

JEGYZŐKÖNYV

Számítógép Architektúrák

Féléves feladat

HTML weblap készítése

Készítette: **Erdélyi Péter**

Neptunkód: **JH3V7T**

Dátum: 2023. november 28.

TARTALOMJEGYZÉK

1. FELADATKIÍRÁS	1
2. A KEZDŐLAP FELÉPÍTÉSE	2
2.1. Üdvözlő képernyő	2
2.2. A <head> elem	2
2.3. Main.css.....	3
2.4. A <body> elem.....	3
3. A TANTÁRGYAK WEBLAPJAINAK FELÉPÍTÉSE	5
3.1. Automatika.html.....	5
3.2. Listák használata	5
3.3. Szöveg formázása.....	7
3.4. Videó lejátszása.....	7
3.5. Sebességváltó csúszka.....	8

1. FELADATKIÍRÁS

Tervezzon meg egy HTML weblapot, használjon modern webfejlesztési elemeket (HTML4/HTML5, CSS, JavaScript). Készítsen egy statikus HTML4/HTML5 oldalt, amely az Ön által tanult/választott három tantárgy tematikáját mutatja be, majd illesszen be egy video fájlt egy oldalra.

Legyen egy kezdőlap, amely a három kurzus menüpontját tartalmazza, ill. a fejléc képet is tartalmazzon. A kezdőlapon egy-egy kurzus menüpontjára kattintunk, megjelenik az adott kurzus tematikájának leírása és hozzá egy kép. Legyen lehetőség visszalépni a kezdőlapra.

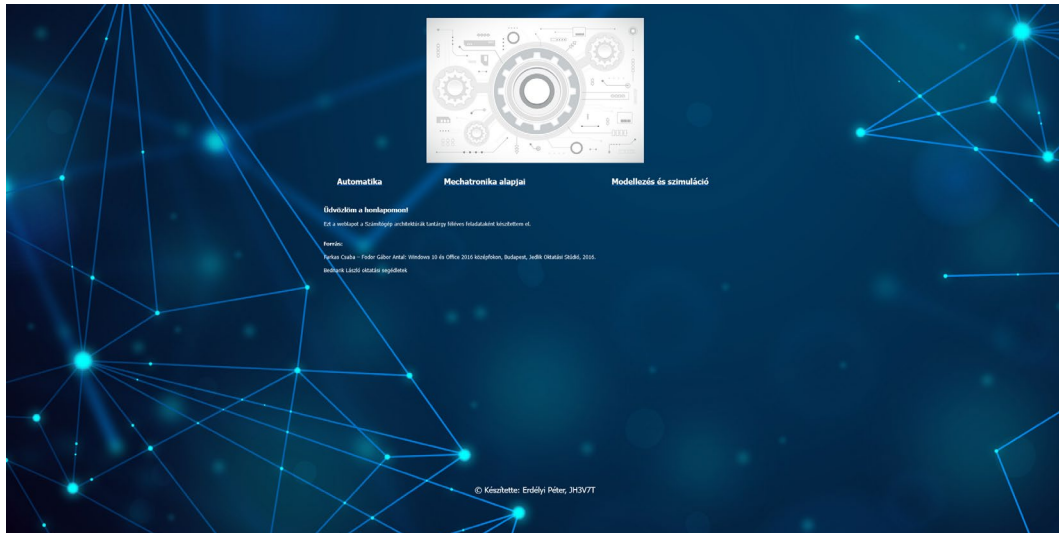
A lábléc tartalmazza a weblap készítőjének nevét és neptunkódját. Legyen egy mappa a képeknek.

JavaSlat: a forrásfájl neve megegyezik az oldal nevével (pl.: kezdolap.html, index.html stb...).

2. A KEZDŐLAP FELÉPÍTÉSE

2.1. Üdvözlő képernyő

A weblapra történő első látogatás alkalmával az alábbi képernyő (1. ábra) fogad:



1. ábra: Üdvözlő képernyő

Itt olvasható néhány, az oldallal kapcsolatos információ, többek között a weblap célja és az elkészítéshez használt források. Az üdvözlő képernyő felső részében elhelyeztem három menüpontot, melyek segítségével elnavigálhatunk az egyes tantárgyak oldalára.

