**Montessori tornyok**

**Készítette:**

* Oláh Gergő
* Vasilescu András
* Gombos Benedek

[**Bevezetés** 3](#_Toc159849230)

[**Ötletszerzés/Találás** 4](#_Toc159849231)

[**Tervezés** 5](#_Toc159849232)

[**Megvalósítás** 6](#_Toc159849233)

[**Tesztelés** 8](#_Toc159849234)

[**Felmerülő problémák** 9](#_Toc159849235)

[**Összegzés** 10](#_Toc159849236)

[**Felhasznált irodalom** 11](#_Toc159849237)

# **Bevezetés**

Csapatunk újabb projektje egy kódbázisról választott feladat volt.

JavaScript feladatnak kellett lennie, ahol tudtuk alkalmazni az eddig tanul JavaScript, HTML, CSS tudásunkat egy nehezebb, moderatív feladatnál.

Főképp a JavaScript tudásunkat kellett előtérbe hejeznünk ebben a feladatban.

Több, mint 12 külön feladat közül tudtunk választani, melyek közül mindegyik lényege a weboldal készítés volt csak néhol a JavaScript, máshol pedig a HTML főbb tudása volt a prioritás, de az összes lényege az volt, hogy az eddig szerzett tudásunkat érvényesítsük egy haladó típusú weboldal készítési feladatban.

# **Ötletszerzés/Találás**

Az ötletünket a kódbázis oldalról kellett kiválasztanunk, mivel a feladat célja volt, hogy arról az oldalról egy kiválasztott minta alapján készítsük el a feladatunkat.

A csapatunk, először egy regisztrációs feladatot nézett, mivel kihívóan nézett ki és egy olyan oldalnak melyet nehezen, de megbírnánk csinálni, ámde ezt a feladatot már más csapat kiválasztotta így értelmetlen lett volna egy és ugyanazt a feladatot ketten csinálni.

Végül egy Montessori torony nevezetű oldal lett a választásunk, mivel senki nem akarta bevállalni nehézsége miatt, ezért döntöttünk úgy, hogy akkor majd mi megcsináljuk.

A példa nehézségét látván egy haladó szintű feladatnak tűnt, így minél nagyobb volt a kihívás annál többet tudunk belőle tanulni, ez volt a fő indok a választás és ötletkeresésnél.

# **Tervezés**

A tervezés kapcsán figyelembe vettük főképp a feladat nehézségét, mivel tudtuk, hogy a választott példa egy nagyobb szintű feladat, mint amivel eddig dolgoztunk.

Először leszögeztük az alapokat, kell egy CSS, egy JavaScript és persze a HTML oldalunk, 3 torony kellett, amit megadunk alapnak és ezek lesznek azok, amiket változtatni fogunk egy függvénnyel JavaScript kódban.

A 3 torony randomizáltan fog legenerálódni és úgy lekészítünk majd 3-9-ig blokkokat a tornyokon, ezeket vagy “Drag-on” egér mozgatással, vagy “On-click” kattintással lenne a blokkok mozgatása.

A Drag-on opció egy nehezebb, de látványosabb lenne a sima On-click opció meg egy egyszerű, de szimpla döntés lenne.

Tervezésnél meg előre megbeszéltük, hogy egy Excel fájlba leírjuk a hibákat és elvárt eredményeket ezzel megkönnyítve a jövőbéli munkánkat.

JavaScriptre tudtuk, hogy nagyobb figyelmet kell fektetnünk, így emiatt úgy terveztük, hogy a kódolás részét együtt csináljuk, de közben persze foglalkozunk a tesztdokumentációval is, így egyszerre vagyunk képesek két feladatot nyugodtan és megfelelően elkészíteni, két legyet egy csapásra.

Dokumentációra meg elég egy ember is, így azt csak egy személynek osztottuk ki.

# **Megvalósítás**

A feladatok kiosztása után jött, hogy milyen sorrendben haladjunk, először úgy döntöttük el, hogy elkezdjük és majd út közben kiderül mit prioritizáljunk, végül a legtöbb feladatot együtt végeztük.

Bővebben menjünk bele a részletekbe.

1. **Kódolás**

Kódolás során először értelemszerűen a HTML kódot kellett megírnunk, mivel, ha nincs alap oldalunk mire írjuk a CSS-t és JS-t? HTML kódot nem akartuk túlbonyolítani így csak az oszlopokat raktuk bele és egy alap indexet adtunk mindegyiknek, illetve a JS-t és CSS-t hozzákötöttük és lényegében a HTML része befejeződött.

