  
**Eötvös Lóránd Tudományegyetem**  
Informatikai kar  
Algoritmusok és Alkalmazásaik Tanszék

Turing gép vizualizálása webes környezetben

Nagy SáraKelemen Márton

mesteroktatóprogramtervező informatikus Bsc  
programtervező matematikus

Budapest 2021

Tartalomjegyzék

[Bevezetés 3](#_Toc60224088)

[1. Elméleti háttér 4](#_Toc60224089)

[Alapfogalmak 4](#_Toc60224090)

[2. fejezet Felhasználói dokumentáció 7](#_Toc60224091)

[A program által megoldott feladat (nem jó ez cím) 7](#_Toc60224092)

[Célközönség (nem jó ez a cím, nem kell ide) 7](#_Toc60224093)

[Rendszerkövetelmények 7](#_Toc60224094)

[Minimális hardveres környezet 7](#_Toc60224095)

[Optimális hardveres környezet 8](#_Toc60224096)

[Szoftveres követelmények 8](#_Toc60224097)

[A szerver-oldali futtatáshoz 8](#_Toc60224098)

[Kliens-oldali futtatáshoz 8](#_Toc60224099)

[Üzembe helyezés 8](#_Toc60224100)

[Hordozhatóság 12](#_Toc60224101)

[A program használata 12](#_Toc60224102)

[Turing-gép létrehozás 12](#_Toc60224103)

[Turing-gép szerkesztő és futtató felület 14](#_Toc60224104)

[Konfigurációs panel 15](#_Toc60224105)

[Gráf diagram 16](#_Toc60224106)

[Hibák és egyéb üzenetek 17](#_Toc60224107)

[3. fejezet Fejlesztői dokumentáció 19](#_Toc60224108)

[A rendszer architektúrája 19](#_Toc60224109)

[Kliens oldal 20](#_Toc60224110)

[Alkalmazott technológiák 20](#_Toc60224111)

[Klens oldali technológiák amiket használok 20](#_Toc60224112)

[Alkalmazott módszerek 21](#_Toc60224113)

[Adatbázis terve 21](#_Toc60224114)

[Modul és osztályszerkezet 21](#_Toc60224115)

[A Kliens-oldal osztályszerkezet 21](#_Toc60224116)

[A felhasználó felület terve 21](#_Toc60224117)

[Telepítés fejlesztői célokra 21](#_Toc60224118)

[Megvalósítás 21](#_Toc60224119)

[4. fejezet 21](#_Toc60224120)

[5. fejezet Tesztelés 22](#_Toc60224121)

[6. fejezet Irodalomjegyzék 23](#_Toc60224122)

# Bevezetés

A szakdolgozat célja, hogy mások számára vizuális eszközökkel könnyebben megérthető legyen a Turin gép működése. A szakdolgozatnak két fő részre van, az algoritmus implementálása egy programozási nyelvre és a szemléltetés kivitelezése. A modell egyetlen programot hajt végre bármilyen inputra (ami szalagon érkezik), azaz tekinthető egy célszámítógépnek. A gép főbb részei: a vezérlőegység (CPU), a szalag, amely az inputot hivatott megvalósítani és egy író-olvasó fej, amely a szalagot még léptetni is tudja. A megvalósítandó gép determinisztikus, továbbá minden esetben definiált az átmenet. A végtelen szalag potenciálisan végtelen tár.

A lehetséges Turing gépek egy előre definiált fájlformátumban adhatók meg. Ezek közül választva futtathatunk többféle gépet is. Egy-egy futtatásánál választható több mód is: lehetséges az eredmény azonnali megtekintése, illetve a futás végig nézhető lassítva és lépésenként, kézileg léptetve is.

# Elméleti háttér

## Alapfogalmak

Forrás: https://web.cs.elte.hu/~tichlerk/logika/h/tg.pdf

A Turing-gép egy olyan M = {Q, Σ, Γ, δ, q0, qi, qn} rendszer, ahol

* Q az állapotok véges, nemüres halmaza,
* q0, qi, qn ∈ Q, q0 a kezdő- qi az elfogadó- és qn az elutasító állapot,
* Σ és Γ ´ábécék, a bemenő jelek illetve a szalagszimbólumok ábécéje úgy, hogy Σ ⊆ Γ és ⊔ ∈ Γ \ Σ.
* δ : (Q \ {qi, qn}) × Γ -> Q × Γ × {L, R, S} az átmenet függvény.

