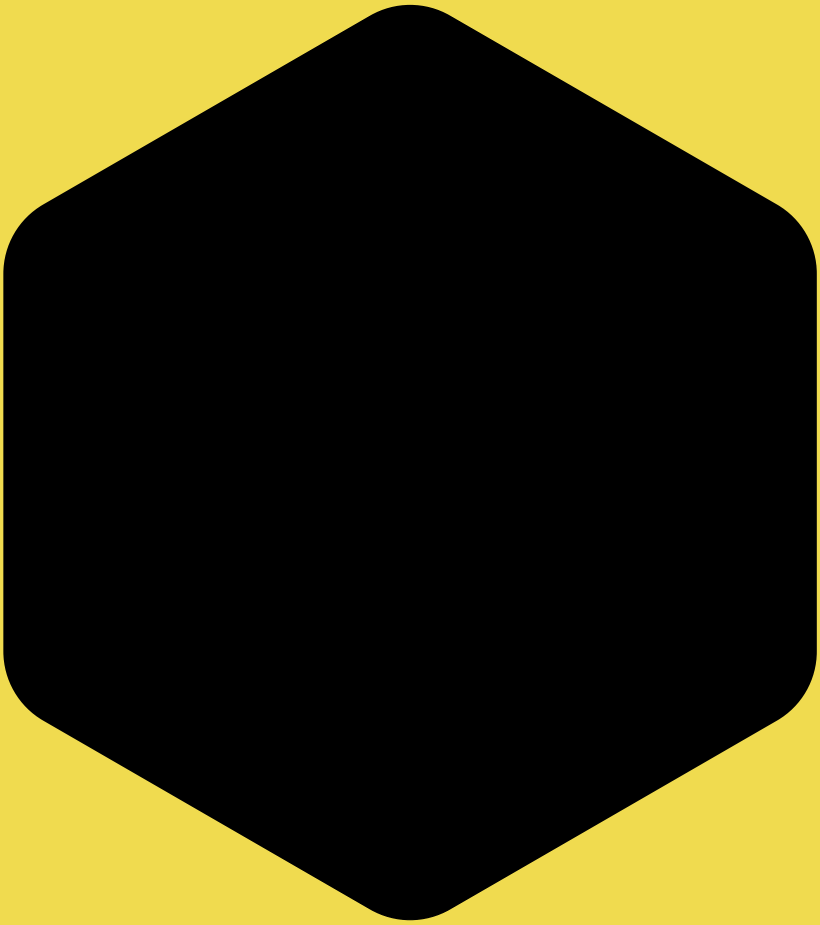

হাতেকলমে জাভাস্ক্রিপ্ট

জাভাস্ক্রিপ্টঃ ইফি, Immediately Invoked Function Expressions (IIFE)

আমাদের সাধারনত কোনো ফাংশন বানানোর পর পরে সেটাকে কল করে ইউজ করতে হয়। কিন্তু যদি আমরা ফাংশন বানানোর সাথে সাথেই সেটাকে কল করতে চাই সেক্ষেত্রে আমরা ইফি Immediately Invoked Function Expressions (IIFE) টেকনিক ইউজ করতে পারি। আজকে তাই ছোটো করে ইফি সম্পর্কেই আলোচনা করবো এবং এর কিছু রিয়েল লাইফ ইউসেজ দেখাবো।

জাভাস্ক্রিপ্ট এ আমরা ফাংশন ক্রিয়েট করতে পারি কয়েকভাবেঃ

```
function aDemoFunc() {
   console.log('Hello World!');
}
```

বা,

```
var aDemoFunc = function() {
   console.log('Hello World!');
}
```

কিন্তু যেভাবেই ফাংশন ক্রিয়েট করি না কেন, আমাদের সেটা ইউজ করতে হলে অবশ্যই ডাকতে হবে।

```
aDemoFunc();
```

এটা কন্সোলে `Hello World!` প্রিন্ট করবে।

কিন্তু আমরা যদি চাই ফাংশন ক্রিয়েট করার সাথে সাথেই সেটা কল করতে তাহলে আমরা সেক্ষেত্রে ইফি ইউজ করতে পারি।

ইফি তে সাধারণত পুরো ফাংশনটাকে প্রথম ব্র্যাকেটস এর ভিতরে রাখতে হয় এবং সবশেষে আরো দুইটা আর্গুমেন্ট ব্র্যাকেটস দিয়ে কল করতে হয়। উদাহরন দেখলে ক্লিয়ার হয়ে যাবে। ধরি উপরের ফাংশনটাই

