

## Proramozási technológia 2. beadandó

# IMPOSSIBLE MISSION

powered by Anonim

## Követelmény & specifikáció: Anonim (5. csoport)

### A program célja

Feladatként kaptunk egy régi Commodore64 játék restaurációját **Java** nyelvi elemek használatával. Az alkalmazás során lehetőségünk van egy titkos ügynök szerepébe bújni és megoldani egy küldetéssorozatot, amelynek végén egy kódot kell megfejtsünk.

### Ismerető

A küldetésünk lényege, hogy a föld alatti labirintusba behatolva megszerezzünk egy **17 betűs jelszót**, mellyel leállíthatjuk a *professzor* számítógépes rendszerét. A laboratóriumban viszont nem lesz "felhőtlen" a keresgélés, a professzor **robotjai** próbálják elheszegetni (elég jó hatásfokkal) a betolakodókat. A laboratóriumban minimum 5-6 szoba van, melyeket **liftek**, **folyosók** kötnek össze. Egy-egy betű megfejtéséhez **2-2 puzzle-darabot** kell úgy összeraknunk, hogy azok egymást lefedve egy majdnem teljesen fehér téglalapot képezzenek. Ezek a puzzle-ok különféle bútorok, irodagépek mögött vannak elrejtve. Mivel egy betűt négy darab kirakásával kapunk meg, így összesen **17-szer 2**, vagyis **34 puzzle-darab** van elrejtve. Természetesen sok olyan bútor is van, amely mögött **nincs semmi** sem elrejtve, ezt azonban nem tudhatjuk meg, amíg át nem kutatjuk. Egy-egy **bútor** átkutatása több-kevesebb időt vesz igénybe. Ezt a kutatást úgy tudjuk végrehajtani, hogy a vezérlővel a tárgy elé állunk, és feléje fordulunk. A megjelenő (*KEEP CALM*) csík hosszának csökkenése jelzi az idő múlását, amit a kutatásra fordítottunk. Ezután vagy kapunk egy puzzle darabot vagy a **NEXT TIME** felirat fog felvillanni. A 17 betűs jelszó megtalálására 60/30/15 percünk van, nehézségi szinttől függően. Ha hibázunk, a gép időt vesz el tőlünk. Egy élet 1 percünkbe kerül. A szolgáltatásokért is idővel kell fizetnünk. Segítségünkre van egy speciális kisszámítógép, melyet a kép alsó harmadában láthatunk, ha a liftet a szobákkal összekötő folyosón állunk. Három jól elkülönített részt különböztethetünk meg: A középső rész az eddig felderített területet térképezi fel. A már bejárt szobák és az őket összekötő folyosó látható ezen a térképen. A játék kezdetén ez majdnem teljesen üres. A villogó pont a pillanatnyi helyzetünket adja meg. Ha sikerült a **két puzzle**-darab összeválogatása, de a gép mégsem fogadja el, azért van, mert a megfejtésük nem áll a helyes irányban. Ha beforgatjuk, a puzzle el fog tűnni mind a munkaasztalról, mint a tárolóból (2 db). A megfejtett betű pedig kiírásra fog kerülni a megfelelő helyen. A játék addig megy amíg a jelszót ki nem rakjuk, vagy az idő le nem jár.

### Követelmények

Az program elkészítése során a **JDK**-t használjuk egy általunk választott **IDE**-n keresztül. A megadott követelmények szerint a játéknak rendelkeznie kell az alábbi lehetőségekkel:

- stopper: az idő mérésére alkalmas funkció, amelynek bármely játékmódban láthatónak kell lennie
- multiplayer mód: IP cím alapján csatlakozhatnak be a játékba hálózaton
- mouse control: a puzzle darabok megfejtése lesz az elsődleges funkciója
- **átszaltózás** special move:
- map: bármely szobában meghívható a játék során adott billentyűkombinációra, valamint elrejthető
- ellenfelek: többféle robot, amelyek különböző mértékben megsebzik a játékost, ezáltal csökkentve a játékidőt
- lift: a játék során a különböző szobákat összekötő helység
- bútorok eltűnése **KEEP CALM** után
- a futtatás után a játék betöltése automatikus, ami után az első választási lehetőség a **Menu Option**
- a játék végén automatikus **override highscore**, ami egy külön adatbázisban lesz tárolva
- a játék adott pillanatában lehet **save** opciót választani, ami kiimportálja a teljesített játékrészletet egy külön file-ba
- pálya: minimum 6 különböző pályát tervezünk statikusra, az idő függvényében dinamikussá alakítjuk
- A játék végén ki kell rakni egy 17 karakter hosszúságú kódot, amely megfejtése után nem robban fel az atombomba és megmenekül a világ. Rossz kód beírása esetén a játékidőből egységnyi levonódás történik. A játékelmény kedvéért a megfejtést nem írjuk bele a dokumentációba.

