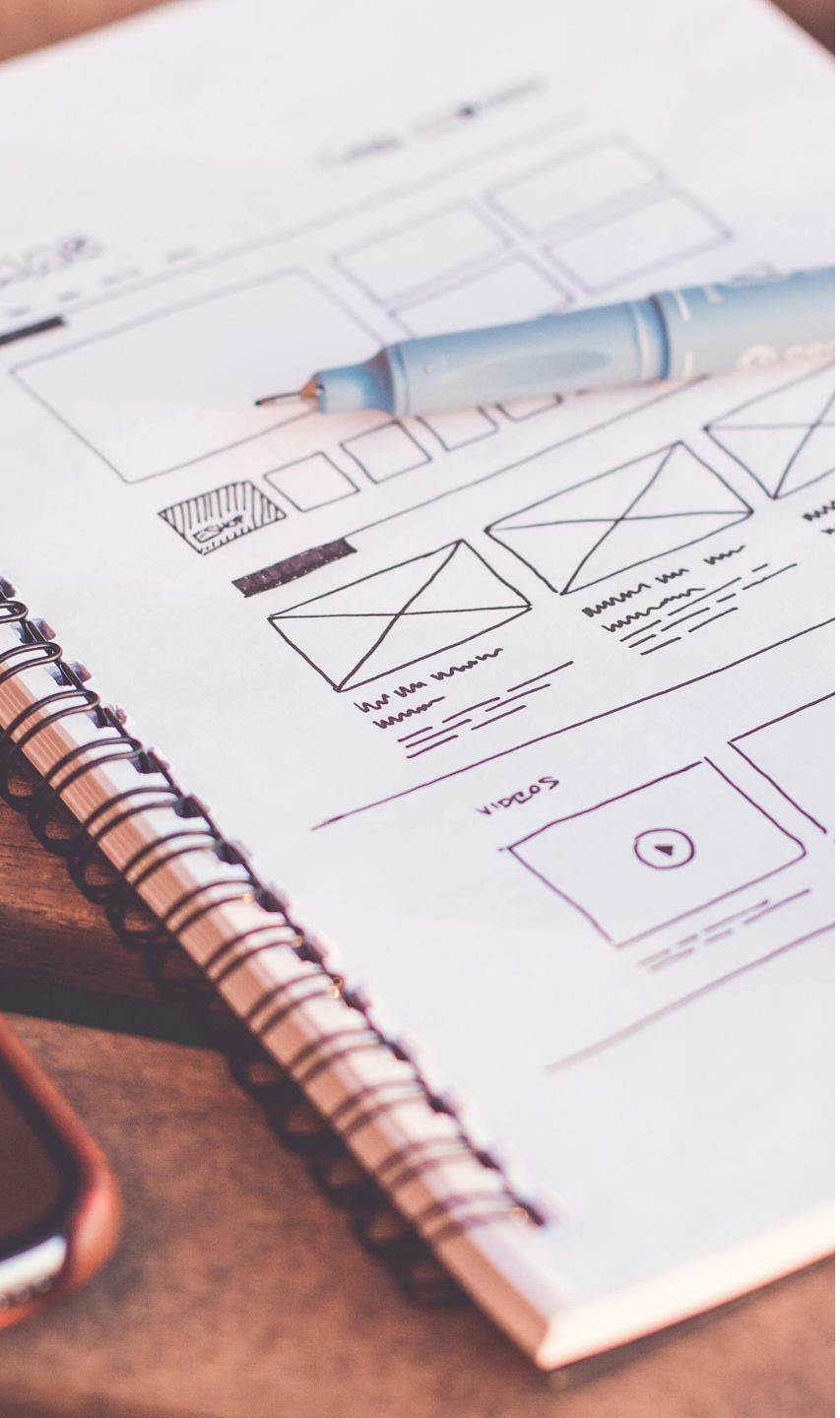


INTRO APP WEB

Politecnico di Torino

Introduzione alle Applicazioni Web

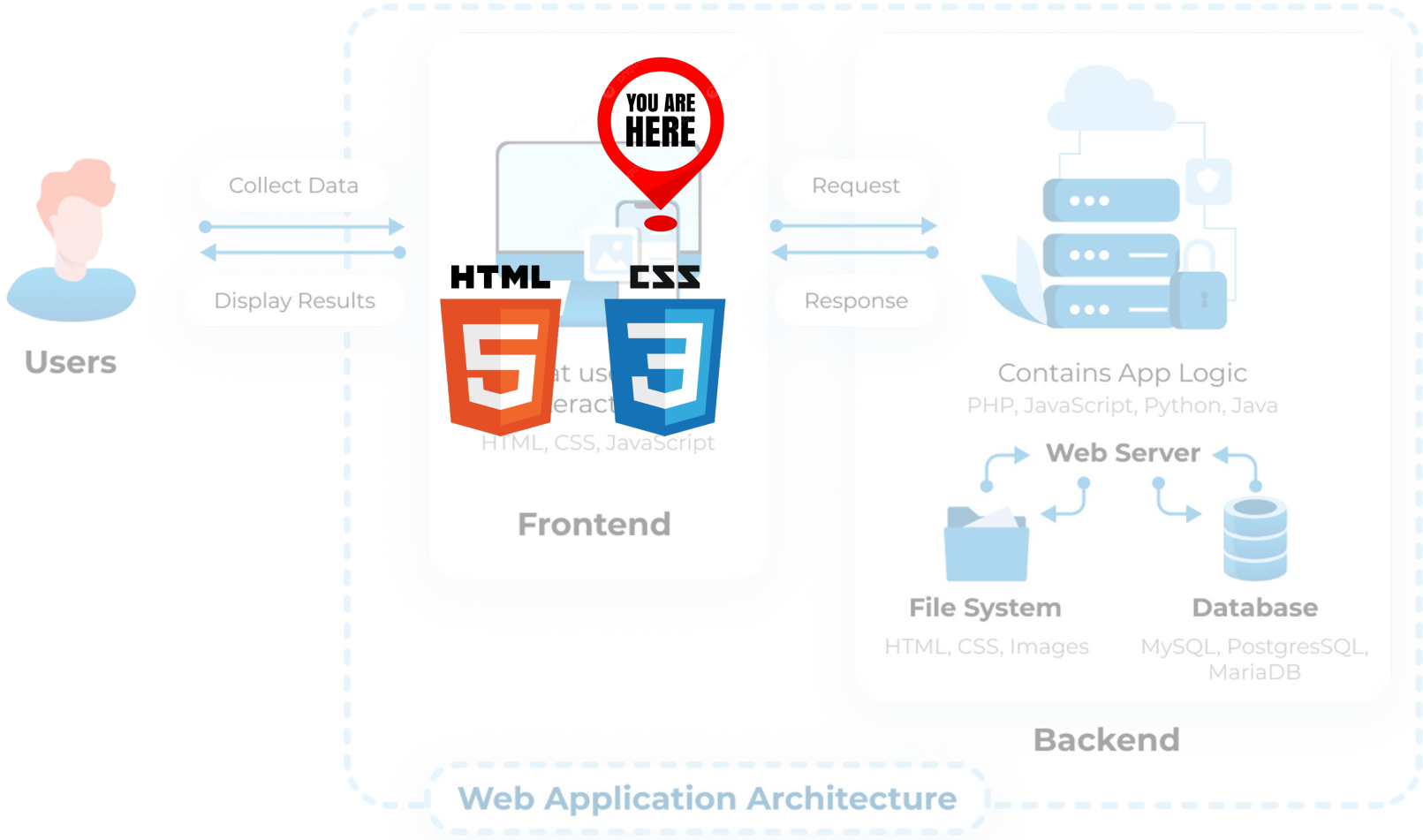
More CSS

Juan Pablo Sáenz

Goals

- Understand how to create **advanced layouts**
- Add **responsiveness** to a web page
- Utilize **libraries**

CSS: where are we?

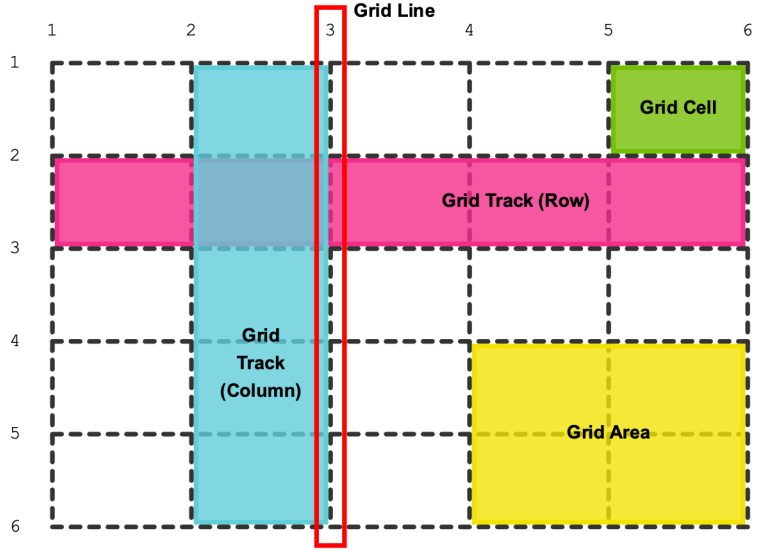


Page Layouts

- **CSS Grid layout**
- **CSS Flexbox**

CSS Grid Layout

A layout system in CSS that enables designing web pages using a **two-dimensional grid structure**, consisting of both **rows** and **columns**.



https://webkit.org/blog/7434/css-grid-layout-a-new-layout-module-for-the-web/

https://medium.muz.li/understanding-css-grid-ce92b7aa67cb

CSS Grid Layout

Create a Grid

Declaring **`display: grid`** on the container element

Define rows and columns

Properties **`grid-template-columns`** and **`grid-template-rows`**

Add a gutter

Use the **`grid-gap`** property

```
<div class="mygridcontainer">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
</div>
```

```
.mygridcontainer {
  display: grid;
  grid-template-columns: 100px 100px 100px;
  grid-template-rows: 150px 150px;
  grid-gap: 1rem;
}
```

