**### 3. óra: Összetett adattípusok és műveletek**

**- Tömbök (Array) alapjai**

**- Object adattípus és alapvető tulajdonságai**

**- Map és Set rövid bemutatása**

**- Gyakorlati feladatok: egyszerű tömbök és objektumok létrehozása és módosítása.**

**Óraterv: Összetett adattípusok és műveletek**

**3. óra célja:** A diákok megismerkednek a JavaScript összetett adattípusaival, mint a tömbök (Array) és objektumok (Object), valamint röviden áttekintjük a Map és Set adattípusokat. Az óra végére a diákok képesek lesznek egyszerű tömböket és objektumokat létrehozni, módosítani és hozzáférni azok elemeihez.

**Időtartam:** 45 perc

**Óra Script**

**0-5 perc: Bevezetés az összetett adattípusok fogalmába**

* **Tanár mondja:** „Eddig egyszerű adattípusokat tanultunk, mint a számok és szövegek. Ma összetettebb adattípusokat fogunk megismerni, amelyek lehetővé teszik több érték tárolását egy struktúrában. Ezekkel az adattípusokkal csoportosítani tudjuk az adatokat.”
* **Példák felhozása:** „Képzeljétek el, hogy egy bevásárlólistát vagy egy osztálynévsort kell tárolnunk. Az ilyen adatok esetén egy tömböt vagy egy objektumot használhatunk.”

**5-20 perc: Tömbök (Array) alapjai**

1. **Tömb fogalma:** „A tömb (Array) egy olyan adattípus, amelyben több értéket tárolhatunk egy helyen, sorrendben. Ezeket az értékeket indexekkel érhetjük el, ahol az első elem indexe mindig 0.”
   * **Példa:** „Egy tömbben így tárolhatjuk a bevásárlólistát: ['alma', 'banán', 'tej']”
2. **Tömb létrehozása és elemek elérése:**
   * **Kód konzolon:** „Próbáljátok ki:

javascript

Copy code

let bevásárlólista = ['alma', 'banán', 'tej'];

console.log(bevásárlólista);

console.log(bevásárlólista[0]); // Első elem: 'alma'

console.log(bevásárlólista[2]); // Harmadik elem: 'tej'

1. **Elemek hozzáadása és módosítása:** „Tömbhöz új elemeket is hozzáadhatunk, és meglévő elemeket módosíthatunk.”
   * **Kód konzolon:** „Próbáljátok ki:

javascript

Copy code

bevásárlólista.push('kenyér'); // Hozzáadjuk a 'kenyér' elemet

console.log(bevásárlólista);

bevásárlólista[1] = 'narancs'; // Módosítjuk a második elemet

console.log(bevásárlólista);

1. **Feladat:** „Hozzatok létre egy tömböt a kedvenc ételeitekkel, és írjátok ki a konzolra. Adjátok hozzá egy új kedvenc étel nevét a tömb végéhez, majd módosítsátok az egyik elemet.”

**20-30 perc: Objektum adattípus és alapvető tulajdonságai**

1. **Objektum fogalma:** „Az objektum (Object) egy összetett adattípus, amely kulcs-érték párokat tárol. Segítségével az adatokat könnyebben rendszerezhetjük, különösen akkor, ha különféle tulajdonságokat kell tárolnunk egy adott dologhoz kapcsolódóan.”
   * **Példa:** „Képzeljük el, hogy tárolni szeretnénk egy személy adatait: neve, életkora, foglalkozása.”
   * **Kód konzolon:** „Létrehozhatunk egy objektumot így:

javascript

Copy code

let szemely = {

nev: 'János',

kor: 25,

foglalkozas: 'programozó'

};

console.log(szemely);

1. **Objektum tulajdonságainak elérése és módosítása:** „Egy objektum tulajdonságait elérhetjük a pont (.) szintaxissal vagy a szögletes zárójeles ([]) módszerrel.”
   * **Kód konzolon:** „Próbáljátok ki:

javascript

Copy code

console.log(szemely.nev); // 'János'

szemely.kor = 26; // Módosítjuk az életkort

console.log(szemely.kor); // 26

szemely['foglalkozas'] = 'tanár';

console.log(szemely.foglalkozas); // 'tanár'

1. **Feladat:** „Készítsetek egy objektumot, amely leírja egy kedvenc filmeteket: legyen benne a címe, a rendező neve és az évszám. Írjátok ki az objektumot a konzolra, majd módosítsátok a rendező nevét és az évszámot.”

