

hacktiv8 / phase-0-activities

Watch

40

Star

70

Fork

174

<> Code

🔔 Issues 0

🔗 Pull requests 0

📁 Projects 0

📊 Insights

✓

📁

Join GitHub today

GitHub is home to over 28 million developers working together to host and review code, manage projects, and build software together.

Sign up

Dismiss


<>

Branch: master

phase-0-activities / modules / js-scope.md




Find file

Copy path

 adhywiranata updated w2 challenges

2ad5813 on Oct 29, 2016

3 contributors



70 lines (52 sloc) | 1.9 KB

Raw

Blame

History

✎

🗑

# Relearn JavaScript Scope

## Objectives

- ☐ Mengulas kembali dan memahami scope lebih lanjut.

## Learnings

Cakupan atau scope pada JavaScript berhubungan erat dengan konsep global dan local. Global dan local ini dipengaruhi oleh lokasi saat kita mendeklarasikan. Variabel local bisa saja memiliki identifier yang sama dengan variabel global, tapi konteksnya berbeda. Jika kita ganti nilai suatu variabel yang namanya sama namun konteksnya berbeda, pengaruh hanya terjadi pada variabel dalam konteks yang kita perlakukan.

```
// global scope
var example = "Global";

function testExample() {
  // local scope
  var example = "Local";
  return example;
}

console.log(example); // Global
console.log(testExample()); // Local
```

Dengan kode di atas, jika kita tidak/belum mendeklarasikan variabel `example` pada scope global, kita tidak bisa melakukan `console.log(example)` karena variabel terkait dianggap tidak tercapai.

Selain variabel bahkan kita bisa mengatur scope function seperti...

```
// global scope
var functionA = function() {
  // local scope in functionA
  var functionB = function() {
    // local scope in functionB, in functionA
  };
};
```

Hukum global dan local adalah hal atau objek global dapat diakses di local, namun tidak sebaliknya.

```
var example = "Example"
var functionA = function() {
  console.log(example + " in A");
  var functionB = function() {
    console.log(example + " in B"); // it's possible
  };
  functionB();
};
functionA();
```

Scope juga berkaitan dengan `this` yang sudah kita gunakan sebelumnya.

```
var functionA = function() {
  console.log(this); // global Window object
}

var sampleObject = {};
sampleObject.functionB = function() {
  console.log(this); // Object of sampleObject
}

functionA();
sampleObject.functionB();
```

Maka dari itu, perhatikanlah dan berhati-hatilah waktu dan letak kita mendeklarasikan variabel atau function.