**### 13. óra: Eseménykezelés I.**

**- Események alapjai és eseményfigyelők bevezetése (`onClick`, `onLoad`)**

**- Gyakorlati feladatok: egyszerű interaktív elemek létrehozása.**

**13. Óra: Eseménykezelés I.**

**1. Bevezetés az eseménykezelésbe (10 perc)**

* **Tanár mondja:** "Az eseménykezelés a weboldalak interaktivitásának kulcsa. Az események olyan műveletek, amelyeket a felhasználó hajt végre, mint például egy gombra kattintás, egy szövegmező kitöltése vagy az egér mozgatása egy elem fölé. Ezekkel az eseményekkel JavaScript segíségével kapcsolatba léphetünk, és interaktív funkciókat hozhatunk létre."
* **Fontos kiemelni:**
  + Az eseménykezelés lehetővé teszi, hogy dinamikusan reagáljunk a felhasználói műveletekre.
  + Események kezelésével megváltoztathatjuk az oldal viselkedését, tartalmát vagy stílusát.
  + A HTML elemekhez eseménykezelőket kapcsolhatunk, amelyek kódot futtatnak az esemény bekövetkeztekor.

**2. OnClick és OnLoad események bemutatása (15 perc)**

* **Tanár mondja:** "Az eseménykezelésnek két nagyon gyakran használt típusa az onClick és az onLoad. Ezek lehetővé teszik, hogy reagáljunk egy gomb megnyomására, illetve az oldal teljes betöltésére."
* **OnClick magyarázata és példa:** "Az onClick esemény akkor fut le, amikor a felhasználó egy elemre kattint. Ezt egyszerűen beállíthatjuk HTML-ben vagy JavaScript-ben."



<button onclick="alert('Gombra kattintottál!')">Kattints rám!</button>

* **Magyarázat:** Itt a onclick attribútummal közvetlenül a HTML-ben határozzuk meg, hogy mi történjen kattintáskor.

**OnClick esemény JavaScript-ben:**

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

<button id="clickButton">Kattints rám!</button>

<script>

const button = document.getElementById('clickButton');

button.onclick = () => {

alert('Gombra kattintottál!');

};

</script>

* **Magyarázat:**
  + - Az onclick attribútum JavaScript-ben is beállítható.
    - Ez egy másik módja a kattintási esemény kezelésének.
* **OnLoad magyarázata és példa:** "Az onLoad esemény akkor fut le, amikor az oldal teljesen betöltődött. Ezzel biztosíthatjuk, hogy egy adott kód csak akkor fut, amikor minden szükséges erőforrás elérhető."

A képen szöveg, Betűtípus, képernyőkép látható

Automatikusan generált leírás

<body onload="console.log('Az oldal betöltődött!')">

<h1>Üdv az oldalon!</h1>

</body>

* **Magyarázat:** A onload attribútummal közvetlenül a body elemhez kapcsolhatjuk az eseményt.

**OnLoad esemény JavaScript-ben:**

A képen szöveg, Betűtípus, képernyőkép látható

Automatikusan generált leírás

<script>

window.onload = () => {

console.log('Az oldal betöltődött!');

};

</script>



* **Magyarázat:**
* A window.onload azt biztosítja, hogy a kód csak akkor fusson le, amikor az egész oldal (beleértve az összes képet és forrást) betöltődött.

**3. Eseményobjektum megértése (10 perc)**

* **Tanár mondja:** "Minden eseményhez tartozik egy eseményobjektum, amely információkat tartalmaz az eseményről. Ezen keresztül megtudhatjuk, milyen esemény történt, melyik elem indított el egy eseményt, vagy például milyen pozícióban történt egy kattintás."
* **Példa az eseményobjektum használatára:**

A képen szöveg, képernyőkép, Betűtípus, képernyő látható

Automatikusan generált leírás

<button id="myButton">Kattints rám!</button>

<script>

const button = document.getElementById('myButton');

button.addEventListener('click', (event) => {

console.log('Esemény típusa:', event.type);

console.log('Eseményt indító elem:', event.target);

});

</script>

* **Magyarázat:**
  + **event.type:** Megadja, hogy milyen típusú esemény történt (pl. kattintás).
  + **event.target:** Az elem, amely az eseményt kiváltotta.

**4. Több esemény kezelése egy elemen (10 perc)**

* **Tanár mondja:** "Egy HTML elemhez több eseménykezelőt is hozzáadhatunk, amelyek különböző eseményekre reagálnak. Ez lehetővé teszi, hogy egy elem különböző felhasználói műveletekre más-más módon reagáljon."
* **Példa bemutatása:**

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

<button id="multiButton">Több esemény</button>

<script>

const button = document.getElementById('multiButton');

// Kattintási esemény

button.addEventListener('click', () => {

console.log('Kattintás történt!');

});

// Egérrel rámutatás eseménye

button.addEventListener('mouseover', () => {

console.log('Az egér az elem fölé került.');

});

</script>

* **Magyarázat:**
  + Egy elemhez különböző eseménytípusokhoz különböző eseménykezelőket rendelhetünk.
  + Például a kattintás és az egérmozgás két külön eseménytípus, amelyekre eltérő funkciók kapcsolhatók.

**5. Gyakorlati Feladatok (15 perc)**

* **Feladat 1:** "Készítsetek egy gombot, amely kattintásra megváltoztatja a szövegét: 'Megnyomtál!'."
* **Feladat 2:** "Hozzatok létre egy szövegmezőt, amelyben minden leütött billentyűt megjelenít a konzolon."
* **Feladat 3:** "Készítsetek egy gombot, amely egérrel rámutatáskor megváltoztatja a hátterének színét, majd visszavált az eredetire, amikor az egér elhagyja az elemet."

**6. Óra Zárása és Kérdések (5 perc)**

* **Tanár mondja:** "Az eseménykezelés alapjainak megértése kulcsfontosságú a dinamikus weboldalak készítéséhez. Ezek a technikák segítenek abban, hogy a weboldalak reagáljanak a felhasználók tevékenységeire, és interaktív élményt nyújtsanak. Ha valakinek kérdése van, most tegye fel!"
* **Otthoni Feladat:** "Hozzatok létre egy interaktív értesítési rendszert, ahol egy gombra kattintva egy új üzenet jelenik meg a felhasználó számára. Az üzenet tetszőleges szöveget tartalmazhat, és stílusa legyen dinamikusan változtatható."