

JEGYZŐKÖNYV

Web technológiák 1.
Féléves feladat

**„Merre tovább? – pályaválasztást segítő
weboldal”**

Készítette: **Molnár Márk**
Neptun-kód: **TLZ12Y**
Dátum: **2025. december**

Miskolc, 2025

Tartalomjegyzék

1. Bevezetés	3
2. Feladat leírása.....	3
3. Weblap felépítése	3
3.1. Fájlszerkezet	3
3.2. Navigációs menü	4
3.3. Kezdőlap – útválasztó	4
3.4. Munka oldal – JSON-os munkaötletek.....	4
3.5. Egyetem oldal – animált idővonal	5
3.6. Egyik sem oldal – ajánló kártyák animációval	5
3.7. Válaszaim oldal – űrlap és PDF generálás	5
4. Scriptek	6
4.1. JSON betöltése és kártyák létrehozása (munka.js)	6
4.2. Egyetemi idővonal jQuery animációja (egyetem.js)	7
4.3. Ajánló kártyák beúszó animációja (egyiksem.js)	8
4.4. Hamburger „Hasznos linkek” panel kezelése	8
4.5. Válaszaim űrlap-ellenőrzés és PDF generálás (valaszaim.js)	9
5. Összegzés, tapasztalatok	10

1. Bevezetés

A web technológiák – HTML, CSS, JavaScript, jQuery, JSON – ma alapvető eszközei a modern, rezponzív weboldalak fejlesztésének. A féléves projekt célja az volt, hogy ezeket a technológiákat egy összefüggő, többoldalas, valós célt szolgáló weboldalban alkalmazzam.

A választott témám egy **pályaválasztást segítő mini weboldal**, amelyet konkrét személynek, a sógoromnak, **Maros Leventének** készítettem. A weboldal segít végiggondolni, hogyan szeretne továbbmenni az életben:

- **Munka** – ha minél előbb pénzt keresne és gyakorlati tapasztalatot szerezne.
- **Egyetem** – ha szívesen tanulna tovább, és diplomát szerezne.
- **Egyik sem** – ha most még egyik sem fér bele, de otthon, autodidakta módon szeretne tanulni.

2. Feladat leírása

A tantárgyi kiírás szerint legalább **öt HTML fájlt**, HTML5 strukturális elemeket (header, nav, section, article, aside, footer), űrlapot, videót, CSS gridet, valamint JavaScript, jQuery, JSON és AJAX megoldásokat kellett használnunk. Ezen felül form-ellenőrzés, jQuery animáció, és JSON-ból történő megjelenítés is szerepelt a követelmények között.

Az én projektem fő céljai:

- **Három fő út bemutatása** (munka, egyetem, autodidakta út), külön HTML oldalakon.
- **Közös, sticky navigációs menü**, amellyel bármelyik oldalról elérhető a többi.
- **Hasznos linkek panel minden útnál** (munka/egyetem/egyik sem), hamburger gombbal.
- **JSON alapú munkaötlet-lista** AJAX hívással (munka oldal).
- **Folyamatos jQuery animáció** az egyetemi évek „idővonala” (egyetem oldal).
- **Scroll-alapú beúszó animáció** az autodidakta ajánlókártyához (egyik sem oldal).
- **Űrlap ellenőrzés és PDF generálás** a „Válaszaim és tervem” oldalon jsPDF segítségével.

A weboldal kifejezetten személyre szabott: a „Válaszaim” oldal PDF-je Levi nevét, profilképét, „aláírását” és az ő válaszait tartalmazza, így ajándékként is használható.

