

JEGYZŐKÖNYV

Webkönyvtárak

Webprogramozás

KavicsDepó

Készítette: **Mátyás Tibor**

Neptunkód: K4WZH6

Dátum: 2024. május 16

Sárospatak, 2024.

Tartalomjegyzék

1. Bevezetés	3
2. Tervezés	3
2.1 Megjelenés megtervezése	3
2.2 A vázszerkezet kialakítása és dizájn tervezése	4
3. Implementálás.....	6
3.1 Elvart HTML elemek, CSS formázás	6
3.2 Videó beágyazása és vezérlése	6
3.3 JQuery animációk, JavaScript.....	7
3.4 JSON, AJAX.....	8
4. Fájl és mappaszerkezet	9

1. Bevezetés

A meghatározott feladat egy olyan weboldal elkészítése volt, amelyben a webkönyvtárak tantárgy keretein belül megismert technológiák kerülnek alkalmazásra a gyakorlatban is. Természetesen a teljesség igénye nélkül, hiszen nem minden terület került felhasználásra.

A weboldal témáját tekintve egy sóderforgalmazással foglalkozó képzeletbeli cég honlapját testesíti meg. A honlap elkészítése előtt megvizsgáltam néhány valós honlapot is, amely ugyanezzel a témával foglalkozik illetve anyag- és információgyűjtést végeztem.

A témát a feladat kiírásnak megfelelően választottam, figyelembe véve, hogy a kért technológiai szempontoknak meg tudjak felelni. Például, hogy Json obejktumot, tömböt tudjak használni stb.

A valódi fejlesztési folyamathoz hasonlóan úgy gondolom itt is érdemes a vízesés modell¹ alapján átgondolni a fejlesztés egyes szakaszait.

A követelmények dokumentálása a feladat kiírásakor már megtörtént. Ezt követően jön a tervezés fázisa, mind a dizájn mind az üzleti logika oldaláról. Majd az implementáció folyamán megvalósításra kerül a projekt. A kódolás után vagy közben zajlik a tesztelés fázisa. Végül a teszteléskor feltárt hiányosságok javítását követően megtörténik a projekt átadása, jelen esetben a Github repositoryba való feltöltés.

A továbbiakban bemutatom a fejlesztés szakaszait tekintettel arra, hogy mennyire felel meg a feladatkiírásnak.

2. Tervezés

2.1 Megjelenés megtervezése

A megjelenés klasszikus, már talán divatját múlt is, de annak érdekében, hogy a lehető leginkább megfeleljen a feladatkiírásnak igyekeztem a kért HTML5 szerkezetben dolgozni. Az index.html tartalmazza a következőket: article, section, aside, nav, header, footer.

Ezek a tag-ek a megjelenésnél azért fontosabb szerepet is ellátnak, ezért használatukat érdemes gyakorlattá tenni, amennyiben frontend fejlesztési feladataink vannak.

A fenti tag-ek az alábbi váz sémában könnyen elhelyezhetők.



1. ábra: Szerkezeti vázrajz

¹ <https://hu.wikipedia.org/wiki/V%C3%ADzes%C3%A9smodell>

2.2 A vázszerkezet kialakítása és dizájn tervezése

A menü felül helyezkedik el a header és a nav tagban, fixált elemként, tehát görgetéskor mindig top:0 és left:0 pozícionálással. Én szeretem ezt a megoldást, mert a felhasználók is kedvelik. Alatt egy animált slider még mindig a header részeként, majd a section-ban az article tagben a cserélődő tartalom az aside tag pedig az oldalsáv szerepét tölti be. Végül a lábléc a footer tagban.

A képen látható megjelenésnek nagyjából az ELTE oldalán is közzétett HTML5 alapszerkezet is megfelel megfelelő CSS-sel kiegészítve. A konstrukció az alábbi:

```
<!DOCTYPE html>
<html lang="hu">
<head>
<title>Oldalszerkezet elemek</title>
<meta charset="utf-8">
</head>
<body>
<main>
<header>
  <h1>Oldal főcíme</h1>
  <nav>
    <ul>
      <li>Menüpont 1.</li>
      ...
    </ul>
  </nav>
</header>
<section>
  <h1>Szakasz főcíme</h1>
  <article>
    ...
  </article>
  <aside>
    <h1>Kapcsolódó linkek</h1>
    <ul>
      <li>Link 1. </li>
      ...
    </ul>
  </aside>
</section>
<footer>
<p>Készítette: ...</p>
<address>
  ...
</address>
</footer>
</main>
</body>
</html>
```

