Australian National University

Full Stack Web Dev Overview

Comp1710/6780 Week 9

Dr Sabrina Caldwell
with thanks to Dr Xuanying Zhu
(xuanying.zhu@anu.edu.au)
School of Cybernetics
The Australian National University

Some initial details regarding final exam

- The final exam is scheduled for Wednesday 14 June 2023 at 9:50am.
- The exam will be undertaken within Wattle, using Proctorio
- 15 minutes reading time, 2 hours writing time
- You can bring one A4 page with notes on both sides
- You can also bring an unannotated paper-based dictionary (no approval required)

All material taught in the course is examinable. You will not be called upon to write code in the exam, though you may be asked to describe what provided code will do.

I will provide a quiz in Week 11 (with questions unrelated to the course) as a practice for ensuring you understand how to use Proctorio and answer the types of questions in the exam, and provide more information about Proctorio use in coming weeks.

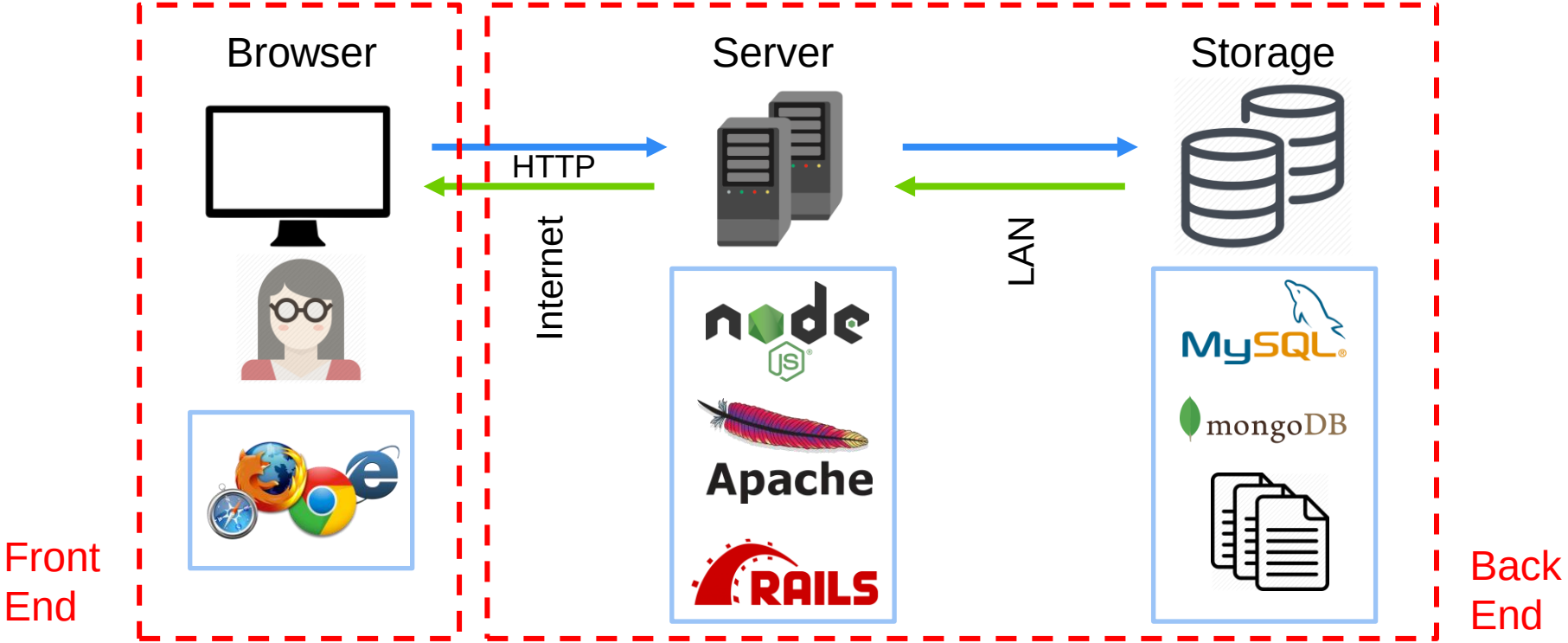
This information is accurate as at 2/5/23 but may change. I'm providing the information to help you start to plan but it is your responsibility to check timetabling as we get nearer to the end of semester and prior to the exam for final details.

