

# **WEB-Entwicklung**

**WS 2023/2024**

**Jonas Heuer**

# Aufbau der Vorlesung

1. Grundlagen
2. HTML
3. CSS
4. JavaScript
5. React
6. Prüfungsprojekt

# GITHUB

<https://github.com/>

- Erstellt euch bitte einen Account

[https://docs.github.com/en/get-started/  
quickstart/set-up-git](https://docs.github.com/en/get-started/quickstart/set-up-git)

- Führt das Setup durch

# Grundlagen

Client

Server

HTTP/s

APIs

**Page Builder**

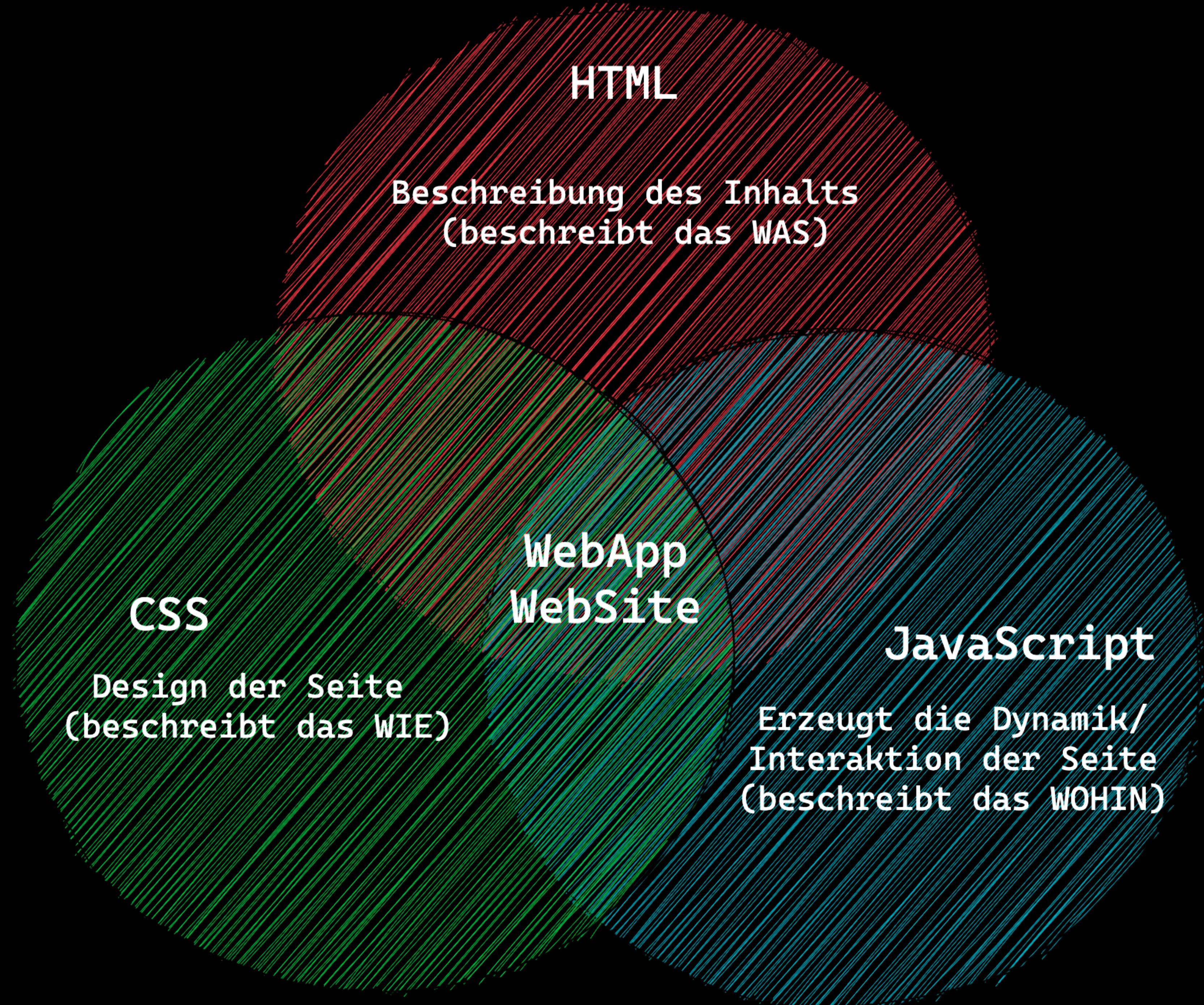
**SPA / PWA**

**Cloud Solutions**

**CMS**

**CDN**

# Vergleich HTML, CSS & JS



**HTML**

Beschreibung des Inhalts  
(beschreibt das WAS)

**CSS**

Design der Seite  
(beschreibt das WIE)

**WebApp**  
**WebSite**

**JavaScript**

Erzeugt die Dynamik/  
Interaktion der Seite  
(beschreibt das WOHIN)

# HTML

```
<!DOCTYPE html>
```

```
<html lang=„en“>
```

```
<head>
```

Der Head enthält die maschineninterpretierbaren Informationen, die sogenannten Metadaten.

