

Cahyo Dwi Raharjo

I am an IT Instructor, software engineer, and researcher.

Experience Background



Cahyo Dwi Raharjo

14 years experience as IT
Instructor & Software
Development

Basic Javascript

Asynchronous



Asynchronous

- ☐ Introduction
- ☐ Callback
- ☐ Promise

Asynchronous



Introduction



Callback

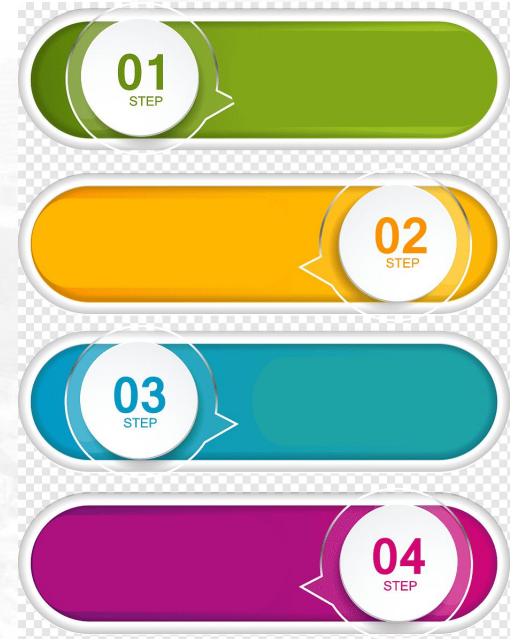


Promise

Single Thread

Selain termasuk Interpreted Language dan Dynamic-Typed Language, JavaScript juga termasuk Single Threaded Programming Language.

Yaitu JavaScript hanya bisa melakukan satu operasi di satu waktu, sehingga code JavaScript dieksekusi secara berurutan dari atas ke bawah layaknya sebuah antrian atau biasa disebut synchronous



Simple Asynchronous

setTimeout adalah function yang menggunakan concept asynchronous

```
console.log('Hi Brachio');  
  
setTimeout(function () {  
  console.log('the time has come');  
}, 3000);  
  
console.log('to learn how to code');
```

Eventloop

Untuk lebih memahami asynchronous pada JavaScript kita perlu mengetahui apa itu Event Loop.

Event loop adalah bagian dari JavaScript Runtime yang bertugas untuk menangani Event Callback, Event Callback sendiri adalah bagian dari code yang dieksekusi setelah event tertentu.

Contoh Kasus: Klik tombol Download di browser.

- mouse click adalah event
- function yang bertugas untuk mengunduh adalah callback

Asynchronous



Introduction



Callback



Promise

Callback

Callback adalah function yang menjadi argument untuk function lain dan akan dieksekusi pada poin tertentu, bisa jadi saat ini atau nanti.

```
const notify = () => {  
  console.log('Download complete!');  
};  
  
const download = (url, callback) => {  
  console.log(`Downloading from ${url}....`);  
  
  callback();  
};  
  
const url = 'example.com';  
  
download(url, notify);
```

Nested Callback

```
const download = (url, callback) => {
  console.log(`Downloading from ${url}....`);

  callback();
};
```

```
const url1 = 'example1.com';
const url2 = 'example2.com';
const url3 = 'example3.com';
```

```
download(url1, function () {
  download(url2, function () {
    download(url3, function () {
      console.log('Download complete!');
    });
  });
});
```

Asynchronous



Introduction



Callback



Promise

Promise

Promise bisa dikatakan sebagai object yang menyimpan hasil dari sebuah operasi asynchronous baik itu hasil yang diinginkan (resolved value) atau alasan kenapa operasi itu gagal (failure reason).

Sample Promise

```
let progress = 100;

const download = new Promise((resolve, reject) => {
  if (progress === 100) {
    resolve('Download complete');
  } else {
    reject('Download failed');
  }
});

download.then((result, error) => {
  console.log(result);
  console.log(error);
});
```


Sample Promise With Handler

```
let progress = 100;

const download = new Promise((resolve, reject) => {
  if (progress === 100) {
    resolve('Download complete');
  } else {
    reject('Download failed');
  }
});

download
  .then((result) => {
    console.log(result); // Download complete
  })
  .catch((error) => {
    console.log(error); // Download failed
  });
```

Sample Promise All

```
const downloadStart = new Promise((resolve, reject) => {
  resolve('0%');
});
const downloadHalf = new Promise((resolve, reject) => {
  resolve('50%');
});
const downloadFull = new Promise((resolve, reject) => {
  resolve('100%');
});

Promise.all([downloadStart, downloadHalf, downloadFull]).then((result) => {
  console.log(result); // [ '0%', '50%', '100%' ]
});
```

