/FRONTEND WORKSHOP

How to frontend

INDEX.HTML

/WHAT IS THIS TOPIC ABOUT?

/Albert Groothedde

Frontend Architect @Kadena

/Steven Straatemans

Frontend Developer @Kadena

/THE PLAN

/01 /ASSIGNMENT

- Show of your skills

/02 /HTML, CSS and JS

- Frontend building blocks

/03 /REACT

- And other frameworks

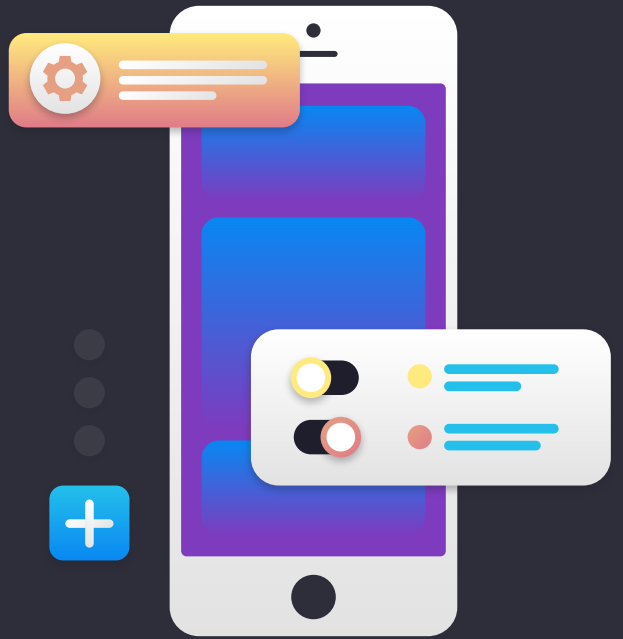
/04 /NEXTJS

- Lets build an app

/01

/ASSIGNMENT

HTML, CSS and Javascript



INDEX.HTML

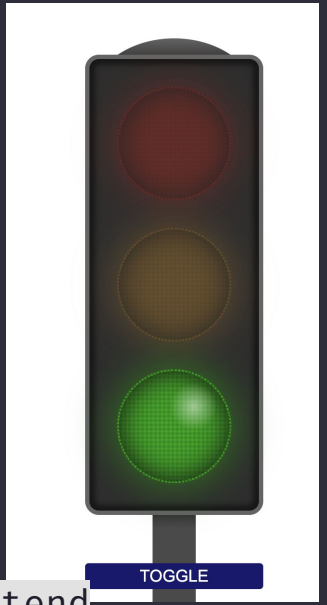
/CREATE A TRAFFIC LIGHT

Create a traffic light in CSS, HTML.

- Think about the HTML (Semantics)
- Make it nice and shiny, show off your skills

Extra:
Add functionality to toggle the lights on and off with JS.

- When off, the orange light blinks



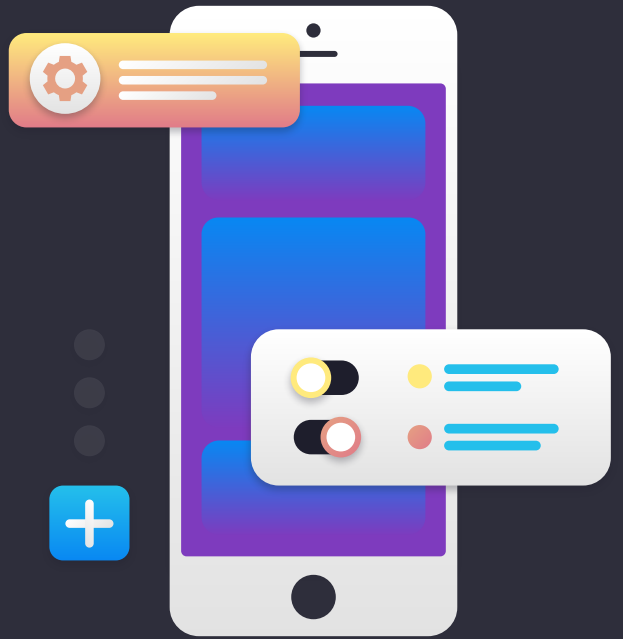
https://github.com/alber70g/harvest-workshop-web-frontend

Toggle Mirror/Extended cmd+fn+f1

/02

/HTML,CSS,JS

HTML, CSS and Javascript




<HTML>




/HTML

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

```
<ol>
  <li/>
  <li/>
  <li/>
</ol>
```

What is the difference?
Is one better than the other?

/HTML Semantics

- More than 100 different
- Why use different tags?

https://developer.mozilla.org/nl/docs/Web/HTML/Element

<CSS>

Cascading

/CSS

```
<ol id="trafficlight" class="trafficlight">
  <li class="light red active" />
  <li class="light yellow" />
  <li class="light green" />
</ol>
```

INDEX.HTML

/CSS Selectors

```
Class                    ⇒ .classname
Id                       ⇒ #id
All                      ⇒ *
Element                  ⇒ p, h1, article, div
Element > element        ⇒ div > p
Element + element        ⇒ p + p
Element ~ element        ⇒ p ~ p
Attribute                ⇒ [target=_blank]
Pseudo-selectors         ⇒ :active, ::after,
                             :first-child,
                             ::first-letter
```

/CSS Cascading

- Most important rule

```
<style>
    div {
        background-color: green;
    }

    div {
        background-color: red;
    }
</style>

<div>text</div>
```

/CSS Cascading

So what happens now?

```
<style>
  ul#mylist > li {
    background-color: green;
  }

  .favourite {
    background-color: red;
  }
</style>

<ul id="mylist">
  <li class="favourite">option 1</li>
  <li>option 2</li>
  <li>option 3</li>
</ul>
```

/CSS Specificity

	Inline styling	ID	Classes	Elements
	0	1	0	2
	0	0	1	0

/CSS Specificity

```
<style>
  ul#mylist > li {
    background-color: green;
  }
</style>

<style>
  .favourite {
    background-color: red;
  }
</style>

<style>
  #mylist .favourite {
    background-color: red;
  }
</style>
```

Inline styling	ID	Classes	Elements
0	1	0	1
0	0	1	0
0	1	1	0

/CSS example

```
<div class="green red">text</div>
```

```
<style>
  .red {
    background: red;
  }
  .green {
    background: green;
  }
</style>
```

/CSS Inheritance

```
<style>
  p {
    color: blue;
  }
</style>
```

```
<p>He-man: I have the <em>power</em>!!</p>
```