CSS Grid Layout

The `fr` unit

A unit which represents a **fraction** of the available space in the grid container

```html
<div class="mygridcontainer">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
</div>
```

```css
.mygridcontainer {
  display: grid;
  grid-template-columns: 1fr 1fr 1fr;
  grid-template-rows: 150px 150px;
  grid-gap: 1rem;
}
```

CSS Grid Layout

The `fr` unit

A unit which represents a **fraction** of the available space in the grid container

The CSS `repeat` function

```
<div class="mygridcontainer">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
</div>
```

```
.mygridcontainer {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  grid-template-rows: repeat(2, 150px);
  grid-gap: 1rem;
}
```

CSS Grid Layout

The `fr` unit

A unit which represents a **fraction** of the available space in the grid container

The CSS `repeat` function

Mixing units

```
<div class="mygridcontainer">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
</div>
```

```
.mygridcontainer {
  width: 100%;
  display: grid;
  grid-template-columns: 100px 30% 1fr;
  grid-template-rows: 200px 100px;
  grid-gap: 1rem;
}
```

CSS Grid Layout

The `fr` unit

A unit which represents a **fraction** of the available space in the grid container

The CSS `repeat` function

Mixing units

Positioning items

```html
<div class="mygridcontainer">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
  <div class="item4">4</div>
  <div class="item5">5</div>
  <div class="item6">6</div>
  <div class="item7">7</div>
  <div class="item8">8</div>
</div>
```

```css
.item1 {
  grid-row-start: 2;
  grid-row-end: 3;
  grid-column-start: 2;
  grid-column-end: 3;
}
```

CSS Grid Layout

The `fr` unit

A unit which represents a **fraction** of the available space in the grid container

The CSS `repeat` function

Mixing units

Positioning items

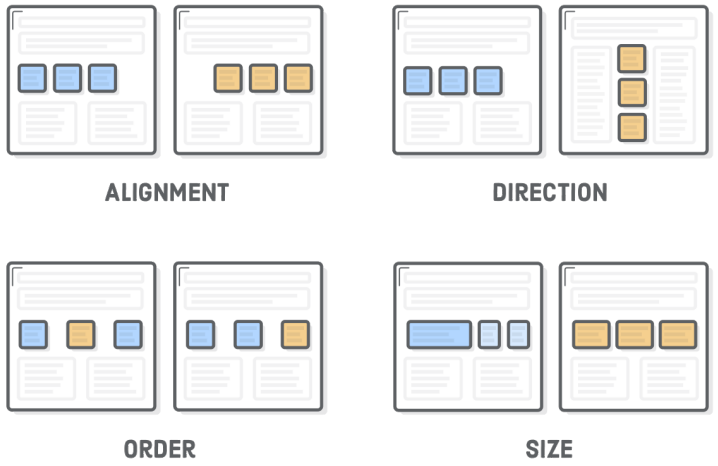
```
<div class="mygridcontainer">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
  <div class="item4">4</div>
  <div class="item5">5</div>
  <div class="item6">6</div>
  <div class="item7">7</div>
  <div class="item8">8</div>
</div>
```

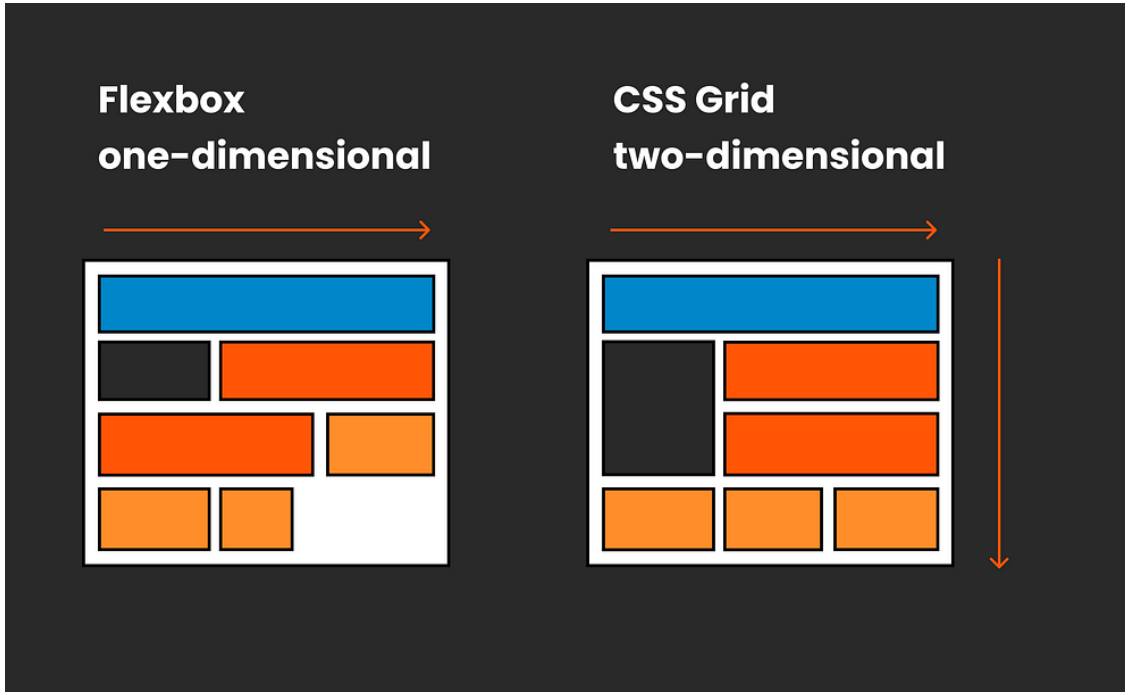
```
.item1 {
  grid-row: 2 / 3;
  grid-column: 2 / 3;
}
```