Következő lépés a JS volt, megakartuk csinálni az alap folyamatokat, bár itt még dilemmába voltunk a generálás szempontjából. Gondolkodtunk a tervben megoszlott véleményen és randomizáltan oldottuk meg az oszlopok legenerálását, így jóval több választék volt arra, hogy hogyan rendeződnek a blokkok így talán könnyebb vagy nehezebb játékmenetet adva számunkra. Következő volt a blokkok úgymond “irányítása”. Tervben megbeszélt Drag-on ötlet tűnt a legjobbnak így az mellett döntöttünk, ám a megvalósítása már okozott némi problémát, mivel néhol az indexelt elemet nem észlelte a function, de ezt letudtuk egyből, mivel rossz index volt megadva egész végig.

Végül a JS kódunkat is befejeztük, sok feladattal és megmérettetéssel, de bebírtuk fejezni és léphettünk is az utolsó részre a kódolásban.

Utolsó és nem nagyon említésre méltó rész a CSS volt. A CSS-ben formáztuk meg a, oszlopokat, blokkokat és úgymond az egész oldalunkat. Hátteret adtunk neki, a blokkoknak és az oszlopoknak egy rendes formát és kinézetet, illetve kicsit jobban szétszedtük őket, hogy ne legyenek annyira közel egymáshoz így több tér legyen a mozgáshoz. Reszponzívvá tettük a weboldalt, hogy bármilyen eszközt is használunk, akkor is megfelelő formában láthassuk a “játékunkat”.

És így a kódolás résszel végeztünk.

1. **Dokumentálás**

Dokumentálás során 2 dokumentáción dolgoztunk miközben a kódolás zajlott.

A tesztdokumentáció Benedek és András írta főképp, mivel ahogy írták a kódot és ahogy javították a hibákat vagy netán kezelték őket úgy tudták közbe a “listába” írni azt, amit találtak és eközben meg íródott a sima Projekt dokumentáció is, de a tesztdokumentáció csak a végén került bele a főbb dokumentációba.

A Projekt dokumentációt 8 részre szedtük, hogy bővebben bele bírjunk menni a részleteke és kibírjuk fejteni azt, ahogy dolgoztunk. Bevezetéstől kezdve a lezárásig le akartunk írni mindent, hogy a folyamat is teljesen átlátható, illetve megérthető legyen. A dokumentációt kódolás közben is írtuk, ahogy említve lett, mivel sokszor előfordult, hogy változtatni kellett rajta, mert lehet más döntésünk lett, máshogy terveztük meg a dolgokat, vagy netán máshogy valósult meg a tervünk, így nem akartunk badarságot írni a dokumentációba ezért számtalanszol változott megvalósítás és még lezárás közben is.

# **Tesztelés**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case Name | The Tester | Test Step Desc. | Expected Result | Comments |  |
| Alap lefuttatás | Benedek | Oldal megnyitása | Index sikeres megtalálása / látása az oldalnak | Az oldal látta és sikeresen megtalálta az indexelt elemet hiba nélkül |  |
| Oldal lefuttatása | Benedek | Játék elkezdése (oldal elfuttatása) | A blokkok elrendezése és legenerálása | Az oldal sikertelenül generálta le a blokkokat, egyik alkalommal képernyőn kívülre, másik alkalommal rossz pozícióba |  |
| Megjelenítés | András | A blokkok megjelenítése | A blokkok elrendezése és legenerálása | Az oldal sikeresen legenerálta a blokkokat, a probléma a kódban a megadott x, y koordináta volt, random függvénnyel végül működött a program |  |
| CSS terv | Benedek | Formázás és elrendezés | Blokkok CSS-ben lévő elrendezése | A blokkok rossz formába jelentek meg, a piramis forma nem volt egyenletes |  |
| CSS terv | Gergő | Formázás és elrendezés | Blokkok CSS-ben lévő elrendezése | A blokkokat sikeresen legeneráltuk és sikerült úgy elkészíteni, hogy a blokkok a rendes piramis formát adják ki. A grid segítségével értük ezt el. |  |
| Beta Test | Gergő | Játék elindítása | A start gomb lenyomásakor a játék elindul és legeneráljuk az oszlopokat | Az oldal nem indult el és 404-es hibát iratott ki |  |
| Beta Test | Benedek | Játék elindítása | A start gomb lenyomásakor a játék elindul és legeneráljuk az oszlopokat | For i in range függvény helyett while loopot használtunk és így megfelelően elindult a játék |  |
| Mozgatás, Drag-on function működése | András | Blokkok mozgatása | Ha "megfogunk" egy blokkot akkor képesek vagyunk kicserélni egy másikkal | Sikeresen képesek voltunk mozgatni és cserélni az elemeket a játékan |  |
| Simítások | András | Többszöri lenyomásra mi történik | Többszöri lenyomásra a program csak egyszer fut le és nem folytonosan | Többszöri lenyomásra megjelenik több különböző torony |  |
| Simítások | Benedek | Többszöri lenyomásra mi történik | Többszöri lenyomásra a program csak egyszer fut le és nem folytonosan | Booleant csináltunk egy false értékkel és ha kap egy true értéket a while loopban akkor nem cserélődik vissza false-ra. |  |
|  |  | Megjegyzés: A táblázat csak néhány tesztelemet tartalmaz és főképp azokat amiket mi jelentősnek tartottunk, mivel újra bírtuk készíteni őket illetve, eléggé gyakori hibák voltak, de ennek ellenére beleraktuk őket, mivel volt 1-2 amivel megküzdöttünk rendesen. |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# **Felmerülő problémák**