• A Turing-gép működésének fázisait a gép konfigurációival írjuk le. A Turing-gép konfigurációja egy uqv szó, ahol q ∈ Q és u, v ∈ Γ∗, v != *ε*.  
A konfiguráció a gép azon állapotát tükrözi amikor a szalag tartalma uv (uv előtt és után a szalagon már csak ⊔ van), a gép a q állapotban van, és a gép író-olvasó feje a v szó első betűjén áll.  
• A gép kezdőkonfigurációja egy olyan q0u szó, ahol u csak Σ-beli betűket tartalmaz.  
  
• Egy Turing-gép konfigurációátmenetét az alábbiak szerint definiáljuk. Legyen uqav egy konfiguráció, ahol a ∈ Γ, u, v ∈ Γ∗.

* Ha δ(q; a) = (r; b; R), akkor uqav |- ubrv’ , ahol v’ = v , ha v != *ε*, különben v’ = ⊔,
* ha δ(q; a) = (r; b; S), akkor uqav |- urbv,
* ha δ(q; a) = (r; b; L), akkor uqav |- u’rcbv, ahol c ∈ Γ és u’c = u, ha u != *ε*, különben u’ =u *és* c = ⊔ .

• Azt mondjuk, hogy M véges sok lépésben eljut a C konfigurációból a C’ konfigurációba (jele C |-\* C’), ha van olyan n ≥ 1 és C1,….., Cn konfigurációsorozat, hogy C1 = C, Cn = C’ és minden 1 ≤ i < n-re, Ci |- Ci+1.

• Ha q 2 {qi, qn}, akkor azt mondjuk, hogy az uqv konfiguráció egy megállási konfiguráció. q = qi esetében elfogadó, míg q = qn esetében elutasító konfigurációról beszélünk.

• Az M által felismert nyelv (amit L(M)-mel jelölünk) azoknak az u ∈ Σ∗ szavaknak a halmaza, melyekre igaz, hogy q0u |-∗ xqiy valamely x, y ∈ Γ∗, y != *ε*  szavakra.

• Egy L ⊆ Σ∗ nyelv Turing-felismerhető, ha L = L(M) valamely M Turing-gépre. Továbbá, egy L ⊆ *Σ*∗ nyelv eldönthető, ha létezik olyan M Turing-gép, mely minden bemeneten megállási konfigurációba jut és felismeri az L-et. A Turing-felismerhető nyelveket szokás rekurzívan felsorolhatónak, az eldönthető nyelveket pedig rekurzívnak is nevezni. A rekurzívan felsorolható nyelvek osztályát RE -vel, a rekurzív nyelvek osztályát pedig R-rel jelöljük.  
• Tekintsünk egy M = [Q, Σ, Γ, δ, q0, qi, qn] Turing-gépet és annak egy u ∈ Σ∗ bemenő szavát. Azt mondjuk, hogy M futási ideje (időigénye) az u szón n (n ≥ 0), ha M a q0u kezdőkonfigurációból n lépésben (konfigurációátmenettel) jut el megállási konfigurációba. Ha nincs ilyen szám, akkor M futási ideje az u-n végtelen.

• Legyen f : N -> N egy függvény. Azt mondjuk, hogy M időigénye f(n) (vagy, hogy M egy f(n) időkorlátos gép), ha minden u ∈ Σ∗ input szóra, M időigénye az u szón legfeljebb f(|u|).

• A **k**-szalagos Turing-gép egy olyan M = [Q, Σ, Γ, δ, q0, qi, qn] rendszer, ahol

* Q az állapotok véges, nemüres halmaza,
* q0; qi; qn ∈ Q, q0 a kezdő- qi az elfogadó- és qn az elutasító állapot,
* Σ és Γ ábécék, a bemenő jelek illetve a szalagszimbólumok ábécéje úgy, hogy Σ ⊆ Γ és ⊔ ∈ Γ \ Σ,
* δ : (Q \ {qi, qn}) × Γk -> Q × Γk × {L, R, S}k az átmenet függvény.

• A **k szalagos Turing-gép** konfigurációja egy szó, ahol q ∈ Q és ui, vi ∈ Γ∗, vi != *ε* (*1* ≤ i ≤ k). Az u szóhoz tartozó kezdőkonfiguráció: ui = *ε* (1 ≤ i ≤ k), v1 = u, és vi = ⊔ (2 ≤ i ≤ k). Időigény: mint az egyszalagosnál (konfigurációátmenetek száma alapján).