CSS Flexbox

A **one-dimensional** layout method for arranging items in rows or columns.

Items **flex** (expand) to fill additional space or shrink to fit into smaller spaces.





https://uxdesign.cc/why-ui-designers-should-understand-flexbox-and-css-grid-e236a9dec37a

https://internetingishard.netlify.app/html-and-css/flexbox/

CSS Flexbox

Create a Flexbox

Declaring **`display: flex`** on the container element.

The direct children of that container become **flex items**.


```html
<div class="myfbcontainer">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
</div>
```

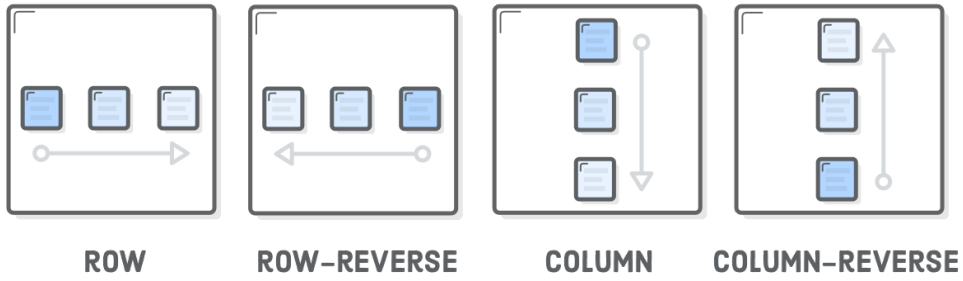
```css
.myfbcontainer {
  display: flex;
}
```

https://internetingishard.netlify.app/html-and-css/flexbox/

CSS Flexbox

The two axes (`flex-direction`)

The **main axis** is defined by the **`flex-direction`** property (default value is **row**).



The **cross axis** runs perpendicular to the main axis.

```
<div class="myfbcontainer">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>
```

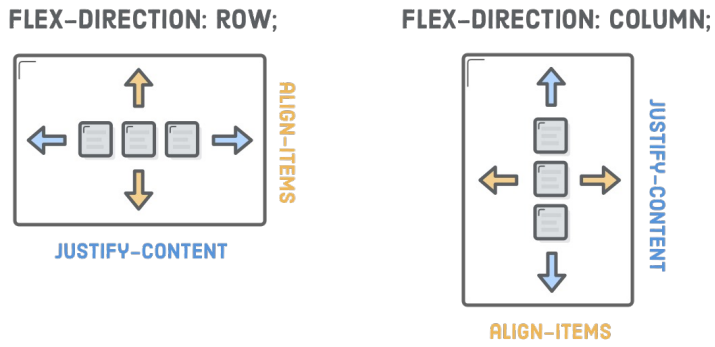
```
.myfbcontainer {
  display: flex;
  flex-direction: row;
}
```

https://internetingishard.netlify.app/html-and-css/flexbox/

CSS Flexbox

Alignment

- **`justify-content`** to align the item on the main axis.
- **`align-items`** property to align the item on the cross axis.



https://internetingishard.netlify.app/html-and-css/flexbox/

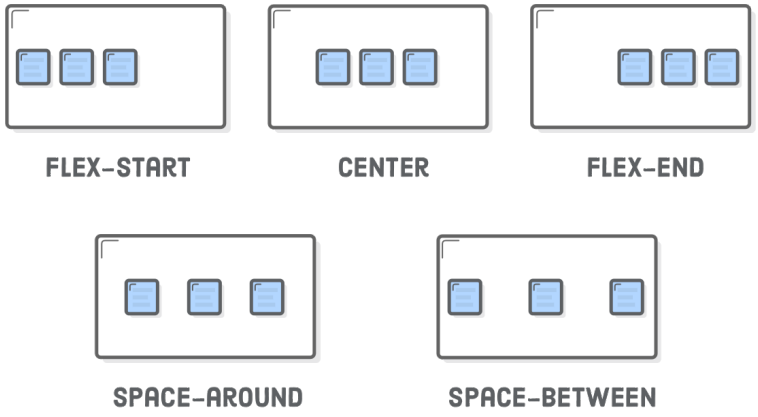
```
<div class="myfbcontainer">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>
```

```
.myfbcontainer {
  display: flex;
  flex-direction: row;
  justify-content: center;
  align-items: center;
}
```