Hierzu gehören:

- Der Titel der Seite
- Skripte
- Stylesheets

```
</head>
```

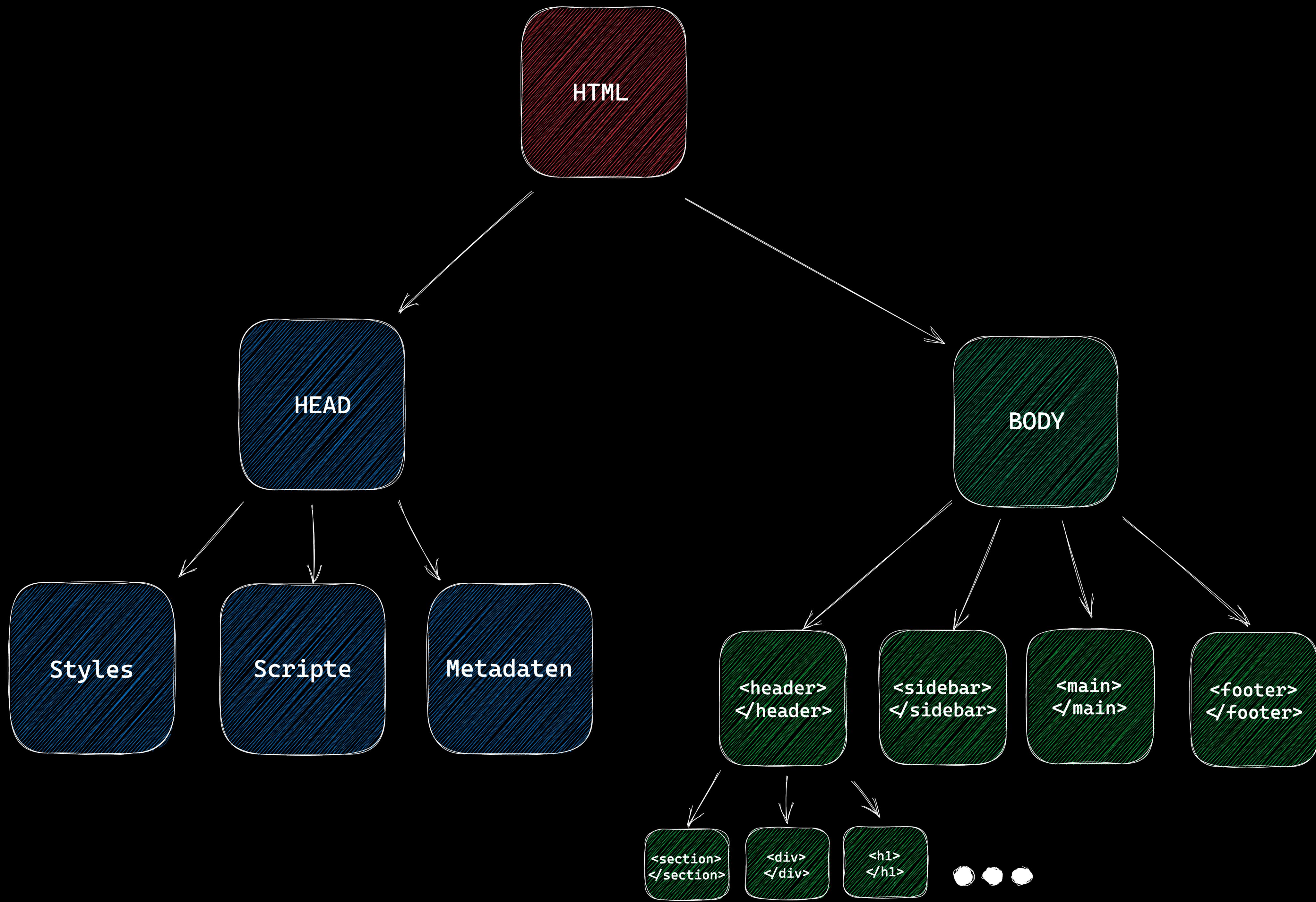
```
<body>
```

Der Body enthält die Informationen, die dem Nutzer zugänglich gemacht werden.

