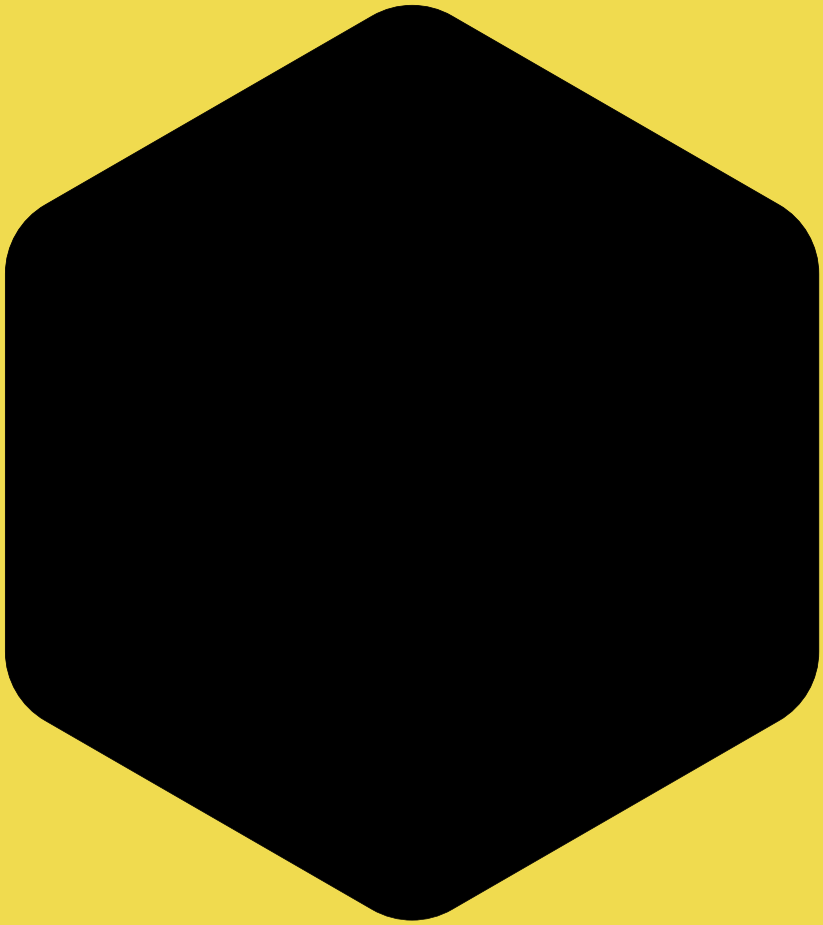
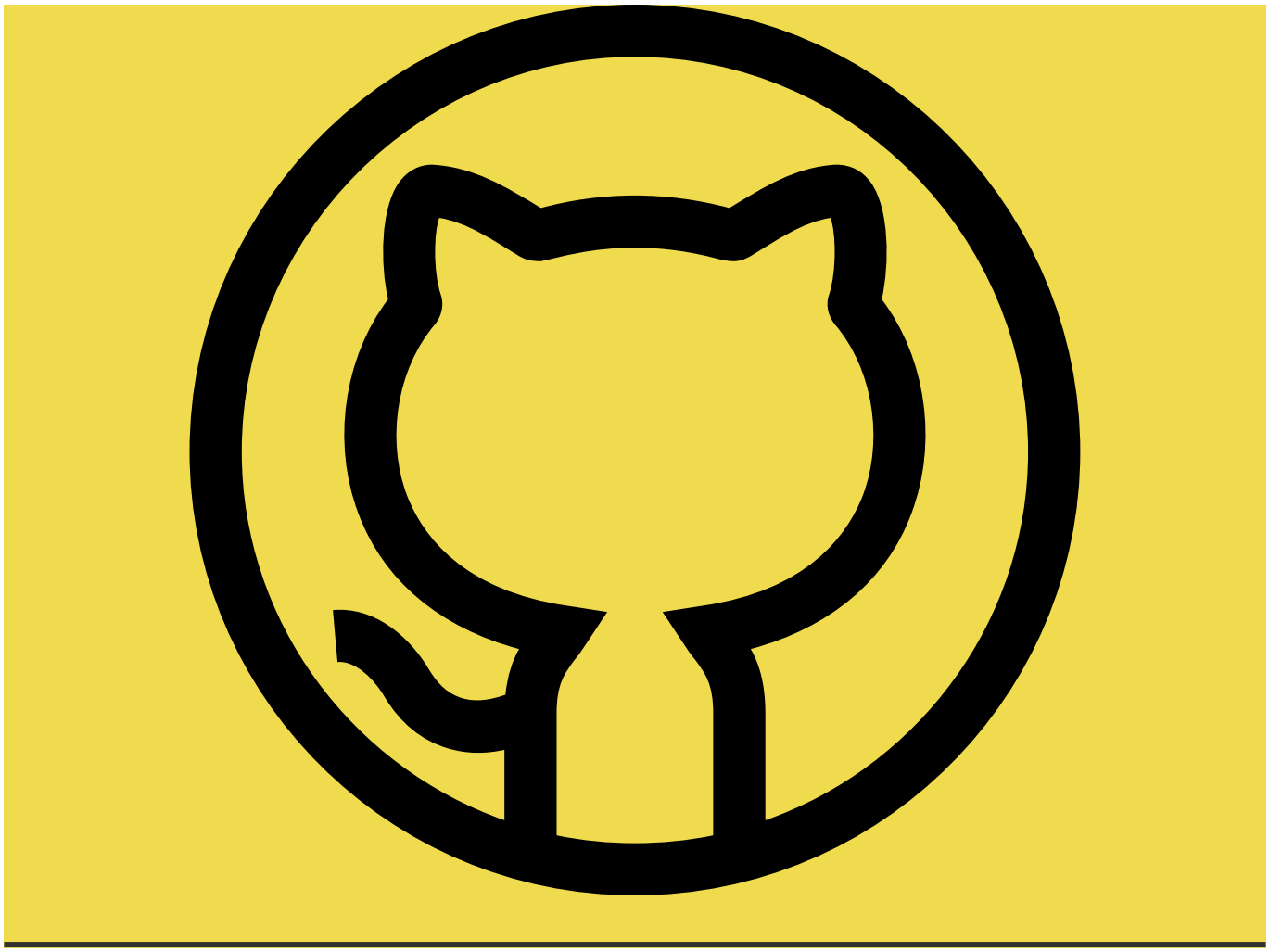
This app works best with JavaScript enabled.