/CSS Inheritance

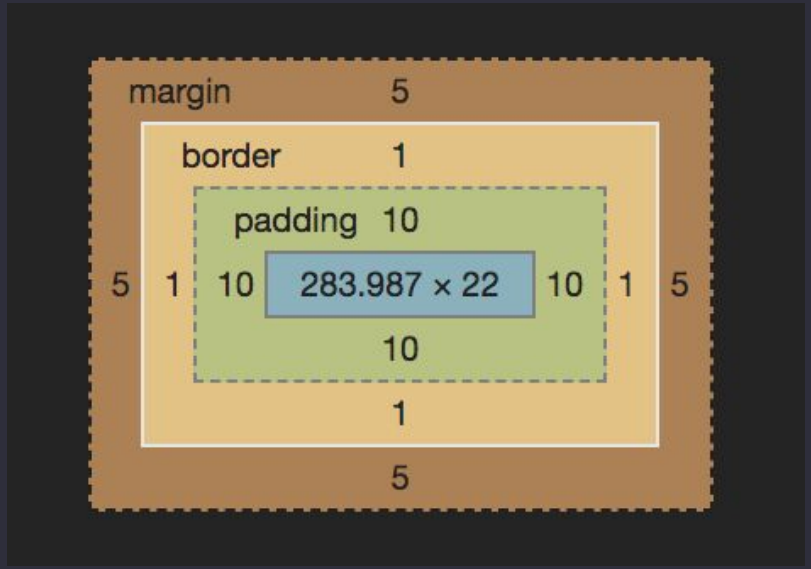
```
<style>
  p {
    border: 1px solid blue;
  }
</style>
```

```
<p>He-man: I have the <em>power</em>!!</p>
```

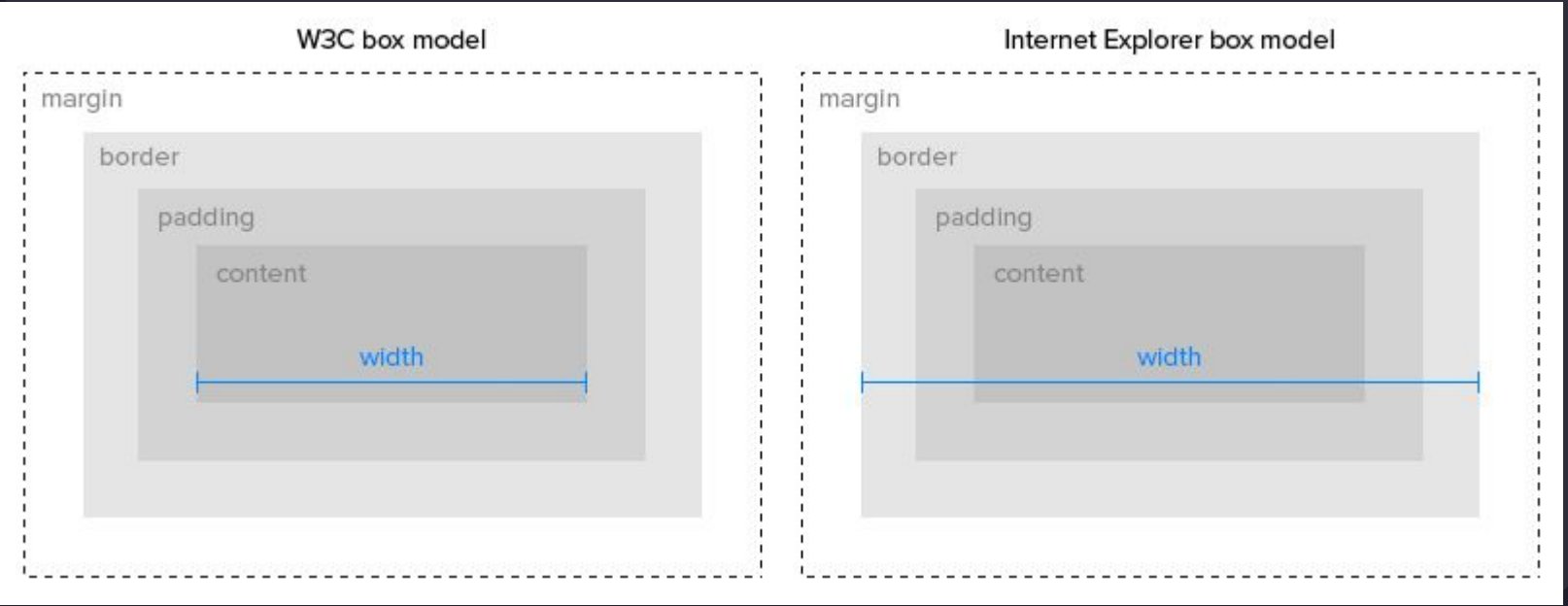
/CSS Inheritance Cheat sheet

- border-collapse
- border-spacing
- caption-side
- color
- cursor
- direction
- empty-cells
- font-family
- font-size
- font-style
- font-variant
- font-weight
- font-size-adjust
- font-stretch
- font
- letter-spacing
- line-height
- list-style-image
- list-style-position
- list-style-type
- list-style
- orphans
- quotes
- tab-size
- text-align
- text-align-last
- text-decoration-color
- text-indent
- text-justify
- text-shadow
- text-transform
- visibility
- white-space
- widows
- word-break
- word-spacing
- word-wrap

/CSS Box model



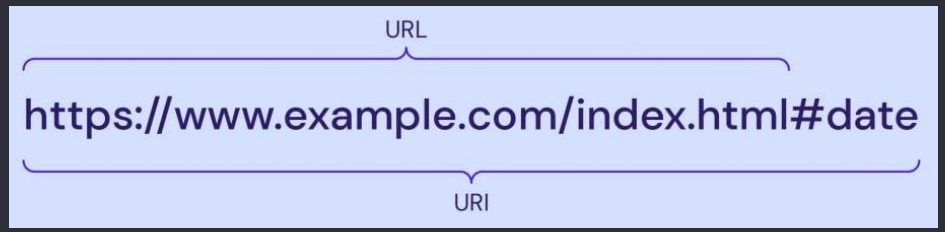
/CSS Box model



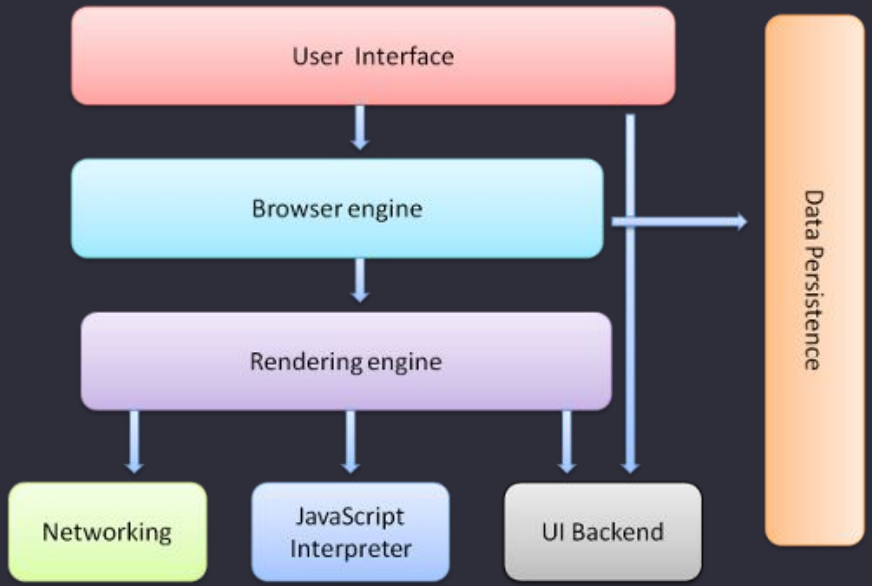
/CSS Box model

```
<style>
  div.content {
    width: 400px;
    voice-family: ""}""";
    voice-family: inherit;
    width: 300px;
  }
</style>
```

