*Tatabánya Zoo Dokumentáció*

# **Felépítés, oldalszerkezet**

Mindegyik oldal Bootstrap v5.3 külső könyvtárra épül. Az oldal a .container-fluid faltól-falig tartó Bootstrapes osztállyal felruházott <div> elem tartalmában van.

Az oldal több eszközön tesztelt, és reszponzív, például iPhone SE, iPhone XR, Samsung S8+, Samsung S20 Ultra. Természetesen ezeken kívül közel minden felbontáson egy tökéletes felhasználói élményt nyújt, és minden funkciója működik.

A reszponzív nézetet a Bootstrap col rendszere biztosítja. A nézetek úgy változnak, ahogy a képernyő mérete csökken vagy nő, eszerint arányosan (százalékosan) változnak az elemek és az őket tartalmazó konténerek. Azokat az eseteket, amiket a Bootstrap nem helyesen oldott meg, media query-kel kezeltük.

Mindegyik oldal HTML5-ös elrendezés szerint épül fel, és egyéni témát követnek – miszerint azonos a fejléc, lábléc tematikája – minimális eltérés a tartalmuk, például a főoldalon az aloldalak gombjai vannak csak ott, amíg az aloldalakon a főoldalé és jelen van. Társul a tematikához az állandó háttér és logó.

A fejléc egyszerűen épül fel, a Bootstrap alapvető oldalról beugró menüjét használtuk egy egyszerű fejléc felépítéssel megspékelve. A <header> tag közé esik a <nav>, amin .navbar és .navbar-dark keretrendszeri osztályok vannak. A <nav> tartalma faltól falig megy a ráadott .container-fluid osztály miatt, ami a főoldalon 3 linkből, és a keretrendszerbeli hambugermenüből áll.

A hamburgermenü minden oldalon jelen van, és lenyomva jobb oldalról beugrik egy sáv, amin tájékozódási linkek vannak.

A legtöbb ikon a bootstrap icons könyvtárat használja.

Kisebb képernyőkön, a reszponzív nézet szerint a cél mindig az volt, hogy mindent megtudjunk jeleníteni, ebből kifolyólag egymás alá törnek az elemek, ha nincs elég hely.

Az összes szöveg font-style-ja a <https://fonts.google.com/> -ról van.  
Jelenestben a Bungee Spice-ot használtuk, mint séma vezető – ez a sárga átmenetes szöveg képzi az összes címet az oldalon. A többi elem, például a paragrafusok Kanit-tal vannak megjelenítve.

**Főoldal**

Az weblapra érve egyből látható a fent említett fejléc, és egy slideshow. A slideshow a keretrendszerbeli mozgó, gombos verzió. A fejléc vonaláig megy a .container-fluid osztálynak köszönhetően. Három kép váltakozik ~ 6 másodperces váltakozással.

A slideshow, avagy slider osztály egy divre van rárakva, társítva a .container-fluid osztállyal. Ezt egy másik div követi, amin a carousel slide osztály, és a carouselExampleIndicators id, illetve a data-bs-ride attribútummal van ellátva.

A carousel slide osztályok azonosítják a keretrendszer számára a slideshowt, az id pedig az elérésért felel. A data-bs-ride a slideshow „elindítását” végzi.

A következő elem egy <div>, amin a carousel-indicators osztály van, ez tartalmazza a navigáló gombokat. Ezek alap <button> tagek, type „button”-al ellátva. A rajtuk lévő data-bs-slide-to”x” a következő képre való automatikus léptetést biztosítja, ha nem mi irányítjuk.

Ezt követően a képet egy carousel-inner osztályú div jön, ez a tartalmat jelzi a keretrendszer számára. Ebben egy abszolút pozíciójú <h4> van, amely a képeken mindig ugyanott jelenik meg.

Az utolsó elem az egy <img>, amin d-block, w-100 és kizárólag a 2. képen „giraffeTop” osztályok vannak. A d-block az elemek egymás alá való törést biztosítja, a w-100 pedig a szélesség teljes, tehát 100%-os kihasználtságát biztosítja. A giraffeTop a kizárólag a második képen megjelenő osztály pedig a megjelenített kép tetejét mutatja, hogy a zsiráf feje tisztán kivehető legyen.

