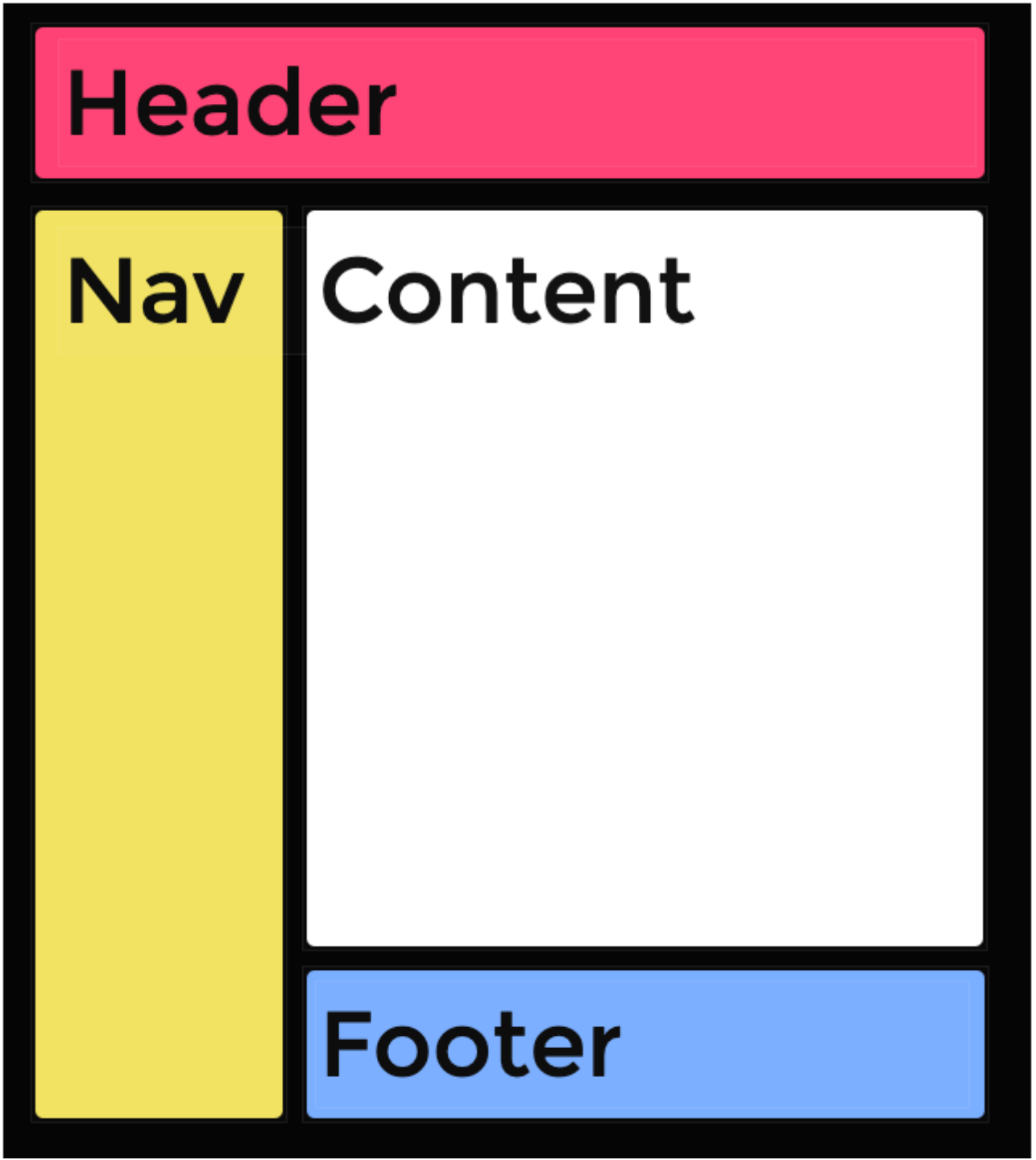
Current aligned | Usage relative | Date relative | Filtered | All

IE	Edge *	Firefox	Chrome	Safari	Opera	Safari on iOS *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baidu Browser	KaiOS Browser
				1 2 3.1-3.2												
6-8		2-3	1 4-25	1 4-6		1 3.2-6.1		1 2.1-4.3								
1 9-10	12-97	3.5-96	26-97	6.1-15.1	10-82	7-15.1		4.4-4.4.4	12-12.1				4-15.0			
1 11	98	97	98	15.3	83	15.3	all	98	64	98	96	12.12	16.0	10.4	7.12	2.5
		98-99	99-101	15.4-TP		15.4										



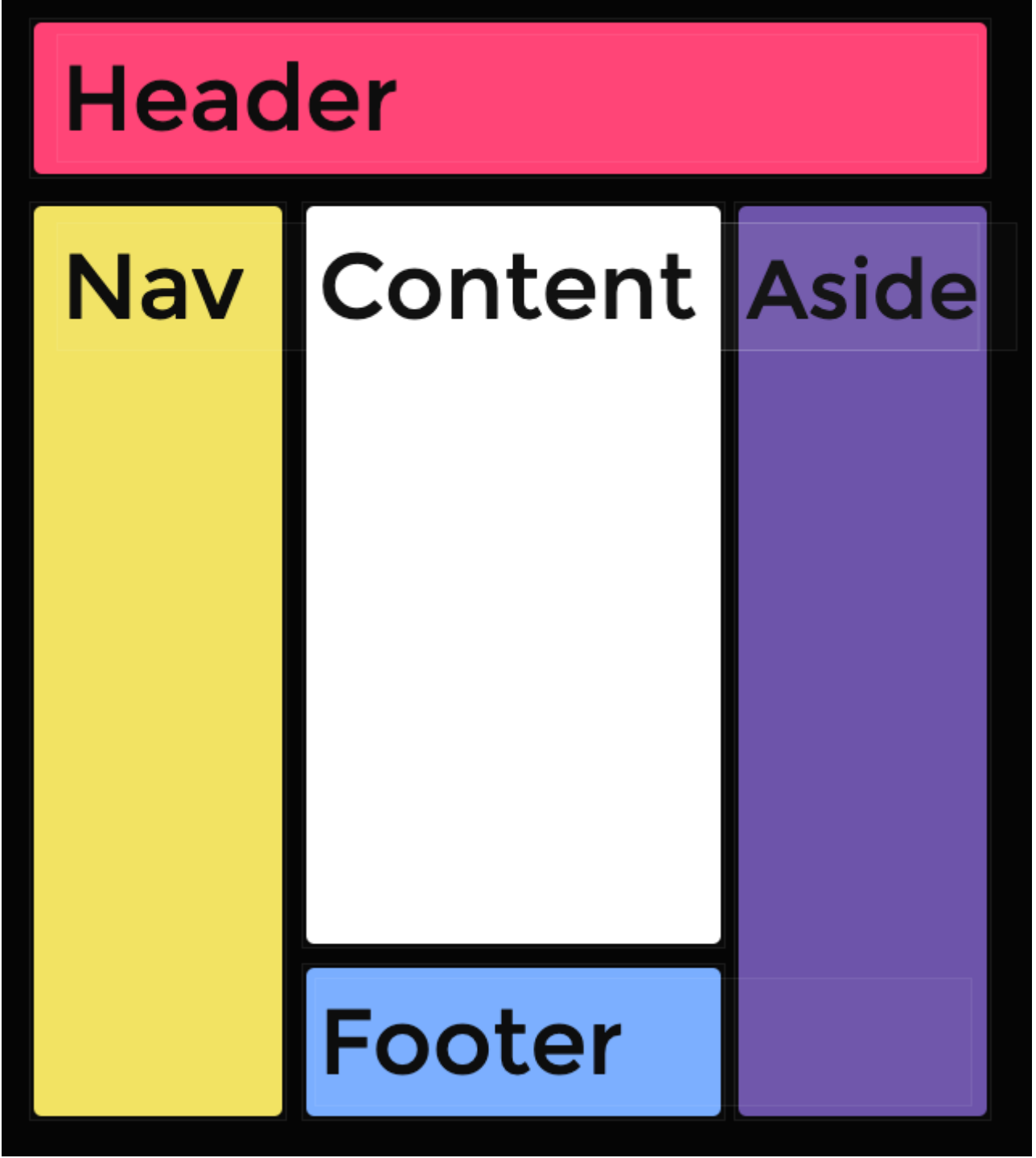
Three Layouts

```css
body {
  display: grid;
  gap: 1em;
  grid-template-rows: auto auto 1fr auto;
  grid-template-areas:
    'header'
    'nav'
    'main'
    'footer';
}

@media (min-width: 40em) {
  body {
    grid-template-columns: 12em auto;
    grid-template-rows: auto 1fr auto;
    grid-template-areas:
      'header header'
      'nav main'
      'nav footer';
  }
}

@media (min-width: 60em) {
  body {
    grid-template-columns: 12em auto 12em;
    grid-template-rows: auto 1fr auto;
    grid-template-areas:
      'header header header'
      'nav main aside'
      'nav footer aside';
  }
  aside { display: block; }
}
```



