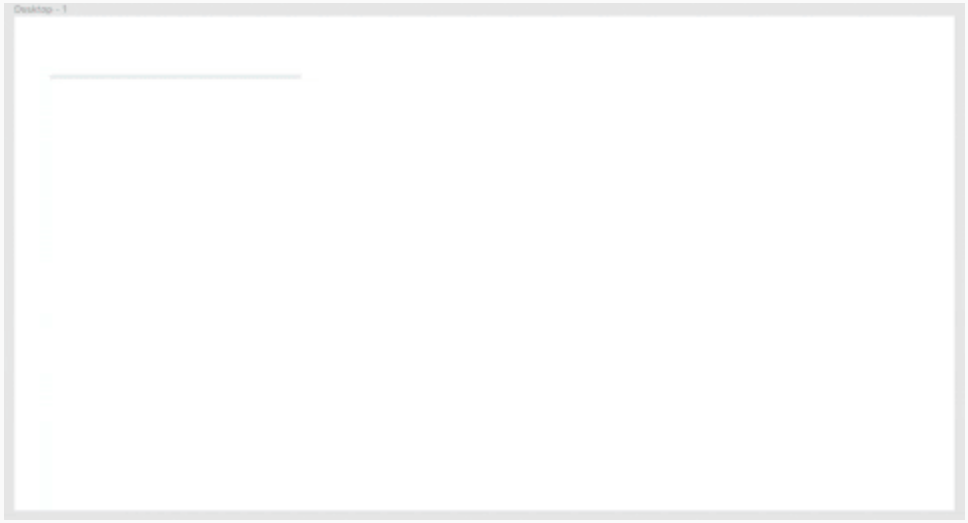
Asynchroon werken

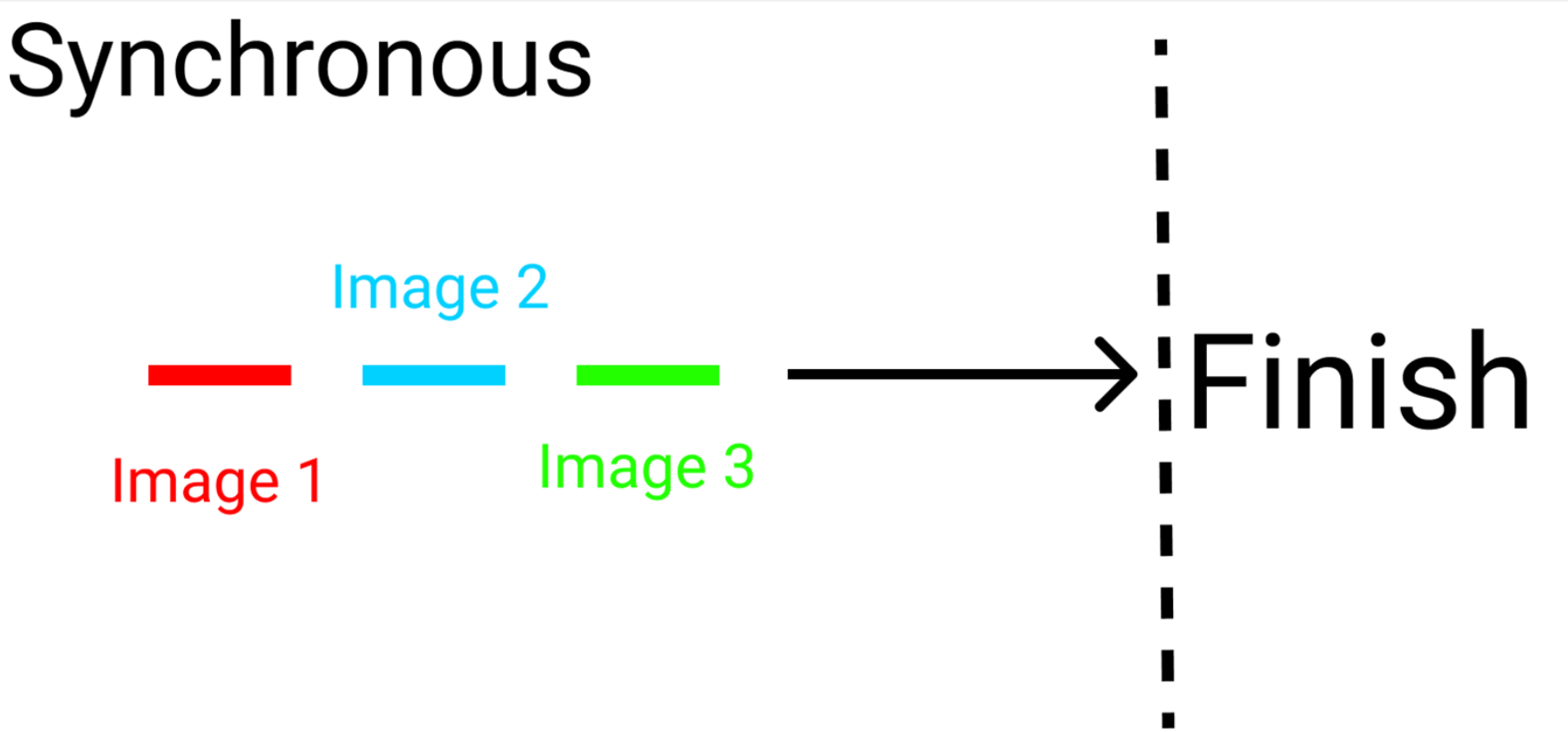
Synchroon werken

- Alle taken worden 1 voor 1 uitgevoerd