Az oldal további részei sorokra vannak osztva, amik oszlopokon épülnek fel, a fent említett col rendszerrel.

Lentebb görgetve láthatunk egy <h3>-as elemet, ami a felhasználót a látogatása megtervezésére buzdítja. Ez gyakorlatilag egy <section>, amin egy .row osztály van a keretrendszerből. Egy nagy oszlop az egész <section>, a cím alatt egy egyszerű <p> egység van, benne egy kis leírás.

A következő sor 3 egyenlő oszlopból áll, itt minimális információ található, egy Google maps integrációval. Ez egy iframe, ami a Google mapsnél generált linkkel hivatkozik.

A másik két oszlop kis információt tartalmaz, és két gombot, amik különálló modalokra mutatnak. Ezek a modallok egy kékes hátteren lévő kis információt hoznak fel.

A modallok a keretrendszerbeli alapvető osztályokkal rendelkezik, hasonlóan a sliderhez.

A következő szakasz 4 oszlopból áll, ami az oldal lakóit mutatja be nagyon egyszerűen.  
Szemléltetésként megjelenik 4 állat, aki reprezentálják az alattuk található fajokat. Mindegyik faj 3 col egységet kapott, amiben egy article van, abban egy figure tag, ami tartalmazza az <img> és <figcaption> tageket.

A Főoldal utolsó tartalmi szekciója az állatkertben történő 3 hír előnézetét tartalmazza,

amik egyenlően 4 col egységre vannak tagolva. A tartalom mindhárom esetben <article>

tagok között van, amit követ egy <img>, egy kiscím (h5), dátum, a hír címe és egy gomb, ami a hírhez vezet.

Ezt követi az utolsó nem tartalmi szekció, ami csak egy szimpla gomb, ami felvisz az oldal tetejére.

Mint legutolsó rész, a <footer> (lábléc) szintén egy sor, amiben 2 oszlop van, 6 col egységgel elosztva. Az oszlopok tartalma informatív, egyszerű elérhetőségeket tartalmaz.

# **Hírek aloldalak**

Az aloldalak követik a főoldal sémáját, tehát egyezik a <footer> és a <header>.

Három <section> van az egész oldalon, amiben mindkét <article> 12 col egységgel van felruházva, tehát a teljes területet kitölti. Az articlenek van egy halvány fekete (#000) háttere, egy minimalista borderrel. (solid 1px #000) Ez körbe öleli a tartalmat, ami egy divben lévő kép, és a diven kívüli cím, paragrafusok és a főoldalhoz hasonló tetejére vezető gomb.

# **Jegyárak aloldal**

Akárcsak az összes megjelenő aloldal, a jegyárak is követi az oldal témáját és rendszerét, tehát ugyan úgy megegyezik a <header> és a <footer>.

Az oldalra térve egyből látható egy táblázat, ami a jegyárakat tartalmazza, a kedvezményekkel együtt. Ezalatt található egy tájékoztató sáv, ami a bankkártyás fizetésről tájékoztat. Az oldal legutolsó tartalmi eleme pedig az online jegyvásárláshoz továbbít, ami tájékoztat az online jegyárakról. Itt található egy gomb, ami a vásárlást lebonyolító oldalra vezet át.

A táblázat egy articleben van benne, ami az összes col egységet használja. A keretrendszer .table-responsive osztályát használtuk, ami egy reszponzív táblázatot készít nekünk. Ez az osztály egy divre van rárakva, felépítése a táblázatnak hasonló,

<table> tagre van rárakva a .table, .table-hover keretrendszerbelii osztályok.

Ezek a bootstrap számára jelzik a táblázatot, és ezalapján kerül a stílus rá az elemeire.

5 sor van, mindegyikben 3 oszlop. Az első sor egy <tr>, tehát <thead> -ben van benne. A maradék négy viszont lábléc nélküli, <tbody>-ban lévő sima <tr> tagekben lévő <td> elemekből áll.

Ezután jön egy <section>, ami tájékoztat a bankkártyás fizetésről, tartalma szimplán egy <article>, amiben egy <p> van, benne az információval.