Outline

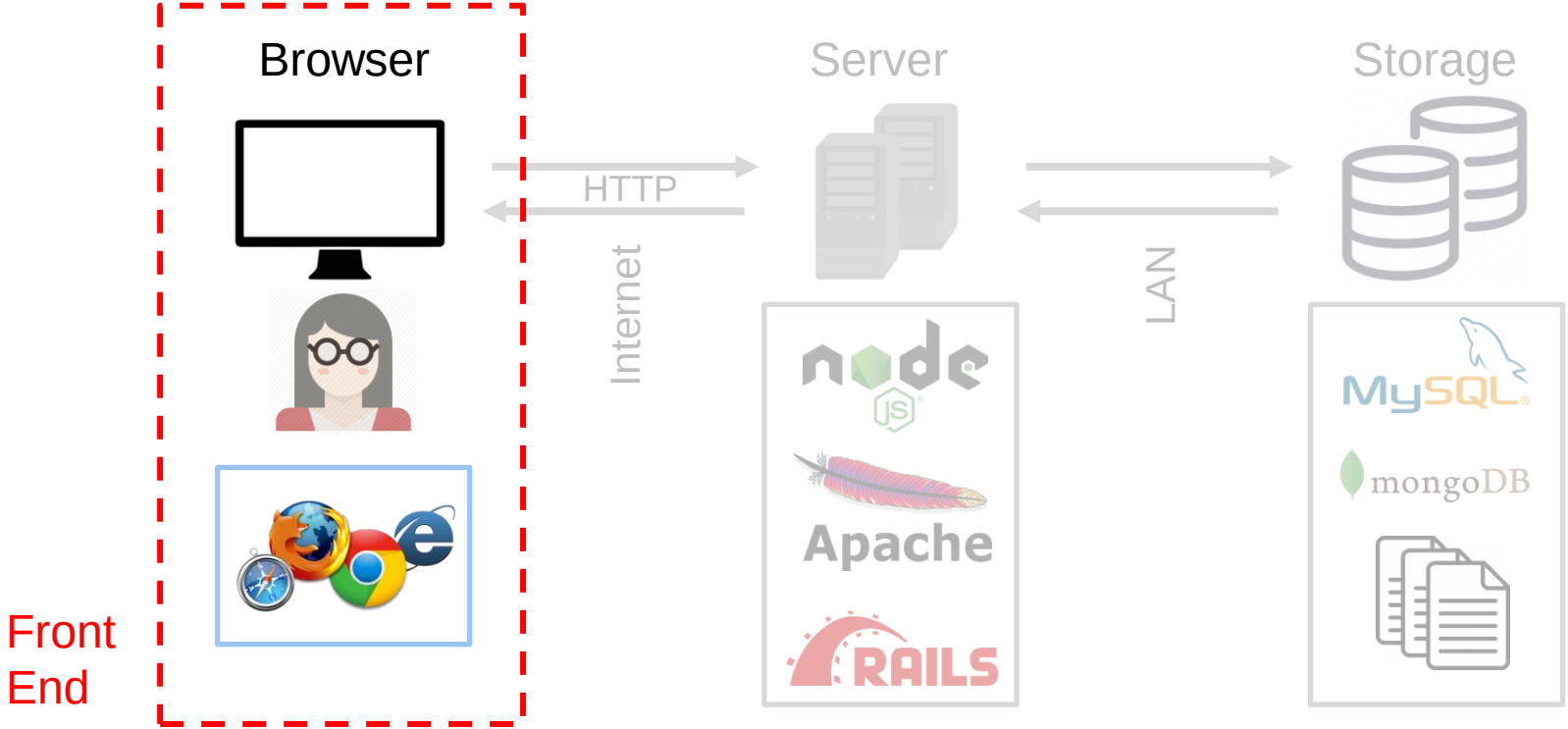
01	Front End Web Development
02	Back End Web Development
03	Common Stacks

(This will be partially covered today, and finished on Thursday)