হাতেকলমে জাভাস্ক্রিপ্ট



জাভাস্ক্রিপ্ট অ্যাডভান্সঃ অবজেক্ট ওরিয়েন্টেড জাভাস্ক্রিপ্ট

জাভাস্ক্রিপ্ট আসলে অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজ(Object Oriented Programming Language)। যদিও অন্যান্য OOP(Object Oriented Programming) ল্যাংগুয়েজগুলো থেকে জাভাস্ক্রিপ্ট এ অনেক কিছু একটু অন্যরকম, কিন্তু তারপরেও এটা অবজেক্ট ওরিয়েন্টেটেড ল্যাংগুয়েজের মধ্যেই পড়ে। নতুন ভার্শনগুলোয় সেগুলো আপডেটও করা হচ্ছে আস্তে আস্তে। আর তাই আজকে এই পর্বে আমরা সেই অবজেক্ট ওরিয়েন্টেড জাভাস্ক্রিপ্ট নিয়ে আলোচনা করবো।

জাভাস্ক্রিপ্ট এ অলমোস্ট সবকিছুই অবজেক্ট। কিছু ছাড়া, যেগুলোকে প্রিমিটিভ ডাটা টাইপ বলে।

জাভাস্ক্রিপ্ট এ সাধারণত দুই ধরনের ডাটা টাইপ আছেঃ

১। প্রিমিটিভ ডাটা টাইপ(Primitive Data Type)

২। অবজেক্ট

১। প্রিমিটিভ ডাটা টাইপ(Primitive Data Type)ঃ জাভাস্ক্রিপ্ট এ প্রিমিটিভ ডাটা টাইপ খুবই অল্প। যেমন নাম্বার(Number), স্ট্রিং(String), বুলিয়ান(Boolean), আন্ডিফাইন্ড(`undefined`), নাল(`null`) এগুলো হচ্ছে প্রিমিটিভ ডাটা টাইপ।

২। অবজেক্টঃ উপরে উল্লেখিত প্রিমিটিভ ডাটা টাইপগুলো ছাড়া বাকি সবই জাভাস্ক্রিপ্ট এ অবজেক্ট। যেমনঃ অ্যারে, ফাংশন, অবজেক্ট, ডেট, নাম্বার/স্ট্রিং/বুলিয়ান এর র‍্যাপার(Wrapper) ফাংশন সবই অবজেক্ট।

এখন প্রিমিটিভ ডাটা টাইপ আর অবজেক্ট এর মধ্যে প্রধান ডিফারেন্স হচ্ছে এদের ভ্যালু স্টোর করার সিস্টেমে। যেখানে প্রিমিটিভ ডাটা টাইপে ভ্যালু নিজের কাছেই স্টোর করে, অন্যদিকে অবজেক্ট সাধারণত সরাসরি ডাটা স্টোর করে না, বরং ঐ ডাটাটা অন্যকোথাও থাকে, অবজেক্ট এ জাস্ট ঐটার রেফারেন্স থাকে।