CSS Flexbox

Alignment

- **justify-content**




```html
<div class="myfbcontainer">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>
```

```css
.myfbcontainer {
  display: flex;
  flex-direction: row;
  justify-content: center;
  align-items: center;
}
```

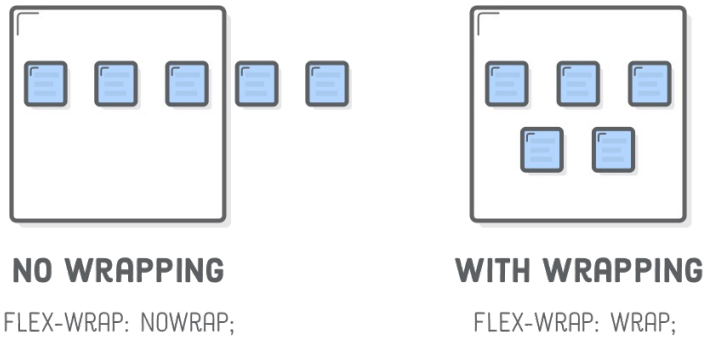
https://internetingishard.netlify.app/html-and-css/flexbox/

CSS Flexbox

Wrapping

It is possible to make flex items wrap across multiple lines.

Defined by the property **`flex-wrap`** with a value of **`wrap`**.




```
https://internetingishard.netlify.app/html-and-css/flexbox/
```

```html
<div class="myfbcontainer">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
</div>
```

```css
.myfbcontainer {
  display: flex;
  flex-direction: row;
  justify-content: center;
  align-items: center;
  flex-wrap: wrap;
}
```

CSS Flexbox

Properties applied to flex items

To control the inline-size of each flex item:

- **`flex-grow`**: Defines how much an item **expands** to fill available space.
- **`flex-shrink`**: Determines how much an item **shrinks** when space is limited.
- **`flex-basis`**: Sets the **initial size** of an item before applying grow or shrink.

Typically combined into the **`flex`** shorthand:
`flex-grow, flex-shrink, flex-basis`

https://internetingishard.netlify.app/html-and-css/flexbox/

```html
<div class="myfbcontainer">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
</div>
```

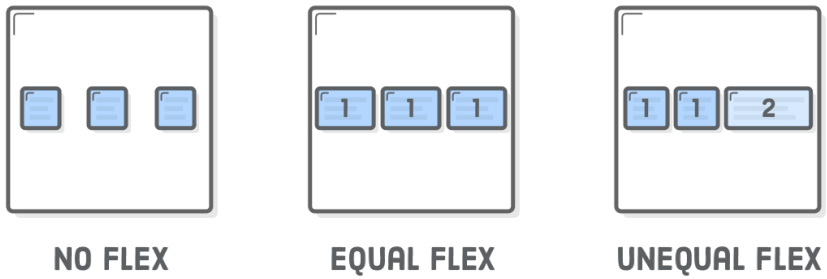
```css
.item1 {
  flex: 1 1 auto;
  /* flex: 1 is the same as writing
flex: 1 1 0 */
}
```

CSS Flexbox

Properties applied to flex items

The **`flex`** property defines the width of individual items in a flex container.

An item with a **`flex`** value of 2 will grow twice as fast as items with the default value of 1.



https://internetingishard.netlify.app/html-and-css/flexbox/

```
<div class="myfbcontainer">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
</div>
```

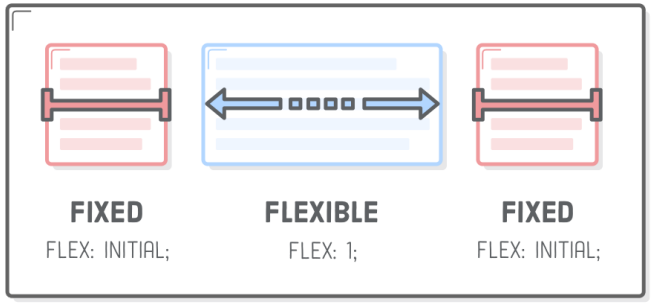
```
.item1 { flex: 1; }
.item2 { flex: 1; }
.item3 { flex: 2; }
```

CSS Flexbox

Properties applied to flex items

The **`flex`** property defines the width of individual items in a flex container.

An item with a **`flex`** value of 2 will grow twice as fast as items with the default value of 1.




```
<div class="myfbcontainer">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
</div>
```

```
.item1 { flex: initial; }
.item2 { flex: 1; }
.item3 { flex: initial; }
```

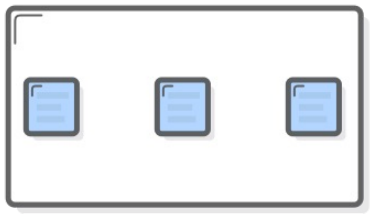
https://internetingishard.netlify.app/html-and-css/flexbox/

CSS Flexbox

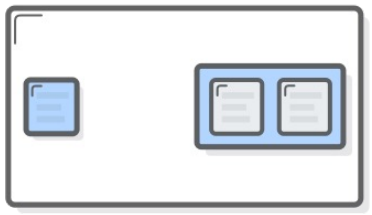
Grouping

Flex containers only know how to position elements that are one level deep

Group flex items using ** ** elements.