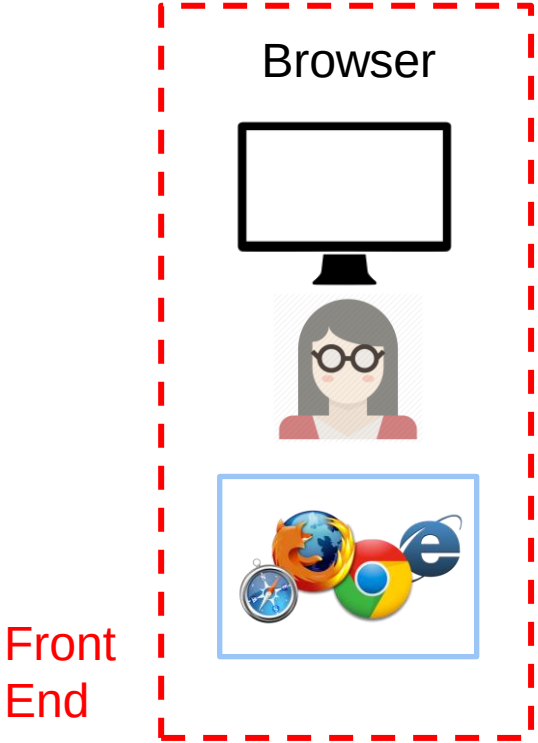
Full Stack Web Application Architecture



Full Stack Web Application Architecture



Front End Development



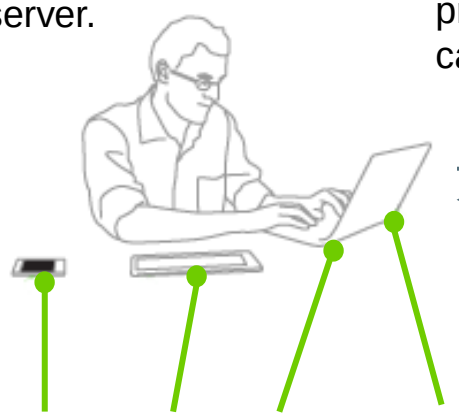
- Also called “Client-side” programming
- It is everything that users see and interact with in the browser.

Front End Development

1 A site is loaded in a browser from the server.

2 **Client-site scripts** Run in the browser and process requests without call-backs to the server

3 When a call to database is required, scripts send requests to the back end.



Responsive design allows a site to adapt to different devices.

Everything a user sees in the browser is a mix of **HTML, CSS, and Javascript**.

4 The **back-end server-side scripts** process the request, pull out data from database and send it back.

Good Web Applications

Design + Implementation

Design

- Good & Bad Design
- Style Guide
  - e.g. Material Design

Implementation

- HTML, CSS, Javascript
- Responsive Design
- CSS frameworks

Good Web Applications

Design

- Good & Bad Design
- Style Guide
  - e.g. Material Design

+

Implementation

- HTML, CSS, Javascript
- Responsive Design
- CSS frameworks

Good Web Applications - Design

Design

- Good & Bad Design
- Style Guide
  e.g. Material Design

Some Design Goals:

- Intuitive to use
  - Don’t need to take a course or read a user manual
- Accomplish task accurately and rapidly
  - Provide needed information and functionality
- Users like the experience
  - Joy rather than pain when using the application

Good Web Applications - Design

Design

- Good & Bad Design
- Style Guide
  e.g. Material Design

Some guiding design principles

- Be consistent
  - Cognitive load less for the user
- Provide context
  - User should not get lost in the app

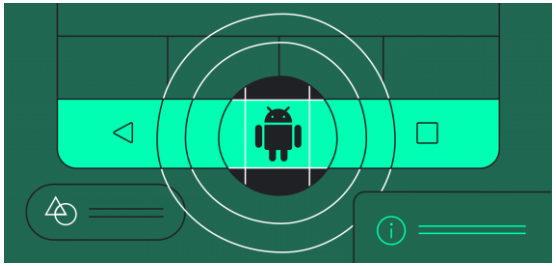
Consistency: Style guides & templates

- A style guide – covers the look and feel
  - Define style, user interactions, layout
- Patterns – do something multiple places in the same way
- Design templates – follow a familiar structure