Three Layouts

```css
body {
  display: grid;
  gap: 1em;
  grid-template-rows: auto auto 1fr auto;
  grid-template-areas:
    'header'
    'nav'
    'main'
    'footer';
}

@media (min-width: 40em) {
  body {
    grid-template-columns: 12em auto;
    grid-template-rows: auto 1fr auto;
    grid-template-areas:
      'header header'
      'nav main'
      'nav footer';
  }
}

@media (min-width: 60em) {
  body {
    grid-template-columns: 12em auto 12em;
    grid-template-rows: auto 1fr auto;
    grid-template-areas:
      'header header header'
      'nav main aside'
      'nav footer aside';
  }
  aside { display: block; }
}
```



Three Layouts

```css
body {
  display: grid;
  gap: 1em;
  grid-template-rows: auto auto 1fr auto;
  grid-template-areas:
    'header'
    'nav'
    'main'
    'footer';
}

@media (min-width: 40em) {
  body {
    grid-template-columns: 12em auto;
    grid-template-rows: auto 1fr auto;
    grid-template-areas:
      'header header'
      'nav main'
      'nav footer';
  }
}

@media (min-width: 60em) {
  body {
    grid-template-columns: 12em auto 12em;
    grid-template-rows: auto 1fr auto;
    grid-template-areas:
      'header header header'
      'nav main aside'
      'nav footer aside';
  }
  aside { display: block; }
}
```

A Note About Breakpoints

I don't have a magic formula for selecting breakpoints.

A Note About Breakpoints

I don't have a magic formula for selecting breakpoints.

You only need to consider them when your design **breaks** at different screen resolutions.

A Note About Breakpoints

I don't have a magic formula for selecting breakpoints.

You only need to consider them when your design **breaks** at different screen resolutions.

You can look at what large companies like Google and Microsoft suggest.

A Note About Breakpoints

I don't have a magic formula for selecting breakpoints.

You only need to consider them when your design **breaks** at different screen resolutions.

You can look at what large companies like Google and Microsoft suggest.

However, I find a good rule of thumb is to select two break points to give your site or app 3 distinct layouts.

A Note About Breakpoints

I don't have a magic formula for selecting breakpoints.

You only need to consider them when your design **breaks** at different screen resolutions.

You can look at what large companies like Google and Microsoft suggest.

However, I find a good rule of thumb is to select two break points to give your site or app 3 distinct layouts.

- Small - up to 27.5em (approximately 440px)

A Note About Breakpoints

I don't have a magic formula for selecting breakpoints.

You only need to consider them when your design **breaks** at different screen resolutions.

You can look at what large companies like Google and Microsoft suggest.

However, I find a good rule of thumb is to select two break points to give your site or app 3 distinct layouts.

- Small - up to 27.5em (approximately 440px)
- Medium - between 27.5 em and 60em (441-960px)

A Note About Breakpoints

I don't have a magic formula for selecting breakpoints.

You only need to consider them when your design **breaks** at different screen resolutions.

You can look at what large companies like Google and Microsoft suggest.

However, I find a good rule of thumb is to select two break points to give your site or app 3 distinct layouts.

- Small - up to 27.5em (approximately 440px)
- Medium - between 27.5 em and 60em (441-960px)
- Large - over 60em (961+ px)

Flexible Images

Header

Navigation

Content

Footer

Just send a large image down the wire and set it's width to 100%. Works everywhere! 👏

Header

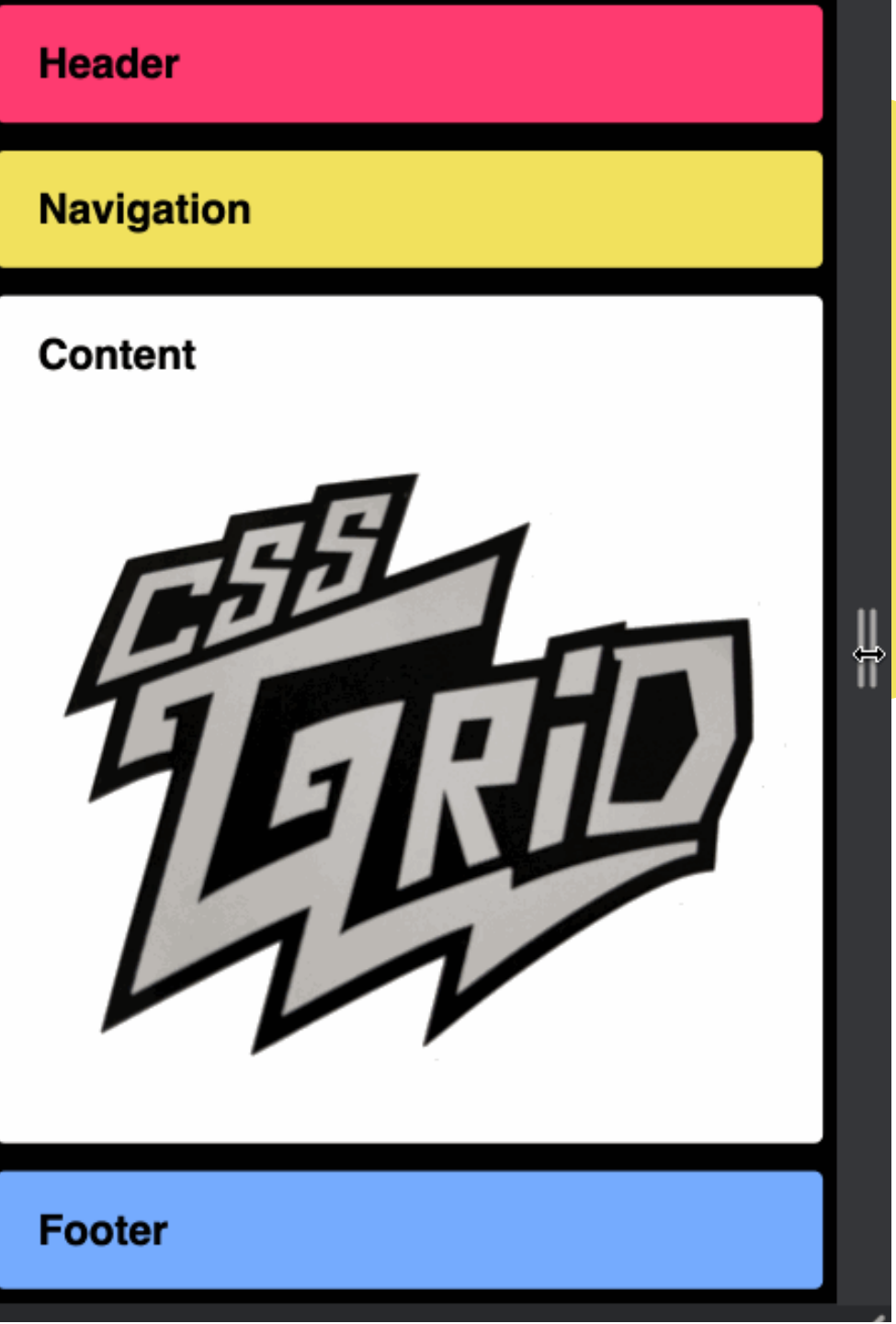
Navigation

Content

Footer

Just send a large image down the wire and set it's width to 100%. Works everywhere! 👏

