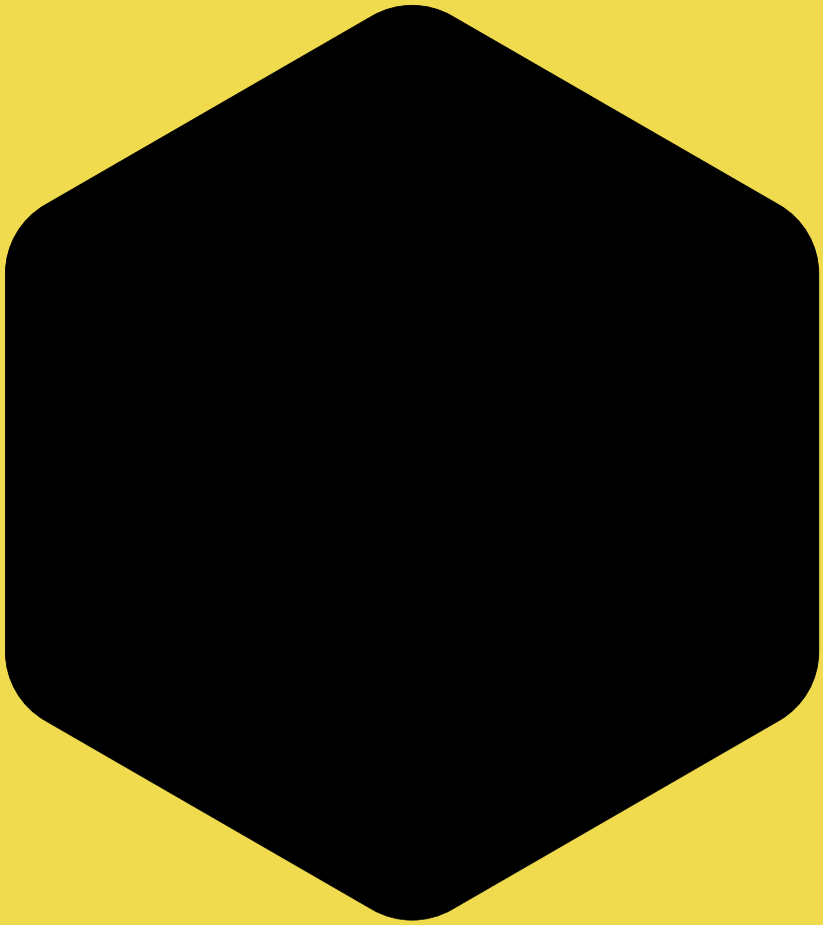
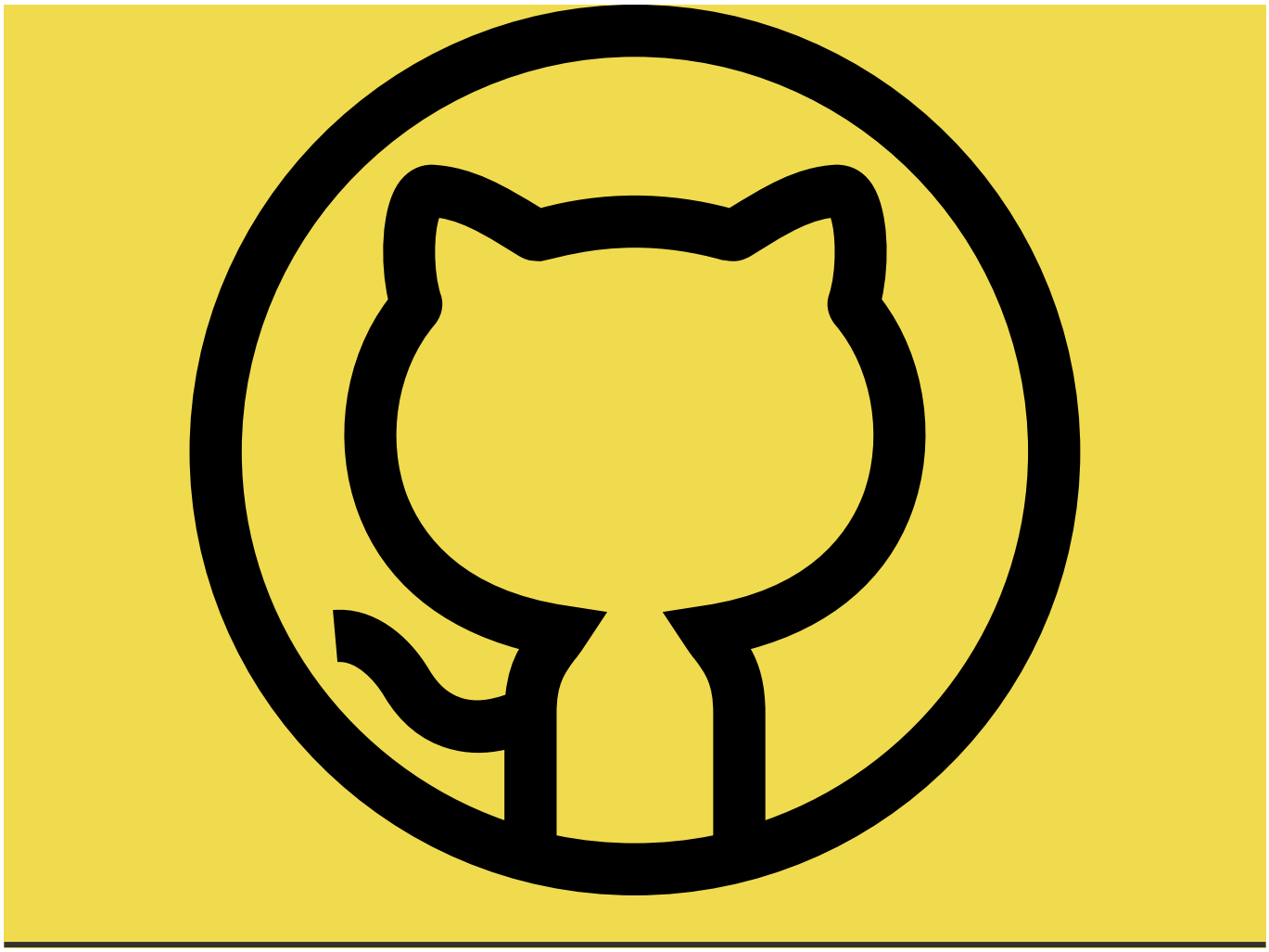
This app works best with JavaScript enabled.