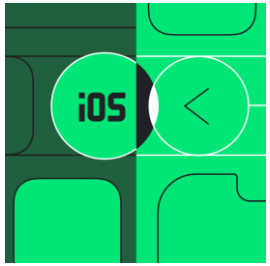
Style Guide Example

Material Design from Google

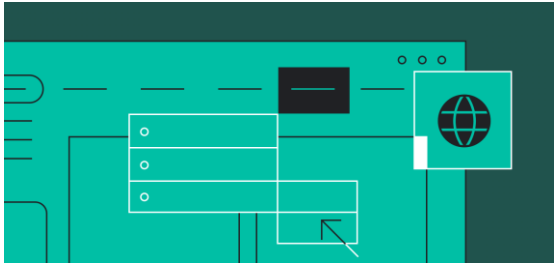
https://material.io/develop/



Android



iOS



Web

Style Guide Example

Material Design from Google

https://material.io/develop/

- Used in Google apps (e.g Android, web apps)
- Two goals:
  1. It unifies Google's numerous products
  2. It unifies Android app interfaces
- Focus on traditional print issues: grids, space, typography, colour etc
- Heavy use of animation to convey action

Style Guide Example

Material Design from Google

https://material.io/develop/

Material Design Foundations

- **Environment** – surfaces, depth, and shadows
- **Layout** – responsive layout grid, white space
- **Navigation** – navigation transitions, search
- **Colour** – suggestions for colours that work well together
- **Typography** – suggestions for point size, weight, spacing
- **Sound** – suggestions for sound attributes
- **Shape** – use shapes to direct attention
- **Motion** – show information, focus attention
- **Interaction** – map touch to actions

Style Guide Example

Material Design from Google

https://material.io/develop/

Colours

Style – Color

Red		Pink		Purple	
500	#F44336	500	#E91E63	500	#9C27B0
50	#FFEBEE	50	#FCE4EC	50	#F3E5F5
100	#FFCDD2	100	#F8BBD0	100	#E1BEE7
200	#EF9A9A	200	#F48FB1	200	#CE93D8
300	#E57373	300	#F06292	300	#BA68C8
400	#EF5350	400	#EC407A	400	#AB47BC
500	#F44336	500	#E91E63	500	#9C27B0
600	#E53935	600	#D81B60	600	#8E24AA
700	#D32F2F	700	#C2185B	700	#7B1FA2
800	#C62828	800	#AD1457	800	#6A1B9A
900	#B71C1C	900	#880E4F	900	#4A148C
A100	#FF8A80	A100	#FF80AB	A100	#EA80FC
A200	#FF5252	A200	#FF4081	A200	#E040FB
A400	#FF1744	A400	#F50057	A400	#D500F9

Source: Google Material Design guidelines

Style Guide Example

Material Design from Google

https://material.io/develop/

Icons

Source: System Icons collection from Material Design Docs by Google

Style Guide Example

Material Design from Google

https://material.io/develop/

Writing and Typography

Roboto Thin
Roboto Light
Roboto Regular
Roboto Medium
Roboto Bold
Roboto Black
Roboto Thin Italic
Roboto Light Italic
Roboto Italic
Roboto Medium Italic
Roboto Bold Italic
Roboto Black Italic