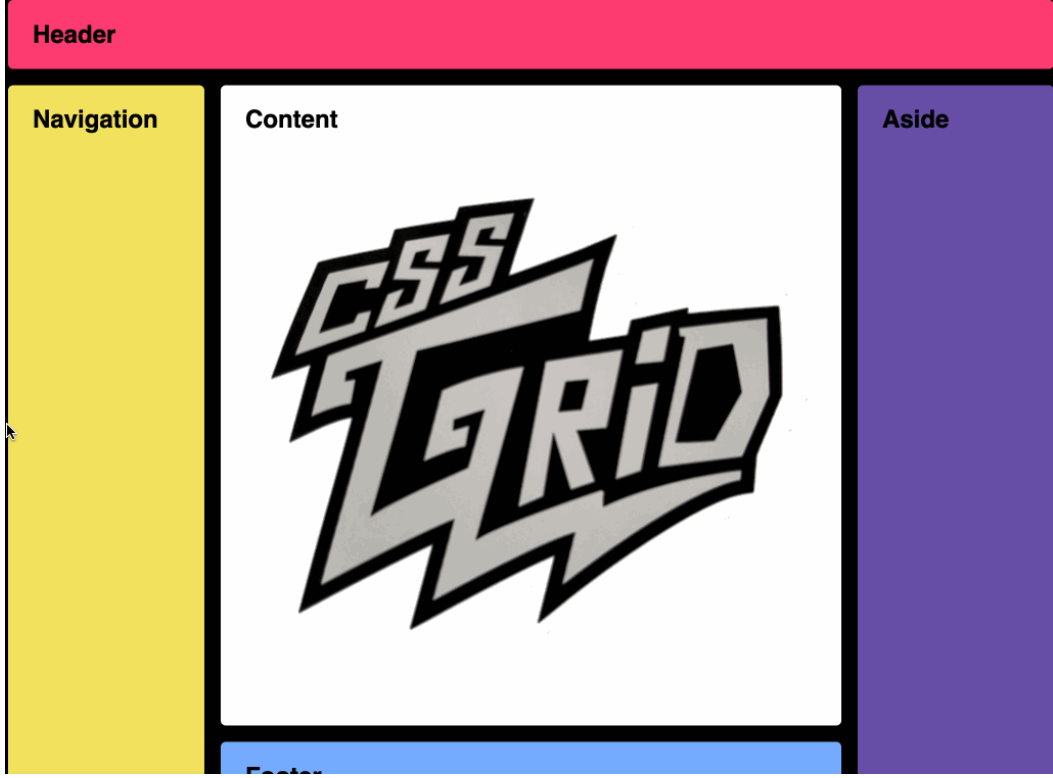
- Except it wastes bandwidth to send a large image when not required 👎



Just send a large image down the wire and set it's width to 100%. Works everywhere! 👏

- Except it wastes bandwidth to send a large image when not required 👎
- But if you pick too small an image it will look grainy when blown up to larger sizes 👎👎

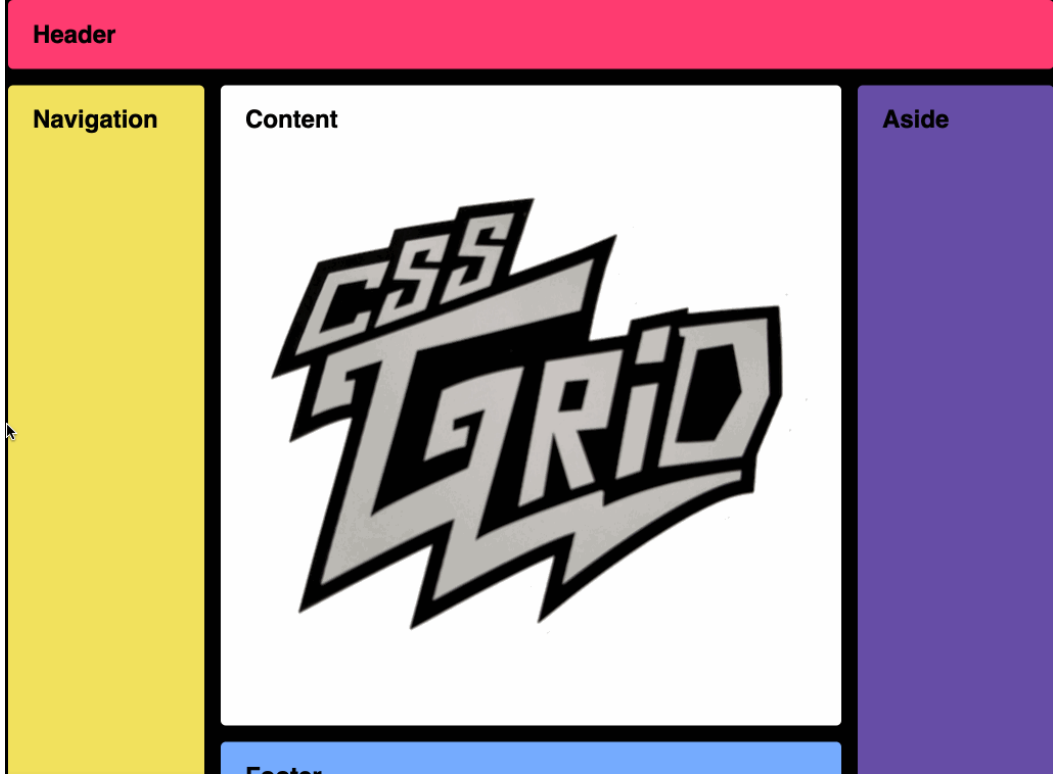
Let the browser choose the right sized image for the device.




```
<img
  sizes="
    (min-width: 960px) 600px,
    (min-width: 640px) 300px,
    150px
  "
  srcset="
    css-grid-mobile.png 150w,
    css-grid-laptop.png 300w,
    css-grid.png 600w
  "
  src="css-grid.png"
/>
```


Let the browser choose the right sized image for the device.

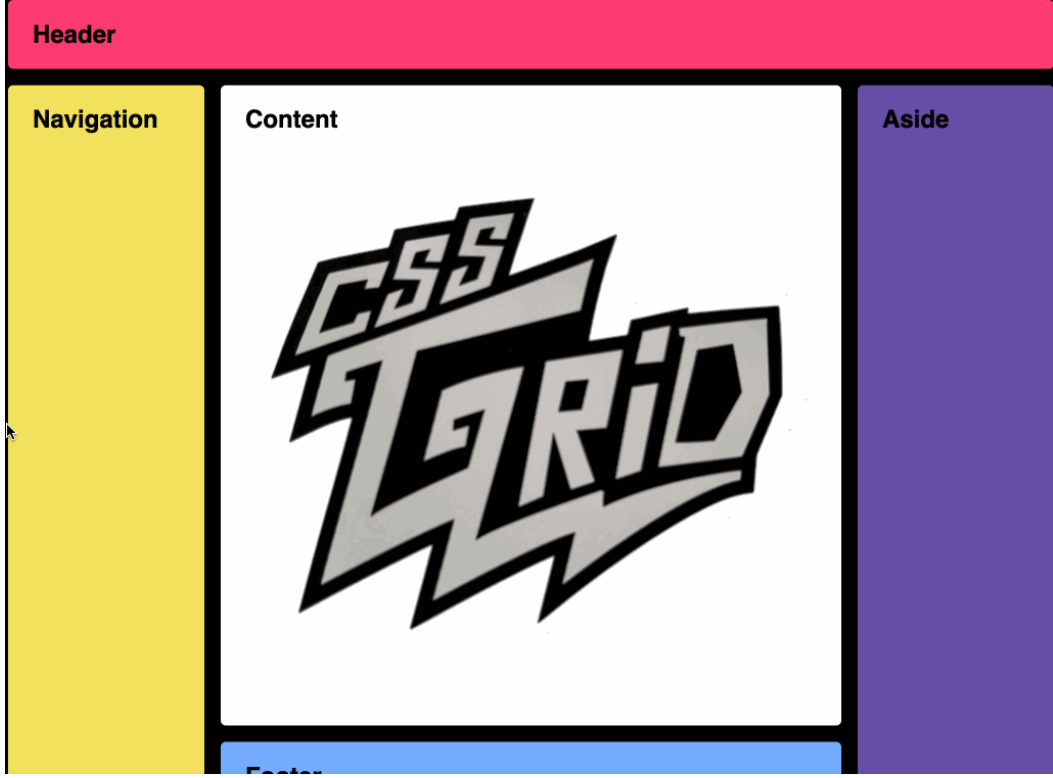
- Checks device width.




```
<img
  sizes="
    (min-width: 960px) 600px,
    (min-width: 640px) 300px,
    150px
  "
  srcset="
    css-grid-mobile.png 150w,
    css-grid-laptop.png 300w,
    css-grid.png 600w
  "
  src="css-grid.png"
/>
```


Let the browser choose the right sized image for the device.