### További fejlesztések (ido szerint, nem garantáljuk)


- Az alábbi `package` segítségével próbáljuk felidézni a régi játékok sípoló processzor hangját:

```
import javax.sound.sampled*;
```

- Easter eggs: a játék során különböző rejtett sprite-k fejlesztünk, amelyeknek valamilyen popkulturális jelentései vannak
- robotok lelövése, fagyasztása

- Cheat kódok **IDDQD**

### Funkciók

- single player: egy játékos módban a vezérlés  irányba történik W/A/S/D gombok segítségével.
- multiplayer: a team úgy döntött, hogy a hálózaton lévő játékosok **ne** egymás ellen, hanem egymás mellett az **időtől** behatárolva játszanak
- stopper: egyben a nehézségi szint is, 60/30/15 perces nehézségi szint, a visszaszámláló mindig látszódik a játék jobb felső sarkában, adott büntetések után adott másodperc levonódik
- egérrel vezérelhető: az alábbi `package`-k importálásával valósítjuk meg

```
import java.awt.MouseInfo;
import java.awt.GridLayout;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.event.MouseListener;
import java.awt.event.MouseEvent;
```

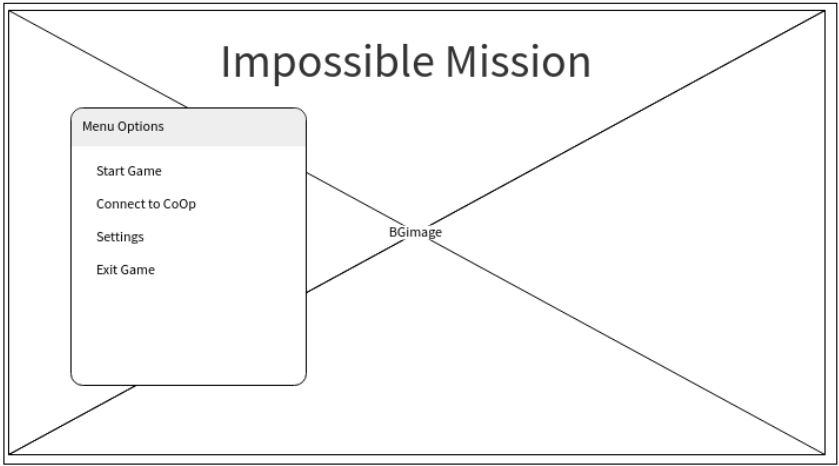
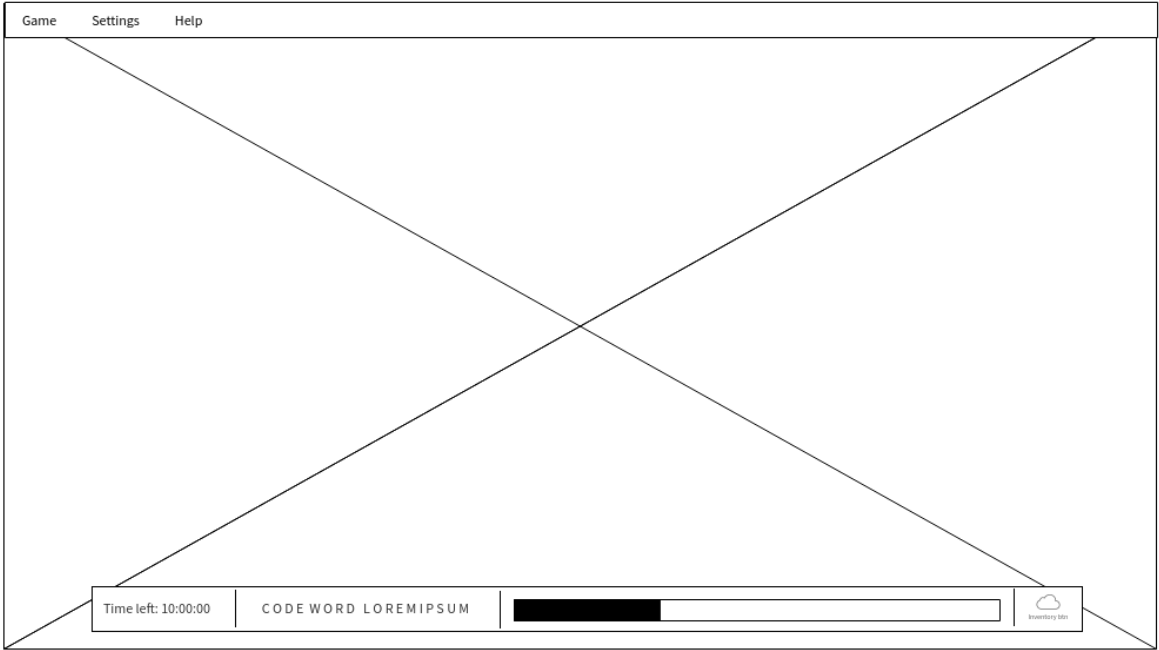
### Verziókövető felület