2.2. A <head> elem

A HTML dokumentumok egyik legfontosabb alkotója a <head> elem, ami magyarul fejrészt jelent. Az itt megadott elemek nem jelennek meg a weboldal részeként, viszont a működéshez szükséges metaadatokat tartalmazzák. A feladatban a 2. ábrán látható <head> rész látható.

```
<head>
  <title>Kezdőlap</title>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" type="text/css" href="css/main.css">
</head>
```

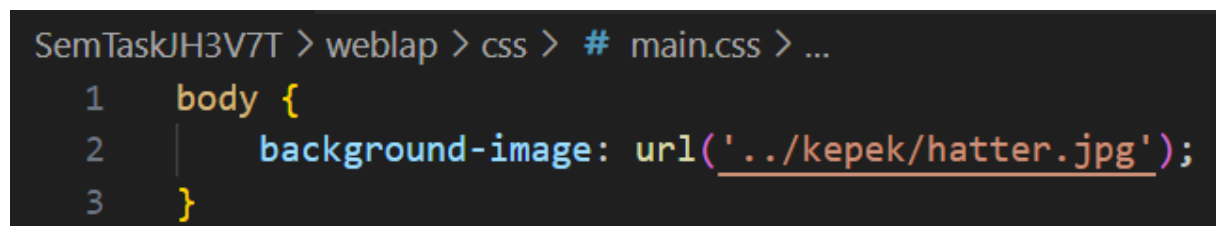
2. ábra: <head> tartalma

A `<title>` azonosítja az adott oldal címét, jelen esetben ez a kezdőlap. Karakterkódolásnak az UTF-8 formátumot jelöltem ki. Továbbá konfiguráltam a `viewport` metarészt is, amely egy felhasználónak a weblap látható területét módosítja, és hozzájárul ahhoz, hogy az oldal eszköztől függően jelenítse meg a tartalmát. Ez a beállítás az oldal szélességét az eszköz szélességéhez igazítja, így a megtekintés nem okoz majd problémát.

A fejrészt a `<link>` elemmel zárom. Célszerű a stílust külön fájlban kezelni, ezért a kezdőlapot (valamint a feladat többi HTML oldalának jelentős részét) hozzákapcsoltam a `main.css` stílusfájlhoz, melynek elérési útvonalát a `href` után adtam meg.

2.3. Main.css

A `main.css` fájl szolgál a stílus definiálására. Erre mutat példát a 3. ábra, ahol a `body` részben szerepel a háttérkép. Mivel a `body` minden HTML dokumentumban jelen van, ezért nem szükséges egyenként beállítani a háttérképet, hanem elegendő a stílusfájlt hozzájuk linkelni.



```
SemTaskJH3V7T > weblap > css > # main.css > ...  
1  body {  
2    background-image: url('../kepek/hatter.jpg');  
3  }
```

3. ábra: Main.css (részlet)

2.4. A `<body>` elem

A `<body>` elem egy HTML oldal törzsét képezi, vagyis bekezdések, táblázatok, listák stb. tartoznak hozzá, ugyanakkor dokumentumonként csak egy törzsrész engedélyezett. A feladatban lévő `index.html` `<body>` részét vázolja a 4. ábra (következő oldal).

```

<body>

  <table align="center" style="table-layout:fixed; font-family:tahoma; font-size:14pt;
    color:■white" border="0">

    <tr style="height:400px">
      <td colspan="3" style="width:1060px; text-align:center">
        <a href="kezdolap.html" target="keret"></a>
      </td>
    </tr>

    <tr style="height:30px; font-weight:bold">
      <td style="text-align:center">
        <a href="automatika.html" target="keret"><font style="color:■white;
          text-decoration:none">Automatika</font></a>
      </td>
      <td style="text-align:center">
        <a href="mechatronika.html" target="keret"><font style="color:■white;
          text-decoration:none">Mechatronika alapjai</font></a>
      </td>
      <td style="text-align:center">
        <a href="modszim.html" target="keret"><font style="color:■white;
          text-decoration:none">Modellezés és szimuláció</font></a>
      </td>
    </tr>

    <tr style="height:700px">
      <td colspan="3" style="text-align:center">
        <iframe name="keret" src="kezdolap.html" width="100%" height="700" scrolling="auto"
          frameborder="0" marginheight="10" marginwidth="20">
          A böngészője nem támogatja a belső keretek használatát!
        </iframe>
      </td>
    </tr>

    <tr style="height:40px">
      <td colspan="3" style="width:1060px; text-align:center">
        &copy; Készítette: Erdélyi Péter, JH3V7T
      </td>
    </tr>

  </table>

</body>