হাতেকলমে জাভাস্ক্রিপ্ট



জাভাস্ক্রিপ্ট অ্যাডভান্সঃ ‘this’ কীওয়ার্ড

জাভাস্ক্রিপ্ট এ সবচেয়ে ট্রিকি এবং কনফিউজিং একটা টপিক বা ওয়ার্ড হচ্ছে `this` তবে আজকে আমি আমার এই লেখায় সেটাকে পানির মতো সোজা করে দিবো। কিন্তু তারপরেও আপনাকে প্রথম প্রথম ফিফটি-ফিফটি চান্স নিয়ে কোডে `this` ইউজ করতে হবে। কয়েকবার প্র্যাকটিস করার পর, রুলসগুলো জানার পর আস্তে আস্তে `this` কীওয়ার্ডটা পুরোপুরি ধরতে পারবেন।

`this` জাভাস্ক্রিপ্ট এ একটা রিসার্ভড কীওয়ার্ড, মানে এই নামটা আর কোনো ভ্যারিয়েবল বা ফাংশনের জন্যে ইউজ করতে পারবেন না। আর এই `this` অনেকভাবে অনেকরকম জায়গায় ব্যবহার করা হয়। তাই আপনি যে প্রথমদিকেই সবজায়গায় বুঝে ফেলবেন `this` টা আসলে কি ব্যাপারটা এরকম নয়।

কিন্তু যদি ব্যাসিক রুলসগুলো জানা থাকে `this` কিভাবে কাজ করে তাহলে আশা করি আরো বড়, জটিল কোডেও আপনার বুঝতে বেশী টাইম লাগবে না।

সাধারণত একটা ফাংশনকে কিভাবে কল করা হচ্ছে সেটার উপর ভিত্তি করে `this` এর ভ্যালু ডিটারমাইন করা হয়। আর এটার ভ্যালু ডিটারমাইন করা হয় এক্সিকিউশনের সময়। কি? কনফিউজিং লাগছে? হ্যাঁ তাহলে আজকে থেকে সব বাদ। শুধুমাত্র নিচের এই চারটা রুলস মনে রাখবেন যেগুলো দিয়ে `this` এর ভ্যালু ডিটারমাইন করা যাবেঃ

