JavaScript ES6 & API Complete Cheatsheet

🔹 ES6 Basics

var / let / const পার্থক্য

- `var` = function scoped, redeclare হয়
- `let` = block scoped, redeclare হয় না
- `const` = block scoped, মান পরিবর্তন করা যায় না

```
var name = "Rahim";
let age = 20;
const country = "Bangladesh";
```

Default Parameter
ফাংশনে প্যারামিটার না দিলে ডিফল্ট ভ্যালু দেওয়া যায়

```
function greet(name = "Guest") {
  console.log(`Hello, ${name}`);
}
greet(); // Hello, Guest
```

Template String
ব্যাকটিক ` ` দিয়ে মাল্টিলাইন ও ডাইনামিক স্ট্রিং

```
const user = "Rahim";
const message = `Hi ${user},
Welcome to our website!`;
```

Arrow Function

```
const add = (a, b) => a + b;
const greet = () => console.log("Hi");
```

Spread Operator
অ্যারে কপি, মার্জ, ম্যাক্স ভ্যালু বের করা

```
const arr = [1,2,3];
const newArr = [...arr, 4,5];
const max = Math.max(...arr);
```

Destructuring

```
const user = {name: 'Rahim', age: 20};
const {name, age} = user;

const nums = [1,2,3];
const [first, second] = nums;
```

Object Methods

```
const obj = {x:1,y:2};
Object.keys(obj);    // ['x','y']
Object.values(obj);  // [1,2]
Object.entries(obj); // [['x',1],['y',2]]
```

Object Loop

```
for (let key in obj) console.log(key, obj[key]);
for (let [key,value] of Object.entries(obj)) console.log(key,value);
```

🔹 More ES6

Optional Chaining
নেস্টেড অবজেক্টে নিরাপদে এক্সেস

```
const user = {address:{city:"Dhaka"}};
console.log(user.address?.city); // Dhaka
```

map / forEach / filter / find

- `map` = নতুন অ্যারে রিটার্ন করে
- `forEach` = শুধু লুপ, রিটার্ন দেয় না
- `filter` = কন্ডিশন মিলে যেগুলো রাখে
- `find` = প্রথম মিলে যাওয়া আইটেম

```
const numbers = [1,2,3,4];
const doubled = numbers.map(n => n*2);
const evens = numbers.filter(n => n%2===0);
const firstEven = numbers.find(n => n%2===0);
numbers.forEach(n => console.log(n));
```

🔹 Common JS Concepts

Primitive vs Non-Primitive

- Primitive: number, string, boolean, null, undefined
- Non-Primitive: object, array, function

Null vs Undefined
`undefined` = মান সেট হয়নি
`null` = ইচ্ছা করে খালি

Truthy / Falsy
Falsy: `0`, `""`, `null`, `undefined`, `false`, `NaN`
Truthy: বাকি সব

== vs ===

- `==` টাইপ কনভার্ট করে কমপেয়ার করে

- `===` টাইপ কনভার্ট না করে কমপেয়ার করে

Hoisting
ভ্যারিয়েবল/ফাংশন আগে থেকেই মেমোরিতে যায়, কিন্তু `let`/`const` টেম্পোরাল ডেড জোনে থাকে

Closure
একটা ফাংশন অন্য ফাংশনের ভ্যারিয়েবল মনে রাখে

```
function outer(){
  let count=0;
  return function inner(){
    count++;
    return count;
  }
}
const counter = outer();
console.log(counter()); //1
console.log(counter()); //2
```

Pass by value/reference
প্রিমিটিভ = value দিয়ে
অবজেক্ট/অ্যারে = reference দিয়ে

🔹 API Basics

API কী?
ইন্টারনেটের মাধ্যমে ডাটা নেওয়া/পাঠানো যায় এমন ইন্টারফেস

JSON parse / stringify

```
const jsonData = '{"name":"Rahim"}';
const obj = JSON.parse(jsonData);
const backToJson = JSON.stringify(obj);
```

GET Data & Display

```
fetch('https://jsonplaceholder.typicode.com/posts')
```

```
  .then(res=>res.json())
  .then(data=>console.log(data));
```

CRUD (GET/POST/PATCH/DELETE)

```
// POST
fetch(url,{
  method:'POST',
  headers:{'Content-Type':'application/json'},
  body:JSON.stringify({title:'New'})
})
```

async/await

```
async function loadData(){
  try {
    const res = await fetch(url);
    const data = await res.json();
    console.log(data);
  } catch(err){
    console.error(err);
  }
}
```

🔹 Debugging JS

Console Methods

```
console.log("debug info");
console.error("error message");
console.warn("warning");
```

Breakpoints
DevTools > Sources ট্যাবে গিয়ে কোডের লাইনে ক্লিক করে সেট করতে হয়

Error Handling

```
try {
  // risky code
} catch (error) {
  console.error(error);
} finally {
  console.log("Always runs");
}
```

🔹 Local Storage

```
// set
localStorage.setItem('name','Rahim');
// get
const name = localStorage.getItem('name');
// object
const obj = {x:1};
localStorage.setItem('obj',JSON.stringify(obj));
const saved = JSON.parse(localStorage.getItem('obj'));
```

🔹 Asynchronous JS & Event Loop

- JS single-threaded
- Event loop async কাজগুলো হ্যান্ডেল করে

Promise

```
const myPromise = new Promise((resolve,reject)=>{
  setTimeout(()=>resolve("Done"),1000);
});
myPromise.then(res=>console.log(res));
```

setInterval / setTimeout

```
setTimeout(()=>console.log("after 1s"),1000);
const id = setInterval(()=>console.log("every 2s"),2000);
```

clearInterval(id);

🎯 Quick Tips & Best Practices

🔗ES6 Best Practices

- Use const by default, let when reassignment needed
- Prefer arrow functions for short operations
- Use template literals for string interpolation
- Destructure objects and arrays when appropriate
- Use spread operator for copying arrays/objects

🔗API Best Practices

- Always handle errors with try/catch or .catch()
- Check response status before processing data
- Use async/await for cleaner code
- Implement loading states in UI
- Cache data when appropriate

🔗Performance Tips

- Use array methods (map, filter, reduce) instead of loops when possible
- Avoid nested loops when possible
- Use event delegation for multiple similar elements
- Debounce API calls for search functionality

🔗Common Pitfalls to Avoid

- Don't forget await with async functions
- Don't mutate original arrays (use spread or slice)
- Don't compare objects with == or === directly
- Don't forget to handle promise rejections
- Don't store sensitive data in localStorage