Hier werden die HTML-Tags zur Strukturierung der Seite eingefügt.

```
</body>
```

```
</html>
```



# Overview of all Elements

<https://developer.mozilla.org/en-US/docs/Web/HTML/Reference>

# Analyse einer Beispieleseite

- Hierarchisch strukturiert
- Verschiedene, sich wiederholende Elemente
- Gruppierung logischer Einheiten

The largest community of photo enthusiasts

Join Today

Snap photos and share like never before

**Sed ut perspiciatis**

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est.

[Learn more](#)

**Nemo enim ipsam**

Consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam.

[Learn more](#)

**Tempor incididunt**

Eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora.

[Learn more](#)

**Sed ut perspiciatis unde omnis**

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora.

[Learn more](#)

# Programmierzeit (ca. 30 Minuten)

# css

```
TagName{  
    //Allgemeine Styles für den Tag  
}  
.ClassName{  
    //Styles für Klassen  
    (wiederverwendbar)  
}  
#id{  
    //Styles für nur ein Element  
}
```

```
(cssSelector):pseudoClass{  
    //Style ändert sich, wenn pseudoClass  
    hinzugefügt wird, z.B. Hover  
}  
  
(cssSelector)::pseudoSelector{  
    //Inhalt dem Element hinzufügen  
}
```

# Keyframes (Animation durch CSS)

```
@keyframes NAME{  
    //beliebige Granularität möglich  
    X%{  
        Background  
        Border  
        Font  
        Font-size  
    }  
    Y%{  
        //Wert  
    }  
    Z%{  
        //Wert  
    }  
}  
  
selector{  
    Animation-name: //Name;  
    Animation-duration: //Zeit;  
    Animation-iteration-count: //Anzahl;  
    //weiter Werte  
}
```