আর যখন আমরা প্রিমিটিভ টাইপের কোনো ডাটা প্যারামিটার/আর্গুমেন্ট হিসাবে কোনো ফাংশনে পাস করি, ঐটা সবসময় ঐ ডাটার কপি পাঠায়, সরাসরি ডাটা পাঠায় না। একটা কমন উদাহরণ দিলে হয়তো বুঝতে পারবেন। নিচের এই প্রোগ্রামটা দেখিঃ

```
var a = 10, b = 20;
```

```
console.log('Before swap: Value of a: ' + a + ' and value
of b: ' + b);
```

```
function swap(a, b) {
   console.log('Before Swap inside function: Value of a:
' + a + ' and value of b: ' + b);
   var temp = a;
   a = b;
   b = temp;
   console.log('After Swap inside function: Value of a: '
+ a + ' and value of b: ' + b);
}
```

```
swap(a, b);
console.log('After swap: Value of a: ' + a + ' and value
of b: ' + b);
```

এ প্রোগ্রামটা রান করালে দেখবেন ফাংশনের ভিতরে `a` এবং `b` ঠিকি সোয়াপ হয়েছে কিন্তু বাইরের `a` আর `b` তে কোনো ইফেক্ট পড়ে নাইঃ

```
> var a = 10, b = 20;
  console.log('Before swap: Value of a: ' + a + ' and value of b: ' + b);
  function swap(a, b) {
     console.log('Before Swap inside function: Value of a: ' + a + ' and value of b: ' + b);
     var temp = a;
     a = b;
     b = temp;
     console.log('After Swap inside function: Value of a: ' + a + ' and value of b: ' + b);
  }
  swap(a, b);
  console.log('After swap: Value of a: ' + a + ' and value of b: ' + b);
  Before swap: Value of a: 10 and value of b: 20
  Before Swap inside function: Value of a: 10 and value of b: 20
  After Swap inside function: Value of a: 20 and value of b: 10
  After swap: Value of a: 10 and value of b: 20
< undefined
```

কারণ এখানে আমরা প্রিমিটিভ টাইপের ডাটা পাঠিয়েছি আর্গুমেন্টে, এটা নিজের ডাটা সেন্ড না করে সেটার কপি সেন্ড করছে, আর তাই এটার নিজের উপর কোনো ইফেক্ট পড়ে নাই।

কিন্তু সেইম কাজ যদি আমরা অবজেক্ট(রেফারেন্স) ডাটা টাইপ দিয়ে করিঃ

```
var obj = {
   a: 10,
   b: 20
};
```

```
console.log('Before swap: Value of a: ' + obj.a + ' and
value of b: ' + obj.b);
```

```
function swap(x) {
   console.log('Before Swap inside function: Value of a:
```

```
' + x.a + ' and value of b: ' + x.b);
   var temp = x.a;
   x.a = x.b;
   x.b = temp;
   console.log('After Swap inside function: Value of a: '
+ x.a + ' and value of b: ' + x.b);
}
```

```
swap(obj);
console.log('After swap: Value of a: ' + obj.a + ' and
value of b: ' + obj.b);
```

এখন দেখবেন আপনার ডাটা ফাংশনের ভিতরেও চেঞ্জ হয়েছে আবার সেটার ইফেক্ট বাইরেও পড়েছে। আপনার আসল `obj` এর ডাটাও চেঞ্জ হয়ে গেছে। এটার কারণ একটাই, এখানে `obj` অবজেক্ট টাইপ ডাটা, যেটা নিজে ডাটা হোল্ড করে না, বরং সেই ডাটার একটা রেফারেন্স হোল্ড করে যে এই ডাটাটা অমুক জায়গায় আছে। আপনি যখন একে আর্গুমেন্টে পাস করলেন, তো আপনি সেই রেফারেন্সটাই পাস করলেন। এখন ফাংশনের ভিতরে যখন ডাটাগুলো সোয়াপ করলেন, সেটা সেই রেফারেন্স অনুযায়ী গিয়ে আসল ডাটাকেই সোয়াপ করে দিছে। এজন্যে ফাংশন শেষ হওয়ার পরেও, ফাংশন থেকে কিছু রিটার্ন না করা সত্বেও আপনার ডাটা দেখবেন চেঞ্জ হয়ে গেছেঃ

```
> var obj = {
    a: 10,
    b: 20
};
console.log('Before swap: Value of a: ' + obj.a + ' and value of b: ' + obj.b);
function swap(x) {
    console.log('Before Swap inside function: Value of a: ' + x.a + ' and value of b: ' + x.b);
    var temp = x.a;
    x.a = x.b;
    x.b = temp;
    console.log('After Swap inside function: Value of a: ' + x.a + ' and value of b: ' + x.b);
}
swap(obj);
console.log('After swap: Value of a: ' + obj.a + ' and value of b: ' + obj.b);
Before swap: Value of a: 10 and value of b: 20
Before Swap inside function: Value of a: 10 and value of b: 20
After Swap inside function: Value of a: 20 and value of b: 10
After swap: Value of a: 20 and value of b: 10
< undefined
```

