

Budapesti Műszaki Szakképzési Centrum
Neumann János Informatikai Technikum
Szakképesítés neve: Szoftverfejlesztő
OKJ száma: 54 213 05

ZÁRÓDOLGOZAT

Boo-T: Bootstrap 4 fejlesztő IDE és framework

Várkonyi Tibor Zoltán
konzulens

Fehér János Zoltán
14. rsze

Budapest, 2021.

Tartalomjegyzék:

| | | |
|-------------|--|------------|
| I. | <i>Bevezető, a probléma rövid ismertetése</i> | 1 |
| II. | <i>Választott téma indoklása</i> | 2 |
| 1. | <i>A Bootstrap 4 keretrendszer bemutatása.....</i> | 2 |
| 2. | <i>Az alkalmazással elkészíthető weblapok alapelvéi</i> | 3 |
| 3. | <i>Az alkalmazással elkészíthető elemek kiválasztási szempontjai.....</i> | 7 |
| III. | <i>A téma kifejtése, fejlesztői dokumentáció</i> | 9 |
| 1. | <i>A programnyelv választásának indoklása</i> | 9 |
| 2. | <i>Fejlesztőkörnyezet.....</i> | 11 |
| 3. | <i>Felhasznált függvénykönyvtárak</i> | 12 |
| 4. | <i>Futtatáshoz szükséges külső alkalmazások.....</i> | 17 |
| 5. | <i>Könyvtárszerkezet</i> | 17 |
| 6. | <i>A program felépítése</i> | 20 |
| 1. | <i>Deklarációs szakasz, az osztályok létrejötte, adatok betöltése.....</i> | 20 |
| 2. | <i>A createAll szál.....</i> | 21 |
| 3. | <i>HighLighter szál.....</i> | 31 |
| 2. | <i>Felhasználói interakció szakasza</i> | 35 |
| 1. | <i>Fájlműveletek.....</i> | 35 |
| 2. | <i>Vágólap műveletek</i> | 39 |
| 3. | <i>Kódgenerálás, tesztelés</i> | 40 |
| 1. | <i>Python alapú fordító.....</i> | 42 |
| 2. | <i>Fortran alapú fordító.....</i> | 51 |
| 3. | <i>HTML környezet generáló script</i> | 75 |
| 4. | <i>Egyéb függvények.....</i> | 79 |
| 5. | <i>ListBox elemek</i> | 89 |
| 3. | <i>Program lezárása</i> | 90 |
| 4. | <i>Tesztelés</i> | 94 |
| 1. | <i>Altair 8800.....</i> | 95 |
| 2. | <i>Sivatagi Róka.....</i> | 98 |
| 3. | <i>Help.....</i> | 100 |
| 5. | <i>Fejlesztési lehetőségek.....</i> | 108 |
| IV. | <i>Felhasználói dokumentáció.....</i> | 108 |
| 1. | <i>A program szintaxisa</i> | 108 |
| 2. | <i>Felhasználói felület.....</i> | 116 |
| V. | <i>Összegzés</i> | 122 |
| VI. | <i>Irodalomjegyzés, hivatkozási jegyzék.....</i> | 124 |

I. Bevezető, a probléma rövid ismertetése

Közhely és bizonyára az olvasó számára sem szolgál újdonságként, hogy életritmusunk, feladataink teljesítése, tanulmányaink elsajátítása és szabadidőnk eltöltése is egészen más hatásfokkal történik, mint generációkkal ezelőtt, ez pedig nem csak a munkavállaló, hanem az ő számára eszközöket fejlesztő szakember elő is egészen új kihívásokat állít.

A korábban hosszú éveket, nem ritkán évtizedeket felölélő technológiai- és szoftveres fejlődést dinamikus- és napi szintű előrelépések jellemzik és ehhez igazodnak a megrendelők elvárásai is, akiknek bevételét nagyban meghatározza, hogy a fejlesztők által elkészített felület milyen minőségű szolgáltatást képes nyújtani és megjeleníteni. A szoftverfejlesztés igen messzire jutott a korai, analitikus feladatokra koncentráló jellegétől, a tevékenységek jelentős része az offline számítógépekről és helyi hálózatokból átkerült az online térbe; felhőalapú szolgáltatások, adatbázis lekérdezések és bővítések, folyamatos kommunikáció jellemzi, mely olcsó, energiatakarékos távoli hozzáférést biztosít¹, így a fejlesztők kellő minőségű munkát tudnak belátható időn belül, vállalható mértékű juttatásokért előállítani². A fejlesztés maga pedig ritkán kezdődik az alapoktól, többnyire félkész alkalmazások, protokollok, eljárások összekapcsolása és üzemeltetése zajlik. Ennek egyik oka, hogy ma a fejlesztők csapatban dolgoznak, a rendelkezésre álló idő szűkössége, a probléma komplexitása és a további fejleszthetőség, akár egy új csapat által megköveteli, hogy bizonyos keretrendszer segítségével történjen a fejlesztés³.

A modern webfejlesztés egyik kulcseleme a Bootstrap, mely nélkül ma igen nehezen képzelhető el dinamikusan változó, a környezethet, elsősorban a képernyőmérethez adaptálódó weboldal, mely a kisebb okoskészülék kijelzőktől a ma elérhető legnagyobb képernyőkig ugyanúgy megnyerő, látványos, nem utolsósorban a fogyasztók számára eladható felületet képes megjeleníteni. Ezen felül ez a csomag tartalmaz olyan előre definiált elemeket, mint pl. navigációs sáv, gombok, legördülő menük, bemeneti mezők, stb., melyek használata jelentősen meggyorsítja a weboldal frontend fejlesztését és ezen felül az elemek megjelenésbeli összhangban is vannak egymással.

¹ ALI. H. DOGRU: Modern Software Engineering Concepts and Practices: Advanced Approache, Middle East Technical University [Turkey], 2010, 383. o.

² MARIA DEL PILAR SALAS ZARATE, GINER ALOR-HERNÁNDEZ, RAFAEL VALENCIA-GARCIA, LISBETH RODRIGUEZ: Analyzing best practices on Web development frameworks IN Science of Computer Programming - 112 [2015], 1-19. o.

³ "The task of the software development team is to engineer the illusion of simplicity." (Grady Booch)

Viszont nem mehetünk el azon tény mellett, mivel a Bootstrap alapvetően a HTML technológián alapszik, a weblap elemeinek elkészítése az ennek megfelelő szintaxison keresztül történhet, mely elsődleges motivációt adta az alkalmazás elkészítéséhez, mivel számomra rendkívül fontos a letisztult, tömör, egyszerű nyelvezet, mely lényegesen kevesebb hibalehetőséget és könnyebb kezelhetőséget biztosít a fejlesztő számára. A cél irányából megközelítve, a következő fejezetben a Bootstrap keretrendszer és jelen alkalmazás céljából releváns elemeinek bemutatására kerül sor.

II. A választott probléma indokása

II/1. A Bootstrap 4 keretrendszer bemutatása

A Bootstrap Mark Otto és Jacob Thornton alkotásaként eredetileg a ma már közismert, főleg az Egyesült Államokban és a Távol-Keleten (Japán, Hong Kong, Taiwan) népszerű, rövid szöveges üzenetek (kezdetben 140, ma 280) posztolására hivatott közösségi portál megjelenítését szolgálta⁴. A fejlesztők igen korán szembesültek vele, hogy a fejlesztés, karbantartás során a rengeteg ellentmondásos elem nehezen áttekinthető összefűzése értékes többletenergiákat von el a csapattól, ezért egy olyan egységes szerkezetet kívántak a teljes webes felületre alkalmazni, mely a tartalmazza a weblap elemek megfelelő stíluslapi tulajdonságait (a weblap elemeihez osztályokat rendelünk a kívánt megjelenés eléréséhez) és alapvető Javascript függvények segítségével biztosítja a weboldal responzivitását és alkalmazkodását a megjelenítő képernyőhöz.

A rendszer felépítésekor a könnyű integrálhatóságra is sok figyelmet fordítottak, mivel a működéshez feltétlen fontos CSS és Javascript elemeket importáló alapvető HTML template a weboldalról könnyen átmásolható, a különféle verziók pedig letöltés és helyi hivatkozás nélkül importálhatóak, így az új verzió megjelenésekor nem válik szükségessé tűzoltás jellegű karbantartás.

A Bootstrap elemvető jellemzői a következők⁵:

a) A HTML5 dokumentumszabvány köré építettség

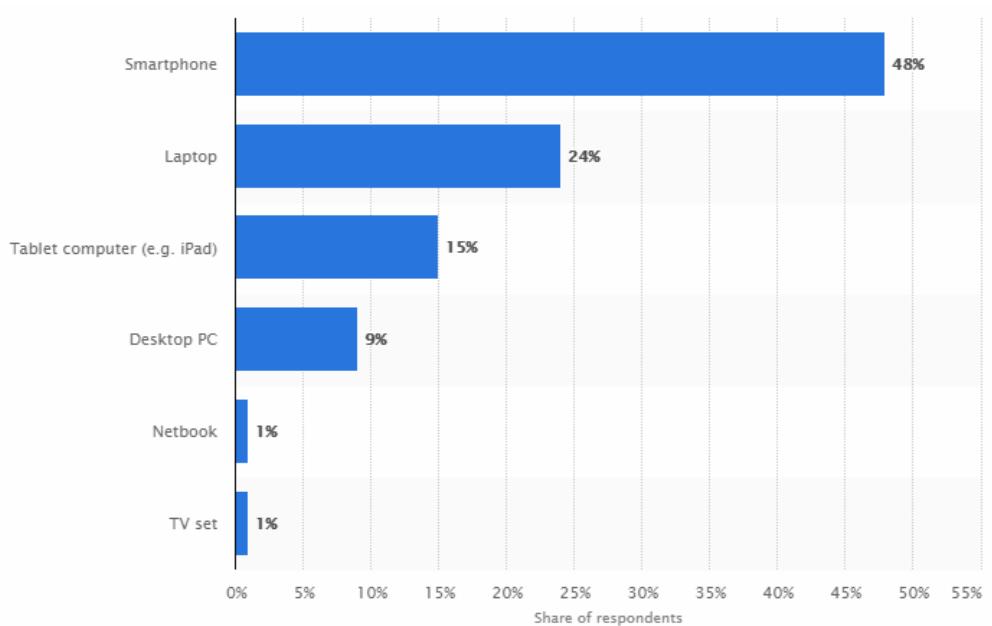
A Bootstrap legfrissebb változata a legújabb HTML szabvány segítségével kezelhető.

b) "Mobile First"

Mivel ma a webes böngészés közel fele okostelefonok segítségével zajlik, a Bootstrap fejlesztői ehhez alkalmazkodva kiemelt figyelmet szenteltek a kisebb képernyőkön való megjelenítés optimalizálására.

⁴ MARK OTTO: Bootstrap from Twitter, URL: https://blog.twitter.com/developer/en_us/a/2011/bootstrap-twitter.html, megjelenés dátuma: 2011.08.19.

⁵ TUTORIALSPOINT: Bootstrap Tutorial [PDF], URL: <https://wiki.lib.sun.ac.za/images/0/07/Bootstrap-tutorial.pdf>, 26. o., letöltve: 2020.10.21



1. ábra: Internetes böngészéshez elsődlegesen használt eszközök megoszlása az Egyesült Királyságban, 2018⁶

c) Reszponzív képek

A képek kitöltik a keretet, melybe helyezték őket.

d) Szabványos megjelenítés

Színek, betűtípusok és linkek megjelenésének szabványosítása.

e) Nagyfokú kompatibilitás

A weboldalnak közel azonosan kell megjelenítődnie azonos képernyő-tulajdonságok mellett, függetlenül az azt megjelenítő böngészőtől.

f) Konténerekbe rendezettség

A keretrendszer a tartalmakat 12 oszloból álló konténerbe helyezi el, melyben képernyőméretekkel függően egyénileg beállítható, hogy adott szélesség mellett hány oszlopot foglaljon el.

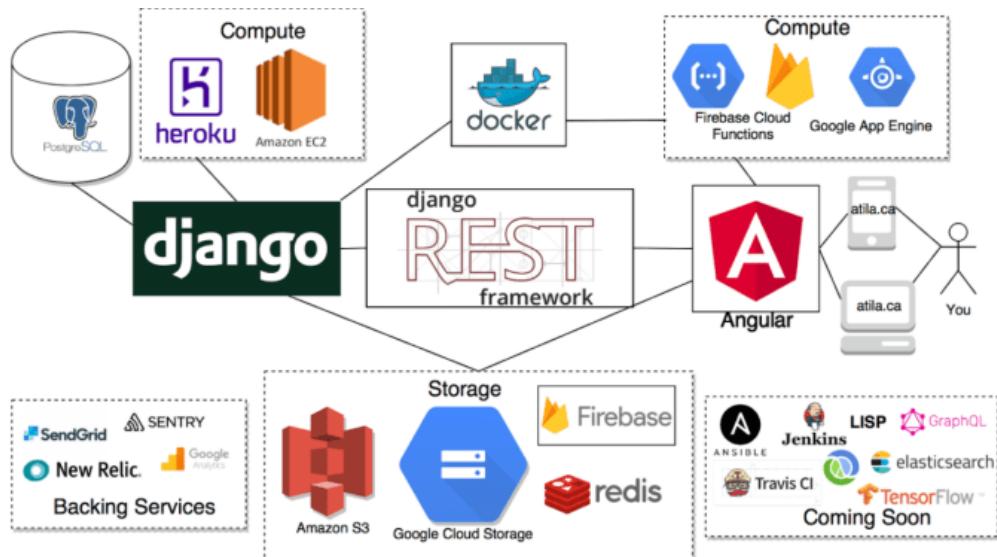
II/2. Az alkalmazással elkészíthető weblapok alapelvei

A Boo-T alkalmazás elkészítésekor elsődleges szempont volt a célcsoport kiválasztása, ennek mérlegelésével határoztam meg a segítségével elkészíthető webes tartalom összetettségét, a kezelőfelület részletezőségét és a program fordítói által alkalmazott szintaxis jellemzőit, formáját.

Jelenleg számtalan professzionális webfejlesztő keretrendszer van használatban, melyek lényegesen leegyszerűsítik a teljesítmény- és sebességorientált fejlesztői munkát,

⁶ JOSEPH JOHNSON: Which one of these devices do you use most for surfing or browsing the internet, URL: <https://www.statista.com/statistics/308449/device-preference-for-internet-browsing-in-the-uk/>, megtekintve: 2020.11.02

valamint a hibajavítást, továbbfejlesztést és áthelyezést is sokkal könnyebbé teszik, ezen felül tekintettel kell lenni a platformfüggetlenségre is.



2. ábra: Egy lehetséges szoftveralkalmazás UML projekttervezés során⁷

A fenti ábrán jól látható, hogy egy viszonylag egyszerű weboldal elkészítéséhez és üzemeltetéséhez is számtalan keretrendszer ismerete javasolt, viszont képzeljük bele magunkat egy kezdő webfejlesztő helyzetébe, aki a fent látható ábrával szembesül: Elég nagy eséllyel az az érzés keríti hatalmába, hogy ezt ő sohasem fogja megérteni. De a kérdést megfogalmazhatjuk egy másik szemszögből is: Vajon minden egyes reszponzív weboldal, mely elsődlegesen a frontend oldalon domborít, igényli-e a fent látható igényességet és körültekintést?

A Boo-T pontosan ebből a szempontból kíván segítséget nyújtani, mivel egy rendkívül letisztult és egyszerű nyelv segítségével képes komplett weboldalak legenerálására és célkitűzésként jelent meg, hogy minden egyetlen HTML dokumentumban helyezzen el, beleértve a stílusjegyeket és javascript elemeket. Jól látható, hogy a Boo-T egyáltalán nem kíván versenyezni az olyan elterjedt és összetett szoftverekkel, mint pl. az Angular, a vue.js, docker vagy django, hanem egy kezdők, tanulók számára is könnyen érthető alternatívát kíván nyújtani.

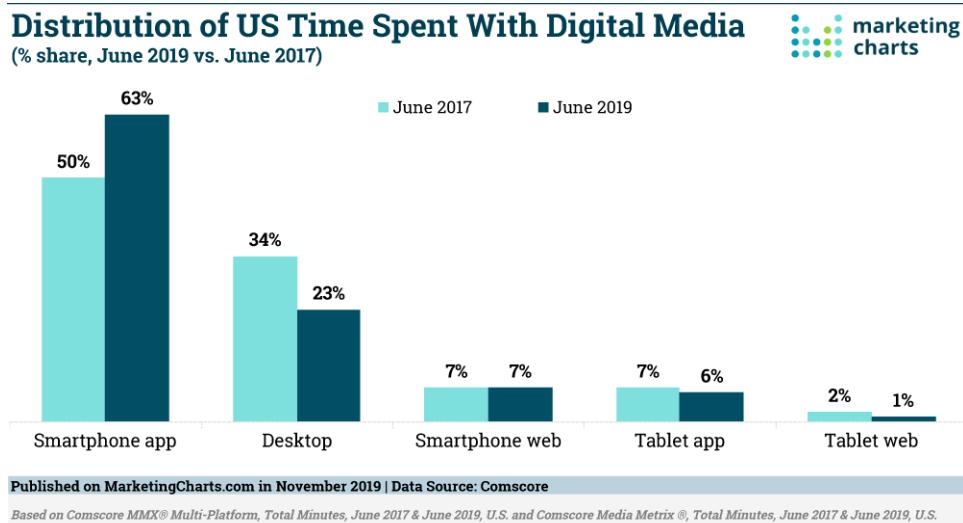
⁷ TOMIWA ADEMIDUN: Why We Chose Angular over React and Django Over Ruby on Rails: How to Choose A Software Startup, URL: <https://hackernoon.com/why-we-chose-angular-over-react-and-django-over-ruby-on-rails-for-atila-ca-77ac03d542cf>, megtekintve: 2020.12.28.

Az alkalmazás megtervezésekor az Angular szolgáltatásai⁸ kerültek előtérbe, mivel elterjedtsége ellenére viszonylag egyszerű, mobile-first weblapok elkészítését teszi lehetővé.

A Boo-T az alábbi webfejlesztési alapelveteket alkalmazza:

- Egyoldalas felépítés**

Szinte webfejlesztési axiómának tekinthető, hogy a fogyasztási szokások jelentősen átalakultak az elmúlt 10 évben a mobil eszközök megjelenésével, így a frontend fejlesztőnek különösen ügyelnie kell rá, hogy a tartalom a kisméretű, forgatható, ezáltal változó szélességű kijelzőn is tökéletesen jelenjen meg.



3. ábra: Internetes szörfölési szokások változása, használt eszköz alapján⁹

Ennek ismeretében lényegesen előnyösebbek az egyetlen weblapból felépülő weboldalak, melyek nem igényelnek összetett, nehezen visszakövethető navigációt.

- Ésszerű görgetés**

Az érintőképernyős vezérlés, lényegében az egér görgőjével megegyezően a függőleges irányú görgetésre van optimalizálva, a horizontális irányú mozgást minden esetben kerülni kell¹⁰. A legtöbb weboldal a márkaeljést, a bannert és a navigációs sávot helyezi el a weboldal felső részében¹¹, a weblap alján egy impresszumot és social link-eket tartalmazó footer található és a kettő között érhető el a tulajdonképpeni tartalom, melyen a

⁸ Angular Docs, URL: <https://angular.io/docs>, megtekintve: 2020.12.28.

⁹ Smartphones Now Account for 70% of US Digital Media Time, URL: <https://www.marketingcharts.com/digital/mobile-phone-111093>, Megtekintve: 2020.12.28.

¹⁰ NICK BABICH: The 12 Do's and Don'ts of Web Design, URL: <https://xd.adobe.com/ideas/principles/web-design/12-dos-donts-web-design-2/>, megtekintve: 2020.12.28.

¹¹ ANDY CRESTODINA: Web Design Standards: 10 Best Practices on the Top 50 Websites, URL: <https://www.orbitmedia.com/blog/web-design-standards/>, Megtekintve: 2020.12.28.

végiggörgetés nem tarthat túl hosszú ideig, mely a weboldal idő előtti elhagyását okozhatja¹².

- **Egyeszerű navigáció**

Az asztali gépekre és nagy képernyőkre tervezett weblapoknál jól mutat egy összetett és sok elemet tartalmazó navigációs menü, ezzel ellentétben a mobil eszközökön éppen egy kevesebb menü elemmel operáló menü mutat jól, érdemes egy kiterjeszhető gombot beépíteni azon gombelemeknek, melyek nyomán a menü nem férne el egyetlen sorban. A menüelemek többnyire az oldal tartalmának egy-egy megjelölt sorára hivatkoznak, fontos, hogy a görgetést mindig lágyra állítsuk biztosítva a megfelelő felhasználói élményt. A választási lehetőségeknél a "kevesebb több" elve érvényesüljön¹³!

Az oldal alján javasolt egy link elhelyezése, mely automatikusan visszagörget az oldal tetejére.

- **Reszponzivitás**

A weboldalnak dinamikusan kell alkalmazkodnia az aktuális képernyőmérethez, elsődlegesen annak szélességéhez, ehhez valamely keretrendszer (flexbox, grid, bootstrap) érdemes alkalmazni.

- **Átlátható elrendezés**

Mivel a kis méretű képernyőket tekintjük az elsődleges megjelenítőknek, kerülni kell az átláthatatlan mennyiségű szöveget és ismerve a felhasználói szokásokat, miszerint a gondos átolvasás helyett inkább csak átfutják adott oldal tartalmát és az érdekesebbnek látszó elemeket tekintik meg¹⁴, azon elemeket kell kiemelnünk, melyekkel a felhasználó biztosan interakcióba lép majd (pl. bejelentkezés gomb), illetve kerülnünk kell a túl részletes képeket, ésszerűbb az ikonszerű, letisztult, könnyen megjegyezhető vizuális elemek használata. Színek terén is szorítkozzunk egy maximum négy színből álló palettára!

- **Gyors betöltés**

A weboldalnak hibamentesnek, jól optimalizáltnak (ez főként a javascript kódokra vonatkozik) és autoplay-mentesnek kell lennie (utóbbi nem csak zavaró, de indokolatlanul megnyújtja a betöltési időt, különösen filmfájlok esetén).

¹² ALAN SMITH: How Scrolling Can Make (Or Break) Your User Experience , URL: <https://usabilitygeek.com/how-scrolling-can-make-or-break-your-user-experience/>, URL: 2020.12.28.

¹³ NICK BABICH: <https://xd.adobe.com/ideas/principles/web-design/12-dos-donts-web-design-2/> (2020.12.28.)

¹⁴ ALAN SMITH: 7 Web Design Rules You Should Never Break, URL: <https://usabilitygeek.com/web-design-rules-you-should-never-break/>, Megtekintve: 2020.12.28.

- **Összeköttetés a közösségi médiával**

Manapság, a web 2.0 utáni világban elkerülhetetlen, hogy az ügyfél folyamatos kapcsolatban maradjon a szolgáltatóval, értesüljön hírekről, akciókról, termékekéről, ebben pedig a különálló weboldalunknál lényegesen hatékonyabb a közösségi médián való jelenlétünk, melyet az ügyfél naponta többször meglátogat, érdemes felhívni az ügyfél figyelmét az ilyen jellegű elérhetőségeinkre.

- **Az érintőképernyő sajátosságai**

Vannak hardware specifikus elemek is, melyekre a tervezéskor figyelnünk kell, pl., hogy egyes css elemek nem megvalósíthatóak érintőképernyőn¹⁵ (pl. :hover selector), mivel azok megtervezésekor még az egérmutató volt az elsődlegesen pointer eszköz. Kialakításkor ügyeljünk arra, hogy a nagyobb ujjmérettel rendelkező felhasználók is könnyedén elérhessék a kívánt tartalmakat és kerüljünk minden olyan megoldást, melyek a képernyő méretének módosítását, az elem nagyítását követelik meg.

II/3. Az alkalmazással elkészíthető elemek kiválasztási szempontjai

A fentiek ismeretében kerültek meghatározásra az alkalmazás által elkészíthető weblapelemek, melyek szinte kivétel nélkül a Bootstrap 4 keretrendszer által jelennek meg. Mivel a jelenlegi dolgozat kereteit és a rendelkezésre álló időt bőven kimerítené egy mindenre kiterjedő, teljesen személyre szabható keretrendszer felépítése és az értelmező fordító megírása, ezért célravezetőbbnek tartottam egy jól elkülöníthető részekből felépülő weboldalséma kialakítását és az ezen belül felhasználható HTML elemek leszűkítését. Ezek, felülről lefelé haladva a következők:

1. Banner:

A banner az oldal tetején megtalálható részletes, figyelemfelhívó elem. Amennyiben egyetlen képet tartalmaz, az statikusan áll, de több kép megadása esetén azok között lágy áttűnéssel animációt tesz lehetővé. Opcionálisan szöveg is megadható, mely a banneren belül szabadon pozícionálható. Amennyiben a képernyő mérete XS, nem látható!

2. Navbar

A navigációs sáv a banner alatt helyezkedik el, elsődleges feladata, hogy automatikusan a kívánt könyvjelzőhöz görgesse a felhasználót. Az alábbi elemek valósíthatóak meg benne:

- **Brand:** Az index oldalra mutató link, szöveggel vagy képpel is megvalósítható.

¹⁵ MEZŐ ISTVÁN: Finally, a CSS only solution to :hover on touchscreens, URL:
<https://medium.com/@mezoistvan/finally-a-css-only-solution-to-hover-on-touchscreens-c498af39c31c>
Megtekintve: 2020.12.28.

- **Linkek:** Szöveges linkek, melyek egy megjelölt könyvjelzőre görgetik át az oldalt.
- **Toggler:** Speciális gomb, mely a már ki nem férő elemeket rejti el, elsődlegesen a közepesnél kisebb képernyőkön.

A Brand a navbar bal szélén, míg a Toggler mindenkorának jobb szélén helyezkedik el.

3. Konténer

A konténer tartalmazza a honlap tartalmi elemeit, sorokra bontva, melyeket a Bootstrap rácsrendszer tesz reszponzív megjelenésűvé. Elviekben itt a felhasználónak végig lehetőséget biztosíthatnánk a weboldal megvalósítására, éppen ezért itt volt szükség a legkörültekintőbb korlátozásra.

A row táblázat (Table) vagy hasáb (Bootrow) lehet, melyet elláthatunk egy id-vel, melyre a navbarban lévő link hivatkozik.

- **Táblázat**

A táblázat esetében tömbként megadhatóak a táblázat oszlopának nevei, valamint az oszlopok sorai és adott sorok tartalmai, utóbbiaknál a tömbök tömbjei helyett egy annál könnyebben vizualizálható és feldolgozható formában, mely esetében a sor száma, a kettőspont és a tömb kerülnek felvitelre.

- **Hasáb**

A hasábok deklarálásakor elsőként meg kell adnunk a hasáb oszlopainak számát és azok arányait, melyek összesen tizenkettőt kell, hogy kiadjanak a Bootstrap szabályai szerint¹⁶, majd az oszlopokon belül az alábbi elemek kerülhetnek elhelyezésre:

- Szöveg, minimális formázási lehetőséggel és igazítással
- Kép, mindenkorának jobb szélén helyezkedik el

4. Footer

Jellemzően a weboldalak alján helyezkedik el, tartalmaz egy gombot, mely visszanavigál az oldal tetejére, egy sort az elérhetőségeknek és egyet a social media ikonoknak, és egyet alulra, mely tartalmazza a készítő nevét és az évet, valamint egy utalást a Bootstrap keretrendszerre. Amennyiben az oldal mérete kisebb, mint közepes, a gomb a gépernyő aljához tapad, igazodva az érintőképernyő sajátosságaihoz.

5. Háttér

A háttér rendkívül fontos eleme a weboldalnak, mely lehet egyszínű, gradient, vagy tartalmazhat egy fixált állású képet, mely kitölți a keretet. Hogy a háttérkép látható

¹⁶ Grid system, URL: <https://getbootstrap.com/docs/4.0/layout/grid/>, Megtekintve: 2021.03.13.

maradjon, a navigációs menü és a konténer sorainak háttere és képei kaphatnak átlátszóságot.

6. Betűtípus:

A betűszínt a paletta határozza meg, a méretet szegmensenként lehet változtatni, a betűtípus az egész dokumentumban egységes.

7. Áttetszőség:

Amennyiben az oldal háttere kép, a láthatóság érdekében az opacity függvényel beállíthatóak a kívánt értékek elemtípusonként.

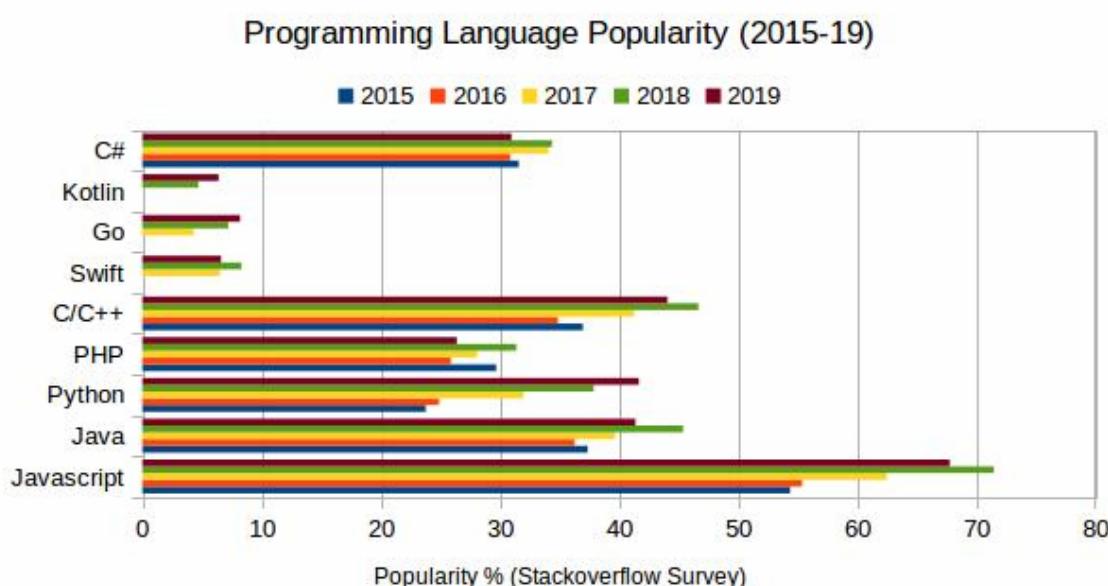
III. A téma kifejtése, fejlesztői dokumentáció

III/1. A programnyelv választásának indoklása

A Python megosztó programnyelvnek tekinthető¹⁷, különösen a professzionálisabb felhasználók körében, így kerülhetetlennek tekintem választásom indoklását, az előnyök és hátrányok ismertetésével.

A választással járó előnyök:

- a) Gyorsan, dinamikusan fejlődő, nagy támogatottságú programnyelv



4. ábra: Programnyelvek népszerűsége a Stackoverflow portál survey-je alapján

(<https://codinginfinite.com/top-programming-languages-2020-stats-surveys/>)

Jól látható, hogy a programnyelvek között a fenti ábrán előkelő és folyamatosan növekedést mutat a Python programozás nyelv népszerűsége, a TIOBE index szerint a

¹⁷ Most Loved, Dreaded, and Wanted Languages, URL: <https://insights.stackoverflow.com/survey/2020#technology-most-loved-dreaded-and-wanted-languages>, megtekintve: 2021.03.13.

C és Java után a harmadik helyet érte el¹⁸, míg a Google PYPL indexe szerint a Google Trends kimutatása szerint a legmagasabb arányban (30,17%) a Python nyelvre kerestek rá a vállalat keresőmotorjával¹⁹. Ebből következik, hogy a projekt elkészülte után a karbantartást és továbbfejlesztést a választott fejlesztőkörnyezetből kifolyólag nem érheti hátrány.

b) Közösségi leg fejlesztett ingyenes függvénykönyvtárak

A Python ingyenesen, kereskedelmi forgalomban is korlátozás nélkül használható programnyelv, melyben ennek megfelelően már a legtöbb felmerülő problémára létezik elérhető függvénykönyvtár, legyen szó bonyolult számításokról vagy fájltípusok olvasásáról és írásáról. A közösségiesség magában hordozza természetesen annak esélyét, hogy adott bővítmény működésében előfordulhatnak hibák, így ezek megválasztását és a hibák korrekcióját megfelelő körültekintéssel kell végeznünk.

c) Jó áttekinthetőség

A kód az OOP és tiszta kód elveknek megfelelő szerkesztése²⁰ esetén könnyen értelmezhető, átlátható, akár a nyelvet csak részlegesen ismerők számára is, melynek fordította a Python egyszerű szintaksisához szokott fejlesztőkről már nem mondható el. Az egyik kiemelkedő ok, amiért a Python népszerű a kezdők körében is, a jó olvashatósága, ezért a kód formázásakor ennek kiemelt szempontnak kell lennie²¹.

d) Produktív fejlesztési idő

A fentiekből együtt következik, hogy a nyelvben elfogadható sebességgel lehetséges a fejlesztés, melyhez a Jetbrains által nyújtott PyCharm integrált fejlesztőkörnyezet további hasznos funkciókat, pl. egyszerű bővítménykezelést és hibakeresési eljárásokat, valamint használható súgót nyújt.

A választással járó hátrányok:

a) Titkosítás hiánya

A Pythonnal szemben komoly kritika, hogy más magas szintű nyelvekkel ellentétben alig tartalmaz titkosítást²², így a fejlesztő a gyakorlatban nem tudja elrejteni az objektumait, csak a megfelelő elnevezésekkel és megjegyzésekkel jelezni a többi fejlesztő számára, hogy melyek felhasználása engedélyezett a számára (“We are all

¹⁸ TIOBE Index for March 2021, URL: <https://www.tiobe.com/tiobe-index/>, Megtekintve: 2021.03.13.

¹⁹ PYPL PopularitY of Programming Language, URL: <https://pypl.github.io/PYPL.html>, Megtekintve: 2021.03.13.

²⁰ AL SWEIGART: Beyond the Basic Stuff with Python, No Strach Press [USA], 2021, 192. o.

²¹ AL SWEIGART: Beyond the Basic Stuff with Python, No Strach Press [USA], 2021, 114. o.

²² 9. Classes: <https://docs.python.org/3/tutorial/classes.html>, megtekintve: 2020.10.10.

consenting adults”, szabad fordításban: "Cselekvőképes felnőtt egyének vagyunk”), a változó- és függvénynevek esetén erre bevett szokás a név eleji alulvonás használata.

b) Sebesség és memóriaigény

A nyelv egyértelműen mérhető és tapasztalható hátránya, hogy lassabb a hardware közelibb nyelveknél, ennek oka, hogy a kód futtatáskor először bytekódba fordítódik, melyet aztán a Python Virtual Machine-nak nevezett, C nyelv alapú program valós időben futtat²³, így a nyelv egyszerre hordozza magában a fordított és interpretált nyelvek jellemzőit²⁴. A futtató környezet az úgynevezett GIL (Global Interpreter Lock) biztosítja, hogy egyszerre csak egy utasítás hajtódjon végre, függetlenül az elérhető magok és szálak számától, ezért a felhasználónak magának kell minden tekintetben a többszálúságról gondoskodnia²⁵.

Hasonló hátránya a nyelv használatának, hogy gyengén típusos és rugalmas adattípusokkal dolgozik, ezáltal a változók nagy memóriaegységeket foglalnak le függetlenül attól, hogy a felhasználó mit szeretne tárolni bennük.

c) Esetleges kompatibilitás elvesztése

A Pythonnal kapcsolatban felmerülő kritika, hogy a közösségi fejleszthetőség nyomán a különféle verziók és a függvénykönyvtárak egymással inkompatibilissé válhatnak, ami veszélyes egymástól függő csomagok esetében, ez pedig megnehezíti a program továbbfejlesztését²⁶.