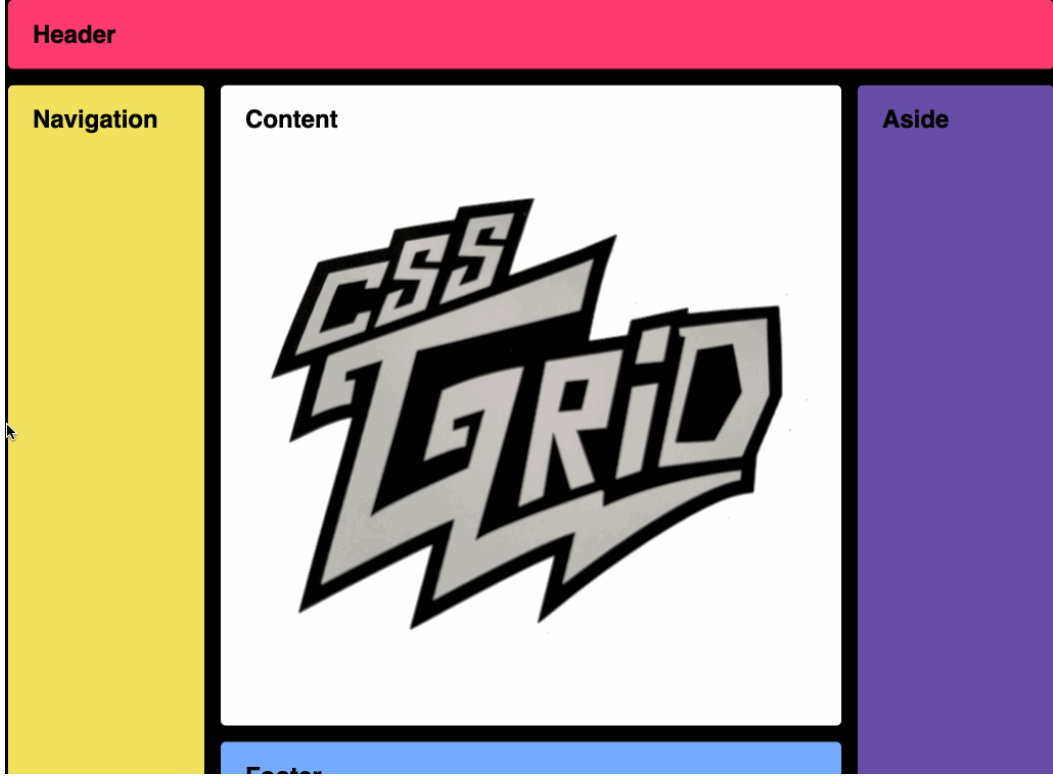
- Checks device width.
- Uses our old friend media query to determine the correct image to use.




```
<img
  sizes="
    (min-width: 960px) 600px,
    (min-width: 640px) 300px,
    150px
  "
  srcset="
    css-grid-mobile.png 150w,
    css-grid-laptop.png 300w,
    css-grid.png 600w
  "
  src="css-grid.png"
/>
```


Let the browser choose the right sized image for the device.

- Checks device width.
- Uses our old friend media query to determine the correct image to use.
- Requests the corresponding image from srcset.

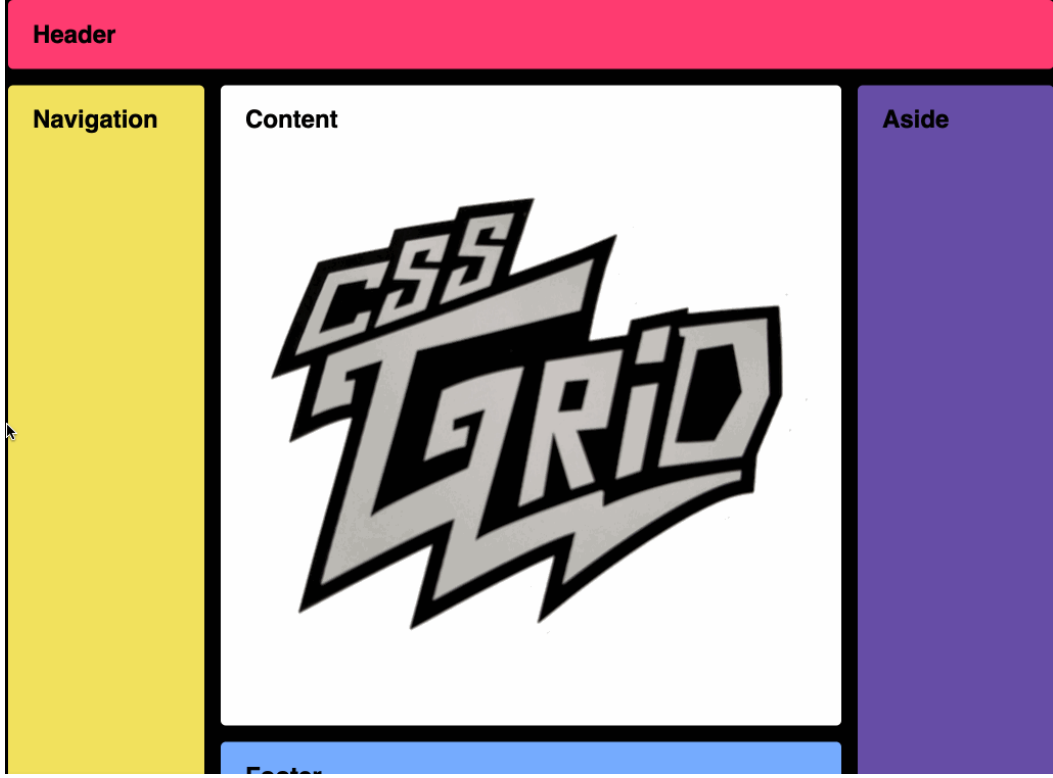



```
<img
  sizes="
    (min-width: 960px) 600px,
    (min-width: 640px) 300px,
    150px
  "
  srcset="
    css-grid-mobile.png 150w,
    css-grid-laptop.png 300w,
    css-grid.png 600w
  "
  src="css-grid.png"
/>
```

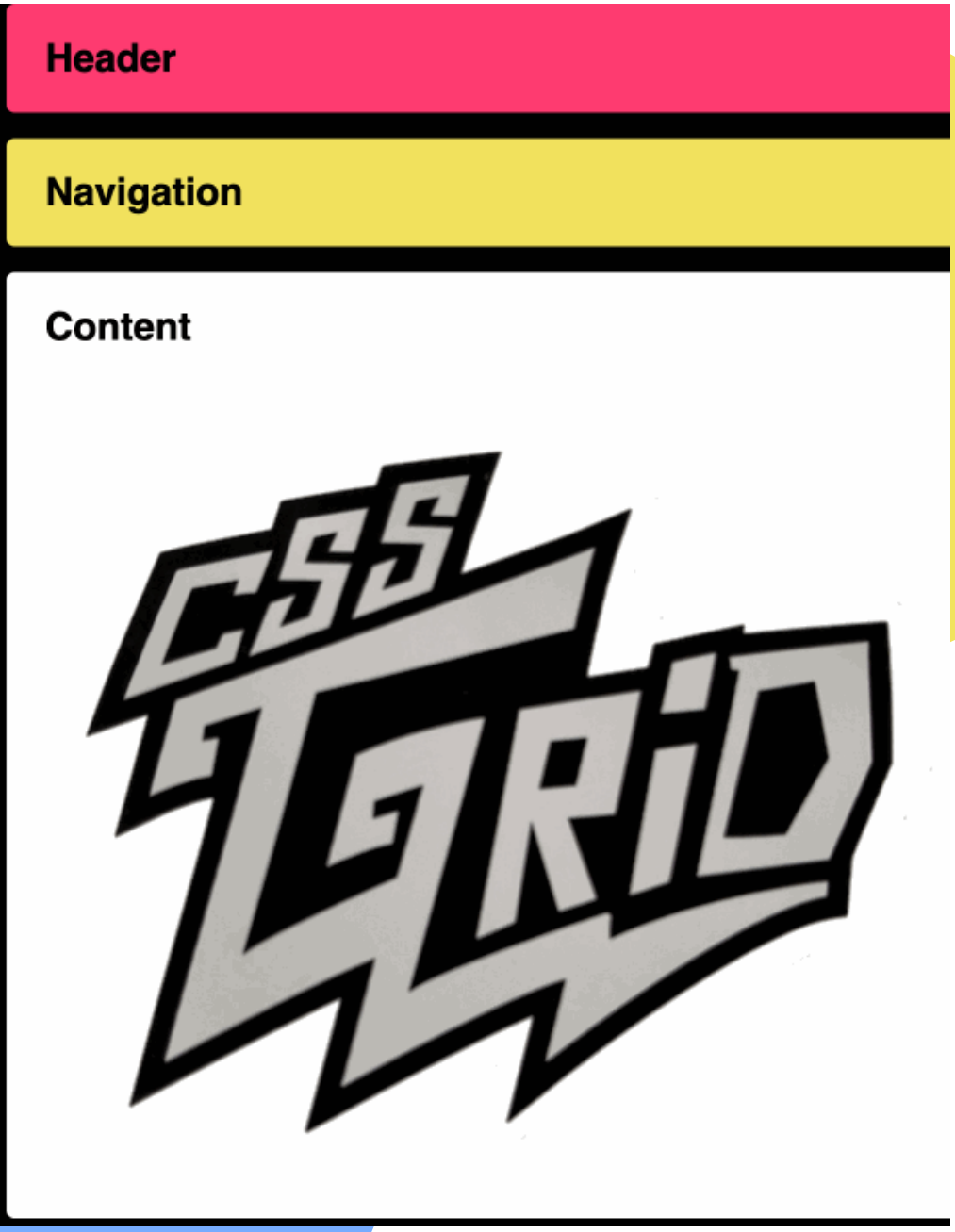

Let the browser choose the right sized image for the device.