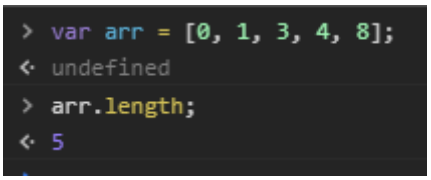
আশা করি এই দুই টাইপের ডাটা সম্পর্কে ধারণা হয়েছে। আরো জানতে চাইলে আপনি মাদার অব অল ল্যাংগুয়েজ C শিখতে পারেন।

এখন অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজগুলোতে সবকিছুই অবজেক্ট। এখন অবজেক্টটা কি? হ্যাঁ অবজেক্ট হচ্ছেন আপনি, আপনি পুরোটা একটা অবজেক্ট। আপনার কিছু প্রপার্টিজ(properties) আছে যেমন আপনার গায়ের রঙ, আপনার উচ্চতা, আপনি কোন দেশে থাকেন, আপনার নাম, আপনার চাকরী এগুলো সবই আপনার সম্পর্কে তথ্য দেয়। আপনাকে কেউ যদি জিজ্ঞাসা করে আপনার নাম কি তাহলে আপনি আপনার নাম বলবেন। ঠিক তেমনি জাভাস্ক্রিপ্ট এর অবজেক্ট এরও এরকম প্রপার্টিজ আছে। আর আমাদের অনেক ভাগ্য ভালো যে জাভাস্ক্রিপ্ট এ আসলে অবজেক্ট এর এইসব প্রপার্টিজ ব্রাউজারের কন্সোলে ওপেন করে দেখা যায়। কার কি প্রপার্টি আছে সবই দেখতে পারবেন ব্রাউজারের কন্সোল থেকে। যাই হউক ধরি আমাদের একটা অ্যারে আছেঃ

```
var arr = [0, 1, 3, 4, 8];
```

এখন এই অ্যারেতে কয়টা ইলিমেন্ট আছে সেটা জানতে আপনার নিজের গণনা করার দরকার নাই, আপনি চাইলে এই অ্যারেকেও জিজ্ঞাসা করতে পারেন। কারণ অ্যারেরও একটা প্রপার্টি আছে যে জানে তার ইলিমেন্ট কয়টাঃ

```
arr.length;
```

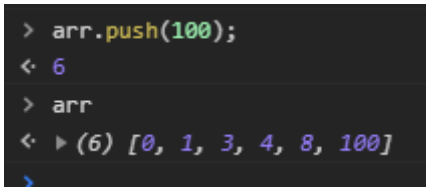


এভাবেই অবজেক্টগুলোতে তাদের নিজস্ব প্রপার্টি থাকে। এখন আপনার কাছে একটা মোবাইল ফোনও আছে(অনেকের কাছে নাও থাকতে পারে তবে ধরে নেই আছে)। এখন এই মোবাইল ইউজ করে আপনি কিছু অপারেশন করতে পারেন, যেমন দূরের কারো সাথে যোগাযোগ করা বা ইন্টারনেট ব্রাউজ করা। এমন অবজেক্টগুলোরও কিছু ম্যাশিন থাকে, যেগুলোকে ওদের মেথড বলে। মেথডও এক রকমের ফাংশন। যেমন আমরা যদি উক্ত অ্যারেতে আরেকটা আইটেম ঢুকাতে চাই তাহলে সেই অ্যারের একটা মেথড ইউজ করতে পারিঃ

```
arr.push(100);
```