NO GROUPING
(3 FLEX ITEMS)



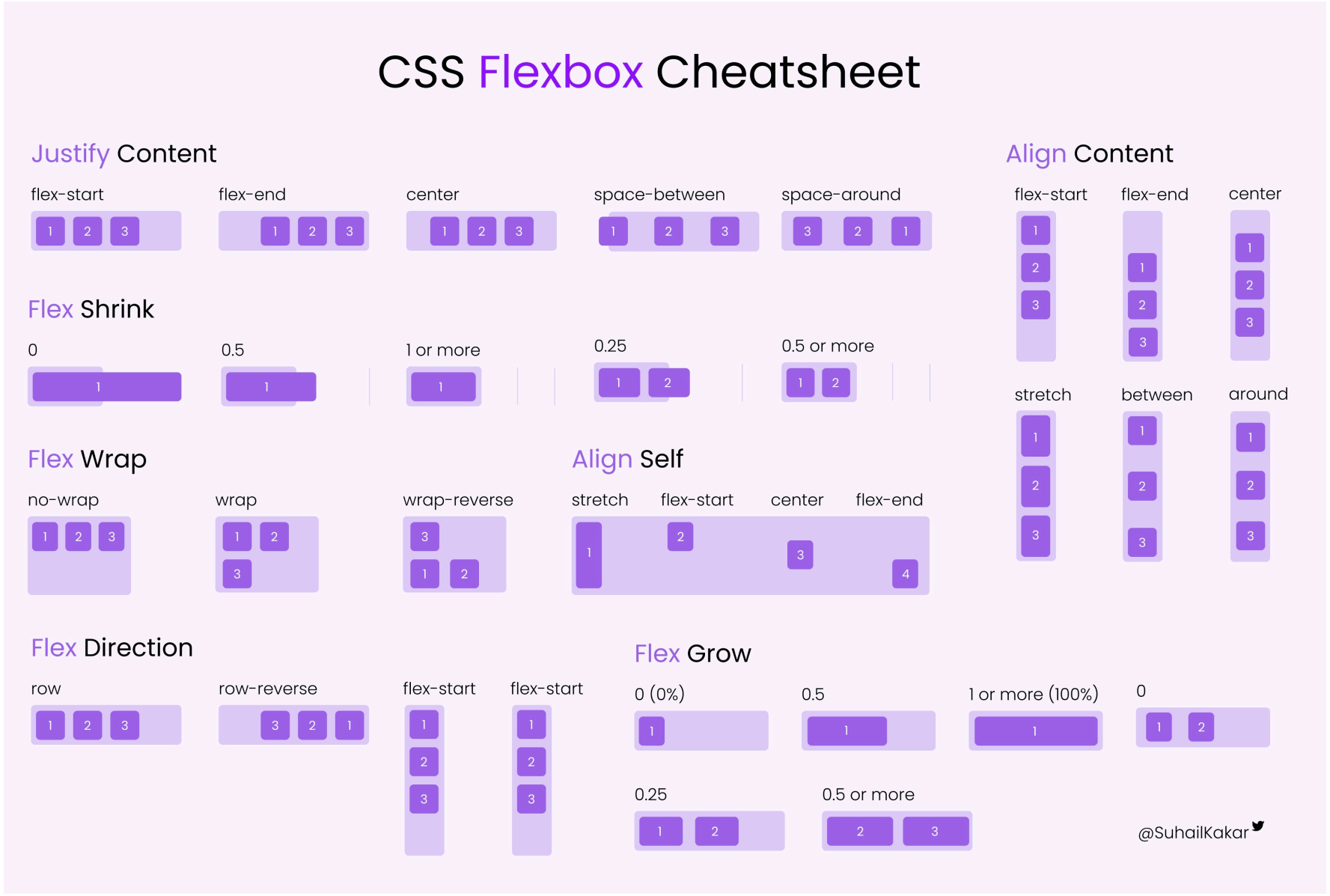
GROUPED ITEMS
(2 FLEX ITEMS)

```html
<div class="myfbcontainer">
  <div class="item1">1</div>
  <div>
    <div class="item2">2</div>
    <div class="item3">3</div>
  </div>
</div>
```

```css
.myfbcontainer {
  display: flex;
  flex-direction: row;
  justify-content: center;
  align-items: center;
  flex-wrap: wrap;
}
```