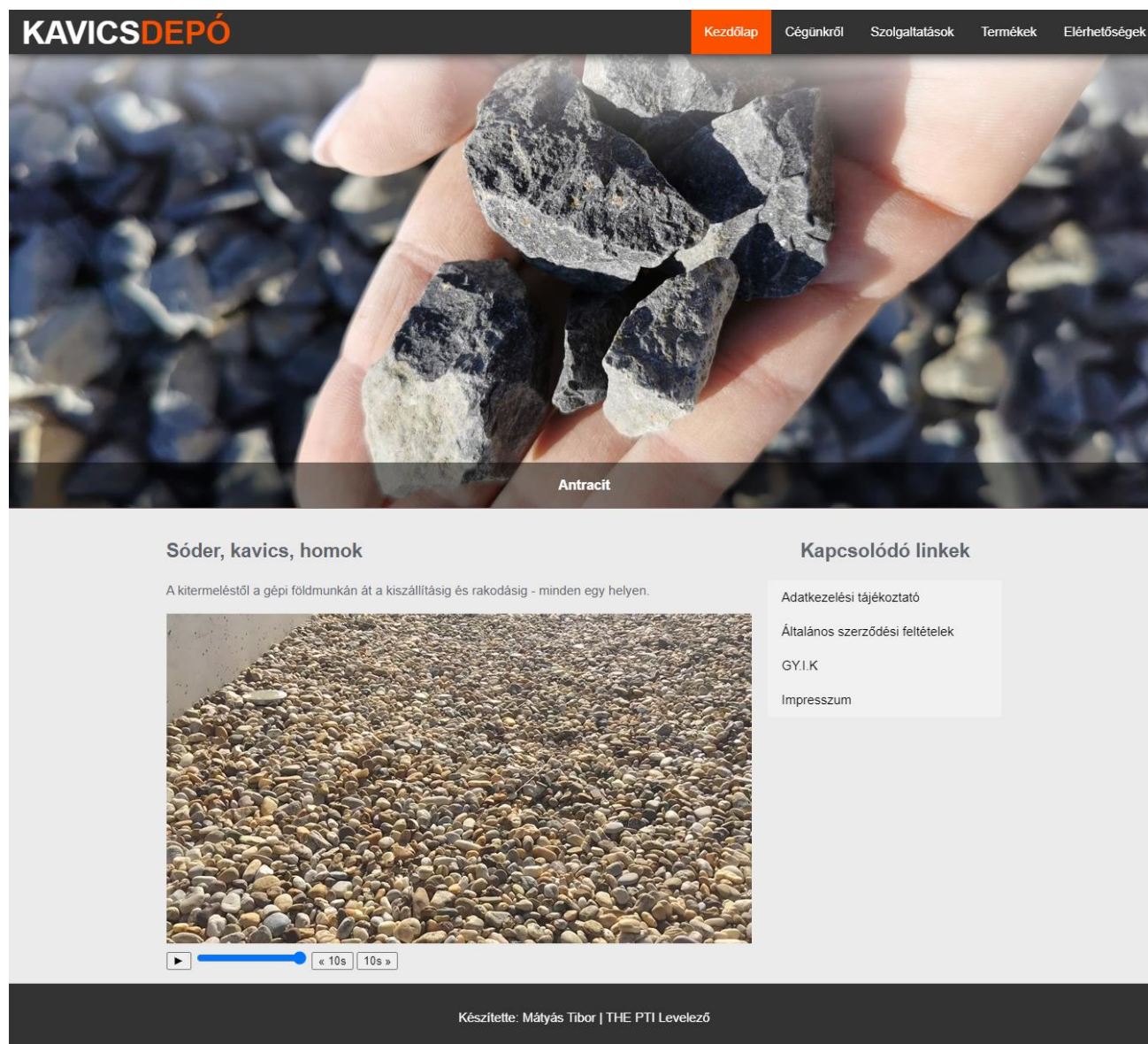
Forrás: https://tamop412.elte.hu/tananyagok/weblapkeszites/lecke7_lap1.html

Ezt felhasználva megalkottam az 1. ábrán található szerkezetet külső CSS fájl segítségével amit az alábbi sorral hivatkoztam az index.html head részében:

```
<link href="css/K4WZH6_style.css" rel="stylesheet" type="text/css" />
```