১। গ্লোবাল রুলস

২। অবজেক্ট রুলস

৩। স্পষ্ট রুলস

৪। `new` কীওয়ার্ড রুলস

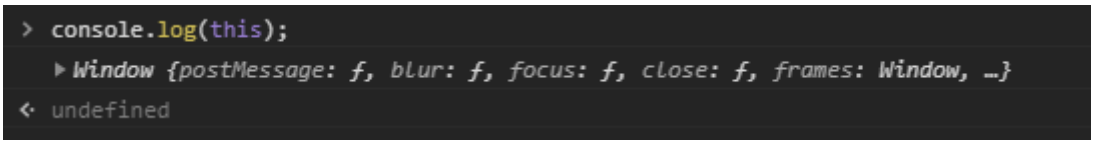
> *আমার এখানে বলা রুলসগুলো অফিশিয়ালভাবে কোথাও বলা নেই, কিন্তু `this` কীওর্য়াড বুঝতে এই রুলসগুলো বেশ কাজে লাগে তাই আমি এভাবে লিখেছি...*

১। গ্লোবাল রুলসঃ যদি আপনি `this` কীওয়ার্ড যেকোনো জায়গায় ইউজ করেন, আপনার কাস্টমভাবে ডিফাইন করা কোনো অবজেক্ট এর ভিতরে ছাড়া, তাহলে সেটা সবসময় আপনার গ্লোবাল অবজেক্টকে ইন্ডিকেট করবে। ব্রাউজারের ক্ষেত্রে গ্লোবাল অবজেক্ট হচ্ছে `window` অবজেক্ট। অন্য এনভারোমেন্ট এ যেমন নোড জেএস এ গ্লোবাল অবজেক্ট হচ্ছে `global`

ধরুন আমরা `this` কে `console.log` করলামঃ

```
console.log(this);
```

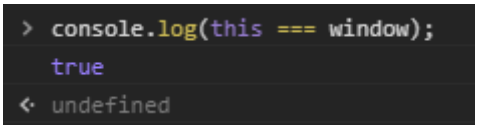
দেখবেন এটা `window` অবজেক্টই দেখাচ্ছেঃ



আরো ভালোভাবে দেখতে এটা দিয়েও দেখতে পারেনঃ

```
console.log(this === window);
```

এটা সত্য দেখাবেঃ



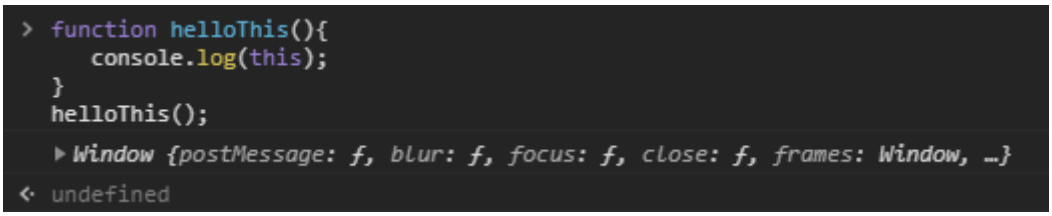
মানে সেইম জিনিসই এরা। গ্লোবাল কন্টেক্সট এ স্ট্রিক্ট(Strict Mode) মোড বা নন-স্ট্রিক্ট মোডে `this` এর ভ্যালু সবসময়ই একরকম থাকবে।

এবার সেইমভাবে যদি আমাদের কোনো ডিফাইন করা ফাংশনের ভিতরেও `this` ইউজ করি, তাহলে সেটার ভ্যালু কি হবে সেটা সম্পূর্ণ ডিপেন্ড করবে আপনি উক্ত ফাংশনটাকে কিভাবে কল করছেন সেটার উপর। এখন এখানে স্ট্রিক্ট মোড ও নন-স্ট্রিক্ট মোডে ভ্যালু ডিফারেন্ট হতে পারেঃ

যেমন এই উদাহরণে যদি দেখিঃ

```
function helloThis(){
   console.log(this);
}
```

```
helloThis();
```



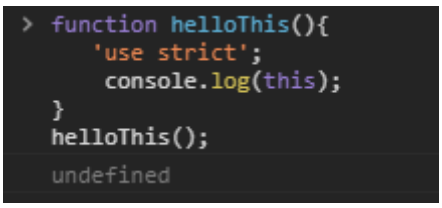
এখানে `this` গ্লোবাল অবজেক্ট(ব্রাউজারের ক্ষেত্রে `window` অবজেক্ট দেখাচ্ছে)। কারণটা আগেই বলেছি যে `this` ফাংশনের ভিতরে ইউজ করা হলেও গ্লোবাল এক্সিকিউশন কন্টেক্সট এর কারণে সেটা গ্লোবাল অবজেক্টকেই ইন্ডিকেট করবে।