এখন দেখেন অ্যারেতে সেটা অ্যাড হয়ে গেছেঃ

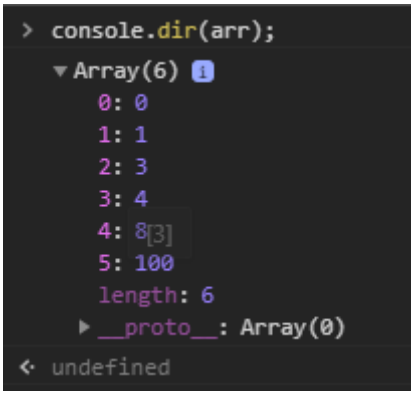
```
arr
```



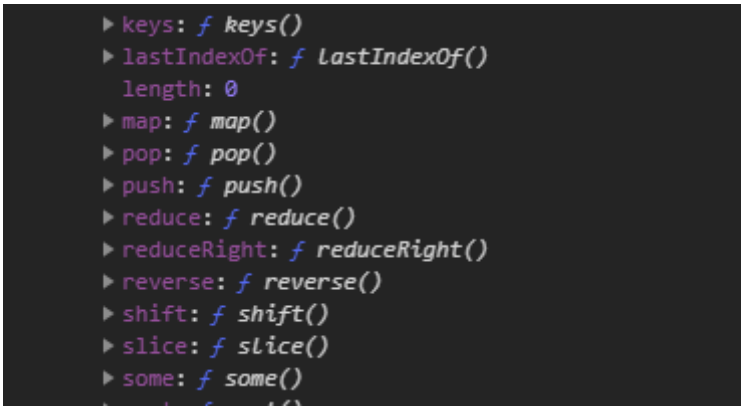
এখন এই `push` মেথড আসলো কোথা থেকে? হ্যাঁ সেটা আপনি এই অবজেক্ট অ্যারেটা ওপেন করে দেখতে পারেনঃ

```
console.dir(arr);
```

এটা রান করলে ব্রাউজারের কন্সোলে দেখবেন আপনার `Array` অবজেক্ট টা আসছে। এখন পাশের ত্রিভুজাকৃতির বাটনে ক্লিক করলে সেই অবজেক্ট টা খুলে যাবেঃ



এখানে আমরা `length` প্রপার্টি দেখতে পাচ্ছি, যেটা এই অ্যারে অবজেক্ট এর একটা প্রপার্টি সেটা কনফার্ম হওয়া গেলো। কিন্তু `push` মেথডটা কোথায়? হ্যাঁ সেজন্যেই নিচে দেখেন `__proto__` নামে আরেকটা জিনিস আছে। ঐটাও ওপেন করতে পারবেন। ওপেন করলে অনেকগুলো প্রপার্টি আর মেথড দেখবেন। এগুলো সবই অ্যারের মেথড। এখানে খুজলে আপনার `push` মেথডও পাবেনঃ



তো এখানেই অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ল্যাংগুয়েজগুলোর আসল খেলা। এভাবেই অবজেক্টগুলো একটা আরেকটার সাথে মিলিতভাবে কমপ্লেক্স একটা স্ট্রাকচার তৈরী করে আমাদের কাজের সুবিধার্থে অনেকগুলো প্রপার্টি আর মেথড একসাথে করে ফেলে। এখানে যেমনটা দেখলেম `length` আর `push` ।

এভাবেই প্রত্যেকটা জাভাস্ক্রিপ্ট অবজেক্ট এর প্রোটোটাইপ প্রপার্টি(Prototype property) থাকে যার জন্যে এরকম প্রপার্টি বা মেথড ধার(Inherit) করে আনা যায়। আর এভাবেই জাভাস্ক্রিপ্ট বাই ডিফল্ট প্রোটোটাইপ প্রপার্টি(Prototype property) তে মেথড বা প্রপার্টিগুলো রাখে আমাদের পরবর্তিতে সেই অবজেক্ট এর সাথে ইউজ করার জন্যে। আমরা চাইলে নিজেওরাও কাস্টমভাবে সেই প্রোটোটাইপ প্রপার্টি(Prototype property) তে মেথড বা প্রপার্টি রাখতে পারবো। তবে সেটা আমি পরে কোনো পর্বে দেখাবো। আজকে জাস্ট মেইন আইডিয়াটা দিয়ে রাখলাম।

প্রোটোটাইপ চেইন(Prototype Chain):

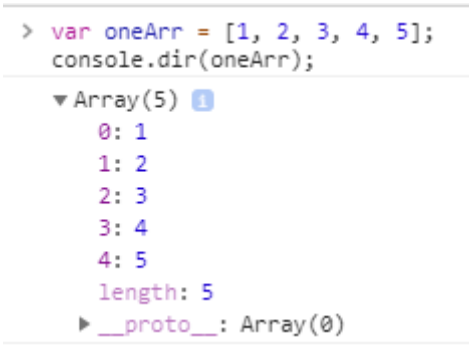
জাভাস্ক্রিপ্ট অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ল্যাঙ্গুয়েজ হলেও মূলত এটা প্রোটোটাইপ বেসড। অন্যান্য

মেজর ল্যাঙ্গুয়েজের অবজেক্ট মডেল আর জাভাস্ক্রিপ্ট এর অবজেক্ট মডেল এক না। জাভাস্ক্রিপ্ট এর সব অবজেক্টেই প্রোটোটাইপ `__proto__` অবজেক্ট পাবেনঃ

```
console.dir({});
```