- Bijvoorbeeld: je hebt 1 hand om 10 taken uit te voeren

Synchroon werken



Synchroon werken



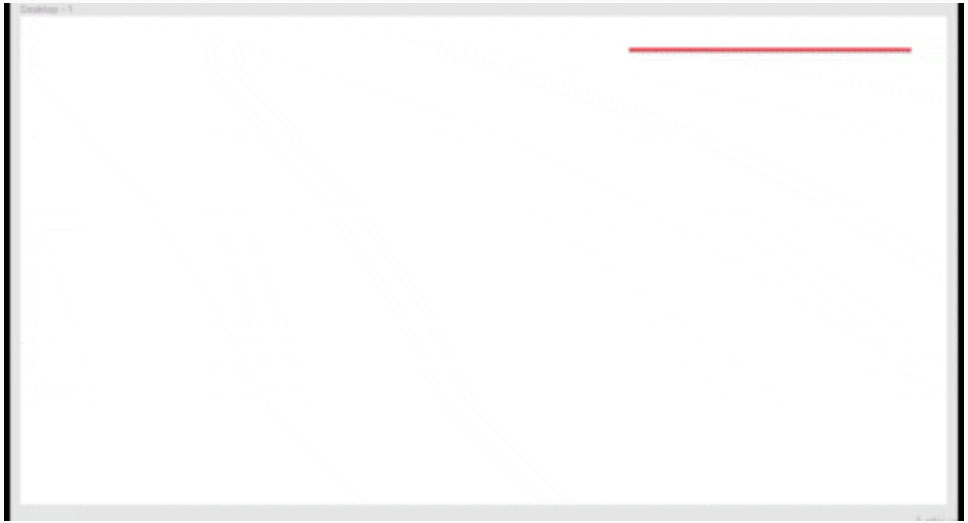
Voorbeeld

[codepen]