III/2. Fejlesztőkörnyezet

Alaplap: MSI Tomahawk B350

CPU: AMD Ryzen 2600

Memória mérete: 32 GB

Tárolók:

- Kingston SA2000M NVMe 500 GB (Windows)
- Seagate Momentum Thin 2,5" SATA 500GB (Linux)

Operációs rendszerek:

- Windows 7 Ultimate SP1 64bit

²³ Inside the Python Virtual Machine, URL: <https://leanpub.com/insidethepythonvirtualmachine/read>, Megtekintve: 2021.03.13.

²⁴ CAIO COZZA: A quick overview of the Python Virtual Machine — Pt. 1, URL: <https://medium.com/@caicozza.art/a-quick-overview-of-the-python-virtual-machine-pt-1-315e74c036f4>, megtekintve: 2020.10.10

²⁵ Chapter 1. Understanding Performant Python, URL: <https://www.oreilly.com/library/view/high-performance-python/9781492055013/ch01.html>, megtekintve: 2021.03.12.

²⁶ Victor Stinner: Python Compatibility Version, URL: <https://www.python.org/dev/peps/pep-0606/#id25>, Megtekintve: 2021.03.13

- Linux Mint 20 64bit

Fejlesztő alkalmazások:

- JetBrains Pycharm Professional 2018.2 (Windows) + Python 3.7
- JetBrains Pycharm Community 2020.2 (Linux) + Python 3.7
- Code::Blocks 20.03 (Windows) 64bit
- Visual Studio Code 1.53.2 (Windows / Linux)
- FontForge 2020-03-14
- Krita 4.2.8 64bit
- Dia 0.97

A Python programnyelv az adattípusok nem a szokványos nyelven hívja, úgymint str (String), bool (Boolean), dict (Map)²⁷, ezért azokat a Pythonban megszokott nevén nevezem.

III/3. Felhasznált függvénykönyvtárak

A nyelv egyik legerősebb jellemzője a széles körben elérhető függvénykönyvtárak, melyek jelentősen megkönnyítik a gyors fejlesztést. A Boo-T esetében az alábbiak kerültek felhasználásra:

- **tkinter**²⁸

A Python alapértelmezett és beépített grafikus alkalmazás fejlesztője, mely a Tcl/Tk programnyelvre és grafikus felületkészítőre épül, arra egy újabb objektum réteget húzva. A tkinter a PIL képkezelő modul segítségével képes megjeleníteni az importált képeket és ikonokat.

Az alábbi modulokban került felhasználásra:

- **Boo-T.py:** Teljes ablakvezérlés (Tk szint)
- **About.py:** Teljes ablakvezérlés (TopLevel szint)
- **Config.py:** messagebox objektum megjelenítése.
- **DisplayLoading.py:** Betöltőkép megjelenítése TopLevel és Canvas felhasználásával.
- **OptionsM.py:** Teljes ablakvezérlés (TopLevel szint)
- **SaveHTML.py:** messagebox objektum megjelenítése.

²⁷ Python Data Types, URL: https://www.w3schools.com/python/python_datatypes.asp, Megtekintve: 2021.03.13

²⁸ Graphical User Interfaces with Tk, URL: <https://docs.python.org/3/library/tk.html>, Megtekintve: 2021.03.13.

- **UndoBuffer.py**: referenciaként megkapott button objektum kezelése.
- **sys²⁹**

Az operációs rendszerrel kapcsolatos információk lekérdezésére és ~műveletek végrehajtására szolgál, melyeket ezáltal a felhasználónak nem kell magának implementálnia.

Csak a Boo-T.py importálja, használatával hozzáadja az "src/" mappát a workspace-hez, így a benne található py modulok importálhatóvá válnak. Továbbá Linux esetén, amennyiben a felhasználó telepítette a felkínált alkalmazással a "Hammerfat.ttf" betűtípuszt, újraindítja az alkalmazást, hogy az immár betölthető legyen.

- **os³⁰**

Operációs rendszerrel- és fájlműveletekkel kapcsolatos függvényeket tartalmaz, ezen kívül korlátozottan képes alprogramok elindítására is.

Az alábbi modulokban került felhasználásra:

- **Boo-T.py**: Ellenőrzi fájlok meglétét, bekéri a workspace teljes elérési útját, megadja az operációs rendszerre jellemző mappaelválasztót, mappát hoz létre, lekéri a ctypes számára a teljes absztrakt útvonalat az importálandó szkriptekhez, valamint fájlokat töröl.
- **About.py**: A 'PYGAME_HIDE_SUPPORT_PROMPT' rendszerváltozó "hide"-ra állításával blokkolja a pygame modul önreklámozó megjelenítését a konzol kimeneten.
- **Config.py**: Ellenőrzi fájlok elérési útját, bekéri a felhasználói nevet, valamint az os.walk függvény segítségével végigjár adott útvonal almappáin és az azokban található file-okon.
- **Dictionaries.py**: Végigmegy a "dicts/" mappa szótár állományain a walk függénnyel.
- **OptionsM.py**: Végigmegy a "dicts/" mappa szótár állományain a walk függénnyel.
- **PythonCompiler.py**: Bekéri az operációsrendszer-függő sorelválasztót.
- **SaveHTML.py**: Bekéri az operációsrendszer-függő mappaelválasztót, a workspace elérési útját, ellenőrzi fájlok meglétét, mappát hoz létre., fájl másolásához teljes elérési útvonalat képez, valamint végigmegy adott könyvtárfán.

²⁹ sys — System-specific parameters and functions, URL: <https://docs.python.org/3/library/sys.html>, Megtekintve: 2021.03.13.

³⁰ os — Miscellaneous operating system interfaces, URL: <https://docs.python.org/3/library/os.html>, Megtekintve: 2020.03.13.

- **re**³¹

Reguláris kifejezések keresésére szolgáló függvénykönyvtár, mely lehetővé teszi adott szövegrészek cseréjét is.

Az alábbi modulokban került felhasználásra:

- **Boo-T.py**
- **Config.py**
- **PythonCompiler.py**
- **SaveHTML.py**
- **pyglet**³²

A tkinterhez hasonló grafikus felület fejlesztő, melynek a font importáló függvényét használja a Boo-T, ezáltal nincs szükség a betűtípus telepítésére. Linux alatt a gnome-font-viewer, display és font-manager programokkal kísérli meg a font telepítését. Kizárálag a Boo-T.py használja.

- **Pillow (PIL)**³³

Képek importálását és szerkesztését teszi lehetővé, együttműködve a tkinter modullal. Az alábbi állományok használják:

- **Boo-T.py**
- **About.py**
- **DisplayLoading.py**
- **Boo-T.py**
- **OptionsM.py**
- **ctypes**³⁴

Különféle függvénykönyvtárakat tartalmaz, valamint lehetővé teszi operációs rendszertől függetlenül C nyelven íródott külső függvénykönyvtárak importálását is.

Az alábbi modulokban került felhasználásra:

- **Boo-T.py:** Betölti a Fortranban írt compilert tartalmazó függvénykönyvtárat és elindítja a compile subroutine-t.
- **Monitor.py:** A windll.user32.screen objektumából lekéri a képernyő paramétereit tuple formájában.

³¹ re — Regular expression operations, URL: <https://docs.python.org/3/library/re.html>, Megtekintve: 2020.03.13.

³² Pyglet Homepage, URL: <http://pyglet.org/>, Megtekintve: 2020.03.13.

³³ Pillow — Pillow (PIL Fork), URL: Pillow — Pillow (PIL Fork), Megtekintve: 2021.03.13.

³⁴ ctypes — A foreign function library for Python, URL: <https://docs.python.org/3/library/ctypes.html>, Megtekintve: 2021.03.13.

- **Xlib**³⁵

Linux alatt az elsődleges képernyő méretének lekérésére szolgál, kizárálag a Monitor.py használja.

- **clipboard**³⁶

Lehetővé teszi a vágólap használatát és az onnan történő másolást. A Boo-T.py és a GetCodeOnly.py használja a scrolledTextBox objektum tartalmának módosítására.

- **Winapps**³⁷

Windows alatt a feltelepített applikációk közül a böngészők elérési útját ellenőrzi a Config.py állományban.

- **Platform**³⁸

Operációs rendszerrel kapcsolatos információk kérhetőek le vele, a Config.py állományban az operációs neve kerül általa lekérésre, majd eltárolásra.

- **subprocess**³⁹

Alprogramok részletesen konfigurálható elindítását teszi lehetővé.

Az alábbi modulokban került felhasználásra:

- **Boo-T.py:** A wheris alkalmazás használatával Linux alatt megállapítja a böngészők elérési útvonalát. Mindkét operációs rendszer alatt elindítja a böngészőket tesztelés céljából a szerkesztett weboldal fordított kódjával. Ezen felül, elindítja a megadott lehetséges betűtípus telepítésére alkalmas szoftvereket
- **Config-T.py:** A whereis alkalmazással a böngészők elérési útját kutatja fel.

- **random**⁴⁰

Pszeudo-Véletlenszám generálásra alkalmas függvénykönyvtár.

Az alábbi modulokban került felhasználásra:

- **About.py:** A labda környezettel való interakcióját és a gép gondolkodását befolyásolja.
- **PythonCompiler.py:** Amennyiben a felhasználó 'random' színpalettát adott meg, 0-27 között generál egy random számot.

³⁵ The Python X Library, URL: <https://github.com/python-xlib/python-xlib>, Megtekintve: 2021.03.13.

³⁶ clipboard 0.0.4, URL: <https://pypi.org/project/clipboard/>, Megtekintve: 2020.03.13

³⁷ winapps - Python library for managing installed applications on Windows, URL: <https://pypi.org/project/winapps/>, Megtekintve: 2021.03.13.

³⁸ platform — Access to underlying platform's identifying data, URL: <https://docs.python.org/3/library/platform.html>, Megtekintve: 2021.03.13.

³⁹ subprocess — Subprocess management, URL: <https://docs.python.org/3/library/subprocess.html>, Megtekintve: 2021.03.13.

⁴⁰ random — Generate pseudo-random numbers, URL: <https://docs.python.org/3/library/random.html>, Megtekintve: 2021.03.13.

- **datetime**⁴¹

Dátumidő műveletekre alkalmas függvényeket tartalmaz, az About.py-ban és a PythonCompiler.py-ban a tizedmásodpercek alkotják az elsődleges seed forrást.

- **pygame**⁴²

Komplex játékfejlesztő függvénykönyvtár, melynek részelemei kerültek használásra az alábbi modulokban:

- **About.py:** Amennyiben a datetime-mal való seed kivételt dob, az egér pozícióját használja a random seed-elésére. Mivel a Python playsound modulja nem kompatibilis a Linux operációs rendszerrel, a hangok megszólaltatását a pygame mixer modulja végzi.
- **PythonCompiler.py:** Amennyiben a datetime-mal való seed kivételt dob, az egér pozícióját használja a random seed-elésére.

- **threading**⁴³

A Boo-T.py-ban külön szálakra választja el a highlighter modult a többi programelementől, így biztosítva számára a folyamatos erőforrást, valamint az alkalmazás általános lelassulásának és a felhasználói élmény romlásának elkerülését.

- **psutil**⁴⁴

Amennyiben a felhasználó Linux alatt a betűtípus telepítése mellett dönt, a Boo-T alkalmazás további futása addig blokkolódik, amíg a psutil.process_iter()-ben megtalálható az elindított, betűtípus telepítésére alkalmas alkalmazás. Kizárolag a Boo-T.py használja.

- **time**⁴⁵

A szintaksis kiemelő szál futását szabályozza az által, hogy a folyamatos while ciklust késlelteti, valamint folyamatos gépelés esetén szabályzásával meggátolja a szintaxis vizsgáló kód futását. Csak a Boo-T.py használja.

- **webbrowser**⁴⁶

⁴¹ datetime — Basic date and time types, URL: <https://docs.python.org/3/library/datetime.html>, Megtekintve: 2021.03.13.

⁴² PyGame, URL: <https://www.pygame.org/>, Megtekintve: 2021.03.13.

⁴³ threading — Thread-based parallelism, URL: <https://docs.python.org/3/library/threading.html>, Megtekintés: 2020.03.13.

⁴⁴ psutil, URL: <https://pypi.org/project/psutil/>, Megtekintés: 2021.03.13.

⁴⁵ time — Time access and conversions, URL: <https://docs.python.org/3/library/time.html>, Megtekintés: 2021.03.13.

⁴⁶ webbrowser — Convenient Web-browser controller, URL: <https://docs.python.org/3/library/webbrowser.html>, Megtekintés: 2021.03.13.

Megnyitja adott weblapot az alapértelmezetten beállított böngésző segítségével. A Boo-T.py esetében így valósul meg a Help menü megjelenítése, mely maga is egy, a Boo-T segítségével készült weboldal.

- **shutil**⁴⁷

Fájlműveleteket tartalmazó függvénykönyvtár. Az alábbi állományokban került felhasználásra:

- **Boo-T.py:** Belső tartalommal rendelkező teljes könyvtárak törlését valósítja meg.
- **SaveHTML.py:** Belső tartalommal rendelkező teljes könyvtárak törlését valósítja meg, továbbá képfájlokat másol.

III/4. Futtatáshoz szükséges külső alkalmazások

A Boo-T független alkalmazás, ennek ellenére szüksége van egyes funkcióhoz külső alkalmazásokra.

- **Betűtípus telepítő**

Amennyiben Linux operációs rendszeren futtatjuk, a pyglet.add_file() függvénye nem képes a betűtípus egyszerű betöltésével biztosítani annak integrálódását az operációs rendszer által elérhető típusok közé, ezért a megfelelő megjelenítéshez javasolt a HammerFat.ttf betűtípus telepítése. A Boo-T a "font-manager"⁴⁸, "gnome-font-viewer"⁴⁹ és "display" alkalmazásokkal való megnyitásra tesz kísérletet. A telepítést és bezárást követően a Boo-T újraindul, automatikusan felismerve a betűtípust.

- **Böngészők**

A weblapok teszteléséhez a Boo-T négy, az elkészültek kor népszerű és támogatott böngészőt támogat, ezek közül legalább egy telepítésére szükség van a megfelelő teszteléshez: **Chrome, Firefox, Opera** és **Edge**

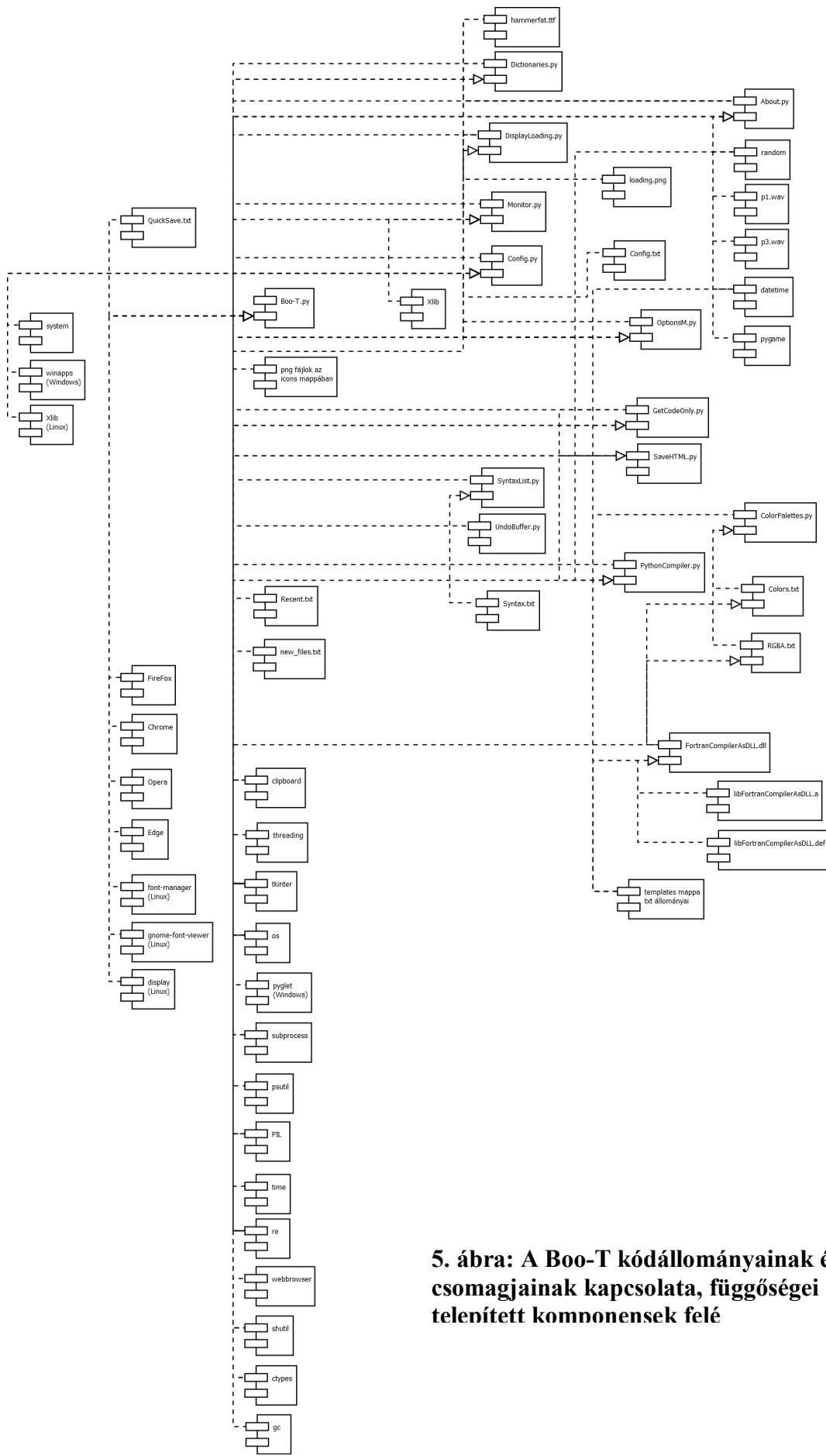
III/5. Könyvtárszerkezet

Az alkalmazás számos forrásfájllal, köztük saját külső állományaival, sémákkal, képekkel és dolgozik, így érdemes ábrázolni a könnyebb megértés érdekében a workspace-t, melyben az alkalmazás fut, felvázolva, melyek azok az elemek, amelyek létfontosságúak a hibátlan futtatás szempontjából.

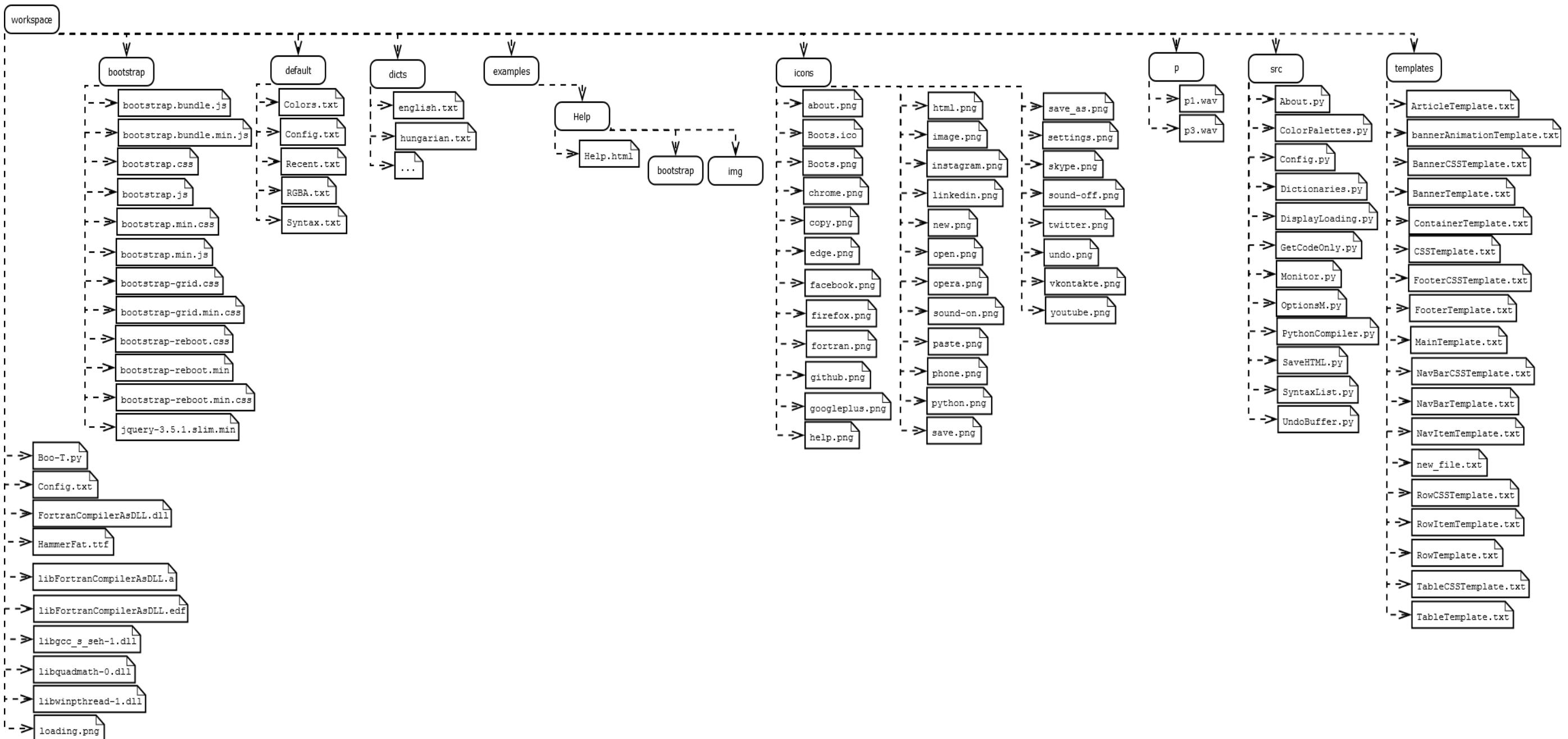
⁴⁷ shutil — High-level file operations, URL: <https://docs.python.org/3/library/shutil.html>, Megtekintés: 2021.03.13.

⁴⁸ Font Manager, URL: <https://github.com/FontManager/font-manager>, Megtekintés: 2021.03.13.

⁴⁹ Gnome-Font-Viewer, URL: <https://launchpad.net/ubuntu/+source/gnome-font-viewer>, Megtekintés: 2021.03.13.



5. ábra: A Boo-T kódállományainak és csomagjainak kapcsolata, függőségei a telenített komponensek felé



6. ábra: A Boo-T alkalmazás workspace-ének könyvtárfája

III/6. A program felépítése

A program felépítése erősen központosított, egyetlen nagy osztálynak, a MainWindow a példánya hozza létre és menedzseli a további, működéshez szükséges osztályok működését. Az osztályokból egy időben csak egy példány létezik, amennyiben több osztály számára is szükség van egy adott osztály elemeinek elérésére, úgy azokat referenciaiként megkapják a MainWindow osztálytól, ezáltal az esetleges összeférhetetlenségek és a memória indokolatlan terhelése is elkerülhető⁵⁰.

A program ebből kifolyólag az objektumokon átívelő állományokon túl erősen épít a függvényekre, így azok kidolgozásánál különösen fontos, hogy bizonyos alapelveknek eleget tegyünk⁵¹:

- A tiszta függvények használata érdekében nem használunk globális változókat.
- A függvényeket átadhatjuk objektumokként más objektumoknak.
- Amennyiben valamely művelet ismétlődik a program futása során, külön függvény meghívásával tesszük lehetővé a folyamatot.
- Amennyiben egy string feldolgozása nehezen olvasható, a könnyebb olvashatóság érdekében ezeket a feldolgozó metódusokat minden esetben külön függvényekben helyezzük el.

A program futása 3 szakaszra tagolható, melyek a fejezet további részeiben kerülnek részletesebb kifejtésre.

1. Deklarációs szakasz, az osztályok létrejötte, adatok betöltése

2. Felhasználói interakció szakasza

3. Bezárás, esetleges maradványok eltakarítása

A működés során a privát elérésű, osztályszintű változók dominálnak, a felhasználó kizárolag az ablakos felületen keresztül léphet interakcióba a programmal.

III/6/1. Deklarációs szakasz, az osztályok létrejötte, adatok betöltése

A szakasz a MainWindow osztály példányosításától (Pythonban nem szükséges a létrejövő osztályt változóhoz rendelni) a rá vonatkozó mainloop eljárás meghívásáig tart, mely a fő ablakot vezérlő szálat az ablak bezárásig folyamatosan életben tartja. Erre a szakaszra a szekvenciális lefutás jellemző.

⁵⁰ ANDREI BOYANOV: Python Design Patterns: For Sleek And Fashionable Code, Megtekintés: 2021.03.13.

⁵¹ STEVEN LOTT: Functional Python Programming, Pakt Publishing [Birmingham - USA], 2015, 37. o.

A MainWindow osztályban első lépésként a threading modul Thread objektuma segítségével 1-1 függvény kötődik 1-1 szálhoz, így biztosítják a program több szalon történő futását. Pythonban a szálak, eltérően a Javától, képes trükközés nélkül objektumuktól független függvények futtatására az osztályon belül.

```
def __init__(self):
    from threading import Thread

    self.__Window = Thread(target=self.__createAll)
    self.__HighLighter = Thread(target=self.__highLighter)
    self.__HighLighter.daemon = True
    self.__Window.start()
    self.__HighLighter.start()
```

7. ábra: szálazás

Az első szál magát az alkalmazás ablakát és objektumait hozza létre, míg a másodlagos szál kizárolagosan a szintaxis kiemelő működtetéséért felelős. A self.__highLighter függvényt kezelő szál daemon-ként fut, aminek működése eltér a megszokottaktól. Míg a legtöbb esetben a daemon egy olyan folyamat, mely a szülője megszűnése után is folyamatosan a háttérben fut, árvaként megkapva szülőnek az elérhető legősibb osztályt⁵², addig Pythonban ezek az osztályok csak addig futnak, amíg legalább egy nem-daemon osztály aktív⁵³.

III/6/1/1. A createAll szál:

A szál létrehozza a self.__keyPress változót, melyet azonnal False-ra állít, ugyanis ennek a változónak az állapotától függ a szintaxiskiemelő működése, False esetén nem történik meg a kiemelő függvény meghívása. A self.__main változóban eltárolt Tk objektum létrehozása következik, mely viszont mindenkorán láthatatlan marad, amíg minden más létrehozó folyamat be nem fejeződik és az első szakasz végére nem érünk.

Következőként létrejönnek az fájlállapotot jellemző változók, melyek friss indításkor egy mentetlen, módosítatlan, új, üres dokumentumot feltételeznek. A kódbeli delimitatort alapértelmezésként "%%" -re állítja a program, illetve, felülírja a fő ablak objektum bezárásához tartozó függvényt a self.__closeWindow() -ra, hogy a nem mentett tartalmak mentésére rákérdezzen, csökkentve a felhasználói hibázások lehetőségét.

⁵² About Daemons and Services, URL:
<https://developer.apple.com/library/archive/documentation/MacOSX/Conceptual/BPSystemStartup/Chapters/Introduction.html>, megtekintve: 2020.03.03

⁵³ threading — Thread-based parallelism, URL: <https://docs.python.org/3/library/threading.html>, megtekintve: 2020.03.03.

```

self.__keyPress = False
self.__main = Tk()
self.__main.withdraw()
self.__main.overrideredirect(True)
self.__main.resizable(False, False)

self.__checkAllLines = True
self.__opened = False
self.__modified = False
self.__path = ""
self.__delimiter = "%%"
self.__tempDelimiter = self.__delimiter

self.__main.protocol('WM_DELETE_WINDOW', self.__closeWindow)

```

8. ábra: Nyitó deklaráció, az ablak elrejtésével

Ezt követően négy, a program működése szempontjából létfontosságú objektum példányainak létrehozása következik.

- **self.__dicts (Dictionaries)**

A program működése során biztosított a többnyelvűség, a hibaüzenetek, menük az előre telepített nyelvek szerint jelennek meg. A nyelveket tartalmazó egyszerű szövegfájlok a "dicts/" könyvtárban találhatóak, melyekben minden sor a kulcs=érték formátumban tartalmazza a megjelenítendő szövegeket, a behelyettesítésre váró elemeket a "#szöveg#" forma jelöli. Az osztály konstruktőrénél létrehoz egy self.__D nevű szótárat, melyben a fájlnevek ".txt" nélküli részeiből jönnek létre a nyelvet jelölő kulcsok, melyeknek értékét egy újabb szótár adja, melyben a kifejezések szerepelnek kulcsként és a hozzájuk tartozó kifejezések értékként:

```
self.__D[kulcs][érték]
```

Az osztály egyetlen publikus függvénye a getWordFromDict, mely paraméterként két értéket, a nyelvet és a kifejezést várja, str formátumban. Mivel a nyelv a konfigurációs állományban kerül eltárolásra, a kifejezéseket az alábbi formában kéri le a program:

```
self.__dicts.getWordFromDict(self.__Config.get_Element("Language"), "word"))
```

- **self.__Config (Config)**

A konfigurációs osztály feladata, hogy eltárolja a program viselkedését tartalmazó értékpárokat és a felhasználónak lehetőséget biztosítson az alkalmazás testreszabására. A konfigurációs fájl, hasonlóan a szótár fájlokhöz, kulcs=érték párokat tartalmaz. A konfigurációs objektum létrejöttekor három feladatra készül fel:

1.) Lekéri a system modul segítségével az operációs rendszer nevét, mely jelentősen befolyásolja a program további működését, azt a self.__os_Name állományban tárolja, mely kívülről a get_OS_Name függvénytelérhető el.

```

def __Load_Config_File(self):
    """Will load the config file located in the root folder."""
    if os.path.exists("Config.txt"):
        try:
            return(self.__loadDict("Config.txt"))
        except:
            return (self.__loadDict("default/Config.txt"))
    else:
        return(self.__loadDict("default/Config.txt"))

```

9. ábra: Config fájl betöltése

- 2.) Betölti a workspace-ben található "Config.txt" állomány tartalmát, az elérés vagy beolvasás meghiúsulása esetén a "default/Config.txt" állományt. Az objektum self.__Config állományának tartalma kívülről a get_Element függvényel érhető el, mely egy str típusú változót vár paraméterként, az elemek módosítására a set_Element nyújt lehetőséget, mely paraméterként a key és value str változókat várja. Ezen felül, publikus még a saveConfig függvény, mely a Config.txt fájlba menti a self.__Config tartalmát a már ismert formátumban.

```

def saveConfig(self):
    file=open("Config.txt", "w")
    for key in self.__Config:
        file.write(key+"="+self.__Config[key]+\n")
    file.close()

```

9. ábra: Config fájl mentése

- 3.) A konfigurációs objektum létrejöttekor ellát még egy speciális feladatot, amennyiben az "AutoCheckForInstalledBrowsers" kulcs értéke True és valamely böngésző útvonalán üres stringet talál, kísérletet tesz az automatikus felkutatásra, melynek eredménye kihat a program működésére, de nem kerül mentésre egészen a felhasználó általi mentés utasítás kiadásáig. A felkutatás böngészőnként és operációs rendszer típusa szerint történik.

Windows operációs rendszer esetében a winapps modul search_installed() függvénye⁵⁴ által lekérésre kerülnek az alkalmazás adatai, melyből regex segítségével kerül kinyerésre az elérési út, Linux rendszer alatt a rendszert képező which segítségével történik az elérési út megállapítása. Amennyiben a keresés nem vezetett eredményre, Windows alatt az alapértelmezett telepítési helyeken is megtörténnek a felkutatások, melyek a Program Files mappákban (32 és 64 bites alkalmazások egyaránt), valamint Opera esetében a felhasználói fiókon belül az appdata mappában ennek végbe. Amennyiben a keresések nem vezetnek eredményre, az alkalmazás rákérdez a tkinter beépített messagebox objektumával, hogy kívánja-e maga az indítófájlt megtalálni, pozitív

⁵⁴ winapps - Python library for managing installed applications on Windows, URL: <https://pypi.org/project/winapps/>, Megtekintve: 2021.03.13.

megerősítés esetén a tkinter beépített filedialog objektumával megkereshetjük a kívánt alkalmazást. A felhasználónak mentenie kell az aktuális konfigurációs állapotot, hogy újraindításkor már a kívánt beállításokkal induljon az alkalmazás.

```

def __GetLocation(self, browser):
    """This is the Windows only applicaton """
    import winapps
    for app in winapps.search_installed(browser):
        if app!="":
            result=self.__Get_App_Path(self.__Regex_Get_Install_Location(app), browser)
            if result!="":
                return(result)
    return("")

def __Regex_Get_Install_Location(self, app):
    import re
    """Searches for the application's path in winapps result list"""
    FindRegex = re.findall(r"install_location=WindowsPath\(\['[a-zA-Z0-9:\/\s\(\)]+\'\)", str(app))
    if len(FindRegex) > 0:
        return(FindRegex[0].replace("install_location=WindowsPath('", "").replace("'", ""))
    return("")

def __LinuxFindCode(self, codepart):
    import subprocess
    try:
        return(str(subprocess.check_output(["which", codepart])[2:-3]))
    except:
        return("")
```

10. ábra: Kísérlet az alkalmazások helyének megállapítására

```

def __CheckFireFox(self, FireFox):
    """Checks browser location by directly searching in the OS and at the default locations as well."""

    if FireFox=="":
        if self.__os_Name=="Windows":
            self.__Config["FireFox"]=self.__GetLocation("FireFox")
            if self.__Config["FireFox"]=="":
                self.__Config["FireFox"] = self.__pathExists("C:\Program Files\Mozilla Firefox\firefox.exe")
            if self.__Config["FireFox"]=="":
                self.__Config["FireFox"] = self.__pathExists("C:\Program Files (x86)\Mozilla Firefox\firefox.exe")
        else:
            self.__Config["FireFox"] = self.__LinuxFindLocation("Firefox")

        if self.__Config["FireFox"]=="":
            self.__Config["FireFox"] = self.__Browser_Search_Window("FireFox")
```

11. ábra: Példa egy böngésző futtatható állományának helyének megállapítására

- **self.__Syntax (SyntaxList)**

A szintaxislista egy egyszerű felépítésű objektum, melyet a PythonCompiler objektum használ fel a szintaktikai értelmezéshez, valamint a fő ablakban is megjelennek egy ListBox formájában, típusuk szerint színnel jelölve, a fejlesztői munkát segítendő. Az objektum konstruktora létrehozza a self.__syntax belső szótárat, melynek kulcsait és értékeit a már fent említett kulcs=érték formában tartalmazza a "default/Syntax.txt". Az objektumból a getKeys() függvény a self.__syntax kulcsait adja vissza listaként, míg a getValueofKey() egy kulcs értékét adja vissza str-ként.

- **self.__monitor (Monitor)**

A monitor szintén egy rendkívül egyszerű objektum, egyetlen feladata az aktuális monitor felbontás lekérése. Windows alatt ez a ctypes.windll.user32⁵⁵ modul segítségével történik, mely a GetSystemMetrics(0) esetén a szélességet, GetSystemMetrics(1) adja vissza, ezeket a self.__screensize változóban tárolja el tuple-ként. Linux alatt az Xlib objektumának Display()⁵⁶ függvénye a ":0" paraméterrel adja vissza az elsődleges képernyő felbontását, melynek width_in_pixels és height_in_pixels változóját tároljuk el a fent említett tuple-ben. A get_screensize() függvénnyel kérhető le a self.__screensize tartalma.