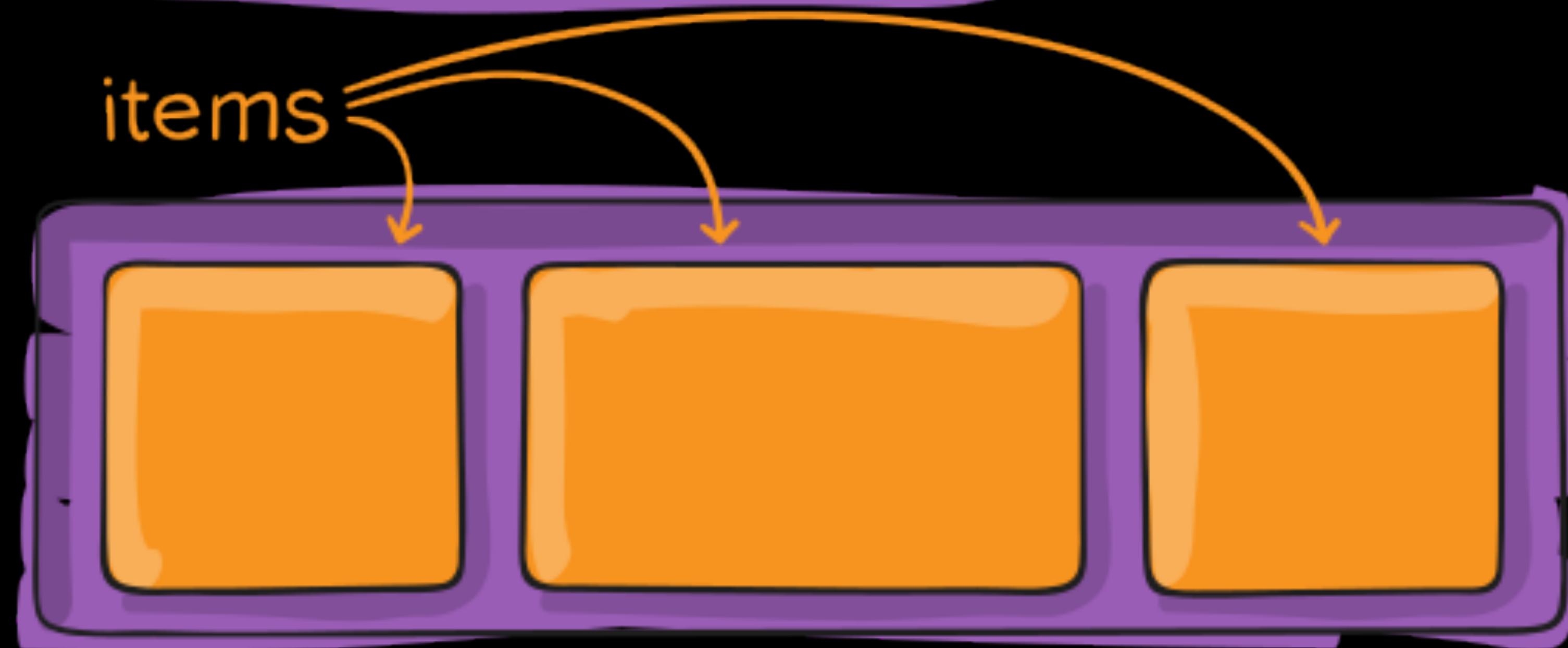
# Flexbox

<https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

container

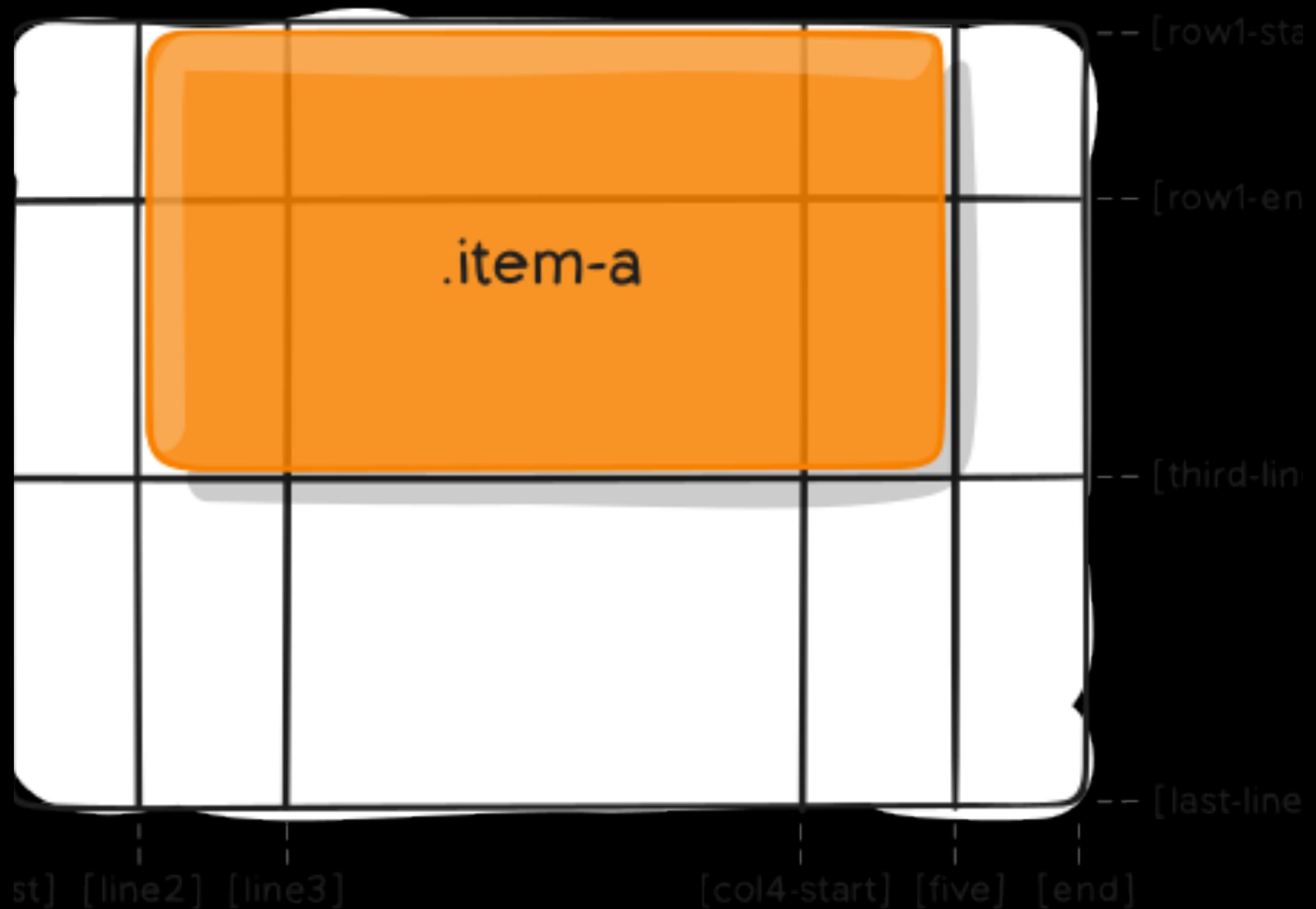


items



# CSS Grid

<https://css-tricks.com/snippets/css/complete-guide-grid/>



# Analyse einer Beispieleseite

- Wiederholende Elemente in Klassen
- Flexbox/Grid verwendet

The largest community of photo enthusiasts

Join Today

Snap photos and share like never before

**Sed ut persiciatis**

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est.