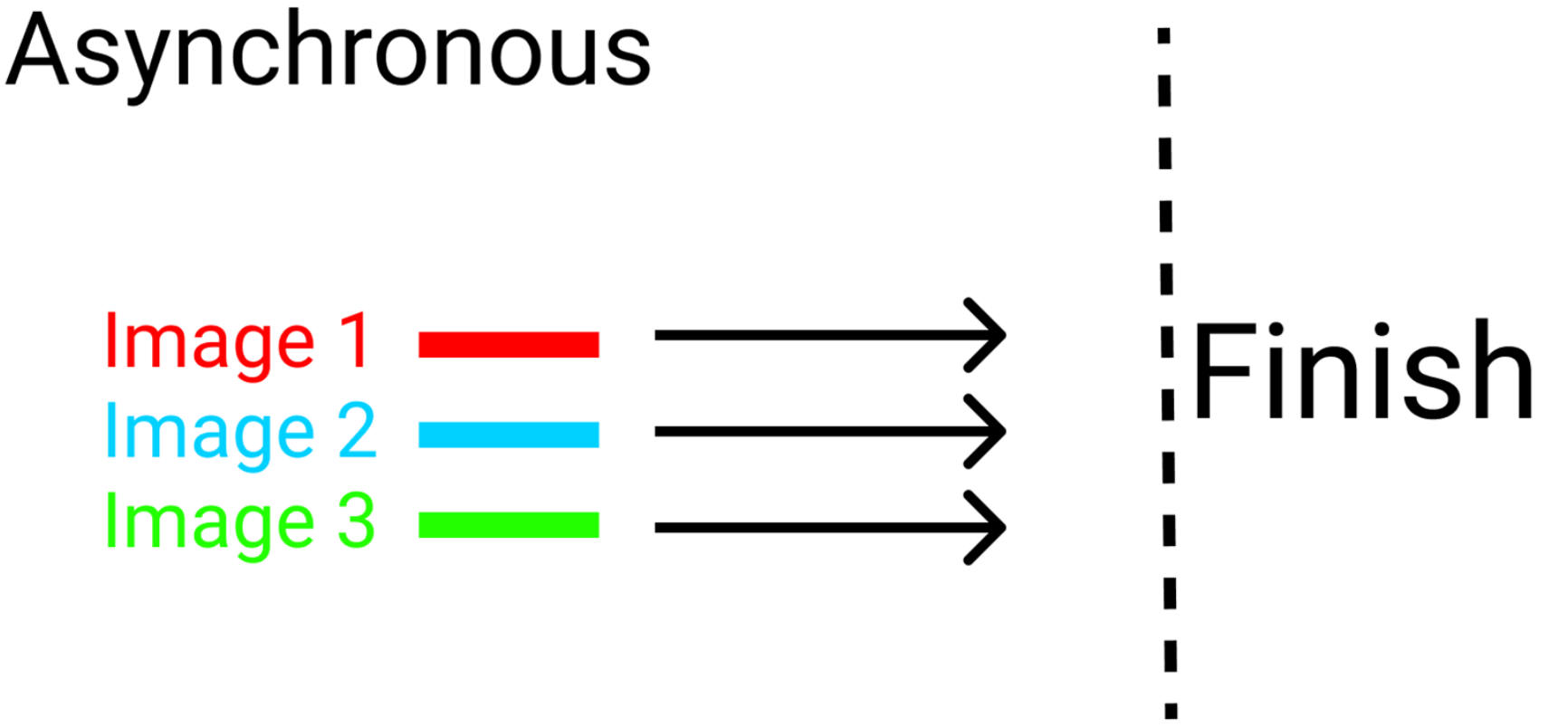
Asynchroon

- Taken gebeuren onafhankelijk en tegelijk
- bijvoorbeeld: 10 handen die 10 taken moeten uitvoeren

Asynchroon



Asynchroon



Voorbeeld

[codepen]