• Szófüggvényt kiszámító Turing-gép:

Az M (determinisztikus) Turing-gép kiszámítja az f : Σ∗ -> Γ∗ szófüggvényt, ha M minden u ∈ Σ∗-ra olyan vqw megállási konfigurációba jut (q ∈ {qi, qn}), ahol vw = f(u) (szóeleji és szóvégi ⊔-ektől eltekintve). Időigény: mint fent (konfigurációátmenetek száma alapján)

# Felhasználói dokumentáció

## A program által megoldott feladat (nem jó ez cím)

Az sok tananyag tanulása során néha kevés idő jut, például egy  
összetettebb algoritmus megértésére. Ez az oktatói program ebben próbál segítséget nyújtani. A gyakorlaton alapul vett gráfos ábrázolással, illetve szalagbemenettel fogja majd illusztrálni a program az algoritmus működését. Ezzel hatékonyabbá válhat a tanulás, egyebeket mellett, például segíthet leellenőrizni az elkészített házifeladat helyességét vagy az órán a tanárnak nem muszáj feltétlen minden alkalommal rajzolni a táblára helyette csak kivetíti ezt a webes alkalmazást.

## Célközönség (nem jó ez a cím, nem kell ide)

A program elsősorban informatikai karon, azon belül formális nyelvek, algoritmusok,  
számításelméletet oktató vagy tanuló egyetemi tanárok és hallgatóik számára készült. Ezt  
mutatja a program weboldal formátuma is: minimális hardverkövetelménye okán nincs szükség komolyabb eszközre, így szinte bármilyen intézményben könnyedén alkalmazni lehet. A program segítséget nyújt a hallgatóknak az egyes előre betáplált példa feladatok  
kipróbálásában, ezek újra szerkesztésében. Továbbá segít az oktatóknak is például a zárthelyi feladatok kitalálásában vagy diákok által megírtak kijavításában.

## Rendszerkövetelmények

### Minimális hardveres környezet

Legegyszerűbben személyi számítógépen érdemes használni a programot. Például a piacon amit kapni egyik legolcsóbb ilyen eszköz a Rasperry Pi 4 modell B miniszámítógépen.

* 1,5 GHz (1 500 MHz)
* 4 GB mermória

Már ez is képes futtatni egy web böngészőt és egy webszervert. Tehát minimális hardveres követelményei vannak a programnak.

### Optimális hardveres környezet

Szerver számítógép. Minta a http://webprogramozas.inf.elte.hu/. Egy ilyen szerver képes egyszerre több felhasználót kiszolgálni, ha mondjuk egy tanteremben vagy előadáson a tanár kérésére minden diáknak ki kell próbálni a programot.

### Szoftveres követelmények

#### A szerver-oldali futtatáshoz

Személy számítógépen acpache webszervert érdemes telepíteni. Ilyet legegyszerűbben a XAMPP nevű szabad és nyílt forrású platformfüggetlen webszerver-szoftvercsomagot érdemes installálni. Egyszerű és bármely desktop operációs rendszerre jó. Továbbá, szerver számítógépre más eljárással érdemes telepíteni webszervert, itt érdemes a szerver üzemeltetők segítségét kérni.

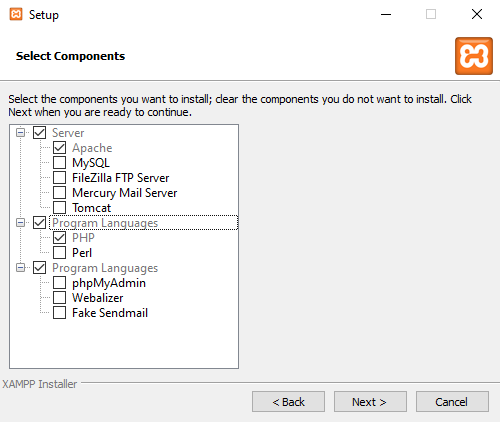
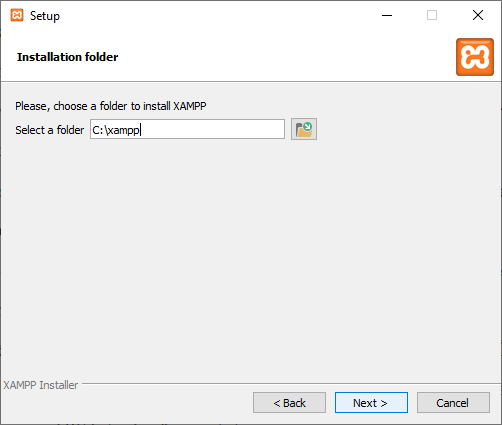
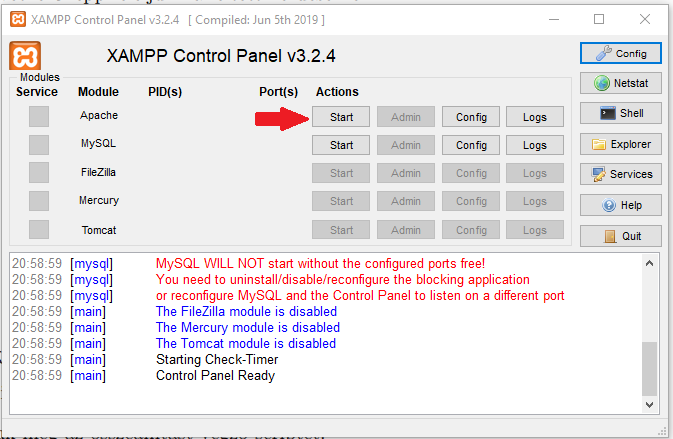
Kliens-oldali futtatáshoz