সেইম জিনিসটা স্ট্রিক্ট মোডে কাজ করবে নাঃ

```
function helloThis(){
   'use strict';
    console.log(this);
}
```

```
helloThis();
```

এখানে দেখবেন আপনার রেজাল্ট `undefined` আসবেঃ



এখন এখানে এক মোডে কাজ করে আর আরেক মোডে না করার আসল কারণ হচ্ছে এখানে আমাদের `'use strict';` ইউজ করায়। এই স্ট্রিক্ট মোড আসলে এসেছে ব্যাড প্র্যাক্টিস অ্যাভয়েড করার জন্যে। জাভাস্ক্রিপ্ট এ অনেক কিছু নিয়মের বাইরে পরে যায়। আমরা এখানে ফাংশনের ভিতরে `this` চাচ্ছি। এখন স্বভাবতই এটা ফাংশন কলের সময় একটা অবজেক্ট খুঁজবে। কিন্তু পাবে গ্লোবাল অবজেক্ট (ব্রাউজারের ক্ষেত্রে `window`), যেহেতু আমরা জানি বাইরের সবকিছুই গ্লোবাল অবজেক্ট এর আন্ডারে। আর তাই যখন আমরা নন-স্ট্রিক্ট মোডে রেজাল্ট দেখি, তখন গ্লোবাল অবজেক্টই দেখি। কিন্তু স্ট্রিক্ট মোড ইউজ করায় আর সেটা হচ্ছে না। এটা স্বভাবতই ঐ ফাংশনের ভিতরে কল করায় একটা অবজেক্ট খুঁজবে। কিন্তু পাবে গ্লোবাল অবজেক্ট(গ্লোবাল এক্সিকিউশন কন্টেক্সট এর কারণে) `window.helloThis()`, যেটা হয়তো আমরা চাচ্ছি না। আমরা কোনো গ্লোবাল অবজেক্ট এর আন্ডারে `window.helloThis()` চাচ্ছি না, বরং আমরা শুধুমাত্র `helloThis()` চাচ্ছি। তাই সেটা `undefined` দেখাবে। এখন এখান থেকেই কন্সট্রাকটর ফাংশনের আইডিয়া আসছে, আর সেই সাথে `new` কীওয়ার্ডের। যেটা আমাদের চার নাম্বার রুলস এ পড়বে। তাই এ সম্পর্কে বিস্তারিত পরে আরেকটা পর্বে আলোচনা করা হবে।

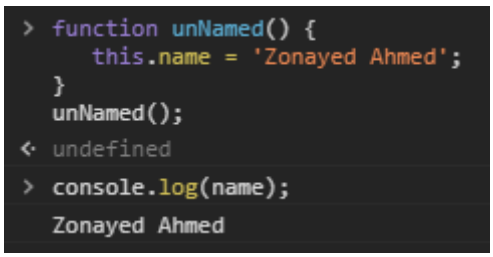
আর অনেকসময় দেখবেন এভাবে একটা ভ্যারিয়েবল নিলেও সেটা কাজ করে অন্যান্য গ্লোবাল ভ্যারিয়েবলের মতোইঃ

```
function unNamed() {
   this.name = 'Zonayed Ahmed';
}
```

```
unNamed();
```

এখন এই ফাংশনের বাইরে `name` নামে কোনো ভ্যারিয়েবল কল করলে দেখবেন সেটা কাজ করছে!

```
console.log(name);
```



কারণটা কি? হ্যাঁ কারণ হচ্ছে আপনি যখন ফাংশন এর ভিতরে `this.name` লিখলেন এবং সেই ফাংশনটা গ্লোবাল কন্টেক্সট এ কল করলেন তখন এই `this.name` আসলে গ্লোবাল অবজেক্ট এর আন্ডারে `name` এ অ্যাসাইন(`window.name`) হয়ে যাচ্ছে। তাই সিম্পলি আপনি এটা বাইরে যেকোনো জায়গায় অ্যাক্সেস পাচ্ছেন। *'strict mode' এ এটা কাজ করবে না।*

এবার কাস্টমভাবে ডিফাইন করা অবজেক্ট এর ভিতরে যদি `this` ইউজ করি তাহলে কি হবে? হ্যাঁ