[Learn more](#)

**>Lorem ipsum dolor**

Amet, consectetur adipisciing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi.

[Learn more](#)

**Nemo enim ipsam**

Consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam.

[Learn more](#)

**Tempor incididunt**

Eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora.

[Learn more](#)

**Sed ut persiciatis unde omnis**

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem

# Programmierzeit (ca. 30 Minuten)

# JavaScript

# JavaScript

- Skriptsprache
- Objektorientiert, aber Klassenfrei
- Prozeduraler Programmablauf
- Funktionale Programmierung möglich
- Wird von uns zur DOM-Manipulation verwendet (Dynamisierung des Inhalts)

# Datentypen

Declaration: let/var/ = X/...args;

- Typdefinition ist in JS nicht notwendig
- Exklusiver Vergleich === vs. Unexklusiver Vergleich ==
- Let: Blockgebundene Deklaration
- Const: Blockgebundene Konstante
- Var: globale Deklaration (veraltet, nicht mehr genutzt)
- ...args = Spreadoperator

# Schleifen/If-else

for-Schleife:

```
for(let l=0; l<10; l++){  
    //do something  
}
```

While-schleife:

```
while(10 === 10){  
    //do Something  
}
```

If-else:

```
if(BEDINGUNG1 === true){  
    //do something  
}  
Else if(BEDINGUNG2 === true){  
    //do something  
}  
Else{  
    //do something  
}  
BEDINGUNG1 ? //do_when_true : //do_when_false ;
```

# Object/Klassen-Declaration

```
Let/Const = {  
    Vorname: „Max“,  
    Name: „Mustermann“  
}  
  
Class Person {  
    constructor (name, Vorname){  
        This.name = name;  
        This.vorname = Vorname;  
    }  
}  
  
Let Max = new Person(„Mustermann“, „Max“)
```

# Function declaration

Var x = function (x,y,...){

    Return x\*y;

}

Const x = (x,y,...) => x\*y / {return x\*y}

# DOM-Manipulation

# Programmierzeit (ca. 30 Minuten)

# React v.18

# React auf dem PC einrichten

- Doku und alle Informationen <https://react.dev/>
- Node.js installieren <https://nodejs.org/en>
- VisualStudio-Code installieren (ES7 React/Redux/GraphQL/React-Native snippets als extension empfohlen => <https://www.digitalocean.com/community/tutorials/the-best-react-extension-for-vs-code> )
- <https://vitejs.dev/guide/>
  - npm create vite@latest

# Quick-Facts

## React

- React denkt in Form von Components
- Components definieren Strukturen deiner Webseite
- Die Webseite setzt sich aus beliebig vielen Components zusammen
- Jede Component ist in sich geschlossen und funktional
- Dom-Rerendering passiert automatisch durch props und states

The largest community of photo enthusiasts

Join Today

Snap photos and share like never before

**Sed ut perspiciatis**  
Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est.

[Learn more](#)

**Nemo enim ipsam**  
Consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam.

[Learn more](#)

**Tempor incididunt**  
Eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora.

[Learn more](#)

**Sed ut perspiciatis unde omnis**  
Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem.

# **Class-Based**

Veralteter Ansatz, historisch  
viel verwendet.

Seid es Hooks gibt, abgelöst  
durch Functional-Based  
Components

# **Functional-Based**

Seid Hooks der Standard.

# How to Style Components

Import „./Component.css“

Const style = {

    backgroundColor: „white“,

    padding: „8px“

};

<Component className=„class“ />

<button style={style}>Button</button>

```
//CSS-File  
.Class{  
}  
}
```

The largest community of photo enthusiasts

Join Today

Snap photos and share like  
never before

#### Sed ut perspiciatis

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit,  
sed quia consequuntur magni dolores eos qui ratione voluptatem sequi  
nesciunt. Neque porro quisquam est.

[Learn more](#)

#### Lorem ipsum dolor

Amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt  
ut labore magna aliqua. Ut enim ad minim veniam, quis nostrud ex-  
ecution laboris nisi.