```

4. ábra: <body> tartalma (index.html)

A fenti ábrából jól látszik, hogy az `index.html` törzsrésze egy táblázat. A táblázat középre rendezett, és fix méretű sorokból (cellákból) tevődik össze. Legfelül található a `kezdolap.jpg`, ami egyúttal biztosítja a kezdőlaphoz való visszalépést. Ezt követi az a sor, melyet a három tantárgynak megfelelően három cellára osztottam fel. Minden tantárgy rendelkezik egy saját weboldallal, ezért közülük bármelyikre kattintva átkerülünk az adott tantárgy oldalára. A táblázat utolsó sora pedig tartalmazza a nevemet és neptunkódomat.

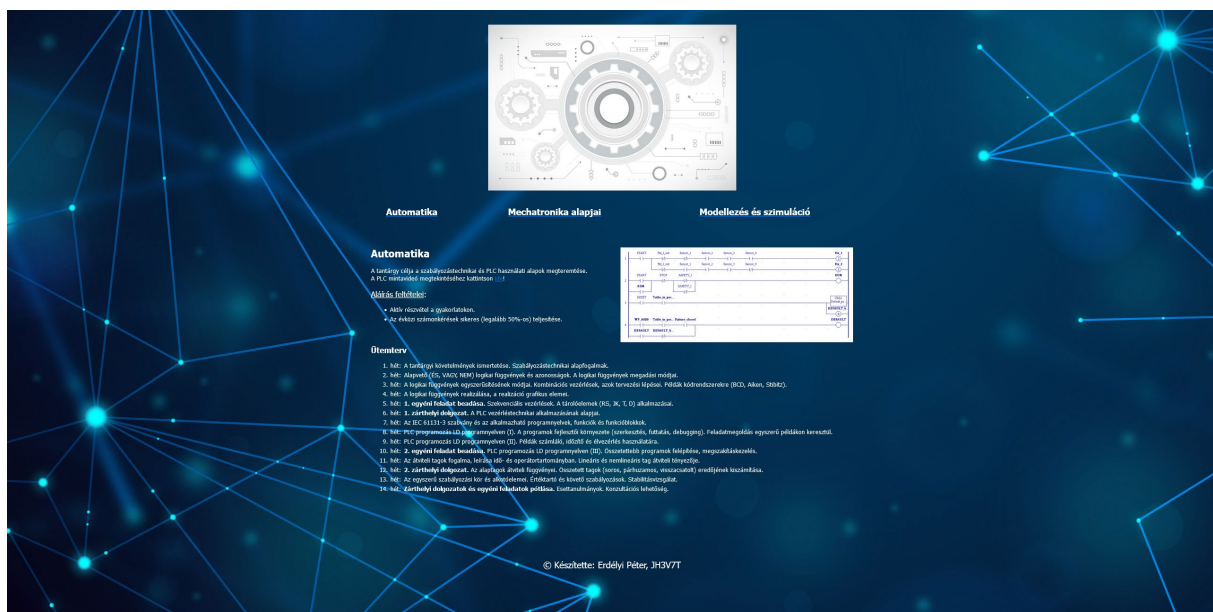
Az `<iframe>` keretnek köszönhetően lehetőség van egy HTML weboldalt egy másikba ágyazni. Ezt a technikát alkalmazva, a táblázat 3. sorában jeleníttem meg az egyes tantárgyak weblapjainak tartalmát.