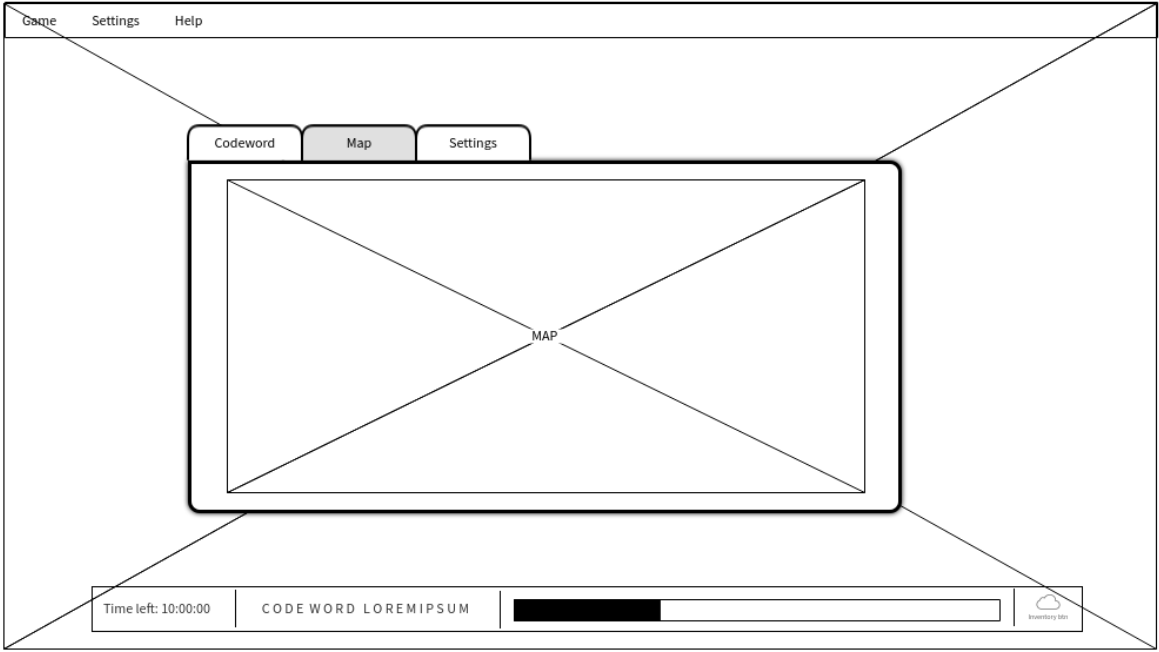
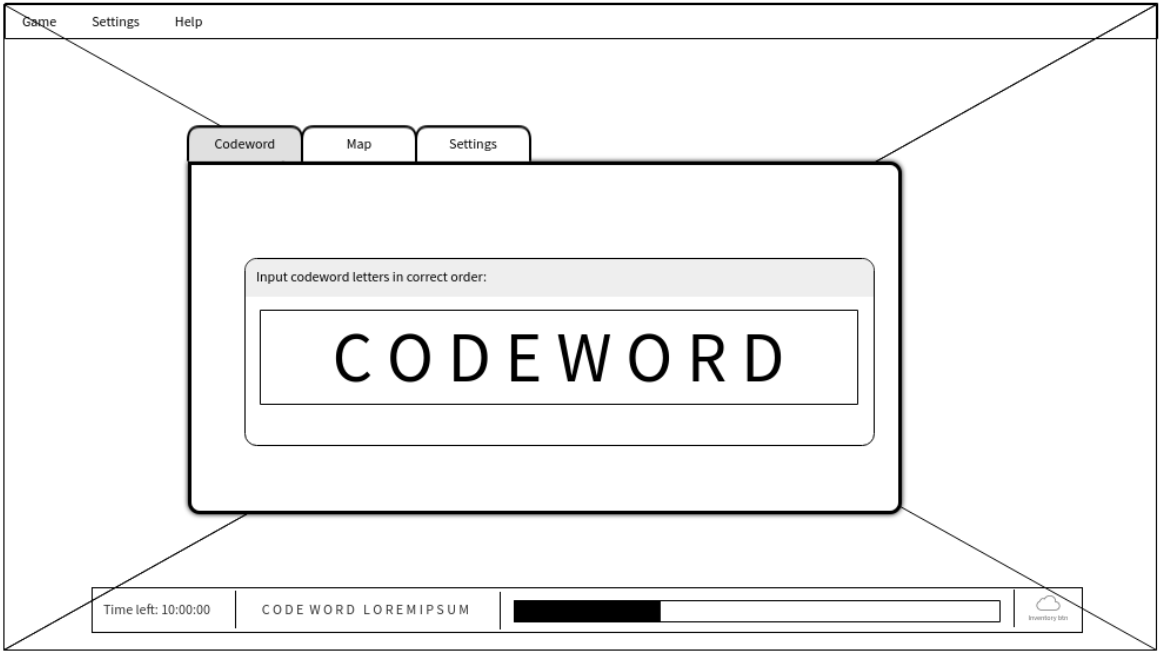
- A követelményhez hozzátartozik a **GitLab** `tool` használata.