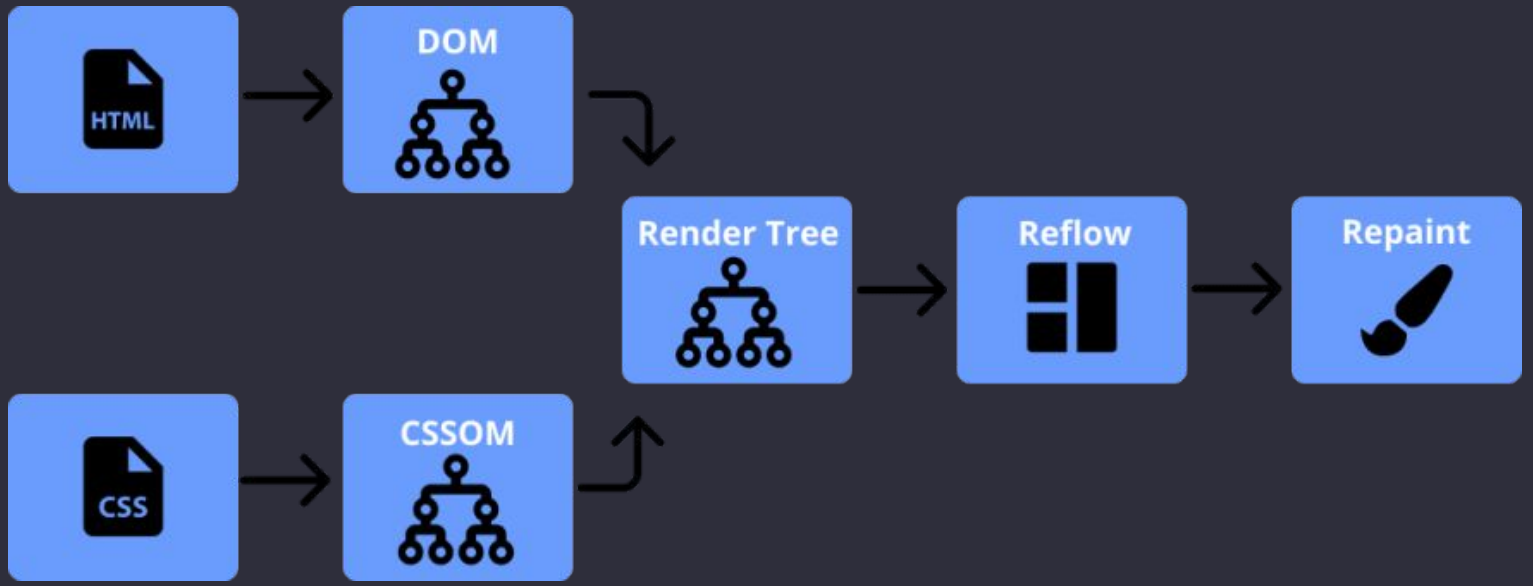
/HOW IS THIS ALL RELEVANT?



/THE BROWSER


User Interface
Browser engine
Rendering engine
Networking
JavaScript Interpreter
UI Backend
Data Persistence

/THE RENDERING ENGINE



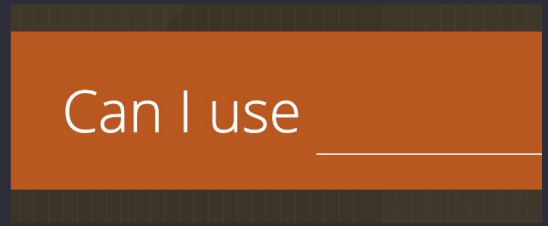
<JAVASCRIPT>

/JS = Ecmascript?

- ECMAscript is the standard
  - Describes the core API
  - `Array.forEach(el ⇒ el + 1)`
    `for(){}`
    `const { x } = y`
- Javascript is an implementation
- Different from Web APIs

/JS versions

- Only since 2012 all browsers Support EcmaScript 5.1
- In 2015 came version 6 (ES2015)
- Since then ECMAScript releases a version every year



/JS Transpilers

- Always use the latest and greatest features
- Determine what browser versions you want to support

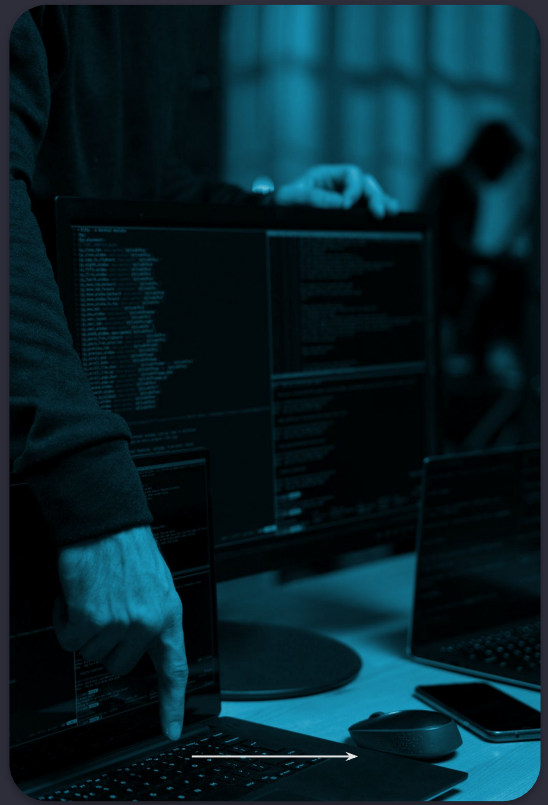
/JS Typescript

- Typescript is a typed superset of Javascript that compiles to plain Javascript
- Adds types to Javascript
- More and more used
- Great tools
  - In node and browser
  - Prevents issues on runtime
  - Great IDE support
  - Has become the standard in development

/TS Why?