3. A TANTÁRGYAK WEBLAPJAINAK FELÉPÍTÉSE

A tantárgyak oldalai között csak tartalmi eltérés van, szerkezetileg viszont megegyeznek. Éppen ezért az alábbiakban ismertetem az `automatika.html` felépítését, a `mechatronika.html` és `modszim.html` ennek mintájára készültek.

3.1. Automatika.html

Az automatika menüpontra kattintva a következő, 5. ábrán vázolt oldalra érkezünk.



5. ábra: *Automatika.html* kinézete

Ezen az oldalon olvasható a tantárgy célja, az aláíráshoz szükséges feltételek, illetve a félévi tematika heti lebontásában. Jobb oldalt beillesztettem egy képet, ami egy PLC program részletét hivatott szemléltetni.

3.2. Listák használata

A tantárgyak weblapjainak egyik közös eleme a lista. A HTML három különböző lista alkotására ad módot: rendezett, rendezetlen és definíciós. A feladatban alkalmaztam egy rendezett és egy rendezetlen listát is, melyek közül az utóbbinak a felépítését illusztrálja a 6. ábra (következő oldal).

```
<ul>
  <li>Aktív részvétel a gyakorlatokon.</li>
  <li>Az évközi számonkérések sikeres (legalább 50%-os) teljesítése.</li>
</ul>
```

6. ábra: Rendezetlen lista felépítése

Egy rendezetlen lista az `` nyitóelemmel kezdődik, és az `` zárótaggal végződik. A közöttük lévő részben helyezkednek el az `` listaelemek.

Az ütemterv egy rendezett listából áll, melynek minden listaeleme tartalmazza a „hét: ” kifejezést. Ezt a hatást a `::before` szelektor felhasználásával értem el (7. ábra). A CSS szelektorjai mintákat keresnek a HTML elemek között, és a saját szabályaink szerint határozzák meg a stílust. Maga a `::before` az öt megelőző elemek elé illeszt be közvetlenül valamit.

```
ol {
  line-height: 150%
}

.utemterv li::before {
  content: "hét: ";
  border-radius: 1px;
  padding-inline: 1px;
  margin-inline-end: 1px;
}
```

7. ábra: Ütemterv lista formázása

Mivel a `` listaelem nemcsak a rendezett, hanem a rendezetlen tagok előtt is előfordul, ezért őket meg kell különböztetni egymástól. A szétválasztáshoz létrehoztam az `utemterv` osztályt, amit a lista fejlécébe beírva, csak az ütemterv listaelemeinek stílusa fog a fentiek szerint megváltozni. Az olvashatóság javítása érdekében pedig a vonalmagasságot 1,5-szeresre állítottam.

3.3. Szöveg formázása

A HTML szöveg formázása stílusfájlok mellett a dokumentumban elhelyezett `<style>` attribútummal is elvégezhető.

```
<br><br>

<font style="font-size:12pt"><u>Aláírás feltételei</u>:</font>
```