- Checks device width.
- Uses our old friend media query to determine the correct image to use.
- Requests the corresponding image from srcset.

In our simple example downloading the image sized for mobile saves 230 kb.




```
<img
  sizes="
    (min-width: 960px) 600px,
    (min-width: 640px) 300px,
    150px
  "
  srcset="
    css-grid-mobile.png 150w,
    css-grid-laptop.png 300w,
    css-grid.png 600w
  "
  src="css-grid.png"
/>
```



<picture/>

Used when you want to change more than the resolution of the image you serve.

Header

Navigation

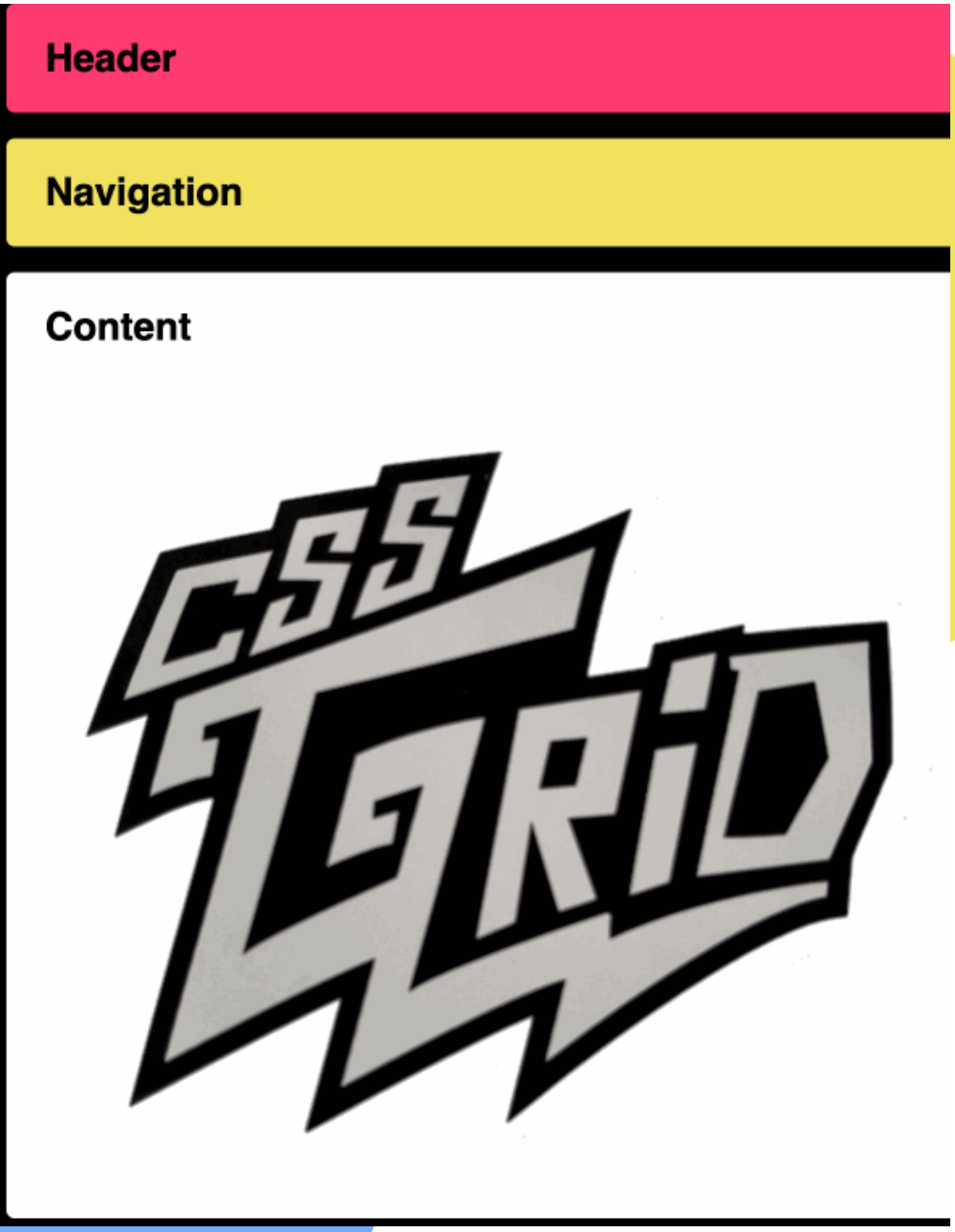
Content

<picture/>

Used when you want to change more than the resolution of the image you serve.

For example dark mode:

```
<picture>
  <source srcset="css-grid-night.png"
          media="(prefers-color-scheme: dark)">
  <img src="css-grid.png">
</picture>
```



<picture/>

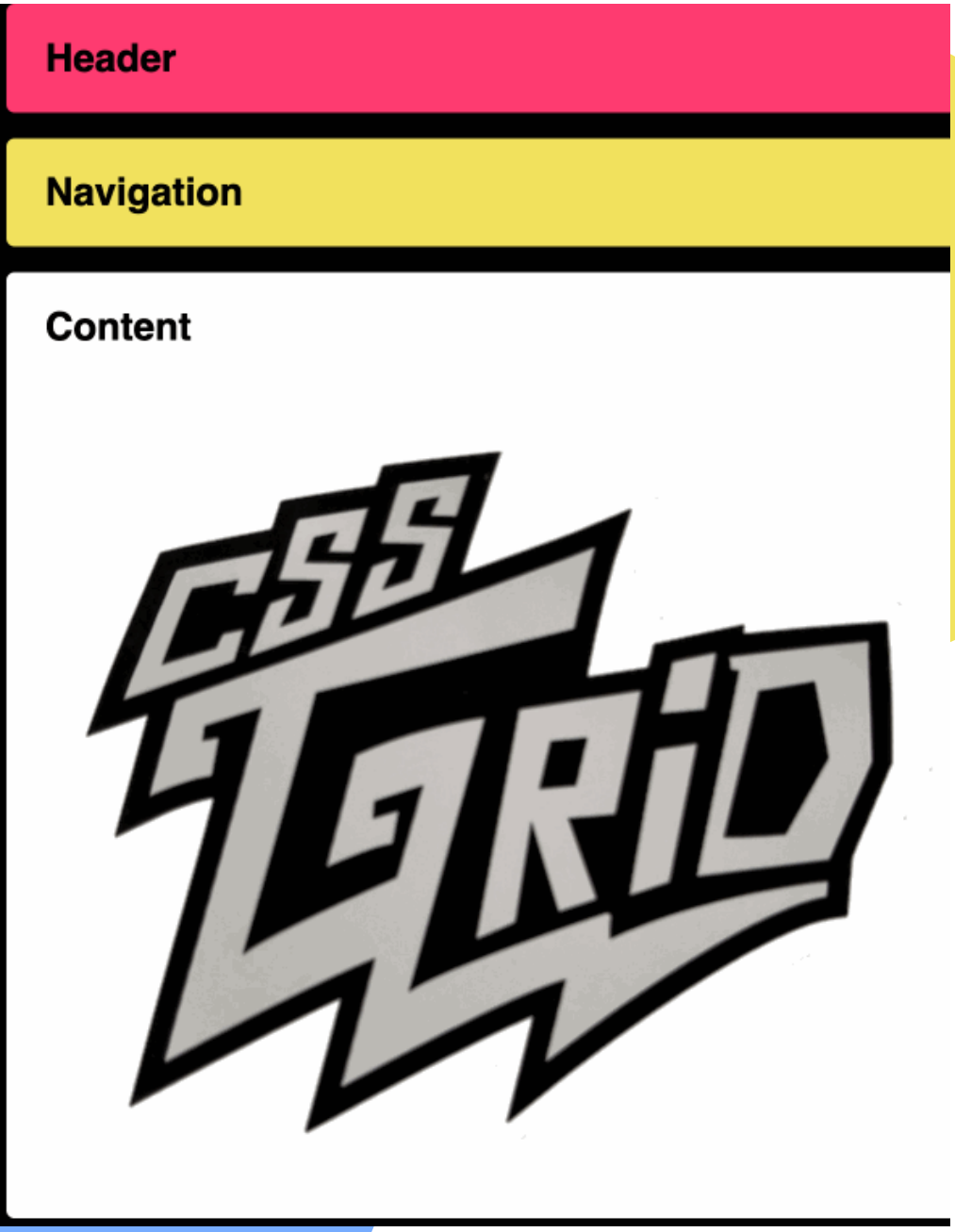
Used when you want to change more than the resolution of the image you serve.

