

This app works best with JavaScript enabled.



হাতেকলমে জাভাস্ক্রিপ্ট



জাভাস্ক্রিপ্টঃ অপারেটর নিয়ে সবকিছু

এই পর্বে আমি অপারেটর নিয়ে বিস্তারিত আলোচনা করবো। অপারেটর খুবই ব্যাসিক জিনিস, আমরা সবাই এমনিতেও জানি। কিন্তু এ সম্পর্কে ভালো ধারণা থাকতে হবে। কারণ অপারেটর অনেকভাবে ইউজ করা হয়।

ধরুন $2 + 8$ এ 6 হয়, এখন এখানে $+$ হচ্ছে অপারেটর, আর 2 এবং 8 হচ্ছে অপারেণ্ড। আমরা এখন এইধরনের অপারেটর নিয়েই আলোচনা করবো।

এই পর্বে অনেককিছু কি কাজে লাগতে পারে সেটা নিয়া কনফিউশন তৈরী হতে পারে, কিন্তু এগুলো

আসলে আপনার সবকিছুই বেইস। তাই আপনার এগুলার ব্যাপারে ক্লিয়ার ধারণা থাকা জরুরী। আপনাকে যে সবকিছু মনে রাখতে হবে বা মুখস্ত করে ফেলতে হবে এমন কোনো কথা নাই। প্র্যাক্টিস করতে করতে এমনই মনের মধ্যে ঢুকে যাবে।

জাভাস্ক্রিপ্ট এ কয়েকরকম এর অপারেটর আছেঃ

১। অ্যারিথমেটিক অপারেটর

২। কম্পারিজম অপারেটর

৩। লজিক্যাল/রিলেশনাল অপারেটর

৪। অ্যাসাইনমেন্ট অপারেটর

৫। কন্ডিশনাল/টার্নারি অপারেটর

এগুলো নাম থেকে অনেকটা ধারণা করা যায় কোনগুলো কোন কাজের জন্য হতে পারে। আমি নিচে বিস্তারিত আলোচনা করছি প্রত্যেকটা। আপনি ব্রোমের ডেভেলপার কন্সোল ওপেন করে প্র্যাক্টিস করবেন একটা একটা করেঃ

১। অ্যারিথমেটিক অপারেটরঃ যোগ, বিয়োগ, গুণ, ভাগ করার জন্যে এই অ্যারিথমেটিক অপারেটর ইউজ করা হয়। তবে এগুলো ছাড়াও আরো কয়েকটা আছেঃ

• ■ — দুইটা অপারেন্ড যোগ করার জন্যে

> 10 + 10

< 20

সাধারণ যোগ

```
> var a = 10;
< undefined
> var b = 10;
< undefined
> a + b
< 20
> |
```

সেইম ভ্যারিয়েবলের ক্ষেত্রেও

- ■ — দুইটা অপারেন্ড বিয়োগ করার জন্যে

```
> 20 - 10
< 10
> var a = 20;
< undefined
> var b = 10;
< undefined
> a - b
< 10
> |
```

বিয়োগ


- ■ — দুইটা অপারেন্ড গুণ করার জন্যে

```

> 20 * 10
< 200
> var a = 20;
< undefined
> var b = 10;
< undefined
> a * b
< 200

```

গুণ

-  — দুইটা অপারেন্ড ভাগ করার জন্যে

```

> 20 / 10
< 2
> var a = 20;
< undefined
> var b = 10;
< undefined
> a / b
< 2

```

ভাগ

স্পেশাল নোটঃ ভাগফল দশমিক এ আসলে দশমিকেই রেজাল্ট শো করবে

```

> 100 / 3
< 33.333333333333336

```

দশমিক এ দেখাচ্ছে।

- **মডুলাস** — ভাগশেষ বের করার জন্যে, এটা আসলে অনেক ইউজফুল একটা অপারেটর। ১৩ ভাগ ৫ এ রেজাল্ট হয় ২, কিন্তু ভাগশেষ থাকে ৩। এই ভাগশেষ বের করতেই এই অপারেটর ইউজ করা হয়

```
> 13 % 5
< 3
```

```
> var a = 13;
< undefined
```

```
> var b = 5;
< undefined
```

```
> a % b
< 3
```