**30-40 perc: Map és Set rövid bemutatása**

1. **Map:** „A Map adattípus hasonló az objektumhoz, de bármilyen adattípust használhatunk kulcsként, nem csak szöveget.”
   * **Kód példa:** „Létrehozhatunk egy Map-et így:

javascript

Copy code

let tanarok = new Map();

tanarok.set('informatika', 'Kovács Péter');

tanarok.set('matematika', 'Nagy Éva');

console.log(tanarok);

1. **Set:** „A Set adattípus olyan, mint egy tömb, de csak egyedi értékeket tárol.”
   * **Kód példa:** „Létrehozhatunk egy Set-et így:

javascript

Copy code

let egyediSzamok = new Set([1, 2, 3, 3, 4]);

console.log(egyediSzamok); // Csak egyszer tárolja a 3-at

1. **Feladat:** „Próbáljátok ki, hogy létrehoztok egy Map-et a kedvenc tantárgyaitokkal és a hozzájuk rendelt tanárokkal, majd egy Set-et a kedvenc számaitokkal.”

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

* **Tanár mondja:** „Ma megismerkedtünk az összetett adattípusokkal: tömbökkel, objektumokkal, valamint a Map és Set használatával. Van kérdésetek ezekkel az adattípusokkal kapcsolatban?”
* **Házi feladat gyakorlásra:** „Gyakoroljátok otthon az objektumok és tömbök létrehozását, és próbáljatok meg egy saját Map-et vagy Set-et készíteni! Ezeket majd a következő órán átnézzük.”

### Óraterv: Összetett adattípusok és műveletek

\*\*3. óra célja:\*\* A diákok megismerkednek a JavaScript összetett adattípusaival, mint a tömbök (Array) és objektumok (Object), valamint röviden áttekintjük a Map és Set adattípusokat. Az óra végére a diákok képesek lesznek egyszerű tömböket és objektumokat létrehozni, módosítani és hozzáférni azok elemeihez.

\*\*Időtartam:\*\* 45 perc

---

#### Óra Script

\*\*0-5 perc: Bevezetés az összetett adattípusok fogalmába\*\*

- \*\*Tanár mondja:\*\* „Eddig egyszerű adattípusokat tanultunk, mint a számok és szövegek. Ma összetettebb adattípusokat fogunk megismerni, amelyek lehetővé teszik több érték tárolását egy struktúrában. Ezekkel az adattípusokkal csoportosítani tudjuk az adatokat.”

- \*\*Példák felhozása:\*\* „Képzeljétek el, hogy egy bevásárlólistát vagy egy osztálynévsort kell tárolnunk. Az ilyen adatok esetén egy tömböt vagy egy objektumot használhatunk.”

\*\*5-20 perc: Tömbök (Array) alapjai\*\*

1. \*\*Tömb fogalma:\*\* „A tömb (Array) egy olyan adattípus, amelyben több értéket tárolhatunk egy helyen, sorrendben. Ezeket az értékeket indexekkel érhetjük el, ahol az első elem indexe mindig 0.”

- \*\*Példa:\*\* „Egy tömbben így tárolhatjuk a bevásárlólistát: `['alma', 'banán', 'tej']`”

2. \*\*Tömb létrehozása és elemek elérése:\*\*

- \*\*Kód konzolon:\*\* „Próbáljátok ki:

```javascript

let bevásárlólista = ['alma', 'banán', 'tej'];

console.log(bevásárlólista);

console.log(bevásárlólista[0]); // Első elem: 'alma'

console.log(bevásárlólista[2]); // Harmadik elem: 'tej'

```

3. \*\*Elemek hozzáadása és módosítása:\*\* „Tömbhöz új elemeket is hozzáadhatunk, és meglévő elemeket módosíthatunk.”

