**### 4. óra: Változók és láthatóságuk**

**- Változók deklarálása: `var`, `let`, és `const`**

**- Láthatóság és scope**

**- Gyakorlati feladatok: különböző típusú változók létrehozása és használata különböző scope-okban.**

**Óraterv: Változók és láthatóságuk**

**4. óra célja:** A diákok megértik, hogyan lehet változókat deklarálni és miért fontos a megfelelő változótípus (var, let, const) kiválasztása. Megismerkednek a változók láthatóságával (scope) és a különböző scope-ok közötti különbségekkel. Az óra végére a diákok képesek lesznek különböző típusú változókat deklarálni és használni különböző scope-okban.

**Időtartam:** 45 perc

**Óra Script**

**0-5 perc: Bevezetés a változók típusainak fontosságába**

* **Tanár mondja:** „A változók segítenek abban, hogy adatokat tároljunk és kezeljünk a programunkban. De amikor változókat hozunk létre, fontos eldönteni, hogy milyen típusú változót használunk – ezt befolyásolja, hogy az adott adatot hol és hogyan szeretnénk elérni.”
* **Példa:** „Gondoljatok arra, hogy egy tábla készítésekor figyelembe vesszük, hogy milyen anyagból készül: fa, műanyag vagy fém. Ugyanez a logika érvényes a változókra is – attól függően, hogy mit akarunk velük tenni, eltérő típusú deklarációkat választunk.”

**5-15 perc: A változók típusai (var, let, const)**

1. **var változó:** „A var kulcsszóval deklarált változók a JavaScript régebbi verzióiban voltak népszerűek, és általánosan használták. Azonban a scope-kezelésük néha problémát okozhat.”
   * **Kód konzolon:**

javascript

Copy code

var szam = 5;

console.log(szam);

1. **let változó:** „A let kulcsszóval deklarált változók csak azon a területen elérhetőek, ahol létre lettek hozva. Ezáltal könnyebben szabályozhatjuk, hogy a változó csak bizonyos részeken legyen elérhető.”
   * **Kód konzolon:**

javascript

Copy code

let nev = 'Anna';

console.log(nev);

1. **const változó:** „A const kulcsszóval deklarált változók értéke nem változtatható meg. Az állandókat és a végleges értékeket jellemzően const-tal deklaráljuk.”
   * **Kód konzolon:**

javascript

Copy code

const PI = 3.14;

console.log(PI);

1. **Feladat:** „Próbáljatok meg létrehozni három változót a var, let és const használatával: az egyik legyen egy név, a másik egy szám, a harmadik pedig egy állandó érték. Írjátok ki ezeket a konzolra.”

**15-30 perc: Változók láthatósága és scope-ok**

1. **Globális és lokális scope-ok:** „A változók érvényességi területe (scope) meghatározza, hogy a változókat hol érhetjük el. A globális scope-ban létrehozott változók mindenhol elérhetők a programban, míg a lokális scope-ban létrehozott változók csak az adott blokkon belül használhatók.”
   * **Példa:**

javascript

Copy code

var globalisValtozo = 'Ez globális';

function teszt() {

let lokalValtozo = 'Ez lokális';

console.log(globalisValtozo); // Elérhető

console.log(lokalValtozo); // Elérhető

}

teszt();

console.log(globalisValtozo); // Elérhető

console.log(lokalValtozo); // Nem elérhető, hibát dob

1. **Blokk scope és funkció scope:** „A let és const blokk scope-ú, míg a var funkció scope-ú. Ez azt jelenti, hogy a var-ral deklarált változók láthatósága szélesebb, míg a let és const változók egy blokkban maradnak.”
   * **Példa:**

javascript

Copy code

if (true) {

var a = 'var változó';

let b = 'let változó';

const c = 'const változó';

}

console.log(a); // Elérhető

console.log(b); // Nem elérhető, hibát dob

console.log(c); // Nem elérhető, hibát dob

1. **Feladat:** „Hozzatok létre egy funkciót, amelyben több változót deklaráltok var, let és const kulcsszóval. Próbáljátok ki, hogy a változókat elérhetitek-e a funkción kívül vagy a funkcióban.”