Manieren van asychroon werken

- Callbacks
- Promises
- async/await

Callbacks

Callbacks

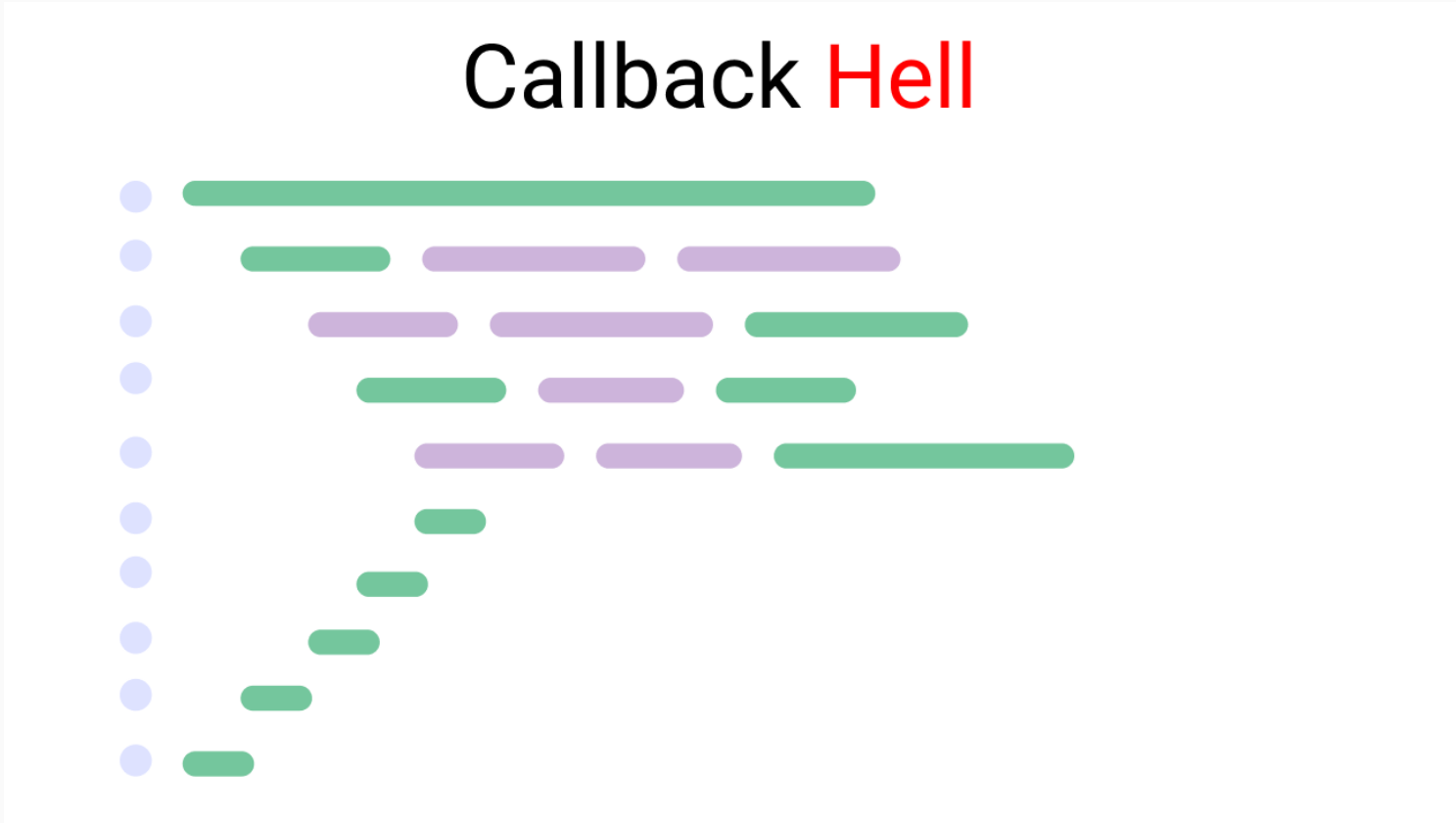
- Hogere orde functies
  - We geven een functie mee als parameter van een andere functie
  - De functie als parameter kunnen we dan uitvoeren in de aangeroepen functie.
- We geven aan de hand van een hogere orde functie mee wat er moet gebeuren wanneer de asynchrone taak klaar is.
- [codepen]

Demo pizza bestellen

1. Bestelling plaatsen => 2s
2. Deeg maken => 2s
3. Saus smeren => 1s
4. Beleggen met toppings => 5s
5. In de oven plaatsen => 10s
6. Doos kiezen => 2s
7. Bestelling leveren => 1s

[codepen]