তাহলে আমাদের দ্বিতীয় রুলস চলে আসবে।

২। অবজেক্ট রুলসঃ এখন কাস্টমভাবে কোনো অবজেক্ট ডিফাইন করে সেটার ভিতর `this` কীওয়ার্ড ইউজ করলে সেটার ভ্যালু আর গ্লোবাল অবজেক্টকে ইন্ডিকেট করবে না। এখানে এটার ভ্যালু চেঞ্জ হয়ে যাবে।

আমরা এভাবে কাস্টমভাবে অবজেক্ট ডিফাইন করিঃ

```
var myCustomObj = {
   name: 'Zonayed Ahmed',
   age: 21,
   job: 'Student'
}
```

এখানে `myCustomObj` হচ্ছে কাস্টমভাবে ডিফাইনকৃত অবজেক্ট। এটার ভিতরে `this` কীওয়ার্ডের ভ্যালু চেঞ্জ হয়ে যাবে। এখানে `this` কীওয়ার্ড সবসময় কাছের কাস্টমভাবে ডিফাইনকৃত অবজেক্টটাকে ইন্ডিকেট করবে। যেমন নিচের এই অবজেক্ট টাকে রান করলেঃ

```
var myCustomObj = {
   name: 'Zonayed Ahmed',
   age: 21,
   job: 'Student',
   msg: function() {
        console.log('My name is '+ this.name);
   }
}
```

এবার ভিতরের ফাংশনটাকে যদি এভাবে কল করিঃ

```
myCustomObj.msg();
```

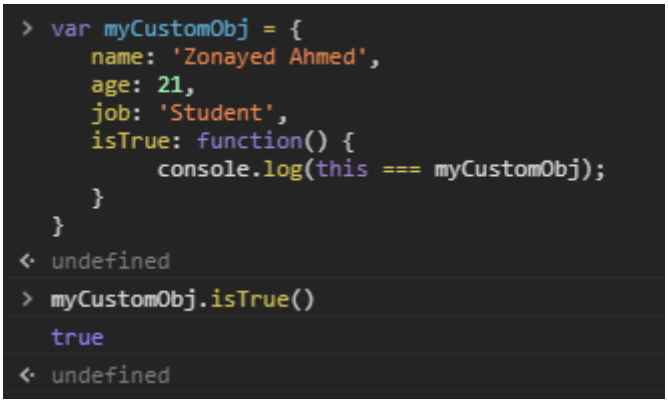
তাহলে এবার ভাবেন তো এখানে ফাংশনের ভিতরের `console.log` এ `this.name` কাকে ইন্ডিকেট করবে? প্রথম রুলস(গ্লোবাল রুলস) এ এটা পড়বে না, কারণ এটা একটা কাস্টমভাবে ডিফাইনকৃত অবজেক্ট এর ভিতরে রয়েছে। এখানে দ্বিতীয় রুলস(অবজেক্ট রুলস) অনুযায়ী এটা কাছের কাস্টমভাবে ডিফাইনকৃত অবজেক্ট কেই ইন্ডিকেট করবেঃ

সে হিসাবে `this.name` `myCustomObj` এর ভিতরের `name` কেই ইন্ডিকেট করবে। এখন আমরা তাহলে আসলে একটা কাস্টমভাবে ডিফাইনকৃত অবজেক্ট এর ভিতরে `this` কীওয়ার্ডের ভ্যালু কি হয় সেটা দেখবোঃ

```
var myCustomObj = {
   name: 'Zonayed Ahmed',
   age: 21,
   job: 'Student',
```

```
    isTrue: function() {
         console.log(this === myCustomObj);
    }
}
```

`this` আর আমাদের ডিফাইনকৃত অবজেক্ট `myCustomObj` কি সেইম জিনিসই?



হ্যাঁ এটা সত্য বলছে, তার মানে এটা আমাদের কাস্টম অবজেক্ট টাকেই ইন্ডিকেট করছে এখানে।

এখন অবজেক্ট এর ভিতরে যদি আরেকটা অবজেক্ট থাকে আর সেখানে `this` ইউজ করা হয় তাহলে কি হবে? হ্যাঁ তাহলে দেখে নেই ঐরকম একটা উদাহরনঃ