3. Weblap felépítése

3.1. Fájlszerkezet

A projekt mappastruktúrája (Neptun kód alapján):

- **TLZ12YWebTech/**
 - index.html – kezdőlap, útválasztó
 - munka.html – munka út
 - egyetem.html – egyetem út
 - egyiksem.html – autodidakta út
 - valaszaim.html – válaszaim és tervem űrlap + PDF
 - style.css – közös stíluslap
 - munka.js – munka oldal JS / JSON betöltés
 - egyetem.js – egyetem oldal jQuery animáció
 - egyiksem.js – egyik sem oldal jQuery animáció

- `valaszaim.js` – űrlapkezelés és PDF generálás
- `munka_adatok.json` – példaként használt munkaötletek
- `kepek/levi.jpg` – Levi profilképe a PDF-hez
- `kepek/alairas.png` – digitális aláírás a PDF-hez
- `kepek/chatgpt.png` – illusztráció az autodidakta oldalhoz
- `kepek/brocode.mp4` – rövid videó (programozós ajánló)

A HTML fájlok mindegyikében megtalálható a header, nav, main, section, aside és footer elemek valamely kombinációja, így teljesül a HTML5 strukturális követelmény.

3.2. Navigációs menü

Az oldal tetején minden oldalon azonos, **sticky** navigáció található:

- `nav.main-nav` – benne 5 menüpont: „Kezdőlap, Munka, Egyetem, Egyik sem, Tervem”.
- A menü flexboxszal középre igazított, lekerekített, árnýekos gombokkal.

Rövid HTML-részlet:

```
<nav class="main-nav">
  <a href="index.html">Kezdőlap</a>
  <a href="munka.html">Munka</a>
  <a href="egyetem.html">Egyetem</a>
  <a href="egyiksem.html">Egyik sem</a>
  <a href="valaszaim.html">Tervem</a>
</nav>
```

A CSS-ben a `.main-nav` sticky pozíciót kap, háttér-blurrel és box-shadow-val. A menüpontok hover állapotban színt váltanak, hogy a felhasználó lássa, hogy kattinthatóak.

3.3. Kezdőlap – útválasztó

Az `index.html` nagyon egyszerű, de vizuálisan erős nyitóoldal. A fő tartalom egy `options-wrapper` div, benne három nagy, színes, „pill” jellegű gomb:

- **MUNKA** – narancssárga gradient háttérrel (`.option-work`).
- **EGYETEM** – kék gradient háttérrel (`.option-uni`).
- **EGYIK SEM** – zöld gradient háttérrel (`.option-none`).

Mindhárom gomb `<a>` elemként viselkedik, a megfelelő aloldalakra navigál. A gombokra húzva az egeret finoman feljebb „ugranak” és világosabbak lesznek.

3.4. Munka oldal – JSON-os munkaötletek

A **Munka** oldal (`munka.html`) három fő részből áll:

1. **Kiemelt kártyák** (`.highlights-grid`):
 - Első fizetésed 
 - Valódi tapasztalat 
 - Iránytű a jövődre 

2. **Hasznos linkek** – hamburger gombbal előhívható `aside.links-aside` (például Profession.hu, CVOnline, LinkedIn).
3. **JSON-ból betöltött munkaötletek:**
A `job-section` részben egy `#job-grid` div található, amelyet a `munka.js` tölt fel AJAX kéréssel a `munka_adatok.json` alapján.

A JSON betöltésével teljesül a JSON + AJAX követelmény: a JavaScript új HTML elemeket hoz létre és illeszt be a DOM-ba.

3.5. Egyetem oldal – animált idővonal

Az **Egyetem** oldal (`egyetem.html`) a bevezető szövegek után két nagyobb blokkot tartalmaz:

1. **Folyamatosan animált idővonal** (`section.uni-timeline`):
 - Három szakasz: 1. év, 2. év, 3. év.
 - minden szakasz egy vékony csík, benne egy `.uni-phase-fill` elem, melyet jQuery animáció tölt meg „zölden”.
2. **Konkrét szakok gridben** (`section.degree-grid`):
 - Gazdaságinformatikus BSc
 - Logisztikai mérnöki BSc
 - Programtervező informatikus BSc

A timeline folyamatosan „körbejár”, ezzel vizuálisan mutatja, hogy az egyetemi évek egymásra épülő folyamatot jelentenek.