- No need to keep different browser support in mind
- Better readable code
- Maintainability
- Less chance at runtime errors, due to type safety

/HOW DOES JS RUN IN YOUR BROWSER



/JS Engines

V8
by Google

Chrome
Edge 79+
Chromium-based
Cross OS Apps
NodeJs

Spider Monkey
by Mozilla

Firefox
GNOME Shell

Chakra Core
by Microsoft

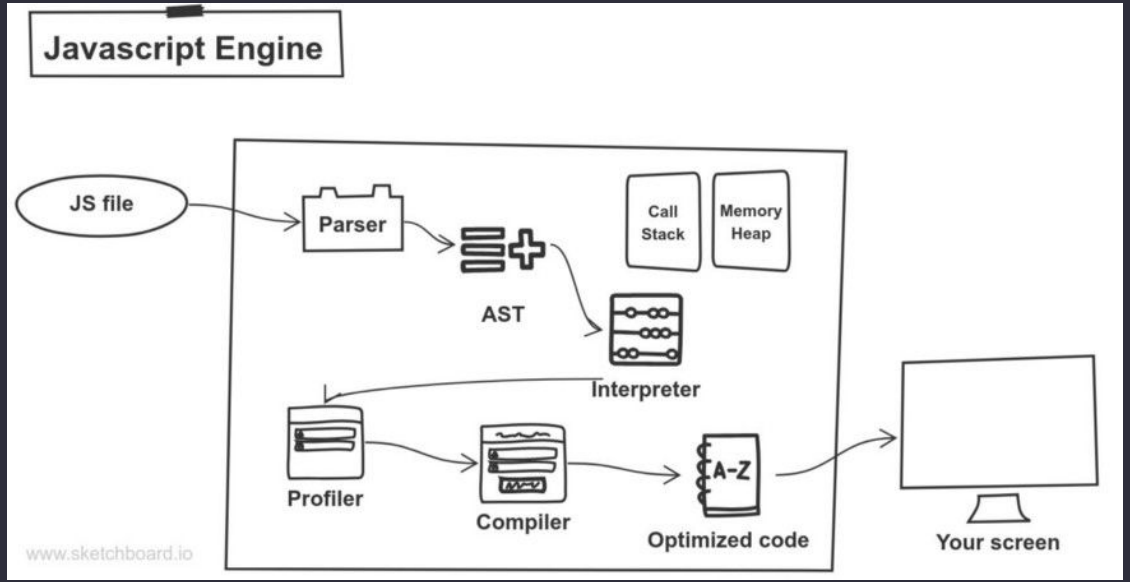
Edge (Edge ⩽ 78)

Javascript Core
Open Source

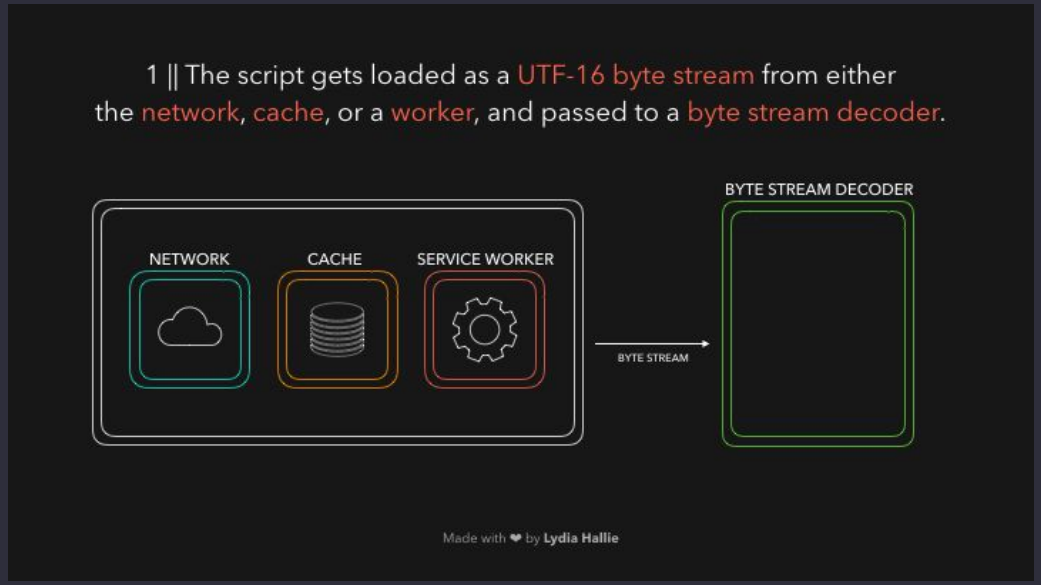
Safari
NW.js
Bun
(JavascriptCore)

/HOW IS THIS RELEVANT?

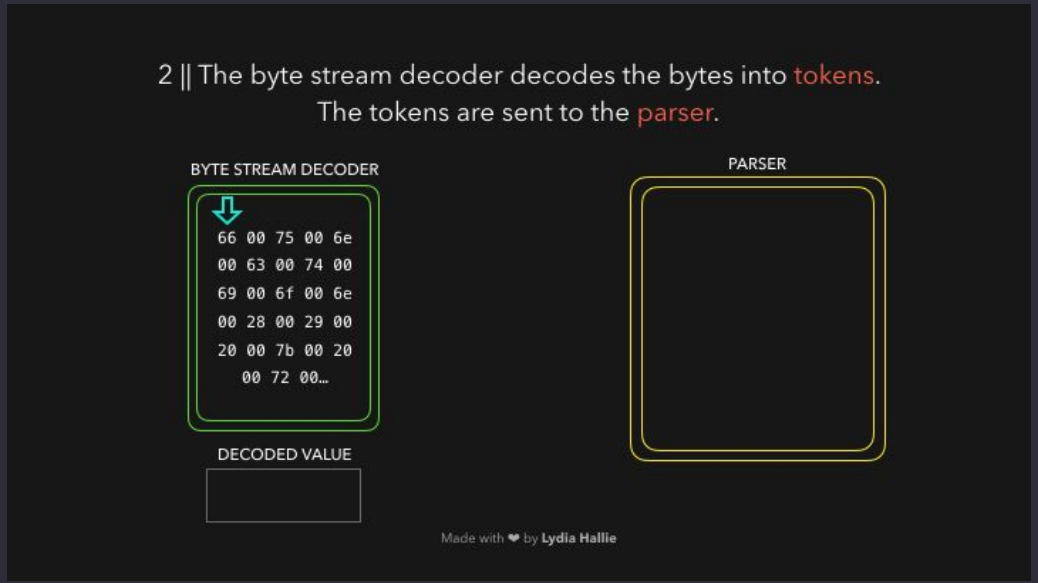
/JS Engine


Javascript Engine
JS file
Parser
AST
Call Stack
Memory Heap
Interpreter
Profiler
Compiler
A-Z
Optimized code
Your screen
www.sketchboard.io