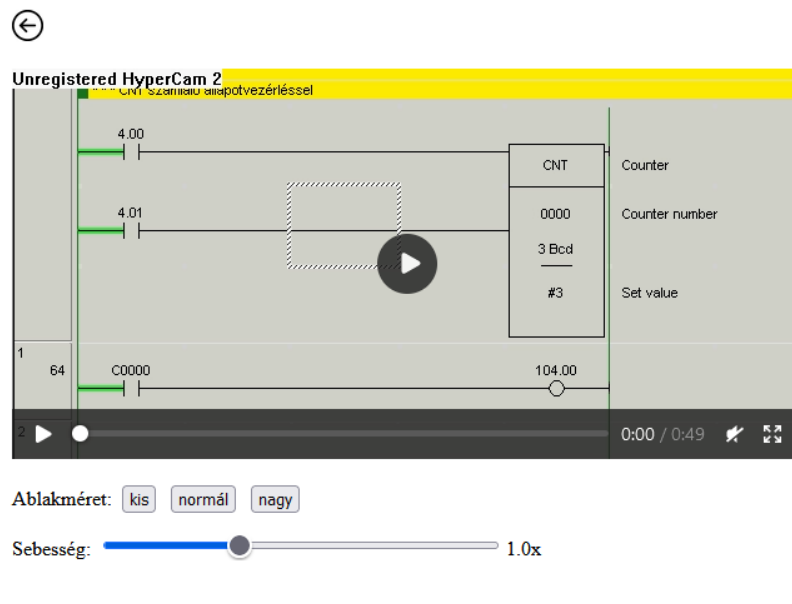
8. ábra: Szöveg formázása

Például az „Aláírás feltételei:” szövegrész (8. ábra) betűméretét 12-esre állítottam, az aláhúzást az `<u>` elemmel eszközöltem. Új sorok közbeiktatására a `
` elemet használtam.

3.4. Videó lejátszása

Az automatika oldalon található a PLC mintavideo linkje. Ezt megnyitva átkerülünk a `video.html` weblapra, melyet a 9. ábra demonstrál. A nyílra lépve bármikor visszatérhetünk a kezdőlapra.

Videó lejátszása



Forrás: Bölcsény Ildi oktatási segédletek

9. ábra: Videó lejátszása (`video.html`)

Videók beillesztése a `<video>` elemmel történik. A `<source>` részben kell megadni a videó elérési útvonalát, aminek külön mappát hoztam létre. A `<type>` a videó formátumának leírására szolgál, ebben a feladatban a PLC mintavideó formátuma mp4. A fájl felbontásának méreteit külön számszerűsítettem, valamint írtam egy tartalék üzenetet arra az eshetőségre, ha a böngésző nem támogatja a videók lejátszását.

A videólejátszó a `<controls>` attribútum hozzáfűzésével alapvető funkciókat kap, mint pl. indítás, megállítás, hangerő-szabályozás. Az ablakméret kiválasztására a teljes képernyőre váltáson kívül három gombot helyeztem el, melyek kattintás hatására módosítják a videó szélességét.

3.5. Sebességváltó csúszka

Sebességváltásra a videó alatti csúszka vehető igénybe. A csúszka működésének értelmezéséhez tekintsük a 10. ábrát.

```
<div style="max-width: 632px;">
  <label for="csuszk" style="font-size:12pt;">Sebesség:&nbsp;</label>
  <input type="range" min="0.5" max="2" step="0.1" value="1" id="csuszk"
  oninput="modositErtek(value)" onchange="modositSebesseg()" style="height: 12px; width: 50%;">
  <output for="csuszk" id="playbackRate" style="font-size:12pt;">1.0</output>x
</div>
```

10. ábra: Sebességváltó csúszka

Az `<input type="range">` egy tartományon belüli értékválasztást valósít meg. A feladatban az alsó határ 0,5 (lassú lejátszás), míg a felső határ 2 (gyors lejátszás), közöttük pedig 0,1-es léptékkel lehet mozogni. A csúszka alapértelmezetten 1-es értékkel jelenik meg, tehát a videó normál lejátszási sebességgel fog indulni.

A sebesség dinamikus változtatásáról egy JavaScript program (11. ábra) gondoskodik, amely a `<script>` elem után foglal helyet. A csúszka mozgatásával meghívom a `modositErtek(value)` függvényt, amely a `value` értékétől függően változtatja a kijelzett sebességszorót. Mivel ezt egy `oninput` eseményhez rendeltem hozzá, ezért az értékváltoztatás azonnal

bekövetkezik. Ezzel ellentétben a `modositSebesseg()` függvény csak akkor hajtja végre a sebességmódosítást, amint a felhasználó elengedte a csúszkát.

```
<script>

    var videom = document.getElementById('videom');

    function modositErtek(sebessegKijelzett) {
        document.querySelector('#playbackRate').value = sebessegKijelzett;
    }

    function modositSebesseg() {
        videom.playbackRate = document.querySelector('#playbackRate').value;
    }

</script>
```

11. ábra: Dinamikus sebességváltás