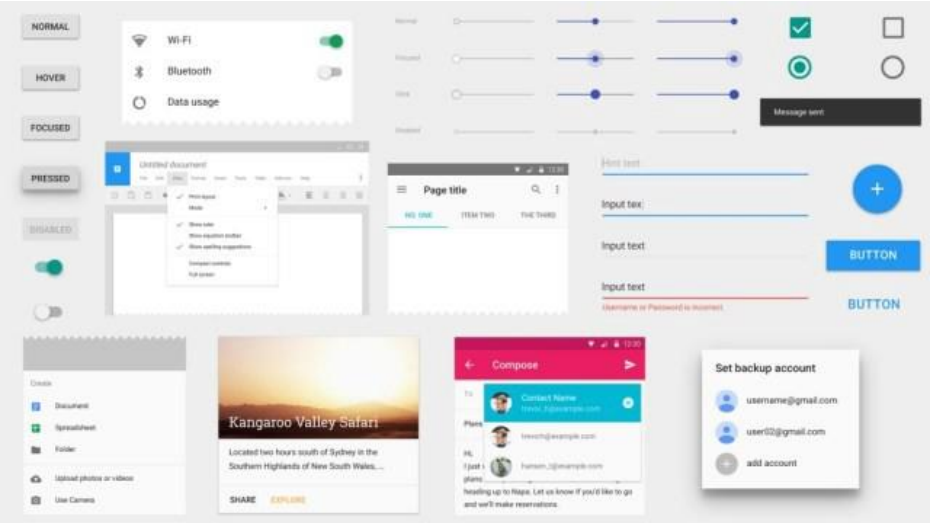
Source: Google Material Design Docs

Style Guide Example

Material Design from Google

https://material.io/develop/

Design Components



Source: Google Material Design Docs

Style Guide Example

Material Design from Google

https://material.io/develop/

How to use it for Web?

Include the material design css and js files.
You can download the files from material design lite

```
<link rel="stylesheet"
href="https://fonts.googleapis.com/icon?family=Material+Icons"
>

<link rel="stylesheet"
      href="https://code.getmdl.io/1.3.0/material.indigo-
pink.min.css">

<script defer
src="https://code.getmdl.io/1.3.0/material.min.js"></script>
```

Good Front-end Applications

Design + **Implementation**

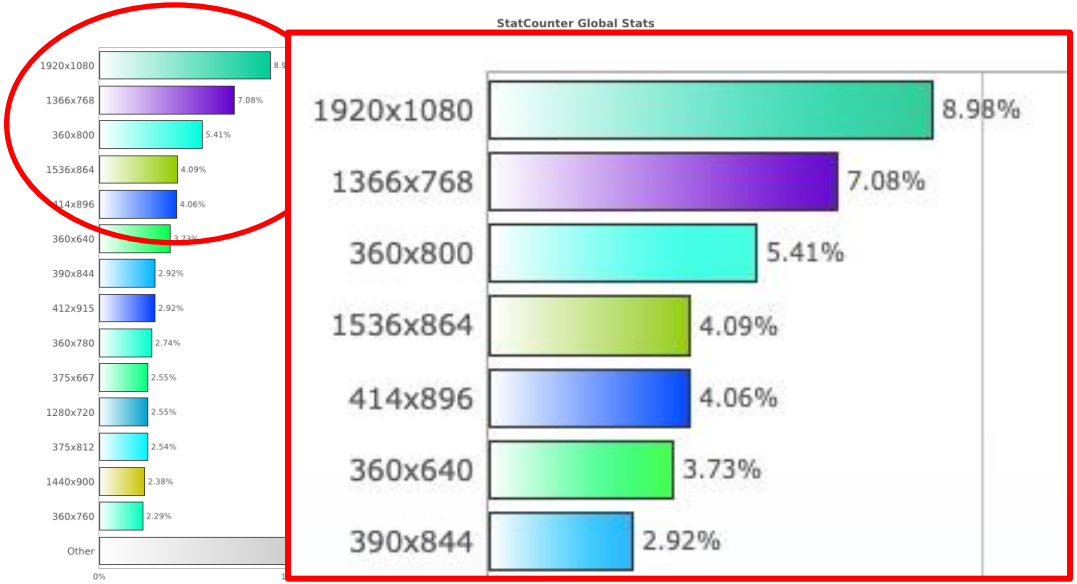
- Good & Bad Design
- Style Guide
  e.g. Material Design

- HTML, CSS, Javascript
- Responsive Design
- CSS frameworks

Responsive Design

Screen Resolution Stats

Top 2: 1920x1080 [8.98%], 1366x768 [7.08%]