Fontos megemlíteni, hogy a monitor objektum nem reagál a képernyőményre változásaira, ellenben manuálisan állítható az ablakméret az OptionsMenu objektum felültén.

```
if system=="Windows":  
    from ctypes import windll as windll  
  
    user32 = windll.user32  
    self.__screensize = user32.GetSystemMetrics(0), user32.GetSystemMetrics(1)  
else:  
    from Xlib.display import Display  
    screen=Display(":0").screen()  
    self.__screensize=screen.width_in_pixels, screen.height_in_pixels
```

12. ábra: Monitor objektum tartalmi része

A betöltés folytatásaként, amennyiben a konfigurációs állományban a "noLoading" értéke hamis, példányosításra kerül a DisplayLoading objektum, mely a rendelkezésre álló képernyőményet adatok alapján létrehoz egy TopLevel objektumot, melyen egy, a betöltőképet tartalmazó Label foglal helyet. A kép 3,5 másodpercre jelenik meg, majd megsemmisíti önmagát és folytatódik a betöltés a betűtípus importálásával.

⁵⁵ Python ctypes.WinDLL() Examples, URL:

<https://www.programcreek.com/python/example/59464/ctypes.WinDLL>, Megtekintés: 2021.03.14.

⁵⁶ Opening the Display, URL: <https://tronche.com/gui/x/xlib/display/opening.html>, Megtekintés: 2021.03.14.

```

from PIL import ImageTk, Image

if size[0]>1268 and size[1]>768:
    self.__w_Size=800
else:
    self.__w_Size=640

self.__h_Size=round((self.__w_Size/800)*300)
self.__Loading_Window=Toplevel()
self.__Loading_Window.geometry("%dx%d+%d+%d" % (self.__w_Size, self.__h_Size,
                                                (size[0]/2)-self.__w_Size/2,
                                                (size[1]/2)-self.__h_Size/2-50))
self.__Loading_Window.overrideredirect(True)
self.__Loading_Window.resizable(False, False)
self.__Img = ImageTk.PhotoImage(Image.open("loading.png").resize((self.__w_Size,
self.__h_Size)))

self.__ImgLabel = Label(self.__Loading_Window, image=self.__Img)
self.__ImgLabel.pack()

self.__Loading_Window.after(3500, self.destroy_loader)
self.__Loading_Window.wait_window()

```

12. ábra: Betöltőképernyő kódja

Windows operációs rendszer alatt a workspace-ben található "HammerFat.ttf" fájlt a pyglet add_file⁵⁷ metódusa hozzáadja a font.families() listához, így azonnal használható. Linux alatt, amennyiben a betűtípus nincsen telepítve, az alkalmazás egy messagebox.askyesno objektummal az alkalmazás javaslatot tesz a telepítésre és bekéri a felhasználó döntését. Igenlő válasz esetében a "font-manager", "gnome-font-viewer", majd "display" programokkal, amennyiben elérhetőek, ebben a sorrendben kísérletet tesz a betűtípus megnyitására, majd blokkolja az alkalmazás további futtatását a megnyitott alkalmazás bezárásáig, melyet követően az alkalmazás újraindul, sikeresen telepítés esetén betöltve az említett betűtípust.

⁵⁷ Loading Custom Fonts, URL:
https://pythonhosted.org/pyglet/programming_guide/loading_custom_fonts.html, Megtekintés: 2021.03.14.

```

if "HammerFat_Hun" not in font.families():
    ham=messagebox.askyesno("HammerFat_Hun", self._dicts.getWordFromDict(self._Config.get_Element("Language"), "linuxFontError"))
    if ham==True:
        Done=self._linuxTryToOpenFontViewer("font-manager")
        if Done==False:
            Done=self._linuxTryToOpenFontViewer("gnome-font-viewer")
        if Done==False:
            Done=self._linuxTryToOpenFontViewer("display")
        if Done==False:
            messagebox.showerror(self._dicts.getWordFromDict(self._Config.get_Element("Language"), "noFontApp"),
self._dicts.getWordFromDict(self._Config.get_Element("Language"), "linuxNoGnome").replace("#path#", '"' + os.getcwd() + os.sep + 'HammerFat.ttf"'))
            if Done==True:
                try:
                    os.execv(sys.executable, ['python'] + sys.argv)
                except:
                    os.execl(sys.executable, os.path.abspath(__file__), *sys.argv)

```

13. ábra: Betöltőképernyő kódja

Ezt követően jönnek létre a MainWindow tartalmi űrlap elemei.

- 1.) Az ablak felveszi a monitor felbontása szerint optimális méretet és pozíciót az asztalon, beállítja a címet, a keret ikonját és elemeit.

```

def __GetWindowSize(self, size):
    if size[0] > 1600 and size[1]>1200:
        s = 4
    elif size[0] > 1280 and size[1]>768:
        s = 3
    elif size[0] > 800 and size[1]>600:
        s = 2
    else:
        s = 1
    return (s)

```

14. ábra: Optimális méret megállapítása

- 2.) A menüsor ikonjai a self.__createButton metódus segítségével jönnek létre, mely egy képet, egy függvényt, a megjelenítendő szöveget létrehozó függvényt, valamint egy pozíciószorzót vár paraméterül.

```

def __createButton(self, image, command, on_Enter, buttonpoz):
    """Generates Button from given data. Image, command and enter-
message generation is unique."""

    button = Button(self.__main, image=image, width=32, height=32, command=command)
    button.place(x=self.__getButtonPoz(buttonpoz), y=1)
    button.bind("<Enter>", on_Enter)
    button.bind("<Leave>", self.__on_leave)
    return(button)

```

15. ábra: A gombokat legeneráló függvény

3.) Létrejön az Undobuffer objektum, mely folyamatosan menti a CodexBox tartalmát a saveBox eljárásban keresztül, melyet közvetve a CodeBox action-jeihez bindingelt __addBuffer eljárás hív meg és paraméterez fel. Az UndoBuffer maximum 25 elemet tárol el, amennyiben ezt meghaladná, a legelsőt törli a self.__Buffer listából. Amennyiben az Undo gomb megnyomásra kerül, az self.undo függvény a legutolsó elemet törli a listából és visszaadja eredményként. Ha a self.__Buffer üres, a gomb inaktívvá válik.

```

def saveBox(self, code):
    if len(self.__buffer)>=self.__maxSize:
        self.__buffer.pop(0)
    self.__buffer.append(code)
    self.__button.config(state=NORMAL)

def undo(self):
    try:
        temp=self.__buffer[-1]
        self.__buffer.pop()
        if len(self.__buffer)==0:
            self.__button.config(state=DISABLED)
        return(temp)
    except:
        return(False)

```

16. ábra: UndoBuffer működése

4.) A böngészők elérhetőségének ellenőrzése eldönti a self.__CheckIfValid() függvényben, hogy a tesztelést elindító gombok aktívak maradhatnak-e, amennyiben nem, az egér föléjük mozgatásakor a beállításra javaslatot tevő üzenet jelenik meg. Létrejön a self.__Hint nevű StringVar és a self.__HintText Label objektum, amely a gombok nevét jeleníti meg az egér föléjük mozgatásakor.

```

def __on_enterFFox(self, event):
    if self.__Config.get_Element("FireFox") == "":
        self.__Hint.set(self.__dicts.getWordFromDict(self.__Config.get_Element("Language"), "browserNotSet").replace("#browser#", "Firefox"))
    else:
        self.__Hint.set(self.__dicts.getWordFromDict(self.__Config.get_Element("Language"), "FFoxTest"))
    self.__setHintTextLocation(6.5)

```

17. ábra: Böngészőhöz tartozó gomb beállítása

5.) Létrejön a self.__CodeBox ScrolledText objektum, mely egy saját Frame-ben helyezkedik el, melynek oka a tkinter sajátos méretezési módszere. A szöveget tartalmazó elemek (gombok, beviteli mezők) hosszát és magasságát a karakterek számával adhatjuk meg, míg más elemek esetében ezt pixelben tehetjük meg, ami jelentősen megnehezíti a méretezést, ennek megoldására az alábbi programozási módszert használható:

- Frame létrehozása, melynek méreteit és helyzetét pixelben adhatjuk meg.

- A frame pack_propagate eljárását "False" paraméterrel hívjuk meg, ezáltal egyetlen gyermeket sem kényszerítheti a métere növelésére.
- Létrehozzuk a ScrolledText objektumot, irreálisan nagy mérettel, a frame-et megjelölve szülőként, így pontosan a frame területét fogja kitölteni.

A self.__CodeBox objektumhoz bindingelésre⁵⁸ kerülnek a különféle billentyű és egérparancsok, utóbbi operációs rendszerenként, mivel a Linux alatt a Tkinter az egérgörgő mozgását gombnyomásként érzékeli. Amennyiben létezik a QuickSave.txt fájl, vagyis az alkalmazás nem szabályosan zárt be, betölti annak tartalmát és a boxba illeszti, majd manuális, minden sorra kiterjedő manuális szintaxis kiemelést végez.

A felületen utoljára a jobb oldalt helyet foglaló elemek létrehozása zajlik le. A ListBox objektumok a hozzájuk tartozó ScrollBar objektummal külön-külön frame-ekbe kerülnek, melyek a pack_propagate⁵⁹ False-ra állításával a megadott kereteken belül tartják az elemeket. A ScrollBar objektumok esetében erre azért van szükség, mivel az objektumok a hagyományos módon nem méretezhetőek, kizárálag a fill paraméter X, Y vagy BOTH értékre állításával érhető el, hogy az alapértelmezett méret helyett a megadott irány(ok)ba kitöltsse a rendelkezésére álló teret⁶⁰. Mindkét ListBox-hoz tartozik egy-egy gomb, melyek szintén egy-egy Frame-ben foglalnak helyet, mivel a gombok mérete a nyelvválasztás lehetősége nyomán változhat, de nem haladhatja meg a rendelkezésére álló helyet. A gombok a felettük lévő ListBox kijelölt eleme alapján végeznek tevékenységet, a self.__recentList ListBox objektum a "default/Recent.txt" tartalmát tölti be, melyben a legutóbb megnyitott objektumok találhatóak, a legújabbal kezdve. A syntaxList esetében a "default/Syntax.txt" töltődik be, melyben a "kulcs=érték" párok a parancsokat és a hozzájuk tartozó argumentumokat tartalmazzák, vesszővel elválasztva. Mivel előfordulhat, hogy egy elem többször is szerepelhet, a lista set-té alakítását követően ismét listává válik és abc szerinti emelkedő rendezéssel kerül a listába felvitelre.

⁵⁸ Binding function in Tkinter, URL: <https://www.geeksforgeeks.org/python-binding-function-in-tkinter/>, Megtekintve: 2021.03.14.

⁵⁹ Python Frame.pack_propagate, URL: https://python.hotexamples.com/examples/Tkinter/Frame/pack_propagate/python-frame-pack_propagate-method-examples.html, Megtekintve: 2021.03.14.

⁶⁰ Python - Tkinter Scrollbar, URL: https://www.tutorialspoint.com/python/tk_scrollbar.htm, Megtekintés: 2021.03.14.

```

        self.__recentList_Frame = Frame(self.__main, width=__windowW - __relativeX - 3 -
17, height=__firstListHeight)
        self.__recentList_Frame.place(x=__relativeX, y=(__relativeY + self.__hammerFont[1] *
2))
        self.__recentList_Frame.pack_propagate(False)

        self.__recentListScroller_Frame = Frame(self.__main, width=15, height=__firstList-
tHeight)
        self.__recentListScroller_Frame.place(x=__windowW -
19, y=(__relativeY + self.__hammerFont[1] * 2))
        self.__recentListScroller_Frame.pack_propagate(False)
        self.__recentListScroller = Scrollbar(self.__recentListScroller_Frame)

        self.__recentList = Listbox(self.__recentList_Frame, width=1000, height=1000,
yscrollcommand=self.__recentListScroller.set, select-
mode=BROWSE,
                bg=self.__color, fg=self.__color2)

```

18. ábra: Frame, Listbox és Scrollbar összekapcsolása

Mivel a ListBox objektumok kijelölt elemének neve nem, csak a kijelölt sor sorszáma kérhető le a curselection függvénnyel⁶¹, külön listában (self.__recentFiles és self.__SYN) kerülnek elhelyezésre ezek az értékek és a selection, valamint a listatagok közötti összeköttetést a sorszámok biztosítják. A két lista között foglal helyet a __loadImageFrame, mely a képfájlok betöltésére létrehozott gombot és a mellette lévő ikont tartalmazza.

6.) A MainWindow objektumhoz bindingelésre kerülnek a funkcióbillentyűk, valamint az általános billentyűlenyomás és felengedés, az ablak láthatóvá tétele megtörténik és a self.createStatusLabel eljárás meghívásával kikerül az üdvözlőszöveg. Az üdvözlőszöveg eljárásnak csak a megjelenítendő szöveget kell átadni, mivel a szöveg hossza alapján állapítja meg a megjelenítés idejét. Végül a self.__main after metódusával egy folyamatos loop ciklus kötődik a fő ablakhoz, mely a konfigurációs állományban meghatározott időközönként önmagát hívja meg és a számláló lejártakor a self.__CodeBox tartalmát a "QuickSave.txt" fájlba menti.

⁶¹ Python - Tkinter Listbox, URL: https://www.tutorialspoint.com/python/tk_listbox.htm, Megtekintés: 2021.03.14.

```

def create_StatLabel(self, text):
    "Because other objects can display message on the main window, the method has an
abstract call."
    try:
        self.__destroy_StatLabel()
    except:
        pass
    self.__StatLabel = Label(self.__main, text=text, font=self.__hammerFont)
    self.__StatLabel.place(x=3, y=self.__main.winfo_height() -
self.__fontSize * 2.25)
    self.__StatLabel.after(len(text) * 50 + 1000, self.__destroy_StatLabel)

```

18. ábra: self.__StatLabel létrehozása és időzítése

III/6/1/2. HighLighter szál:

A szál létrehozza elindítja a self.__HighLighter eljárást, mely megvalósít egy végtelen ciklust. Ebben, a CPU terhelését csökkentendő, 0.1 másodpercenként kerül leellenőrzésre a self.__keyPress állapota, amennyiben False, vagyis nem történt billentyű lenyomás, vagy még javában tart a gépelés, a ciklus további elemei nem hívódnak meg. Amennyiben a az érték True, újabb 0.6 másodperces várakozás következik, mivel lehetséges, hogy a felhasználó csak rövid időre szakította meg a gépelést (tehát az érték ismét False), akkor a kiemelő költséges eljárásának meghívása feleslegesen terhelné meg a processzort, így a loop visszatér a kiinduló állapotába.

```

import time
while True:
    time.sleep(0.10)
    if self.__keyPress == True:
        time.sleep(0.60)
        if self.__keyPress == True:
            self.__keyPress = False
            self.__highLighter_Code()

```

19. ábra: highLighter loop

Amennyiben ismételten True értéket tartalmaz a self.__keyPress, az állapota False-ra változik és meghívódik a self.__highLighter_Code eljárás, mely betölti a CodeBox tartalmát és sorokra bontással egy lista állományt hoz létre, valamint az INSERT pozíció⁶² lekérésével meghatározza, melyik sorban történt módosítás. Amennyiben a delimiterban módosulás következett be, a teljes szöveg ellenőrzését biztosító self.__checkAllLines értéke True-ra változik. Ezt követően, a megjelölt területen (adott sor vagy teljes szöveg) minden tag törlésre kerül.

⁶² Tkinter 8.5 reference: a GUI for Python - 24.1. Text widget indices, URL:
<https://anzeljg.github.io/rin2/book2/2405/docs/tkinter/text-index.html>, Megtekintés: 2021.03.14.

A következő sorokban a delimiter kerül meghatározásra, amennyiben a kód első sora helyes szintaktikával adja meg azt, a delimiter szót követően egy szóközzel elválasztva, a következő szóközig bezárólag, akkor ez a karakterlánc fogja a további sorokban is a delimiter szerepét betölteni. Amennyiben a delimiter nem megállapítható, az addig használt, vagy az alapértelmezett ("%%") delimiter kerül beállításra.

```
def __getDelimiter(self):
    try:
        if self.__CodeBox.get("1.0", 2.0).startswith("delimiter"):
            regex = re.findall(r"delimiter\s.+", self.__CodeBox.get("1.0", END))[0]
    ]
        line = regex.split(" ")
        for word in line:
            if word!="delimiter" and word!="":
                return(word)

    except:
        return(self.__delimiter)
```

19. ábra: highLighter loop

A kiemelő utolsó lépésként hajtja végre a tulajdonképpeni elsődleges feladatait, melyeket a self.__standard_tinting, self.__between_tinting és self.__comment_tinting hajtanak végre.

- Az alap kiemelő a parancsok és argumentumok tagjeit adja a CodeBox soraihoz, előbbiek esetében csak az első nyitózároljelig történik a vizsgálat, illetve, amennyiben az előző sor nem tartalmazza a delimiter-t, el sem kezdődik. A soron belül a kezdőpozíciótól a következő space-ig vizsgálja a substring-et, majd, mennyiben az része a megadott szólistának, megkapja a megfelelő formázást. Amennyiben a self.__checkAllLines értéke False, csak adott sor kerül vizsgálatra.

```

temp = ""
x = 0
for charnum in range(0, len(lines[linenum])):
    if lines[linenum][charnum].isalpha() == False and lines[linenum][charnum] != "-":
        if temp in refList:
            self.__CodeBox.tag_remove(remove, str(linenum + 1) + "." + str(x), str(linenum + 1) + "." + str(charnum))
            self.__CodeBox.tag_add(tag, str(linenum + 1) + "." + str(x), str(linenum + 1) + "." + str(charnum))
            x = charnum + 1
            temp = ""
        if tag=="Arg" and lines[linenum][charnum] == "(":
            break
        else:
            temp += lines[linenum][charnum]
if temp in refList:
    self.__CodeBox.tag_add(tag, str(linenum + 1) + "." + str(x),
                           str(linenum + 1) + "." + str(len(lines[linenum])))

```

20. ábra: Parancs kifejezések kiemelése

- A köztes kijelölő kifejezetten a stringjelöléssel ellátott részek formázására szolgál, külön-külön vizsgálva a ", ' és ` karaktereket. Működése során végigmegy a dokumentum minden során, amennyiben a megadott karaktert megtalálja, megjelöli annak karakterszámát a kezdő sorszámnak, majd amennyiben újból adott karaktert találja a pozíión, záró pozícióként használja azt. Külön vizsgálat biztosítja, hogy amennyiben már adott szakasz megkapta a 'string' tag-et, a karakter megtalálása ne számítson újabb stringkijelölő kezdő vagy végpontjának. Mivel a dokumentum minden sorát vizsgálja, hatása sorokon átívelő.

```

start = 0
startY = 0
turnedOn = False
__tempStringed = []

for y in range(0, len(lines)):
    for x in range(0, len(lines[y])):
        if turnedOn == True:
            __tempStringed.append(str(y + 1) + "." + str(x+1))

        if lines[y][x] == char:
            if turnedOn == False:
                if (str(y + 1) + "." + str(x+1)) not in self.__alreadyStringed:
                    turnedOn = True
                    start = x
                    startY = y
            else:
                turnedOn = False
                if (str(str(startY + 1) + "." + str(start))) not in self.__alreadyStringed:
                    if (str(y + 1) + "." + str(x+1)) not in self.__alreadyStringed:
                        self.__CodeBox.tag_add(tag, str(startY + 1) + "." + str(
                            start),
                                               str(y + 1) + "." + str(x+1))
                        self.__alreadyStringed.extend(__tempStringed)
                        __tempStringed = []

```

21. ábra: String-ek kiemelése

- A kommentkiemelő minden sort hátukról kezdve megvizsgál és amennyiben megtalálja a delimiter stringet a sorban, e pozíiónak és a sor hosszának segítségével hozzáadja adott sorhoz a kommentkiemelő tag-ét.

```

for linenum in range(0, len(lines)):
    for charnum in range(len(lines[linenum]), 0, -1):
        if lines[linenum][charnum:charnum + len(self.__delimiter)] == self.__delimiter:
            self.__CodeBox.tag_add("comment", str(linenum + 1) + "." + str(charnum),
                                   str(linenum + 1) + "." + str(len(lines[linenum])))

```

22. ábra: Kommentkiemelő

Az eljárás utolsó tevékenységeként a minden sor ellenőrzésére kiterjedő kitétel False-ra állítódik.

III/2. Felhasználói interakció szakasza

Ebben a szakaszban a program futása folyamatos és a felhasználó által kezdeményezett folyamatok jellemzik, leszámítva a háttérben folyamatosan futó szintaxiskiemelő loop-ot, valamint a beállított időközönként meghívott automatikus mentést.

A kód betáplálásának elsődleges helye a CodeBox objektum, melyet a kiemelő folyamatosan monitoroz és az automatikus mentés időközönként elment. Amennyiben a felhasználó nyomva tartja a ctrl gombot, valamint az egérgörgőt lépteti, 12 és 48-as méret között szabadon változtathatja a betűméretet, viszont ez ne kerül mentésre addig, amíg a felhasználó nem menti az OptionsMenu ablakon a felhasználói beállításokat.

```
def __mouse_Wheel(self, event):
    """If ctrl is pressed and the user roll the mouse's wheel, the font size will be
    changed.
        Need to call updateCodeBox for the actual change."""
    if self.__box_Ctrl_Pressed:
        if ((event.delta > 0 or event.num==4) and int(self.__Config.get_Element("Box
FontSize")) < 48):
            self.__Config.set_Element("BoxFontSize", str(int(self.__Config.get_Ele
ment("BoxFontSize")) + 1))
            self.updateCodeBox()

        if ((event.delta<0 or event.num==5) and int(self.__Config.get_Element("BoxFo
ntSize")) > 12):
            self.__Config.set_Element("BoxFontSize", str(int(self.__Config.get_Eleme
nt("BoxFontSize")) - 1))
            self.updateCodeBox()
```

23. ábra: Betűméret módosítása egérgörgővel

Az ablak felső részén gombok találhatóak, melyek segítségével változatos műveletek hajthatóak végre, ezek az alábbi csoportba oszthatóak:

- Fájlműveletek
- Vágólap műveletek
- Kódgenerálás, tesztelés
- Egyéb

III/2/1. Fájlműveletek

A program a fájlműveletek kapcsán három változóban tárolja el a szerkesztett állomány állapotát: A self.__opened abban az esetben True, ha a nem új fájlon dolgozunk, csak új fájl készítésekor változik az értéke False-ra. A self.__modified ezzel szemben minden alkalommal True állapotot vesz fel, amikor a CodeBox tartalma megváltozik, és minden fájlművelet False-ra állítja az értékét. A self.__path tárolja a megnyitott fájl útvonalát, új fájl esetén az értéke "".

- **New:** Amennyiben az állomány módosított, rákérdez a mentésre a self.__askForSave metódus meghívásával. Törli a CodeBox tartalmát, az állapotjelzők a kiinduló állapotukat veszik fel. Ha a konfigurációs állomány "loadTemplate" kulcsa "True"-t tartalmaz, a CodeBox-ba betöltésre kerül a "tempates/new_file.txt" tartalma.

```
def __doNew(self):
    """if box was modified, asks if you want to save your file, and deletes the box.

    """
    if self.__modified == True:
        self.__askForSave()

        self.__deleteBox()
        self.__opened = False
        self.__path = ""

        if os.path.exists("tempates/new_file.txt") and self.__Config.get_Element("loadTemplate")=="True":
            self.__openFile("tempates/new_file.txt", False)
```

24. ábra: Új fájl készítése

- **Open:** Amennyiben az állomány módosított, rákérdez a mentésre a self.__askForSave metódus meghívásával. A filedialogban megadható, hogy milyen típusú fájl kerüljön megnyitásra ("*.boo", "*.txt" vagy bármilyen). A filedialog csak a fájl elérési útját tárolja, mely továbbításra kerül az általános fájlmegnyitó eljárás felé, egy "True" bool átadásával.

Az openfile eljárás minden esetben az elérési utat kapja meg str-ként és az addRecent tartalmát bool-ként. Az eljárás a minden sor szintaktikai ellenőrzését biztosító változó tartalmát True-ra állítja, majd, ha a fájlnév tartalmazza a "new_file.txt" szegmenst, akkor mind az addRecent és a self.__opened értéke hamis lesz, mivel az új fájl séma kerül betöltésre, ellenkező esetben a self.__opened tartalma True lesz. A következő szakaszban két körben is kísérlet következik az állomány betöltésére, első esetben szövegként importálással, utf-8 kódolással, amennyiben ez sikertelen, bináris módban⁶³. Ha minden eset kivételt dob, hibaüzenet jelenik meg a fájlmegnyitás sikertelenségéről, ellenkező esetben meghívódik a self.__openSuccess eljárás a megnyitott fájllal, az elérési úttal és az addRecent-tel, mint paraméterrel.

⁶³ Python File I/O - Read and Write Files, URL: <https://www.tutorialsteacher.com/python/python-read-write-file>, Megjelenés: 2021.03.14.

```

def __openFile(self, openname, addRecent):
    self.__checkAllLines = True
    if "new_file.txt" in openname:
        addRecent = False
        self.__opened = False

    else:
        self.__opened = True

    try:
        opened = open(openname, "r", encoding='utf-8')
        self.__openSuccess(opened, openname, addRecent)

    except Exception as e:
        try:
            opened = open(openname, "rb")
            self.__openSuccess(opened, openname, addRecent)
        except:
            m = self.__dicts.getWordFromDict(self.__Config.get_Element("Language"),
"fileOpenError").replace("#path#", str(openname))
            messagebox.showerror(
                self.__dicts.getWordFromDict(self.__Config.get_Element("Language"),
"fileOpenErrorTitle"),
                m + "\n" + str(e))

```

25. ábra: Fájlbetöltés

A self.__openSuccess eljárás a CodeBox-ba betölti a megnyitott tartalmat és meghívja az CodeBox-ot, mely beállítja a szövegméretet, a CodeBox és a ListBox-ok színeit, valamint manuálisan meghívja a self.__highLighter_Code szintaxiskiemelő eljárást. A betöltés végén a self.__path megkapja a megnyitott fájl nevét és a self.__modified False értéket kap.

```

def updateCodeBox(self):
    """Changes the light/dark them for the box, also changes the font size.
    If the listbox are existing, updates their colors too."""

    if int(self.__Config.get_Element("BoxFontSize")) > 48:
        self.__Config.set_Element("BoxFontSize", "48")
    if int(self.__Config.get_Element("BoxFontSize")) < 12:
        self.__Config.set_Element("BoxFontSize", "12")

    if (self.__Config.get_Element("DarkBox") == "False"):
        self.__color = "white"
        self.__color2 = "black"
    else:
        self.__color = "black"
        self.__color2 = "lightgray"
    hammerFont = self.__getHammerFont()

    self.__CodeBox.config(bg=self.__color, fg=self.__color2,
                          width=68,
                          height=50,
                          wrap=CHAR)

    self.__CodeBox.config(font=hammerFont)
    try:
        self.__recentList.config(bg=self.__color, fg=self.__color2)
        self.__syntaxList.config(bg=self.__color)

        if (self.__Config.get_Element("DarkBox") == "False"):

            self.__syntaxList.config(fg="blue")
            for num in range(0, self.__syntaxList.size()-1):
                if self.__SYN[num] in self.__Syntax.getKeys():
                    self.__syntaxList.itemconfig(num, {"fg": "green"})

        else:
            self.__syntaxList.config(fg="light sky blue")

            for num in range(0, self.__syntaxList.size()-1):
                if self.__SYN[num] in self.__Syntax.getKeys():
                    self.__syntaxList.itemconfig(num, {"fg": "lime"})
            self.__higher_Code()
    except Exception as e:
        pass
    self.__addTags()
    self.__CodeBox.pack()

```

26. ábra: CodeBox formázó eljárás

- **Save és SaveAs:** Amennyiben az állomány megnyitott fájl, a hagyományos mentés gomb meghívja a self.__Saver eljárást, átadva neki a self.__path értékét paraméterként, ellenkező esetben a __doSaveAs eljárás kerül meghívásra, melyet a mentés másként gomb alapértelmezetten meghív. Utóbbi esetében a filedialog bekéri

a mentendő fájl nevét és helyét, majd azt átadja a self.__Saver eljárásnak paraméterként.

A fájlmentő eljárás megvizsgálja, hogy a fájlnév üres-e, illetve tartalmazza-e a "new_file.txt" szegmenst, amennyiben bármelyik igaz, a mentés nem hajtódik végre. Ha a fájlnév nem tartalmaz kiterjesztést, a fájl végére kerül a ".boo" kiterjesztés. A CodeBox tartalma utf-8 kódolással mentésre kerül, majd a fájlművelet lezárul. A self.__opened állapota True, a self.__modified állapota False lesz, a fájl a __recentList részévé válik, a self.__path megkapja a mentett fájl elérési útját és a gyorsmentés törlésre kerül. Bármilyen kivétel keletkezése esetén messagebox jelenik meg a nyelvi szótár állományban eltárolt értékek szerint.

```
def __Saver(self, savename):
    try:
        if savename!="" and ("new_file.txt" not in savename):
            if savename.endswith(".boo") == False or savename.endswith(".txt"):
                savename += ".boo"
            opened = open(savename, "w", encoding='utf-8')
            opened.write(self.__getCodeFromBox())
            opened.close()
            self.__opened = True
            self.__modified = False
            self.__addToRecent(savename)
            self.__path = savename
            self.__deleteQuick()

    except Exception as e:
        messagebox.showerror(
            self.__dicts.getWordFromDict(self.__Config.get_Element("Language"), "fileSaveErrorTitle"),
            self.__dicts.getWordFromDict(self.__Config.get_Element("Language"), "fileSaveError").replace("#path#", savename) + "\n" + str(e))
```

27. ábra: A fájlmentés folyamata

III/2/2. Vágólap műveletek

A vágólap műveletekhez a vágólapra másolás, az onnan történő beillesztés és a korábban említett visszavonás gomb és az ahhoz kötődő UndoBuffer objektum tartoznak. A vágólap műveleteket a clipboard függvénykönyvtár segítségével hajthatjuk végre. Másolás művelet esetén a CodeBox kijelölt része egyszerű szövegként a vágólapra kerül, beillesztéskor a CodeBox "INSERT" pozíciójelöljéhez, mely a kurzor aktuális helyét jelöli, kerül a beillesztendő szöveg és a következő szintaxis kiemeléskor minden sor formázása megtörténik.

```

def __doPaste(self):
    import clipboard

    self.__CodeBox.insert(INSERT, clipboard.paste())
    self.__checkAllLines = True

def __doCopy(self):
    import clipboard
    clipboard.copy(self.__CodeBox.selection_get())

```

28. ábra: Vágólap műveletek

III/2/3. Kódgenerálás, tesztelés

A kódgenerálás meghívó függvényeit a MainWindow objektum tartalmazza, mivel a segédablak nélkül is képesnek kell lennie a CodeBox tartalma alapján html objektum legenerálására. A html kódot generáló gomb megnyomásával meghívodik a self.__getCodeOnly eljárás, mely létrehoz egy példányt a GetCodeOnly objektumból. A GetCodeOnly egy ScrolledTextBox objektumot, valamint három gombot tartalmaz, előbbi színeit a konfigurációs állomány "DarkBox" kulcsa alapján állítja be. A gombok mindegyike külön Frame-ben található annak érdekében, hogy azok ne haladhassák meg a számukra kijelölt maximális területet az ablakon.

A self.__copyButton a vágólapra másolja az ablak self.__codebox objektumának tartalmát, a self.__saveButton kezdeményezi a fájl mentését és az állományok másolását, majd bezárja az ablakot, míg a self.__cancelButton kizárolag az ablakot zárja be.

```

self.__copyButton = self.__createButton(self.__buttonsFrame1, self.__dicts.getWordFromDict(
    self.__Config.get_Element("Language"),
    "copyToClipBoard"),
    self.__doCopy)

self.__saveButton = self.__createButton(self.__buttonsFrame2, self.__dicts.getWordFromDict(
    self.__Config.get_Element("Language"),
    "save"),
    self.__doSaveAs)

self.__cancelButton = self.__createButton(self.__buttonsFrame3, self.__dicts.getWordFromDict(
    self.__Config.get_Element("Language"),
    "Cancel"),
    self.__destroyWindow)

```

29. ábra: Gombok függvényei

A kód legenerálása a master objektumon, vagyis a MainWindow-on keresztül történik, közvetlenül az ablakelemek legenerálása után, meghívva a compileCode függvényt, mely a legenerált kódot adja vissza. A CodeBox objektumból kinyert kódot két, eltérő

környezetben elkészített compiler is képes html objektummá és környezetévé fordítani, ezek egyike, mint maga az IDE, Python-ban íródott, míg az alternatív változata Fortran 90 nyelv segítségével.

- A Python alapú fordító esetében a Compiler objektum példányosításra kerül, majd miután a fordítás végbement, a publikus "compiled" változó kerül visszaadásra.
- A Fortran 90 compiler ezzel szemben a ctypes függvénykönyvtár CDLL⁶⁴ parancsával importálja Windows alatt a "FortranCompilerAsDLL.dll" állományt. A CodeBox.boo tartalma, mint temp.boo fájlba mentésre kerül, a fortran objektum compile eljárása meghívásra kerül, ha az végzett, a legenerált temp.txt importálásra kerül, majd a temp.txt törlése után a beolvasott szöveg kerül visszaadásra. Mivel a Fortran 90 kód csak C binding után használható függvénykönyvtárként a Python alapú rendszerekben, a binding korlátozza a character típusú változók hosszúságát 1-re, tömb típusú változó (character(1), allocatable) esetében viszont a Fortran függvénykönyvtár nem kapja meg a jogosultságot a memóriaterület olvasására.

Eredetileg Linux alá is elkészült volna a fordító Fortran 90 változata, viszont míg a Code::Blocks környezetében tökéletesen működött, addig Python-ból akár futtatható script, akár ".so" függvénykönyvtár állományként folyamatosan szegmentációs hibákat okozott, melyek javítása meghaladta a kompetenciahatárimat és a dolgozat kereteit.