3.6. Egyik sem oldal – ajánló kártyák animációval

Az **Egyik sem** oldal (`egyiksem.html`) azt az utat mutatja be, amikor valaki **autodidakta** módon szeretne haladni.

Fő blokkok:

- Bevezető szövegek a saját tempójú tanulásról.
- „**Saját javaslataim**” ajánló kártyák (`section.recommendations`), benne:
 - BroCode – programozás lépésről lépésre (videóval).
 - ChatGPT – beszélgetés, ötletelés, magyarázatok (képpel).
- A kártyák `.reco-card` osztályt kapnak, és csak akkor „úsznak be” és válnak láthatóvá (opacity + transform animáció), amikor a felhasználó legörget addig (jQuery-vel figyelt scroll).

Ez a rész valósítja meg a jQuery-alapú scroll animációt.

3.7. Válaszaim oldal – űrlap és PDF generálás

A **Válaszaim és tervem** oldal (`valaszaim.html`) egy több kérdésből álló űrlap:

1. **Melyik utat választom most?** – rádiógombok (munka / egyetem / egyik sem).
2. **Milyen területen szeretnék sikeres lenni?** – datalistás szövegmező.
3. **Konkrét lépések** – több soros textarea.
4. **Ma ezt az egy dolgot biztosan megteszem** – egy soros input.
5. **Mikor fogom elkezdeni?** – rádiógombok (valójában csak a „Ma” opció aktív).

6. **Fogadalmak** – négy jelölőnégyzet (nem halogatók, felelősség, visszajelzéskérés, nem csak aludni kelek fel).
7. **Színválasztó** – a jövő színe, amely a PDF-beli keret színét határozza meg.
8. **Dátum** – cél elérésének napja.

Az oldal alján két gomb szerepel:

- „**Válaszaim letöltése PDF-ben**” – #pdf-button.
- „**Válaszok törlése**” – #clear-answers.

A PDF generálását a `valaszaim.js` végzi a jsPDF könyvtár segítségével. A generált dokumentum két oldalas, tartalmazza Levente nevét, profilképét, a választott utat, az általa beírt lépésekét, valamint egy személyes „utóiratot”.

4. Scriptek

Ebben a fejezetben a fontosabb JavaScript / jQuery részeket mutatom be rövid magyarázattal.

4.1. JSON betöltése és kártyák létrehozása (munka.js)

A `munka.js` feladata, hogy a `munka_adatok.json` fájl tartalmát betöltsse, és az alapján kártyákat generáljon a „Munka” oldal „Milyen munkákat próbálhatnék ki?” szekciójában.

Rövidített kódrészlet:

```
document.addEventListener('DOMContentLoaded', function () {
  const jobGrid = document.getElementById('job-grid');
  if (!jobGrid) return;

  jobGrid.innerHTML = '<p class="job-loading">Példák
betöltése...</p>';

  fetch('munka_adatok.json')
    .then(response => response.json())
    .then(jobs => {
      jobGrid.innerHTML = '';
      jobs.forEach(job => {
        const card = document.createElement('article');
        card.className = 'job-card';

        const meta = [];
        if (job.type) meta.push(job.type);
        if (job.hours) meta.push(job.hours);

        card.innerHTML =
          `<h3>${job.title}</h3>` +
          (meta.length ? `<p class="job-
meta">${meta.join(' • ')}` : '') +
          `<ul>` +
          (job.goodFor ? `<li><strong>Kinek
való:</strong> ${job.goodFor}</li>` : '') +
          `</ul>`;
      });
    });
});
```

```

        (job.description ?
`<li>${job.description}</li>` : '') +
        (job.pay ? `<li><strong>Kb. fizetés:</strong>
${job.pay}</li>` : '') +
        `</ul>`;

        jobGrid.appendChild(card);
    });
}
.catch(() => {
    jobGrid.innerHTML =
        '<p>Nem sikerült betölteni a példákat, de a saját
tervez így is működik. 😊</p>';
});
});

```

A script:

- DOMContentLoaded eseményre fut.
- fetch segítségével tölti be a JSON-t (AJAX).
- Dinamikusan hoz létre <article> elemeket, és beszúrja a DOM-ba.