ES6 és CSS3 támogatású webböngészők:

* Desktopon: legalább az alábbi kiadásokon: Firefox 24, Chrome 37, Safari 9, Edge 12, Opera 24
* Mobil, tablet eszközökön: IOS safari 10, Android Browser 81

## Üzembe helyezés

Ha **személyi számítógépről** szeretnénk hostolni a programot, akkor szűkség van egy webszerver szoftverre. Én a XAMPP-t tudom ajánlani én is ezt használtam a fejlesztés során.

* https://www.apachefriends.org/hu/download.html (a link 2020.12.23-án érvényben lévő) töltsük le az operációs rendszerünkhöz a megfelelőt illetve PHP 8-as verziót tartalmazzon.
* Telepítéskor lehet választani a szoftver csomag mely részei legyenek csak telepítve, ekkor a csak az Apache webservert pipáljuk be, másra nincs szükségünk.
* Ezután kérdezni fogja hol legyen a XAMPP telepítési helye, itt érdemes az operációs rendszerrel azonos lemezre telepíteni annak is főkönyvtárjába.  
  
* Ha telepítés végbement akkor, akkor lehet máris indítani a webszerververt. Ehhez megkell keresni a „xampp-control.exe” indító alkalmazást. Erre kattintva, jelenik meg a control panel ahol az Apache webservernek a start gombjára kell rá kattintani.   
  
* Ahhoz, hogy lefordítódjanak a PHP fájlok, el is kell őket helyezni egy megfelelő könyvtárban. Telepítés során ha az általam ajánlott könyvtárat vesszük akkor annak egy alkönyvtárába kell a tenni ami ezen az elérési úton van. C:\xampp\htdocs

Továbbá, ha **szerver számítógépről** szeretnék több felhasználónak elérhetővé tenni a programot akkor ellenőrizni kell rajta, van e már telepítve *PHP* fordító, ha nincs akkor érdemes a rendszer karbantartóját segítségül hívni.

**Kliens oldalon,** ha a felhasználó szeretné használni a programot, akkor csak egy böngészőre van szüksége, ez nem jelenthet nagyobb gondot, legtöbb operációs rendszerrel ellátott eszközön alapvetően van. Ha személy számítógépről történik a hostolás akkor az apache szervert indítva, a böngészőbe címsorába „localhost/szakdolgozat” -t kell begépelni. A „szakdolgozat” szót arra kell kicserélni amilyen a „C:\xampp\htdocs” alatt található alkönyvtár neve.

Ha szerver számítógépen van futtatva a szakdolgozati program, akkor egy URL link segítségével bármilyen webböngészővel ellátott eszközzel ellehet érni a programot, feltéve ha nincs jogosultsághoz kötve a szerver látogatása.