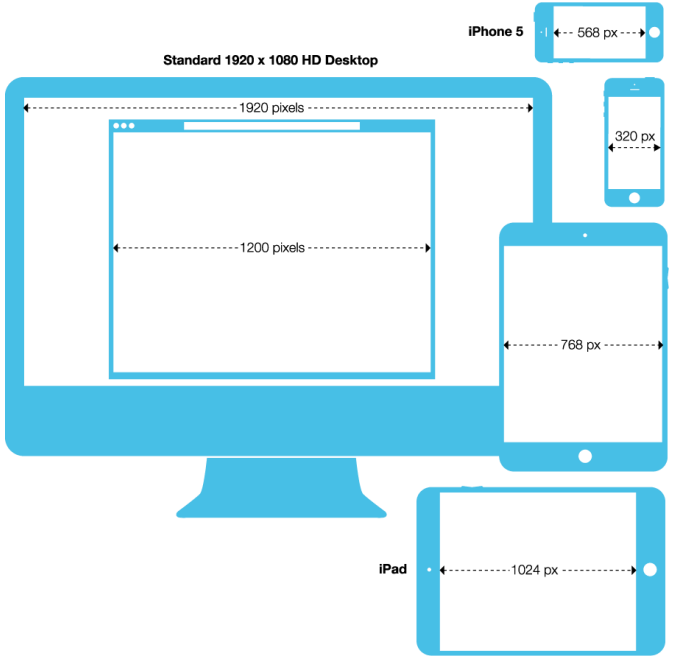
Implementation

- HTML, CSS, Javascript
- Responsive Design
- CSS frameworks

Source: Stat Counter

Responsive Design

Build N versions of each web application?



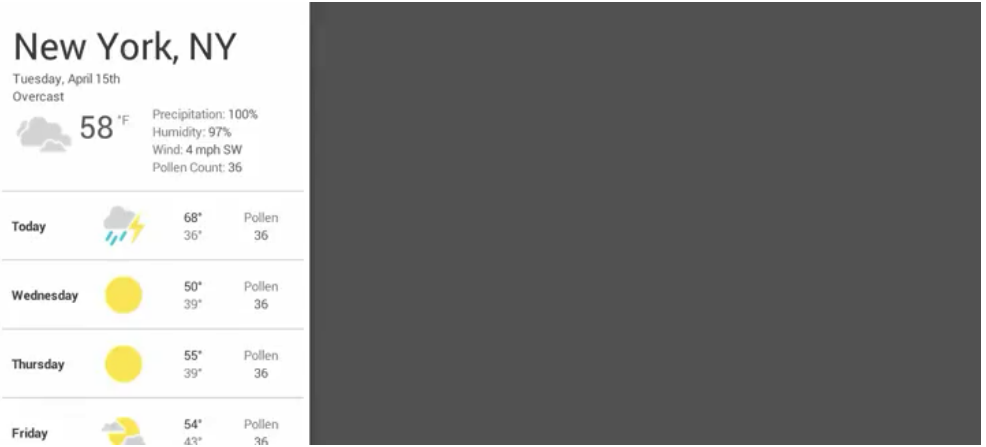
Source: Design Insights

Implementation

- HTML, CSS, Javascript
- Responsive Design
- CSS frameworks

Responsive Design

Example of Responsive Design



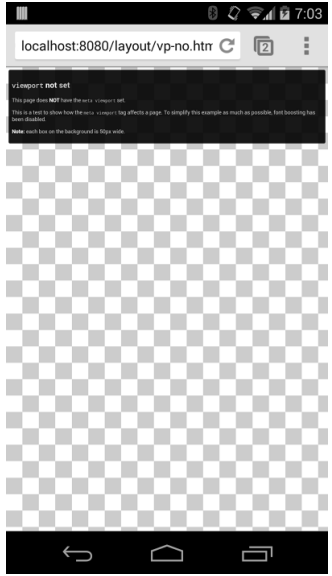
Source: Design Insights

Implementation

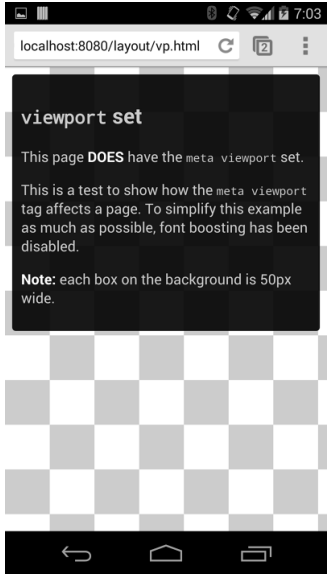
- ✓ HTML, CSS, Javascript
- ➡ Responsive Design
- CSS frameworks

Responsive Design

Pages optimized for a variety of devices must include a meta viewport tag in the head.



