Nama : Ferli Andriansyah

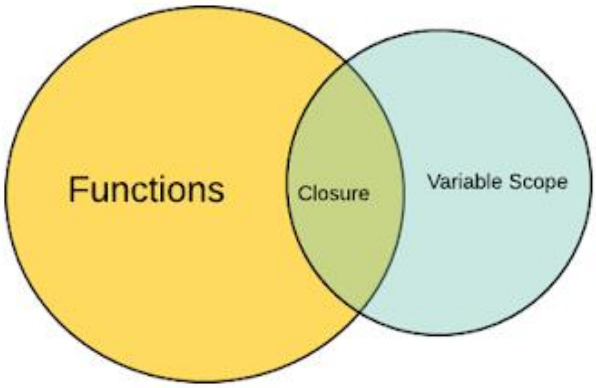
NIM : 120140018

Kelas : Pengembangan Aplikasi Mobile (RA)

Tugas Individu 2

1. Closure

Merupakan sebuah fungsi yang dijalankan atua dieksekusi oleh fungsi lainnya sehingga pada fungsi tersebut dapat mengakses sebuah variabel yang terdapat dalam sebuah lexical scope pada fungsi induknya.



Closures ini bisa mengakses ke variabel-variabel yang ada di dalam scope yang berbeda:

1. Variabel yang ada di dalam scopenya sendiri
2. Variabel yang ada di dalam scope global
3. Variabel yang ada di dalam scope function di luarnya / parent functionnya

Dan closures juga dapat mengakses ke:

1. Parameter yang dia punya
2. Parameter yang ada di dalam function di luarnya / parent functionnny

Contoh :

```
function KamenRider(){
    var berubah = "Henshin!";
     function Decade(){
         console.log(berubah + " Kamen Rider Decade");
     }
     return Decade();
}
KamenRider();
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

PS C:\Users\ASUS> node "d:\Belajar\PAM\tempCodeRunnerFile.js"
Henshin! Kamen Rider Decade
PS C:\Users\ASUS>
```

Immediately-Invoked Function Expression

Immediately-Invoked Function Expression (IIFE) adalah salah satu pendekatan dalam mengurangi penggunaan variabel global. Dengan pola ini, kita dapat memanggil suatu fungsi tanpa harus menyimpannya, dan dapat 'menyembuyikan' kode yang kompleks beserta variabel-variabel yang ada dialamnya yang berpotensi mengotori global scope.

Contoh :

```
(function(){
    console.log("Form Rider Kiva garuru!")
})();
(Henshin = function(berubah = "Orange"){
    console.log(berubah + "Lock on orange arm Hanamichi on stage !");
})();
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

PS C:\Users\ASUS> node "d:\Belajar\PAM\function KamenRider(){.js"
Form Rider Kiva garuru!
OrangeLock on orange arm Hanamichi on stage !
```

First-Class Function

Bahasa pemrograman dapat mendukung first-class function jika memberlakukan function sebagai first-class object. First-class object adalah sebuah entitas (dapat berbentuk function atau variabel) yang dapat dioperasikan dengan car ayang sama seperti entitas lain. Operasi tersebut biasanya mencakup passing entitas sebagai argument, return entitas dari sebuah function, dan assign entitias ke dalam variabel, object atau array.

Contoh :

```js
//fungsi dapat di assign ke dalama variabel
var henshin = () => {
    return "Lock On";
};
console.log(henshin());

//fungsi dapat menjadi argumen di fungsi lain
var HenshinDecade = (berubah, decade) => {
    return berubah() + " " + decade;
};
console.log(HenshinDecade(henshin, "Decade"));

//fungsi dapat direturn dari fungsi lain
var FinalFormDecade = attackrider => {
    return decade => {
        return attackrider + " " + decade;
    };
};
var FinalFormAttackRider = FinalFormDecade("Henshin");
console.log(FinalFormAttackRider("Final Attack Rider Decade"));
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
PS C:\Users\ASUS> node "d:\Tugas S5\PAM\Tugas 2\Immedietly invoked function expresion.js"
Lock On
Lock On Decade
Henshin Final Attack Rider Decade
PS C:\Users\ASUS>
```

High-order Function

High-order function adalah fungsi yang menerima fungsi lain sebagai parameternya dan/atau mengembalikan fungsi lain sebagai keluarannya. High-order function merupakan salah satu konsep terpenting pada paradigm pemrograman fungsional, fungsi ini dapat mendefinisikan apakah sebuah bahasa pemrograman bisa disebut fungsional.

Contoh :

Excecution Context

Execution context adalah suatu kedaan ketika kode javascript di run / dijalankan, environtment (lingkup) dimana kode tersebut dieksekusi sangatlah penting, interpreter akan menentukan kode tersebut termasuk pada environtment yang mana. Kode tersebut akan ditentukan menjadi salah satu dari 3 environtment :

1. Kode Global (Global Code) : default environtment, dimana kode pertama kali dieksekusi.
2. Kode Fungsi (Function Code) : ketika kode dijalankan masuk ke dalam suatu fungsi.
3. Kode Eval (Eval Code) : teks yang akan dieksekusi di dalam internal fungsi eval.

