


Feinkonzept Taschenrechner

- Erstellung UX-Design durch Designer (Annahme: ist bereits erstellt)
- Aufsetzen der Entwicklungsumgebung (bspw. React) 30min
- Umsetzung Design im Frontend (HTML)
 - Rahmen des Taschenrechners 15min
 - Zahlen-Anzeige 30min
 - Tastenfeld Zahlen und Operatoren 30min
- Entwicklung der Logik (JavaScript)
 - Klickfunktionalität Tasten 30min
 - Rechnungslogik 115min
- Tests für die Logik des Taschenrechners (Jest)
 - Tests für Addition, Subtraktion, Multiplikation, Division 45min
 - Test für Weiterrechnen 15min

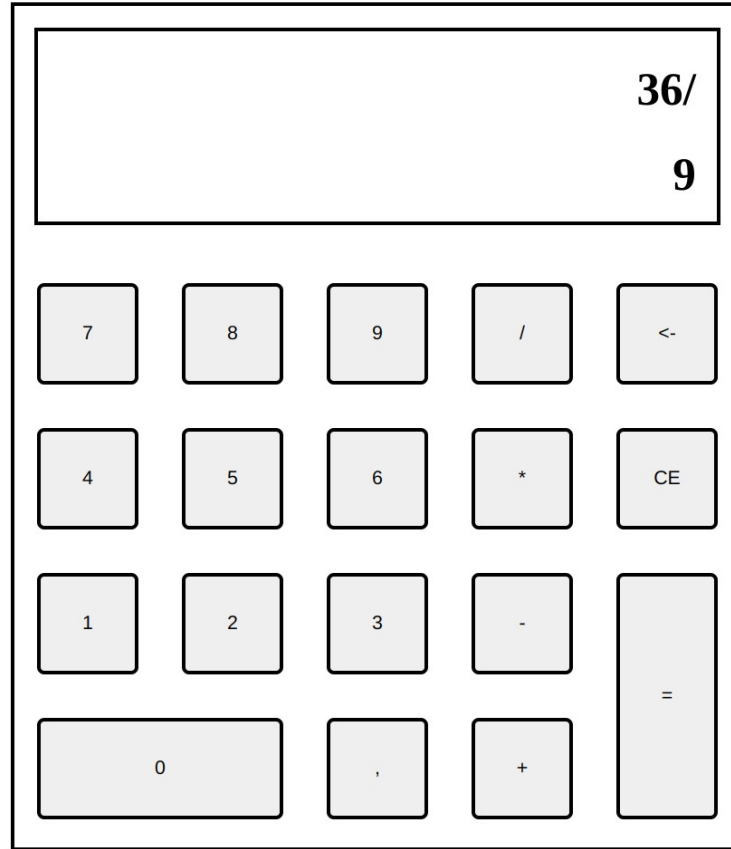
Summe

300min \triangleq 5 Stunden

src > assets > Components > button-field >  button-field.component.scss >  .button

```
1  .button-field {
2    display: grid;
3    grid-template-columns: repeat(5, 1fr);
4    grid-gap: 30px;
5    grid-auto-rows: minmax(70px, auto);
6    justify-items: center;
7    position: relative;
8    text-align: center;
9    margin-left: 3%;
10   margin-right: 3%;
11   margin-top: 40px;
12 }
13
14 .button {
15   border: solid black 3px;
16   border-radius: 5px;
17   width: 100%;
18   height: 100%;
19 }
20
21 .zero {
22   grid-column: 1/3;
23   grid-row: 4;
24 }
25
26 .button-calculate {
27   grid-column: 5;
28   grid-row: 3/5;
29 }
```

Taschenrechner:





Erweiterung


- Erstellung UX-Design durch Designer (Annahme: ist bereits erstellt)
- Aufsetzen der Entwicklungsumgebung (bspw. React) 30min
- Umsetzung Design im Frontend (HTML)
 - Rahmen des Taschenrechners 15min
 - Zahlen-Anzeige 30min
 - Tastenfeld Zahlen und Operatoren 30min
- Entwicklung der Logik (JavaScript)
 - Klickfunktionalität Tasten 30min
 - Rechnungslogik 115min
 - **Zusätzliche Zeit für Einführung der Klammerberechnung 45min**
- Tests für die Logik des Taschenrechners (Jest)
 - Tests für Addition, Subtraktion, Multiplikation, Division 45min
 - Test für Weiterrechnen 15min


Summe


300min \triangleq 5 Stunden


 Frame Calculator
+ field: type
+ method(type): type


 Display
+ displayedValue: number
+ showNumber(number): nu


 ButtonField
+ Button: button


 <i>Button</i>
+ text type
+ (type): type

 Button Number
- Number: Number
+ getNumber(number): numl

 Button Operator
- operator: String
+ getOperator(String):String

 Button Comma
- operator: String
+ getComma(String):String

 Button Calculate
- operator: String
+ calculate(String): Number

 Button Brackets
- bracker: String
+ getOperator(String):String

Hinweis:

- ButtonField hätte nur einen Emitter für die Buttons zur Verfügung stellen und App selbst entscheiden sollen, was bei welchem Button passiert