এই `__proto__` টাকে আসলে ডান্ডার প্রোটো (dunder proto) বলে। সব অবজেক্টেই পাবেন এটাঃ

```
var oneArr = [1, 2, 3, 4, 5];
console.dir(oneArr);
```



এভাবে যখন আমরা আমাদের অ্যারের ভিতরে প্রথম ওপেন করার পর শুধুমাত্র ইলিমেন্ট আর `length`

প্রপার্টি টা দেখলেম শুধুমাত্র, বাকিগুলো বা `push` মেথড কিন্তু সেখানে ছিলো না। সেটা যখন আমরা আবার ডান্ডার প্রোটো `__proto__` (Dunder Proto) ওপেন করলাম তখন আমরা `push` পাইলাম। আবার একদম নিচে গেলে আবার দেখবেন সেই ডান্ডার প্রোটো আরেকটা আছে। ঐটা ওপেন করলেও আরো কিছু মেথড পাবেন। এখন এটাকেই আসলে প্রোটোটাইপ চেইন বলে। এখানে যখন একটা মেথড কল করা হয়, প্রথমে সেটা একদম প্রথম থেকে অবজেক্ট একদম নিজের থেকে খোঁজাখুঁজি শুরু করে, সেখানে না পাওয়া গেলে ভিতরে(ডান্ডার প্রোটোর ভিতরে) আবার খোঁজা হয়, সেখানেও না পাওয়া গেলে আরো ভিতরে(আরেকটা ডান্ডার প্রোটোর ভিতরে) খোঁজা হয়... এরকমভাবে খুঁজতেই থাকে যতক্ষন না সেই মেথড বা প্রপার্টি না পাওয়া যায়।

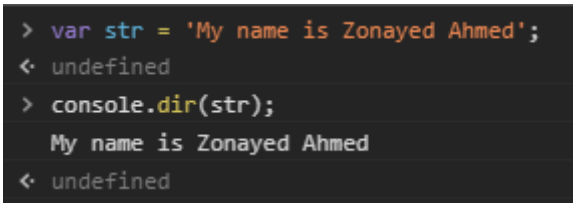
তো এভাবেই জাভাস্ক্রিপ্ট এর অবজেক্ট ওরিয়েন্টেড এর ব্যাপারটা প্রোটোটাইপ বেসড কাজ করে। সব অবজেক্ট এভাবে একটা আরেকটার সাথে লিঙ্ক হয়ে থাকে এই মডেলের মাধ্যমে।

এখন আমরা এখানে জানলাম স্ট্রিং নাকি অবজেক্ট না, প্রিমিটিভ। তাহলে স্ট্রিং এর তো কোনো প্রপার্টি বা মেথড থাকার কথা না। নাকি? চলুন দেখি তাহলেঃ

```
var str = 'My name is Zonayed Ahmed';
```

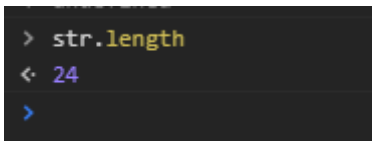
দেখি তাহলে কিছু আছে নাকিঃ

```
console.dir(str);
```



কিচ্ছু নাই! হ্যাঁ এটাই আমরা আশা করছিলাম। কারণ এটা প্রিমিটিভ, তাই স্বাভাবিকভাবেই এর কোনো প্রপার্টি বা মেথড থাকার কথা না। কিন্তু দেখি একটা অপারেশন চালিয়েঃ

```
str.length
```



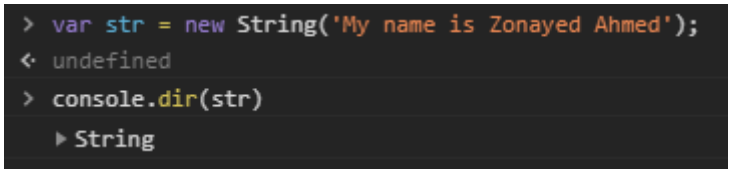
কিন্তু একি! এরো দেখি `length` প্রপার্টি আছে! কিভাবে কি...? প্রিমিটিভ হলে তো কোনো প্রপার্টি থাকার কথা না। হ্যাঁ এখানেই আসে র‍্যাপার(Wrapper) ফাংশনের কথা।

আপনি সেইম স্ট্রিং টা একটা স্ট্রিং র‍্যাপার(Wrapper) ফাংশন দিয়েও নিতে পারবেনঃ

```
var str = new String('My name is Zonayed Ahmed');
```