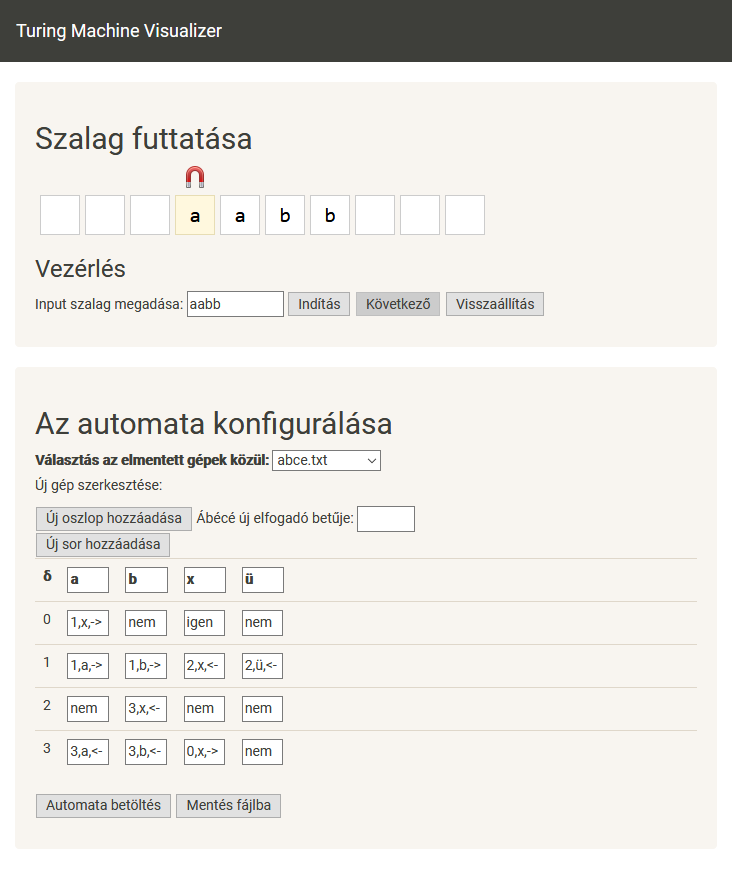
### Hordozhatóság

A program egyszerűen költöztethető. Ellenőrizni kell futtatható e az új környezetben PHP fájl illetve még milyen régi a fordító program. Ha ezek teljesülnek akkor program forrásfájlokat egyszerűen átlehet másolni egyik helyről a másikra. Semmilyen egyéb más be teendő nincs

## A program használata

### Turing-gép létrehozás

Személyi számitógépről hostolva meg kell bizonyosodni hogy fut-e a webszerver a háttérben, csak ezután indítható a program. Ha pedig webszerverről akarjuk elérni akkor bármilyen internetezésre alkalmas eszközzel, például: egy telefon vagy tabletnek a webböngészőjébe kell begépelnie megfelelő URL-t ami a programra mutat. Ez lehet localhost vagy lehet egy szokványos internetre mutató link. Ha megfelelő az URL akkor az alábbi [ÁBRA] ablakával találkozik. A program 2 fő részből áll. Kezdetben ez csak részben láthatók. További elemek a program futása során kerülnek csak elő.  
 Csak akkor indítható, ha a két komponens is készen áll. Ugyanis egyik komponens a másik komponens nélkül nem adnak elég adatot ahhoz, hogy gép el tudjon indulni. Első a szalag bediktálás, második a gép konfigurációjának megadása. Ha a felhasználó rátekint az oldal felépítésére, akkor azt látja, hogy a szalag vezérlő elem az oldal tetején helyezkedik el, míg a konfigurációs panel az oldalján. A két panel sorrendben való kitöltésére nincs szabály megadva, egyetlen lényeges szempont van, hogy minden beviteli mező ki kell töltve legyen.



### Turing-gép szerkesztő és futtató felület

Az itt látható egyik főkomponens [ÁBRA] segítségével lehet többek között megadni a szalagot, annak működését vezérelni, illetve animálva lehet látni az olvasó fejet ahogy olvassa a szalag betűit. A megadni kívánt input szalagot az (*Input szalag megadása* ) mezőbe bele kell kattintani egérrel, és be kell gépelni egy szót ami csak betűket tartalmaz, nem lehet ékezetes, nem lehet nagy betű és szám sem lehet. Az angol ábácé betűt fogadja csal el. Az (*Input szalag megadása* ) mező megfelelő kitöltés után lehet csak az indítás gombra kattintani. Ha a másik -fő komponens- konfiguráció is megfelelően van kitöltve akkor indul csak el az automata. Az adatok betáplálása és indítása után, a működő gépet nem lehet szerkeszteni csak ha végbement (*Terminált*). Lehet megállítani a működésben lévő automatát, lehet léptetni tekerni, újraindítani.



### Konfigurációs panel

Itt látható a program másik fő komponense [ÁBRA]. Ezen a kis felületen rengeteg funkció lett összeállítva. Elsőként ki lehet választani már előre elkészített példa konfigurációkat. Ezeket később átszerkesztve el lehet menteni bármikor újra lehet beinportálni  
Ha, a felhasználó nem szeretne régebbi beálltásokat alkalmazni akkor saját maga is létrehozhat egyet.

Ezt az alábbi táblázatban teheti meg. A táblázatot szabadon lehet szerkeszteni, bármelyik cellát felül lehet írni. Új sorokat és oszlopokat lehet hozzáadni. Új oszlopokat csak akkor lehet, ha kötelezően megadunk egy elfogadó betűt amit atáblázat fejlécébe helyeződik. Új sornál nem kell ilyesmit tenni, ott automatikusan generálódik egy új szám a táblázat első oszlopába, ez megegyezik az állapotok számával, figyelve a korábbiakat ne legyen duplikáció és növekvő sorrend legyen.

Ha a felhasználó törölni szeretne az oszlopok vagy a sorokból, akkor a egér segítségével sor esetén a sor elejére kell kattintani, oszlop esetén az oszlop fejlécére kell, ekkor mindkét esetben a sor vagy az oszlop törlésre kerül a táblázatból.