**30-40 perc: Gyakorlati példák változók használatára különböző scope-okban**

1. **Feladat:** „Hozzatok létre egy globális változót és egy lokális változót ugyanazzal a névvel egy funkción belül, és írjátok ki mindkettőt. Figyeljétek meg, hogy a globális változó nem érhető el a funkció scope-ján belül, ha helyette ugyanazzal a névvel létrehoztok egy lokális változót.”
   * **Példa kód:**

javascript

Copy code

let ertek = 'globális érték';

function teszt() {

let ertek = 'lokális érték';

console.log(ertek); // lokális érték

}

teszt();

console.log(ertek); // globális érték

1. **Értékek változtatása const-tal:** „Próbáljátok meg, hogy a const-tal deklarált változónak új értéket adtok. Ez hibát fog dobni, hiszen egy const érték nem módosítható.”
   * **Kód példa:**

javascript

Copy code

const PI = 3.14;

PI = 3.14159; // Hibát dob, mert a const értéke nem módosítható

**40-45 perc: Összefoglalás és kérdések**

* **Tanár mondja:** „Ma megismertük a JavaScript különböző változótípusait, mint a var, let és const, valamint azt, hogy a különböző típusok hogyan befolyásolják a változók láthatóságát a programban. Van kérdésetek ezzel kapcsolatban?”
* **Házi feladat gyakorlásra:** „Próbáljátok meg írni egy programot, ahol különböző scope-okban használjátok a változókat, és nézzétek meg, hogyan viselkednek a különböző típusok. Teszteljétek, hogy mely változókat érhetitek el a különböző részeken!”

**Óra összefoglalás**

Az óra során a diákok megtanulták, hogyan kell változókat deklarálni, és miért fontos a megfelelő típus kiválasztása (var, let, const). Megértették a scope fogalmát, és kipróbálták, hogyan érhetnek el vagy nem érhetnek el különböző scope-okban deklarált változókat.

### Óraterv: Változók és láthatóságuk

\*\*4. óra célja:\*\* A diákok megértik, hogyan lehet változókat deklarálni és miért fontos a megfelelő változótípus (var, let, const) kiválasztása. Megismerkednek a változók láthatóságával (scope) és a különböző scope-ok közötti különbségekkel. Az óra végére a diákok képesek lesznek különböző típusú változókat deklarálni és használni különböző scope-okban.

\*\*Időtartam:\*\* 45 perc

---

#### Óra Script

\*\*0-5 perc: Bevezetés a változók típusainak fontosságába\*\*

- \*\*Tanár mondja:\*\* „A változók segítenek abban, hogy adatokat tároljunk és kezeljünk a programunkban. De amikor változókat hozunk létre, fontos eldönteni, hogy milyen típusú változót használunk – ezt befolyásolja, hogy az adott adatot hol és hogyan szeretnénk elérni.”

- \*\*Példa:\*\* „Gondoljatok arra, hogy egy tábla készítésekor figyelembe vesszük, hogy milyen anyagból készül: fa, műanyag vagy fém. Ugyanez a logika érvényes a változókra is – attól függően, hogy mit akarunk velük tenni, eltérő típusú deklarációkat választunk.”

\*\*5-15 perc: A változók típusai (var, let, const)\*\*

1. \*\*var változó:\*\* „A `var` kulcsszóval deklarált változók a JavaScript régebbi verzióiban voltak népszerűek, és általánosan használták. Azonban a scope-kezelésük néha problémát okozhat.”

- \*\*Kód konzolon:\*\*

```javascript

var szam = 5;

console.log(szam);

```

2. \*\*let változó:\*\* „A `let` kulcsszóval deklarált változók csak azon a területen elérhetőek, ahol létre lettek hozva. Ezáltal könnyebben szabályozhatjuk, hogy a változó csak bizonyos részeken legyen elérhető.”

- \*\*Kód konzolon:\*\*

```javascript

let nev = 'Anna';

console.log(nev);

```

3. \*\*const változó:\*\* „A `const` kulcsszóval deklarált változók értéke nem változtatható meg. Az állandókat és a végleges értékeket jellemzően `const`-tal deklaráljuk.”