[Learn more](#)

#### Tempor incididunt

Eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est  
dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius  
modi tempora.

[Learn more](#)

#### Nemo enim ipsam

Consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt.  
Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet,  
consectetur, adipisci velit, sed quia non numquam.

[Learn more](#)

#### Sed ut perspiciatis unde omnis

Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit  
sed quia consequuntur magni dolores eos qui ratione voluptatem se-  
qui nesciunt. Neque porro quisquam est, qui dolorem.



# Props

Component:

```
<div>{props.children}</div>
```

```
<p>Hier steht {props.prop1} und hier {props.prop2}</p>
```

Aufruf:

```
<Component Prop1=„asdf“ Prop2 = „fasdfasdf“>Dieser Inhalt landet in  
den Children</Component>
```

# Dynamik Components

Component:

```
<button onClick={}>{props.children}</button>
```

```
<input onChange={}>Hier steht {props.prop1} und hier {props.prop2}</input>
```

Aufruf:

```
<Component onClick={} onInputChange{}>Dieser Inhalt landet in den  
Children</Component>
```

# Two-Way-Binding

```
Const var = 1;
```

```
Change = (event) => {  
  var = event.target.value;  
}  
  
```

```
<Person  
  change = change  
/>  
Person:  
<input  
  type=„text“  
  onChange={props.change}  
  Value={props.name}  
/>  
<div>{props.name}</div>
```

# Conditional Rendering

{state === true?  
<div>...</div> : null} =>  
in der Return Function

//nur ein Element/eine  
Component möglich

Let comp = null  
if(state.value){  
 Comp = (  
 <div>...</div>  
 );  
};  
Return:  
{comp}

# List rendering

```
Let renderComps = null;  
if(state.comps !== null){  
  renderComps = (  
    <div>  
      state.comps.map((comp, index) => {  
        Return <Comp prop1={comp.prop1}  
               key={index}/>  
      })  
    </div>  
  )  
}
```

# All about Hooks

```
Const [var1, setVar1] =  
useState(x);
```

```
Const [var2, setVar2] =  
useState(y);
```

```
return(<div>{var1}</div>)
```

```
setVar1(z);
```

```
useEffect(() => {  
    // do what you want to do, after  
    Component did mount  
});  
  
useEffect(() => {  
    //update when component Mounts  
    Return ()=>{  
        // update when component unmount  
    }  
},[//value to listen to]);
```

# Routing

index.js:

```
Import {BrowserRouter} from  
'react-router-dom';
```

```
<BrowserRouter>  
  <App/>  
</BrowserRouter>
```

To use the URL Params there is the useParams Hook let {paramName, paramName2} = useParams();

App.js

```
Import {Route, Routes, Link} from 'react-router-dom';  
<Link to='/'>test</Link>  
<Link to='/test2'>test2</Link>  
<Link to='/test3'>test3</Link>  
<Routes>  
  <Route path='/' exact element={<TestComp />} />  
  <Route path='/test2' element={<Test2Comp />} />  
    <Route index element={<Comp />} /> Requires <Outlet/> in Test2Comp  
    <Route path='/:id' element={<Comp />} />  
  <Route path='*' element={<Comp />} />  
</Routes>
```

# Portfolioprojekt DHBW

# Projekt „QUIZ“

- Es wird eine QUIZ-App bestehend aus mindestens 3 Seiten erstellt
  - Startseite
    - Der Nutzer kann seinen Namen eingeben
    - Es gibt eine Navigationsmöglichkeit zur Quizseite
    - Der Nutzer kann aus 3 frei wählbaren Kategorien für das Quiz wählen
      - Die Fragen der gewählten Kategorien werden zufällig für das Quiz verwendet
  - Quizseite
    - Der Nutzer bekommt ein Quiz mit mindestens 5 Fragen mit je 4 Antwortmöglichkeiten (wie gut hier die Qualität der Fragen und Antwort ist, ist mir persönlich egal)
    - Korrekt beantwortete und falsch beantwortete Fragen werden entsprechend markiert
    - Es gibt eine Navigationsmöglichkeit zur Ergebnisseite
  - Ergebnisseite
    - Auf der Ergebnisseite wird das Ergebnis in der Form „NAME DES NUTZERS, du hast x/y Fragen richtig beantwortet“ angezeigt
    - Es gibt eine Navigationsmöglichkeit zurück zur Startseite

