

II. Megvalósított metódusok leírása

❖ **Project Package**

➤ **Cell:**

- `public Cell(int r, int c, String n)`: a CTOR-ja beállítja az egyes attribútumok értékeit
- `public int GetRow()`: visszaadja, hogy a cella melyik sorban van
- `public int GetColumn()`: visszaadja, hogy a cella melyik oszlopban van
- `public String GetName()`: Cella nevét adja vissza

➤ **ElectronHead, ElectronTail, Empty, Wire:**

- Mindegyik a Cell osztályból öröklődik, és egyedül egy konstruktoruk van, amely a paraméterben megadott értékek szerint létrehozza az egyes típusú osztályokat.

➤ **Main:**

- `public static void main(String[] args)`: A JVM (Java virtual machine) ezt a main metódust indítja el a program futásának elején.

➤ **ReadBack:**

- `public void readFiles(MyComboBox mcb)`: A paraméterként kapott JComboBox-ba deszerializálja az egyes fájlokat (amelyek 1-1 grid objektumot tartalmaznak) a megadott „SavedFiles” nevű mappából, amelyen iterátor segítségével végigmegy.

➤ **SaveProject:**

- `public SaveProject(WireMap wm, MyComboBox mcb)`: Szerializációt valósít meg, a paraméterben megadott WireMap objektumban található 2 dimenziós Cell tömböt írja ki a felhasználó által megadott nevű fájlba.
- `public String getFileName()`: Visszaadja az adott SaveProject objektumhoz tartozó fájl nevét, ahova a szerializáció megtörtént.

➤ **WireMap:**

- `public WireMap(int r, int c)`: a sorok, és az oszlopok számának megadásával beállítja a tagváltozóinak a méretét (grid, és tempgrid), majd meghívja az InitGrids() függvényt, amely a két tömb egyes értékeit allokalja (kezdetben mindegyik Empty típusú objektum)

❖ **GUI_Swing Package**

➤ **WireFrame:**

- `public WireFrame()`: A megjelenítendő Swinges elemek példányosítását itt kezelem, illetve elrendezését itt állítom be.
- `public void SetWmap(WireMap wm)`: Beállítja a wMap nevű változót, és annak méretével megegyező CellGui 2D-s tömböt hoz létre.
- `public int getNumOfRowsOfGrid()`: A CellGui grid sorainak számát adja vissza.
- `public int getNumOfColumnOfGrid()`: A CellGui grid oszlopainak számát adja vissza.
- `public int getSpeed()`: Gyorsaság értékét adja vissza (minél kisebb annál kevesebbet vár a Thread -> annál gyorsabb)
- `public String getChoosenButton()`: A kiválasztott nevét adja vissza.
- `public boolean GetPaused()`: Megvan-e állítva?
- `public WireMap getwMap()`: A WireFrame objektumhoz tartozó aktuális WireMap objektum referenciáját adja meg.
- `public void SetMiddleClicked(int r, int c)`: Az egér középső gombjának lenyomására beállítható a pozíciója, ahol lenyomtuk a gombot.
- `public void setSpeed(String str)`: A gyorsaságot lehet vele átállítani (lassú/közepes/gyors).

- `public void ChangeSize(String str)`: Ha a paraméter „Fullscreen”, akkor a Frame mérete kitölti a teljes képernyőt, ha „Windowed”, akkor a konstruktorban is beállított méretű lesz.
- `public void ChangeCard(String str)`: Az egyes JPanelek között lehet navigálni, amelyek Layoutja a CardLayout.
- `public void SetImportButton(boolean b, Color fg)`: Importbuttont módosítja, amely csak akkor csinál bármit, ha a JComboBox Main Modelljében az Import Project-en belül vagyunk.
- `public void actionPerformed(ActionEvent event)`: A fő panelon lévő gombok, interfészek működését figyeli, és automatikusan módosítja őket, amennyiben a felhasználó rákattint valamelyikre, vagy valamit módosít a programban.
- `public void keyPressed(KeyEvent event)`: Egy B opció megvalósításra szolgál, ugyanis az egyes kártyapanelek között lehet használni adott billentyűket is a navigáláshoz. Az egyes event-ek hatására változtatja a dolgokat.
- `public void UpdateColor(Cell[][] grid, boolean b)`: Amennyiben valamilyen változás történne, módosítja az egyes CellGui objektumok háttérének színét. Üres-fekete, vezeté-k-sárga, elektronfej-kék, elektron farok-piros. Mindig végigmegy a ciklus az egész 2D-s tömbön.

