JavaScript Alapok

Bevezetés

A JavaScript egy dinamikus, interpretált programozási nyelv, amelyet elsősorban weboldalak interaktívvá tételére használnak. 1995-ben Brendan Eich fejlesztette ki mindössze 10 nap alatt.

Alapvető típusok

Primitív típusok

1. **String** - Szöveges adatok
   - Példa: `"Hello World"`, `'JavaScript'`
   - Template literal: `` `Hello ${name}` ``
2. **Number** - Számok
   - Egész számok: `42`, `-17`
   - Lebegőpontos: `3.14`, `2.718`
   - Speciális értékek: `Infinity`, `NaN`
3. **Boolean** - Logikai értékek
   - Csak két érték: `true` és `false`
4. **Undefined** - Nem definiált érték
   - Automatikus alapérték inicializálatlan változóknál
5. **Null** - Üres érték
   - Szándékosan üres objektum referencia
6. **Symbol** - Egyedi azonosító (ES6+)
7. **BigInt** - Nagy egész számok (ES2020+)

Változók deklarálása

let

```
let age = 25;
age = 26; // Megváltoztatható
```

const

```
const PI = 3.14159;
// PI = 3.14; // HIBA! Nem módosítható
```

var (kerülendő)

```
var oldStyle = "ne használd";
```

Operátorok

Aritmetikai operátorok

- Összeadás: `+`
- Kivonás: `-`
- Szorzás: `*`
- Osztás: `/`
- Maradék: `%`
- Hatványozás: `**`

Összehasonlító operátorok

- Egyenlő (érték): `==`
- Szigorúan egyenlő (érték és típus): `===`
- Nem egyenlő: `!=`
- Szigorúan nem egyenlő: `!==`
- Nagyobb: `>`
- Kisebb: `<`

Logikai operátorok

- ÉS: `&&`
- VAGY: `||`
- NEGÁLÁS: `!`

Vezérlési szerkezetek

if-else

```
if (age >= 18) {
    console.log("Felnőtt");
} else if (age >= 13) {
    console.log("Tizenéves");
} else {
    console.log("Gyerek");
}
```

switch

```
switch(day) {
    case "Hétfő":
        console.log("Hét eleje");
        break;
```

```
    case "Péntek":
        console.log("Majdnem hétvége");
        break;
    default:
        console.log("Hét közepe");
}
```

for ciklus

```
for (let i = 0; i < 10; i++) {
    console.log(i);
}
```

while ciklus

```
let i = 0;
while (i < 10) {
    console.log(i);
    i++;
}
```

Függvények

Klasszikus függvény

```
function greet(name) {
    return `Hello, ${name}!`;
}
```

Arrow function (ES6+)

```
const greet = (name) => `Hello, ${name}!`;
```

Alapértelmezett paraméterek

```
function multiply(a, b = 1) {
    return a * b;
}
```

Tömbök (Arrays)

Létrehozás

```
const fruits = ["alma", "banán", "cseresznye"];
const numbers = [1, 2, 3, 4, 5];
```

Gyakori metódusok

- `push()` - Hozzáad a végére
- `pop()` - Eltávolít a végéről
- `shift()` - Eltávolít az elejéről
- `unshift()` - Hozzáad az elejére
- `slice()` - Részletet vág ki
- `splice()` - Módosít elemeket
- `map()` - Átalakítja az elemeket
- `filter()` - Szűri az elemeket
- `reduce()` - Egyetlen értékre redukál

Példák

```
const doubled = numbers.map(n => n * 2);
const evens = numbers.filter(n => n % 2 === 0);
const sum = numbers.reduce((acc, n) => acc + n, 0);
```

Objektumok

Létrehozás

```
const person = {
    name: "János",
    age: 30,
    city: "Budapest",
    greet: function() {
        console.log(`Szia, ${this.name} vagyok!`);
    }
};
```

Hozzáférés

```
console.log(person.name); // Pont jelöléssel
console.log(person["age"]); // Szögletes zárójellel
```

Destructuring

```
const { name, age } = person;
console.log(name); // "János"
```

Modern JavaScript (ES6+)

Template literals

```
const name = "Világ";
console.log(`Hello, ${name}!`);
```

Spread operator

```
const arr1 = [1, 2, 3];
const arr2 = [...arr1, 4, 5, 6];
```

Rest paraméterek

```
function sum(...numbers) {
    return numbers.reduce((a, b) => a + b, 0);
}
```

Promises

```
const myPromise = new Promise((resolve, reject) => {
    setTimeout(() => resolve("Kész!"), 1000);
});

myPromise.then(result => console.log(result));
```

Async/Await

```
async function fetchData() {
    try {
        const response = await fetch('api/data');
        const data = await response.json();
        return data;
    } catch (error) {
        console.error(error);
    }
}
```

Hibakezelés

```
try {
    // Kockázatos kód
    throw new Error("Valami hiba történt");
} catch (error) {
    console.error("Hiba:", error.message);
} finally {
    console.log("Mindig lefut");
}
```

Összefoglalás

A JavaScript egy sokoldalú nyelv, amely folyamatosan fejlődik. Az alapok elsajátítása után képes leszel: - Interaktív weboldalakat készíteni - Szerveroldali

alkalmazásokat írni (Node.js) - Mobil appokat fejleszteni (React Native) - Desktop alkalmazásokat létrehozni (Electron)

A gyakorlás a kulcs a JavaScript elsajátításához!