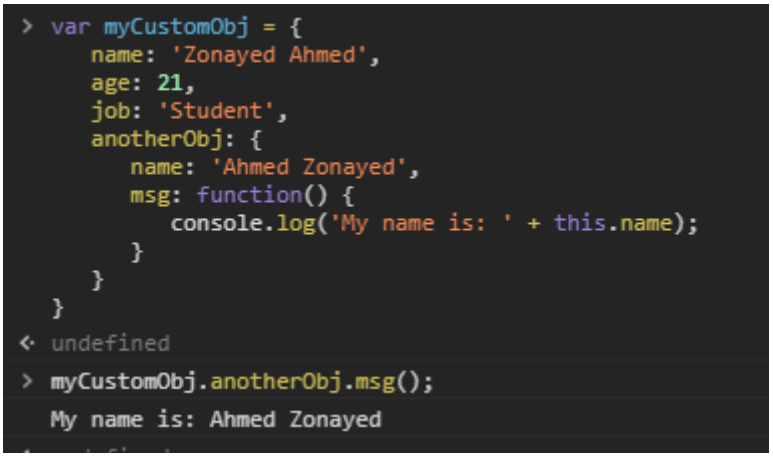
```
var myCustomObj = {
   name: 'Zonayed Ahmed',
   age: 21,
   job: 'Student',
   anotherObj: {
      name: 'Ahmed Zonayed',
      msg: function() {
         console.log('My name is: ' + this.name);
      }
   }
```

```
}
```

এখন দ্বিতীয় অবজেক্ট এর `msg` ফাংশনটাকে কল করিঃ

```
myCustomObj.anotherObj.msg();
```

এখন বলেন তো দেখি এখানে `this` কাকে ইন্ডিকেট করবে? `myCustomObj` নাকি `anotherObj` কে? দ্বিতীয় রুলসটা আবার মনে করার চেষ্টা করুন। `this` কীওয়ার্ডের ভ্যালু ডিটারমাইন করা হয় কাছের অবজেক্টটাকে দেখে। এখানে কাছের অবজেক্ট `anotherObj` , তাই এখানে `this.name` `anotherObj` এর `name` টাকেই ইন্ডিকেট করবেঃ



আরো ক্লিয়ায়লি দেখতেঃ

```
var myCustomObj = {
   name: 'Zonayed Ahmed',
```

```
    age: 21,
    job: 'Student',
    anotherObj: {
       name: 'Ahmed Zonayed',
       value: function() {
          console.log(this);
       }
    }
}
```

এখন এখানে দ্বিতীয় অবজেক্ট এর `value` ফাংশনটাকে কল করলেঃ

```
myCustomObj.anotherObj.value();
```

তাহলে দেখবেন এটা দ্বিতীয় অবজেক্ট অর্থাৎ `anotherObj` কেই প্রিন্ট করছেঃ

এখন আপনি চাইলে এখানে `this` এর ভ্যালু কাস্টমভাবেই ডিফাইন করে দিতে পারবেন, কোন অবজেক্ট কে ইন্ডিকেট করবে সেটা বলে দিতে পারবেন। এখানেই আসবে তৃতীয় রুলস(স্পষ্ট রুলস)

৩। স্পষ্ট রুলসঃ আপনারা হয়তো `call`, `bind`, `apply` মেথডের কথা শুনেছেন। এগুলোই আসলে স্পষ্টভাবে `this` কীওয়ার্ডের ভ্যালু সেট করতে ইউজ করা হয়। কোথাও যদি দেখেন এগুলো ইউজ হয়েছে তাহলে খুব সহজেই সেখানে `this` কীওয়ার্ড কাকে ইন্ডিকেট করছে সেটা ধরে ফেলতে পারবেন। কারণ এই `call`, `bind`, `apply` মেথডগুলো ইউজ করে প্রথম প্যারামিটারেই `this` কীওয়ার্ড কাকে ইন্ডিকেট করবে সেটা সেট করা যায়। এগুলো প্রত্যেকটা নিয়ে আলাদা আলাদা পর্ব লেখা হবে তাই এখানে বিস্তারিত আলোচনা করা হবে না। তবে আগের উদাহরণটায় আমরা একটা ইউজ করে দেখবো। আমাদের অবজেক্ট টা এরকম ছিলো দেখতেঃ

```
var myCustomObj = {
   name: 'Zonayed Ahmed',
   age: 21,
   job: 'Student',
   anotherObj: {
      name: 'Ahmed Zonayed',
      value: function() {
         console.log(this);
      }
   }
}
```