ভাগশেষ দেখাচ্ছে

- **ইনক্রিমেন্ট** — এটা আপনার অপারেন্ড এর সাথে ১ যোগ করবে। ধরুন আপনার একটা ভ্যারিয়েবল আছে **a** যেটার ভ্যালু ১০, এখন **a++** লিখলে এটার ভ্যালুর সাথে এক যোগ হবে। এটা **a++** এটার শর্টকাট বলা চলে।

```
> var a = 10;
< undefined
```

```
> a++;
< 10
```

```
> a
< 11
```

ভ্যালু ইনক্রিমেন্ট হচ্ছে

- **ডিক্রিমেন্ট** — অনেকটা ইনক্রিমেন্ট অপারেটরের মতোই কিন্তু ভ্যালু ১ কমাতে। সেইমভাবে এটাও **a--** এটার শর্টকাট।

```
> var b = 5;
```

```
< undefined
```

```
> b--;
```

```
< 5
```

```
> b
```

```
< 4
```

ভ্যালু ডিক্রিমেন্ট হচ্ছে

লক্ষণীয়ঃ এই ইনক্রিমেন্টাল/ডিক্রিমেন্টাল অপারেটরগুলো আপনার ভ্যারিয়েবলের আগে এবং পরে দুই পজিশনেই বসতে পারে। পার্থক্য হলো আগে থাকলে ভ্যালুর ইনক্রিমেন্ট/ডিক্রিমেন্ট আগে হয়, তারপর ভ্যালু রিটার্ন করে। আর পরে থাকলে আগে রিটার্ন করে, পরে ভ্যালুর ইনক্রিমেন্ট/ডিক্রিমেন্ট হয়।

```
> var a = 10;
```

```
< undefined
```

```
> a++
```

```
< 10
```

```
> |  
  a++ ১০ ই দেখাচ্ছে
```

```
> var a = 10;
```

```
< undefined
```

```
> a++
```

```
< 10
```

```
> a
```

```
< 11
```

কিন্তু পরে আবার অ্যাক্সেস করায় দেখা যাচ্ছে এটার ভ্যালু পরিবর্তন হয়েছে।

```

> var a = 10;
< undefined
> ++a;
< 11
> a
< 11

```

এখানে প্রথম রিটার্নই করছে চ্যাঞ্জ হওয়া ভ্যালু

সেইম ডিক্রোমেন্টাল অপারেটরের ক্ষেত্রেও। ক্ষেত্রবিশেষে কাজে লাগে এগুলো অনেক, তাই মাথায় থাকা ভালো।

২। কম্পারিজম অপারেটরঃ একটা ভ্যালুর সাথে আরেকটা কম্পেয়ার করার জন্য। দুইটা ভ্যালু সমান কি সমান না, নাকি বড় না ছোটো। এরা রেজাল্ট হিসাবে হয় সত্য **true** অথবা মিথ্যা **false** রিটার্ন করে।

- **==** (ইকুয়াল) — দুইটার ভ্যালু সমান কিনা দেখার জন্যে। সমান হলে সত্য **true** নাইলে মিথ্যা **false** রিটার্ন করবে।

```

> 10 == 10
< true
> var a = 10;
< undefined
> var b = 10;
< undefined
> a == b
< true

```

সমান সমান কি?


```
> 10 == 20
< false
> var a = 10;
< undefined
> var b = 20;
< undefined
> 10 == 20
< false
```

সমান সমান? না...

- **≠** (ইকুয়াল না) — দুইটার ভ্যালু সমান না হলে সত্য **≠** দেখাবে

```
> 10 != 10
< false
> var a = 10;
< undefined
> var b = 10;
< undefined
> a != b
< false
>
```

সমান না তো?

```
> 10 != 20
< true
> var a = 10;
< undefined
> var b = 20;
< undefined
> a != b
< true
```

সমান না তো?

- **(বড়)** — একটা আরেকটা থেকে বড় কিনা সেটা দেখার জন্যে। বড় হলে সত্য নাইলে মিথ্যা

```
> 20 > 10
< true
> var a = 20;
< undefined
> var b = 10;
< undefined
> a > b
< true
```

বড় তো?

```

> 10 > 20
< false
> var a = 10;
< undefined
> var b = 20;
< undefined
> a > b
< false
>

```

বড় তো?

- **lessThan**(ছোট) — একটা আরেকটা থেকে ছোটো কিনা সেটা দেখার জন্যে। ছোটো হলে সত্য **lessThan** নাইলে মিথ্যা **lessThan**

```

> 10 < 20
< true
> var a = 10;
< undefined
> var b = 20;
< undefined
> a < b
< true
> |

```

ছোটো তো?

```
> 20 < 10
< false
> var a = 20;
< undefined
> var b = 10;
< undefined
> a < b
< false
>
```

ছোটো তো?