A CSS fájlban az oldal nem cserélődő részének stílusformázását rögzítettem elsőként. Később már csak kisebb változtatásokat végeztem a teszteléskor. Az egyes tartalmi elemek, úgy mint a termékek

megjelenítése, az űrlapelemek formázása került be utolsóként a CSS fájlba. Mivel a feladat kiírásában szerepel, hogy belső CSS formázást is kell alkalmazni a head részben a JavaScript miatt elhelyeztem a láthatóságra vonatkozó formázást, aminek a célja, hogy a slider képeit a JavaScript jeleníti meg, így azok alapértelmezetten display: none CSS formázást kaptak. Végül kialakult a weboldal dizájnya is az alábbi módon:



2. ábra: Az elkészült webdizájn

A weboldal megjelenését most kevésbé gondos tervezés előzte meg, hiszen Photoshop vagy Corel látványterv nem készült, mint ahogy logó sem került tervezésre. Igyekeztem egy egységes színsémát kialakítani, ami néhány egyszerű színre épül: #222, #fb5000; #ebeb, #000 és #fff, valamint a címsorok és a menü megjelenését kellett megterveznem.

A navigáció tervezésekor törekedtem rá, hogy horizontális és vertikális menüre is legyen megvalósítás. A sémák a W3school oldalán is megtalálhatóak (formázott listákról van szó amiket nav tagek között helyezünk el).²

² https://www.w3schools.com/css/css_navbar_horizontal.asp

3. Implementálás

3.1 Elvárt HTML elemek, CSS formázás

A feladat kiírásában szerepel, hogy milyen elvárt HTML tagek kerüljenek elhelyezésre és formázásra: div, span, p, címsorok, képek, táblázat, linkek, html5 elemei. Ezek java része egyértelmű, hogy hol jelenik meg. Néhány esetben viszont nem magától értetődő a felhasználás. A span taget én a cégnévvel használtam fel. Maga a cégnév (jelen esetben KavicsDepó) egy h1 címsor. Viszont félig fehér félig #fb5000 hexadecimális. Itt a címsoron belüli subformázást segíti a span tag.

Az űrlapelemeket az elérhetőségek menüpontba helyeztem el. A formázás külső CSS fájlban történik. Az űrlap validálását pedig részben a böngésző is elvégzi valamint JQuery is. Aszínválasztó kivételével a meghatározott űrlapelemeket elhelyeztem: szöveges beviteli mező: egy, több soros, adatlista, jelölőnégyzet, rádiógomb, dátumválasztás, gombok.

A CSS fájlban nagyrészt a HTML5 elemeket formáztam meg és pozicionáltam, illetve osztályszintű formázásokat végeztem. Azonosítóra (#id) csak nagyon kevés formázást írtam a feladat jellege miatt. ID attribútumot első sorban a JQuery miatt használtam selectorként, de ezek a megjelenést nem befolyásolták.

3.2 Videó beágyazása és vezérlése

Ugyancsak elvárás volt a továbbiak használata: táblázat, lista és videó. A táblázatokat én az űrlapelemek rendezésére használtam fel valamint a termékek lapon a kiegészítő információk megjelenítésére.

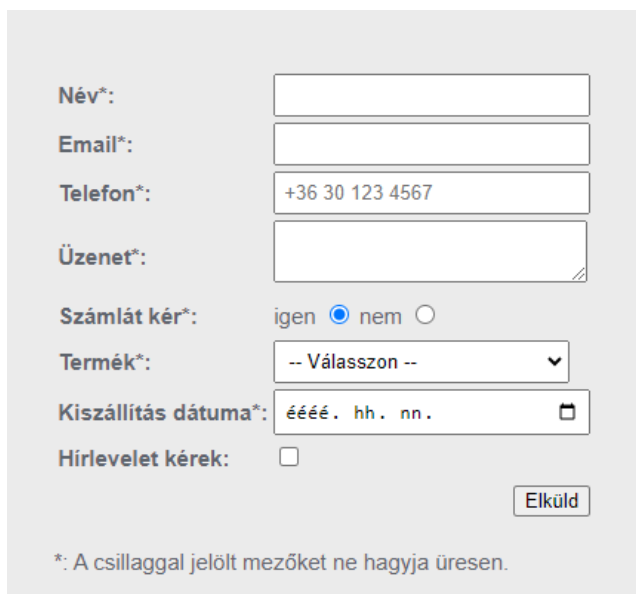
```
<video class="video" style="width:100%">
  <source src="images/kavics.mp4" type="video/mp4" />
  <source src="images/kavics.mp4" type="video/webm" />
  <p>Your browser doesn't support HTML5 video.</p>
</video>
```

Az mp4 videó lejátszót a kezdőlapra illesztettem be a vezérlőkkel együtt. Magát a forrás videót az interneten kerestem és mp4 formátumban letöltöttem, mivel azt a böngésző maga is képes kezelni.