এখন `value` ফাংশনটাকে কল করলে এরকম প্রিন্ট হবেঃ

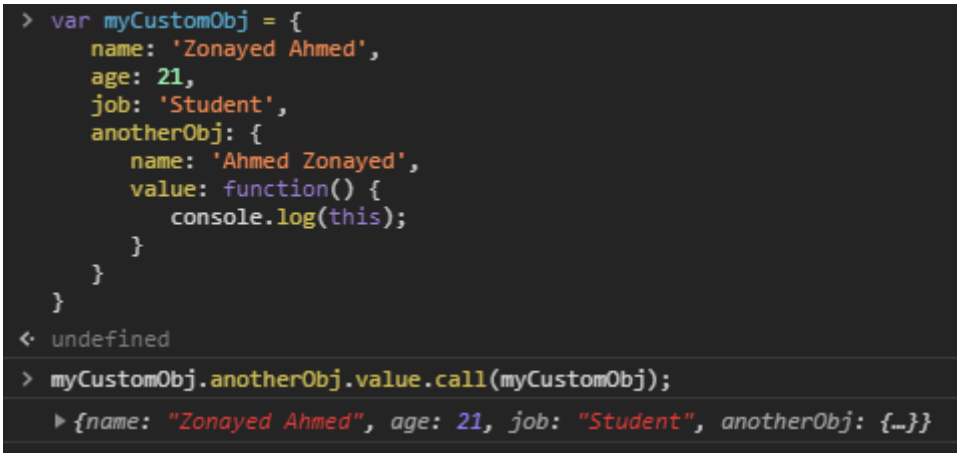
```
myCustomObj.anotherObj.value();
```

মানে এখানে `this` `anotherObj` কে ইন্ডিকেট করছে। কিন্তু আমরা যদি কাস্টমভাবে বলে দিতে চাই `anotherObj` না বরং আমরা চাই এখানে `this` `myCustomObj` কে ইন্ডিকেট করুক তাহলে এভাবে call মেথড ইউজ করে বলে দিতে পারিঃ

```
var myCustomObj = {
   name: 'Zonayed Ahmed',
   age: 21,
   job: 'Student',
   anotherObj: {
      name: 'Ahmed Zonayed',
      value: function() {
         console.log(this);
      }
   }
}
```

এখন এখানে `this` আমাদের কথামতো যাতে `myCustomObj` কেই ইন্ডিকেট করে সেটা call মেথড ইউজ করে বলে দিবোঃ

```
myCustomObj.anotherObj.value.call(myCustomObj);
```



`call`, `bind`, `apply` এই মেথডগুলো শুধুমাত্র ফাংশনের সাথে ইউজ করা যায় যেমনটা আমরা উদাহরণটায় করেছি। অন্য কোনো ডাটা টাইপ যেমন অবজেক্ট, স্ট্রিং, নাম্বার বা বুলিয়ানের সাথে ইউজ করা যাবে না। এগুলো নিয়ে বিস্তারিত পরের পর্বগুলোয় আলোচনা করা হবে। এখানে জাস্ট আমি ধারণা দিলাম। এভাবেই `this` এর ভ্যালু কাস্টমভাবে বা স্পষ্টভাবে বলে দেওয়া যায়।

৪। `new` কীওয়ার্ড রুলসঃ শেষ কিন্তু বহুল ব্যবহৃত আরেকটা কীওয়ার্ড `new` ও `this` কীওয়ার্ডের ভ্যালু ডিটারমাইন করতে পারে। কোথাও `this` কোনোভাবে এই `new` কীওয়ার্ডের আওতায় থাকলে সেটার ভ্যালুও অন্যরকম হতে পারে। এ নিয়ে আমি পরে বিস্তারিত পর্ব লিখবো। এর জন্যে আমাদের আরো কিছু কনসেপ্ট ক্লিয়ার করে নিতে হবে আগে।এখানে জাস্ট ধারণা দিয়ে রাখলাম এই কীওয়ার্ডও `this` এর ভ্যালু ডিটারমাইন করতে পারে।

আশা করি এখন থেকে আর এই `this` কীওয়ার্ড নিয়ে আর কনফিউশন সৃষ্টি হবেনা।

তবে সবসময় মনে রাখবেনঃ

- গ্লোবালি যেকোনো জায়গায়, কাস্টমভাবে তৈরীকৃত অবজেক্ট এর ভিতরে ছাড়া [redacted] সবসময় গ্লোবাল

অবজেক্টকেই ইন্ডিকেট করবে।

- এটা নরমাল রেগুলার ফাংশন কলেও গ্লোবাল অবজেক্ট কে পয়েন্ট করে
- এটার ভ্যালু অ্যাসাইন হয় না যতক্ষন না পর্যন্ত আপনি ডিক্লেয়ারকৃত ফাংশনটাকে কল না করছেন বা এটা যেখানে ইউজ করা হয়েছে সেটাকে কল না করছেন।