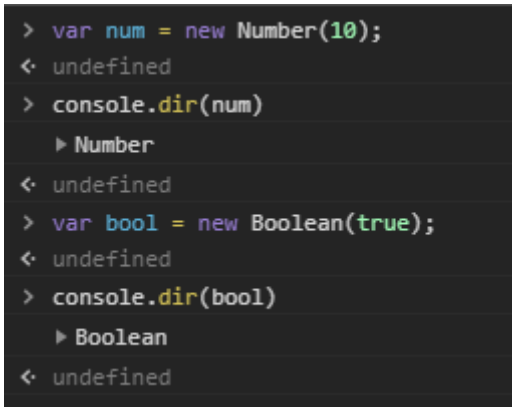
এখন এটাকে খুলে দেখলেঃ

```
console.dir(str);
```



এখন এখানে এটা অবজেক্ট। আপনি খুলে ভিতরে এর প্রপার্টি বা মেথডগুলোও দেখতে পারবেন। র‍্যাপার(Wrapper) ফাংশনের কাজগুলো ঠিক তাই। এটা একটা প্রিমিটিভকে অবজেক্ট করে ফেলতে পারে, আর সেই সুবিধার্থে আমরা অনেকগুলো প্রপার্টি ও মেথড এর অ্যাক্সেস পাই। এখন স্ট্রিং এর জন্যে `String` আর নাম্বারের জন্যে `Number` এবং বুলিয়ান এর জন্যে `Boolean` র‍্যাপার ফাংশন রয়েছেঃ

```
var num = new Number(10);
console.dir(num);
var bool = new Boolean(true);
console.dir(bool);
```



এগুলোর কাজই হচ্ছে কাঙ্ক্ষিত প্রিমিটিভ ডাটা টাইপকে অবজেক্ট টাইপ করে ফেলা আর সেই সুবাধে আমরাও অনেক প্রপার্টি বা মেথডের অ্যাক্সেস পাই।

কিন্ত উপরের উদাহরণে দেখলাম প্রিমিটিভ টাইপ নেওয়ার পরও আমাদের কিছু প্রপার্টির অ্যাক্সেস পাচ্ছি। কিভাবে? হ্যাঁ এখানেই আসল ফ্লেক্সিবিলিটি। আমাদের সেই প্রিমিটিভ টাইপেও ঐটার অবজেক্ট রূপে ঠিক যতগুলো প্রপার্টি বা মেথডের অ্যাক্সেস পাইতাম ঐটাতেও আমরা অটোম্যাটিকালি সেগুলো পাবো। কারণ আমরা যখন সেই প্রিমিটিভ এর কোনো প্রপার্টি বা মেথডের জন্য কোড রান করি, জাভাস্ক্রিপ্ট তখন সেটাকে ফোর্স করে র‍্যাপার ফাংশন দিয়ে(`new String`, `new Boolean`, `new Number`) অবজেক্ট রূপে নিয়ে যায় আর আমাদের কোডও ঠিকমতো রান করে। আমাদের ভাবাই লাগে না। এজন্যেই আমরা যখন প্রিমিটিভ টাইপের ডাটা খুলতে চেষ্টা করি, তখন কিছু পাই না। কিন্তু কোনো মেথড বা প্রপার্টি দিলে তখন আবার কাজ করে। কারণ আমরা যখনি কোনো মেথড বা প্রপার্টি দেই, তখনি সেটাকে জাভাস্ক্রিপ্ট ফোর্স করে সেটাকে অবজেক্ট এ নিয়ে যায়। আবার মনে রাখবেন একবার প্রপার্টি বা মেথড অ্যাক্সেস করার কারণে কিন্তু আপনার প্রিমিটিভ টাইপের ডাটা অবজেক্ট রূপে পার্মানেন্টলি চলে যায় না, কারণ এখানে আপনার মেইন ডাটা টাচ করাই হয় না। আপনি চাইলে আপনার কন্সোলেও দেখতে পারেন একটা ট্রাই করে।

উপরের অপারেশনটা আরো গভীরভাবে দেখলেঃ

```
var str = 'My name is Zonayed Ahmed';
```

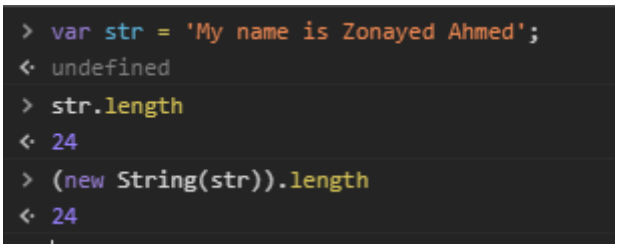
এখন আমরা `length` অ্যাক্সেস করতে চাই এভাবেঃ

```
str.length
```

কিন্তু পিছনে আসলে যা হয়ঃ

```
(new String(str)).length
```

এর ফলে আমরা দেখতে পাই আমাদের ফলাফলঃ



কিন্তু এটা দেখেন পার্মানেন্ট কোথাও অ্যাসাইন করা হচ্ছে না, তাই আমাদের আসল প্রিমিটিভ প্রিমিটিভই থেকে যাচ্ছে। ঐ ডাটাকে টাচই করা হয় নাই।