⁶⁴ CDLL, URL: <https://www.kite.com/python/docs/ctypes.CDLL>, Megtekintés: 2021.03.14.

```

def compileCode(self):
    if self.__Config.get_Element("FortranCompiler") == "False":
        import PythonCompiler

        Compiler = PythonCompiler.Compiler(self.__CodeBox.get(0.0, END), self.__Config, self.__dicts, self.__Syntax, self)

        return Compiler.compiled

    else:
        import ctypes

        file = open("temp.boo", "w", encoding='utf-8')
        file.write(self.__CodeBox.get(0.0, END))
        file.close()

        path=os.path.abspath("FortranCompilerAsDLL.dll")
        fortran = ctypes.CDLL(path)
        fortran.compile()
        file = open("temp.txt", "rb")
        txt = file.read()
        txt = self.__createString(txt)
        file.close()
        os.remove("temp.txt")
        os.remove("temp.boo")
        return(txt)

```

30. ábra: Kódfordítók használata

III/2/2/1. Python alapú fordító

A fordító példányosítja self.__Colors néven a ColorPalettes objektumot. A ColorPalettes objektum a "default/Colors.txt" és "default/RGBA.txt" állományok alapján létrehozza a self.__colors és self.__rgbaCodes szótár állományokat. A forrásfájlok a már megismert formulával, egyenlőség jelleggel elválasztva tartalmazzák a kulcsokat és az értékeket, utóbbiak elválasztása visszük segítségével történik. Előbbi kulcsként a színsémák neveit, értékként a hozzájuk tartozó négy színt tartalmazza, míg utóbbi a színeket és a hozzájuk tartozó 3db RGB értéket. Lekérhetők a színkulcsok külön, a szín megadásával a négy sémaszín, valamint a szín megadásával annak RGB értékei.

```

def __getColors(self):
    for line in open("default/Colors.txt", "r").readlines():
        xxx = line.split("=")
        self.__colors[xxx[0].strip()] = []
        for color in xxx[1].split(","):
            self.__colors[xxx[0].strip()].append(color.strip())

def __getRGBA(self):
    for line in open("default/RGBA.txt", "r").readlines():
        xxx = line.split("=")
        self.__rgbaCodes[xxx[0].strip()] = xxx[1].strip()

```

31. ábra: Színek beolvasása

A fordító osztályszintű változókkal dolgozik, melyek tartalma a sorok értelmezése közben dinamikusan változik. A könnyebb értelmezhetőség szempontjából az alábbi táblázatban tekinthetők meg:

| Változó neve | Változó típusa | Alapértelmezett értéke |
|--------------------------------|----------------|---|
| self._author | str | "" |
| self._title | str | "" |
| self._description | str | "" |
| self._keywords | str | "" |
| self._background | str | "" |
| self._bannerText | str | "" |
| self._bannerCSS | str | "" |
| self._navbarCSS | str | "" |
| self._footerCSS | str | "" |
| self._navbarOpacity | float | 1.0 |
| self._tableOpacity | float | 1.0 |
| self._rowOpacity | float | 1.0 |
| self._footerOpacity | float | 1.0 |
| self._fontfamily | str | "Arial" |
| self._lang | str | "en" |
| self._charset | str | "UTF-8" |
| self._palette | str | "random" |
| self._mainBody | str | "" |
| self._tableCSS | str | "" |
| self._rowCSS | str | "" |
| self._mainTemplate | str | self._templateLoader("MainTemplate") |
| self._mainTemplateChanged | bool | False |
| self._bannerTemplate | str | self._templateLoader("BannerTemplate") |
| self._bannerTemplateChanged | bool | False |
| self._navbarTemplate | str | self._templateLoader("NavBarTemplate") |
| self._navbarTemplateChanged | bool | False |
| self._footerTemplate | str | self._templateLoader("FooterTemplate") |
| self._footerTemplateChanged | bool | False |
| self._containerTemplate | str | self._templateLoader("ContainerTemplate") |
| self._containerTemplateChanged | bool | False |
| self._cssTemplate | str | self._templateLoader("CSSTemplate") |
| self._cssTemplateChanged | bool | False |
| self.compiled | str | "" |
| self._error | bool/str | False |
| self._number | int | 0 |

32. ábra: A fordító állandó változói

A fordító működését a "templates/" mappában található szöveges állományok határozzák meg. Ezek kivétel nélkül html/css/javascript kódokat tartalmaznak, melyek tartalmaznak olyan, "#" jelekkel határolt részeket, melyek helyére a compiler más szövegrészleteket helyez, vagy adat hiányában üres stringgel helyettesít.

```
.row-item
{
    color: #Color4#;
    transition: 0.25s;
    display: flex;
    justify-content: center;
    align-items: center;
    border-radius: 15px;
}
```

33. ábra: Példa a behelyettesíthető szövegre (RowCSSTemplate.txt)

A behelyettesítés alanya lehet egy más meglévő, módosított template, így a sémák egymásba ágyazásával alakul ki a végleges, teljes html dokumentum.

```
<body>
    <div class="container">
        #Banner#
        #NavBar#
        #Container#
        #Footer#
    </div>
</body>
```

34. ábra: MainTemplate body része

A __templateLoader függvény a séma neve alapján összeállítja a relatív betöltési utat, és visszaadja a fájl tartalmát.

```
def __templateLoader(self, s):
    return (open("templates/" + s + ".txt", "r").read())
```

35. ábra: Sémabetöltő függvény

A fordító először megtisztítja a kódot a kommentektől, majd eltünteti a newline karaktereket. A kódban található egy speciális, html tag-nak látszó forma, mely a behúzásokat hivatott kezelni, "<space:X>" formátumban. Az X helyén minden esetben egy szám áll, melynek megfelelő számú tabulátor () kerül a "tag" helyére.

```
def __removeComments(self, code):
    return(re.sub(rf"\{self.__master.getDeliminator()\}.*\n", self.__master.getDeliminator(), code))

def __removeNewLines(self, code):
    return(code.replace("\n", ""))
```

36. ábra: Kommentek és newline karakterek törlése

```

def __spaceTags(self, code):
    import re
    parts= re.findall(r"<space:\d+>", code)
    for item in parts:
        code=code.replace("<space:"+item[7:-1]+">>", " " * int(item[7:-1]))
    return(code)

```

37. ábra: SpaceTag elemek fordítása

A kódot ezek után a delimitatorok mentén a fordító sorokra tördeli, és az így keletkezett lista elemeit soronként értelmezi, a sorszámot a self.__number változóban tárolja a hibaüzenetben való könnyebb azonosítás érdekében, mivel az üres sorokat a fordító nem számolja. Amennyiben a self.__error értéke False-ról string-re változik, a fordítás megszakad és a fordított kód helyett a hibaüzenet kerül a self.compiled változóba.

```

for line in code:
    self.__number+=1
    if re.findall(r"[a-zA-Z]", line)!=[]:
        try:
            self.__compile(line)
        except Exception as e:
            self.__error = self.__dicts.getWordFromDict(
                self.__Config.get_Element("Language"), "syntaxError").replace("#python#", str(e))
        if self.__error!=False:
            break

```

38. ábra: A soronkéti fordítás loop-ja

A self.__compile eljárás elsőként a self.__Command_and_Argument függvényvel parancsra és argumentumjaira osztja fel a sort. A parancsok állnak minden Boo-T nyelven írt sor elején, melyek az angol abc betűiből, esetleges "-" jelből és egy zárójelbe foglalt argumentumból, esetleg argumentum listából állnak.

```

def __Command_and_Argument(self, line):
    return(line.split(",")[0], line.replace(line.split(",")[0], "", 1))

```

39. ábra: A parancsot és argumentumát elválasztó függvény

Amennyiben a parancs nem szerepel a szintaxis lista kulcsai között, a program beállítja a hibaüzenetet a szótár alapján.

```

if line[0] not in self.__Syntax.getKeys():
    self.__error = self.__dicts.getWordFromDict(
        self.__Config.get_Element("Language"), "errorInvalidCommand").replace("#line#", line[0])

```

40. ábra: Hibaüzenet beállítása

Ellenkező esetben a parancs alapján egy if then else elágazás kezdődik, melyeken belül a parancsok alapján folytatódik az argumentumok vizsgálata. Egyszerűbb parancsok

esetében az argumentum értékét felveszi a parancshoz tartozó változó, míg az összetett argumentumok esetében, a vesszők mentén elválasztva folytatódik az elemek vizsgálata.

```
if line[0] == "keywords":  
    self.__keywords = args
```

41. ábra: A self.__keywords változtatás nélkül veszi át az argumentum tartalmát

A felhasználó nem valószínűsíthető, hogy minden esetben helyesen jár el a kód megírásakor, így pl. a program hibaüzenettel reagál, amennyiben a delimiter nem az első sorban kerül meghatározásra, továbbá hibaüzenetet dob akkor is, ha a Navbar, Banner, Footer elemek egynél többször kerültek meghatározásra. Más esetekben, ha többször kerül ugyanaz az elem beállításra (pl. az oldal címe vagy kódolása), az utolsó beállítás lesz meghatározó.

```
if self.__bannerTemplateChanged==False:  
    self.__bannerTemplateChanged = True  
else:  
    self.__canBeOnlySetOnceError(line[0])
```

42. ábra: Példa a többszörös beállítás tiltására

A parancsok többszörös argumentumai többnyire az alábbi formátumban adhatóak meg:

- **Egyszerű szöveg:** Többnyire linkek, id-k, számok beállításakor használatosak, vessző karakter nem használható.
- **Önálló parancs:** Boolean-ként értelmezhető tételek True-ra állítása (pl. sticky, imgfilter)
- **String formátumú szöveg:** Címek és szövegrészletek megadásakor használatos, melyek összetett argumentumon belül találhatóak és vesszőt is tartalmazhatnak. A szöveget ', " vagy ` jelek közé kell elhelyezni, javasolt, hogy a szöveg közben a másik két string-jelöltöt használjuk.
- **Kulcs/Érték pár:** Ebben az esetben a kifejezés a self.__splitByEQ függvény segítségével felosztásra kerül, előbbi a változóra utal, melynek értéket adunk.

```

args = self.__splitComma(args)
for arg in args:
    Key = self.__splitByEQ(arg, 0)
    Value = self.__splitByEQ(arg, 1)

    if Key=="container":
        self.__rowOpacity=float(Value)
    elif Key == "navbar":
        self.__navbarOpacity = float(Value)
    elif Key == "table":
        self.__tableOpacity = float(Value)
    elif Key == "footer":
        self.__footerOpacity = float(Value)
    else:
        self.__argumentError(args, "opacity")

```

43. ábra: Opacity változók beállítása

- Összetett argumentum:** Az összetett argumentum szintaxisa megegyezik a parancsokéval, zárójelbe zárva a saját argumentumaival, melyek szintén egyszerű szöveg, String formátumú szöveg, valamint Kulcs/Érték páros szintaxist használhatnak.

```

elif subArg.startswith("text"):
    textstuff = self.__splitComma(self.__Command_and_Argument(subArg
)[1][1:-1])

    self.__bannerText=textstuff[0][1:-1]
    self.__bannerTextSize=textstuff[1]
    self.__bannerTextAlign = textstuff[2]

```

44. ábra: "Banner" parancs "text" összetett argumentumának feldolgozása

Amennyiben nem elfogadható argumentum kerül bármelyik szakaszban feldolgozásra, meghívódik a self.__argumentError eljárás.

Az oldalelemek legenerálásáért felelős parancsok esetén többnyire betöltésre kerül egy séma és rendelkeznek speciális belső változókkal, melyek deklarálásukkor rendelkeznek alapértelmezett értékekkel és a parancs feldolgozása végén értékeik behelyettesítődnek a megfelelő sémába. A table és bootrow parancsok esetében, melyek az oldal törzsét alkotó táblázatok és hasábok legenerálására szolgálnak, a módosított sémák hozzáadódnak a self.__mainBody változó tartalmához, mivel ezekből tetszőleges számú helyezhető el az oldalon, viszont a hozzájuk tartozó CSS séma betöltése egyszer elegendő, mivel minden esetben ugyanaz a CSS séma tartozik adott elemekhez.

```

__rowTemplate = __rowTemplate.replace("#rowitems#", os.linesep.join(__rowItems)).rep
lace("#id#", str("id='"+_id+"'")).replace("#filter#", __imgfilter)
self.__mainBody+=__rowTemplate

```

45. ábra: Séma módosítása és self.__mainBody végéhez illesztése

Vannak speciális eljárások, melyeket kiemelnék az alapvetően egyszerűen fordítható elemek közül.

- **palette:**

A "basics" parancshoz kötődik, mely az author, language és charset elemeket is beállítja. A palette három módon állítható be: Egy 0 és 27 közötti számmal, a paletta pontos megnevezésével, vagy a "random" szó megadásával. Ha a számok intervalluma vagy a szín megnevezése nem található meg a self.__Colors objektumban található kulcsok között, a program hibaüzenetet dob. Random esetén a program elsődlegesen a datetime töredékmásodpercei⁶⁵ segítségével, hiba esetén az egérkursor pozíciójából generál véletlen-számot 0 és 27 között.

```
elif Key == "palette":
    if Value not in self.__Colors.getColors():
        try:
            if Value == "random":
                import random
                import datetime
                try:
                    random.seed(int(str(datetime.datetime.now()).split(".")[1]))
                except:
                    os.environ['PYGAME_HIDE_SUPPORT_PROMPT'] = "hide"
                    import pygame.mouse as M
                    random.seed(M.get_pos()[0] + M.get_pos()[1])
                    num = random.randint(0,27)
            else:
                num=int(Value)
                Value=list(self.__Colors.getColors())[num]
        except:
            self.__error = self.__dicts.getWordFromDict(
self.__Config.get_Element("Language"), "errorColorPalette").replace("#Value#", Value)
```

45. ábra: Séma módosítása és self.__mainBody végéhez illesztése

- **animation:**

A banner parancshoz kötődik, animáló fejlécet tesz lehetővé. Esetében nem oldható meg egyszerű behelyettesítés, mivel a megadott elemek száma változó, az animáció idejét követően a linkek felsorolása következik, ezért először meg kell állapítani az egy tételere fordítható arányt az animációban, valamint az áttűnések arányos hosszát.

```
stuff = self.__Command_and_Argument(subArg)[1][1:-1]
stuff = self.__splitComma(stuff)
self.__time=stuff[0]
move = (100//(len(stuff)-1))//10
still = 100//(len(stuff)-1) - move
number = 0
```

46. ábra: A fentiek megvalósítása

⁶⁵ Python Datetime, URL: https://www.w3schools.com/python/python_datetime.asp, Megjelenés: 2021.03.14.

Az animáció elkészítéséhez egy adott sémába kell a megfelelő alkalmatossal elkészíteni az állapotot jelző sort, a százalékkal együtt, valamint a végén a 100%-os állapottal, a legelső formával zárni az animációt. A képek áttűnéskor homályossá és fekete-fhérré válnak, így kevésbé zavaróak az esetleges méret- és felbontásbeli különbségek. A self.__animPartMaker függvény behelyettesíti a megfelelő értékeket.

```

animpart = "\t\t#number#{ background-
image: url('#img#'); filter: saturate(#sat#) blur(#blur#px) ;}" + os.linesep

for imageNum in range(1, len(stuff)):
    __replacer += self.__animPartMaker(animpart, str(number), stuff[imageNum], "3", "0
.25")
    __replacer += self.__animPartMaker(animpart, str(number+move), stuff[imageNum], "0
", "1")
    number+=still
    __replacer += self.__animPartMaker(animpart, str(number-
move), stuff[imageNum], "0", "1")
    __replacer += self.__animPartMaker(animpart, str(number), stuff[imageNum], "3", "0
.25")
    number+=move
    __replacer+=animpart.replace("#number#", "100").replace("#img#", stuff[1]).replace(
"#blur#", "3").replace("#sat#", "0.25")
    self.__bannerAnimation = self.__bannerAnimation.replace("#animationThings#", __rep
lacer)

def __animPartMaker(self, animpart, number, img, blur, sat):
    return animpart.replace("#number#", str(number)).replace("#img#", img).replace(
#blur#, blur).replace( "#sat#", sat)

```

47. ábra: Az animációs elemek legenerálása

- **rate:**

A Bootstrap rácsrendszer egyik alapszabálya, hogy a sorokat tizenkét oszlopra osztja és ezen belül határozzák meg, hogy a sorba helyezett elemeink mekkora hányadát foglalják annak el. A Boo-T esetében a bootrow parancson belül a rate argumentum kétféle értéket vehet fel, az "auto" esetében tesztelőleges számú elem helyezhető el benne és a méretezéskor egyenlő szélességet vesznek fel, illetve megadhatóak az arányok, visszövel elválasztva. A compiler ellenőrzi, hogy a számok összege 12 e, valamint, hogy az elemek száma megegyezik-e az arányszámok számával, ellenkező esetben hibaüzenetet dob.

```

    elif item.startswith("rate"):
        if "auto" in item:
            __rates="auto"
        else:
            __rates = self.__splitComma(self.__Command_and_Argument(item)[1][1:-1])
            sum=0
            for number in __rates:
                sum+=int(number)
            if sum!=12:
                self.__error = self.__dicts.getWordFromDict(
                    self.__Config.get_Element("Language"), "errorNot12")

```

48. ábra: A rate arányszámainak ellenőrzése

```

if __rates!="auto" and len(__rowItems)!=len(__rates):
    self.__error = self.__dicts.getWordFromDict(
        self.__Config.get_Element("Language"), "errorNoMatch")

```

49. ábra: Elemek számának ellenőrzése

- **footer:**

A footer elem tartalmazza a közösségi média ikonokat, melyeknek a különféle képernyőméretekben ugyanúgy dinamikusan és ízlésesen kell megjelennie, ezért az elemek száma alapján változik, mekkora részét tölhetik ki a soroknak a közösségi média ikonok.

```

if len(__socials) == 0:
    xs=12
elif len(__socials) < 5:
    xs = int(12/len(__socials))
elif len(__socials) < 7:
    xs = 4
elif len(__socials) > 6 :
    xs = 3
for media in social_icons:
    if media in __socials.keys():
        __icons.append(str("\t\t\t<div class='col-"+str(xs)+" col-md text-center'>" + os.linesep + "\t\t\t\t<a href='"+ __socials[media] + "' target='_blank'><i mg src='img/"+media+".png' class='img-fluid'></a>" + os.linesep + "</div>"))

```

50. ábra: Közösségi média ikonok méretezése

Amennyiben a fordítás sikeresen zárult, a __createCompiled metódus összefűzi a sémákat, a behelyettesíthető kifejezésekkel pedig, amennyiben adott template módosításra került, a generált tartalomra, amennyiben nem, üres stringre cseréli. Ezt követően történik meg a színek behelyettesítése, amennyiben adott elem opacity értéke nem 1.0, RGBA értékkel. Egyéb esetben a self.compiled a hibaüzenetet fogja tartalmazni.

```

def __addColors(self):
    for number in range(0,4):
        if self.__palette!="":
            self.__compiledReplacer("#Color"+str(number+1)+"#", self.__Colors.getPalette(
self.__palette)[number])

```

```

        self.__compiledReplacer("#NavbarOpacityColor"+str(number+1)+"#", str("rgba(" +
+ self.__Colors.getRGBA(self.__Colors.getPalette(self.__palette)[number]) + "," + str(se
lf.__navbarOpacity) + ")"))
        self.__compiledReplacer("#TableOpacityColor"+str(number+1)+"#", str("rgba(" +
self.__Colors.getRGBA(self.__Colors.getPalette(self.__palette)[number]) + "," + str(sel
f.__tableOpacity) + ")"))
        self.__compiledReplacer("#RowOpacityColor"+str(number+1)+"#", str("rgba(" + s
elf.__Colors.getRGBA(self.__Colors.getPalette(self.__palette)[number]) + "," + str(self.
__rowOpacity) + ")"))
        self.__compiledReplacer("#FooterOpacityColor"+str(number+1)+"#", str("rgba(" +
+ self.__Colors.getRGBA(self.__Colors.getPalette(self.__palette)[number]) + "," + str(se
lf.__footerOpacity) + ")"))

```

51. ábra: Színek behelyettesítése

III/2/2/2. Fortan 90 alapú fordító

A Fortran 90 fordító dll függvénykönyvtárként kerül importálásra a CDLL függvény által. Maga a fájl kiterjesztése és a modern Fortran általában a Fortran 90 kifejezéssel kerül említésre, de a kód során használtam olyan elemeket, melyek 1995-ben vagy 2003-ban kerültek a nyelvbe, illetőleg a GNU Fortran fordítóba⁶⁶.

A Fortran programok jellemzően szintaxis terén könnyen értelmezhetőek egy, a nyelvet behatóan nem ismerő programozó számára is, viszont mivel a programok, különösen a program és függvények, eljárások deklarációs szakaszai hosszúak, ezek áttekintése nehézkessé válhat. Az egyik alapszabály, hogy a program és minden modul tartalmazza az "implicit none"⁶⁷ kifejezést a use szekciók után, mivel ezáltal hibás vagy elmaradt deklarálás esetén a program egyértelmű hibajelzést ad, elmaradása esetén az i, j, k, l, m, n kezdetű változók integer, minden más real típust kap, illetve, hasonlóan a Python szerkesztési elvekhez, Fortran alatt is érdemes minden fontosabb, többször végrehajtandó és esetlegesen nehezen olvasható műveleteket külön eljárásokba vagy függvényekbe elhelyezni⁶⁸. Általában, ha egy sémába kívántam értéket elhelyezni, egy eddig értékkel fel nem töltött változónak kívántam értéket adni vagy kisebb méretű karakterláncban kíntam módosítást eszközölni, függvényekkel dolgoztam, viszont amennyiben egy hosszsabb, dinamikus méretezésű karakterláncban hajtottam végre változást, inkább szubrutinokat használtam a megfelelő változó intentjének inout megjelölésével⁶⁹. Egy strukturált program komponenseinek tesztelése is lényegesen egyszerűbbé válik⁷⁰.

⁶⁶ Fortran History. URL: <http://fortranwiki.org/fortran/show/Fortran+History>, Megtekintés: 2021.03.14.

⁶⁷ Implicit None, URL: <http://www.personal.psu.edu/jhm/f90/statements/implicit.html>, Megtekintés: 2021.03.14.

⁶⁸ Norman S. Clerman, Walter Spector: Modern Fortran, Cambridge University Press [USA], 2012, 3-4. o.

⁶⁹ Fortran/Fortran procedures and functions, URL:

https://en.wikibooks.org/wiki/Fortran/Fortran_procedures_and_functions, Megtekintve: 2021.03.14.

⁷⁰ Norman S. Clerman, Walter Spector: Modern Fortran, Cambridge University Press [USA], 2012, 3-5. o.

Ezen felül, a modern Fortran 2003 óta lehetőséget nyújt a Type nevű összetett adatállomány osztályá alakítására, mely eredeti működése a C nyelvekben ismert Structures adattípuséhoz hasonlít, melyhez a contains adattag segítségével, egy belső függvénynév és egy külső függvény " $=>$ "-vel való összekapcsolásával osztályszintű metódusokat alkothatunk⁷¹, köztük a konstruktornal, melyet manuálisan kell meghívni. Fontos, hogy class adattípusként a metódusokban és függvényben szerepelnie kell a meghívó objektumnak, ennek segítségével Fortran alatt is biztosíthatóak az OOP alapelvek⁷².

Mivel a Fortran továbbra is viszonylag kevésé ellátott string műveletekkel, valamint teljes egészében mellőzi az összetett adatszerkezeteket, leszámítva a tömb és type típusokat, ezért a szövegkezelő függvényeket, az automatikus fájlkezelőket és operációs rendszer szintű változókat, valamint a lista és szótár összetett adatszerkezeteket magamnak kellett implementálnom. Ezek teljes dokumentálása meghaladja a dolgozat kereteit, ezért csak a program szempontjából fontosabbakat említeném.

1) String műveletek (string_func.f90):

- **string_replace_all**

A megkapott string hosszánál az behelyettesítendő string hossza + 1000 hosszúságú új stringet képezünk, majd egy do while ciklus során folyamatosan veszük a lecserélendő substring méterének megfelelő hosszúságú szegmenseket, egyezés esetén a stringben belül cserére kerül minden előfordulása a keresett substringnek.

⁷¹ Fortran/structures, URL: <https://en.wikibooks.org/wiki/Fortran/structures>, Megtekintve: 2021.03.14.
Norman S. Clerman, Walter Spector: Modern Fortran, Cambridge University Press [USA], 2012, 182-183. o.

```

function string_replace_all(string, original, new) result(new_string)
    character(LEN=*, kind=1) :: string, original, new
    character(LEN=len(string)+len(new)+1000, kind=1) :: new_string
    integer :: x1, x2, dif

    !dif = len(new) - len(original)
    new_string = string
    x1=1
    x2=x1+len(original)-1

    do while(x2<len(new_string))
        x2 = x1+len(original)-1
        if (new_string(x1:x2) == original) then
            new_string = new_string(1:x1-1)//new//new_string(x2+1:len(new_string))
            !x1 = x1 - dif

        end if
        x1 = x1 + 1

    end do
    new_string = trim(new_string)

end function

```

52. ábra: string_replace_all függvény

- **startswith/endswith**

Visszaadja értékül, hogy adott string a megadott szövegrésszel kezdődik vagy ér véget.

```

function startswith(string, start) result(isIt)
    character(len=*, kind=1) string, start
    logical :: isIt

    isIt = .FALSE.
    if (string(1:len(start)) == start) isIt = .TRUE.

end function

function endswith(string, ender) result(isIt)
    character(len=*, kind=1) string, ender
    logical :: isIt

    isIt = .FALSE.
    if (trim(string(len(string)-
len(ender)+1:len(string))) == trim(ender)) isIt = .TRUE.

end function

```

53. ábra: startswith és endswith függvények

- **split**

A delimiter szerint felosztja a stringet annyi szegmensre, amit maximális értéknek kapott. Elsöként megvizsgálja, hány alkalommal szerepelt a delimiter a stringben, ez alapján allokálja a tömböt, melynek minden eleme olyan hosszú, mint a feldarabolandó

string, végül addig vizsgálja a kezdőponttól a stringet, amíg a delimiterba nem fut, ekkor a megállapítva a végpontot és a substringet a tömbhöz adja, míg az új kezdőpont a végpont és a delimiter hossza alapján ismét megállapításra kerül.

```

function split(string, delimiter, maximum) result(temp_array)
    character(len=*, kind=1) :: string, delimiter
    character(len=len(string), kind=1), dimension(:), allocatable :: temp_array
    integer :: num, x1, x2, alloc_stat, maximum, PrevPoint, delay

    num = 1

    do x1 = 1, len(string), 1
        x2 = x1+len(delimiter)-1
        if (string(x1:x2)==delimiter) then
            num=num+1
        endif
        if (num==maximum) exit
    end do

    !allocate(character(longest)::temp_array%array(num), stat = alloc_stat)
    !write(*,*) size(temp_array%array)
    allocate(temp_array(num), stat=alloc_stat)

    num = 1
    PrevPoint = 1
    delay = 0
    do x1 = 1, len(string), 1
        x2 = x1+len(delimiter)-1
        if (delay>0) then
            delay = delay - 1
            cycle
        end if
        if (string(x1:x2)==delimiter) then
            temp_array(num) = string(PrevPoint:x1-1)

            PrevPoint = x1 + len(delimiter)
            delay = len(delimiter) - 1
            num = num + 1
        end if
    end do

    temp_array(num) = string(PrevPoint:len(string))

end function

```

52. ábra: split függvény

- **splitBySpaces**

Az előző függvény némileg módosított, jelen fordítónkhoz alakított változata, mely a többszörös szóközök okozta üres elemek elkerülése végett, némileg más módon osztja fel a stringet.

```

function splitBySpaces(string, maximum) result(temp_array)
    character(:), allocatable :: string
    integer :: maximum, x1, x2, num, theSize, wordNum, alloc
    character(10000), dimension(:), allocatable :: temp_array
    logical :: newWord

    newWord = .TRUE.
    x1 = 1
    x2 = 1
    theSize = 0
    do num = 1, len(string), 1
        if (string(num:num) == " ") then
            if (newWord .EQV. .FALSE.) then
                theSize = theSize +1
            end if
            newWord = .TRUE.
        else
            if (newWord .EQV. .TRUE.) then
                x1 = num
                x2 = num
                newWord = .FALSE.
            else
                x2 = num
            end if
        end if
    end do
    newWord = .TRUE.
    x1 = 1
    x2 = 1
    wordNum = 1
    if (theSize>maximum) theSize = maximum
    allocate(temp_array(theSize), stat=alloc)

    do num = 1, len(string), 1
        if (string(num:num) == " ") then
            if (newWord .EQV. .FALSE.) then
                if (wordNum<theSize) then
                    temp_array(wordNum) = trim(string(x1:x2))
                    wordNum = wordNum +1
                else
                    temp_array(wordNum) = trim(string(x1:len(string)))
                    exit
                end if
            end if
            newWord = .TRUE.
        else
            if (newWord .EQV. .TRUE.) then
                x1 = num
                x2 = num
                newWord = .FALSE.
            else
                x2 = num
            end if
        end if
    end do
end function

```

53. ábra: splitBySpaces függvény

- **lower**

Kisbetűssé alakítja a megadott stringet.

```
function lower(s) result(s2)
    character(len=*, kind=1) :: s
    character(len=:, kind=1), allocatable :: s2
    integer :: num

    s2=s
    do num=1, len_trim(s), 1
        if (ichar(s(num:num))>64 .AND. ichar(s(num:num))<91) then
            s2(num:num) = char(ichar(s(num:num))+32)

        end if
    end do
end function
```

54. ábra: splitBySpaces függvény

- **charToInt, intToChar, charToReal, realToChar**

Egyszerű konverziókat biztosító függvények. Integer character-ré konvertálása esetében annak hossza megegyezik az integer számjegyeinek hosszával, realból való konverzió esetén az egész rész kitölti a szükséges teret, míg a törtrész 12 számjegyig kerül feltüntetésre. Amennyiben a törtszám nullánál kisebb, kiegészül a visszaadott character egy nullával.

```

function charToInt(c) result(i)
    character(len=*) :: c
    integer :: i

    read(c, *) i
end function

function intToChar(i) result(c)
    character(len=10000) :: c_temp
    character(len=:), allocatable :: c
    integer :: i

    write(c_temp, "(I0)") i
    c = trim(c_temp)
end function

function charToReal(c) result(r)
    character(len=*) :: c
    real :: r

    read(c, *) r
end function

function realToChar(r) result(c)
    character(len=10000) :: c_temp
    character(len,:), allocatable :: c
    real :: r
    integer :: num

    write(c_temp, "(F0.12)") r
    if (c_temp(1:1) == ".") c_temp = "0//c_temp
    do num = len_trim(c_temp), 1, -1
        if (c_temp(num:num) == "0") c_temp(num:num) = " "
        if (c_temp(num:num) == ".") exit
    end do

    c = trim(c_temp)
    if (c(len(c):len(c)) == ".") c = c//"0"
end function

```

55. ábra: Konvertáló függvények

2) File műveletek (file_func.f90):

- **loadText**

A megadott elérési úton lévő fájlt betölti a load függvényel, az arrayToText függvényel szöveggé fűzi, megállapítja az utolsó érvényes karakter pozíóját, majd az így kapott szöveget visszaadja.

```

function loadText(path) result(text)
    character(len=:, kind=1), allocatable :: path, text
    character(len=10000, kind=1), dimension(:), allocatable :: array
    integer :: lastChar, io

    array = load(path)
    text = arrayToText(array)
    !write(*,*) path, ":", len(text), ":", text(len(text)-10: len(text))
    lastChar = getLastChar(text)
    text=text(1:lastChar)
end function

```

56. ábra: Fájlbetöltés egyszerű szövegként

- **getTheLastChar**

Az ASCII tábla szerinti utolsó érvényes karakter pozíóját adja vissza adott karakterláncban.

```

function getLastChar(text) result(lastChar)
    integer :: num, lastChar
    character(:, ), allocatable :: text
    lastChar=1
    do num=1,len_trim(text), 1
        if (ichar(text(num:num))>32 .AND. ichar(text(num:num))<127) lastChar=num

    end do
end function

```

57. ábra: Utolsó érvényes karakter megállapítása

- **loadLines**

Betölti a megadott útvonalon található fájl tartalmát a load függvényvel és visszaadja a tömböt.

```

function loadLines(path) result(array)
    character(len=:, kind=1), allocatable :: path
    character(len=10000, kind=1), dimension(:), allocatable :: array

    array = load(path)

end function

```

57. ábra: Fájl betöltése tömbként

- **load**

Ez a függvény végzi a tulajdonképpeni fájlbetöltést. Előként ellenőrzi, hogy a megadott úton fájl elérhető-e, amennyiben nem, hibaüzetet ír ki a konzolra, ellenkező esetben megszámolja a fájl sorait, allokálja a tömböt a memóriában, majd betölti a sorokat és lezárja a fájlt.

```

function load(path) result(array)
    character(len=:, kind=1), allocatable :: path
    character(len=10000, kind=1), dimension(:), allocatable :: array
    integer :: io, lineNumber, alloc, num
    logical :: exists

    inquire(file=path, exist=exists)
    if (exists .EQV. .FALSE.) then
        write(*,"(A,A,A)") "File(", path, ") does not exist!"
        allocate(array(0), stat=alloc)
    else
        lineNumber=checkLines(path)
        allocate(array(lineNumber), stat=alloc)
        open(UNIT=11, FILE=path, iostat=io, STATUS="old", action="read")
        do num = 1, lineNumber, 1
            read(11, "(A)", iostat=io) array(num)
        end do

        close(11)
    end if
end function

```

58. ábra: Fájlbetöltés automatizálása függénnyel

- **checklines**

Adott fájl sorainak számával tér vissza.

```

function checkLines(path) result(lineNumber)
    character(len=:], allocatable :: path
    integer :: io, lineNumber
    character(len=:], allocatable :: temp

    open(UNIT=11, FILE=path, iostat=io, STATUS="old", action="read")
    lineNumber=1
    do
        read(11, "(A)", iostat=io) temp
        if (io/=0) exit
        lineNumber = lineNumber + 1
    end do
    close(11)

end function

```

58. ábra: Sorok megszámolása adott fájlból.

- **arrayToText**

A megadott tömb elemmeit szöveggé fűzi össze.

```

function arrayToText(array) result(text)
    character(len=:, kind=1), allocatable :: text
    character(len=10000, kind=1), dimension(:), allocatable :: array
    integer :: num

    text=""
    do num=1, size(array), 1
        text = text//trim(array(num))//achar(13)//achar(10)

    end do
end function

```

59. ábra: Összefűzés tagolt szöveggé

- **arrayToTextWithoutNewLines**

A fentivel megegyező, de newline karakter nélkül.

- **saveText**

Szövegként kimenti a megadott karakterláncot.

```

subroutine saveText(path, string)
    character(len=:, kind=1), allocatable :: string, path
    integer :: io

    open(UNIT=11, FILE=path, iostat=io, STATUS="replace", action="write")
    write(*,*) trim(string)
    close(11)

end subroutine

```

60. ábra: Mentés egyszerű szövegként

3) Véletlenszám generálás (**randomInteger.f90**):

- **randInt**

A megadott intervallumban véletlenszámot generál. A beépített GNU függvény egy 0-1 közötti real-t ad vissza⁷³, majd a maximum-minimum különbségével felszorozva, a minimumot hozzáadva a kívánt intervallumon belül tud számokat visszaadni.

```

function randInt(minN, maxN) result(numb)
    logical :: integ
    integer :: numb, minN, maxN
    real :: temp

    CALL RANDOM_NUMBER(temp)
    numb = (temp*(MaxN-MinN))+MinN

end function

```

61. ábra: Véletlenszám generálás

⁷³ RANDOM_NUMBER — Pseudo-random number, URL:
https://gcc.gnu.org/onlinedocs/gfortran/RANDOM_005fNUMBER.html, Megtekintés: 2021.03.14.

4) Operációs rendszer műveletek (os_func.f90):

- **getFilesFromSubFolder**

Meghívja adott operációs rendszer parancssorát és a megadott mappáról listát készítet és azt fájlba íratja⁷⁴, majd a listát betölti tömbként.

```
function getFilesFromSubFolderWin(dir) result(array)
    character(:), allocatable :: dir, command, path
    character(10000), dimension(:), allocatable :: array, array2
    integer :: alloc, num

    call system(command="dir //dir// /s /b /o:gn>filelist.txt")
    path = "filelist.txt"
    array2 = loadLines(path)

    allocate(array(size(array2)-1), stat=alloc)
    call system(command="del filelist.txt")
    !allocate(array(1), stat=alloc)
    do num = 1, size(array2)-1, 1
        array(num) = array2(num)
    end do
end function
```

62. ábra: Mappa tartalmának bekérése Windows alatt

- **getSep**

Operációs rendszer mappaelválasztó jelének lekérése.

```
function getSep(os_name) result(sep)
    character(len,:), allocatable :: os_name
    character(1) :: sep

    if (os_name=="Windows") then
        sep="\"
    else
        sep="/"
    end if
end function
```

62. ábra: Mappaelválasztó jelet visszaadó függvény

⁷⁴ How do I print a listing of files in a directory?, URL:
<https://www.computerhope.com/issues/ch000772.htm>, Megtekintés: 2021.03.14.