আমি সরাসরি ক্রিয়েট করে সাথে সাথেই কল করতে চাইঃ

```
(function aDemoFunc() {
   console.log('Hello World!');
})();
```

এখানে পুরো ফাংশনটা প্রথম ব্র্যাকেটস এর ভিতরে চলে যাবে। এবং সবশেষে আরো দুইটা ব্র্যাকেটস হবে যেখানে যদি উক্ত ফাংশনের কোনো আর্গুমেন্ট থাকে তাহলে পাস করতে পারবেন।

দেখবেন এটা এক্সিকিউট করলে সাথে সাথেই কন্সোলে `Hello World!` প্রিন্ট হচ্ছে। আলাদা করে আবার ফাংশনটাকে কল করতে হচ্ছে না।

সেইমভাবে ফাংশন এক্সপ্রেশন এর ক্ষেত্রেও কাজ করবে। তবে এক্ষেত্রে ফাংশন থেকে রিটার্নকৃত ভ্যালু উক্ত ভ্যারিয়েবলটায় স্টোর হয়ে যাবে ইমিডিয়েটলি।

```
var sum = (function() {
   return 10 + 20;
})();
```

এটা এক্সিকিউট করে `sum` এর ভ্যালু চ্যাক করলে দেখবেন এটা 30 দেখাচ্ছে। আপনাকে সেইমভাবে আলাদা করে সেই ফাংশনটাকে কল করতে হচ্ছে না।

এবার গেলো তো Immediately Invoked Function Expressions বা ইফি নিয়ে আলোচনা, কিন্তু এটা আসলে কিভাবে আমাদের উপকারে লাগবে? হ্যা এবার আমি সেটা নিয়েই আলোচনা করবো।

যেহেতু এখানে পুরোটা ফাংশন প্রথম ব্র্যাকেটস এর ভিতরে থাকে, তাই এখানে পুরোটা ফাংশন স্টেটমেন্ট

না বরং এক্সপ্রেশন হিসাবে গণ্য হবে।

এটা দিয়ে আপনি প্রাইভেট ফাংশন ক্রিয়েট করতে পারবেন। এই ফাংশন বাইরে কোথাও ইউজ করতে পারবেন না। না পারবেন এর ভিতরের কিছু ইউজ করতে। সো প্রাইভেসি মেইন্টেইন করতে চাইলে বা আপনি যদি চান কোনো ফাংশনের ভিতরের ডাটা বাইরে এক্সপোজ না করতে তাইলে এটা খুব ভালো একটা ইউসেজ হতে পারে। যেমনঃ

```
(function aDemoFunc() {
   console.log('Hello World!');
})();
```

এটা যেমন সাথে সাথে কন্সোলে উক্ত লেখা প্রিন্ট করে ফেলবে, কিন্তু আপনি পরে যদি কোথাও `aDemoFunc()` কল করেন, এটা কাজ করবে না।

```
> aDemoFunc()
✖ ▶Uncaught ReferenceError: aDemoFunc is not defined
      at <anonymous>:1:1
>
```

ফাংশনের আসল কাজ হচ্ছে একটা নির্দিষ্ট সমস্যার সমাধান করা। আমরা যখন বড় প্রোজেক্ট তৈরী করি, তখন যতটুকু সম্ভব পুরো প্রোজেক্টটাকে মডুলার করার চেষ্টা করি। একেকটা কাজের জন্যে একেকটা ফাংশন তৈরী করি। ধরি একটা জাভাস্ক্রিপ্ট এর গেইম এ আপনার পয়েন্ট গণনার জন্যে একটা ফাংশন থাকবে, আরেকটা থাকবে গেইম কন্ট্রোলের জন্যে, আরেকটা থাকতে পারে ইউজার ইন্টারফেজের পরিবর্তনের জন্যে। তো এভাবে একেকটা কাজের জন্যে একেকটা ফাংশন ইউজ করলে পুরো প্রোজেক্ট এর স্ট্রাকচার যেমন সুন্দর হবে তেমনি কোনো বাগ খুজে পেতে বা ভবিষ্যৎ ডেভেলপারদের জন্যেও কোডিং স্ট্রাকচার বুঝতে সমস্যা হবে না। আরেকটা সমস্যা আমাদের যেটা খুব বেশী হয়। আমরা