Legutolsó tartalmi szekcióként pedig az Online jegyvásárlás van, ami 3 egyenlő oszlop, ezekben piktogramok szerepelnek az online árakkal feltüntetve. Középen pedig egy keretrendszerbeli gomb, ami a jegyvásárlásra készített oldalra vezet.

# **Online jegyvásárlás aloldal**

Az eddigi aloldalakkal gyakorlatilag megegyezik, annyi különbséggel, hogy a tartalmi egység nem megy ki faltól falig, tehát nem egy .container-fluid, hanem egy .container osztállyal rendelkezik, ami csak a képernyő ~80%-át használja fel. Ebben egy szimpla szekció van, benne a címmel, és egy keretrendszeres validációs kitöltendő formmal.  
A form alatt van két gomb, ami a vásárlás és a Vissza! gomb, ami a jegyárak oldalra vezet vissza.

# **Támogatás aloldal**

Relatív különleges az alábbi aloldal, hisz a többitől eltér a szöveg rendezési iránya, ugyanis itt text-align:left, tehát balra igazított szöveggel dolgoztunk desktop nézeten – ugyanis ez telefonon az olvashatóság miatt középre igazított. A szöveg igazításától függetlenül a séma változatlan, a lábléc, fejléc és a fel a tetejére gomb egyeznek.

Az első <section> egy keretrendszerbeli sor, tehát elemei oszlopba tagolandóak.

A szekció oszlop tagolása különleges abból a szempontból, hogy az <article> ismét 12 col egységet tartalmaz, viszont azon belül megjelenik egy új sor, és ismét elosztódnak a col egységek. A cím (h2) alatt található egy <p>, ami ismeretterjesztő célt szolgálva bemutatja az állatkert támogatásának fontosságát.

Ezután egy alcím jelenik meg, ami egy <h3>-as tagben van, és felhívja a felhasználót arra, hogy Ő is adományozzon. Egy az online jegyárakhoz hasonló form tekinthető meg a <h3> tag alatt. Bekéri a felhasználótól nevet, emailt, az adomány összegét, és a bankkártya információkat. Lehetőség van a hírlevélre való feliratkozásra is.

Ezalatt a kitöltendő form alatt más támogatási módokat sorol fel al az oldal egy új szekció keretében, például az örökbefogadást, és az önkéntességet. Ezek alatt modalhoz vezető gombok vannak, melyek informcáiót jelenítenek meg rányomva a gombra.

Ezután az önkénteséggel kapcsolatos beszámolók jönnek, amik egy külön szekció sorában lévő 3 oszlopra tagolt rész, mindegyik 4 col egységet elfoglalva. Egy <h4> mint cím, és egy paragrafus a tartalma ezeknek a 4 col egységeknek, amik egy <article> tagra vannak rárakva. Itt ismerteti pár önkéntes beszámolóját a programról.  
Legvégül megjelennek a támogatók logói, a Mi Tatabányánk, Városunk Tatabánya, Tatabányai Járási Egyesített Szocális Intézmények, Bánki TSZC, és a Gerecse program.

# **JavaScript magyarázat**

Az első függvényünk a changeClass()  
A képen szöveg, képernyőkép, képernyő, Betűtípus látható

Automatikusan generált leírás

Ez a függvény a hírek oldalakon használatos, ahol a nagyobb kijelzőkön való lábléc helyét biztosítjuk. Erre azért volt szükség, mert egy nem hosszú, viszont magas monitoron a lábléc irreálisan közel volt a szöveghez, így viszont megfelelően simul, ha nagy a képernyő, és akkor is ha kicsi. A függvényünk során inicializálunk egy „footer” változót, aki a changeFooter osztályt keresi a dokumentumban. Ellenőrzi az ablak szélességét a window.innerWidth tulajdonsággal. Ha az ablak szélessége kisebb, mint 993 pixel, akkor eltávolítja a fixed-bottom keretrendszer osztályt, amely rögzíti a böngésző aljára fixált pozicíóba a láblécet. Ezzel ellentétben a sticky-lg-bottom akkor mutatja a láblécet az egész szélességen, ha túl görgetünk azon.

A JavaScript következő része noha nem függvény, viszont említendő darabja:  
A képen szöveg, képernyőkép, Betűtípus, képernyő látható

Automatikusan generált leírás

Ezt a részt a Bootstrap szolgáltatja, és a validációt ellenőrzi a formoknál.