A vezérlők JavaScriptben írt funkcióit külső JavaScript fájlban helyeztem el:

```
<script src="js/video.js" type="text/javascript"></script>
```

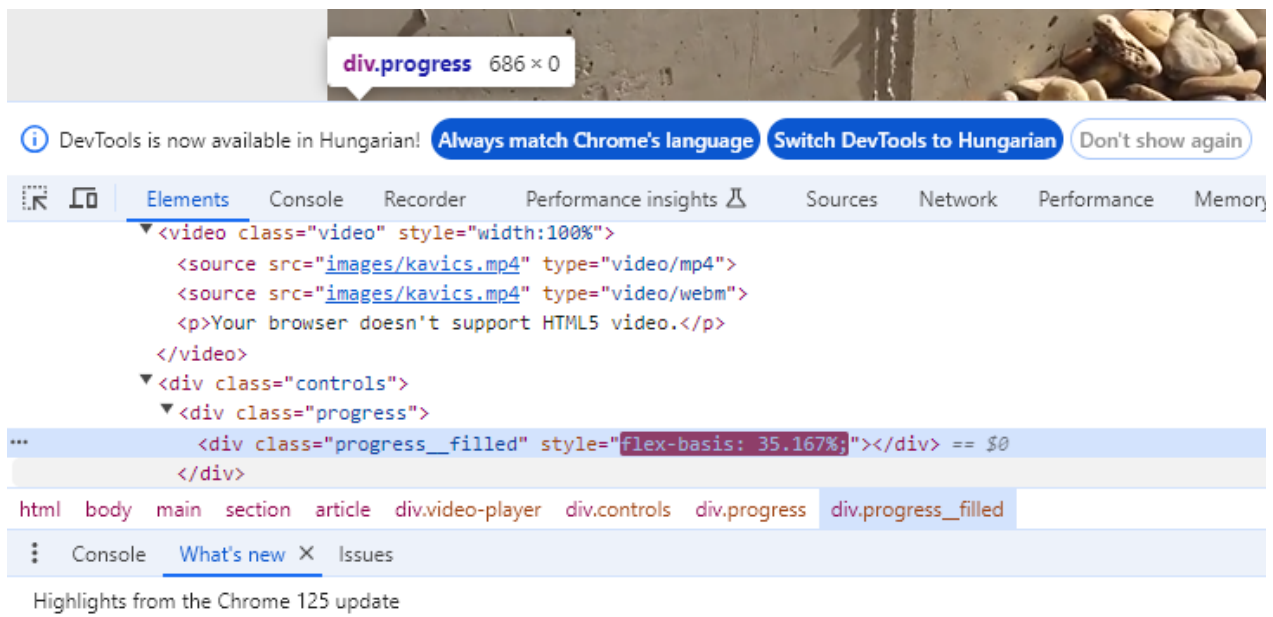
Magához az implementációhoz sok hasznos információ fellelhető az interneten³, én is ott kerestem megvalósítási példákat, hiszen ezzel a témával kevésbé foglalkoztunk órai keretek között. Idő hiányában nem tudtam minden részt megvalósítani. Pl. a progressbar is működik, de nincs CSS formázva, így nem látszik, de kódvizsgálat közben szépen látszik, ahogy változnak lejátszás közben az értékek.



The screenshot shows a contact form with the following elements:

- Név*:** Text input field.
- Email*:** Text input field.
- Telefon*:** Text input field containing "+36 30 123 4567".
- Üzenet*:** Text area.
- Számlát kér*:** Radio buttons for "igen" (selected) and "nem".
- Termék*:** Dropdown menu with "-- Válasszon --".
- Kiszállítás dátuma*:** Date picker showing "éé.éé. hh. nn.". There is a calendar icon to the right.
- Hírlevelet kérek:** Checkbox.
- Elküld** button.
- Footer:** "*: A csillaggal jelölt mezőket ne hagyja üresen."