- **getLineSep**

Visszaadja a kiválasztott operációs rendszer szerinti newline karaktert.

```
function getLineSep(os_name) result(sep)
    character(len=::), allocatable :: os_name
    character(::), allocatable :: sep

    if (os_name=="Windows") then
        sep=achar(13)//achar(10)
    else
        sep=achar(10)
    end if
    sep = trim(sep)
end function
```

63. ábra: Újsor karakter visszaadása

- **getFileNameFromPath**

Egy elérési útvonalból visszaadja a fájlnevet. Választható opció, hogy a kiterjesztéssel vagy az nélkül kérjük vissza a nevet.

```
function getFileNameFromPath(path, os, ext) result(filename)
    character(::), allocatable :: path, filename, os
    logical :: ext
    character(100), dimension(::), allocatable :: parts
    !Ext is about if you want to have the extension or don't.

    parts = split(path, getSep(os), 99999)
    filename = parts(size(parts))
    if (ext .EQV. .FALSE.) then
        parts = split(filename, ".", 2)
        filename = parts(1)
    end if

end function
```

64. ábra: Fájlnév kinyerése az elérési útból

5) Lista adatszerkezet (**list.f90**):

A lista adatszerkezet megvalósításával a Fortran egyik legnagyobb hiányossága került pótlásra, mivel a korábbi fix méret, vagy az azzal szemben álló manuális allokáció/deallokáció rendkívül sok hibalehetőséget eredményezett⁷⁵. A listát, a célnak megfelelően, szöveges tömblistaként hoztam létre⁷⁶, melyben egész- és valós számok is eltárolhatóak a már korábban megmutatott konvertálási lehetőségek okán.

⁷⁵ Common Causes of Segmentation Faults (Segfaults), URL:
[https://www.nas.nasa.gov/hecc/support/kb/common-causes-of-segmentation-faults-\(segfaults\)_524.html](https://www.nas.nasa.gov/hecc/support/kb/common-causes-of-segmentation-faults-(segfaults)_524.html),
Megtekintés: 2021.03.14.

⁷⁶ Különbség a tömblista és a kapcsolt lista között, URL:
<https://hu.sawakinome.com/articles/software/difference-between-array-list-and-linked-list-3.html>,
Megtekintés: 2021.03.14.

```

type CharList
    private
        character(len=10000, kind=1), dimension(:), allocatable :: list1
        character(len=10000, kind=1), dimension(:), allocatable :: list2
        integer :: lenght, alloc_stat, insertLoc, longest
        logical :: secondIsTheMain

    contains
        procedure :: create => Constructor
        procedure :: add => Add
        procedure :: get => Get
        procedure :: getAll => GetAll
        procedure :: insert => Insert
        procedure :: removeByNum => RemoveByNum
        procedure :: removeByName => RemoveByName
        procedure :: indexOf => getLocation
        procedure :: containsItem => ContainsItem
        procedure :: sort => sorter
        procedure :: getSize => GetSize
        procedure :: set => setter
        procedure :: occurrences => Occurrences
        procedure :: RemoveDuplicates
        procedure :: getAsIntegers => GetAsIntegers
        procedure :: getAsReals => GetAsReals
        procedure :: removeNaN => RemoveNaN
        procedure :: real2Char => Real2Char
        procedure :: char2Real=>Char2Real
        procedure :: int2Char=>Int2Char
        procedure :: char2Int=>Char2Int
    end type

```

65. ábra: Lista adatszerkezet függvényeivel

A lista a hagyományos tömblista felépítését követi. Két tömb található benne, melyek 10000 karakter hosszú character elemeket tartalmazhatnak. A listának tartalmaznia kell az aktuálisan leghosszabb eleme méretét, az utolsó valódi elem pozíóját, valamint, hogy a két tömb közül melyik van éppen használatban.

A dinamikus méretezés működése egyszerű: Amennyiben a lista méretét már meghaladná a beillesztett elemek száma, az éppen nem használt tömb deallkolásása után duplázott mérettel allokálódik, az aktív tömb elemei átmásolásra kerülnek és a secondIsTheMain értéke reciprokára változik.

Fontos, hogy Fortranban nincsen automatikus konstruktora az osztályoknak⁷⁷, a felhasználónak magának kell meghívnia azokat, a call list%create forma szerint.

⁷⁷ Object-Oriented Programming in Fortran 2003 Part 1: Code Reusability, URL: <https://gist.github.com/n-s-k/522f2669979ed6d0582b8e80cf6c95fd>, Megtekintés: 2021.03.14.

```

subroutine Constructor(this)
    class(CharList), intent(inout) :: this
    this%lengtht = 8
    this%insertLoc = 1
    this%secondIsTheMain = .FALSE.
    this%longest=0
    allocate(this%list1(this%lengtht), stat=this%alloc_stat)
    allocate(this%list2(this%lengtht), stat=this%alloc_stat)
end subroutine

```

66. ábra: Lista adatszerkezet konstruktora

A lista létrejöttekor minden két tömbjét nyolcas méterre allokálja, a következő beillesztés helyét az első hely pozícióra állítja és az első tömböt teszi aktívvá. A listán a szokásos műveletek végezhetőek el.

- **add**

Segítségével a list végére illeszthető elem. Amennyiben már meghaladnánk a méretet, meghívásra kerül a duplázó függvény.

```

subroutine Add(this, value)
    class(CharList), intent(inout) :: this
    character(len=:, kind=1), allocatable :: value

    if (this%insertLoc == this%lengtht) call doubleTheSize(this)
    if (this%secondIsTheMain.EQV..TRUE.) then
        this%list2(this%insertLoc) = value
    else
        this%list1(this%insertLoc) = value
    end if
    this%insertLoc = this%insertLoc + 1
    if (len(value)>this%longest) this%longest = len(value)
end subroutine

```

67. ábra: Elem hozzáadása a listához

- **doubleTheSize (private)**

Megduplázza a maximális méretet és elvégzi a szükséges allokációkat, valamint az adatok átmásolását.

```

subroutine doubleTheSize(this)
    class(CharList), intent(inout) :: this
    integer :: num

    this%length = this%length*2
    if (this%secondIsTheMain.EQV..FALSE.) then
        this%secondIsTheMain = .TRUE.
        deallocate(this%list2)
        allocate(this%list2(this%length), stat=this%alloc_stat)
        this%list2(1:size(this%list1))=this%list1(1:size(this%list1))
    else
        this%secondIsTheMain = .FALSE.
        deallocate(this%list1)
        allocate(this%list1(this%length), stat=this%alloc_stat)
        this%list1(1:size(this%list2))=this%list2(1:size(this%list2))
    end if
end subroutine

```

68. ábra: A méretet dinamikusan növelő függvény

- **get**

Visszaadja az adott helyen, az aktuálisan aktív többen szereplő értéket, lenyelves a felesleges szóközököt.

```

function Get(this, num) result(item)
    class(CharList) :: this
    integer :: num
    character(len=:, kind=1), allocatable :: item

    if (this%secondIsTheMain.EQV..FALSE.) then
        item = trim(this%list1(num))
    else
        item = trim(this%list2(num))
    end if
end function

```

69. ábra: Optimalizált adatkérés

- **getAll**

A lista minden tagját visszaadja tömbként, az aktuálisan legnagyobb méterű adattag méretére allookálva.

```

function GetAll(this) result(array)
    class(CharList) :: this
    integer :: alloc_stat
    character(len=this%longest, kind=1), dimension(:), allocatable :: array

    allocate(array(this%insertLoc), stat=alloc_stat)
    if (this%secondIsTheMain.EQV..TRUE.) then
        array = this%list2(1:this%insertLoc-1)
    else
        array = this%list1(1:this%insertLoc-1)
    end if
end function

```

70. ábra: minden elem lekérése

▪ **insert**

Elem beszúrása a megadott pozícióra. Az elemeket jobbra tolja, szükség esetén bővíti a lista memóriaterületét.

```
subroutine Insert(this, value, location)
    class(CharList), intent(inout) :: this
    character(len=:, kind=1), allocatable :: value
    integer :: location

    if (this%insertLoc == this%length) call doubleTheSize(this)
    if (this%secondIsTheMain .EQV..TRUE.) then
        this%list2(location+1:size(this%list2)) = this%list2(location:size(this%list2)-1)
        this%list2(location) = value

    else
        this%list1(location+1:size(this%list1)) = this%list1(location:size(this%list1)-1)
        this%list1(location) = value

    end if
    this%insertLoc = this%insertLoc + 1
    if (len(value)>this%longest) this%longest = len(value)
end subroutine
```

71. ábra: Elem beszúrása megadott helyre

▪ **removeByNum**

Elem törlése adott pozíóról, az utána következőket eggyel balra tolja.

```
subroutine RemoveByNum(this, location)
    class(CharList), intent(inout) :: this
    integer :: location

    if (this%secondIsTheMain .EQV..TRUE.) then
        this%list2(location:size(this%list2)-1) = this%list2(location+1:size(this%list2))
    else
        this%list1(location:size(this%list1)-1) = this%list2(location+1:size(this%list1))
    end if
    this%insertLoc = this%insertLoc - 1
    call checkLongest(this)
end subroutine
```

72. ábra: Sorszám szerinti törlés

▪ **checkLongest (private)**

Visszaadja a listában található leghosszabb elem hosszát.

```

subroutine checkLongest(this)
  class(CharList), intent(inout) :: this
  integer :: num

  this%longest=0
  if (this%secondIsTheMain .EQV..TRUE.) then
    do num=1, this%insertLoc-1, 1
      if (this%longest<len_trim(this%list2(num))) this%longest=len_trim(this%list2(num))
    end do
  else
    do num=1, this%insertLoc-1, 1
      if (this%longest<len_trim(this%list1(num))) this%longest=len_trim(this%list1(num))
    end do
  end if
end subroutine

```

73. ábra: Leghosszabb elem méretének megállapítása

▪ **removeByName**

Törli megadott elem első előfordulását a listából. Megállapítja a getLocation segítségével az első előfordulást, majd a fent már megismert pozíció alapján történő törlést alkalmazza.

```

subroutine RemoveByName(this, value)
  class(CharList), intent(inout) :: this
  character(len=:, kind=1), allocatable :: value
  integer :: X
  X = getLocation(this, value)
  call RemoveByNum(this, X)
end subroutine

```

74. ábra: Törlés név alapján

▪ **indexOf**

Visszaadja megadott elem első előfordulási helyét, amennyiben nem található, nullát ad vissza.

```

function getLocation(this, value) result(X)
    class(CharList), intent(inout) :: this
    character(len=:, kind=1), allocatable :: value
    integer :: X, num

    X = 0
    do num=1, this%insertLoc-1, 1
        if (this%secondIsTheMain.EQV..TRUE.) then
            if (trim(value) == trim(this%list2(num))) then
                X = num
                exit
            end if
        else
            if (trim(value) == trim(this%list1(num))) then
                X = num
                exit
            end if
        end if
    end do
end function

```

75. ábra: Elem helyének lekérése

▪ **containsItem**

Visszaadja, hogy adott elem megtalálható-e a listában. Amennyiben a getLocation függvény nulla értéket ad vissza, .FALSE., amennyiben nem, .TRUE. értékkel tér vissza.

```

function ContainsItem(this, value) result(con)
    class(CharList), intent(inout) :: this
    character(len=:, kind=1), allocatable :: value
    logical :: con

    if (getLocation(this, value)/=0) then
        con = .TRUE.
    else
        con = .FALSE.
    end if
end function

```

76. ábra: Contains függvény

▪ **set**

Adott pozíció lévő elem értékét megváltoztatja.

```

subroutine setter(this, value, num)
    class(CharList), intent(inout) :: this
    character(len=:, kind=1), allocatable :: value
    integer :: num

    if (this%secondIsTheMain.EQV..TRUE.) then
        this%list2(num) = value
    else
        this%list1(num) = value
    end if
end subroutine

```

76. ábra: Elem módosítása adott pozíión

- **getSize**

Visszaadja a lista abszakt méretét, vagyis a következő elem helyénél eggyel kisebb értéket.

```
function GetSize(this) result(s)
    class(CharList), intent(inout) :: this
    integer :: s

    s = this%insertLoc-1

end function
```

77. ábra: Méret lekérése

A további függvények és eljárások, mivel a fordítóban nem kerültek felhasználásra, nem kerülnek hasonlóan részletes kifejtésre. A listában lehetőség van megszámolni adott elem előfordulását, törölni a duplákat, ezáltal halmazzá alakítani a listát, rendezni az elemeket abc szerint, lekérni az elemeket integer vagy real tömbként (akár nagyság szerint rendezve is), törölni a számként nem értelmezhető elemeket.

6) Dictionary adatszerkezet (dict.f90):

A szótár adatszerkezet a lista adatszerkezetre épít, két listából áll, ezek egyike a kulcsokat, a másik a hozzájuk tartozó értékeket tartalmazza. Mivel az adatszerkezet függvényeinek nagy része a megfelelő lista adatszerkezet függvényét hívja meg, csak a fontosabbak kifejtését tartom szükségesnek.

```
type Dictionary
    type(CharList) :: keys, values

    contains
        procedure :: create => DictConstructor
        procedure :: add => AddDict
        procedure :: get => GetDict
        procedure :: remove => RemoveDict
        procedure :: getKeys => GetKeys
        procedure :: getValues => GetValues
        procedure :: containsKey => containsKey
        procedure :: containsValue => containsValue
        procedure :: getSize => GetSizeDict
        procedure :: set => SetKey
        procedure :: getAsInt => GetAsInt
        procedure :: getAsReal => GetAsReal
    end type
```

78. ábra: Szótár adatszerkezet

- **add**

Amennyiben a kulcs még nem létezik, létrehozza azt, hozzáadva a kulcsot és az értéket a listákhoz, ellenkező esetben az adott kulcs pozícióján lévő értéket írja felül.

```

subroutine AddDict(this, key, value)
  class(Dictionary), intent(inout) :: this
  character(:, ), allocatable :: key, value

  if (this%keys%containsItem(key).EQV..FALSE.) then
    call this%keys%add(key)
    call this%values%add(value)
  else
    call SetKey(this, key, value)
  end if
end subroutine

```

79. ábra: Szótárhoz új elem rendelése

▪ **remove**

Lekéri a kulcs helyét a kulcslistából, majd az azon a pozícion lévő értéket törli, majd a kulcsot is eltávolítja.

```

subroutine RemoveDict(this, key)
  class(Dictionary), intent(inout) :: this
  character(:, ), allocatable :: key, value

  call this%values%removeByNum(this%keys%index0f(key))
  call this%keys%removeByName(key)
end subroutine

```

80. ábra: Törlés a szótárból

▪ **getKeys / getValues / containsKey / containsValue**

Természetesen van lehetőség lekérni a kulcs vagy érték listák tartalmát is tömbként lekérni, illetve, megvizsgálni, hogy megtalálható-e adott kulcs vagy érték a szótárban.

```

function GetKeys(this) result(array)
  class(Dictionary) :: this
  character(len=10000, kind=1), dimension(:), allocatable :: array

  array = this%keys%getAll()
end function

function GetValues(this) result(array)
  class(Dictionary) :: this
  character(len=10000, kind=1), dimension(:), allocatable :: array

  array = this%values%getAll()
end function

```

81. ábra: Lekérések a szótár elemeivel

```

function containsKey(this, key) result(bool)
    class(Dictionary) :: this
    character(:), allocatable :: key
    logical :: bool

    bool = this%keys%containsItem(key)

end function

function containsValue(this, value) result(bool)
    class(Dictionary) :: this
    character(:), allocatable :: value
    logical :: bool

    bool = this%values%containsItem(value)

end function

```

81. ábra: Egyszerű tartalmazás vizsgálat

7) Fordító tényleges működése:

A Fortran 90 alapú fordító működése nagyrészt megegyezik a Python alapúval, miután a hasonló összetett adatszerkezetek implementálásra kerültek, a legnagyobb különbség az egyes részelemek a szegmentációs és allokációs problémák elkerülése érdekében történő, nagyobb számú lemezre írásában található.

Mivel a Python objektumok nem adhatóak át a Fortran számára, annak el kell készítenie a saját szótárait, melyekből dolgozik, erre külön függvényt hoztam létre.

```

path = "default/Syntax.txt"
SyntaxList = create_Dict(path)
path = "default/Colors.txt"
ColorsList = create_Dict(path)
path = "default/RGBA.txt"
RGBAList = create_Dict(path)
path = "Config.txt"
Config= create_Dict(path)

```

81. ábra: Szótárak létrehozása

A betöltő függény betölti egyesével a sorokat, majd elválasztja őket az egyenlőségjel mentén, az elsőt a kulcsá, utóbbit az értékké alakítva.

```

function create_Dict(path) result(TempDict)
    type(Dictionary) :: TempDict
    character(len=100), dimension(:), allocatable :: temp, line_array
    character(len,:), allocatable :: path, line, command, arguments
    integer :: num

    temp = loadLines(path)
    call TempDict%create()

    do num = 1, size(temp), 1
        line = trim(temp(num))
        line_array = split(line, "=", 2)
        command = trim(line_array(1))
        arguments = trim(line_array(2))
        call TempDict%add(command, arguments)
    end do

end function

```

82. ábra: Ideiglenes szótár létrehozása és visszaadása

Sajátos adattípus a wordDict, mely a self.__dict-tel megegyező szerepet tölt be. Egy karakterláncot tartalmaz a nyelvvel, valamint egy szótárat, mely a kifejezéseket és a hozzájuk tartozó szövegeket tartalmazza.

```

type wordDict
    character(:), allocatable :: language
    type(Dictionary) :: words

end type

```

83. ábra: wordDict adattípus

A gyakorlatban ezen adattípus tömbje kerül felhasználásra, melynek során minden forrásfájlt betöltünk a "dicts/" mappából és azokat feldolgozva a megszokott kulcs=érték képzéssel szótárat hoz létre, a fájl nevéből pedig elkészíti a karakterláncot, mely a nyelvet jelöli.

```

function createDicts(os, dir) result(dicts)
    type(wordDict), dimension(:), allocatable :: dicts
    character(:), allocatable :: os, dir, filename, temp, pattern, word, message, de
liminator
    integer :: alloc, num , num2
    character(100), dimension(:), allocatable :: listOfFiles, parts
    character(10000), dimension(:), allocatable :: lines

    listOfFiles = getFilesFromSubFolder(os, dir)
    allocate(dicts(size(listOfFiles)), stat=alloc)

    do num = 1, size(listOfFiles), 1
        temp = trim(listOfFiles(num))
        pattern = ".txt"
        if (endswith(trim(temp), pattern)) then
            filename = getFileNameFromPath(temp, os, .FALSE.)
            dicts(num)%language = filename
            lines = loadLines(temp)
            call dicts(num)%words%create()
            do num2=1, size(lines), 1
                delimiter="="
                parts = split(lines(num2), delimiter, 2)
                word = trim(parts(1))
                message = trim(parts(2))
                call dicts(num)%words%add(word, message)
            end do
        end if
    end do

end function

```

83. ábra: A wordDict konstruktora

Szavakat a már megszokott módon lehet a dicts objektumból lekérni.

```

function getWordFromDict(dicts, lang, word) result(message)
    character(:), allocatable :: lang, word, message
    type(wordDict), dimension(:), allocatable :: dicts
    integer :: num

    message = "Not found!"

    do num = 1, size(dicts), 1
        if (dicts(num)%language == lang) message = trim(dict(dicts(num)%words%get(word)))
    end do
end function

```

84. ábra: Szavak lekérése

A Python szülőprogram és a Fortran kód között a merevlemez teremt kapcsolat, mely nem elengáns megoldás, viszont a C/Fortran szövegbeli inkompatibilitás miatt elkerülhetetlen, ugyanis a Fortran külön tárolja el a stringjei hosszát és a karaktereket magukat csak egy char-tömbként kezeli, addig a C szabadon írja a karaktereket a memóriába, majd egy null

terminatorral zárja le őket⁷⁸, így a C binding mindenkorban 1 char hosszúságú "stringek" átadását teszi lehetővé. Alternatív megoldásként felmerült a karakterek egyenkénti átadása, majd a null terminator olvasását követően a tényleges fordítás indítása, de hatékonyság szempontjából nem okozna teljesítménynövekedést.

Miután a "temp.boo" fájlból a kód sorai betöltésre kerültek, a kommentek, a sorok elején található esetleges felesleges szóközök törlésre kerülnek, majd a szöveg newline karakterek nélküli összefűzése után a delimitár mentén ismét sorokra bontásra kerül, melyeket a program majd egyesével feldolgoz. Mivel a Fortran erősen típusos nyelv⁷⁹, a hibát az errorText karakterlánc tartalma jelzi, amennyiben nem "No Error" a tartalma, a feldolgozás megszakad.

A sablonok a templateLoader eljárás segítségével töltődnek be, mely a nevet várja bemenetként.

```

subroutine templateLoader(string, name)
    character(:), allocatable, intent(inout) :: string
    character(:), allocatable :: name, path

    path = "templates://"//name//".txt"
    string = loadText(path)
    string = trim(string)

end subroutine

```

85. ábra: Sablon betöltése

A soronkénti fordítást a compileLine eljárás végzi. Az elvek megegyeznek a Python fordítóban leírtakkal, az egyetlen tényleges különbség, hogy a honlapelemek, miután elkészültek (background, banner, navbar, footer), a lemezre kerülnek kírásra, míg a mainBody.txt append művelettel gyarapodik, ha új elem kerül legenerálásra.

```

subroutine appendFile(string, name)
    character(:), allocatable :: name, string, temp
    integer :: io

    open(unit=11, file=name, status="unknown", position="append", action="write")
    write(11,*) string
    close(11)

end subroutine

```

86. ábra: Fájlhoz írás Fortran alatt

⁷⁸ Converting Between FORTRAN and C Strings, URL:

<http://starlink.eao.hawaii.edu/docs/sun209.htm/sun209se6.html>, Megtekintés: 2021.03.14.

⁷⁹ Strongly Typed Language: <https://www.sciencedirect.com/topics/computer-science/strongly-typed-language>, Megtekintés: 2021.03.14.

Miután a kód lefordult és a honlap részei ideiglenesen kimentésre kerültek, a finalBuilder eljárás készíti el a végleges honlapot. Ez a már megismert templateLoader-rel betölti a MainTemplate és CSSTemplate állományokat, majd a honlapelemeket is folyamatosan betölti, ügyelve arra, hogy az utolsó valós karakterig használjuk fel a betöltött állományt, majd a templateChanger segítségével a megfelelő helyre illesztjük, amennyiben a hozzá tartozó változó értéke .TRUE., ellenkező esetben üres char-t helyezünk el.

```
subroutine templateChanger(compiled, var, string, changeTo)
    character(:), allocatable, intent(inout) :: compiled
    character(:), allocatable :: string, changeTo
    logical :: var
    if (var .EQV. .TRUE.) then
        compiled = string_replace_all(compiled, string, changeTo)
    else
        compiled = string_replace_all(compiled, string, "")
    end if
end subroutine
```

87. ábra: Tartalom beillesztése a sémába

Amennyiben egy fájl már felhasználásra került, vagy arra kísérlet történt, kísérlet történik a törlésére is, ezt a deleteFile eljárás teszi meg.

```
subroutine deleteF(path)
    character(:), allocatable :: path
    integer :: io

    open(unit=1234, iostat=io, file=path, status='old')
    if (io == 0) close(1234, status='delete')

end subroutine
```

88. ábra: Fájl törlése Fortranban

A színek behelyettesítése szintén a már megismert módon történik, a kód "temp.txt" néven kerül elmentésre.

III/2/2/3. HTML környezet generáló script

Az elkészült kódot a self.__saveButton-ra kattintva menthetjük el, ennek során létrejön a SaveHTML objektum példánya. Ebben elsőként az objektum bekéri a felhasználótól a mentés helyét, amennyiben nem kapja ezt meg készen, amennyiben böngészővel való tesztelés történik, a "temp/temp.html" alapján egy új temp mappa lesz a mentés helye.

```

if self.__Config.get_OS_Name() == "Windows":
    filepath = "/".join(savename.split("/")[0:-1]) + "/"
    sep = "/"
else:
    filepath = os.sep.join(savename.split(os.sep)[0:-1]) + os.sep
    sep = os.sep

if savename == "temp/temp.html":
    filepath=os.getcwd()+sep+"temp"+sep

```

89. ábra: Mentés helyének megállapítása

Amennyiben a mentés helyén nem létezik img és bootstrap mappa, ezek létrehozásra kerülnek, illetve, amennyiben a mentendő fájlnév nem ".html"-ra végződik, akkor kiegészítésre kerül.

A self.__searchForImagesPaths függvény felkutatja a html kódban található képekre mutató linkeket, majd amennyiben nem webcímre mutatnak, a kép másolásra kerül az "img/" mappába⁸⁰, amennyiben más fájl található ott ezen a néven, a self.__getSaveName a self.__changeName segítségével, egy számláló generálásával egy három számjegyből álló értéket helyez el a fájlnév mögé (tehát elmeletben 1000 azonos nevű kép importálható).

```

def __getSaveName(self, filename, sep, filepath, already):
    if filename in already:
        savename = self.__changeName(str(filepath + "img" + sep), filename, sep)
    else:
        savename = filename
        already.append(filename)
    return(savename)

def __changeName(self, path, file, sep):
    num = 1
    file2=file
    if (os.path.exists(path+file)):
        file2 = ".".join(file.split(".")[:-1])+"_000."+file.split(".")[-1]
        while (os.path.exists(path+file2)):
            file2 = ".".join(file.split(".")[:-1])+"_"+str(f'{num:03d}')+"."+file.split(".")[-1]
            num=num+1
    return (file2)

```

90. ábra: A név megváltoztatását lehetővé tevő kód részlet

A fájlok másolása a self.__copyFile eljárással történik, mely bármilyen fájlt képes egy másik elérési útra átmásolni.

⁸⁰ Python | shutil.copy() method, URL: <https://www.geeksforgeeks.org/python-shutil-copy-method/>, Megtekintés: 2021.03.14.

```

def __copyFile(self, dir, file, sep, sourcedir, destfile):
    from shutil import copyfile
    src = sourcedir+sep+file
    dest = str(dir + destfile).replace(os.sep, sep)
    copyfile(src, dest)

```

91. ábra: Fájlmásoló függvény

```

def __searchForImagesPaths(self, code, sep, filepath, ):
    import re
    regex=re.findall(r"src='[-:a-zA-Z0-9./\\]+'", code)
    regex2=re.findall(r"url\('[-:a-zA-Z0-9./\\]+'\)", code)
    already = []
    source_paths = []
    for item in regex:
        temp = item.replace("src=", "").replace("'", "")
        img_types = ["jpg", "bmp", "gif", "png", "tiff", "webp", "avif", "ico", "ap
ng", "svg", "jpeg"]

        if (temp.split(".")[-1] in img_types) and temp.startswith("http") == False:
            if (temp.startswith("/") or temp[1:3] == ":/"):
                filename=temp.split(sep)[-1]
                if filename in ["email.png", "phone.png", "skype.png", "facebook.png",
"youtube.png", "instagram.png", "vkontakte.png", "googleplus.png", "linkedin.png", "twit
ter.png", "github.png"]:
                    continue
                savename = self.__getSaveName(filename, sep, filepath, already)

                if temp in source_paths:
                    pass
                else:
                    source_paths.append(temp)
                    self.__copyFile(str(filepath+"img"+sep), filename, sep, sep.join
(temp.split(sep)[0:-1]), savename)
                    code=code.replace(item, "src='img/" + savename + "'")
                    code=code.replace(item.replace("src", "href", 1), "href='img/" + sav
ename + "'")
            else:
                filename=temp.split(sep)[-1]
                if filename in ["email.png", "phone.png", "skype.png", "facebook.png",
"youtube.png", "instagram.png", "vkontakte.png", "googleplus.png", "linkedin.png", "twit
ter.png", "github.png"]:
                    continue
                savename = self.__getSaveName(filename, sep, filepath, already)

                if temp in source_paths:
                    pass
                else:
                    source_paths.append(temp)
                    sourcedir = sep.join(self.__path.split(sep)[:-
1]) + sep + sep.join(temp.split(sep)[0:-1])
                    self.__copyFile(str(filepath+"img"+sep), filename, sep, sourcedi
r, savename)
                    code=code.replace(item, "src='img/" + savename + "'")
                    code=code.replace(item.replace("src", "href", 1), "href='img/" + sav
ename + "'")
            for item in regex2:

```

```

        temp = item.replace("url","",).replace("!", "").replace(")", "")
        img_types = ["jpg", "bmp", "gif", "png", "tiff", "webp", "avif", "ico", "ap
ng", "svg", "jpeg"]

        if (temp.split(".")[-1] in img_types) and temp.startswith("http") == False:
            if (temp.startswith("/") or temp[1:3] ==(":/"):
                filename=temp.split(sep)[-1]
                if filename in ["email.png","phone.png","skype.png","facebook.png",
"youtube.png", "instagram.png", "vkontakte.png", "googleplus.png", "linkedin.png", "twit
ter.png", "github.png"]:
                    continue
                savename = self.__getSaveName(filename, sep, filepath, already)
                if temp in source_paths:
                    pass
                else:
                    source_paths.append(temp)
                    self.__copyFile(str(filepath+"img"+sep), filename, sep, sep.join
(temp.split(sep)[0:-1]), savename)
                    code =code.replace(item, "url('img/" + savename + "')")
                else:
                    filename=temp.split(sep)[-1]
                    if filename in ["email.png","phone.png","skype.png","facebook.png",
"youtube.png", "instagram.png", "vkontakte.png", "googleplus.png", "linkedin.png", "twit
ter.png", "github.png"]:
                        continue
                    savename = self.__getSaveName(filename, sep, filepath, already)
                    if temp in source_paths:
                        pass
                    else:
                        source_paths.append(temp)
                        sourcedir = sep.join(self.__path.split(sep)[::-
1]) + sep + sep.join(temp.split(sep)[0:-1])
                        self.__copyFile(str(filepath+"img"+sep), filename, sep, sourcedi
r, savename)
                    code =code.replace(item, "url('img/" + savename + "')")

    return(code)

```

92. ábra: Képek másolása

A felhasználó által beállított képeken felül a közösségi média ikonok, valamint a bootstrap mappa tartalma is másolásra kerül.

```

for img in ["email.png","phone.png","skype.png","facebook.png", "youtube.png", "ins
tagram.png", "vkontakte.png", "googleplus.png", "linkedin.png", "twitter.png", "github.p
ng"]:
    self.__copyFile(str(filepath+"img"+ sep), img, sep, str(os.path.abspath(os.getc
wd())+sep+"icons"), img)

    for root, dirs, files in os.walk("bootstrap/"):
        for file in files:
            self.__copyFile(str(filepath+"bootstrap"+ sep), file, sep, str(os.path.absp
ath(os.getcwd())+sep+"bootstrap"), file)

```

93. ábra: Közösségi média ikonok és bootstrap fájlok másolása

A kód gyors teszteléséhez lehetőség van a böngészők ikonjaival fémjelzett ikonokkal egy hasonló fordítást és generálást elvégezni, melynek végén a frissen generált oldal a böngészőbe töltődik. minden gomb a hozzá tartozó függényt hívja meg, amely elsőként ellenőrzi, létezik-e már a temp mappa, amennyiben igen, törli annak tartalmát.

```
def __removeTemp(self):
    if os.path.exists("temp"):
        import shutil
        shutil.rmtree("temp")

def __doViewFireFox(self):
    try:
        self.__removeTemp()
        self.__createTempAndViewItInBrowser("FireFox")
    except Exception as e:
        messagebox.showerror(title=self.__dicts.getWordFromDict(self.__Config.getElement("Language"), "testError"), message=e)
```

94. ábra: FireFox indító eljárás, valamint a temp mappát törlő függvény

A meghívott eljárás elsőként létrehozza a temp mappát, majd lefuttatja a már ismert függvényt, megjelölve, hogy "temp/temp.html" lesz az elérési út. Végül, a self.__Config állományban eltárolt elérési út alapján, a subprocess függvény segítségével meghívásra kerül a böngésző az oldallal, mint paraméterrel.

```
def __openSiteWithBrowser(self, browser, site):
    import subprocess
    if self.__Config.get_OS_Name() == "Linux":
        subprocess.run([self.__Config.get_Element(browser), site])
    else:
        subprocess.Popen(str(self.__Config.get_Element(browser)) + " " + site))
```

95. ábra: Böngészővel való tesztelés alkalmazás elindításával

III/2/4. Egyéb függvények

Az egyéb függvények tartalmazzák azokat a menüpontokat, amelyik minden hagyományos desktop alkalmazás részét képezik.

A beállítások menü létrehoz egy példányt az OptionsMenu objektumból, melynek segítségével testre szabható a Boo-T működése. A már megismert, működést meghatározó állományok (self.__dicts, self.__Config, self.__hammerFont, __monitor). a böngészők ikonjai átadásra kerültek és létrejön egy fix méretű TopLevel objektum, mely minimálisan, de a képernyő méretéhez igazodik. Az ablakelemek a MainWindow self.createStatusLabel eljárásán keresztül a fő ablakon jelenítenek meg segítségeket az ablakelemekkel kapcsolatban.

```

if __s[0] < 800 or __s[1] < 600:
    __w = 480
    __h = 240
else:
    __w = 640
    __h = 380

```

96. ábra: OptionMenu méreteződése

Jelen ablakelem négy fő részre osztható, melyek a következők:

- **Compiler beállítások**

A fordító beállításoknál egy LabelFrame objektum jön létre, melyen 1-1 Label (image), Label (text) és RadioButton foglal helyet. Amennyiben a programot Linux operációs rendszeren futtatjuk, a compiler minden esetben Python-ra kerül beállításra.

```

self.__compilerPython, self.__imgPython, self.__imgPythonLabel = self.__compiler
LabelButton(1, "Python", 10, hammerheight, hammerFont, "icons/python.png")
self.__compilerFortran, self.__imgFortran, self.__imgFortranLabel = self.__compi
lerLabelButton(2, "Fortran", round(__w / 4), hammerheight, hammerFont, "icons/fortran.
png")

if self.__Config.get_OS_Name() == "Linux":
    self.__compilerFortran.config(state="disabled")
    self.__imgFortranLabel.config(state="disabled")

```

97. ábra: Python compiler felület beállítása

- **Böngészők útvonalának beállítása**

Szintén egy LabelFrame objektumon belül található. A keret tartalmaz egy Checkbutton objektumot a hozzá tartozó Label-lel, melynek segítségével az "AutoCheckForInstalledBrowsers" kulcs értéke állítható be, befolyásolva az indításkori böngésző ellenőrzést. Ezen felül négy Label található a hozzá tartozó Button objektummal, melyek megnyitják a böngészők útjának beállítására szolgáló filedialog ablakot, a self.__Config objektumban már megismert módon.

```

self.__iconFireChrome, self.__buttonSettingsChrome = self.__createBrowserLabelButton(
    self.__imgChrome, hammerheight, __buttonWidth, 0, self.__setPathChrome, "Chrome")
def __createBrowserLabelButton(self, image, hammerheight, w, plus, command, browserna
me):
    icon = Label(self.__browserFrame, image=image, width=32, height=32)
    icon.place(x=5, y=round(hammerheight / 4) + hammerheight + plus)
    button = Button(self.__browserFrame, width=w, command=command,
                    text=self.__dicts.getWordFromDict(self.__Config.get_Element("Language"),
                                                    "browserPathButtonText").replace(
                                                    "#browser#", browsename), font=self.__hammerFont)
    button.place(x=44, y=round(hammerheight / 4) + hammerheight + round(
        (32 - hammerheight / 2) / 8) + plus)
    return(icon, button)

```

98. ábra: Böngésző beállító ikon és gomb a hozzá tartozó létrehozó függvényel

- Alkalmazás általános beállításai

Ezen LabelFrame tartalmazza az alkalmazás működésének legalapvetőbb változóit. A LabelFrame felső részében három Label és a hozzájuk tartozó OptionMenu található. A nyelvet beállító OptionMenu tartalmát a "dir/" mappában elérhető nyelvi állományok határozzák meg, míg a színek beállításáért felelős self.__colorOption az aktuálisan beállított színséma szerint változtatja meg saját színét a __changeOptionColorCOLOR segítségével. Az automatikus monitorozás érdekében egy trace kerül rendelésre a self.__boxColorVar állapotához, mely meghívja a fenti függvényt, amennyiben állapotában változás történt.