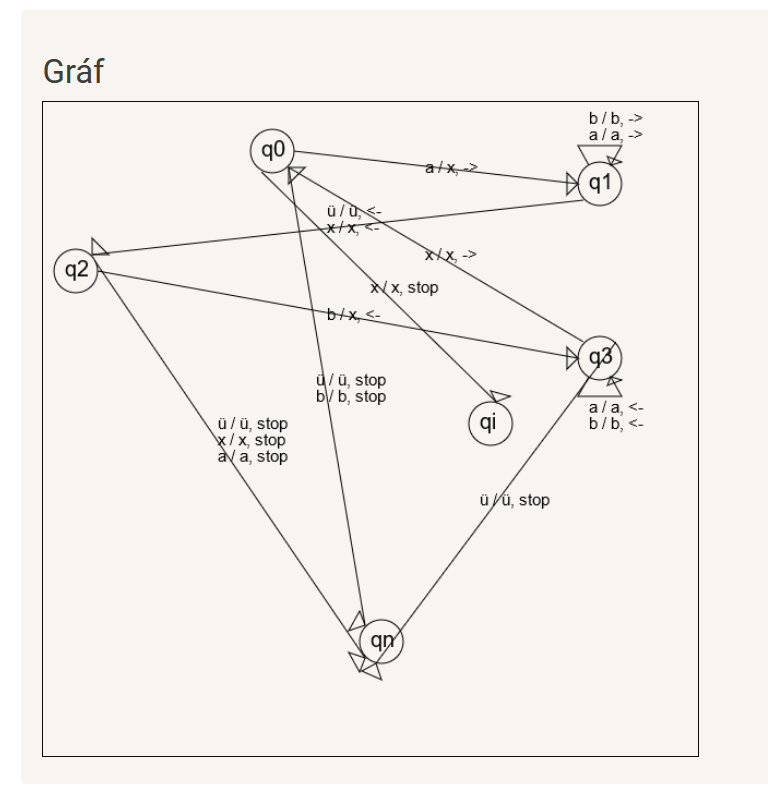
A kész konfigurációt szükség esetén elmenti is lehet egy fájlba (*Mentés fájlba*), amit később újra lehet be lehet importálni.

Mindezek után a Gépbe be lehet tölteni az adatokat az (Automata betöltése) gombbal.



* + 1. Gráf diagram

Az automata helyes betöltése után, egy új panel nyíilik meg a konfigurációs panel alatt. Itt kerülnek kirajzolásra a gráf diagram elemei, ahogy az ábra is mutatja [ÁBRA] Ezek leggyakrabban beállítástól függően, állapotokat és átmeneteket jelölnek. Az állapotok gömbök képében jelennek meg. Míg az átmenetek, vonallal, nyillal a végén, jelezve a haladási irányt, rajzolódik ki a gömbök között. Minden átmenet rendelkezik utasítással. Ezek szövögesen, külön külön az átmenetek fölé kerülnek kirajzolásra.



* + 1. Hibák és egyéb üzenetek

Adott esetben megtörténhet, hogy a felhasználó szembesül valamilyen hibajelenséggel. Túlságosan behatóan nem kell ismernie ezeket de nem árt néhány jelenséggel tisztában lennie.

