Javascript les 8

functies

functie

=

mini-systeem binnen je code

functie

=

vaak:

input -> transformatie -> output

functie
waarom?

- één keer goed uitzoeken, daarna ‘nooit’ meer over nadenken
- code die steeds ongeveer hetzelfde is, makkelijk hergebruiken
- programma’flow’ leesbaar houden
- jouw geweldige ideeën voor anderen bruikbaar maken (en daar heel rijk mee worden)

functie
waarom?

- één keer goed uitzoeken, daarna ‘nooit’ meer over nadenken
- code die steeds ongeveer hetzelfde is, makkelijk hergebruiken
- programma’flow’ leesbaar houden
- jouw geweldige ideeën voor anderen bruikbaar maken (en daar heel rijk mee worden)

functie
waarom?

- één keer goed uitzoeken, daarna ‘nooit’ meer over nadenken
- code die steeds ongeveer hetzelfde is, makkelijk hergebruiken
- programma’flow’ leesbaar houden
- jouw geweldige ideeën voor anderen bruikbaar maken (en daar heel rijk mee worden)

functie
waarom?

- één keer goed uitzoeken, daarna ‘nooit’ meer over nadenken
- code die steeds ongeveer hetzelfde is, makkelijk hergebruiken
- programma’flow’ leesbaar houden
- jouw geweldige ideeën voor anderen bruikbaar maken (en daar heel rijk mee worden)

Maken

```
function naam() {
  // functie-code
}
```

Een functie zonder input

Gebruiken

```
naam();
```

Maken

```
function naam(param) {
  // functie-code
}
```

param is de input. Dit mogen meerdere dingen zijn, gescheiden door komma’s

Gebruiken

```
naam(param);
```

Maken

```
function naam(param) {
  // functie-code
}

// ‘naam’ bedenk je zelf.
// Kies de naam zo dat deze beschrijft wat de functie doet.
```

Gebruiken

```
naam(param);
```

Maken

```
function square(x) {
    let result = x * x;
    console.log(result);
}

// rekent het kwadraat uit
// van een getal
```

Gebruiken

```
square(10);

//resultaat = 100
```

Maken

```
function square(x) {
    let result = x * x;
    return result;
}

// print niks, maar geef het
// resultaat terug
```

Gebruiken

```
console.log(square(10));
```

of

```
let result = square(10);
console.log(result);
```

Maken

```
function square(x) {
    return x * x;
}

// print niks, maar geef het
// resultaat terug
```

Gebruiken

```
console.log(square(10));
```

of

```
let result = square(10);
console.log(result);
```

Maken

parameters (input)

Gebruiken

```
function note(pitch) {
  makeNote(pitch,0.5,250);
}
```

```
note(60);
note(66);
//resultaat, er wordt
//een noot afgespeeld
```

Maken

parameters
(variabelen als input)

Gebruiken

```
function note(pitch) {
  makeNote(pitch,0.5,250);
}
```

```
let notes = [60, 63, 62, 67];
let step = 2;

note(notes[step]);
//resultaat, er wordt
//een noot afgespeeld (62)
```

Maken

```
function chord(notes) {
  makeNote(notes[0],0.5,250);
  makeNote(notes[1],0.5,250);
  makeNote(notes[2],0.5,250);
}
```

```
//ook functies
```

Gebruiken

```
chord([60,64,68]);

//resultaat: er worden
//drie noten
//afgespeeld
```

transformatie

Maken

```
function chord(notes) {
  //code in de functie
  //kan alles zijn
  //wat je maar wil
  //als het maar Javascript is
}
```

Gebruiken

```
chord([60,64,68]);
```

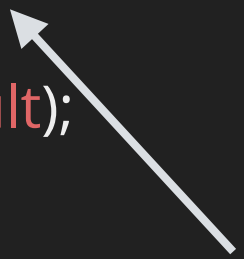
Maken

parameters
(input)

Gebruiken

```
function telop(a,b) {
  let result = a + b
  console.log(result);
}

//je kan meerdere parameters
//meegeven
```



```
telop(60,62);

//resultaat: er wordt
//122 naar de console
//gelogd.
```

Maken

```
function telop(a,b) {
  let result = a + b
  return result;
}
//return stuurt het resultaat terug
```

Gebruiken

```
let resultaat = telop(6,7);
//resultaat: de functie doet iets en geeft het resultaat terug naar de plek waar de functie is aangeroepen
```

Kijk even in de syllabus voor meer uitleg en voorbeelden