Az első sor ((() => { gyakorlatilag azonnal lefut az oldal betöltésekor, az önjáró függvcény kezdetét és végét jelenti.

’use strict’ a kód szigorú üzemmódban fog lefutni, emiatt kiemeli és kijavítja a hibákat azáltal, hogy szigorúan fut. Tiltja az implicit deklarációt.

`conts forms = document.querySelectorAll(’.needs-validation’)` sor pedig kiválasztja az összes HTML elemet amelyek a needs-validation osztállyal rendelkeznek, és a forms konstansba menti őket. Ezek azok az összes forms gyakorlatilag, amit használtunk. Lásd.: Támogatás aloldal, Jegyárak online aloldal.

Array.from(forms).forEach(form => {…} sor Array.form(forms része a HTMLCollection (A lista, amit a forms-al kaptunk vissza a QuerySelectorAll miatt.)

Ezt így tömbbé alakítja, és a forEach ciklussal végig megy az összes űrlapon.

form.addEventListener('submit', event => {...}, false) sor pedig beállít egy eseményfigyelőt, az űrlapok „submit” tehát küldés eseményére, amikor gyakorlatilag a gomb lenyomása történik. Ezután ellenőrzi hogy érvényesek-e az űrlap adatai a checkValidity() függvénnyel. Ha nem érvényes, akkor event.preventDefault() megakadályozza a beküldést.

event.stopPropagation() megakadályozza a más eseménykezelők futását.

form.classList.add('was-validated'): Hozzáadja az "was-validated" osztályt az űrlaphoz, amely a Bootstrap keretrendszerben gyakran használatos, és jelzi, hogy az érvényesítés már megtörtént. Ezzel a stílusok vagy jelzések megjelenhetnek az érvényesítési eredményekkel a felhasználók számára.

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

Ez a függvény az első sorában kiválasztja a .needs-validation osztályú elemeket.  
Ezután ellenőrzi, hogy az űrlap érvényes inputtal rendelkezik-e, és hasonlóan az előbbi részlethez, ha nem érvényes, akkor event.preventDefault() megakadályozza a beküldést.

event.stopPropagation() megakadályozza a más eseménykezelők futását.

Abban az esetben, ha az űrlap érvényes, az else ágon lefut az alert(’Sikeres fizetés’);

ami tájékoztatja a felhasználót a vásárlás sikeréről.

form.classList.add('was-validated'); hozzáadja a was-validated osztályt az űrlaphoz, ami a Bootstrap számára jelzi a helyes validációt.

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

Az alábbi függvény hozzáadja az overflow-os osztályú elemekhez a padding 0 és Y axisú görgetést, ami kiküszöböli a modalok megnyitásánál lévő háttér mozgását.

A képen szöveg, képernyőkép, Betűtípus látható

Automatikusan generált leírás

Az alábbi kódrészlet első sora létrehoz egy max nevű változót, ez lesz a for ciklus futásának hossza. A select változóba tároljuk a dokument <select> elemét.

Ezután kezdődik az iteráció, ami nullától indul, és 99-ig megy egyesével. Minden futás során létrehoz egy új <option> taget, és az i értékét adja meg neki, és az innerHTML, tehát a belsejébe is az I kerül. Ezután hozzáadja a generált <option> taget a selectünkbe.

A képen szöveg, képernyőkép, Betűtípus, szoftver látható

Automatikusan generált leírás Ez a függvény egy eseménykezelőt (event listener)-t tartalmaz, amely a legördülő menü (<select>) értékének változásakor frissíti a kiválasztott érték alapján kiszámolt jegyárakat egy HTML elemen keresztül.

A legelső három sorban hozzáadunk egy event listenert a ticketSelect id-vel rendelkező elemhez akkor, ha változik az.  
Ezután jön a funkció, amely kiszámolja a jegyárat miután deklarálta a select értékét a documentből, létrehozza a calculatedPrice változót, ami a selectünk értéke beszorzása a felnőtt jegy árával. Ezután azonosítja a checkOut osztályú <p>-t, és beleírja a tartalmába a végösszeget.