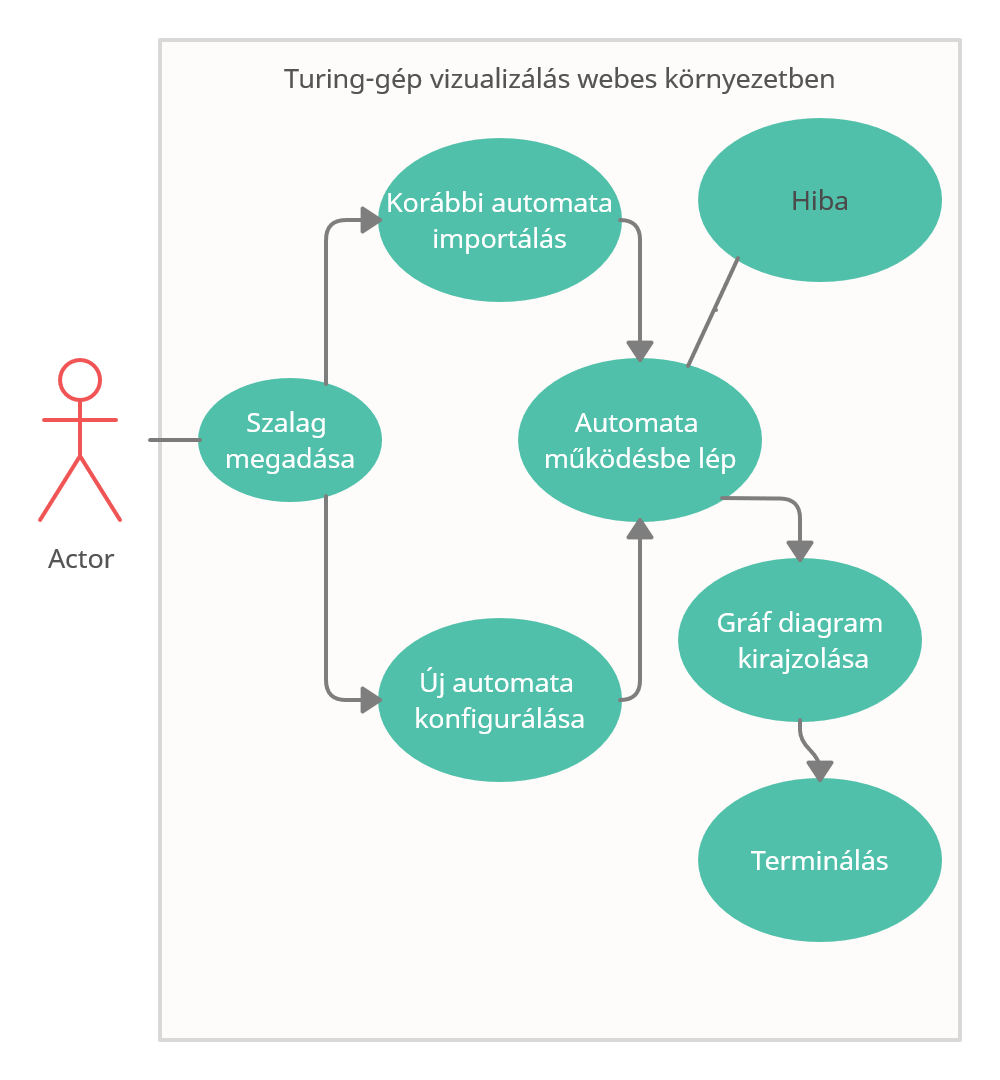
* *„Probléma merült fel az oldal betöltése közben”* Több dolog okozhatja ezt. Leginkább az internet elérésével lehetnek gondok. Ekkor érdemes ellenőrizni az internet kapcsolatot a eszközön amivel próbáljuk futtatni a programot.
* *„Hiba történt a fájlok importálása közben”* Megeshet hogy kész munkánk elveszhetnek, pléldául a vírusírtó által, vagy a program átmozgatása közben. Mindenesetre, a forrásfájlok mellett megtalálható egy mappa ahol az elmentett gépi kunfigurációk találhatók, minden indítás előt érdemes ellenőrizni ezt a könyvtárat.
* *„Futás közbeni hibaüzenetek”* Minden hiba, amit a program jelezhet, szöveges formában jelenik meg. Általában felugró szövegdoboz formájában. Ilyen hibák lehetnek: rosszul megadott paraméterek, nem megfelelően kitöltött szövegbevitelimezők. Érdemes ezeket újból alaposabban átvizsgálni mit rontottunk el.

# Fejlesztői dokumentáció

## Tervezés

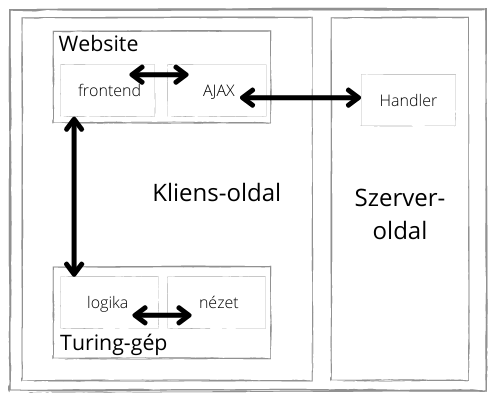
A tervezési fázis során

## Felhasználói esetek



## A rendszer architektúrája

Az eredeti elgondolásnak megfelelően, **kliens-szerver** architektúra szerint alakítottam ki a program szerkezetét. Kód mennyiségének tekintetében, a kliens oldalra fordítottam több erőforrást. Ezt eredetileg is így terveztem, inkább a felhasználó barátibb megjelenésre tettem a hangsúlyt.



A kliens a szerverrel HTTP protokollon keresztül kommunikálnak, AJAX segítségével. A szerver egyedül a fájlkezeléssel foglalkozik. Feladata beolvasni a fájlt, azt feldolgozni és tovább küldeni a kliens oldalra. Ha a kliens oldal elvégezte az adatok feldolgozását akkor, azt elküldi a szerver felé fájlba mentés céljából. A kliens oldalon történik a gép logikája, megjelenítés, és a felhasználó interakciók kezelése.