- Az előadó által létrehozott felület mellett használt eszközök a fejlesztés során:

Wireframe







User Story

AS A		USER
I WANT TO		choose mode
1	GIVEN	Player User
	WHEN	click
	THEN	Player User mode chosen
2	GIVEN	Inventory User
	WHEN	click
	THEN	Inventory User mode chosen
3	GIVEN	Menu User
	WHEN	click
	THEN	Menu User mode chosen

További user story beimportálva:

AS A		Player User
I WANT TO		Movement
1	GIVEN	Out of Room
	WHEN	enter Door
	THEN	Door -> Denied or Allowed
2	GIVEN	Into Room
	WHEN	move
	THEN	cases -> Injury or Prop Interaction
I WANT TO		Out of Room
1	GIVEN	Denied
	WHEN	open
	THEN	repeat
2	GIVEN	Allowed
	WHEN	open
	THEN	win
I WANT TO		Into Room
1	GIVEN	Injury
	WHEN	connect robot
	THEN	Time Penalty -> Incorrect
2	GIVEN	Prop Interaction
	WHEN	open
	THEN	Get Letter (Allowed) or No Letter

AS A		Inventory User
I WANT TO		Open Inventory
1	GIVEN	Input Code
	WHEN	enter Code
	THEN	Code -> Correct or Incorrect (Time Penalty)
2	GIVEN	Open Map
	WHEN	spectator
	THEN	close Map
3	GIVEN	Open Settings
	WHEN	choose
	THEN	Change Settings

AS A		Menu User
I WANT TO		Start Game
1	GIVEN	Choose Difficulty
	WHEN	enter Difficulty
	THEN	Game -> Movement
I WANT TO		Start CoOp
1	GIVEN	Start Game
	WHEN	set Game
	THEN	Start Game
I WANT TO		Open Settings
1	GIVEN	Change Settings
	WHEN	choose
	THEN	Change Settings
I WANT TO		Exit Game
1	GIVEN	nothing
	WHEN	ending Scene
	THEN	Override the Highscore

code

```
/** * Ez csak a következő beadandóban fog kelleni. */
public class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

Longplay

Ez sajnos nem fog működni miután pdf-be konvertáljuk a html vagy md file-t.



-----