অনেকগুলো ভ্যারিয়েবল নিয়ে নিতে একটা সময় দেখি পার্ফেক্ট নাম খুজে পাই না। হয়তো এই নাম আগে ইউজ করা হয়েছে সেইম প্রোগ্রামের কোনো অংশে বা অন্যরকম নাম দিলে বুঝতে সমস্যা হয়ে যেতে পারে পরবর্তিতে। তো সেক্ষেত্রে এইরকম প্রাইভেট ফাংশন ক্রিয়েট করে নিলে আপনি একই নামে ভ্যারিয়েবল পৃথক পৃথক ফাংশনে ইউজ করতে পারবেন যেহেতু এরা সম্পূর্ন প্রাইভেট। সেক্ষেত্রে আপনার জন্যে আরো সিমেন্টিক কোড লিখা সম্ভব হবে।

মডুলার প্রোগ্রামিং এ আপনি প্রাইভেট ফাংশন থেকে কিছু ভ্যালু পেতে চাইলে ফাংশন এক্সপ্রেশন ইউজ করতে পারেন এবং সেই ফাংশন থেকে ভ্যালু অবজেক্ট বা অ্যারে আকারে রিটার্ন করে দিতে পারেন। তাইলে সেটা পরবর্তিতে বাইরেও ইউজ করতে পারবেন। যেমনঃ

```
var controller = (function() {
   var a = {
           name: 'Zonayed Ahmed',
           uid: 1062
           };
   return a;
})();
```

এখানে `controller` এ একটা অবজেক্ট স্টোর হয়েছে যেটা প্রাইভেট ফাংশনের ভিতর থেকে রিটার্ন করা হয়েছে।

এভাবে আপনার কাজ শেষে যেকোনো কিছু রিটার্ন করতে পারবেন যেটা আপনি পরবর্তিতে বাইরেও ইউজ করতে পারবেন।

```
var interface = (function() {
   return 'Hello ' + controller.name;
```

```
  })();
```

এখন `interface` কল করলে দেখবেন `Hello Zonayed Ahmed` প্রিন্ট হয়েছেঃ

```
> interface
<· "Hello Zonayed Ahmed"
>
```

তো এভাবেই আপনি আপনার পুরো প্রোজেক্টটিকে মডুলার করতে পারবেন। একটা ফাংশন থেকে ডাটা আরেকটা ফাংশনে নিতে পারবেন। আরো সিমেন্টিক, গুড প্র্যাক্টিস ওয়ালা প্রোগ্রামিং করতে পারবেন।

তো আজকে এই পর্যন্তই, ভালো থাকবেন আর পাশের মানুষটিকে ভালো রাখবেন।

আপনার মন্তব্যঃ

যদি এই পোস্টে কোন ভুল(যেকোনো ধরনের) পেয়ে থাকেন অথবা কোনো ব্যাপারে সন্দেহ থাকে তাহলে এখানে জানাতে পারবেন।

☰

জাভাস্ক্রিপ্ট ব্যাসিক

জাভাস্ক্রিপ্ট অ্যাডভান্স

জাভাস্ক্রিপ্ট ইএস৬

জাভাস্ক্রিপ্ট ডম ম্যানিপুলেশন

নিত্যদিনের জাভাস্ক্রিপ্ট

জাভাস্ক্রিপ্ট অ্যালগরিদম ও ডাটা স্ট্রাকচার

জাভাস্ক্রিপ্ট সফট স্কিল

সম্পর্কেঃ

প্রোজেক্টটি সম্পূর্ণ সোর্স কোডসহ গিটহাবে রয়েছে। আপনার ভালো লেগে থাকলে স্টার দিয়ে আসবেন। আপনার পরামর্শ, মন্তব্য এবং ভুলত্রুটি গিটহাবে ইস্যু করে দেওয়ার জন্যে অনুরোধ থাকলো

আপনার জন্যঃ

রিঅ্যাক্ট জেএস শিখুন
আমি মিডিয়ামে
আমার ব্লগ

রিসোর্সঃ

এমডিএন ডকুমেন্টেশন
ইউডেমি কোর্স
Eloquent JavaScript
You Don't Know JS

❤ এর সাথে ডেভেলপ করেছে জুনায়েদ আহমেদ