➤ CellGui:

- `public CellGui(int r, int c, JPanel cellButtonsPanel, WireMap wM, WireFrame wf, MyComboBox mc)`: Beállítja az egyes tagváltozók értékeit, illetve a cellák megjelenítéséért is felelős, példányosítja az egyes GUI osztály változóit. Hozzáadja a cellát a nagyobb panelhez, amely az egyes gombokat tartalmazza.
- `public void actionPerformed(ActionEvent e)`: Attól függően, hogy mi a choosenbutton string értéke, megváltoztatja az adott cellát, amikor rákattintunk.
- `public void SetChoosenButton(String cb)`, `public void SetMyCell(Cell c)`, `public Cell GetMyCell()`: Getter és Setter függvények, az egyes tagváltozók értékeire.
- `public void mouseClicked(MouseEvent event)`: Ha az adott cella üres, akkor beimportálható a cella pozíciójától kezdve az egyes előre elkészített áramköri elemek, attól függően, hogy melyik lett kiválasztva.

➤ MyComboBox:

- `public MyComboBox(WireFrame wf)`: Felépíti a JComboBoxok hierarchiáját, van egy fő(kész komponensek importálása, létező project-ek importálása), amely a sub értékeit aszerint állítja be, hogy melyik opció van jelenleg benne kiválasztva. Így lényegében az egymástól elkülönülő parancsokat szét lehet választani. A CTOR végén meghívja még a ReadBack osztály CTOR-ját, mert a program indításának az elején be kell olvasni a már elmentett fájlokat.
- `public void AddProject(String str, WireMap wmin)`: Új Project-et ad hozzá a programhoz. Az összes elmentett Project egy HashMap-ben van eltárolva, ahol a kulcs a fájlok nevei, az értékek pedig a beolvasott WireMap objektumok.
- `public JComboBox<String> getSubBox()`, `public String[] getMainStrings()`, `public WireMap getProjectName(String str)`, `public WireMap GetWmap()`: Getter függvények amely az egyes logikák szerint visszaadják az egyes tagváltozókat vagy visszatérnek egy válasszal.
- `public boolean IsContains(String str)`: Megmondja, hogy létezik-e az adott paraméterhez tartozó érték a HashMap-ben.
- `public void ReadBack()`: Program futása közben elmenti az adott Project-et, ilyenkor lefut a ReadBack osztály CTOR-a.

- `public void actionPerformed(ActionEvent event)`: A Main panel változtatásakor a Sub panelek közötti váltást valósítja meg, illetve figyeli, hogy éppen melyik item-et választotta ki a felhasználó.

➤ **MyMenu:**

- `public MyMenu(WireFrame f, MyComboBox mcb)`: A menüben lévő Swing-es elemek kezdeti kinézetét, funkcionalitását, helyét állítja be.
- `public void SetWmap(WireMap wm)`: A WireMap típusú belső változójának referenciáját módosítja.
- `public void actionPerformed(ActionEvent event)`: Ha a kilépés gombot megnyomjuk, mielőtt bezárulna a program, egy popup Window megkérdezi tőlünk, hogy biztosan kilépünk-e. Válaszlehetőségek: Igen (nincs mentés), igen és mentek is(van mentés), cancel (visszadob a főmenübe).
- `public void paintComponent(Graphics g)`: A háttérkép megjelenítésének céljából felülírt függvény.

➤ **OptionsPanel:**

- `public OptionsPanel(WireFrame wf)`: A beállítások panelen lévő Swing-es elemek kezdeti kinézetét, funkcionalitását, helyét állítja be.
- `public void actionPerformed(ActionEvent event)`: A program beállításait figyeli. SUBMIT gomb megnyomása szükséges, hogy véglegesítsük azt, hogy valamit átállítottunk (pl.: képernyő mérete).
- `public void paintComponent(Graphics g)`: Teljes mértékben megegyezik a MyMenu osztályban lévő függvénnyel.