https://internetingishard.netlify.app/html-and-css/flexbox/

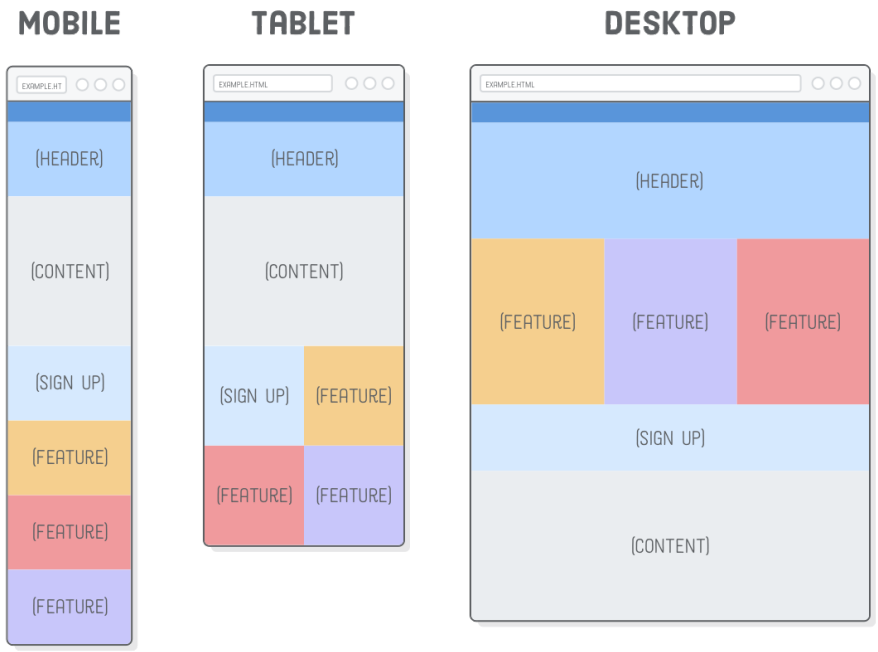
CSS Flexbox Cheatsheet

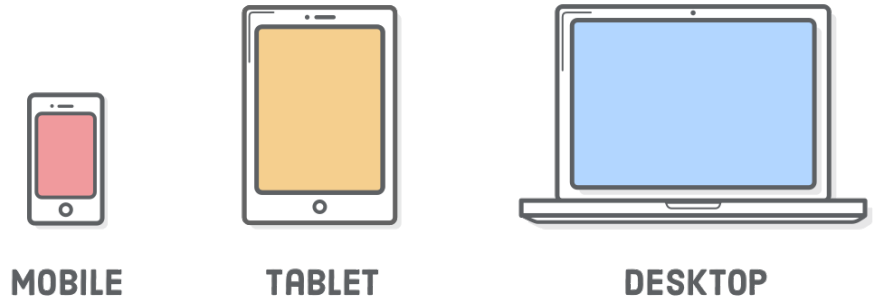


Let's see it in practice

Responsive Design

- Ensures websites adapt to different **screen sizes** and **devices**.
- Improves **usability** and **accessibility** across all devices.



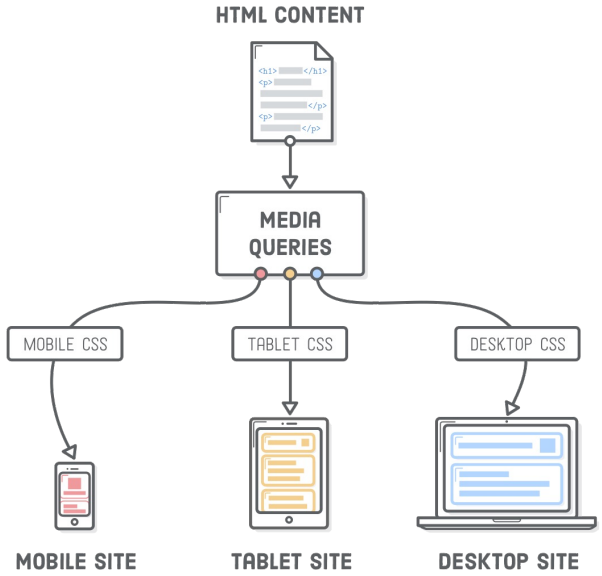


https://internetingishard.netlify.app/html-and-css/responsive-design/

Responsive Design

Accomplished through **CSS media queries**

- A way to **conditionally** apply CSS rules



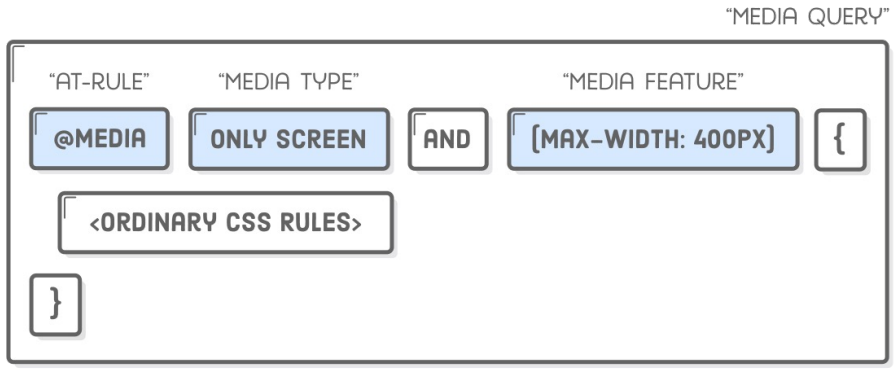

```css
/* Mobile */
@media(max-width:400px) {
  body {
    background-color: #F09A9D;
    /* Red */
  }
}
```