4.2. Egyetemi idővonal jQuery animációja (egyetem.js)

Az egyetemi idővonalhoz jQuery-t használok. A .uni-phase-fill elemek szélessége 0-ról 100%-ra nő, majd a script a következő sávot animálja, végül végtelen ciklusban ismétli.

Rövidített kód:

```

$(function () {
    const $fills = $('.uni-phase-fill');
    if (!$fills.length) return;

    function animatePhase($el) {
        return $el
            .stop(true, true)
            .css({ width: '0%', opacity: 1 })
            .animate({ width: '100%' }, 1200)
            .delay(200)
            .animate({ opacity: 0.3 }, 250)
            .animate({ opacity: 1 }, 0);
    }

    function loopTimeline() {
        let i = 0;

        function next() {
            const $current = $fills.eq(i);
            animatePhase($current).promise().done(function () {
                i = (i + 1) % $fills.length;
                next();
            });
        }
    }
}

```

```

        next();
    }

    loopTimeline();
} );

```

Ez a script teljesíti a jQuery animációs követelményt: HTML elem osztály alapján kiválasztása, animálása, majd promise segítségével egymás utáni láncolása.

4.3. Ajánló kártyák beúszó animációja (egyiksem.js)

Az autodidakta oldalon az ajánlókártyák csak akkor „jönnek be”, amikor a felhasználó legörget addig.

```

$(function () {
    function revealCards() {
        const winTop      = $(window).scrollTop();
        const winBottom = winTop + $(window).height();

        $('.reco-card').each(function (index) {
            const $card = $(this);
            if ($card.hasClass('reco-visible')) return;

            const elTop = $card.offset().top;

            if (winBottom > elTop + 40) {
                setTimeout(function () {
                    $card.addClass('reco-visible');
                }, index * 150);
            }
        });
    }

    revealCards();
    $(window).on('scroll', revealCards);
} );

```

A `.reco-visible` osztály a CSS-ben felel azért, hogy az opacity 0-ról 1-re válson, és a kártya a helyére csússzon.

4.4. Hamburger „Hasznos linkek” panel kezelése

Mindhárom út (munka, egyetem, autodidakta) oldalán ugyanaz a logika vezérli a hamburger gombot:

```

(function () {
    const toggle = document.querySelector('.links-toggle');
    const panel  = document.querySelector('.links-aside');

    if (!toggle || !panel) return;

    toggle.addEventListener('click', function () {
        panel.classList.toggle('open');
        toggle.classList.toggle('open');
    });
})();

```

- A .links-toggle gombra kattintva a .links-aside panel open osztályt kap, vagy elveszíti.
- Az open osztály a CSS-ben display: block-ra állítja a panelt, különben rejtve marad.

4.5. Válaszaim űrlap-ellenőrzés és PDF generálás (valaszaim.js)

A valaszaim.js a legösszetettebb script:

- 1. Profilkép és aláírás előtöltése**
 - o levi.jpg és alairas.png betöltése rejttet <canvas> segítségével, majd toDataURL() használata, hogy a jsPDF-be be lehessen tenni a képeket.
- 2. Dátummező automatikus kitöltése**
 - o A mai dátum kerül az #szerzodes_datum mezőbe, ha még nincs megadva.
- 3. Úrlap ellenőrzés a PDF gomb előtt**
 - o A #pdf-button kattintására először ellenőrzi a kötelező kérdéseket.
 - o Ha bármelyik hiányzik, a #formError elembe piros szöveg kerül (CSS: .form-error { color:#ef4444; font-weight:600; }), például: „2. kérdés megválaszolása kötelező.”, és a PDF készítés leáll.