Callback hell



Promises

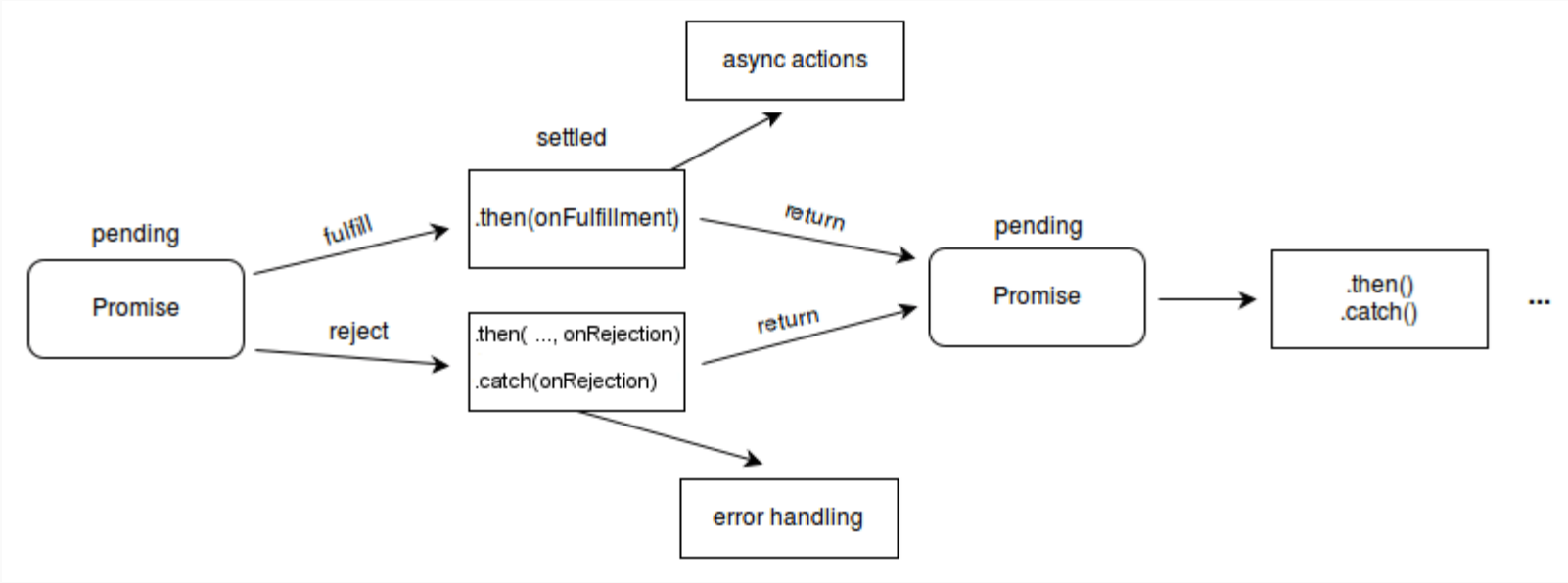
Promises

Promises Format

Promises

- Een promise is een belofte dat er ooit iets zal komen. We weten alleen niet wanneer.
- resolve & reject kunnen we gebruiken om in een promise aan te duiden of de actie gelukt is of niet
- then, catch en finally worden gebruikt om te reageren op een promise.
  - Hebben we al gebruikt met fetch
- [codepen]
- [codepen2]

Promises


async actions
settled
pending
Promise
fulfill
.then(onFulfillment)
return
pending
Promise
.then()
.catch()
...
reject
.then(..., onRejection)
.catch(onRejection)
return
error handling

Promises in real world

Welke promises worden hier allemaal gemaakt?

1. rekentaak maken
2. leerkracht roepen
3. leerkracht verbetert

Promises in real world

```
const makeHomework = new Promise(...)
const callTeacher = new Promise(...)
const gradeAssignment = new Promise(...)

function giveScore(){
  ...
}

makeHomework.then(()=> {
  return callTeacher()
}).then(()=>{
  return gradeAssignment()
}).then(()=> {
  giveScore();
})
```

Promise.All

Met Promise.all kunnen we naar een array van promises luisteren en iets uitvoeren wanneer alle promises succesvol zijn geweest.

Async await

async await

- Met async await, kunnen we onze promises nog duidelijker/leesbaarder schrijven.
- async kan niet gebruikt worden zonder await en omgekeerd.
- Door het keyword async voor een functie te plaatsen, wordt deze functie automatisch een promise.
- Met try en catch kunnen we rejected promises opvangen.
- Het await keyword zorgt ervoor dat de promise uitgevoerd wordt en het resultaat terug geeft.

Async await voorbeelden

https://codepen.io/matthiasdruwe/pen/jOKNwQw

https://codepen.io/matthiasdruwe/pen/YzvKQRm

Bron

https://www.freecodecamp.org/news/javascript-async-await-tutorial-learn-callbacks-promises-async-await-by-making-icecream/