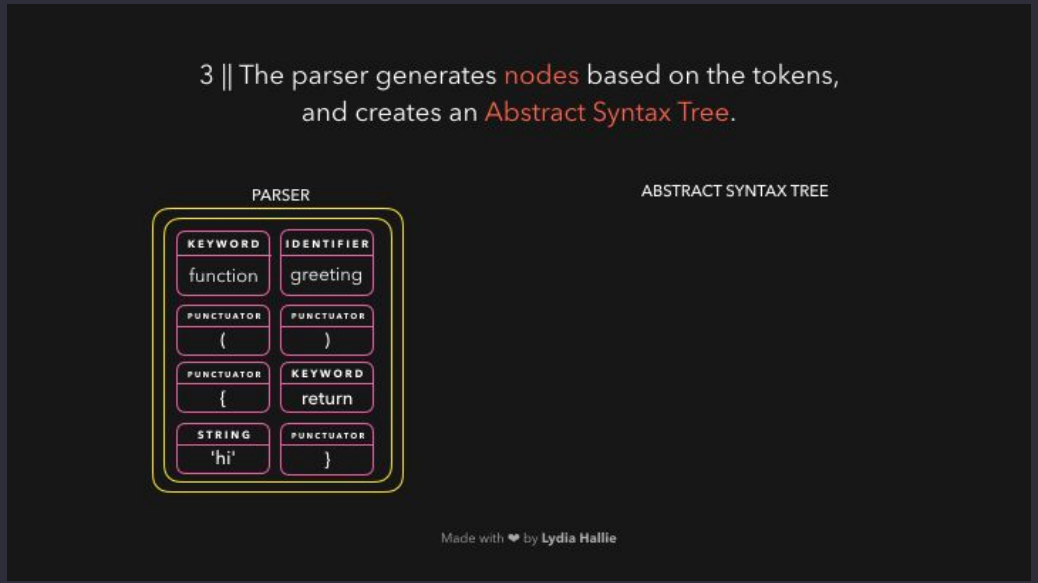
/JS Engine


1 || The script gets loaded as a UTF-16 byte stream from either the network, cache, or a worker, and passed to a byte stream decoder.
NETWORK
CACHE
SERVICE WORKER
BYTE STREAM
BYTE STREAM DECODER
Made with ❤ by Lydia Hallie

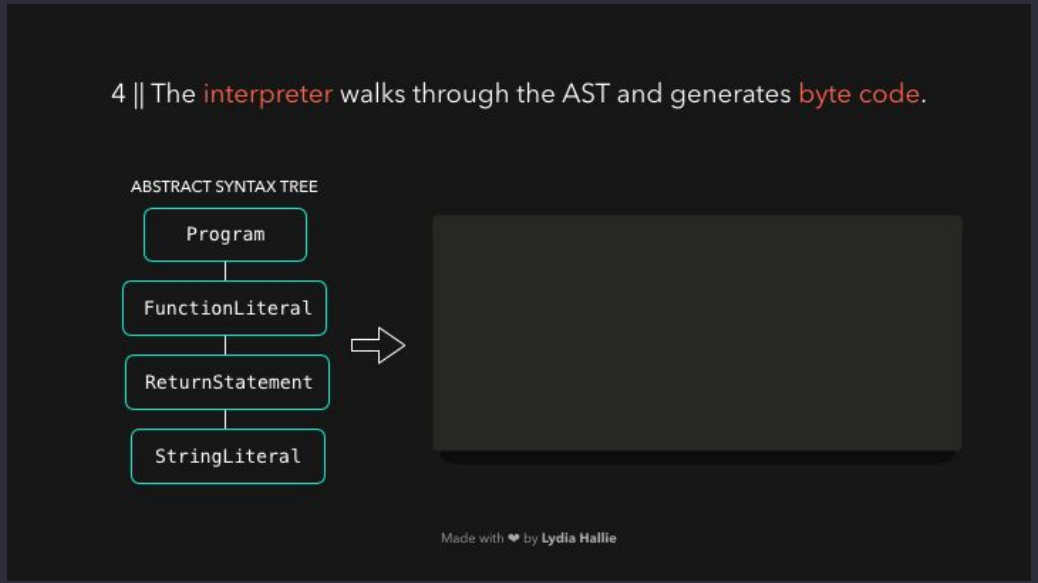
/JS Engine


2 || The byte stream decoder decodes the bytes into tokens.
The tokens are sent to the parser.
BYTE STREAM DECODER
66 00 75 00 6e
00 63 00 74 00
69 00 6f 00 6e
00 28 00 29 00
20 00 7b 00 20
00 72 00…
DECODED VALUE
PARSER
Made with ❤ by Lydia Hallie

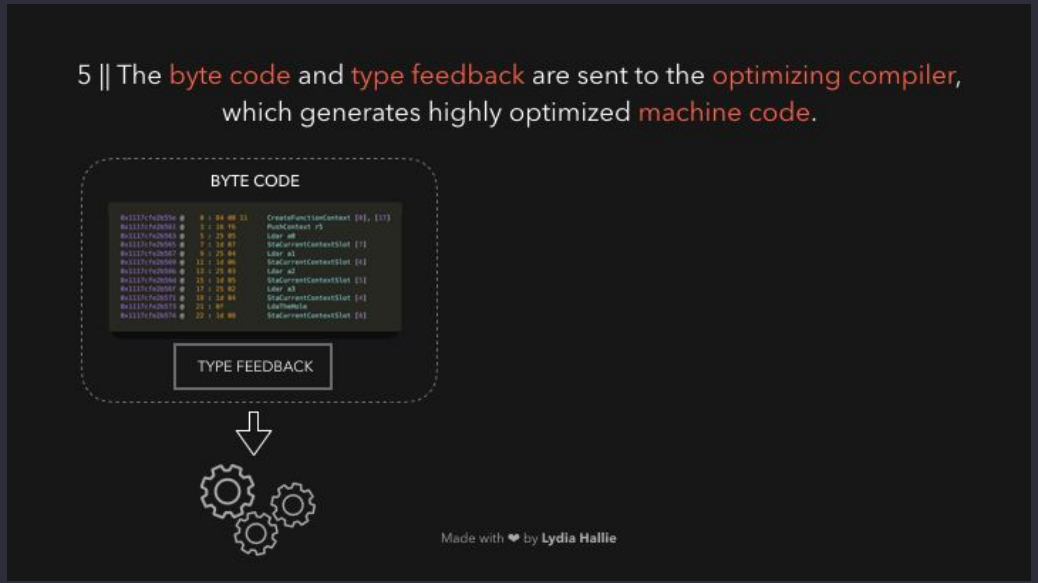
/JS Engine


3 || The parser generates nodes based on the tokens,
and creates an Abstract Syntax Tree.
PARSER
KEYWORD
function
IDENTIFIER
greeting
PUNCTUATOR
(
PUNCTUATOR
)
PUNCTUATOR
{
KEYWORD
return
STRING
'hi'
PUNCTUATOR
}
ABSTRACT SYNTAX TREE
Made with ❤ by Lydia Hallie