Solve Case Callback

```
const download = (url) => {  
  return new Promise((resolve, reject) => {  
    resolve(`Downloading from ${url}....`);  
  });  
};  
  
const url1 = 'example1.com';  
const url2 = 'example2.com';  
const url3 = 'example3.com';  
  
Promise.all([download(url1), download(url2), download(url3)]).then((result) => {  
  for (let downloadInfo of result) {  
    console.log(downloadInfo);  
  }  
  console.log('Download Complete');  
});
```

Async Await

Async/Await diperkenalkan di ES8 / ES2017 untuk handle operasi asynchronous dengan syntax yang lebih mirip dengan synchronous.

```
const getStatus = (url) => {  
  console.log(`Downloading from ${url}...`);  
  return new Promise((resolve, reject) => {  
    setTimeout(() => {  
      resolve('Download Complete');  
    }, 3000);  
  });  
};  
  
async function download(url) {  
  let status = await getStatus(url); // tunggu sampai promise selesai  
  console.log(status);  
}  
  
const url = 'example.com';  
download(url);
```

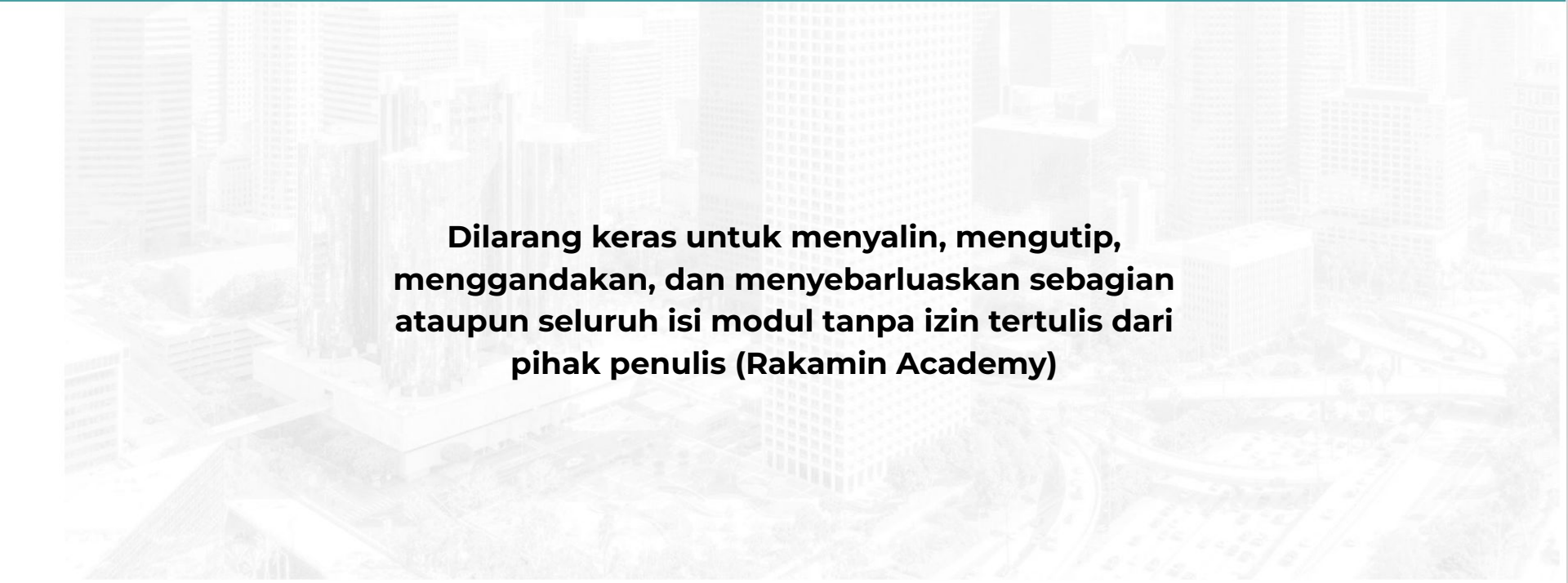
Reference material

A faded, light-colored background image of a city skyline with various skyscrapers and buildings.

<https://devsaurus.com/javascript-asynchronous>

Terima kasih!

Copyright Rakamin Academy

A faded, grayscale background image of a city skyline with various skyscrapers and buildings.

**Dilarang keras untuk menyalin, mengutip,
menggandakan, dan menyebarkan sebagian
ataupun seluruh isi modul tanpa izin tertulis dari
pihak penulis (Rakamin Academy)**