For example dark mode:

```
<picture>
  <source srcset="css-grid-night.png"
          media="(prefers-color-scheme: dark)">
  <img src="css-grid.png">
</picture>
```

Other options:

- Serve zoomed-in images to smaller screens



<picture/>

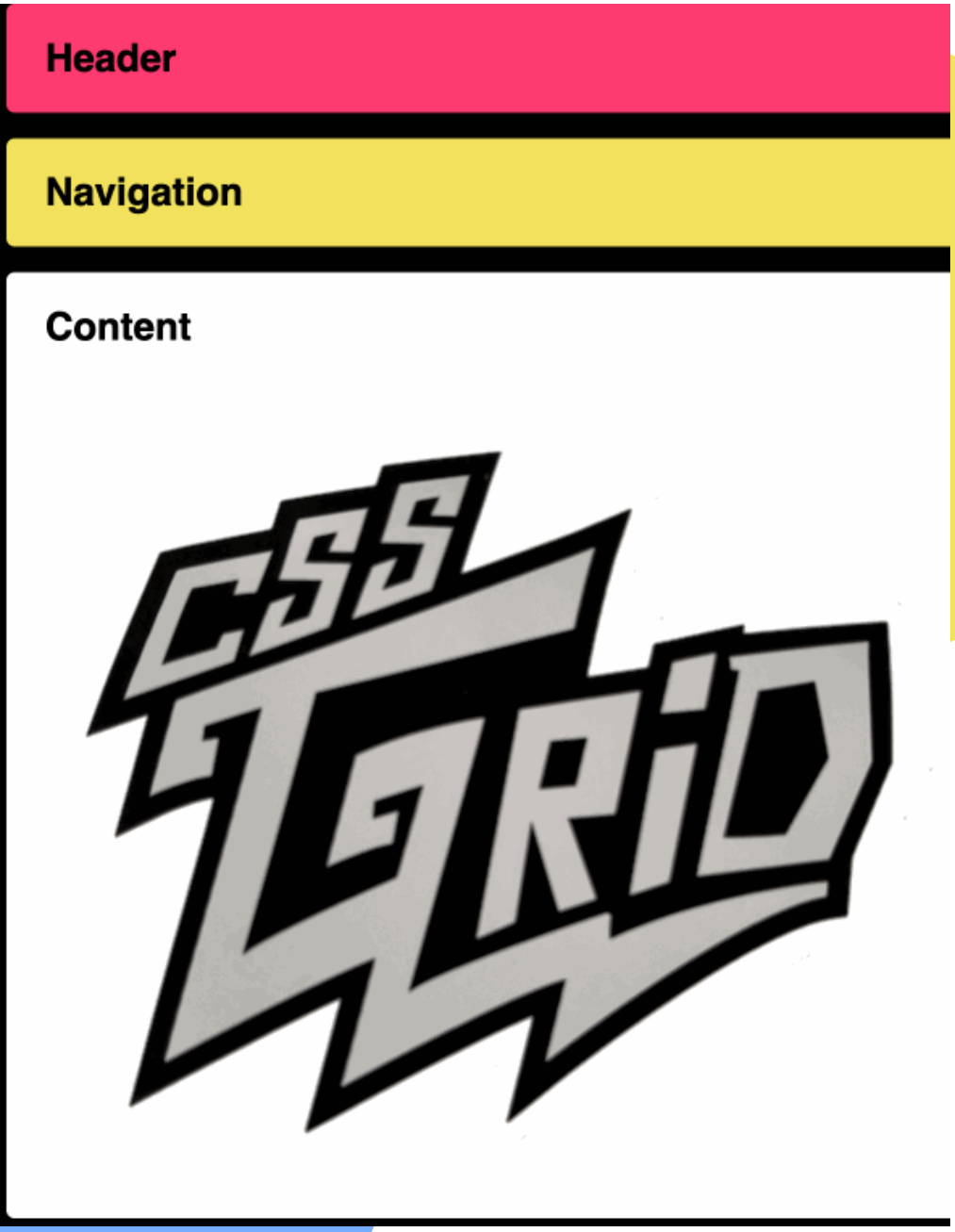
Used when you want to change more than the resolution of the image you serve.

For example dark mode:

```
<picture>
  <source srcset="css-grid-night.png"
          media="(prefers-color-scheme: dark)">
  <img src="css-grid.png">
</picture>
```

Other options:

- Serve zoomed-in images to smaller screens
- Don't send animated gifs if the user prefers-reduced-motion.



<picture/>

Used when you want to change more than the resolution of the image you serve.

For example dark mode:

```
<picture>
  <source srcset="css-grid-night.png"
          media="(prefers-color-scheme: dark)">
  <img src="css-grid.png">
</picture>
```

Other options:

- Serve zoomed-in images to smaller screens
- Don't send animated gifs if the user prefers-reduced-motion.
- Use the latest image formats while providing a fallback for legacy browsers.

Can I Use <picture/>?

Picture element - LS

A responsive images method to control which image resource a user agent presents to a user, based on resolution, media query and/or support for a particular image format

Usage % of all users ?

Global 96.72%

Current aligned | Usage relative | Date relative | Filtered | All

IE	Edge *	Firefox	Chrome	Safari	Opera	Safari on iOS *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baidu Browser	KaiOS Browser
		2-33	4-36		10-23											
	12	3 34-37	1 37	3.1-9	2 24	3.2-9.2										
6-10	13-97	38-96	38-97	9.1-15.1	25-82	9.3-15.1		2.1-4.4.4	12-12.1				4-15.0			
11	98	97	98	15.3	83	15.3	all	98	64	98	96	12.12	16.0	10.4	7.12	2.5
		98-99	99-101	15.4-TP		15.4										

In Conclusion...

In Conclusion...

- Use CSS Grid to layout your app or site

In Conclusion...

- Use CSS Grid to layout your app or site
- Use grid areas and skip the complicated line-based syntax.

In Conclusion...

- Use CSS Grid to layout your app or site
- Use grid areas and skip the complicated line-based syntax.
- Lean towards using srcset and sizes for images

In Conclusion...

- Use CSS Grid to layout your app or site
- Use grid areas and skip the complicated line-based syntax.
- Lean towards using srcset and sizes for images
- Until you need to use non-size media-queries or nonstandard image formats

Thank You!

Questions?

Why didn't you talk about my favourite framework?


#Shahs
bravo
I HATE IT

Okay, I don't hate JS frameworks but...

Okay, I don't hate JS frameworks but...

In recent years we've pivoted towards better developer experience at the expense of the user experience. None of the things we've talked about today require a byte of JavaScript in order to work.

Okay, I don't hate JS frameworks but...

In recent years we've pivoted towards better developer experience at the expense of the user experience. None of the things we've talked about today require a byte of JavaScript in order to work. Reducing the amount of JS downloaded/parsed/run will increase the performance of your site/app.

Okay, I don't hate JS frameworks but...

In recent years we've pivoted towards better developer experience at the expense of the user experience. None of the things we've talked about today require a byte of JavaScript in order to work.

Reducing the amount of JS downloaded/parsed/run will increase the performance of your site/app.

Bias towards web fundamentals before reaching for that JS framework.

What about laying out components?


THIS IS A GREAT QUESTION


FLEXBOX

Can I Use Flex Box?

CSS Flexible Box Layout Module - CR

Method of positioning elements in horizontal or vertical stacks. Support includes all properties prefixed with `flex`, as well as `display: flex`, `display: inline-flex`, `align-content`, `align-items`, `align-self`, `justify-content` and `order`.