/JS Engine


4 || The interpreter walks through the AST and generates byte code.
ABSTRACT SYNTAX TREE
Program
FunctionLiteral
ReturnStatement
StringLiteral
Made with ❤ by Lydia Hallie

/JS Engine


5 || The byte code and type feedback are sent to the optimizing compiler, which generates highly optimized machine code.
BYTE CODE
TYPE FEEDBACK
Made with ❤ by Lydia Hallie

/JS Engine

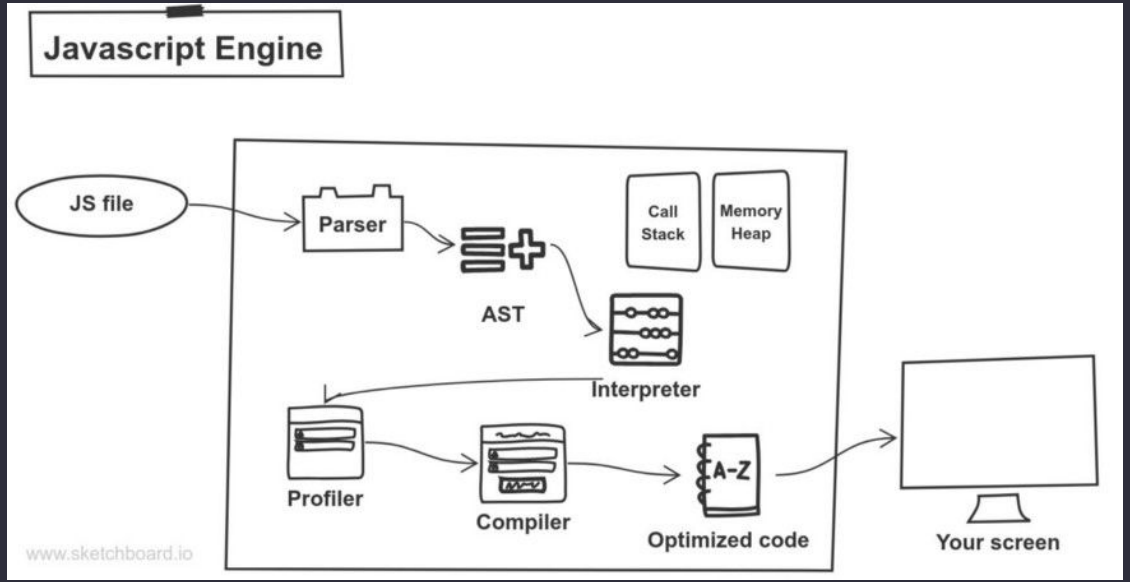
```
function sum(a,b) {
    return a + b;
}

sum(1, 2);
```

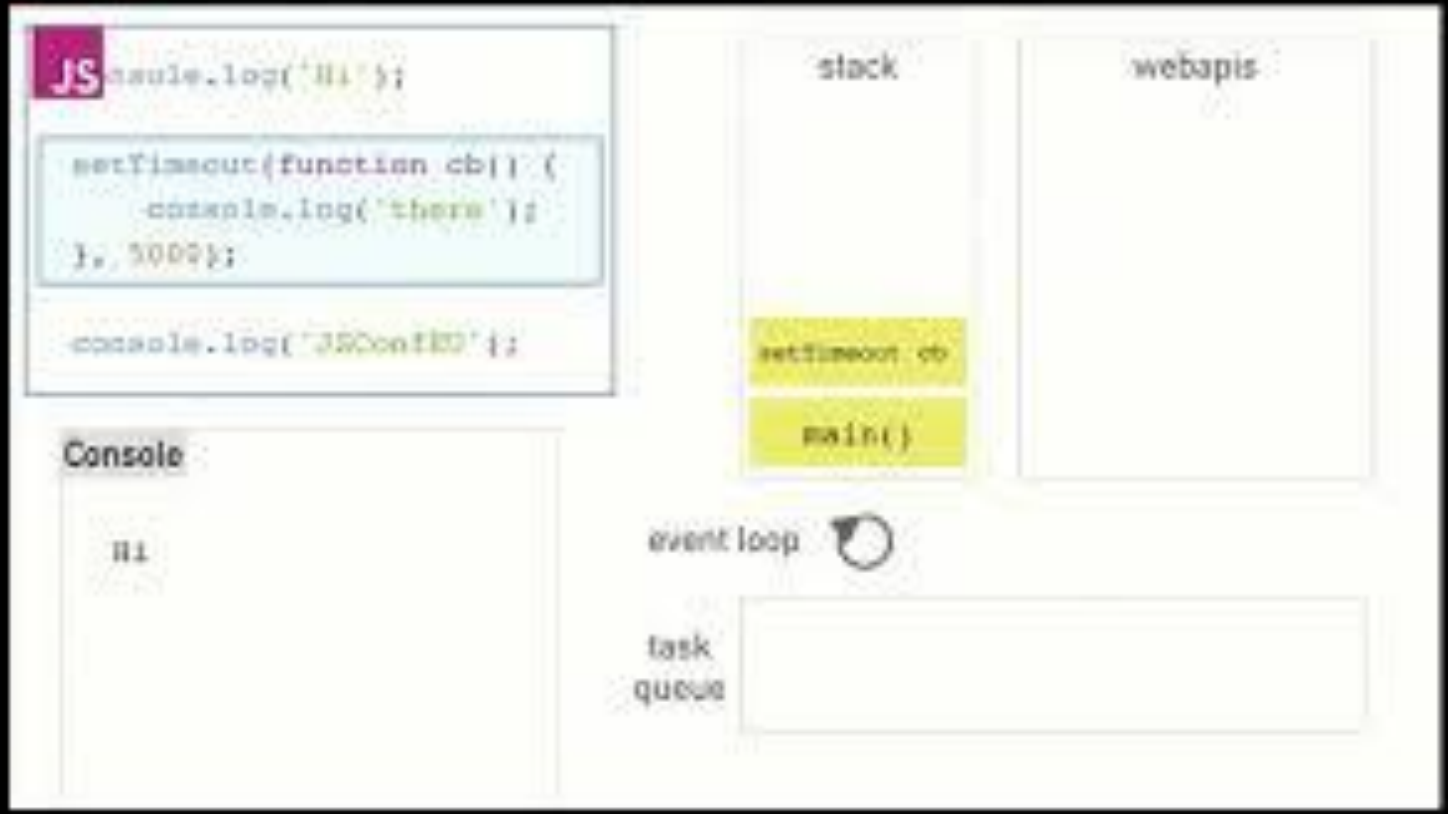
```
function sum(a,b) {
    return a + b;
}

sum('1', 2);
```

/JS Engine


Javascript Engine
JS file
Parser
AST
Call Stack
Memory Heap
Interpreter
Profiler
Compiler
A-Z
Optimized code
Your screen
www.sketchboard.io