- **Greater than or equal to (bড় অথবা ইকুয়্যাল)**—একটা আরেকটা থেকে বড় বা সমান সমান কিনা সেটা দেখার জন্যে।
বড় বা সমান হলে সত্য **true** নাইলে মিথ্যা **false**

```
> 10 >= 10
< true
> var a = 10;
< undefined
> var b = 10;
< undefined
> a >= b
< true
>
```

বড় অথবা সমান সমান কি?

```
> 20 >= 10
< true
> var a = 20;
< undefined
> var b = 10;
< undefined
> a >= b
< true
> |
```

বড় অথবা সমান সমান কি?

```
> 10 >= 20
< false
> var a = 10;
< undefined
> var b = 20;
< undefined
> a >= b
< false
```

বড় অথবা সমান সমান কি?

- **lessThanOrEqual**(ছোটো অথবা ইকুয়াল) — একটা আরেকটা থেকে ছোটো বা সমান সমান কিনা সেটা দেখার জন্যে। ছোটো বা সমান হলে সত্য **lessThanOrEqual** নাইলে মিথ্যা **lessThanOrEqual**

```
> 10 <= 10
```

```
< true
```

```
> var a = 10;
```

```
< undefined
```

```
> var b = 10;
```

```
< undefined
```

```
> a <= b
```

```
< true
```

ছোটো অথবা সমান সমান কি?

```
> 10 <= 20
```

```
< true
```

```
> var a = 10;
```

```
< undefined
```

```
> var b = 20;
```

```
< undefined
```

```
> a <= b
```

```
< true
```

ছোটো অথবা সমান সমান কি?

```
> 20 <= 10
< false
> var a = 20;
< undefined
> var b = 10;
< undefined
> a <= b
< false
```

ছোটো অথবা সমান সমান কি?

গুরুত্বপূর্ণঃ এগুলার বাইরেও আরো মোস্ত ইউস কিছু অপারেটর রয়েছে। যেমন `===` এবং `!==` এগুলো ব্যাসিকেলি ইকুয়্যাল `==` বা ইকুয়্যাল না `!=` অপারেটরের মতোই কিন্তু মেইন পার্থক্য হলো ট্রিপল ইকুয়্যাল অপারেটর ভ্যালু দুইটা সমান কিনা সেটা চ্যাক করে এবং সাথে দুইটা একই টাইপের কিনা সেটাও চ্যাক করে। যেখানে ডাবল অপারেটরগুলো শুধুমাত্র ভ্যালু দুইটা সমান কিনা চ্যাক করে, টাইপ চ্যাক করে না।

```
> var a = '10';
< undefined
> var b = 10;
< undefined
> a == b
< true
> a === b
< false
```

এখানে প্রথমটা নাফার মনে হলেও এটা স্ট্রিং, তাই ডাবল অপারেটর ট্রু দেখালেও ট্রিপল ফলস দেখাচ্ছে

```

> var a = '10';
< undefined
> var b = 10;
< undefined
> a != b
< false
> a !== b
< true

```

সেইম কেইস আগেরটার মতোই

এগুলো আসলে অনেক ইউজ হয় এবং জাভাস্ক্রিপ্ট এ অনেকটা ইউনিক তাই ভালো করে মনে রাখা বা এগুলো সম্পর্কে ধারণা রাখা ভালো।

৩। লজিক্যাল অপারেটরঃ তিনরকমের হতে পারেঃ

- **AND** (এন্ড) — সাধারণত দুইটা অপারেন্ড এর মাঝখানে বসে। এবং যদি দুইটা স্টেটমেন্ট সত্য হয় তাহলে পুরোটা সত্য নাইলে যেকোনো একটাও যদি মিথ্যা হয় তাহলে পুরোটাই মিথ্যা। আর যদি দুইটাই মিথ্যা হয়, তাহলেও পুরোটা মিথ্যা।

```

> true && true
< true
> (10 < 20) && (20 < 30)
< true

```

সবগুলো সত্য হলেই তবে পুরোটা সত্য


```
> true && false
< false
> (10 < 20) && (10 > 20)
< false
```

একটা মিথ্যা হলেই পুরোটা মিথ্যা

```
> true && true && true && false && true
< false
```