Lehetőség van beállítani a nyelvet, a programablak méretét (illetve azt automatikussá tenni, valamint a ListBox és TextBox elemek színsémáit beállítani világos vagy sötét téma.

```
self.__changeOptionColorCOLOR("a", "b", "c")
self.__boxColorVar.trace_add("write", self.__changeOptionColorCOLOR)
def __changeOptionColorCOLOR(self, a, b, c):
    """Changes color of the optionmenu to white or black, depending on it's value"""
    if self.__boxColorVar.get() == self.__dicts.getWordFromDict(self.__Config.get_Element("Language"), "light"):
        self.__colorOption.config(bg="white", fg="black", activebackground="white", activeforeground="black")
    else:
        self.__colorOption.config(bg="black", fg="lightgray", activebackground="black", activeforeground="lightgray")
```

99. ábra: Az OptionMenu-t színező függény és a változóhoz trace rendelése

Ezen felül CheckBox-ok és Entry-k segítségével adhatóak meg a további konfigurációs beállítások. A CheckBox állapota aktívvá vagy inaktívvá teszi az Entry. utóbbi esetben törölve annak értékét, ezek szintén a változójukhoz rendelt trace-ekkel valósulnak meg.

```
self.__boxInf.trace_add("write", self.__checkInfBox)

def __checkInfBox(self, a, b, c):
    self.__enableDisableBox(True, self.__boxInf.get(), self.__recentEntry, self.__recentNum)
def __enableDisableBox(self, reverse, val, entry, num):
    """Sets entry if checkbox is changed."""
    if reverse==True:
        val = not val
    if val==True:
        entry.config(state=NORMAL)
    else:
        num.set("")
        entry.config(state=DISABLED)
```

100. ábra: Trace rendelése változóhoz és a meghívott függvények

Lehetőség van beállítani a CodeBox betűméretét és azt automatikussá tenni, beállítani az utoljára megnyitott állományok maximális számát és azt végtelenné tenni, megadni, milyen időközönként történjen automatikus mentés, valamint be és kikapcsolni, hogy megjelenjen-

e indításkor a betöltőkép, valamint, hogy új fájl létrehozásakor betöltsük-e a sémát. A CheckBox-ok értékeihez szintén trace-t rendelünmk, hogy az Entry-k állapotát automatikusan megváltoztathassuk.

- **Állapot módosító gombok**

A Frame három Button objektumot tartalmaz, melyek közül a self.__mainButtonOk először a konfigurációs állományba menti, majd meghívja a beállításokat fájlba mentő függvényt, a self.__mainButtonCancel egyszerűen csak bezárja az ablakot, míg a self.__mainButtonDefaults betölti a "default/Config.txt" tartalmát a konfigurációs állományba és menti is az aktuális konfigurációs fájlba. Ekkor, amennyiben a self.__haveToReSize állapota True, a MainWindow objektumon meghívásra kerül a createMainWindow eljárás, mely ismételten létrehozza annak megjelenését az új méretekkel.

```

def __saveSettingsToConfig(self):
    self.__haveToReSize=False

    if self.__intCompiler.get()==1:
        self.__Config.set_Element("FortranCompiler", "False")
    else:
        self.__Config.set_Element("FortranCompiler", "True")

    if self.__autoSearch.get()==0:
        self.__Config.set_Element("AutoCheckForInstalledBrowsers" , "False")
    else:
        self.__Config.set_Element("AutoCheckForInstalledBrowsers" , "True")

    if self.__boxColorVar.get()==self.__dicts.getWordFromDict(self.__Config.get_Element("Language"), "light"):
        self.__Config.set_Element("DarkBox", "False")
    else:
        self.__Config.set_Element("DarkBox", "True")

    temp=self.__Config.get_Element("Language")
    self.__Config.set_Element("Language", self.__langVar.get())

    temp=self.__Config.get_Element("StaticSize")
    if self.__windowSize.get()=="Auto":
        self.__Config.set_Element("StaticSize", "0")
    else:
        self.__Config.set_Element("StaticSize", self.__windowSize.get()[0])
    if temp!=self.__Config.get_Element("StaticSize"):
        self.__haveToReSize=True

    if self.__boxAuto.get()==True:
        self.__Config.set_Element("BoxFontSize", "0")
    else:
        self.__Config.set_Element("BoxFontSize", self.__boxFontSize.get())

    if self.__boxInf.get()==True:
        self.__Config.set_Element("MaxRecent", "0")
    else:
        self.__Config.set_Element("MaxRecent", self.__recentNum.get())

    if self.__boxQuick.get() == False:
        self.__Config.set_Element("AutoSave", "0")
    else:
        self.__Config.set_Element("AutoSave", self.__quickNum.get())

    if self.__loadDisplay.get() == False:
        self.__Config.set_Element("noLoading", "True")
    else:
        self.__Config.set_Element("noLoading", "False")

    if self.__loadTemplate.get() == True:
        self.__Config.set_Element("loadTemplate", "True")
    else:
        self.__Config.set_Element("loadTemplate", "False")

```

101. ábra: Beállítások konfigurációba mentése

```

def __setWindowLayout(self):
    if self.__Config.get_Element("FortranCompiler") == "True":
        self.__intCompiler.set(2)
    else:
        self.__intCompiler.set(1)

    if self.__Config.get_Element("AutoCheckForInstalledBrowsers") == "True":
        self.__autoSearch.set(1)
    else:
        self.__autoSearch.set(0)

    self.__langVar.set(self.__Config.get_Element("Language"))

    self.__windowSize.set(self.__windowSSize[int(self.__Config.get_Element("StaticSi
ze"))])
    if self.__Config.get_Element("DarkBox") == "True":
        self.__boxColorVar.set(self.__dicts.getWordFromDict(self.__Config.get_Elemen
t("Language"), "dark"))
    else:
        self.__boxColorVar.set(self.__dicts.getWordFromDict(self.__Config.get_Elemen
t("Language"), "light"))

    if self.__Config.get_Element("BoxFontSize")=="0":
        self.__boxAuto.set(True)
        self.__boxFontSize.set("")
    else:
        self.__boxAuto.set(False)
        self.__boxFontSize.set(self.__Config.get_Element("BoxFontSize"))

    if self.__Config.get_Element("MaxRecent")=="0":
        self.__boxInf.set(True)
        self.__recentNum.set("")
    else:
        self.__boxInf.set(False)
        self.__recentNum.set(self.__Config.get_Element("MaxRecent"))

    if self.__Config.get_Element("AutoSave")=="0":
        self.__boxQuick.set(False)
        self.__recentNum.set("")
    else:
        self.__boxQuick.set(True)
        self.__quickNum.set(self.__Config.get_Element("AutoSave"))

    if self.__Config.get_Element("noLoading")=="False":
        self.__loadDisplay.set(True)
    else:
        self.__loadDisplay.set(False)

    if self.__Config.get_Element("loadTemplate")=="True":
        self.__loadTemplate.set(True)
    else:
        self.__loadTemplate.set(False)

    self.__master.updateCodeBox()

```

102. ábra: Ablakelemek beállítása a konfigurációs állomány alapján

Az alkalmazáshoz tartozik egy súgó menü is, mely a `self.__openWithDefaulBrowser` eljárás segítségével megnyitja az "examples/Help/Help.html" fájlt az alapértelmezett böngészőben a webbrowser függvénykönyvtár segítségével, mely az `open` függvényenél egyszerűen megnyitoja az adott elérési utat.⁸¹ Hitelességi okokból fontosnak éreztem, hogy a súgó menü egy olyan honlap legyen, melyet magával az alkalmazással is könnyen el lehet készíteni.

```

def __doHelp(self):
    done = False
    try:
        self.__openWithDefaulBrowser(os.getcwd() + os.sep + "examples/Help/Help.html")
        done = True
    except:
        pass

    if done == False:
        for browser in ["Chrome", "FireFox", "Edge", "Opera"]:
            if (self.__Config.get_Element(browser) != ""):
                self.__openSiteWithBrowser(browser, "examples/Help/Help.html")
                done=True
                break
    if done == False:
        messagebox.showerror(self.__dicts.getWordFromDict(self.__Config.get_Element("Language"), "noBrowser"), self.__dicts.getWordFromDict(self.__Config.get_Element("Language"), "noBrowserError"))

def __openWithDefaulBrowser(self, site):
    import webbrowser
    webbrowser.open(site)

```

103. ábra: Alapértelmezett böngészőben megnyitás függvényei

Az alkalmazáshoz tartozik még egy névjegy, mely az `AboutMenu` objektum példányosításával jelenik meg, az elemeket az `AboutM` Toplevel objektumra helyezi fel, melynek mérete nem idomul a képernyő felbontásához. Az ablakban kis animációk és játék tekinthetők meg, kizárolag a Tkinter elemeinek felhasználásával. Az `authorFrame` és a `versionFrame` tartalmazza az `authorLabel` és `versionLabel` elemeket, melyek a készítő nevét és a program verziószámát tartalmazzák, a Toplevel kötött, 0.02 másodpercenként meghívott `self.__Animation` folyamatosan módosítja a Label-ek X pozícióját, ezáltal azt az érzést keltheti, hogy a szövegelemek ellenirányú mozgással pattognak a Frame két széle között.

⁸¹ `webbrowser` – Displays web pages, URL: <https://pymotw.com/2/webbrowser/>, Megtekintés: 2021.03.14.

```

def __Animation(self):
    if self.__direction==False:
        if self.__theX2==0:
            self.__direction=True
            self.__theX=295
        else:
            self.__theX2-=2
            self.__theX+=2.6

    else:
        if self.__theX2==230:
            self.__direction=False
            self.__theX=0
        else:
            self.__theX2+=2
            self.__theX-=2.6

    self.__placeText()
    self.__modifyPlayField()
    self.__AboutM.after(20, self.__Animation)

```

104. ábra: Animációs loop

Az ablaknak fontos eleme még egy apró játék, egy Pong alkalmazás, mely az 1972-es játék AY-3-8500 változatának "tenisz" játékát hivatott életre kelteni⁸². Irányíthatjuk a játékot a billentyűzet fel és lefele kurzornyilaival, de az egér görgőjével is, mely közelebb áll az akkor jellemző analóg potméterek (paddle) használatához annak ellenére, hogy működése szerint egy digitális spinner⁸³. Ezeken felül az s/S és r/R gombok is bindingelésre kerültek, előbbivel a hang kapcsolható be/ki, utóbbival a játék indítható útra. A hang állapota módosítható a self.__soundButton objektumra való kattintással is. Alapértelmezetben a hang kikapcsolt állapotban van!

```

def __soundChange(self):
    if self.__sound==False:
        self.__sound=True
        self.__soundButton.config(image=self.__imageOn)
        self.__playsound("p/p3.wav")
    else:
        self.__sound=False
        self.__soundButton.config(image=self.__imageOff)

```

105. ábra: Hanghatásokat be/ki kapcsoló függvény

⁸² AY-3-8500, URL: <https://de.zxc.wiki/wiki/AY-3-8500>, Megtekintés: 2021.03.14.

⁸³ Paddle (Game Controller), URL: [https://en.wikipedia.org/wiki/Paddle_\(game_controller\)](https://en.wikipedia.org/wiki/Paddle_(game_controller)), Megjelenés: 2021.03.14.

```

def __wheel(self, event):
    if (event.delta > 0 or event.num==4):
        self.__upPressed("123")
    elif (event.delta < 0 or event.num==5):
        self.__downPressed("123")

def __upPressed(self, event):
    if self.__bat1XY[1]>5:
        self.__bat1XY[1]-=10

def __downPressed(self, event):
    if self.__bat1XY[1]<205:
        self.__bat1XY[1]+=10

```

106. ábra: Ütő mozgatása

A játék a hagyományos pong szabályok szerint 15-ig számítja a pontokat (4 bites számláló volt ezekben a gépekben), ezt követően az ütközések nem adnak többé hangot.

```

def __resetThings(self, event):
    self.__bat1XY = [30, 105]
    self.__bat2XY = [350, 105]
    self.__points = [0, 0]
    self.__resetBall()
    self.__collisionDelay=0
    self.__CPUCounter = 0
    self.__CPUMoveUnit=0

def __resetBall(self):
    import random
    import datetime

    self.__ballXY = [190, 120]
    if self.__sound==True:
        self.__playsound("p/p1.wav")
    random.seed(int(str(datetime.datetime.now()).split(".")[1]))
    self.__ballDir=random.randint(0,7)
    self.__ballSpeed = 1

```

107. ábra: A játékot befolyásoló változók

A játék során egy két int-et tartalmazó lista tartalmazza a pontszámokat, a labda irányát egy 0-7 közötti számot tartalmazó változó, a self.__ballDir határozza meg, minden elem X,Y pozícióját egy két int-et tartalmazó lista kezeli. A labda kezdői sebessége 1.

A játék maga egy Canvas felületen⁸⁴ zajlik, mely minden képfrissítéskor törlődik a self.__gameField.delete("all") parancs segítségével, majd a self.__plaffieldHalf eljárás legenerálja a térfelét középen elválasztó szaggatott vonalakat, majd a self.__createElements legenerálja az ütőket és a labdát a megfelelő helyekre, valamint a pontszámokat.

⁸⁴ Canvas Widgets, URL: https://www.python-course.eu/tkinter_canvas.php, Megtekintés: 2021.03.14.

```

    def __createElements(self):
        self.__gameField.create_rectangle(self.__bat1XY[0], self.__bat1XY[1], self.__bat
1XY[0]+10, self.__bat1XY[1]+40, fill="white")
        self.__gameField.create_rectangle(self.__bat2XY[0], self.__bat2XY[1], self.__bat
2XY[0]+10, self.__bat2XY[1]+40, fill="white")
        self.__gameField.create_rectangle(self.__ballXY[0], self.__ballXY[1], self.__bal
lXY[0]+10, self.__ballXY[1]+10, fill="white")

        scoreFont=[self.__hammerFont[0], 40, "bold"]

        self.__gameField.create_text(155,30,fill="white",font=scoreFont,
                                    text="{:2}".format(self.__points[0]))

        self.__gameField.create_text(235,30,fill="white",font=scoreFont,
                                    text="{:2}".format(self.__points[1]))

```

109. ábra: Pályaelemek megrajzolása

Az ütközésekben, valamint a gép reakciójánál egy késleltető változó van használban annak érdekében, hogy a labda ne generáljon "beragadásos" ütközést, illetve, a CPU se legyen verhetetlen azáltal, hogy folyamatosan korrigálja helyzetét a labda Y pozíciójának ismeretében.

A labda ütközése esetén, amennyiben az ütközés egy ütővel történik, az ütő középpontjától mért pozíciói alapján történik a lepattanás irányának meghatározása, a falakról való lapattanás esetén viszont, amennyiben merőlegesen érezik a labda, azonos eséllyel pattan 45 fokban valamely irányba és 5% eséllyel pattan vissza merőlegesen, minden más esetben többnyire a függőleges irányát megváltoztava, az eredeti horizontális irányba halad tovább, minimális eséllyel merőlegesen pattan fel.

```

if self.__ballDir==1:
    num2 = random.randint(-3, 3)

    if num== -1:
        self.__ballDir=4
    else:
        self.__ballDir=3
elif self.__ballDir==7:
    num2 = random.randint(-3, 3)

    if num== -1:
        self.__ballDir=4
    else:
        self.__ballDir=5
elif self.__ballDir==0:
    num2 = random.randint(-10, 10)
    if num2<0:
        self.__ballDir=3
    elif num2>0:
        self.__ballDir=5
    else:
        self.__ballDir=4

```

110. ábra: Részlet, példa a labda irányváltoztatására

Amennyiben a self.__sound értéke True, a labda ütközésekor sajátos pong hangeffekt hallható, melynek egy változata szólal meg, ha a labda elhagyja a teret és valamely játékos pontot kap. A hangokat a pygame mixer modulja⁸⁵ adja, mivel Linux alatt az alapértelmezett playsound nem működött, ugyanis egyes disztribúciókban nem találhatóak meg azok a dependenciák, amelyekre épül⁸⁶, a véletlenszám generáláshoz alapértelmezetten a datetime nyújt segítséget, hiba esetén a pygame mouse modulja segítségével az egérkurzor pozíciója nyújtja a seed-et.

```
try:  
    random.seed(int(str(datetime.datetime.now()).split(".")[1]))  
except:  
    os.environ['PYGAME_HIDE_SUPPORT_PROMPT'] = "hide"  
    import pygame.mouse as M  
    random.seed(M.get_pos()[0]+M.get_pos()[1])
```

111. ábra: Véletlenszám generálás a játék során

III/2/5. ListBox elemek

A korábban már említett, jobb oldalon található listadobozok megkönnyítik a fejlesztő munkáját. A legutóbbi megtekintett fájlokat tartalmazó lista a self.recentFiles lista elemeinek a fájlnév szegmensét jeleníti meg, így a ListBox-ban és a valódi listában pozíciók között kapcsolat van, még azonos fájlnevek esetében is működőképes. Amennyiben a kijelölés megváltozik, statLabel formájában megjelenik a fő ablak alján a teljes elérési link, a könnyebb beazonosítás érdekében. A gombra kattintva a fájl betöltésre kerül.

A szintaxislista a kezdő programozó mankójául szolgál, a self__Syntax kulcsait és a hozzájuk tartozó értékeket is tartalmazza, attól függően színezve őket, hogy parancsként vagy argumentumként érvényesülnek a kódban. A gombra kattintva a CodeBox kurzorának helyére kerül a kijelölt parancs.

Az alkalmazás rendelkezik gyorsbillentyűkkel is, a ctrl+görgővel történő betűméret módosításon felül, ezeket foglalja össze az alábbi táblázat:

⁸⁵ Pygame Mixer, URL: <https://www.pygame.org/docs/ref/mixer.html>, Megtekintve: 2021.03.14.

⁸⁶ TaylorSMarks/playsound, URL: <https://github.com/TaylorSMarks/playsound/issues/27>, Megtekintve: 2021.03.14.

| Billentyű | Esemény |
|-----------|---|
| F1 | Új fájl megnyitása (self.__doNew eljárás meghívása) |
| F2 | Fájl megnyitása (self.__doOpen meghívása) |
| F3 | Fájl mentése (self.__doSave meghívása) |
| F4 | Kód fordítása (self.__getCodeOnly meghívása) |
| F5 | Kép beszúrása (self.__loadImagePath meghívása) |
| F6 | Sötét/Világos mód közötti váltás (self.__lightDark meghívása) |
| F7 | Visszavonás művelet (self.__doUndo meghívása) |
| F8/End | Deliminátor beszúrása (self.__insertDeliminator) |
| F9 | OptionsMenu megnyitása (self.__OptionsMenu meghívása) |
| F10 | Alapértelmezett böngészőben megnyitása (self.__doViewDefault meghívása) |

112. ábra: Bindingelt gyorsbillentyűk

A self.__lightDark függvény a ellentétes értékűre váltja a self.__Config állományban a "DarkBox" értékét, viszont ezt a külső állományba nem menti. Végül meghívja az update eljárást

```
def __lightDark(self):
    if self.__Config.get_Element("DarkBox") == "True":
        self.__Config.set_Element("DarkBox", "False")
    else:
        self.__Config.set_Element("DarkBox", "True")
    self.updateCodeBox()
```

113. ábra: Váltás a színsémák között

A self.__insertDeliminator pedig, megszámolja a sorokat (a kommentek eltüntetése után a szöveget felosztja a stringek szerint) és a sor végére elhelyezi zárójelben, kommentként a sor számát, így amennyiben hibaüzenet érkezik, lényegesen könnyebb lesz a hely meghatározása.

```
def __insertDeliminator(self):
    try:
        self.__CodeBox.insert(INSERT, self.__deliminator)
    except:
        self.__CodeBox.insert(INSERT, "%%")
        self.__deliminator = "%"

    import re
    lines=re.sub(rf"{self.__deliminator}.*\n", self.__deliminator, self.__CodeBox.get(0, INSERT))
    lines=lines.split(self.__deliminator)

    self.__CodeBox.insert(INSERT, " ("+str(len(lines)-1)+")")
    self.__CodeBox.insert(INSERT, "\n")
```

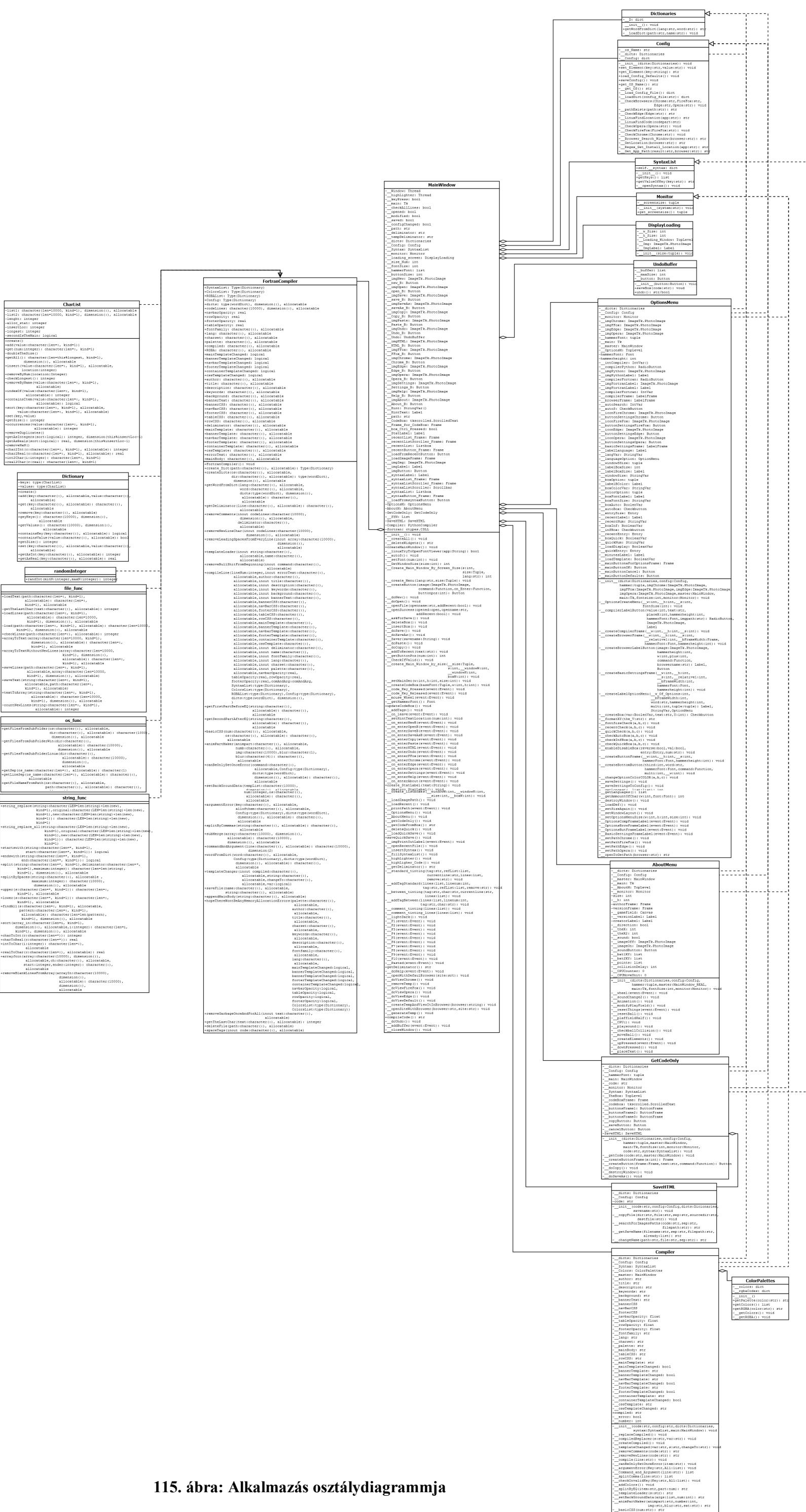
114. ábra: Deliminator beszúrása

III/3. Program lezárása

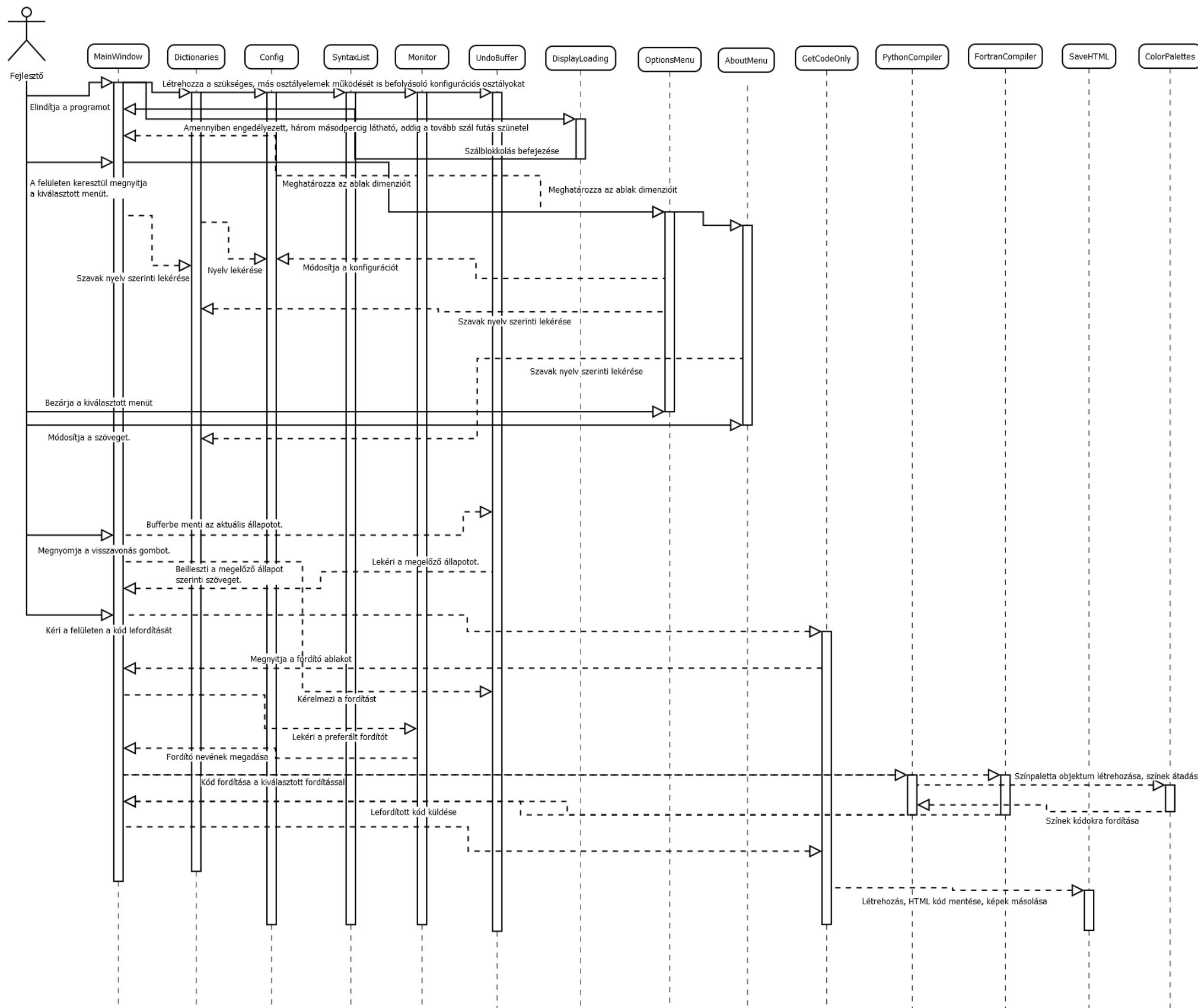
Amennyiben a felhasználó bezárja a programot (vagyis annak futása nem valamely hiba folytán szakad meg), amennyiben a self.__modified értéke True, a program rákérdez a mentésre. Ezt követően a self.__main-t megsemmisíti, majd törli, és meghívja a biztonság

kedvéért a garbage collector-t is (maga a garbage collector, mint minden modern programnyelv esetében, automatikusan szabadítja fel a helyet⁸⁷). Mivel a szintaxiskiemelő szál daemon-ként futott, nem-daemon szál hiányában az ő futása is megszűnik. A program elindulásához hasonlóan működése felhasználói beavatkozást nem igényel, lefutása szekvenciális jellegű.

⁸⁷ AL SWEIGART: Beyond the Basic Stuff with Python, No Starch Press [USA], 2021., 230. o.



115. ábra: Alkalmazás osztálydiagrammja



116. ábra: Alkalmazás szekvenciadiagramma

III/4. Tesztelés

Az alkalmazás tesztelése weblapok elkészítésével történt, mivel komplex, ablakos alkalmazásról beszélünk, a helyes futattás nem tesztelhető csupán a részelemek futtatásával, bár a Python erre tartalmaz beépített függvénykönyvtárat (pytest⁸⁸). A rendszerszintű tesztelés mellett a compilerek, mint komponensek külön-külön is tesztelésre kerültek. A súgóként funkcionáló Help.html eleve Boo-T nyelven íródott, megmutatva a fejlesztő környezet lehetőségeit. Három weblap készült el a program tesztelése során.

III/4/1. Altair 8800

```
basics@author=Member - Fehér János Zoltán,  
language=en,  
charset=UTF-8,  
palette=7) %%  
  
keywords(Altair 8800,70s computing, vintage, computer science) %%  
description(Just a simple site dedicated to the Altair 8800 computer) %%  
title(Altair 8800) %%  
background(gradient=HOR) %%  
font-family("Gulim") %%  
  
banner(animation(40s, src/01.jpg,  
                  src/02.jpg,  
                  src/03.jpg,  
                  src/04.jpg,  
                  src/05.jpg),  
        text("Altair 8800", 8em, center),  
        height=300) %%  
  
navbar(brand(image=src/mits.svg), sticky,  
       item("The Hardware", hw),  
       item("Programing", prog),  
       item("Gallery", pic),  
       item("Price", price)  
     ) %%  
  
bootrow(id=hw, rate(7,5), article(  
  title="Hardware", title-align=left, rawtext=  
  "The Altair 8800 from Micro Instrumentation Telemetry Systems (MITS) of  
  Albuquerque, NM, is considered by many to be the first personal computer - a  
  computer that is easily affordable and obtainable. The entire Altair 8800 system is  
  comprised of a metal case, a power supply, a front panel with switches, and a
```

⁸⁸ MICHAL GORELICK, IAN OZSVÁRD: High Performance Python 2nd edition, O'Reilly Media, Inc, USA [2020] 57. o.

passive motherboard with expansion slots. All of the circuitry - the CPU and memory, are on cards which plug into the expansion slots, which MITS called the Altair Bus."),

image=src/07.jpg) %%

bootrow(id=prog, rate(7,5), article(

title="Programming", title-align=left, rawtext=

"The ALTAIR 8800 has 78 basic machine language instructions. Since many of the instructions can be modified to affect different registers or register pairs, more than 200 variances of the basic instructions are possible. Each instruction is presented as a standardized mnemonic or machine language code. Instructions may occupy from one to three sequential (serial) bytes, and the appropriate bit patterns are included. A condensed summary of the complete instruction set showing the mnemonics and instructions in both binary and octal is included as an Appendix."),

image=src/09.png) %%

bootrow(id=pic, rate(auto),

image=Src/10.jpg,

image=Src/11.jpg,

image=Src/12.jpg) %%

bootrow(rate(auto),

image=Src/13.jpg,

image=Src/14.jpg,

image=Src/16.jpg) %%

table(id=price, columns("Function", "Kit", "Assembled"), inverted,

row("1K static RAM", "\$97", "\$139"),

row("2K static RAM", "\$145", "\$195"),

row("4K static RAM", "\$195", "\$275"),

row("Serial Interface", "\$119", "\$139"),

row("Paralell Interface", "\$92", "\$114"),

row("Casette Interface", "\$128", "\$175")

) %%

footer(button="Keyboard is for dummies!",

facebook=https://www.facebook.com/groups/265721677918/,

youtube=https://www.youtube.com/channel/UCly5UrhN90ULLSGnZjzR7Fw,

skype=#,

phone=#,

email=grant@stockly.com) %%



The Hardware Programming Gallery Page

Hardware

The Altair 8800 from Micro Instrumentation Telemetry Systems (MITS) of Albuquerque, NM, is considered by many to be the first personal computer – a computer that is easily affordable and obtainable. The entire Altair 8800 system is comprised of a metal case, a power supply, a front panel with switches, and a passive motherboard with expansion slots. All of the circuitry – the CPU and memory, are on cards which plug into the expansion slots, which MITS called the Altair Bus.

Programming

The ALTAIR 8800 has 78 basic machine language instructions. Since many of the instructions can be modified to affect different registers or register pairs, more than 200 variances of the basic instructions are possible. Each instruction is presented as a standardized mnemonic or machine language code. Instructions may occupy from one to three sequential (serial) bytes, and the appropriate bit patterns are included. A condensed summary of the complete instruction set showing the mnemonics and

117. ábra: Altair 8800 desktop nézetben

Programming

Keyboard is for dummies!