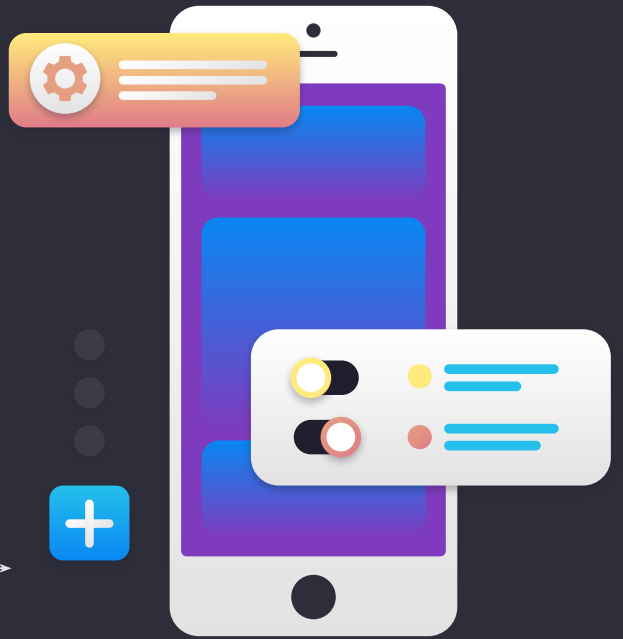
/JS Engine - Event Loop



/03

/FRAMEWORKS

React, Svelte, Angular, Vue



/FRAMEWORKS why?

- Code Reusability and Maintainability
- Declarative Syntax
- Component-Based Architecture
- State Management

/Maintainability

- Global variables for state
- Manual DOM manipulation
- No structure
- Hard to scale

```
const button = document.querySelector("#toggleButton");
const trafficlight = document.querySelector("#trafficlight");
const lights = trafficlight.querySelectorAll("li");
let activeLight = 0;
let isActive = true;
let lightInterval;

const toggleLights = () => {
  isActive = !isActive;
  if (!isActive) {
    stopLights();
  } else {
    startLights();
  }
};

button.addEventListener("click", toggleLights);

const changeLight = () => {
  lights.forEach((light, idx) => {
    light.classList.remove("active");

    if (idx === activeLight) {
      light.classList.add("active");
    }
  });

  activeLight === 0
    ? (activeLight = lights.length - 1)
    : (activeLight = activeLight - 1);
};

const stopLights = () => {
  clearInterval(lightInterval);
  lights.forEach((light, idx) => {
    light.classList.remove("active");
  });
  lights[1].classList.add("blink");
};
const startLights = () => {
  activeLight = 0;
  changeLight();
  lights[1].classList.remove("blink");
  lightInterval = setInterval(changeLight, 2000);
};
startLights();
```

INDEX.HTML

/Declarative Syntax

Angular

```
@Component({
  selector: 'app-root',
  template: `<ng-template>
    <h1>Items</h1>
      <ul>
        @for (item of items; track item.id; let index = $index) {
          <li>{index + 1}: {item.name} x {item.qty}</li>
        }
    </ul>
  </ng-template>`})
export class ItemsComponent {
  items = [
    { id: 3, name: "orange", qty: 7 }
  ]
}
```

React

```
const ItemsComponent = () => {
  const items = [
      { id: 3, name: "orange", qty: 7 }
  ]
  return <>
    <h1>Fruit</h1>
    <ul>
      {items.map((item, index) =>
        <li>{index + 1}: {item.name} x {item.qty}</li>)
      )}
    </ul>
  </>;
}
```

Svelte

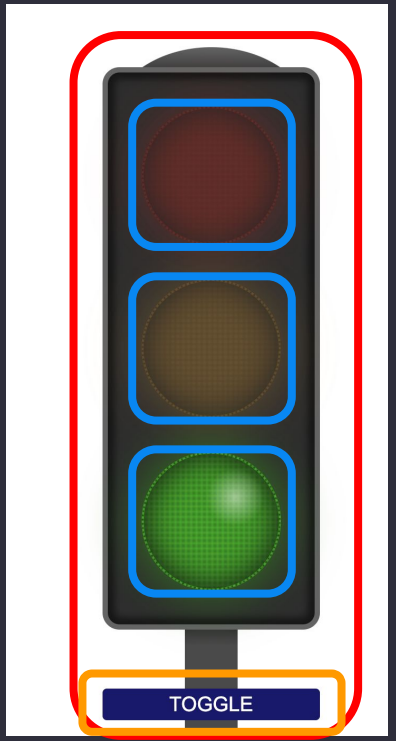
```
<script>
    const items = [
        { id: 3, name: "orange", qty: 7 }
    ]
</script>
<h1>Fruit</h1>
<ul>
  {#each items as item, index}
    <li>
      {index + 1}: {item.name} x {item.qty}
    </li>
  {/each}
</li>
```

<REACTJS>

INDEX.HTML

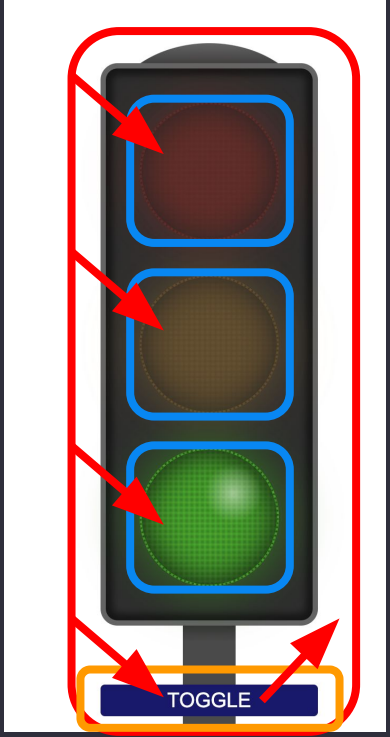
/Component Based

- Traffic Light Component
- Light Component
- Button Component



/State Management

- Traffic Light Component
  - Which light should be on
  - When the light be on
  - Which color is the light
- Light Component
  - Takes color as an argument
- Button Component
  - Takes a callback as an argument



/Live Coding

`Counter` Component

Please, Follow Along!