A cím alatt, azokat a problémákat értjük, amik felmerültek, ám nem érdemeltek külön helyet a tesztdokumentumban.

Mind JS-ben, HTML-ben merültek fel hibák, amiket néha nem tudtunk re kreálni, ezzel nem készítettünk külön-külön részeket és lépéseket ezekhez a hibákhoz.

**JavaScript:** Felmerülő pár probléma a metódusokkal és a szerverrel volt, ahonnan ugye az adatokat kértük le.

Metódusnál elvárt eredmény volt, hogy a blokkokat úgy generálja le ahogy kértük mégis, volt számos alkalom, mikor a metódus nem csak nem generálta le, de ha le is akkor rosszul. A 3 blokk közül egyet sem generált le, vagy legenerálta csak képernyőn kívülre, úgy néz ki egy el írt kódsor okozta, de ezt nem vettük észre sok ideig, míg egy ellenőrző oldal mutatta ki nekünk.

A szerver meg főképp arra értetendő, hogy az oldal, amit használtunk nem funkcionált rendesen, vagy nem értük el szimplán.

Ritkábban, de felmerült, hogy HTML kódban lévő minimális elírás indexelésnél vagy adott tagnél kardinális pont volt a JS függvény működésekor, illetve sok hiba forrása is volt a rossz indexelés.

Így összegezve a felmerülő problémáink főképp azok a hibák voltak, ami felett sok irányításunk nem volt, ha meg volt is akkor nagyon minimális el írások voltak a problémáink okozói, minimális el írások voltak, mégis nagy hibákat készítettek.

# **Összegzés**

Összegezve a Montessori feladat egy haladó szintű és nehézkes, de megoldható feladat volt.

Sokat izzadtunk az egész feladattal, mivel többet kellett használnunk a JS és HTML tudásunkat, mint eddig, JavaScriptben formázni a blokkokat, megszabni, hogyan keletkezzenek, irányításuk.

A csoport remekül dolgozott összhangban és ha bármi baj is volt, vagy netán valamelyik csapattagunk elakadt volna, egyből segítettünk egymásnak. A beosztott feladatát mindenki elvégezte és a legjobb tudása szerint megcsinálta, még akkor is ha nem volt hatalmas tapasztalat az adott feladathoz, mégis remekül befejezte mindenki az adott feladatot.

Tesztelésnél, kód írásnál is az együtt dolgozás, összhang és a kommunikáció volt a kulcs a tökéletes előre haladáshoz.

**Pros:** Az összhang tökéletes volt, a csapatmunka főbb részei teljesültek, ha a kiadott feladatokra értjük a csoport úgy dolgozott, ahogy egy csoportnak kellett. A folytonos kommunikáció, megadott hibákat leírtuk, közöltük egymásnak, a internetes felületet rendszeresen nézte mindenki, hogy ne legyen bármi le maradás vagy félreértés, a csoport munkája kiemelkedő volt a társalgási szintben és remek volt a feladat megoldására értetendő szinten.

**Cons:** A csoportnak, viszont volt némi problémája, amibe beleütközött. Kommunikáció terén néhány félnek nem volt meg a megfelelő eszköze vagy ideje arra, hogy a megbeszéléseket megtartsuk, ez nem lényeges hiba, de ha negatívat kell írni a csoport munkájáról csak ez tud eszünkbe jutni.

# **Felhasznált irodalom**

<https://codepen.io/mateusPaixao/pen/abvyjJR>

[ondrag Event (w3schools.com)](https://www.w3schools.com/jsref/event_ondrag.asp)

[How To Create a Draggable HTML Element (w3schools.com)](https://www.w3schools.com/howto/howto_js_draggable.asp)

[HTMLElement: drag event - Web APIs | MDN (mozilla.org)](https://developer.mozilla.org/en-US/docs/Web/API/HTMLElement/drag_event)