Rövidített ellenőrző kód:

```
function validateForm() {
    const errorEl = document.getElementById('formError');
    if (errorEl) errorEl.textContent = '';

    // 1. kérdés - útválasztás (radio)
    const path =
document.querySelector('input[name="valasztott_ut"]:checked');
    if (!path) {
        if (errorEl) errorEl.textContent = '1. kérdés megválaszolása kötelező.';
        return false;
    }

    // 2. kérdés - terület
    const terulet = document.getElementById('siker_terulet');
    if (!terulet || !terulet.value.trim()) {
        if (errorEl) errorEl.textContent = '2. kérdés megválaszolása kötelező.';
        return false;
    }

    // 3. kérdés - lépések
    const lepesek = document.getElementById('lepesek');
    if (!lepesek || !lepesek.value.trim()) {
        if (errorEl) errorEl.textContent = '3. kérdés megválaszolása kötelező.';
        return false;
    }

    // 4. kérdés - mai lépés
    const mai = document.getElementById('mai_lepes');
    if (!mai || !mai.value.trim()) {
```

```

        if (errorEl) errorEl.textContent = '4. kérdés megválaszolása
kötelező.';
            return false;
    }

    return true;
}

document.getElementById('pdf-button').addEventListener('click',
function () {
    if (!validateForm()) return;
    // ... ha minden rendben, itt indul a jsPDF generálás ...
});
```

4. PDF létrehozása jsPDF-vel

A validáció után a script:

- new jsPDF({ orientation: 'p', unit: 'mm', format: 'a4' }) hívással hoz létre dokumentumot.
- Beállítja a betűtípuszt Times-ra (doc.setFont ('times', 'normal'));).
- A lap tetejére **Levi profilképét** helyezi színes keretben.
- Középre írja a nevet és egy rövid szlogent („Maros Levente – sikérének titkai”).
- A felhasználó válaszai alapján folyamatos szöveget generál, pl.:
 - Melyik utat választotta (munka/egyetem/autodidakta).
 - Melyik területen szeretne sikeres lenni.
 - Milyen lépéseket írt be.
 - Mi az az egy dolog, amit ma biztosan megtesz.
- Az oldal aljára kerül az aláírás kép, alatta szöveg: „Maros Levente, a legsikeresebb ember aláírása”.
- **A második oldal** egy személyes „Utóirat” rövid igékkel, jókívánságokkal.

5. „Válaszok törlése” gomb

A #clear-answers gomb visszaállítja az űrlapot alapértelmezett állapotra:

- Törli a szövegmezők és textarea tartalmát.
- Visszaállítja a rádiógombokat („Ma” opción).
- Kiveszi a jelölönégyzetek pipáit.
- Visszaállítja az alap színt és a dátumot.

5. Összegzés, tapasztalatok

A „Merre tovább?” projekt során sikerült olyan, többoldalas weboldalt készíteni, amely:

- **Minden követelményt** teljesít a Web technológiák 1 féléves feladatához kapcsolódóan: HTML5 struktúra, legalább öt HTML oldal, űrlap elemek széles skálája, videó, CSS grid, JavaScript, jQuery animáció, JSON + AJAX betöltés, form-ellenőrzés, dinamikus DOM-módosítás.
- Valós, személyes témát dolgoz fel: segít Leventének végiggondolni a jövőjét, és konkrét tervet készíteni.

- A fő funkciók – munkalehetőségek listája, egyetemi idővonal, autodidakta javaslatok, valamint a személyre szabott PDF – minden segíti a felhasználót abban, hogy ne csak „nézegessen” egy oldalt, hanem **cselekvési tervet** is kapjon.

Fejlesztőként a projekt közben:

- Gyakoroltam a **moduláris JavaScript** írást (külön JS fájlok: munka.js, egyetem.js, stb.).
- Jobban megértettem a **fetch + JSON** használatát és a jQuery animációs lehetőségeit.
- Először használtam a gyakorlatban a **jsPDF** könyvtárat képekkel, több oldallal, és dinamikusan generált szövegekkel.

A projekt végére nemcsak a kreditet szeretném megszerezni, hanem azt is, hogy Levente a PDF-et látva egy kicsit motiváltabb legyen: akár a munkát, akár az egyetemet, akár a saját útját választja – a lényeg, hogy **elinduljon**.