আপনি চাইলে অনেকগুলোও একসাথে এভাবে ইউজ করতে পারবেন

- **||**(অর)— সাধারণত দুইটা অপারেন্ড এর মাঝখানে বসে। এবং যদি যেকোনো একটা সত্য হয় তাহলে সত্য **||**, দুইটাই যদি সত্য **||** হয় তাহলেও পুরোটা সত্য **||**। আর যদি একমাত্র দুইটা স্টেটেন্টই মিথ্যা **||** হয় তাহলেই পুরোটা মিথ্যা **||** হবে

```
> true || true
< true
> (10 < 20) || (20 < 30)
< true
```

সবগুলো সত্য হলে পুরোটাই সত্য

```
> true || false
< true
> (10 < 20) || (10 > 20)
< true
```

যেকোনো একটা সত্য হলেই সবটা সত্য

```
> false || false
< false
> (10 > 20) || (20 > 30)
< false
```

সবগুলো মিথ্যা হলে তবেই পুরোটা মিথ্যা

```
> false || false || false || true || false
< true
```

এভাবে অনেকগুলো একসাথে ইউজ করা যাবে

- **(নট)** — সাধারণত একটা স্টেটমেন্ট এর পূর্বে বসে। এবং সেটা যদি সত্য হয় তাহলে এটা রিটার্ন করবে মিথ্যা, আর মিথ্যা হলে রিটার্ন করবে সত্য। মানে উল্টো

```
> !true
< false
> !false
< true
```

উল্টো

```
> !(10 > 20)
< true
> !(20 < 10)
< true
```

৪। অ্যাসাইনমেন্ট অপারেটরঃ সমান, যোগ সমান, বিয়োগ সমান, গুণ সমান, ভাগ সমান, মডুলাস সমান।

- সিম্পল অ্যাসাইনমেন্ট অপারেটরঃ আমরা অলরেডি ইউজ করেছি এই অপারেটর। এটা ভ্যারিয়েবলে ভ্যালু অ্যাসাইন করার জন্যে ইউজ করা হয়।

```
> var a = 10;
< undefined
```

- যোগ এবং অ্যাসাইনমেন্ট অপারেটরঃ নিজের সাথে নিজের ভ্যালু যোগ করে সেটাকে আবার নিজের সাথেই অ্যাসাইন করা একই সাথে। ■ এটার ফুল ফর্ম হচ্ছেঃ

```
a = a + 10;
```

```
> var a = 10
< undefined
> a += 10;
< 20
> a
< 20
```

- বিয়োগ এবং অ্যাসাইনমেন্ট অপারেটরঃ এটাও শর্টকাট আগেরটার মতোই। নিজের সাথে নিজের

ভ্যালু বিয়োগ করে সেটাকে আবার নিজের সাথেই অ্যাসাইন করা একই সাথে। এটার ফুল ফর্ম হচ্ছেঃ

```
a = a - 10;
```

```
> var a = 20;
< undefined
> a -= 10;
< 10
> a
< 10
```

- গুণ এবং অ্যাসাইনমেন্ট অপারেটরঃ এটাও শর্টকাট। নিজের সাথে নিজের ভ্যালু গুণ করে সেটাকে আবার নিজের সাথেই অ্যাসাইন করা একই সাথে। এটার ফুল ফর্ম হচ্ছেঃ

```
a = a * 5;
```

```

> var a = 5;
< undefined
> a *= 5
< 25
> a
< 25
>

```

- **■** ভাগ এবং অ্যাসাইনমেন্ট অপারেটরঃ এটাও শর্টকাট। নিজের সাথে নিজের ভ্যালু ভাগ করে সেটাকে আবার নিজের সাথেই অ্যাসাইন করা একই সাথে। **■** এটার ফুল ফর্ম হচ্ছেঃ

```
a = a / 2;
```

```

> var a = 10;
< undefined
> a /= 2;
< 5
> a
< 5

```

- **■** ভাগশেষ এবং অ্যাসাইনমেন্ট অপারেটরঃ এটাও আরেকটা শর্টকাট। নিজের সাথে নিজের ভ্যালু মড করে সেটাকে আবার নিজের সাথেই অ্যাসাইন করা একই সাথে। **■** এটার ফুল ফর্ম হচ্ছেঃ

```
a = a % 5;
```

```
> var a = 13;
```

```
< undefined
```

```
> a %= 5;
```

```
< 3
```

```
> a
```

```
< 3
```

৫। কন্ডিশনাল/টার্নারি অপারেটর: `?` : এটাও আরেকটা শর্টকাট। উদাহরন দেখলে বুঝতে পারবেনঃ