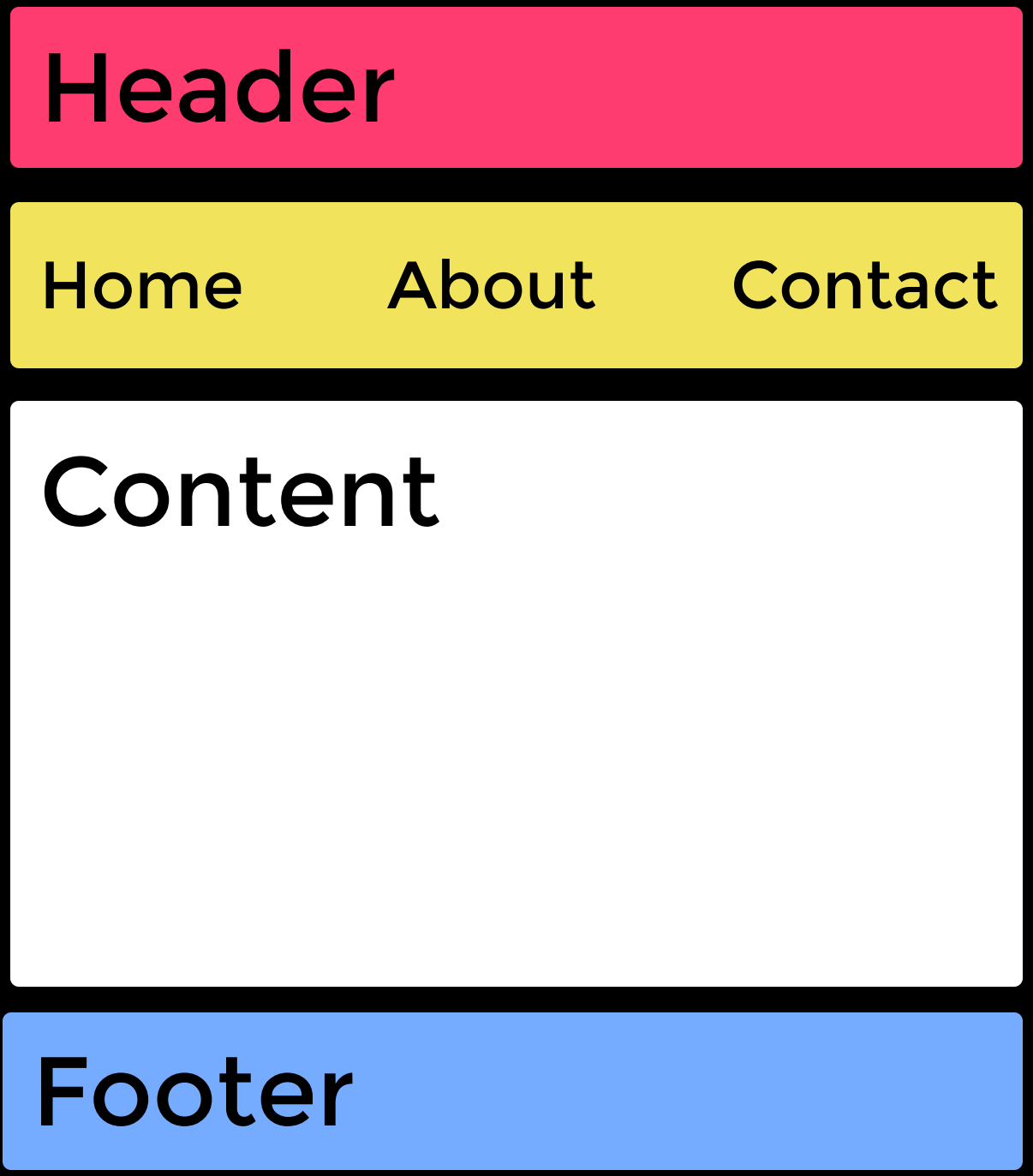
Usage % of all users ?