### Kliens oldal

* **Frontend:** a weboldal megjelenítésért felel. A felhasználó itt lép kapcsolatba a programmal. Minden felhasználói cselekvés itt kerül lekezelésre. Minden a fájlokkal kapcsolatos művelet, frontend oldalról az AJAX modul segítségével kerül a szerver oldalra.
* **Ajax modul:** kapcsolatot teremt a szerver és kliens oldalakkal. Nagy előnye, hogy frontend oldal frissítése a háttérben történik, nem tölti be újra az egész oldalt. Ettől ergonomikusabbá válik a felhasználó számára, másrészről egy jól használható technológia kifejezetten a két oldali kommunikáció lebonyolítására.
* **Turing-gép futtatása és konfigurálása:** a program egészéhez képest egy nagy részét fed le. Egy nézetből és egy logikai egységből áll. A belső logika állapota kerül megjelenítésre nézet által, illetve a nézet kezeli a felhasználó iterációit, amely a logikának kerül átadásra.

## Alkalmazott technológiák

Kliens oldal egy weboldal, melynek elnegedhetetlen programnyelve a HTML, de ettől még csak astatikus az oldal. Dinamikus használatához Javascript objektumorientált szkript programozási nyelvet használok. Logika és nézet használatához megvalósításához elengedhetetlen hogy a weboldal dinamikusan kövesse a felhasználói utasításokat. Ezen kívül, szebb megjelenés érdekében, használok Bootsrapt és CSSt, illetve a Javascpriten belül jQuery könytárat és AJAX modult használok.

Szerver oldalon PHP 7es verzióját használom. Ennek fordítása a hosztoló gépen történek és nem a kliens webbönégszőjében: Emiatt a hosztoló gépen kell lennie PHP 7 forditónak.

#### Klens oldali technológiák amiket használtam

* **HTML:** weboldal leíró nyelv, szabványosított. 5-ös, leújabb verzióját használtam, számos újítása van aminek köszönhetően kevesebb szkriptet kellett írnom.
* **CSS:** ugyancsak szabványosított nyelv. Weboldalak stílusát vele leírni. Programozási nyelvnek nem lehet mondani, inkább egy statikus kódolásnak lehet nevezni amit a W3C szabvány alapján lehet. A webböngésző fordítja le úgy ahogy HTML-t vagy JavaScriptet.
* **JavaScript:** objektumorientált szkript nyelv. Ez a nyelv nem szabványosított, legismertebb böngészők különböző fordítóprogramokat használnak rá. Én fejlesztés során a web böngészőnek Mozilla Firefoxot használtam ami SpiderMonkey fordító motort használ.
* **Bootstrap:** JavaScript és CSS keretrendszer. Segítségével az oldalt könnyebben stílusosabbá lehet alakítani. Legnagyobb előnye hogy reszponzívvá teszi a weboldal felületet, azaz követi a webböngésző méretét ami gyakran megegyezik a készülék méretével. Pl.: telefon, tablet kijelző
* **jQuery:** JavaScript keretrendszer, egyszerűbbé teszi a Selectorok használatát, CSS és DOM navigációt, AJAX kezelést teszi egyszerűbbé.
* **AJAX:** aszinkron interaktív programozási nyelv. JavaScriptet kiegészítésre alkalmas, arra ha a kliens és szerver kapcsolatát szeretnénk kivitelezni. Kis mennyiségű adatcserénél érdemes használni, mert az oldal teljes újra töltése nélkül lehet az oldal elemeit frissíteni.

#### Szerveroldali technológiák amiket használtam

* **PHP7:** Azért válaszottam ezt a tecnhológiát mert, az alapképzésen a törzs tárgyak között ezt tanították nekünk. A legismertebb szerveroldali szkriptnyelv, sajnos az idő kissé eljárt felette, más programozási nyelvek is alkalmasak már hasonló feladatok ellátására. Ami előnye, hogy a régi létezése okán nagy a support közösség, ezáltal, ha elakadtam egy kérdésben, könnyedén találtam rá megoldást az interneten.

## Alkalmazott módszerek

MV (Model-View) architektúra

Turing gép vezérlőnél használom

* A **model** tárolja az állapotot, ami hatással van a kapcsolodó műveletekre, algoritmusokra. Az állapotváltozás befolyásolja ezek működését.
* A **view** rajzolja ki model állapotát. Kezeli a felhasználói interakciókat és azokát átadja model számára feldolgozásra

## Modul és osztályszerkezet

## A Kliens-oldal osztályszerkezet

## A felhasználó felület terve

## Telepítés fejlesztői célokra

## Megvalósítás

# Tesztelés

# Irodalomjegyzék