যদি `a > b` হয় তবে `100` অন্যথায় `200`।

```
> var a = 10;
```

```
< undefined
```

```
> var b = 20;
```

```
< undefined
```

```
> var c = a > b ? 100 : 200;
```

```
< undefined
```

```
> c
```

```
< 200
```

৬। অন্যান্য অপারেটরঃ

- **typeof** অপারেটরঃ আমরা আগের পর্বেও এটা ইউজ করেছি। এটাও আসলে একটা অপারেটর। ডাটার টাইপ বের করতে ইউজ করা হয়। আপনি চাইলে এভাবে [redacted] ফাষ্ট ব্র্যাকেটস এর ভিতরে, বা এভাবেও [redacted] লিখতে পারবেন। এটা সবসময় ভ্যালু যে টাইপের সেটাই স্ট্রিং হিসাবে রিটার্ন করবেঃ

```
> var a = 10;
< undefined
> typeof a
< "number"
> typeof(a)
< "number"
>
```

typeof কোনটা কিভাবে রিটার্ন করেঃ

Type	String Returned by typeof
Number	"number"
String	"string"
Boolean	"boolean"
Object	"object"
Function	"function"
Undefined	"undefined"
Null	"object"

আজকের পর্ব এইটুকুই। ভালো থাকবেন, সবাইকে ভালো রাখবেন। আর মনে রাখবেন প্র্যাক্টিসের উপর

কিছু নাই। আপনি উদাহরণ গুলো যতরকমভাবে পারবেন প্র্যাক্টিস করবেন। আপনার নিজের থেকেও প্র্যাক্টিস করবেন যতভাবে পারেন। কোনো সমস্যা হলে গুগল তো আছেই। আর আমাকে ইমেইল করুন বা এখানে কমেন্ট করতে পারেনঃ

Email: zonayedpca@gmail.com

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

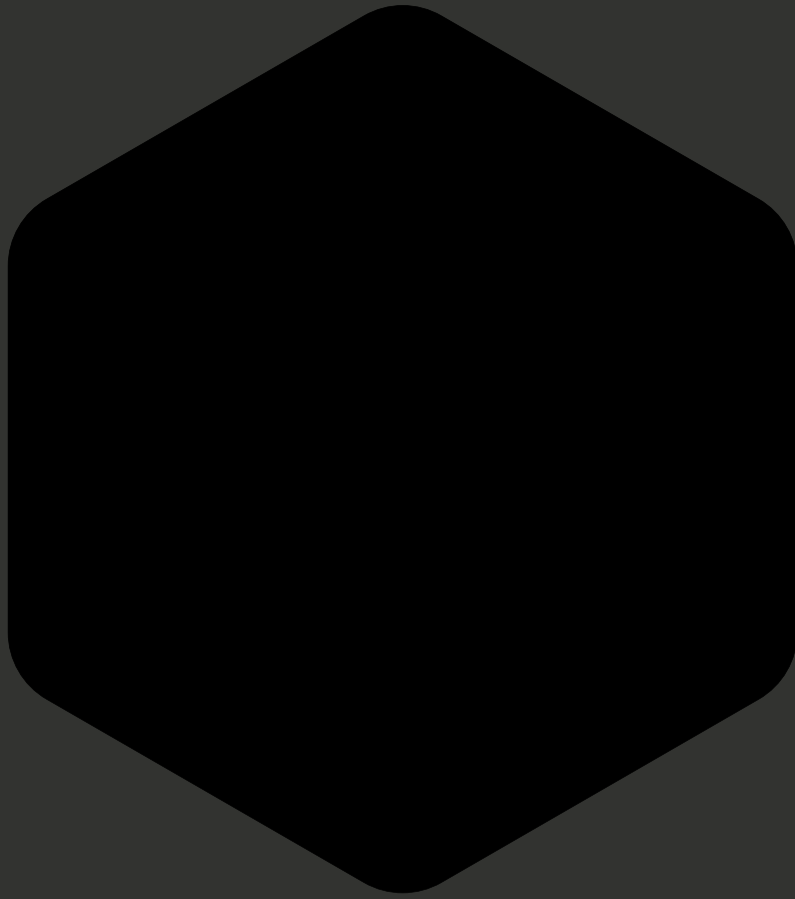
রিসোর্সঃ

এমডিএন ডকুমেন্টেশন

ইউডেমি কোর্স

Eloquent JavaScript

You Don't Know JS



♥ এর সাথে ডেভেলপ করেছে **জুনায়েদ আহমেদ**