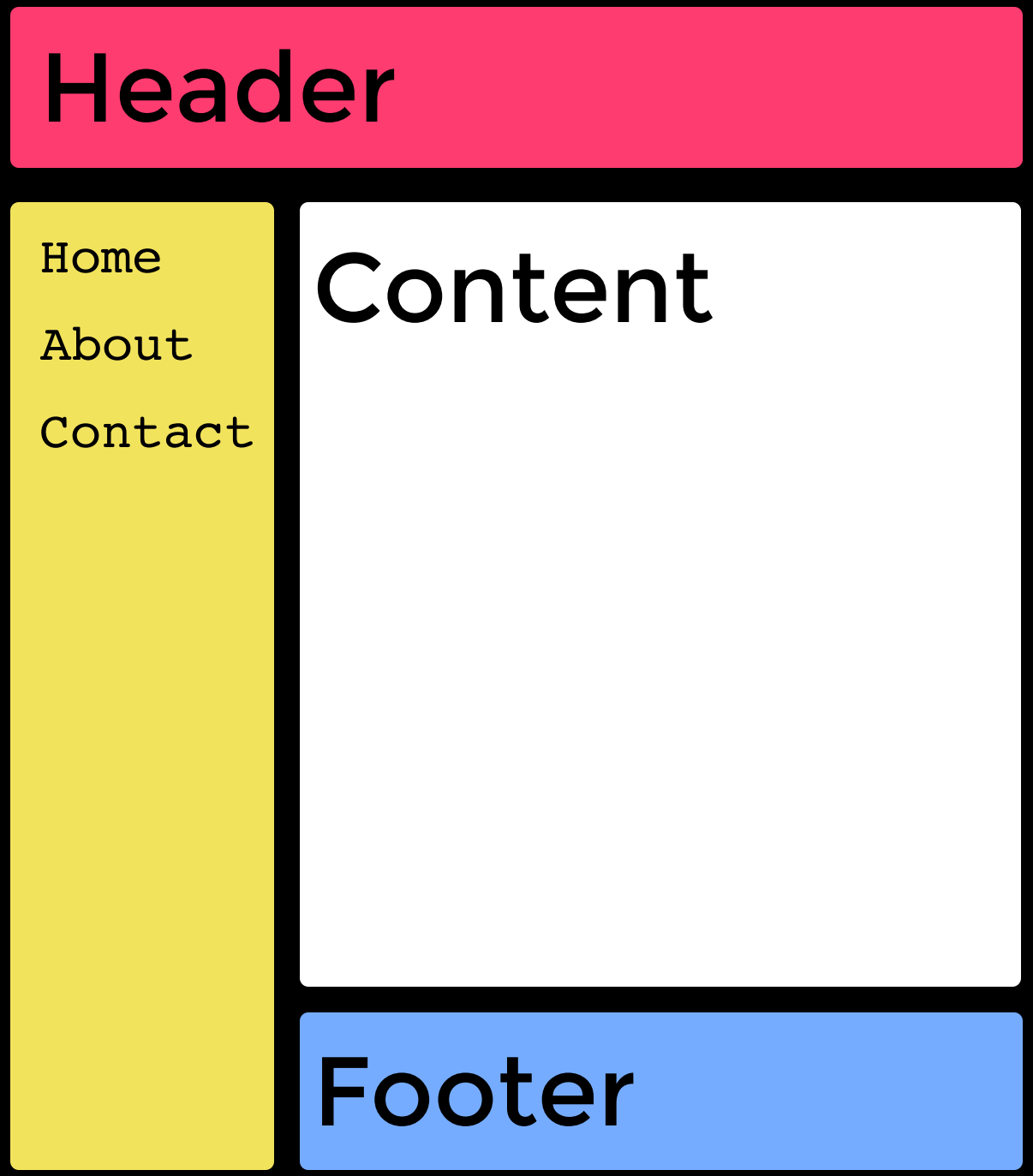
Global 98.32% + 1.14% = 99.46%

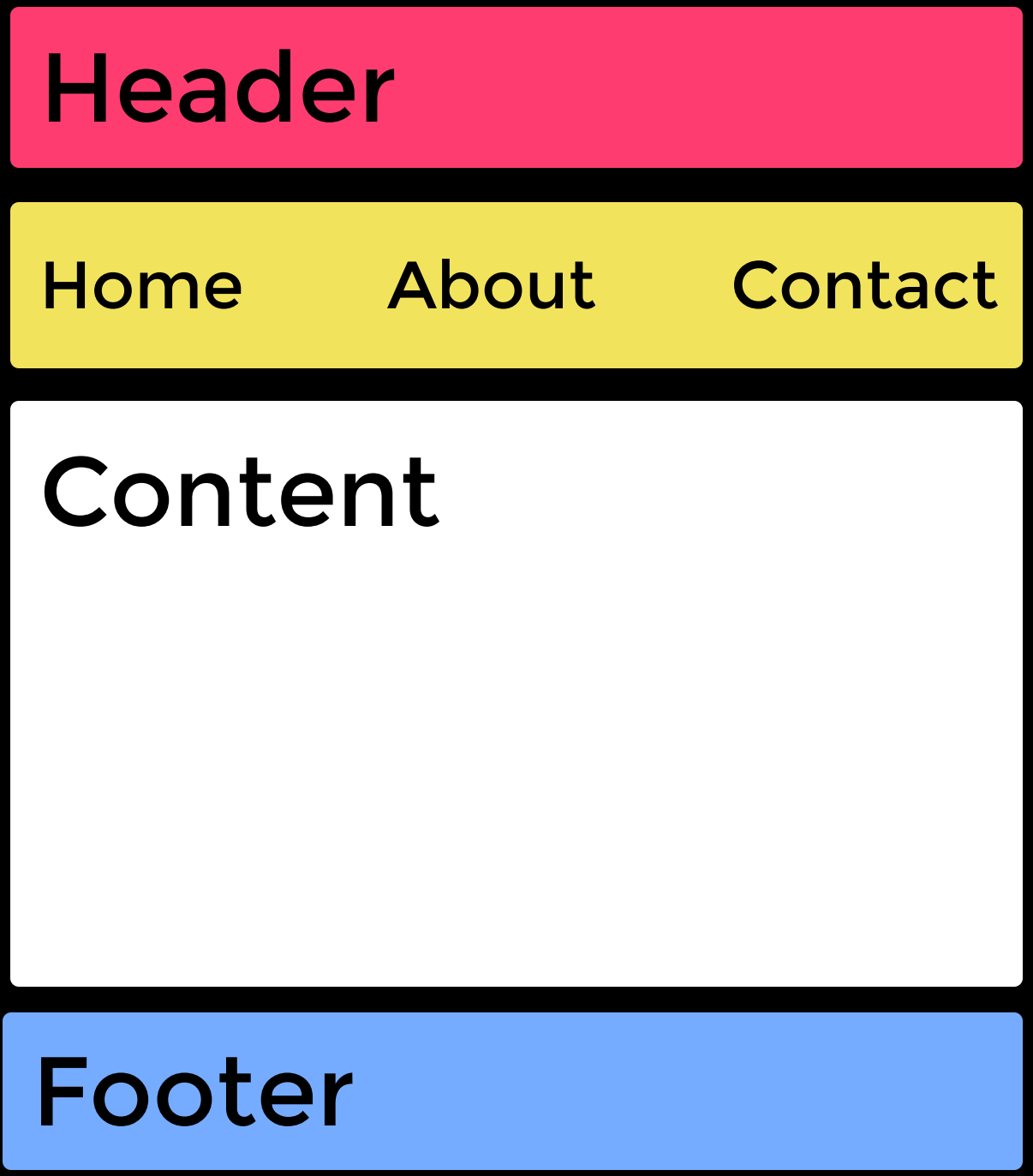
unprefixed: 98.21% + 0.65% = 98.85%

Current aligned | Usage relative | Date relative | Filtered | All

IE	Edge	Firefox	Chrome	Safari	Opera	Safari on iOS	Opera Mini	Android Browser	Opera Mobile	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baidu Browser	KaiOS Browser
					10-11.5											
		1 2-21	1 4-20	1 3.1-6	12.1	1 3.2-6.1										
6-9		3 22-27	21-28	6.1-8	15-16	7-8.4		1 2.1-4.3	12							
2 4 10	12-97	28-96	29-97	9-15.1	17-82	9-15.1		4.4-4.4.4	12.1				4-15.0			
4 11	98	97	98	15.3	83	15.3	all	98	64	98	96	12.12	16.0	10.4	7.12	2.5
		98-99	99-101	15.4-TP		15.4										


Header
Home About Contact
Content
Footer

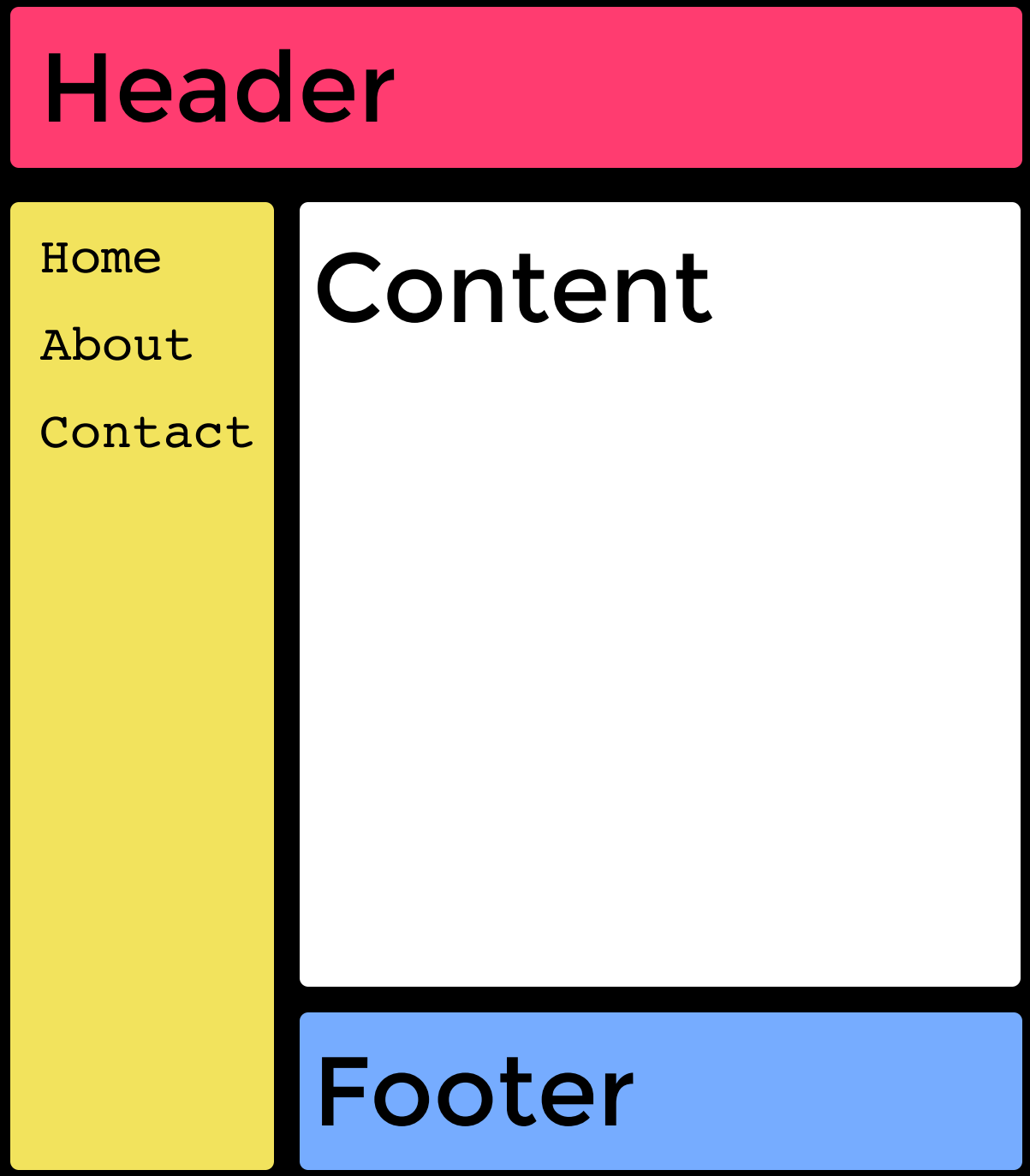

Header
Home
About
Contact
Content
Footer



Mobile Nav

```css
nav {
  grid-area: nav;
  display: flex;
  justify-content: space-between;
}
```

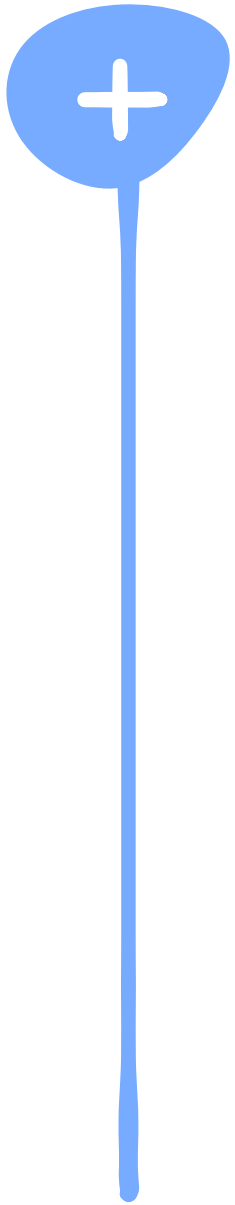
```html
<html>
  <body>
    <header>Header</header>
    <nav>
      <p>Home</p>
      <p>About</p>
      <p>Contact</p>
    </nav>
    <main>Content</main>
    <footer>Footer</footer>
  </body>
</html>
```



Laptop Nav

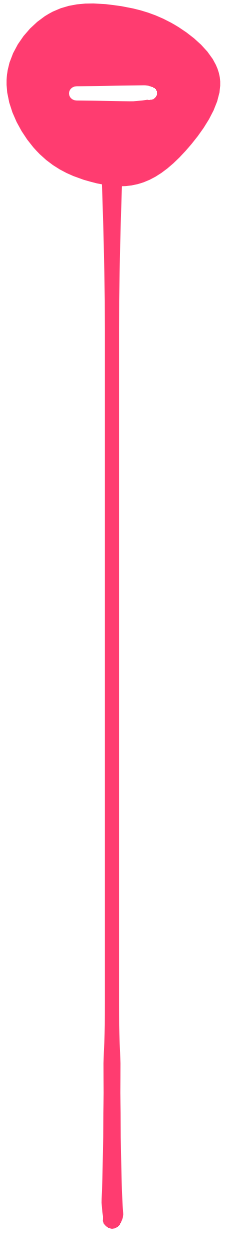
```css
@media (min-width: 40em) {
  nav {
    flex-direction: column;
    justify-content: flex-start;
  }
}
```

```html
<html>
  <body>
    <header>Header</header>
    <nav>
      <p>Home</p>
      <p>About</p>
      <p>Contact</p>
    </nav>
    <main>Content</main>
    <footer>Footer</footer>
  </body>
</html>
```



Flex Box

- Layout in one dimension
- Content out



CSS Grid

- Two-dimensional layout
- Layout in

Thank You!
For real this time

Questions?