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

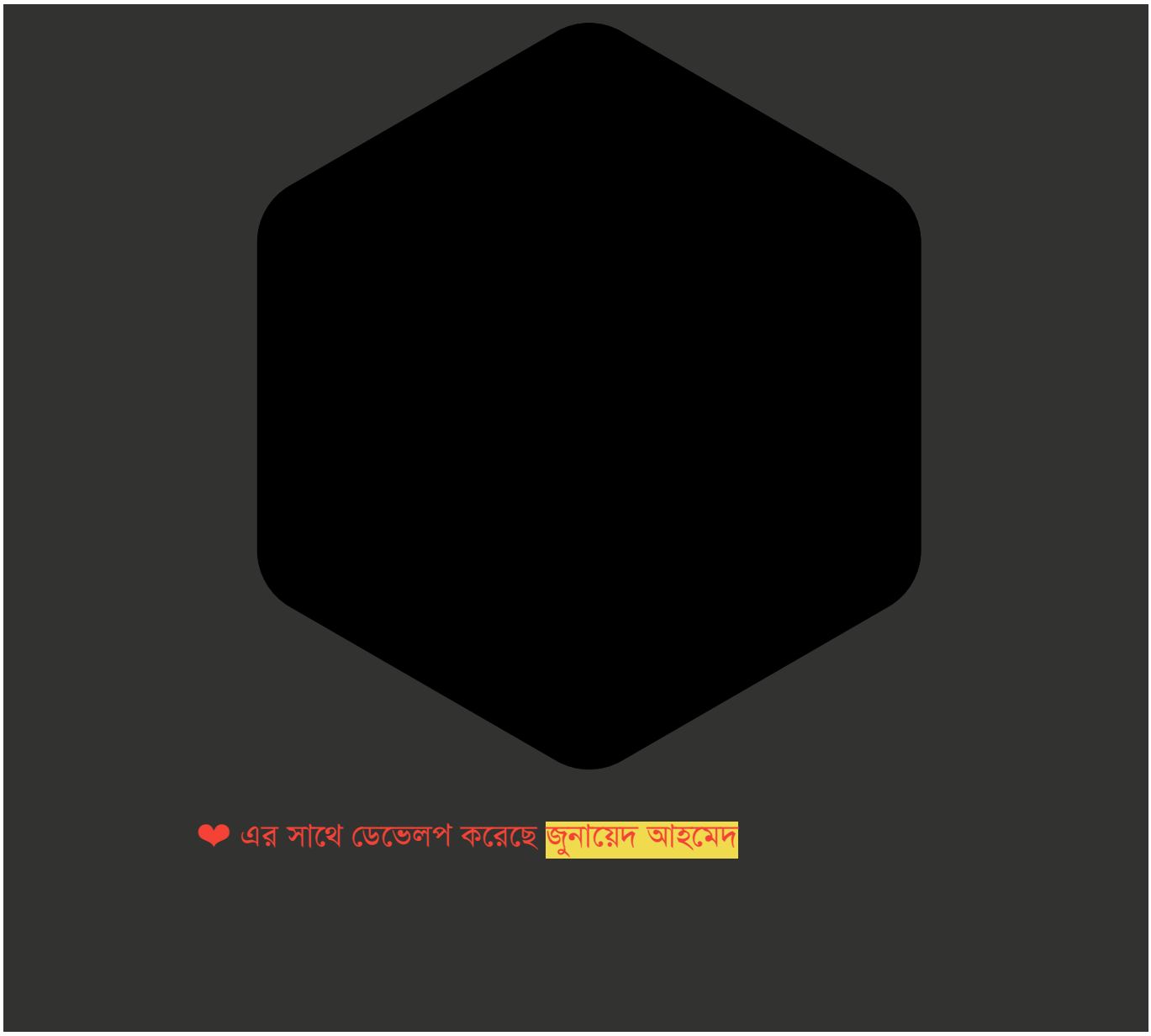
প্রোজেক্টটি সম্পূর্ণ সোর্স কোডসহ গিটহাবে রয়েছে। আপনার ভালো লেগে থাকলে স্টার দিয়ে আসবেন। আপনার পরামর্শ, মন্তব্য এবং ভুলত্রুটি গিটহাবে ইস্যু করে দেওয়ার জন্যে অনুরোধ থাকলো

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