Page without a viewport set

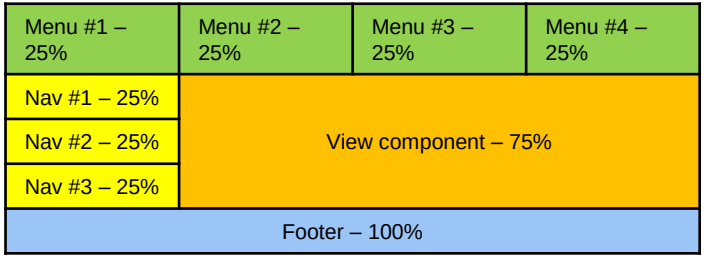


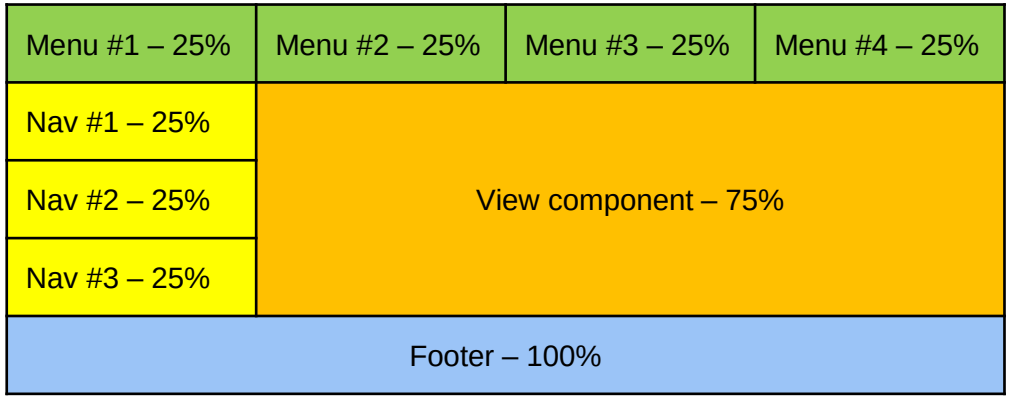
Page with a viewport set

```
<meta name="viewport"
content="width=device-width,
initial-scale=1.0">
```

1 **Set the view port**

Responsive Design






```
.col-1 {width: 8.33%;}
.col-2 {width: 16.66%;}
.col-3 {width: 25%;}
.col-4 {width: 33.33%;}
.col-5 {width: 41.66%;}
.col-6 {width: 50%;}
.col-7 {width: 58.33%;}
.col-8 {width: 66.66%;}
.col-9 {width: 75%;}
.col-10 {width: 83.33%;}
.col-11 {width: 91.66%;}
.col-12 {width: 100%;}
```

2 **Add grid layout system with relative (e.g. 50%) rather than absolute (e.g. 50pt) measures**

Responsive Design

Auto-scale images and videos to fit in screen region

Source: Responsive Images

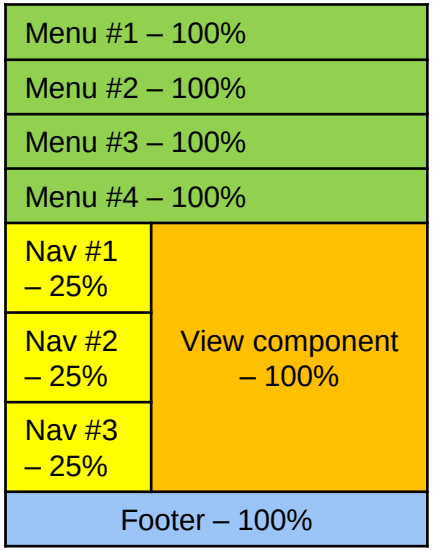
```
img, embed, object, video {
    width: 100%;
    height: auto;
}
```

3 **Make components support relative sizes**

Responsive Design

CSS Breakpoints to control layout

```
@media only screen and (min-width: 768px) {
    /* tablets and desktop layout */
}
@media only screen and (max-width: 767px) {
    /* phones layout */
}
@media only screen and (max-width: 767px) and
(orientation: portrait) {
    /* portrait phones layout */
}
```