https://internetingishard.netlify.app/html-and-css/responsive-design/

Responsive Design

Accomplished through **CSS media queries**

- A way to **conditionally** apply CSS rules

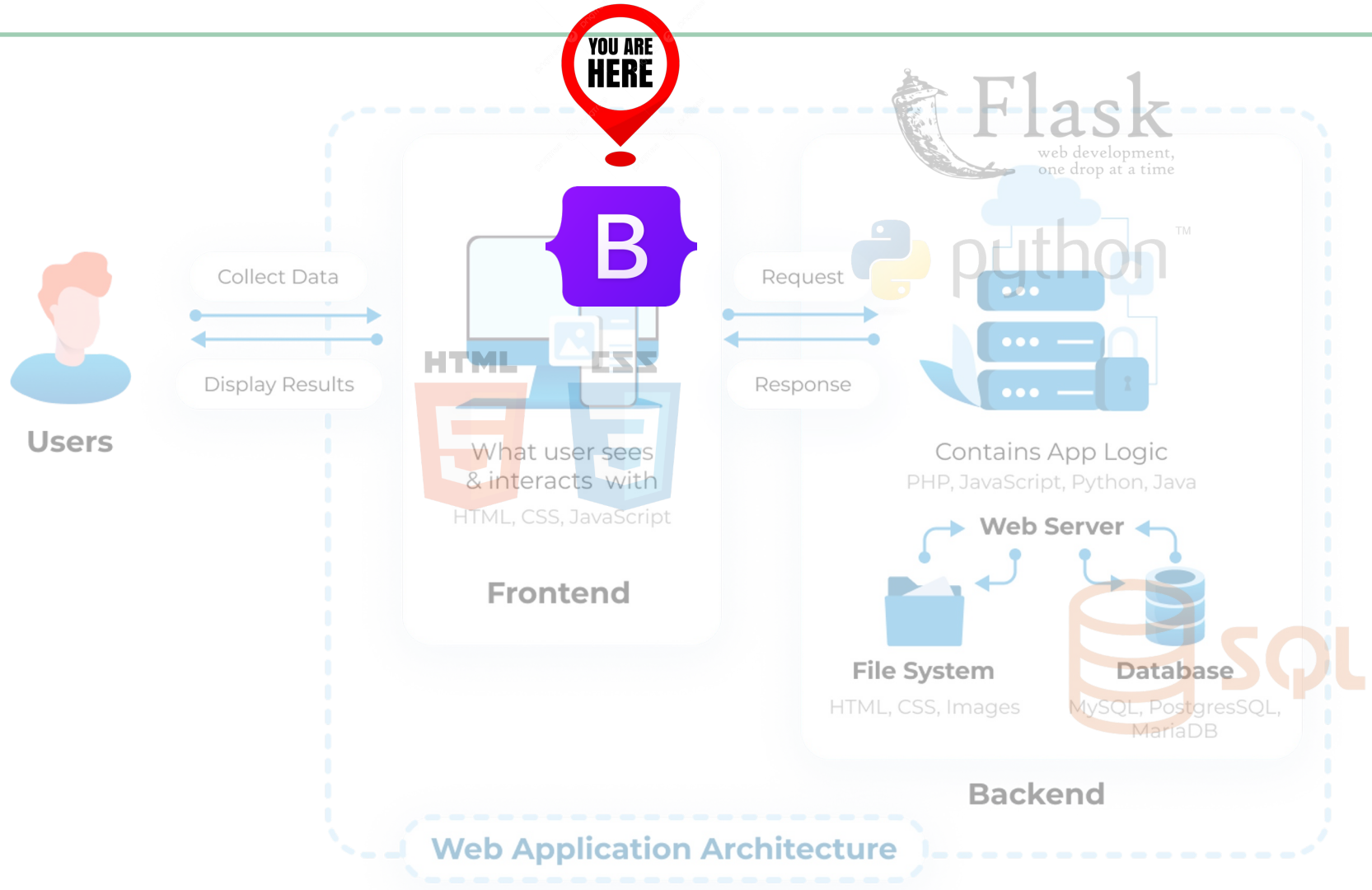



```
@media (min-width: 30em) and (orientation:
landscape) {
/* Restrict styles to landscape-oriented devices
with a width of at least 30 ems */
}

@media (min-height: 680px), screen and
(orientation: portrait) {
/* Apply the style if the user's device has either
a minimum height of 680px or is a screen device in
portrait mode */
}

@media not screen and (color), print and (color) {
/* Apply the style if it's not a colored screen
nor a colored printer */
}
```

https://internetingishard.netlify.app/html-and-css/responsive-design/

Let's see it in practice

📍 CSS Bootstrap: where are we?



Licenza