118. ábra: Altair 8800 mobil nézetben

III/4/2. Sivatagi Róka

```
deliminator β
title(A sivatagi roka) β
basics(author=Fehér János Zoltán, language=hu, palette=sand) β
keywords(roka,emlos,sivatag,termeszet) β
description(Csak egy cuki állatká!) β
font-family("Gabriola","Times New Roman") β
opacity(navbar=0.70,container=0.65,footer=0.70) β

background(image=src/RokaPlus/Roka02.jpg) β

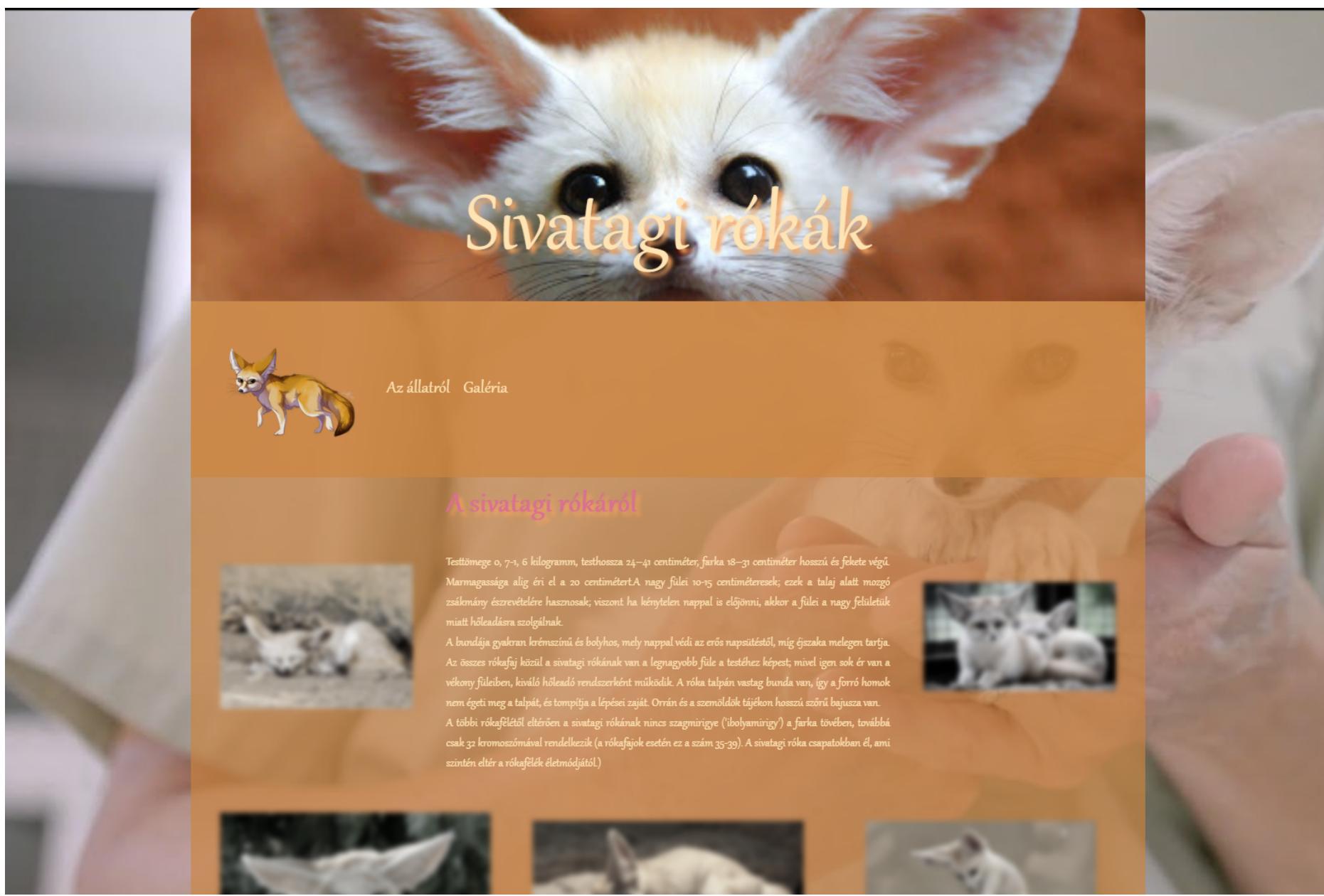
banner(size=cover, text("Sivatagi rókák",7em,center), animation(40s,
    src/Roka04.jpg,
    src/RokaPlus/Roka01.jpg,
    src/Roka01.jpg
), height=350) β

navbar(brand(image=src/desertfox.png), sticky,
    item("Az állatról",1),
    item("Galéria",2)) β

bootrow(id=1,rate(3,6,3), imgfilter,
    image=Src/Roka02.jpg,
    article(title="A sivatagi rókáról", title-align=left,
        rawtext="Testtömege 0,7-1,6 kilogramm, testhossza 24–41 centiméter, farka  

18–31 centiméter hosszú és fekete végű. Marmagassága alig éri el a 20 centimétert. A nagy fülei 10-15 centiméteresek; ezek a talaj alatt mozgó zsákmány észrevételére hasznosak; viszont ha kénytelen nappal is előjönni, akkor a fülei a nagy felületük miatt hőleadásra szolgálnak.<br>A bundája gyakran krémszínű és bolyhos, mely nappal védi az erős napsütéstől, míg éjszaka melegen tartja. Az összes rókafaj közül a sivatagi rókának van a legnagyobb füle a testéhez képest; mivel igen sok ér van a vékony füleiben, kiváló hőleadó rendszerként működik. A róka talpán vastag bunda van, így a forró homok nem égeti meg a talpát, és tömpítja a lépései zaját. Orrán és a szemöldök tájékon hosszú szörű bajusza van.<br>A többi rókafelétől eltérően a sivatagi rókának nincs szagmirigy ('ibolyamirigy') a farka tövében, továbbá csak 32 kromoszómával rendelkezik (a rókafajok esetén ez a szám 35-39). A sivatagi róka csapatokban él, ami szintén eltér a rókafélék életmódjától.)"),
    image=Src/Roka03.jpg) β

bootrow(id=2, rate(auto), imgfilter,
    image=src/Roka05.jpg,
    image=src/Roka07.jpg,
    image=src/Roka09.jpg) β
footer(button="Felülré!") β
```



119. ábra: Sivatagi Róka desktop nézetben



120. ábra: Sivatagi Róka mobil nézetben

III/4/3. Help

delimiter *hullahopp*

basics(author=Fehér János Zoltán (Member),

language=en,

charset=utf-8,

palette=27) *hullahopp*

keywords(Help, Boo-T, Bootstrap) *hullahopp*

description(This is the help file for the Boo-T application) *hullahopp*

title(Help) *hullahopp*

background(image=src/Back.jpg) *hullahopp*

font-family("Myriad Pro Light", "Informal Roman", "Gabriola", "Arial") *hullahopp*

opacity(navbar=0.55, container=0.75, table=0.75, footer=0.55) *hullahopp*

banner(size=cover, height=450, animation(50s,

src/Spooky1.jpg,

src/Spooky2.jpg,

src/Spooky3.jpg,

src/Spooky4.jpg),

text("Help with Boo-Ting!!", 4em, center)

) *hullahopp*

navbar(brand(image=src/ghost.png), expand=xl,

item("Welcome", welcome),

item("Basics", basics),

item("Page Settings", page),

item("Banner", banner),

item("NavBar", nav),

item("BootRow", bootr),

item("Tables", tables),

item("Footer", footer)

) *hullahopp*

bootrow(id=welcome, rate(auto), article(title="Greetings!", title-align=left,

rawtext="You just opened the help file for Boo-T and as you can see, it was created in Boo-T as well. Well, well, let's get started!
The software was designed to be as simple as it can be for a beginner, so you won't gave up too early! Let me explain the most basic things!
In the main windows, you can see the Syntax List, these are the only things you need to work with!

The software generates an HTML file and injects the CSS into it, the only thing you will find in your folder is the <i>'img'</i> that contains the images you used. Because I don't want you to lose your images, Boo-T will copy them to that folder and changes the path in the HTML file if you are using an absolute path like <i>'C:/Documents/powerpuffgirls.png'</i>.
")

) *hullahopp*

```

bootrow(id=basics, rate(auto),
    article(title="I. Basics",
        title-align=left,
        rawtext="In the syntax of Boo-T, there are <b>commands</b> and
<b>arguments</b>. Commands are the most important, they are at the beginning of
lines, they are followed by rounded brackets. The line is always ended by a
delimiter, which also separates code and comments, you can write anything after
the delimiter till the next newline. So the syntax basically looks like
this:<br><br><b><u>command</u></b>(<i>arguments</i>)
comment<br><br>Setting the <b>delimiter</b> is really special compared to the
other commands. You can set it only in the very first line of your code (other
ones will be ignored), writing '<b><u>delimiter</u></b>' without the rounded
brackets you would usually use with commands and putting the desired delimiter
<b>after a single space.</b><br>Most commands are only needed to be used once
(except for bootrows and tables), the newer ones are just overriding the older
ones.<br><b>Arguments</b> are separated by commas inside the rounded brackets.
There are three kind of appearances of arguments:<br><ul><li>Just a single
word</li><li>Two words splitted by an equality symbol, like
<i>arg=something</i></li><li>A word with rounded brackets, like commands,
containing several words separated by commas, like
<i>arg(something,something,something)</i></li></ul>Because of this wrapping you can
place spaces and newlines nearly anywhere, so your code can be really easy to
read, like I did with the banner for this site:<br><br><u>banner</u>(size=cover,
height=450,
```

```

    animation(50s,<br><space:10>Src/Spooky1.jpg,<br><space:10>Src/Spooky2.jpg,<br>
    ><space:10>Src/Spooky3.jpg,<br><space:10>Src/Spooky4.jpg),<br><space:10>text('Help
with Boo-Ting!', 4em, center)<br>) %%<br>Now you know the most basic
things, we are gonna go for the 'page settings'!"))
    hullahopp
```

```

bootrow(id=page, rate(auto),
    article(title="II. Page Settings",
        title-align=left,
        rawtext="Creating the HTML template for your website can be really boring,
so I'm taking care of that for you! These are the things you can set in Boo-
T:<ul><li><u>basics</u>(author=,language=,charset=,palette=)<br>Author, language and
charset basically just <b>sets the meta tags</b>, if you ignore or forget them,
language will be set to <i>english</i> and charset to <i>UTF-8</i> (anything else
for charset is <u>not recommended</u>!) Palette has a basic set of palettes,
containing 4 colors for building up the entire site. You can call them by name
(you can find them in folder <i>'default/Colors'</i>), use a number between 0-27 or
set it to random that will save your site with a random palette each time you
save you generate your code.</li><li><u>keywords</u>()<br>You can put there
keywords, separated by commas, it will be directly copied to the meta tag on
your site.</li><li><u>description</u>()<br>Same as the keywords, it will be copied
directly.</li><li><u>title</u>()<br>And this one as
well!</li><li><u>background</u>({color, gradient=, image=})<br>Setting the background
```

is really simplified. You can set it to 'color' that will create a solid color. Gradient will produce a color gradient based on the color palette, you can set it to HOR, VER or DIG, representing the direction. If you set it to 'image', the background will be filled with a fixed image.<u>font-family</u>()
This will set the fonts your website gonna use. If you don't set it, Arial will be chosen.<u>opacity</u>({container=, tables=, navbar=, footer=})
Sets opacity for the given elements, default is 1.0. You have to use float numbers between 0.0 and 1.0! "

)> **hullahopp**

```
bootrow(id=banner, rate(auto),
        article(title="III. Banner",
               title-align=left,
               rawtext="Banner is a really important part for a website. On small devices, like phones, the banner won't be displayed along with the part outside the container. Inside the <b>banner()</b> command, there are a lot to set. <b>Size</b> can be <i>'contain'</i> or <i>'cover'</i>, left out will display a cover-style banner. <b>Height</b> is the vertical size of your banner in pixels (you <u>don't have</u> to put there 'px'!). The title of your site can be set via the <b>text()</b> argument, it contains three elements, you <u>cannot</u> change their orders!<ol><li><b>"The Title Text as a String"</b></li><li><b>The size of the title text</b>, can be px or em.</li><li><b>The alignment of the text</b>, can be left, center or right</li></ol>There are two ways to set the background picture of the banner:<ul><li><b>image=</b><br>Works as the image argument for the background, it will cover the division.</li><li><b>animation()</b></li>Contains the time and the images, separated by commas. <u>First</u> you have set the time for the whole animation circle, then add the link for the pictures, one by one, creating a list. It will have a blurry, grayscale animation for transition.</li></ul>For the banner example, you can see one at the Basics section!"
```

)> **hullahopp**

```
bootrow(id=nav, rate(auto),
        article(title="IV. NavBar",
               title-align=left,
               rawtext="Navbars are the only navigation tool and you can only add link to ids available on the current site, because on mobile phone, it's a lot easier to use a onepage site with vertical scrolling. Building Navbars is pretty easy, you set it with the <b>navbar()</b> command. There are two kind of elements you can add:<ul><li><b>brand({String", image=}</b><br>Brand is the very item of the navbar which is not affected by the toggle button. The brand can be a <b>"String"</b>, a plain text or you can add an image tag and put there an image.</li><li><b>item("String", id)<br>Items are the links for the parts of your website. They can be only string items and you have to put there an id that was added to a bootrow, table or footer item.</li><li><b>sticky</b><br>If you set it, the NavBar will stay at the top as you progress scrolling down.</li><li><b>expand=</b><br>You can add here the screen size where the navbar should be expanded. You can use sm, md, lg, xl or a number between 0-
```

```
4.</li></ul> An example for the NavBar  
code:<br><u>navbar</u>(brand(image=ghost.png),sticky,expand=xl<br><space:20>item("Item1", item1),<br><space:20>item("Item2", item2),<br><space:20>item("Item3", item3)) %%"  
)) hullahopp
```

```
bootrow(id=bootr, rate(auto),  
    article(title="V. BootRow",  
        title-align=left,  
        rawtext="BootRows are maybe the most important and complex components  
of your website, you can have more than one, they use the column alignment of  
Bootstrap. You insert them with the <u>bootrow()</u> command. There are several  
elements you can add to bootrows:<ul><li><b>id=</b><br>With the id, you can add  
your row to one of the navbar links. You just have to type it, no string format  
needed.</li><li><b>rate</b>()<br>Rates can have two kind of values, auto and a list  
of numbers, separated by commas. If you set it to auto, all the items you will  
have to fit in the row with the same width. This is great for an image gallery,  
take this example:</li></ul>"  
)) hullahopp
```

```
bootrow(rate(auto),  
    image=src/Zerg1.jpg,  
    image=src/Zerg2.jpg,  
    image=src/Zerg3.jpg,  
    image=src/Zerg4.jpg,  
    image=src/Zerg5.jpg,  
    image=src/Zerg6.jpg) hullahopp
```

```
bootrow(rate(auto),  
    article(rawtext="Of course, you can wrap it into two rows, for your taste.")  
)) hullahopp
```

```
bootrow(rate(auto),  
    image=src/Zerg1.jpg,  
    image=src/Zerg2.jpg,  
    image=src/Zerg3.jpg  
) hullahopp
```

```
bootrow(rate(auto),  
    image=src/Zerg4.jpg,  
    image=src/Zerg5.jpg,  
    image=src/Zerg6.jpg) hullahopp
```

```
bootrow(rate(3,6,3),  
    image=src/Zerg5.jpg,  
    article(rawtext="But you can do it like this, where the column in the middle  
takes up twice the place."),  
)) hullahopp
```

`image=src/Zerg6.jpg) hullahopp`

```
bootrow(rate(auto),
        article(rawtext="<ul><li><b>article(title="" , title-align=center,
rawtext="")</b></li><br>Articles are walls of text, the object you are reading right
now is an arcticle as well. Articles have three part:<ul><li>title="String"</li>The text
for the title, given as a string.<li><b>title-align</b>={left, center, right}<br>This will
set the alignment of the title.</li><li><b>rawtext=""</b><br>This is the most advanced
argument in the whole environment, this is the mathod for embending test. You
can use here any html tags to make your text more pleasant for the eyes, here
you can find some examples, I added some spaces so they won't be
compiled:</li>< b > is for <b>bold</b> text, < i > is for <i>italic</i>, < u > stand
for <u>underlined</u>. <br>You can add new lines by inserting a < br >, since
normal enters won't work. You can create list by < ul > < ol > and itt items with
< li >. I created a new tag that is not used by html standards: < space:X > will
add X spaces to you text, < space:10 > works <space:10> like
this.</ul><li><b>image</b>=<br>Works like as mentioned
before.</li><li><b>imgfilter</b><br>This argument changes the image behavior by
making it blurry and grayscale unit the mouse goes by.</li></ul>")) hullahopp
```

```
bootrow(rate(auto), image=src/Zerg5.jpg,
        article(title="centered title", title-align=center,
        rawtext="This is a normal image, i have to write here some uninteresting
thing to make it work.")) hullahopp
```

```
bootrow(rate(auto), image=src/Zerg5.jpg, imgfilter,
        article(title="right-aligned title", title-align=right,
        rawext="This one is filtered! Just showing you the basic syntax for a
bootrow:<br><u>bootrow</u>(id=Two, rate(3,6,3),
imgfilter,<br><space:20>image=05.jpg,<br><space:20> article(title="This is the Title!",
title-align=center,<br><spcae:20>rawtext="And this is some really awesome
text!" ),<br><space:20>image=06.jpg<br><space:20>) %%")) hullahopp
```

```
bootrow(id=tables, rate(auto),
        article(title="VI. Tables", title-align=left,
        rawtext="Tables are a little oldschool, but can give your visitors many
information collected into a small place. They are invoked with the <u>table</u>()
command, and there are additional fields to fill them with
content:<ul><li><b>id=</b><br>Same as the
bootrow.</li><li><b>columns("string","string","string")</b><br>You have to add the
columns names as a list of strings.</li><li><b>row("string","string","string")</b><br>You
add rows with a single subcommand that contains a list of
strings</li><li><b>inverted</b><br>Will let you change the colors of the table, it may
be easier to read that way!</li></ul>So the syntax of the example tables looks like
this:<br><u>table</u>(id=Protoss,
columns("Unit","HP/Shield","Mana","Mineral","Gas"),<br><space:20>row("Zealot","100/60
","0","100","0"),<br><space:20>row("Dragoon","100/80","0","125","50"),<br><space:20>r
```

```
ow("High Templar", "40/40", "250", "50", "150"),  

    <br><space:20>row("Dark  

Templar", "80/40", "0", "125", "100")<br>The second one has the "inverted" argument  

added!")) hullahopp
```

```
table(id=Protoss1, columns("Unit", "HP/Shield", "Mana", "Mineral", "Gas"),  

    row("Zealot", "100/60", "0", "100", "0"),  

    row("Dragoon", "100/80", "0", "125", "50"),  

    row("High Templar", "40/40", "250", "50", "150"),  

    row("Dark Templar", "80/40", "0", "125", "100") hullahopp
```

```
table(id=Protoss2, inverted, columns("Unit", "HP/Shield", "Mana", "Mineral", "Gas"),  

    row("Zealot", "100/60", "0", "100", "0"),  

    row("Dragoon", "100/80", "0", "125", "50"),  

    row("High Templar", "40/40", "250", "50", "150"),  

    row("Dark Templar", "80/40", "0", "125", "100") hullahopp
```

) *hullahopp*

```
bootrow(id=footer, rate(auto),  

    article(title="VII. Footer", title-align=left,  

        rawtext="Footers are showing the user where they can contact you, so they  

are really important to make. Footers in Boo-T are prebaked, so you don't have  

to worry about it. Footers are created with the <u>Footer()</u> command and  

basically, you have to enter two kind of arguments. The button will scroll back to  

the top and on smaller screens, it will be attached to the bottom. Also, the Footer  

contains a lot of buttons to social media.<ul><li><b>button="String"</b><br>You can  

add the text to the button as a string.<li><b>{media}</b>=</li><br>You can add a  

social media icon and your desired link. Media can be email, phone, skype,  

facebook, youtube, instagram, vkontakte, googleplus, linkedin, twitter or  

github.</li>The footer of this site looks like this:<br>footer(button="Scroll back to  

top!",<br><space:20>facebook=https://www.facebook.com/jnszlttn.fhr/,<br><space:20>emai  

l=feher.janos.zoltan@gmail.com,<br><space:20>phone=+36705977991,<br><space:20>gi  

thub=https://github.com/MemberA2600,<br><space:20>youtube=https://www.youtube.co  

m/user/M3MB3Rrr,<br><space:20>linkedin=https://www.linkedin.com/in/j%C3%A1nos-  

zolt%C3%A1n-feh%C3%A9r-4378828b/) "))) hullahopp
```

```
footer(button="Scroll back to top!",  

    facebook=https://www.facebook.com/jnszlttn.fhr/,  

    email=feher.janos.zoltan@gmail.com,  

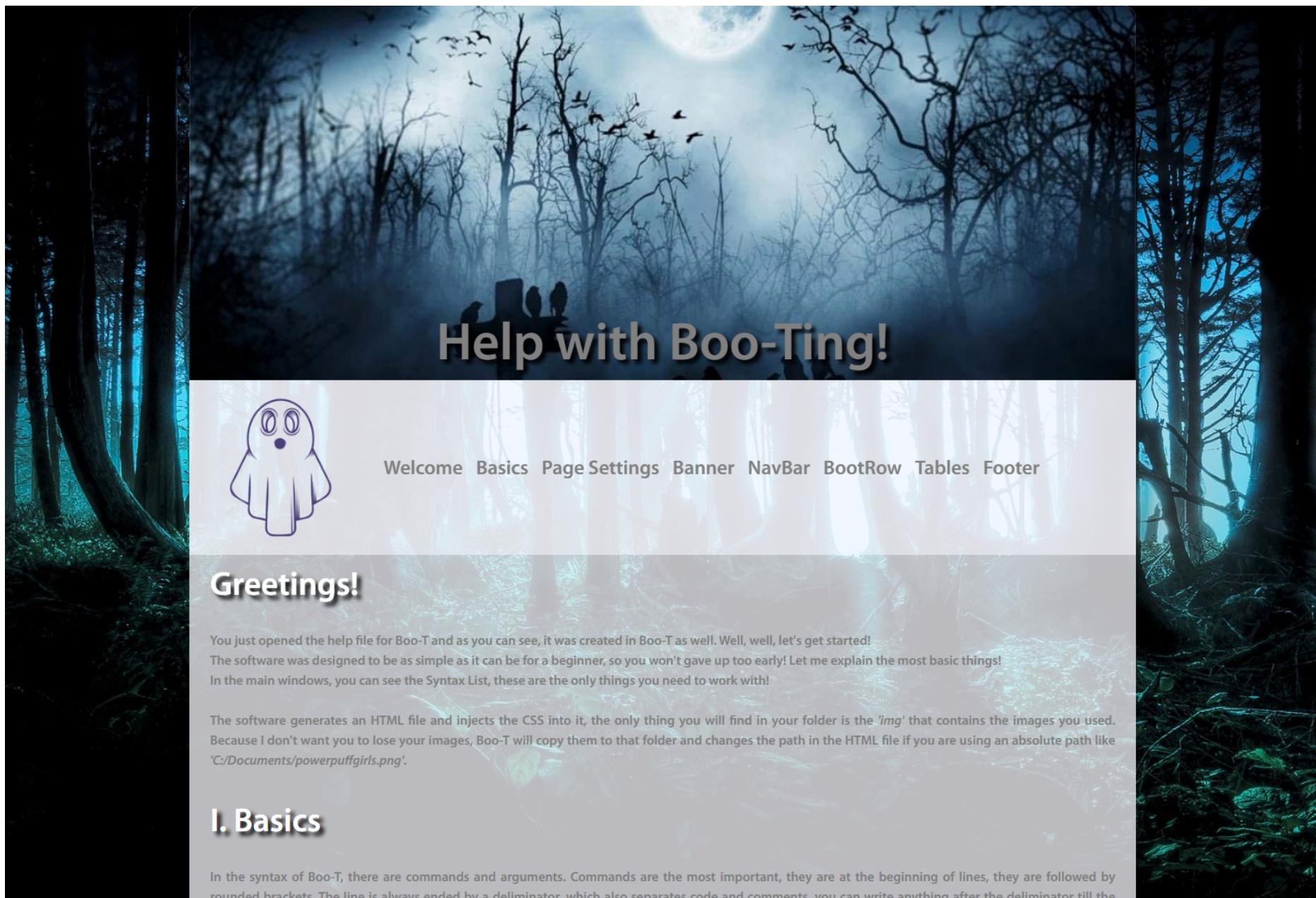
    phone=+36705977991,  

    github=https://github.com/MemberA2600,  

    youtube=https://www.youtube.com/user/M3MB3Rrr,  

    linkedin=https://www.linkedin.com/in/j%C3%A1nos-zolt%C3%A1n-feh%C3%A9r-  

4378828b/) hullahopp
```



121. ábra: Help desktop nézetben

A screenshot of a mobile application window titled "Greetings!". It features a white ghost icon at the top. The main text area contains introductory text. On the right side, there is a sidebar with a menu icon and a section titled "I. Basics" with descriptive text. A scroll bar is visible on the right edge.

122. ábra: Help mobil nézetben

III/5. Fejlesztési lehetőségek

A program a jelenlegi formájában is alkalmas az egyszerűbb weblapok elkészítésére, de egy program, különösen egy fejlesztőkörnyezet soha nem tekinthető teljesnek és minden lefedőnek.

- A Fortran 90 alapú fordító Linux alatti hibaelhárítása mindenféleképpen elvégzendő feladat, mivel egy alkalmáznak illendő minden kompatibilis operációs rendszeren azonos szintű szolgáltatást nyújtania (platformfüggetlenség elve⁸⁹).
- A tkinter és a Python kompatibilis a macOS operációs rendszerrel, ezért a Boo-T macOS változatát is el lehet készíteni.
- A mobiltelefonon való helyes megjelenítés érdekében olyan tesztfelület implementálása vagy integrálása, mely a mobiltelefonok környezetét emulálva képes helyesen megjeleníteni az általunk tesztelt weblapot⁹⁰.
- A nyelv további elemekkel való bővítése, elsősorban a bootrow rendelkezik kihasználatlan potenciállal, de a kontérner általánosságban is kaphat új beépülő elemeket.
- Optimalizálás, további szálakra bontás.
- Kódban szereplő elemek, különös tekintettel a MainWindow és OptionsMenu objektumokra, további alobjektumokra való bontása
- Ésszerűbb, grid rendszerű elrendezés az ablakos felületen, ezáltal a fix ablakméretek helyett dinamikus ablakméretezés lehetősége⁹¹.

IV. Felhasználói dokumentáció

IV/1. A programnyelv szintaxisa

Mivel egy önálló programnyelvről beszélünk, a legfontosabb megismerni magát a nyelvet és a környezetet, amit használunk. A  ikonra kattintva az oldal felületén minden információt megkaphatunk a Boo-T működéséről, az oldal mellett megtalálható a forráskódja is, hiszen a súgó fájl külső program bevonása nélkül, ezzel az alkalmazással készült.

A programnyelv deliminátoros, vagyis a sorok végét nem a newline karakter, hanem a delimiter határozza meg, ez mögött, mint az assembly programnyelvekben, a

⁸⁹ Platformfüggetlenség, URL: <https://www.huwiki.org/wiki/Platform-f%C3%BCggetlens%C3%A9g>, Megtekintés: 2021.03.14.

⁹⁰ ChromeDriver - WebDriver for Chrome, URL: <https://chromedriver.chromium.org/mobile-emulation>, Megtekintve: 2021.03.14.

⁹¹ Python - Tkinter grid() Method, URL: https://www.tutorialspoint.com/python/tk_grid.htm, Megtekintés: 2021.03.14.

kommentek helye található. A delimiter az egyetlen parancs, mely kizárálag a kód első sorában szerepelhet és az egyetlen, mely az argumentumát nem kell zárójelbe helyezni, ugyanis a parancs után helyezett delimiter maga is delimiterként funkcionál. Amennyiben nem definiáljuk, a kódban a két egymás mögé helyezett százalékjel ("%%") lesz a delimiter.

delimiter // megjegyzés

A nyelv maga rendkívül leegyszerűsített, nem tartalmaz változókat, ciklusokat, elágazásokat, a fordítása szekvenciális rendben történik, a legenerált html mégsem ennek sorrendjében születik, mivel a legenerálható lap fix elhelyezkedésű, tehát amennyiben a footer elemet állítjuk össze elsőjént sem fogja a footer megelőzni a többi elemet. A parancsok,, a delimiter parancsot leszámítva, minden zárójelbe fogják az argumentumokat, melyeket vesszők választanak el. Egyes esetekben kizárálag, máskor teljesen elhanyagolható a sorrend, ezek minden egyes esetben feltüntetésre kerülnek! Az argumentummok lehetnek:

- **Egyszerű szövegek**
- **Önálló parancsok**
- **"Stringként megjelenő szövegek"**
 - **Kulcs=Érték párok**
 - **Összetett argumentumok()**

Az összetett argumentum típusra ugyanazok a szabályok vonatkoznak, mint a parancsokra, azzal a kivétellel, hogy további összetett argumentumot nem tartalmazhatnak, tehát maximálisan két szinten jelenhetnek meg a zárójelek. A String formátumot többnyire olyan esetekben érdemes alkalmazni, ahol a fordítót megavarák a szövegen belüli vesszők és hibás fordítást eredményezne hiányuk. A Stringeket határolhatja ", ' és ` jel is, egy soron belül tartózkodjunk a vegyes használattól. Példa egy fiktív sorra:

parancs(valami, kulcs=érték, "string, string", összetett(valami, valami)) %% megjegyzés

A sorokat,mivel a newline nem számít, a könnyebb értelmezhetőség és olvashatóság kedvéért nyugodtan tagoljuk:

**parancs(valami,
 valami,**

kulcs=érték,
"string, string",
összetett(valami, valami)) %% megjegyzés

Vegyük sorra a felhasználható parancsokat:

- **basics(author=név, language=nyelv, charset=utf-8, palette=random)**

Az oldal alapvető változóit állítja be, mint a készítő neve, a nyelv és a karakterkészlet, ezeket sima szövegként kell megadni. Amennyiben ezeket kihagyjuk, a név helyett üres String, a nyelv helyen "en", a karakterkészletén pedig "utf-8" fog szerepelni. A paletta nemileg ezzel szemben a négy alapsín határozza meg, amellyel dolgozik, a paletta nevének vagy sorszámnak megadásával tudjuk az oldalt színezni, a "random" szó megadásával minden legeneráláskor más séma kerül felhasználásra. Alapértelmezetten a "black" sémát használjuk. A színek a következő oldalon láthatóak!

- **opacity(container=0.50, tables=0.75, navbar=0.75, footer=0.75)**

Az előzővel szorosan összefüggésben, beállíthatjuk a különféle oldalelemek áttetszőségét, ennek jelentősége abban rejlik, ha oldalunk hátterének egy képfájlt választottunk, akkor ennek segítségével láthatóvá tehetjük. Fontos: minden tábla és minden bootrow azonos áttetszőséggel rendelkezik, egyéni beállításra nincsen lehetőség.

| Sorszám | Név | Szín - 1 | Szín - 2 | Szín - 3 | Szín - 4 |
|---------|------------|--------------|------------------|------------------|-------------------|
| 0 | white | white | lightgray | gray | black |
| 1 | black | black | darkgray | gray | white |
| 2 | red | red | orangered | darkred | goldenrod |
| 3 | blue | blue | royalblue | slateblue | silver |
| 4 | yellow | yellow | gold | goldenrod | black |
| 5 | green | green | seagreen | mediumaquamarine | mediumspringgreen |
| 6 | purple | magenta | mediumpurple | plum | paleturquoise |
| 7 | gray | gray | lightslategray | darkgray | whitesmoke |
| 8 | lime | lime | limegreen | yellowgreen | maroon |
| 9 | peach | peachpuff | wheat | tomato | tan |
| 10 | cream | snow | seashell | tan | salmon |
| 11 | spring | springgreen | mediumaquamarine | mediumseagreen | maroon |
| 12 | autumn | burlywood | chocolate | darkgoldenrod | brown |
| 13 | gold | gold | goldenrod | darkgoldenrod | palegoldenrod |
| 14 | cyan | lightcyan | cyan | darkcyan | indigo |
| 15 | fire | orangered | red | firebrick | yellow |
| 16 | ice | whitesmoke | lightblue | skyblue | snow |
| 17 | sea | lightskyblue | lightblue | lightseagreen | seashell |
| 18 | darkpurple | orchid | rebeccapurple | purple | violet |
| 19 | sand | sandybrown | peru | palevioletred | moccasin |
| 20 | beige | ivory | beige | bisque | chocolate |
| 21 | tomato | lightsalmon | darksalmon | orangered | tomato |
| 22 | brown | saddlebrown | sienna | sandybrown | peru |
| 23 | silver | silver | powderblue | ivory | linen |
| 24 | orange | orange | darkorange | orangered | wheat |
| 25 | metalhead | black | darkred | red | darkslategray |
| 26 | gothgirld | black | deeppink | hotpink | ivory |
| 27 | blackmetal | black | ghostwhite | white | gray |

123. ábra: Színsémák

- **keyword**(fun, joy, happiness)

Egyszerű felsorolásként beilleszthetjük azokat az elemeket, amiket kulcsszavakként a honlapba fejlécébe szeretnénk írni. Alapértelmezett értéke: "".

- **description**(This is a really cool website!)

Egyszerű szövegként írjuk be a szöveget, amit a weboldal leírásának meg szeretnénk adni! Alapértelmezett értéke: "".

- **title**(Very funny website!)

Egyszerű szövegként írjuk be a címet, amit a weboldal címsorában és a böngészőben megjelenő fülön szeretnénk látni. Alapértelmezett értéke: "".

- **font-family**("Arial", "Comic Sans")

Egyszerű szöveges felsorolásként, vagy Stringek felsorolásaként írjuk be a betűtípusokat, amiket a weboldal megjelenítéshez használni szeretnénk. Alapértelmezett értéke: "Arial".

- **background**(image=lol.png)

A háttér beállításához három módszert alkalmazhatunk:

- **color**

Egyszínű háttér, mely a színséma első színét állítja be.

- **gradient=HOR**

Színátmenetes hátteret állít be a színséma első és második színe segítségével. Három lehetséges értéket vehet fel:



124. ábra: Átmenetes hátterek

- **image=img/veryhappy.png**

Egyszerű háttérkép beállítása, mely az oldal görgetésétől függetlenül, fixáltan helyezkedik el.

- **banner**(image=soocool.png,
size=cover, height=400,

```
text("CoolBanner", 3em, center))
```

A banner az oldal felső részében helyezkedik el és a rendkívül kis méretű képernyőket (xs méret, 576px alatt) leszámítva minden esetben látható. Három eleme van, a szöveg, a méret és a kép, esetlegesen képek animációja.

- A **text** elemben a zárójelben a fent látható sorrendben kell az elemeket felsorolni, elsőkétn String formátumban a megjelenítendő szöveget (alapértelmezett értéke = ""), a betűméretet (lehet em vagy px, kiírandó, alapértelmezetten 3em), valamint a szöveg horizontális tájolását (alapértelmezetten center).
- A **size** a banner keretkitöltési módját állítja be, lehetséges értékei: **cover** (alapértelmezés), **contain**.
- A **height** a banner magasságát állítja be, pixelben. Csak a számot kell megadni. Alapértelmezett értéke: 200.

A banner háttérképének beállítására kétféle lehetőség van:

- **image=what.png**
Egyszerű, statikus háttérképet állít be.
- **animation(20s, img/01.jpg, img/02.jpg, img/03.jpg)**
A képek egy egyszerű animációval, átmenettel váltakoznak, az animáció a végéhez érve újraindul. Az összetett argumentumban felsorolásszerűen kell megadni a kívánt elemeket, kezdve az animáció idejével, majd felsorolva tételesen az abban részt vevő képeket.
- **navbar(brand(image=wow.png), sticky, expand=lg,**
item("Happy", happy),
item("Sad", sad),
item("Angry", angry))

A navbar az a navigációs felület, amellyel az oldalon belül lévő különféle szegmensekre ugorhatunk. Mivel mobilra optimalizált weblapokat készíthetünk, a lapokon külső linkekkel nem helyezhetünk el, illetve, az automatikus görgetés smooth-ként működik a jobb felhasználói élmény érdekében.

- **brand(image=wow.png) || brand("SuchWoW")**

A **brand** a navbar legelső eleme, mely a weboldal nevére utal és linkje az oldal címére mutat. Beállíthatunk egy képet a már megszokott módon, vagy oldalunk nevét Stringként. Alapértelmezett értéke = "".

- **item("Joy", joy)**

A navbar további elemeit, melyek egyszerű linkek, tételesen felsorolhatjuk, minden egyes item hozzáadódik a navbarunkhoz a feldolgozás sorrendjében. A link szövegét Stringként adhatjuk meg, az **id**-t pedig sima szövegként (nem kell a "#").

- **sticky**

Amennyiben megadjuk a **sticky** tag-et, a navbar lefele görgetéskor is az oldal tetején marad.