- \*\*Kód konzolon:\*\*

```javascript

const PI = 3.14;

console.log(PI);

```

4. \*\*Feladat:\*\* „Próbáljatok meg létrehozni három változót a `var`, `let` és `const` használatával: az egyik legyen egy név, a másik egy szám, a harmadik pedig egy állandó érték. Írjátok ki ezeket a konzolra.”

\*\*15-30 perc: Változók láthatósága és scope-ok\*\*

1. \*\*Globális és lokális scope-ok:\*\* „A változók érvényességi területe (scope) meghatározza, hogy a változókat hol érhetjük el. A globális scope-ban létrehozott változók mindenhol elérhetők a programban, míg a lokális scope-ban létrehozott változók csak az adott blokkon belül használhatók.”

- \*\*Példa:\*\*

```javascript

var globalisValtozo = 'Ez globális';

function teszt() {

let lokalValtozo = 'Ez lokális';

console.log(globalisValtozo); // Elérhető

console.log(lokalValtozo); // Elérhető

}

teszt();

console.log(globalisValtozo); // Elérhető

console.log(lokalValtozo); // Nem elérhető, hibát dob

```

2. \*\*Blokk scope és funkció scope:\*\* „A `let` és `const` blokk scope-ú, míg a `var` funkció scope-ú. Ez azt jelenti, hogy a `var`-ral deklarált változók láthatósága szélesebb, míg a `let` és `const` változók egy blokkban maradnak.”

- \*\*Példa:\*\*

```javascript

if (true) {

var a = 'var változó';

let b = 'let változó';

const c = 'const változó';

}

console.log(a); // Elérhető

console.log(b); // Nem elérhető, hibát dob

console.log(c); // Nem elérhető, hibát dob

```

3. \*\*Feladat:\*\* „Hozzatok létre egy funkciót, amelyben több változót deklaráltok `var`, `let` és `const` kulcsszóval. Próbáljátok ki, hogy a változókat elérhetitek-e a funkción kívül vagy a funkcióban.”

\*\*30-40 perc: Gyakorlati példák változók használatára különböző scope-okban\*\*

1. \*\*Feladat:\*\* „Hozzatok létre egy globális változót és egy lokális változót ugyanazzal a névvel egy funkción belül, és írjátok ki mindkettőt. Figyeljétek meg, hogy a globális változó nem érhető el a funkció scope-ján belül, ha helyette ugyanazzal a névvel létrehoztok egy lokális változót.”

- \*\*Példa kód:\*\*

```javascript

let ertek = 'globális érték';

function teszt() {

let ertek = 'lokális érték';

console.log(ertek); // lokális érték

}

teszt();

console.log(ertek); // globális érték

```

2. \*\*Értékek változtatása const-tal:\*\* „Próbáljátok meg, hogy a `const`-tal deklarált változónak új értéket adtok. Ez hibát fog dobni, hiszen egy `const` érték nem módosítható.”

- \*\*Kód példa:\*\*

```javascript

const PI = 3.14;

PI = 3.14159; // Hibát dob, mert a const értéke nem módosítható

```

\*\*40-45 perc: Összefoglalás és kérdések\*\*

- \*\*Tanár mondja:\*\* „Ma megismertük a JavaScript különböző változótípusait, mint a `var`, `let` és `const`, valamint azt, hogy a különböző típusok hogyan befolyásolják a változók láthatóságát a programban. Van kérdésetek ezzel kapcsolatban?”

- \*\*Házi feladat gyakorlásra:\*\* „Próbáljátok meg írni egy programot, ahol különböző scope-okban használjátok a változókat, és nézzétek meg, hogyan viselkednek a különböző típusok. Teszteljétek, hogy mely változókat érhetitek el a különböző részeken!”

---

### Óra összefoglalás

Az óra során a diákok megtanulták, hogyan kell változókat deklarálni, és miért fontos a megfelelő típus kiválasztása (`var`, `let`, `const`). Megértették a scope fogalmát, és kipróbálták, hogyan érhetnek el vagy nem érhetnek el különböző scope-okban deklarált változókat.