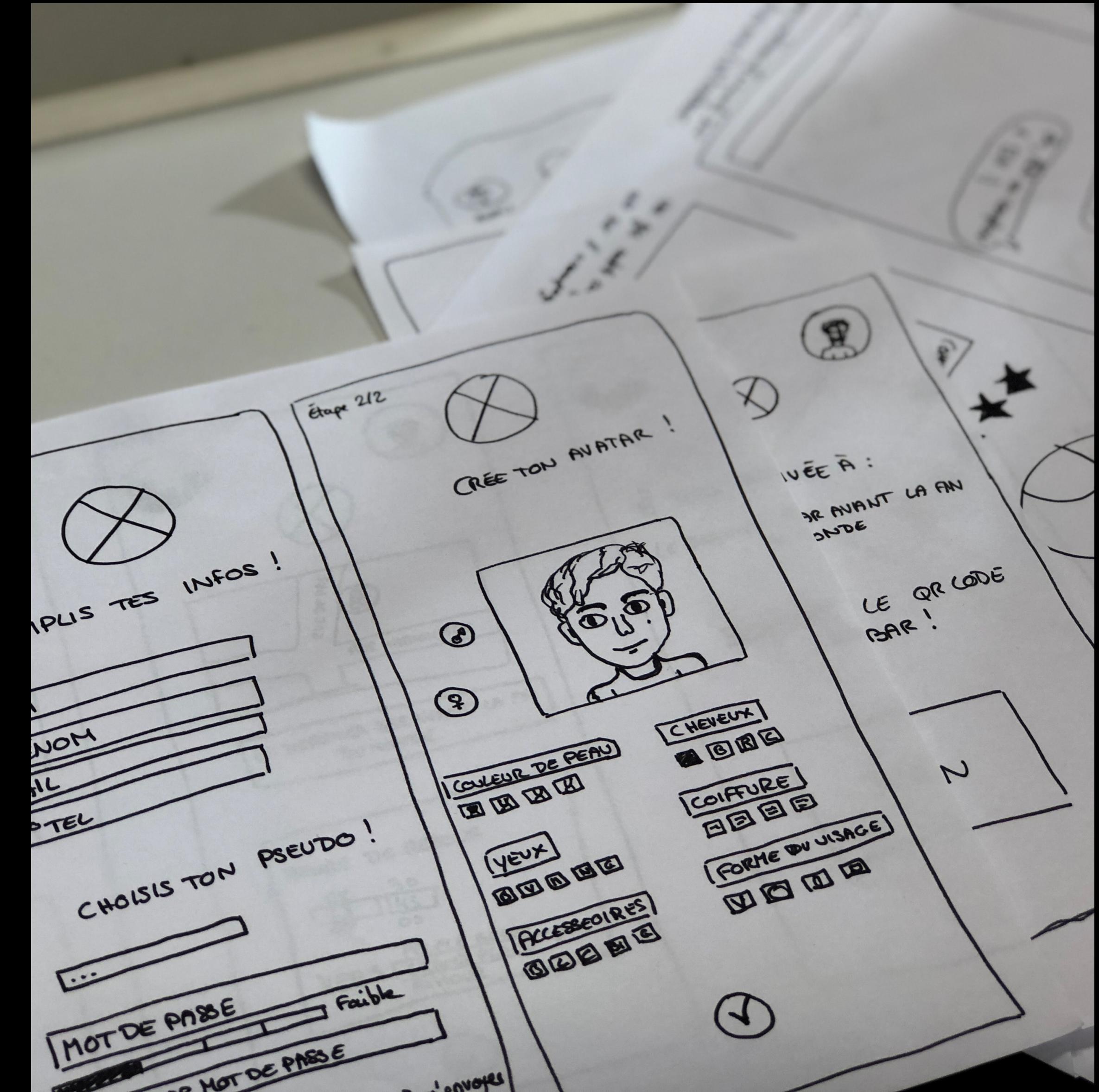
**40 %**

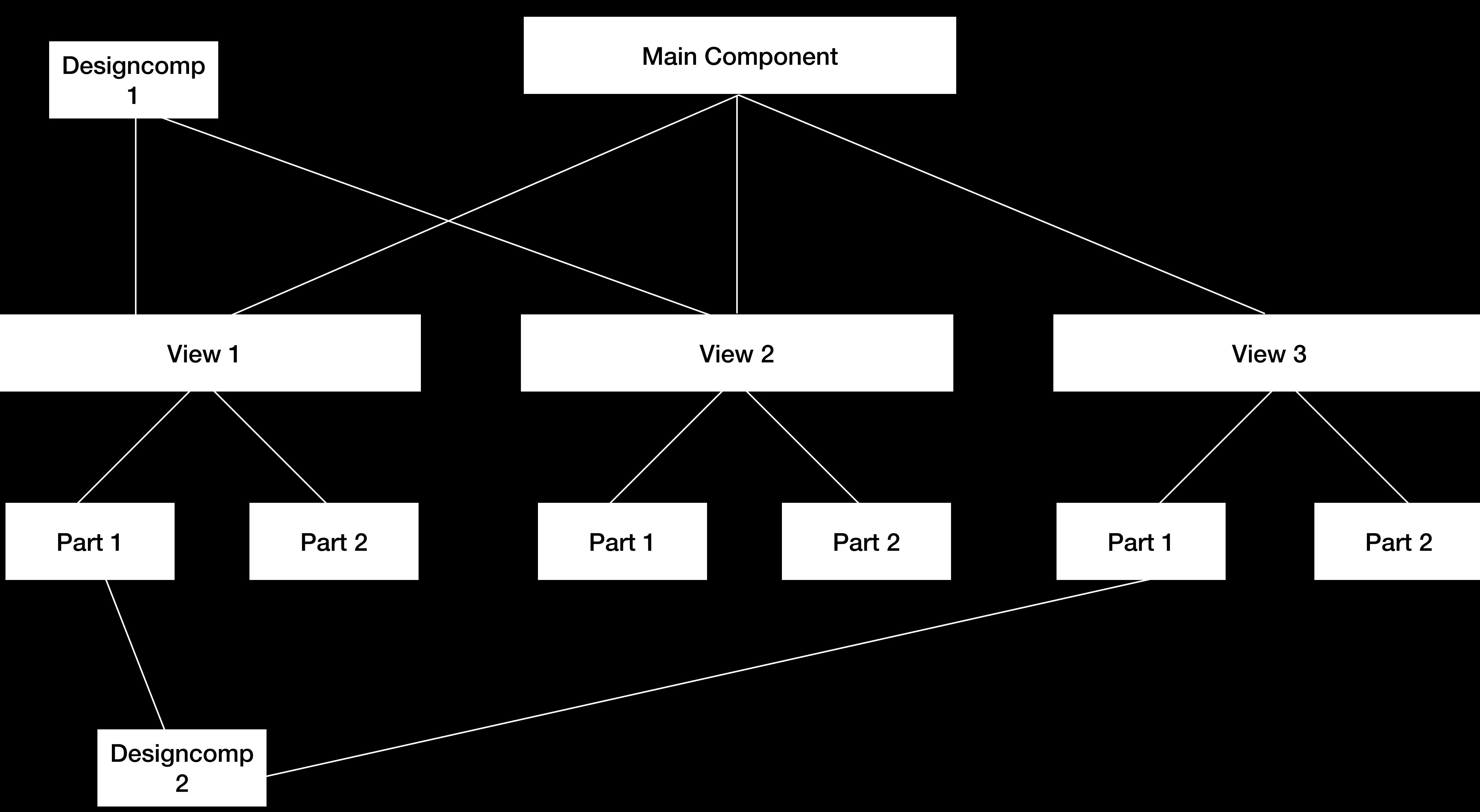
|                 |    |
|-----------------|----|
| Paper Prototype | 20 |
| Klassendiagramm | 20 |
| Requirements    | 20 |
| Konzept         | 20 |
| Resümee         | 20 |

**60 %**

|   |    |
|---|----|
| Struktur  | 10 |
| Min einmal FlexBox und min 1 mal Grid<br>in min 1 externen CSS-File und min 1<br>mal einer Style Variable | 10 |
| Router  | 10 |
| Conditional & List -<br>Rendering   | 10 |
| use State & use Params  | 10 |
| Ausschließlich functional<br>Based Components   | 10 |
| Props & Two Way Binding   | 10 |
| Naming  | 10 |
| Codequalität  | 10 |
| Funktion  | 10 |

# Paper Prototyping





# Formale Richtlinien

Abgabe des Projektes über Github **und** Moodle (Dokumentation (PDF) und Code)

Matrikelnummer steht auf der Dokumentation (PDF) und im Code

Dokumentation wird als PDF abgegeben/Name des PDFs ist die Matrikelnummer (GitHub **und** moodle)

Projekt in Github: Commithistorie muss vorhanden sein (mindestens 10 commits, die den Fortschritt zeigen)

Jeder gibt sein eigenes Projekt (Dokumentation und Code) ab, welches er zu 100% selbst erstellt

ABGABE: 28.02.2024 23:59 Uhr

Github: ws23@jonas-heuer.com als Contributor hinzufügen (Nutzername: WebDevWS23)