- Counter component w/ state
- Separate into components, passing state
- Style the components

`github.com/alber70g/harvest-workshop-web-frontend`

/UseState

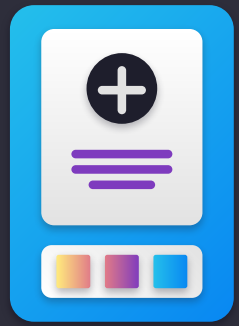
```
const [count, setCount] = useState(0);
```

/UseEffect

```
useEffect(() => {
    // This runs after every render
    document.title = `You clicked ${count} times`;

    return () => {
      // Cleanup, if necessary
    };
  }, [count]); // Only re-run the effect if count changes
```

<LETS BUILD STUFF>



INDEX.HTML

/Port your Traffic Light

Keep in mind:

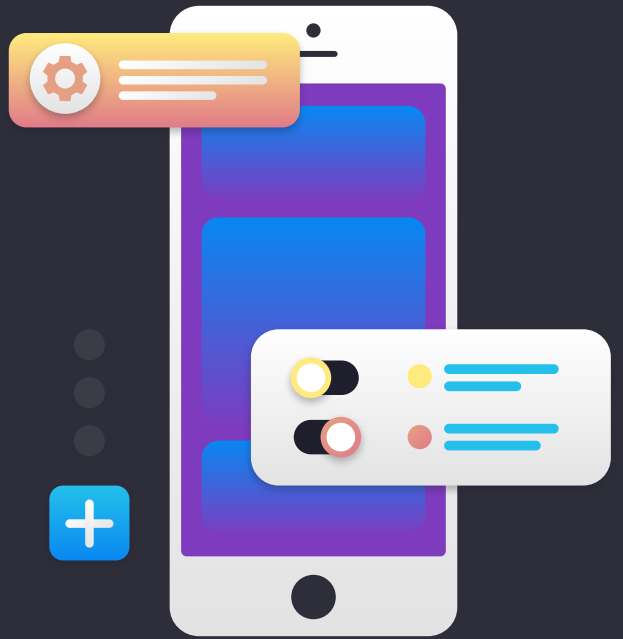
- What parts of the traffic light can be reusable components
- What props do I need to add
- Where do I keep my state

- Make sure the whole trafficlight is reusable

/04

/NEXTJS

Lets build an app

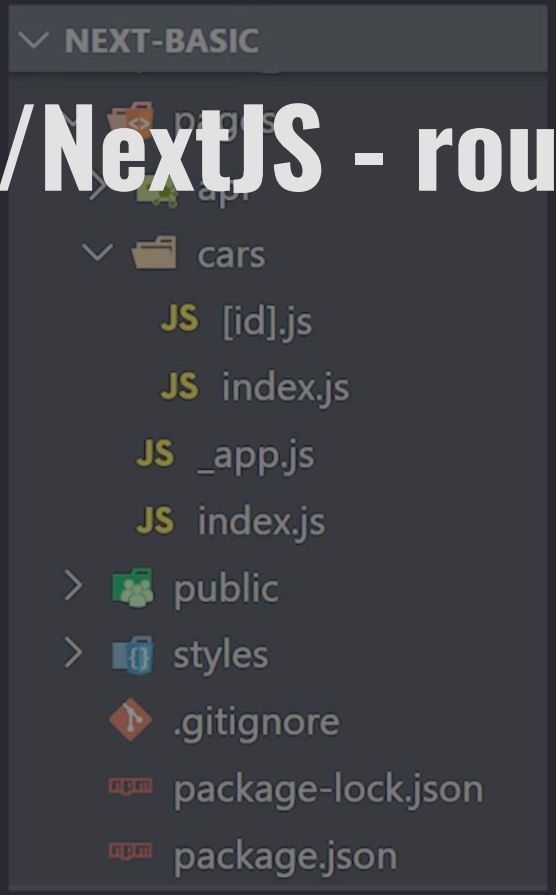


/NextJS - what?

Development
- 0 config
- file-structure = routing
- Hot Module Reloading (no browser reloads)
- Styling
- API routes

Build/serve options
- static site generation
- server side rendering
- incremental static regeneration

/NextJS - routing




NEXT-BASIC
pages
api
cars
[id].js
index.js
_app.js
index.js
public
styles
.gitignore
package-lock.json
package.json
example.com/cars/:id
example.com/cars
example.com/

/NextJS - rendering?

- **S**tatic **S**ite **G**eneration
- **S**erver-**s**ide **R**endering
  - Regeneration on each retrieval, per page
  - Fast retrieval of data
- **I**ncremental **S**tatic **R**egeneration
  - In-between SSG and SSR
  - Fetch data based on cache strategy

/NextJS - css

- Css Modules
- CSS = Scoped
- No need for fancy naming conventions
  - `import style from '../styles/my.module.css'`

/NextJS - “backend”

- api/<route>
  - req.method = “POST” | “GET” | “...”

<Zandvoort 2023>

<a  >

/F1 Race Zandvoort Replay

Required:

- Start Race
- Stop Race
- Render the drivers:
  - In the correct order on track
  - Pit stop indicator
  - Gap to leader

Optional:

- Pause / Resume Race
- Show speed with colours when car brakes or accelerates
- Visualize the gap to leader by Showing the distance relative to each other

/F1 Race Zandvoort Replay

- clone this repo
  `https://github.com/alber70g/harvest-workshop-web-frontend`
- `cd virtual-race`
- `npm install`
- `npm run dev`

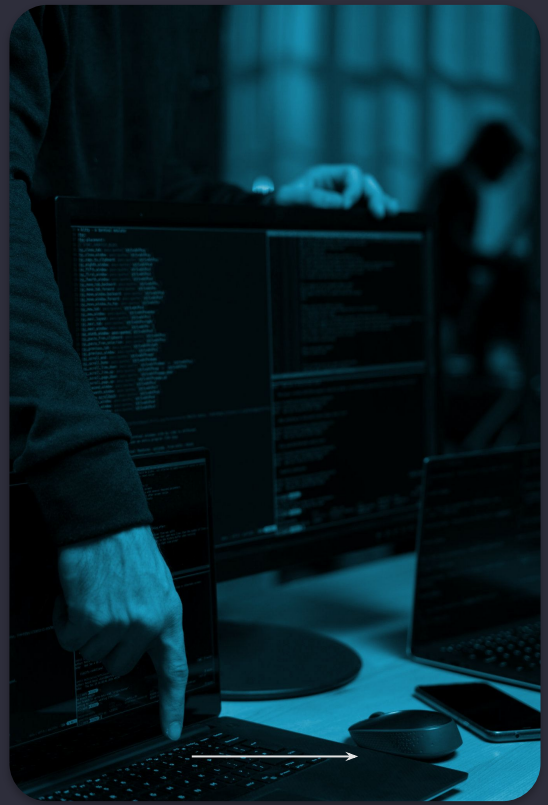
/F1 Race Zandvoort Replay

- Render the drivers in a list. Create and use a `DriverComponent`
- Create a link to a details page, `details/[driver-number]`. On that page show details of the car. Throttle, KM/h, Driver Details etc. Use progress-bar as visualiser
- Create an interval that will retrieve data every 500ms and order the drivers by position
- Create the stop, pause and resume buttons
- Visualise the distance of the drivers relative to each other
- Retrieve and plot the location data with x,y coordinates on canvas. Extend the `api/now.ts` with location data

/THANK YOU

questions?

albertgroothedde@gmail.com
steven@straatemans.nl



Feedback formulier software

Als Harvest willen wij ons continu verbeteren. Hiervoor is jullie feedback van cruciaal belang, deze kun je geven door de QR te scannen of naar de volgende link te gaan:

https://tinyurl.com/HarvestMCFeedback