- **expand=lg**

Meghatározza, mekkora méret felett legyen automatikusan kibontva a navbar, ellenkező esetben egy gombra kattintva nyílik ki legördülő menüként. Megadható egy 0-4-ig, vagy a megfelelő méret névével ("xs", "sm", "md", "lg", "xl").

- **table(id=nice, columns("Mood", "Power", "Wish"),
row("Happy", "Happiness", "More joy."),
row("Sad", "Sadness", "Less pain."),
row("Angry", "Anger", "More hatred.))**

A table az egyik legegyszerűbb eszköze az adathalmaz érzékletes ábrázolására, viszonylag egyszerűen létrehozhatjuk őket a fent látható szintaxissal.

- **id=valami**

A navbarban található linkhez kiadott id hivatkozás itt állítható be.

- **columns("One", "Two", "Three")**

Az oszlopnevek megadását String formátumban, vesszővel elválasztva tehetjük meg.

- **row("1", "2", "3")**

Ugyanígy adhatjuk mega sorok tartalmát, egyesével.

- **inverted**

Megadásával a table színei megfordulnak, jobban illeszkednek a világosabb háttérhez.

- bootrow(id=mehhh, rates(3,6,3), imgfilter,
image=first.jpg,**

```
article(title="Happy Title", title-align=center,
        rawtext="<b>This a bold and <i>this is an italic bold
text!</i></b>"),
        image=second.jpg)
```

A bootrow a legösszetettebb elem a szintaxisban, mely a legjobban kihasználja a bootstrap nyújtotta lehetőségeket, komplex hasábokat képezhetünk weboldalunk számára.

- **id=valami**

A navbarban található linkhez kiadott id hivatkozás itt állítható be.

- **rates(auto) || rates(4,4,4)**

A Bootstrap alapszabálya, hogy 12 oszlopra osztja fel a hasábot, a rates-zel ezt tudjuk szabályozni. Amennyiben az auto értéket adjuk meg, minden elem automatikus méretezéssel kerül be a sorba, amennyiben megadjuk az arányszámokat vesszőkkel elválasztva, a megadott elemek ebben a sorrendben kapják meg az arányszámukat. A program ellenőrzi, hogy a megadott számok összege 12-e, illetve, hogy a számok száma és az elemek száma a sorban egyezik-e.

- **image=something.png**

A képeket egyszerűen a kulcs-érték pár formában adhatjuk meg beillesztésre.

- **article(title="WOW", title-align=right, rawtext="LOL")**

Az **article** komplett bejegyzéseket tárol el a mezőben. A cím mérete minden esetben 3em, viszont a **title-align** argumentummal megadható, hogy vízszintesen a mező mely részére legyen tájolva. A szövegtörzs maga minden justify elrendezésű.

A **title** egyszerű Stringként tárolja el a blokk címét, a **rawtext** esetében szintén Stringet kell megadni, viszont használhatóak az egyszerűbb, már ismert html tag-ek is.

- ✓ Félkövér tag
- ✓ <i>Dőlt tag</i>
- ✓ <u>Aláhúzott tag</u>
- ✓
: Új sor
- ✓ <space:X>: Saját tag, tetszőleges számú space beszúrása, az X helyére történő szám írásával.

- **imgfilter**

A képekhez egy speciális filtert rendel, ugyanazokkal a beállításokkal, mint amivel a bannerban található animáció során a képek áttűnést produkálnak. A képek szürkeárnyalatosak és homályosak, csak akkor válnak élessé és színessé, ha az egérmutató fölött visszük.

footer(id=foot, facebook=link, button="It is a good day to die!")

A footer a lap alján található elem, mely elsődlegesen a készítő elérhetőségeit tartalmazza, továbbá egy gombot, amivel visszatérhet a felhasználó az oldal tetejére. Amennyiben az képernyő mérete közepesnél kisebb, a gomb kitölti a lap alját és ahhoz "tapad", az érintőképernyőt használók számára jelentősen meg könnyítve a navigációt.

▪ id=valami

A navbarban található linkhez kiadott id hivatkozás itt állítható be.

▪ {media}=link

A footerben helyezhetőekel a közösségi média ikonok, a név mellé a linket kell elhelyezni, a már megismert kulcs=érték szintaxissal. A Boo-T az alábbi elemeket támogatja:

- **facebook**
- **twitter**
- **googleplus**
- **skype**
- **youtube**
- **vkontakte**
- **linkedin**
- **email**
- **instagram**
- **github**
- **phone**

▪ button="Some cool text"

A gombon olvasható felirat egy String formátumú szöveggel módosítható.

IV/2. Felhasználói felület

A program ablakos felületet használ, melyen a kód a többsoros beviteli mezőbe illeszthető, a szintexiskiemelő a gépeléssel szinkronban színezi át a szöveget könnyen olvasható és áttekinthető formába. Ezen felül gombokkal, választógombokkal és jelölőnégyzetekkel, beviteli mezővel és listával, valamint felugró ablakokkal találkozhat a felhasználó a program működtetése során.

The screenshot shows the Boo-T application window. The title bar has the text "Boo-T". The menu bar includes "File", "Edit", "Format", "Tools", and "Help". The toolbar contains icons for new file, open file, save, save as, print, copy, cut, paste, find, and settings. The main area is a code editor with the following content:

```
deliminator hullahopp
basics(author=Fehér János Zoltán (Member),
language=en,
charset=utf-8,
palette=27) hullahopp

keywords(Help, Boo-T, Bootstrap) hullahopp
description(This is the help file for the Boo-T application) hullahopp
title(Help) hullahopp

background(image=E:/PyCharm/P/Boo-T/Back.jpg) hullahopp
font-family("Myriad Pro Light", "Informal Roman", "Gabriola", "Arial") hullahopp
opacity(navbar=0.55, container=0.75, table=0.75, footer=0.55) hullahopp

banner(size=cover, height=450, animation(50s,
E:/PyCharm/P/Boo-T/Spooky1.jpg,
E:/PyCharm/P/Boo-T/Spooky2.jpg,
E:/PyCharm/P/Boo-T/Spooky3.jpg,
E:/PyCharm/P/Boo-T/Spooky4.jpg),
text("Help with Boo-Ting!", 4em, center)
) hullahopp

navbar.brand(image=E:/PyCharm/P/Boo-T/ghost.png), expand=xl,
item("Welcome", welcome).
```

The sidebar on the right includes "Recent Files" with "Help.boo", "Open File" (button), "Import Image" (button), and a "Syntax List" pane containing:

- DIG
- HOR
- VER
- animation
- article
- author
- auto
- background
- banner
- basics
- bootrow
- brand

125. ábra: Világos mód

Boo-T

Recent Files
Help.boo

delimiter hullahopp
basics(**author**=Fehér János Zoltán (Member),
 language=en,
 charset=utf-8,
 palette=27) hullahopp

keywords(Help, Boo-T, Bootstrap) hullahopp
description(This is the help file for the Boo-T application) hullahopp
title(Help) hullahopp

background(**image**=E:/PyCharm/P/Boo-T/Back.jpg) hullahopp
font-family("Myriad Pro Light", "Informal Roman", "Gabriola", "Arial") hullahopp
opacity(**navbar**=0.55, **container**=0.75, **footer**=0.55) hullahopp

banner(**size**=cover, **height**=450, **animation**(50s,
 E:/PyCharm/P/Boo-T/Spooky1.jpg,
 E:/PyCharm/P/Boo-T/Spooky2.jpg,
 E:/PyCharm/P/Boo-T/Spooky3.jpg,
 E:/PyCharm/P/Boo-T/Spooky4.jpg),
 text("Help with Boo-Ting!", 4em, center))
) hullahopp

navbar(**brand**(**image**=E:/PyCharm/P/Boo-T/ghost.png), **expand**=xl,
 item(["Welcome"], welcome)).

Open File
Import Image

Syntax List

- DIG
- HOR
- VER
- animation
- article
- author
- auto
- background
- banner
- basics
- bootrow
- brand
- ...

Paste Text

126. ábra: Sötét mód

Menü



Új fájl

Törli az aktuális munkamenetet, opcionálisan betölti az új fájl sémát.



Megnyitás

Korábbi munkamenetet megnyitását teszi lehetővé.



Mentés

Elmenti az aktuális munkamenetet.



Mentés másként

Új fájlba elmenti az aktuális munkamenetet.



Másolás

Vágólapra másolja a kijelölt szövegrészleteket.



Beillesztés

Beilleszti a szövegdobozba a vágóapon lévő szöveget.



Visszavonás

Visszatér a munkamenet eggyel korábbi állapotába.



Kódgenerálás

Legenerálja és menthetővé teszi a weboldalt.



Teszt FireFox-szal

Azonali tesztet biztosít FireFox segítségével.



Teszt Chrome-mal

Azonali tesztet biztosít Chrome segítségével.



Teszt Edge-el

Azonali tesztet biztosít Edge segítségével.



Teszt Opera-val

Azonali tesztet biztosít Opera segítségével.



Beállítások

Megnyitja a beállítási lehetőségeket tartalmazó ablakot.



Súgó

Megnyitja a fejlesztési tippeket tartalmazó weblapot.



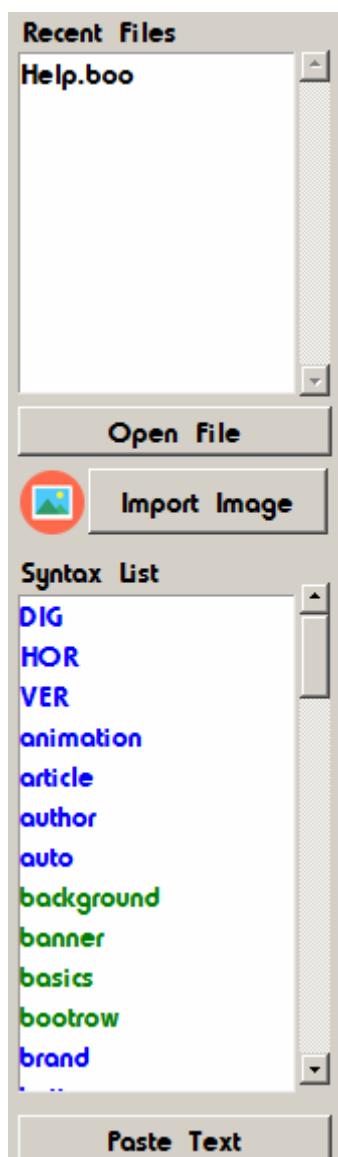
Névjegy

Megnyitja a névjegy ablakot.

Az ablak oldalán látható két lista, a felsőben a legutóbb megnyitott fájlok, míg a lentiben a már említett szintaxis elemei találhatóak tételesen felsorolva.

A legutóbbi fájlok listájában lévő fájlokat értelemszerűen meg lehet nyitni, míg a szintaxislista elemeit a gombbal a szövegdobozhoz lehet adni. A két elem között található a képimportáló, mely egyszerű fájldialógus ablakon keresztül segít a képek elérhetőségét a kódhoz adni. Használatának jelentősége, hogy mindenkor a teljes utat használja, a fordító pedig, ha ilyen utat talál, a képfájlt a weblap mellett található "img/" mappába másolja, a kódban pedig lecseréli a teljes utat erre a relatív útvonalra, vagyis a fejlesztőnek nem kell foglalkoznia a mappaszerkezet kialakításával. Ugyanígy másolásra kerülnek a Bootstrap rácsrendszer elemei.

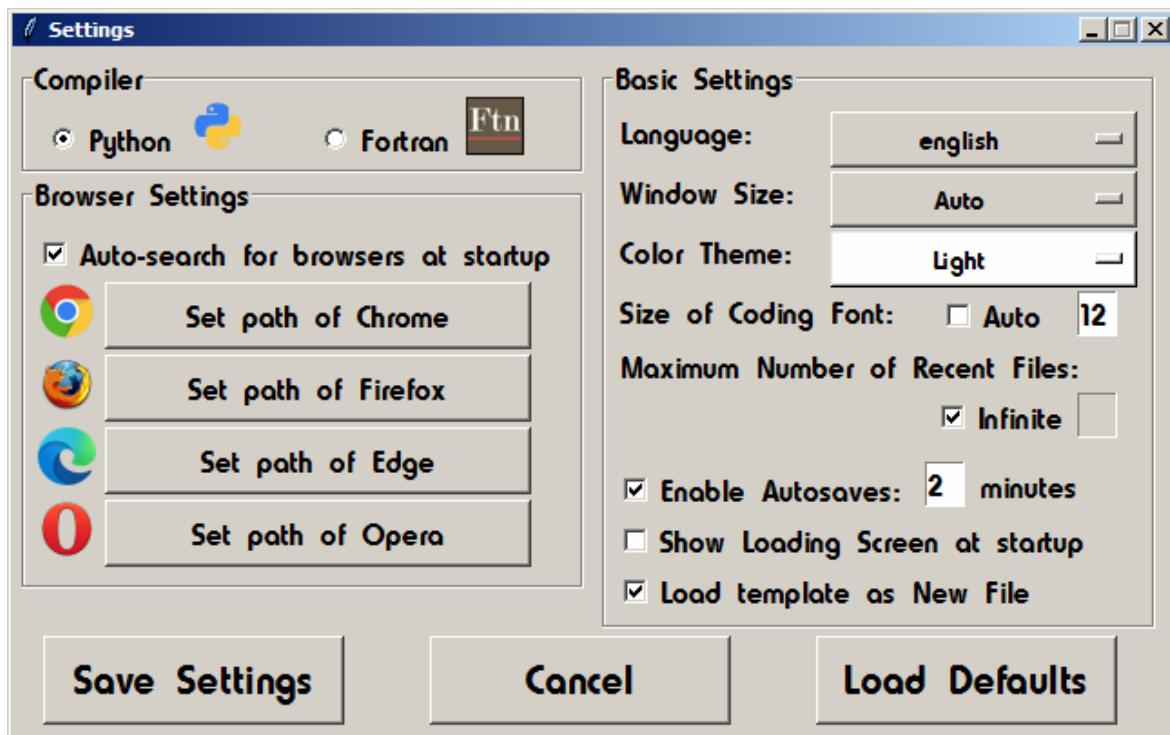
A Beállítások menüre kattintva megjelenik az ablak, amelyen keresztül a Boo-T működése módosítható. Az oldal alján található három gombbal menthető a konfiguráció, visszavonhatóak a változtatások és betölthetjük az alapértelmezett konfigurációt, amennyiben úgy ítélik meg, hogy az alkalmazás működését maguktól nem tudnának az addig tapasztalt felhasználói élménnyel egy szintre hozni.



127. ábra:
Listadobozok

Az ablak első keretében a fordító típusa állítható be, a Python alapú fordító minden operációs rendszeren, míg a Fortran alapú csak Windows operációs rendszeren használható. A böngésző beállítóban megadható, hogy amennyiben adott böngészőre nincsen elérési út beállítva, az alkalmazás tegyen kísérletet annak felkutatására, illetve, manuálisan is megkereshetjük őket vagy módosíthatjuk a már megadott elérési utakat.

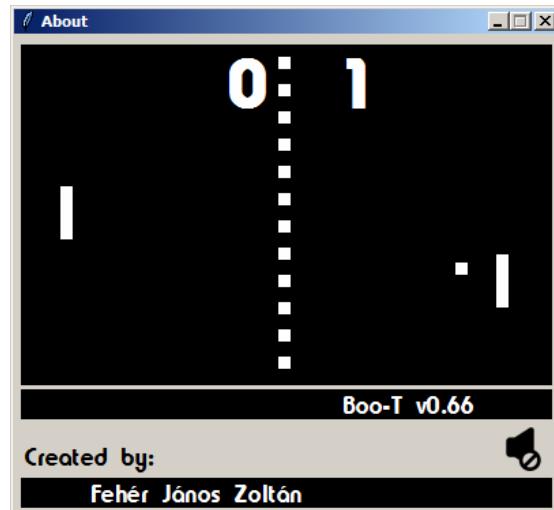
A jobboldali menüben kiválaszthatjuk a használni kívánt nyelvet, az ablak méretezését vagy válthatunk a világos/sötét mód között. Az ablakméret automatikusan állítása esetén a program a betöltéskor határozza meg, melyik méretet volna az elsődleges monitorunk felbontásához mérten a legoptimálisabb.



128. ábra: Beállítások menü

A három választóelem alatt kiválasztható, hogy a kódmező betűtípusát manuálisan adjuk meg, vagy az ablakméret alapján a program határozza meg azt, beállíthatjuk, hogy a program hány legutóbb megnyitott állományt tároljon el, valamint hogy milyen időközönként készüljön a munkánkról automatikus mentés arra az esetre, ha valamiért az alkalmazás működésében hiba következne be.

A keret alsó felén található még még jelölőnégyzet, ezekkel állíthatjuk be, hogy a program indításakor töltődjön-e be az üdvözlöképernyő, valamint, hogy új fájl készítésekor a program betöltsse e a rendelkezésre álló vonalvezető sémát, vagy teljesen üres dokumentummal kezdődjön a fejlesztés.



129. ábra: Névjegy menü

Említendő még a névjegy menü, mely a verziószámot, a készítő nevét és egy apró játékprogramot tartalmaz.

| | |
|------------|--|
| F1 | Súgó megnyitása |
| F2 | Fájl mentése |
| F3 | Fájl megnyitása |
| F4 | Kód fordítása |
| F5 | Kép beszúrása |
| F6 | Sötét/Világos mód közötti váltás |
| F7 | Visszavonás művelet |
| F8 | Delimiter beszúrása sorkommenttel |
| F9 | Beállítások menü megnyitása |
| F10 | Alapértelmezett böngészőben tesztelés |
| End | Delimiter beszúrása sorkommenttel |

130. ábra: Gyorsbillentyűk

A program használata során a fenti gyorsbillentyűk használatosak. Kiemelendő, hogy az F6 billentyűvel történő színséma váltás nem mentődik automatikusan a merevlemezen található állományba. Az F8/End billentyűkkel elérhető delimiter beszúrás az alábbi formátumban beszúrja a sor végére a delimiter-t és az adott parancs sorszámát kommentként.

parancs() %% (2)

Ennek előnye, ha a feldolgozás során hiba keletezik, a fordító minden esetben megjelöli, melyik parancsnál talált nem értelmezhető paramétert, könnyebben beazonosítható lesz a hiba forrása hosszú kód esetében. Jellemzően a nem megfelelő zárójelezés, valamint a vesszőhibák okoznak hibát a kód fordításában.

V. Összegzés

A Boo-T ugyan fel nem veheti a versenyt a professzionálisan, nagy költségvetéssel fejlesztett weblapfejlesztőkkel, de azon eredeti célkitűzéseknek eleget tesz, hogy

- Letisztult, egyszerű nyelvezetével és felületével a kezdő fejlesztők számára támponot és kezdeti sikerélményeket nyújt.
- Minimális erőfeszítéssel reszponzív, vizuálisan kielégítő honlapot hozhatnak vele létre ingyenesen magánszemélyek, kis költségvetésű szereplők.

Az alkamazás mind Windows, mind Linux alatt működőképes és ellátja a feladatát, így a szabad szoftverhasználat híveinek sem kell lemondania a használatáról. A fejlesztés folyamán részlegesen sikerült az OOP alapelvek betartása, mivel a kissé monolitikus MainWindow objektum, amennyiben több idő állt volna a fejlesztésben rendelkezésre, valamint az ezzel kapcsolatos ismeretanyag is még a fejlesztés kezdeti szakaszában rendelkezésre állt volna, lehetőség lett volna a különböző, jól körülírható mintába rendezett ablakelemek csoporthainak objektumokká szervezésére. A vártnál a már említett ablakos kezelőfelület lényegesen tovább tartott, míg a konfigurációs állományok, a háttérmiűveletek vagy éppen a fordító állomány implementálása, melyek fejlesztése a tervnek megfelelő ütemben, terjedelemmel és hatékonyssággal haladt.

A Fortran 90 alapú fordítóhoz az összetett adattípusok és megvalósítása mutatkozott a legérdekesebb és legnagyobb kihívásnak, mivel ezekkel a nyelv annak ellenére nem rendelkezik, hogy már elérhető a 2018-as változat, viszont az igények nyomán inkább a C nyelvvel való kompatibilitást és a számoláshoz köthető elemeket fejlesztették. A nyelv komoly hátrányának bizonyult, hogy bizonyos méretek felett a memóriaallokáció nem működik tökéletesen, így előfordult dinamikusan növekvő karakterláncok és tömbök esetében (különösen 1000 karakter felett), hogy a beolvasás során a következő elem eleje és feldolgozásra kerül, ami ellen az utolsó érvényes karakter beállításával volt lehetőségem védekezni, viszont Linux alatt kénytelen voltam a használatát az alkalmazásból eltávolítani, mivel a Code::Blocks környezetében nem, de a Python alkalmazásból meghívva jellemzően rossz memóriacímról olvasta be a kód sorait, így a szótár állomány kulcsait és hozzá tartozó kifejezéseit kísérelte meg kódként értelmezni. Ugyan a hiba hardkódolással történő kezelése esetén a fordítandó kódból használható weboldal készült, de ezt a legkevésbé sem találtam elegáns és megengedhető megoldásnak⁹², ezért inkább **kivettem**, mint lehetőséget, Windows alatt hasonló

⁹² AL SWEIGART: Beyond the Basic Stuff with Python, No Starch Press [USA], 2021, 196. o.

problémákat nem tapasztaltam, így ott megtartottam a compiler-választás lehetőségét, de a Fortran változat inkább, mint érdekességet, nem, mint állandó használati lehetőséget javaslom.

Összefoglalón, a feladat teljesült, a program működőképes, de mint minden alkalmazás esetében, végelegesnek és teljesnek nem tekinthető. Ha egy egyszerű hármas modellben akarjuk ábrázolni a projekt sikereségét (Make it work, make it right, make it fast⁹³), az első tekintetében a program jól teljesít, a helyes és szép megvalósítás esetében érheti kritika, sebességében a végrehajtás jónak mondható, Python fordítóval a fordítás ideje emberi léptékben nem érzékelhető, Fortran alapú esetében (a külső I/O műveletek miatt) tizedmásodpercek kérdése.

⁹³ MICHA GORELICK, IAN OZSVALD: Hight Performance Python 2nd edition, O'Reilly Media, Inc, USA [2020] , 56. o.

VI. Irodalomjegyzés

VI/1. Könyv formátumú irodalmak

- AL SWEIGART: Beyond the Basic Stuff with Python, No Stach Press [USA], 2021
- ALI. H. DOGRU: Modern Software Engineering Concepts and Practices: Advanced Approach, Middle East Technical University [Turkey], 201
- MARIA DEL PILAR SALAS ZARATE, GINER ALOR-HERNÁNDEZ, RAFAEL VALENCIA-GARCIA, LISBETH RODRIGUEZ: Analyzing best practices on Web development frameworks IN Science of Computer Programming - 112 [2015]
- MICHA GORELICK, IAN OZSVALD: Hight Performance Python 2nd edition, O'Reilly Media, Inc [USA], 2020
- STEVEN LOTT: Functional Python Programming, Pakt Publishing [Burningham - USA], 2015,
- NORMAN S. CLERMAN, WALTER SPECTOR: Modern Fortran, Cambridge University Press [USA], 2012

VI/2. Névvel vállalt cikkek

- ALAN SMITH: 7 Web Design Rules You Should Never Break, URL: <https://usabilitygeek.com/web-design-rules-you-should-never-break/>, Megtekintve: 2020.12.28.
- ALAN SMITH: How Scrolling Can Make (Or Break) Your User Experience , URL: <https://usabilitygeek.com/how-scrolling-can-make-or-break-your-user-experience/>, URL: 2020.12.28.
- ANDREI BOYANOV: Python Design Patterns: For Sleek And Fashionable Code, Megtekintés: 2021.03.13.
- ANDY CRESTODINA: Web Design Standards: 10 Best Practices on the Top 50 Websites, URL: <https://www.orbitmedia.com/blog/web-design-standards/>, Megtekintve: 2020.12.28.
- CAIO COZZA: A quick overview of the Python Virtual Machine — Pt. 1, URL: <https://medium.com/@caiocozza.art/a-quick-overview-of-the-python-virtual-machine-pt-1-315e74c036f4>, megtekintve: 2020.10.10
- JOSEPH JOHNSON: Which one of these devices do you use most for surfing or browsing the internet, URL: <https://www.statista.com/statistics/308449/device-preference-for-internet-browsing-in-the-uk/>, megtekintve: 2020.11.02
- MARK OTTO: Bootstrap from Twitter, URL: https://blog.twitter.com/developer/en_us/a/2011/bootstrap-twitter.html, megjelenés dátuma: 2011.08.19.
- MEZŐ ISTVÁN: Finally, a CSS only solution to :hover on touchscreens, URL: <https://medium.com/@mezoistvan/finally-a-css-only-solution-to-hover-on-touchscreens-c498af39c31c> Megtekintve: 2020.12.28.
- NICK BABICH: <https://xd.adobe.com/ideas/principles/web-design/12-dos-donts-web-design-2/> (2020.12.28.)
- NICK BABICH: The 12 Do's and Don'ts of Web Design, URL: <https://xd.adobe.com/ideas/principles/web-design/12-dos-donts-web-design-2/>, megtekintve: 2020.12.28.

- TOMIWA ADEMIDUN: Why We Chose Angular over React and Django Over Ruby on Rails: How to Choose A Software Startup, URL: <https://hackernoon.com/why-we-chose-angular-over-react-and-django-over-ruby-on-rails-for-tila-ca-77ac03d542cf>, megtekintve: 2020.12.28.

VI/3. Dokumentációs leírások

- 9. Classes: <https://docs.python.org/3/tutorial/classes.html>, megtekintve: 2020.10.10.
- About Daemons and Services, URL: <https://developer.apple.com/library/archive/documentation/MacOSX/Conceptual/BSSystemStartup/Chapters/Introduction.html>, megtekintve: 2020.03.03
- Angular Docs, URL: <https://angular.io/docs>, megtekintve: 2020.12.28.
- Binding function in Tkinter, URL: <https://www.geeksforgeeks.org/python-binding-function-in-tkinter/>, Megtekintve: 2021.03.14.
- Canvas Widgets, URL: https://www.python-course.eu/tkinter_canvas.php, Megtekintés: 2021.03.14.
- CDLL, URL: <https://www.kite.com/python/docs/ctypes.CDLL>, Megtekintés: 2021.03.14.
- Chapter 1. Understanding Performant Python, URL: <https://www.oreilly.com/library/view/high-performance-python/9781492055013/ch01.html>, megtekintve: 2021.03.12.
- ChromeDriver - WebDriver for Chrome, URL: <https://chromedriver.chromium.org/mobile-emulation>, Megtekintve: 2021.03.14.
- clipboard 0.0.4, URL: <https://pypi.org/project/clipboard/>, Megtekintve: 2020.03.13
- Common Causes of Segmentation Faults (Segfaults), URL: https://www.nas.nasa.gov/hecc/support/kb/common-causes-of-segmentation-faults-segfaults_524.html, Megtekintés: 2021.03.14.
- Converting Between FORTRAN and C Strings, URL: <http://starlink.eao.hawaii.edu/docs/sun209.htm/sun209se6.html>, Megtekintés: 2021.03.14.
- ctypes — A foreign function library for Python, URL: <https://docs.python.org/3/library/ctypes.html>, Megtekintve: 2021.03.13.
- datetime — Basic date and time types, URL: <https://docs.python.org/3/library/datetime.html>, Megtekintve: 2021.03.13.
- Font Manager, URL: <https://github.com/FontManager/font-manager>, Megtekintés: 2021.03.13.
- Fortran History. URL: <http://fortranwiki.org/fortran/show/Fortran+History>, Megtekintés: 2021.03.14.
- Fortran/Fortran procedures and functions, URL: https://en.wikibooks.org/wiki/Fortran/Fortran_procedures_and_functions, Megtekintve: 2021.03.14.
- Fortran/structures, URL: <https://en.wikibooks.org/wiki/Fortran/structures>, Megtekintve: 2021.03.14.
- Gnome-Font-Viewer, URL: <https://launchpad.net/ubuntu/+source/gnome-font-viewer>, Megtekintés: 2021.03.13.
- Graphical User Interfaces with Tk, URL: <https://docs.python.org/3/library/tk.html>, Megtekintve: 2021.03.13.
- Grid system, URL: <https://getbootstrap.com/docs/4.0/layout/grid/>, Megtekintve: 2021.03.13.

- How do I print a listing of files in a directory?, URL: <https://www.computerhope.com/issues/ch000772.htm>, Megtekintés: 2021.03.14.
- Implicit None, URL: <http://www.personal.psu.edu/jhm/f90/statements/implicit.html>, Megtekintés: 2021.03.14.
- Inside the Python Virtual Machine, URL: <https://leanpub.com/insidethepythonvirtualmachine/read>, Megtekintve: 2021.03.13.
- Különbség a tömblista és a kapcsolt lista között, URL: <https://hu.sawakinome.com/articles/software/difference-between-array-list-and-linked-list-3.html>, Megtekintés: 2021.03.14.
- Loading Custom Fonts, URL: https://pythonhosted.org/pyglet/programming_guide/loading_custom_fonts.html, Megtekintés: 2021.03.14.
- Object-Oriented Programming in Fortran 2003 Part 1: Code Reusability, URL: <https://gist.github.com/n-s-k/522f2669979ed6d0582b8e80cf6c95fd>, Megtekintés: 2021.03.14.
- Opening the Display, URL: <https://tronche.com/gui/x/xlib/display/opening.html>, Megtekintés: 2021.03.14.
- os — Miscellaneous operating system interfaces, URL: <https://docs.python.org/3/library/os.html>, Megtekintve: 2020.03.13.
- Paddle (Game Controller), URL: [https://en.wikipedia.org/wiki/Paddle_\(game_controller\)](https://en.wikipedia.org/wiki/Paddle_(game_controller)), Megjelenés: 2021.03.14.
- Pillow — Pillow (PIL Fork), URL: Pillow — Pillow (PIL Fork), Megtekintve: 2021.03.13.
- platform — Access to underlying platform's identifying data, URL: <https://docs.python.org/3/library/platform.html>, Megtekintve: 2021.03.13.
- Platformfüggetlenség, URL: <https://www.huwiki.org/wiki/Platform-f%C3%BCggetlens%C3%A9g>, Megtekintés: 2021.03.14.
- psutil, URL: <https://pypi.org/project/psutil/>, Megtekintés: 2021.03.13.
- Pygame Mixer, URL: <https://www.pygame.org/docs/ref/mixer.html>, Megtekintve: 2021.03.14.
- PyGame, URL: <https://www.pygame.org/>, Megtekintve: 2021.03.13.
- Pyglet Homepage, URL: <http://pyglet.org/>, Megtekintve: 2020.03.13.
- Python - Tkinter grid() Method, URL: https://www.tutorialspoint.com/python/tk_grid.htm, Megtekintés: 2021.03.14.
- Python - Tkinter Listbox, URL: https://www.tutorialspoint.com/python/tk_listbox.htm, Megtekintés: 2021.03.14.
- Tkinter 8.5 reference: a GUI for Python - 24.1. Text widget indices, URL: <https://anzeljg.github.io/rin2/book2/2405/docs/tkinter/text-index.html>, Megtekintés: 2021.03.14.
- Python - Tkinter Scrollbar, URL: https://www.tutorialspoint.com/python/tk_scrollbar.htm, Megtekintés: 2021.03.14.
- Python | shutil.copy() method, URL: <https://www.geeksforgeeks.org/python-shutil-copy-method/>, Megtekintés: 2021.03.14.
- Python ctypes.WinDLL() Examples, URL: <https://www.programcreek.com/python/example/59464/ctypes.WinDLL>, Megtekintés: 2021.03.14.

- Python Data Types, URL: https://www.w3schools.com/python/python_datatypes.asp, Megtekintve: 2021.03.13
- Python Datetime, URL: https://www.w3schools.com/python/python_datetime.asp, Megjelenés: 2021.03.14.
- Python File I/O - Read and Write Files, URL: <https://www.tutorialsteacher.com/python/python-read-write-file>, Megjelenés: 2021.03.14.
- Python Frame.pack_propagate, URL: https://python.hotexamples.com/examples/Tkinter/Frame/pack_propagate/python-frame-pack_propagate-method-examples.html, Megtekintve: 2021.03.14.
- random — Generate pseudo-random numbers, URL: <https://docs.python.org/3/library/random.html>, Megtekintve: 2021.03.13.
- RANDOM_NUMBER — Pseudo-random number, URL: https://gcc.gnu.org/onlinedocs/gfortran/RANDOM_005fNUMBER.html, Megtekintés: 2021.03.14.
- re — Regular expression operations, URL: <https://docs.python.org/3/library/re.html>, Megtekintve: 2020.03.13.
- shutil — High-level file operations, URL: <https://docs.python.org/3/library/shutil.html>, Megtekintés: 2021.03.13.
- Strongly Typed Language: <https://www.sciencedirect.com/topics/computer-science/strongly-typed-language>, Megtekintés: 2021.03.14.
- subprocess — Subprocess management, URL: <https://docs.python.org/3/library/subprocess.html>, Megtekintve: 2021.03.13.
- sys — System-specific parameters and functions, URL: <https://docs.python.org/3/library/sys.html>, Megtekintve: 2021.03.13.
- TaylorSMarks/playsound, URL: <https://github.com/TaylorSMarks/playsound/issues/27>, Megtekintve: 2021.03.14.
- The Python X Library, URL: <https://github.com/python-xlib/python-xlib>, Megtekintve: 2021.03.13.
- threading — Thread-based parallelism, URL: <https://docs.python.org/3/library/threading.html>, megtekintve: 2020.03.03.
- threading — Thread-based parallelism, URL: <https://docs.python.org/3/library/threading.html>, Megtekintés: 2020.03.13.
- time — Time access and conversions, URL: <https://docs.python.org/3/library/time.html>, Megtekintés: 2021.03.13.
- TUTORIALSPOINT: Bootstrap Tutorial [PDF], URL: <https://wiki.lib.sun.ac.za/images/0/07/Bootstrap-tutorial.pdf>, 26. o., letöltve: 2020.10.21
- Victor Stinner: Python Compatibility Version, URL: <https://www.python.org/dev/peps/pep-0606/#id25>, Megtekintve: 2021.03.13
- webbrowser — Convenient Web-browser controller, URL: <https://docs.python.org/3/library/webbrowser.html>, Megtekintés: 2021.03.13.
- webbrowser – Displays web pages, URL: <https://pymotw.com/2/webbrowser/>, Megtekintés: 2021.03.14.
- winapps - Python library for managing installed applications on Windows, URL: <https://pypi.org/project/winapps/>, Megtekintve: 2021.03.13.
- winapps - Python library for managing installed applications on Windows, URL: <https://pypi.org/project/winapps/>, Megtekintve: 2021.03.13.

VI/4. Egyéb hivatkozások

- AY-3-8500, URL: <https://de.zxc.wiki/wiki/AY-3-8500>, Megtekintés: 2021.03.14.
- Most Loved, Dreaded, and Wanted Languages, URL: <https://insights.stackoverflow.com/survey/2020#technology-most-loved-dreaded-and-wanted-languages>, megtekintve: 2021.03.13.
- PYPL PopularitY of Programming Language, URL: <https://pypl.github.io/PYPL.html>, Megtekintve: 2021.03.13.
- Smartphones Now Account for 70% of US Digital Media Time, URL: <https://www.marketingcharts.com/digital/mobile-phone-111093>, Megtekintve: 2020.12.28.
- TIOBE Index for March 2021, URL: <https://www.tiobe.com/tiobe-index/>, Megtekintve: 2021.03.13.