4 **Add @media rules based on screen sizes**

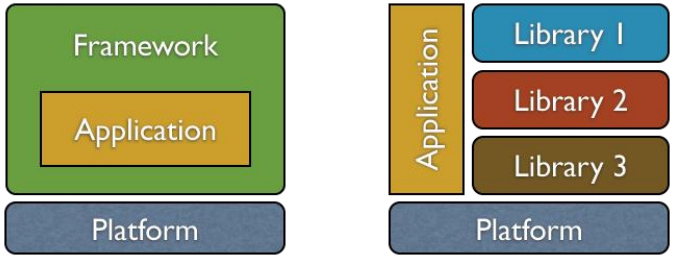
CSS Frameworks

A framework is a standardized set of concepts, practices and criteria for dealing with a common type of problem.

Difference

Libraries: You are in control of when the library should perform a particular function.

Frameworks: The control flow is in the framework and you can customize it.



Source: Tom Lokhorst's blog

Implementation

- HTML, CSS, Javascript
- Responsive Design
- CSS frameworks

CSS Frameworks

Advantages

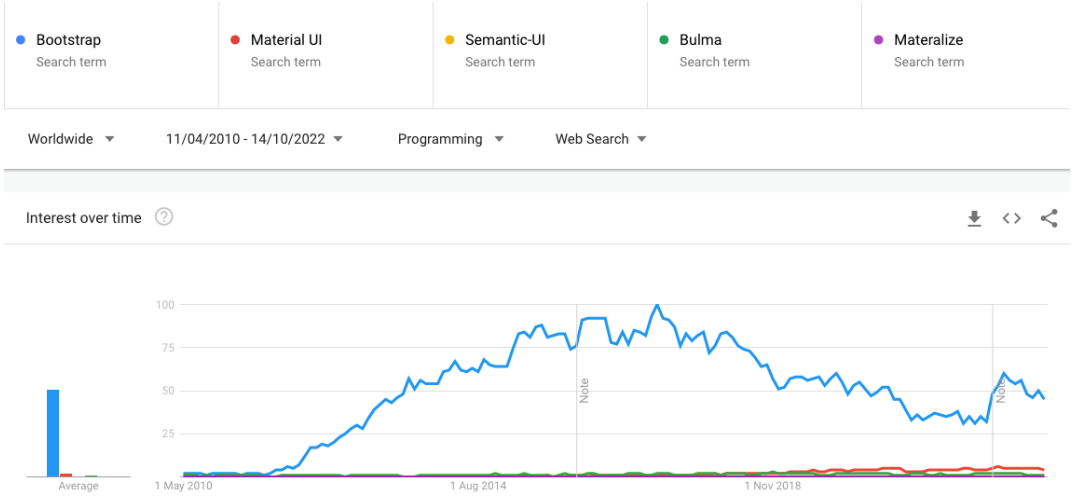
- Easier code maintenance
- Coherent organizational structure
- Responsive media queries
- Uniform styling across buttons, forms etc.
- Consistent set of fonts and icons

Source: troxler's github

Implementation

- ✓ HTML, CSS, Javascript
- ✓ Responsive Design
- ➡ CSS frameworks

CSS Frameworks



Source: Google Trends

CSS Frameworks GitHub stars

- Bootstrap: 160K Stars
- Material UI: 82K Stars
- Semantic-UI: 50.2K Stars
- Bulma: 46.5K Stars
- Materialize: 38.7K Stars

CSS Frameworks

What are the best CSS frameworks to learn?

Depends on your skills. Having a solid understanding of HTML, CSS, and JavaScript is still the most important skill overall.

Bootstrap
https://getbootstrap.com/

Materialize
https://materializecss.com/

Bulma
https://bulma.io/

Semantic UI
https://semantic-ui.com/

Material UI
https://material-ui.com/

Good Front-end Applications

Design + **Implementation**

- Good & Bad Design
- Style Guide
  - e.g. Material Design

- HTML, CSS, Javascript
- Responsive Design
- CSS frameworks

Full Stack Web Application Architecture