- \*\*Kód konzolon:\*\* „Próbáljátok ki:

```javascript

bevásárlólista.push('kenyér'); // Hozzáadjuk a 'kenyér' elemet

console.log(bevásárlólista);

bevásárlólista[1] = 'narancs'; // Módosítjuk a második elemet

console.log(bevásárlólista);

```

4. \*\*Feladat:\*\* „Hozzatok létre egy tömböt a kedvenc ételeitekkel, és írjátok ki a konzolra. Adjátok hozzá egy új kedvenc étel nevét a tömb végéhez, majd módosítsátok az egyik elemet.”

\*\*20-30 perc: Objektum adattípus és alapvető tulajdonságai\*\*

1. \*\*Objektum fogalma:\*\* „Az objektum (Object) egy összetett adattípus, amely kulcs-érték párokat tárol. Segítségével az adatokat könnyebben rendszerezhetjük, különösen akkor, ha különféle tulajdonságokat kell tárolnunk egy adott dologhoz kapcsolódóan.”

- \*\*Példa:\*\* „Képzeljük el, hogy tárolni szeretnénk egy személy adatait: neve, életkora, foglalkozása.”

- \*\*Kód konzolon:\*\* „Létrehozhatunk egy objektumot így:

```javascript

let szemely = {

nev: 'János',

kor: 25,

foglalkozas: 'programozó'

};

console.log(szemely);

```

2. \*\*Objektum tulajdonságainak elérése és módosítása:\*\* „Egy objektum tulajdonságait elérhetjük a pont (`.`) szintaxissal vagy a szögletes zárójeles (`[]`) módszerrel.”

- \*\*Kód konzolon:\*\* „Próbáljátok ki:

```javascript

console.log(szemely.nev); // 'János'

szemely.kor = 26; // Módosítjuk az életkort

console.log(szemely.kor); // 26

szemely['foglalkozas'] = 'tanár';

console.log(szemely.foglalkozas); // 'tanár'

```

3. \*\*Feladat:\*\* „Készítsetek egy objektumot, amely leírja egy kedvenc filmeteket: legyen benne a címe, a rendező neve és az évszám. Írjátok ki az objektumot a konzolra, majd módosítsátok a rendező nevét és az évszámot.”

\*\*30-40 perc: Map és Set rövid bemutatása\*\*

1. \*\*Map:\*\* „A Map adattípus hasonló az objektumhoz, de bármilyen adattípust használhatunk kulcsként, nem csak szöveget.”

- \*\*Kód példa:\*\* „Létrehozhatunk egy Map-et így:

```javascript

let tanarok = new Map();

tanarok.set('informatika', 'Kovács Péter');

tanarok.set('matematika', 'Nagy Éva');

console.log(tanarok);

```

2. \*\*Set:\*\* „A Set adattípus olyan, mint egy tömb, de csak egyedi értékeket tárol.”

- \*\*Kód példa:\*\* „Létrehozhatunk egy Set-et így:

```javascript

let egyediSzamok = new Set([1, 2, 3, 3, 4]);

console.log(egyediSzamok); // Csak egyszer tárolja a 3-at

```

3. \*\*Feladat:\*\* „Próbáljátok ki, hogy létrehoztok egy Map-et a kedvenc tantárgyaitokkal és a hozzájuk rendelt tanárokkal, majd egy Set-et a kedvenc számaitokkal.”

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

- \*\*Tanár mondja:\*\* „Ma megismerkedtünk az összetett adattípusokkal: tömbökkel, objektumokkal, valamint a Map és Set használatával. Van kérdésetek ezekkel az adattípusokkal kapcsolatban?”

- \*\*Házi feladat gyakorlásra:\*\* „Gyakoroljátok otthon az objektumok és tömbök létrehozását, és próbáljatok meg egy saját Map-et vagy Set-et készíteni! Ezeket majd a következő órán átnézzük.”

---

### Óra összefoglalás

Az óra során a diákok megtanulták, hogyan kell tömböket és objektumokat létrehozni, módosítani és elérni azok elemeit. Röviden áttekintették a Map és Set adattípusok alapjait is, így átfogó képet kaptak a JavaScript összetett adattípusairól.