³ <https://cloudinary.com/blog/build-a-custom-html5-video-player-with-javascript>



3. ábra: videóvezérlők működés közben

3.3 JQuery animációk, JavaScript

A többi elvárt tag (div, p, címsorok, képek, linkek) számos helyen megjelennek a weboldalon, több HTML fájlban is, főleg ott ahol hosszabb szövegek vannak (pl. cegunkrol.html). Képeket globális szinten a slider megvalósításánál használtam és a termékek megjelenítésekor.

Mindkét esetben fontos szerepet játszik a JQuery vagy a JavaScript. A slider például automatikusan cseréli a képeket, időzítve az alábbi egyszerű JavaScript kóddal:

```
<script>
    var myIndex = 0;
    carousel();

    function carousel() {
        var i;
        var x = document.getElementsByClassName("mySlides");
        for (i = 0; i < x.length; i++) {
            x[i].style.display = "none";
        }
        myIndex++;
        if (myIndex > x.length) {myIndex = 1}
        x[myIndex-1].style.display = "block";
        setTimeout(carousel, 2000);
    }
</script>
```

Ez a script „mySlides” osztálynevet bejárva setTimeout függvény segítségével időnként módosítja a kép CSS tulajdonságát.

A jQuery animációt a termékek.html-ben a „Bővebben” gombra kattintva tudjuk megnézni. Ilyenkor egyszerűen toggle metódussal be vagy kikapcsoljuk a termékleírást:

```
$(document).ready(function(){
```



```

    $(".bovebbenBtn").click(function(){
        var id = $(this).attr("rel");
        $("#leiras"+id).toggle("slow");
    });
});

```

Ugyancsak fontos szerepet játszik a JQuery az űrlapelemek ellenőrzésében. Nem teljesértékű gyakorlati megvalósítást írtam idő hiányában. Egyenlőre az űrlapelem ürességét ellenőriztem és a telefonszám esetében, hogy számokat írt-e be a felhasználó. Az email esetében van néhány vizsgálat még, amely ellenőrzi, hogy helyes-e az email formátum. Ennek ellenére azért lehetnek esetek, amikor helytelen adatbevitel történhet. Érdekes szerveroldalon is ellenőrizni a kapott adatokat. Az alábbi sorok az email mező ellenőrzését mutatják, amit a mező elhagyásakor futtatok (tehát nem küldéskor):

```

$("#emailIpt").blur(function(){
    var testEmail = /^[A-Z0-9._%+-]+@[A-Z0-9-]+\.[A-Z]{2,4}$/i;
    if (testEmail.test($(this).val()) && $(this).val()!="") $(this).css({borderColor:"green"});
    else $(this).css({borderColor:"red"});
});

```

3.4 JSON, AJAX

Ezt az elvárt technikát is az elerhetosegek.html fájlba helyeztem. Az órai munka alapján a telefonszámot, címadatokat egy külső .json fájlba helyeztem el az órán megismert formátumba tömbelemként. Majd JQuery AJAX segítségével betöltjük a fájl tartalmát és HTML elemeket hozzáadva az oldal betöltődésekor az #area azonosítójú div-be helyezzük.

A megoldás a load.js fájlban található:

```

$("#area").html("");

$.getJSON("elerhetosegek.json", function(data)
{
    $("#area").append("<b>Cím:</b>");
    $("#area").append("&nbsp;" + data.cim["iranyitoszam"]);
    $("#area").append("&nbsp;" + data.cim["varos"]);
    $("#area").append("&nbsp;" + data.cim["utca"]+"<br><br>");
    $("#area").append("<b>Telefonszám:</b><br><ul>");

    for(let i=0; i<data["telefonszam"].length;i++)
    {
        $("#area").append("<li>" + data["telefonszam"][i].tipus + "&nbsp;" +
data["telefonszam"][i].szam + "</li>");
    }

    $("#area").append("</ul>");

});

```


4. Fájl és mappaszerkezet

A projektet alkotó HTML fájlok a gyökérben helyezkednek el és a json fájl is. A JQuery számára külön mappát készítettem csakúgy mint a JavaScript és a képi elemek számára is. Gyakorlatilag igyekeztem az egyes technológiákat elkülöníteni külön könyvtárakba ahogyan az alábbi ábrán is látható:

Élő megtekintési link:

<http://legyenhonlapod.hu/WebProgK4WZH6/>

Github:

<https://github.com/matyastibor/K4WZH6Web-Konyv/tree/main/WebProgK4WZH6>