❖ **Components package**

➤ **ComponentsClass:**

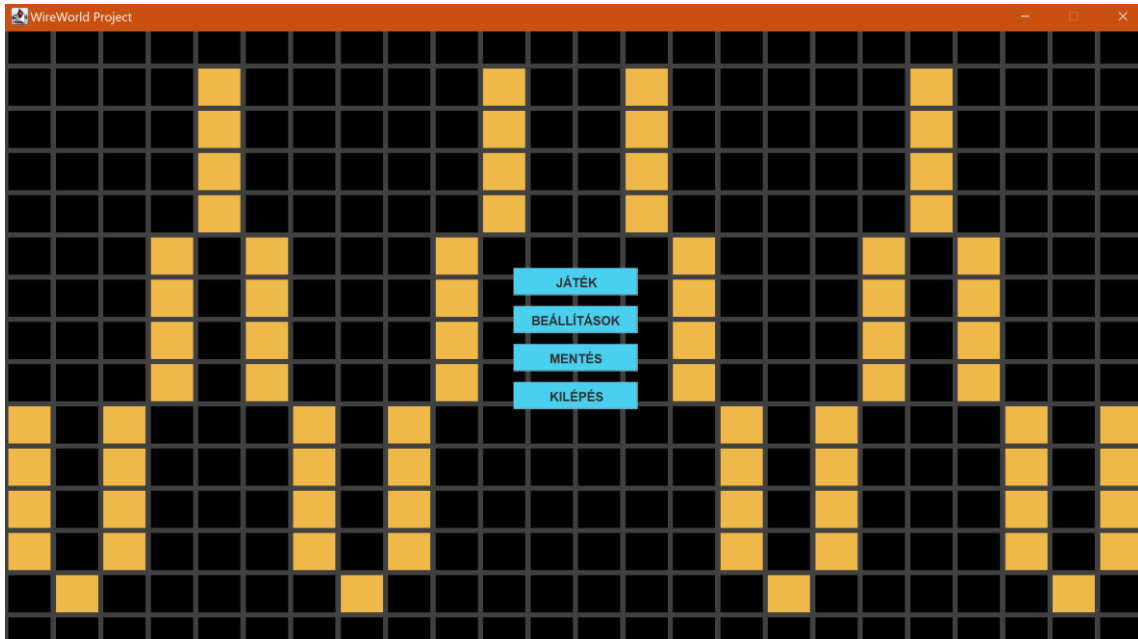
- `public ComponentsClass (int r, int c, WireMap wMap)`: Az egyes alosztályok hívják meg, majd a 3 tagváltozó értékét beállítja a paraméterként kapottára. (melyik WireMap-en melyik sor-oszlop pozíciótól kezdje a rajzolást)
- `abstract void Draw()`: Absztrakt függvény, az összes alosztálynak meg kell valósítani a definícióját.

➤ **ConductorDiode, NotConductorDiode, TripleWire, OrGate, XorGate, FlipFlop:** (Ezek az osztályok mind a ComponentsClass nevű absztrakt osztály egyes alosztályai.)

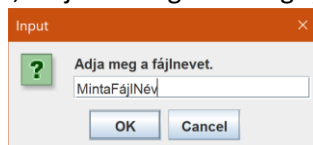
- `CTOR()`: Meghívja az ős CTOR-át; beállítja, hogy melyik WireMap-en melyik sor-oszlop pozíciótól kezdje a rajzolást.
- `Draw()`: Minden egyes osztály más és más áramköri elemet jelenít meg, ezért a Draw függvényt mindegyiknek implementálnia kell. A függvény lényegében az egyes Cellákat módosítja vezetékre.

III. Felhasználói kézikönyv

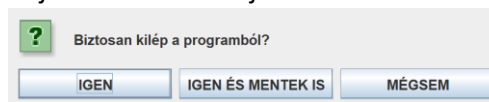
➤ Főmenü:



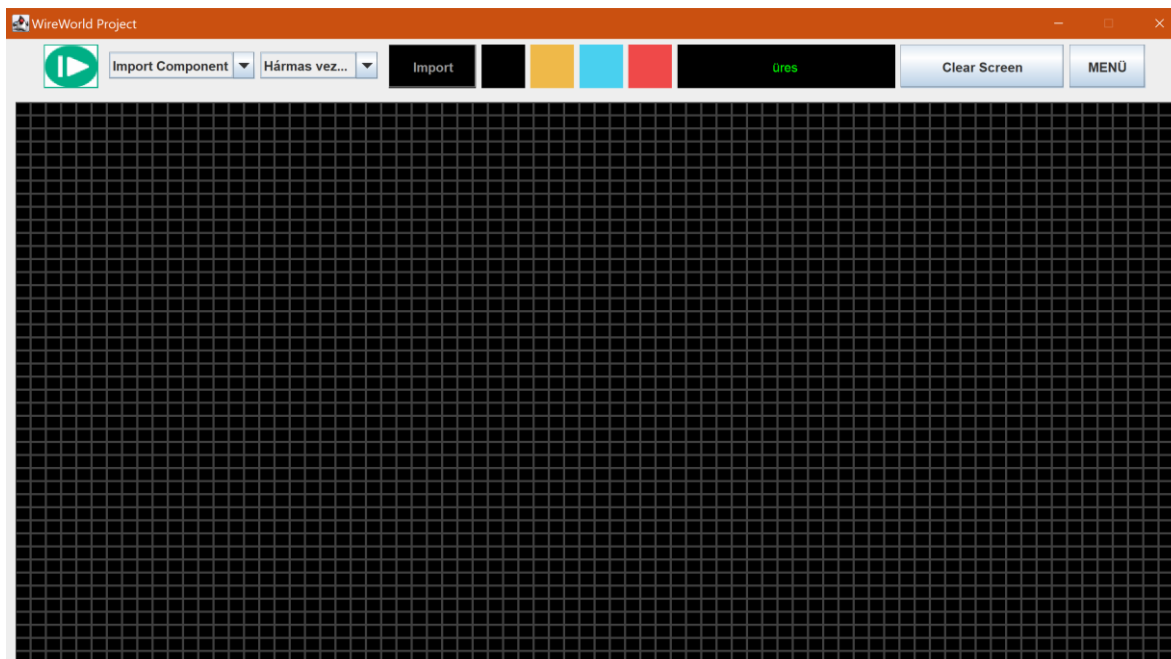
- A főmenüben 4 darab gomb található, az egyes gombok megnyomásával az alábbiakat tudjuk megtenni:
 - Játék: Játéktérbe való belépés.
 - Beállítások: A beállítások panel nyitható meg, ahol konfigurálni tudjuk a program beállításait.
 - Mentés: A Project-et, amin jelenleg dolgozunk, elmenti. Amennyiben menteni szeretnénk a felnyíló ablakban meg kell adnunk, hogy mi legyen a fájl neve, majd az OK gomb megnyomásával nyugtázzhatjuk(ábrán látható:).



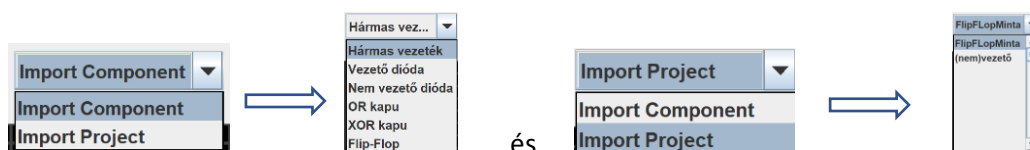
-
- Kilépés: kilépés a programból.(amennyiben itt a második opciót választjuk, még el tudjuk menteni a Project-et a fentiek szerint)




➤ JÁTÉKTÉR:



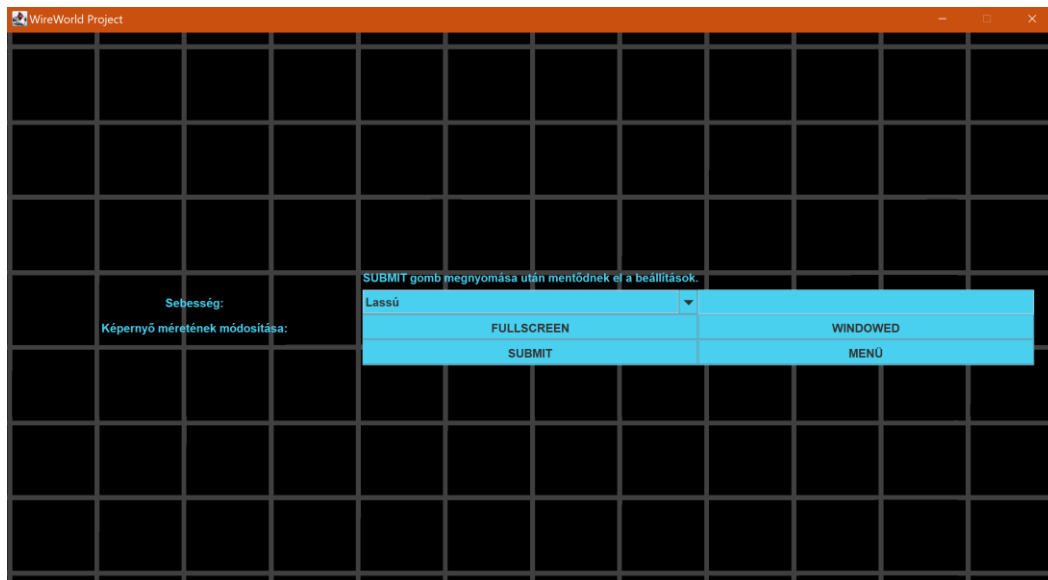
- A játéktér kettő fő részből áll: felső állapotosáv, és az alatta látható 2D-s pálya
- Felső állapotosáv: (a kezelhető funkciók balról jobbra)
 - Zöld Play-Stop gomb megnyomásával megállítható, illetve elindítható a wireworld működése. Ha megállítjuk a folyamatot, akkor a táblán lévő elektronfejek, és elektron farkak megállnak.
 - Import Component: Itt lehet kiválasztani a legördülő menüpontok közül, amennyiben egy kész elektronikai komponenst vagy egy elmentett Project-et szeretnénk beimportálni a pályánkra.
 - Az előbbieken kiválasztott opciók alapján a következő legördülő menüben kiválasztható, hogy melyik komponenst is szeretnénk pontosan beilleszteni, vagy melyik Project-t szeretnénk beimportálni (Ilyenkor vigyázni kell, mert a jelenlegi felül fog íródni az újra!)



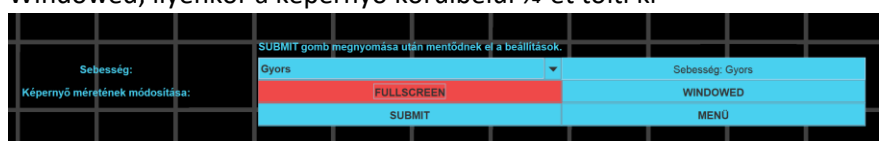
- Import gomb: Csak akkor használható, ha az Import Project menüben vagyunk, és kiválasztottunk egy általunk beimportálandó Project-et. Ilyenkor az Import szöveg zöld színű, ellenkező esetben sötét szürke.
- A 4 darab színes gomb rendre:
 - 1. fekete – üres mező rakható le a megnyomása után
 - 2. sárga – vezeték mező rakható le a megnyomása után
 - 3. kék – elektronfej mező rakható le a megnyomása után
 - 4. piros – elektron fark mező rakható le a megnyomása után

-  elektronfej
 - Közvetlenül mellettük az aktuálisan kiválasztott nevét látjuk zöld színnel.
- Clear Screen: a gomb megnyomása után az adott pálya törlődik, természetesen előtte a program megkérdezi, hogy biztosan clear-elni akarjuk-e.
- Menü: Megnyomásával visszavigálunk a főmenübe.
- 2D-s pálya:
 - A pálya egyes elemei szerkeszthetőek. Megnyomásukkor az aktuálisan kiválasztott színes gombnak megfelelő értékre fog változni az az elem, amelyet megnyomtunk.
 - Miután kiválasztottuk az egyes áramköri komponenst, a táblára úgy tudjuk felvinni, hogy megnyomjuk valamelyik egységen az egér görgőjét. Fontos, hogy ez lesz a berajzolódó komponens legeleje, és ettől a ponttól jobbra fog kirajzolódni.

➤ Beállítások:



- 2 dolgot lehet itt beállítani, amelyek csak a SUBMIT gomb megnyomásakor állítódnak át ténylegesen:
 - A program működésének sebességét, amely lehet:
 - Lassú (például megfigyeléshez ideális)
 - Közepes
 - Gyors
 - A program ablakának méretét (az éppen kiválasztott gomb háttere pirossá válik), amely lehet:
 - FullScreen, ilyenkor a teljes képernyőt kitölti
 - Windowed, ilyenkor a képernyő körülbelül ¾-ét tölti ki



- Amennyiben végeztünk a módosításokkal, a SUBMIT gombbal elkönnyelhetjük őket, majd a MENÜ gombbal vagy ESC-el visszavigálunk a Főmenübe.