Contoh :

Excecution Stack

Javascript interpreter pada browser dibuat dengan single thread, yang berarti hanya ada satu hal pada browser yang bisa dikerjakan dalam satu waktu. Actions atau events harus menunggu untuk dieksekusi, dan dikumpulkan di tempat yang bernama Excecution Stack. Jadi execution stack adalah tempat dimana ditumpuknya suatu action atau event yang menunggu untuk dieksekusi.

Contoh :

```
function name(){
    var a = "Tsukasa";
    Say();
    console.log(a);
}

function Say(){
    var a = "Hai";
    introduce();
    console.log(a);
}

function introduce(){
    var a = "watashiwa kamen raida da";
    console.log(a);
}
name();
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

PS C:\Users\ASUS> node "d:\Tugas S5\PAM\Tugas 2\Execution Stack.js"
watashiwa kamen raida da
Hai
Tsukasa
PS C:\Users\ASUS>
```

Event Loop

Event loop adalaah proses yang hanya memiliki satu thread dengan banyak loop atau proses perulangan yang tidak terhingga / terus menerus. Pad acara ini kita hanya mengerjakan satu task dan untuk menghandle banyak request kita melakukan seleksi dan prioritas terhadap task yang dikerjakan.

Contoh:

```
function say(){
    console.log("Hai");
    setTimeout(function(){
        console.log("Gaes");
    },0);
    console.log("apa kabar?");
};
say();
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

PS C:\Users\ASUS> node "d:\Tugas S5\PAM\Tugas 2\event Loop.js"
Hai
apa kabar?
Gaes
PS C:\Users\ASUS>
```

Callbacks

Callback secara harfiah adalah menghubungi kembali. Pada javascripts sendiri callback adalah sebuah fungsi yang dieksekusi di dalam fungsi lain yang memanggilnya.

Contoh :

```
1  const henshin = function(){
2      console.log("Silver Arm Hakugin New stage!");
3  };
4  setTimeout(henshin, 1000);
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

PS D:\Tugas S5\PAM\Tugas 2> node "d:\Tugas S5\PAM\Tugas 2\Callback.js"
Silver Arm Hakugin New stage!
```

Promises dan Async/Await

Promise di dalam javascript dapat diilustrasikan seperti janji di dunia nyata. Ada 2 kemungkinan akan terjadi yaitu janji ditepati atau janji tidak ditepati.

Untuk membuat promise cukup dengan memanggil constructor nya. Constructor promise akan menerima argument sebuah fungsi dengan 2 parameter yaitu resolve dan reject.

ada 3 state yang mungkin terjadi dengan promise :

- Panding -> sedang dalam proses
- Resolved -> janji di tepati atau berhasil
- Rejected -> gagal atau janji di batalkan

Async/await adalah suatu fitur baru di javascript yang digunakan untuk menangani hasil dari sebuah promise. Caranya adalah dengan menambahakan kata ‘async’ di depan sebuah fungsi untuk mengubahnya menjadi asynchronous. Sedangkan await berfungsi menunda sebuah kode dijalankan, sampai proses asynchronous berhasil.

Contoh:

```
1  async function decade(){
2      var promise1 = new Promise((resolve, reject) => {
3          setTimeout(() =>resolve("Final Form!"), 1000)
4      });
5      let result = await promise1;
6      console.log(result);
7  }
8  decade();
9
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

PS D:\Tugas S5\PAM\Tugas 2> node "d:\Tugas S5\PAM\Tugas 2\tempCodeRunnerFile.javascript"
Final Form!
PS D:\Tugas S5\PAM\Tugas 2>
```

Berikut adalah lampiran github tugas 2 disni

Daftar Pustaka

https://devsaurus.com/javascript-function

Closures? Apaan, tuh? — Alasan Kamu Benci Javascript | by Riva Yudha | Medium

Immediately-Invoked Function Expression (IIFE) dan Motivasinya (icalrn.id)

Mengenal Lebih Dalam Tentang First Class Function Pada Javascript | by Arya Rifqi Pratama | Medium

Higher-order Function — Paradigma Fungsional Praktis, Part 4 | by Bobby Priambodo | Paradigma Fungsional | Medium

https://www.shinta.dev/2020/10/apa-itu-closure.html#:~:text=Closure%20pada%20JavaScript&text=Di%20JavaScript%20sendiri%2C%20closure%20merupakan,fungsi%20induk(parent%20function).

Apa itu Execution Context dan Execution Stack pada JavaScript ? - Forum Sekolah Koding

Concurrency vs Event Loop. Serupa Tapi Tak Sama | by D. Husni Fahri Rizal | The Legend | Medium

Pengertian callback pada Javascript – PT Proweb Indonesia