আশা করি আজকে অবজেক্ট সম্পর্কে ভালো আইডিয়া হয়েছে। আর কখনো এটা নিয়ে কনফিউজ হবেন

না। তারপরেও কোনো প্রশ্ন থাকলে আমাকে ইমেইল করতে পারেন `contact@zonayed.me` এ

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

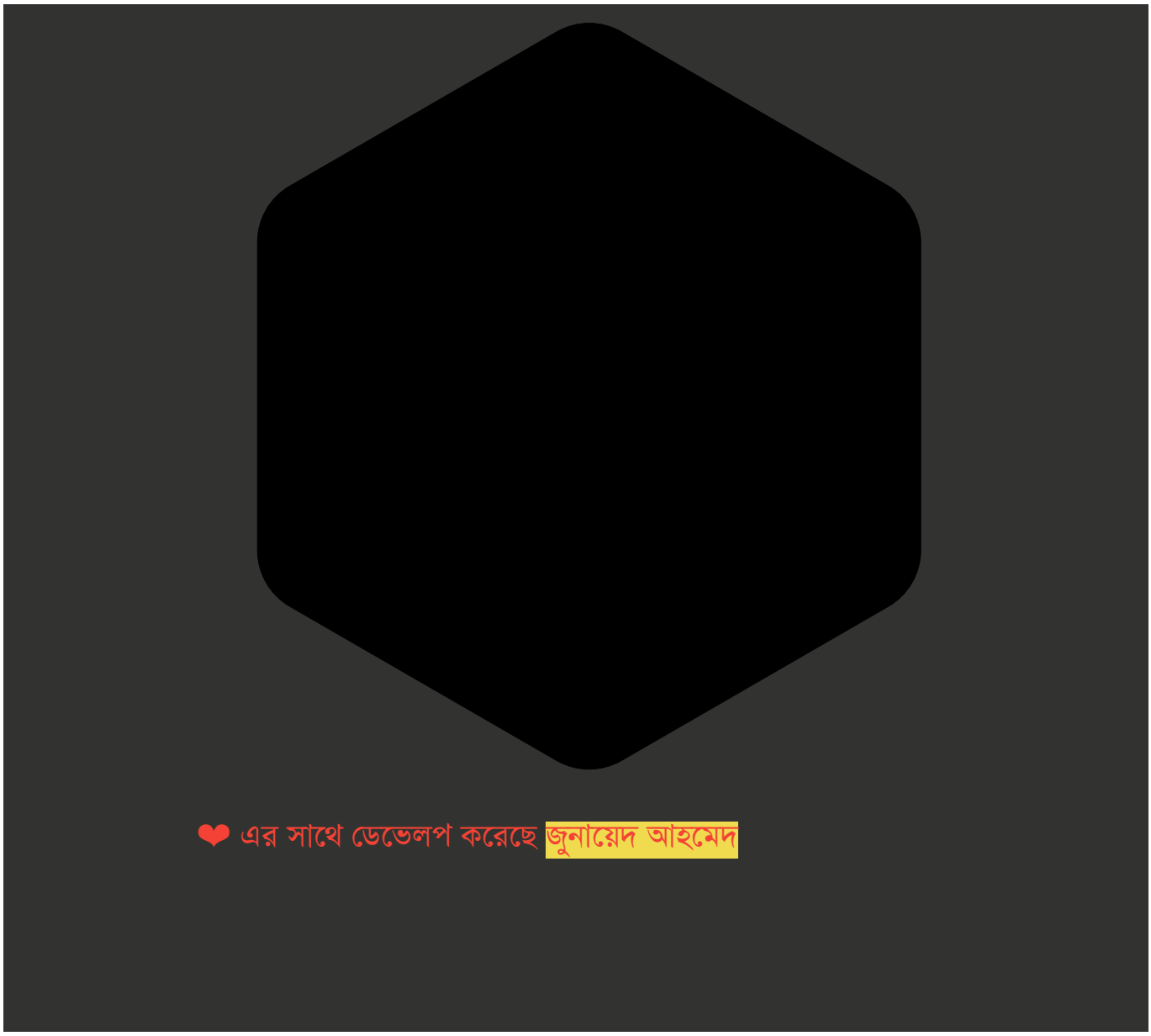
প্রোজেক্টটি সম্পূর্ণ সোর্স কোডসহ গিটহাবে রয়েছে। আপনার ভালো লেগে থাকলে স্টার দিয়ে আসবেন। আপনার পরামর্শ, মন্তব্য এবং ভুলত্রুটি গিটহাবে ইস্যু করে দেওয়ার জন্যে অনুরোধ থাকলো

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