ŽILINSKÁ UNIVERZITA V ŽILINE
Fakulta riadenia
a informatiky

Semestrálna práca z predmetu

vývoj aplikácií pre mobilné zariadenia

BLUDISKO

Vypracoval: Peter Fabo

Študijná skupina: 5ZIF11

Akademický rok: 2024/25

V Žiline dňa 5.6.2025

Obsah

Úvod

Ako aplikáciu som sa rozhodol spraviť hru na báze hľadania cesty cez bludisko a hlavnou úlohou bude nájsť cieľ/východ z bludiska.

Prehľad podobných aplikácií

Existuje mnoho aplikácií s týmto žánrom, napríklad:

- Maze: Puzzle and Relaxing Game
- Maze Go
- Bludiště: Labyrinth Games

Tieto hry majú podobné prvky, ako sú rôzne typy bludísk, herné postavy a mechanizmy dosiahnutia cieľa. Taktiež disponujú bodovým systémom na hodnotenie výkonu hráča.

Analýza navrhnutej aplikácie

Navrhovaná aplikácia obsahuje nasledujúce funkcionality:

- Hranie hry: hráč bude navigovať postavu cez bludisko a hľadať východ.
- Generovanie náhodných bludísk s rôznou úrovňou obťažnosti.
- Na začiatku si hráč zadá/vyberie svoje meno a vyberie veľkosť bludiska, v ktorom chce hrať.
- Bludisko sa vygeneruje a hráč sa bude pohybovať pomocou štyroch tlačidiel vedľa/pod bludiskom.
- Cieľom bude nájsť cestu ku cieľu.
- Pri dosiahnutí cieľa sa otvorí okno o štatistikách, priráta skóre do profilu a hra sa pregeneruje na nové bludisko.
- Možnosť prepnutia na iného hráča s uloženými údajmi o skóre.

Architektúra aplikácie

Táto časť dokumentácie vysvetľuje vnútornú štruktúru aplikácie, jej hlavné komponenty a spôsob správy dát.

Databázová časť

Aplikácia používa Room databázu, ktorá pracuje nad SQLite a slúži na ukladanie dôležitých dát hry.

Účel:

- Uchovávanie profilov hráčov (meno, skóre, počet hier).
- Ukladanie uložených hier – stav mapy, pozície hráča a cieľa.

Entity:

Item – hráčsky profil (tabuľka profils)

- Meno, skóre a počet odohraných hier.

compMazes – uložená mapa (tabuľka mazes)

- Rozmery mapy, pozícia hráča a cieľa, skóre a uložené rozloženie mapy.

Týmto spôsobom si aplikácia dokáže pamätať profil hráča aj jeho uloženú hru a umožniť jej pokračovanie alebo opätovné načítanie.

Po spustení aplikácie sa v triede MyApplication vytvorí kontajner s repozitármi. V MainActivity sa pomocou tohto kontajnera vytvoria ViewModely (ItemViewModel, MazeViewModel), ktoré spravujú dáta pre UI. Tieto ViewModely tak majú okamžitý prístup k databáze cez repozitáre.

NavHost

NavHost je centrálny komponent, ktorý zabezpečuje prechod medzi rôznymi obrazovkami v aplikácii. Každá obrazovka, na ktorú sa používateľ dostane, dostáva referencie na ViewModely – špeciálne triedy, ktoré uchovávajú a spravujú dáta potrebné pre danú obrazovku. Vďaka tomu sú všetky obrazovky schopné pristupovať k aktuálnym údajom a reagovať na ich zmeny.

Pri spustení aplikácie sa v NavHoste tiež inicializuje SoundManager, ktorý spravuje prehrávanie zvukov v rámci aplikácie. To zabezpečuje, že zvukové efekty sú pripravené hneď od začiatku.

Ďalšou dôležitou funkciou NavHostu je sledovanie orientácie zariadenia (či je obrazovka na výšku alebo na šírku). Táto kontrola prebieha asynchrónne a pri zmene orientácie NavHost automaticky presmeruje používateľa na správnu verziu obrazovky prispôsobenú danej orientácii.

Okrem toho NavHost sleduje aj aktuálny stav aplikácie, nazývaný stage, ktorý určuje, či sa aplikácia nachádza v režime konfigurácie (napríklad výber profilu) alebo priamo v hre. Na základe tohto stavu NavHost zabezpečuje, že používateľ vždy vidí správnu obrazovku pre aktuálny režim aplikácie.

Správa spätného tlačidla – ConfirmExitOnBackHandler

Pre lepšie používateľské ovládanie aplikácie je implementovaný mechanizmus, ktorý zachytáva stlačenie tlačidla späť. Namiesto okamžitého ukončenia aplikácie sa zobrazí dialógové okno s otázkou, či chce používateľ naozaj ukončiť aplikáciu.

Používateľ má na výber:

- **Áno –** aplikácia sa ukončí a zároveň sa uvoľnia všetky zvukové zdroje, čím sa zastavia prehrávané zvuky.
- **Nie –** dialóg sa zatvorí a aplikácia pokračuje v prevádzke, pričom zvuky bežia ďalej.

Tento prístup zabraňuje náhodnému alebo neúmyselnému ukončeniu hry či nastavení a zvyšuje komfort používania.

ViewModely:

Slúžia na uchovávanie a správu dát počas navigácie medzi screenami. Umožňujú zdieľať stav a informácie medzi rôznymi obrazovkami aplikácie bez ich straty pri zmene UI komponentov. Tým zabezpečujú plynulý a konzistentný tok dát.

MazeInfoViewModel

ViewModel MazeInfoViewModel slúži na správu a uchovávanie stavu bludiska počas behu aplikácie. Obsahuje interný stav objektu MazeInfo, ktorý drží základné parametre bludiska, ako sú jeho rozmery (x, y), pozície hráča a cieľa (playerX, playerY, finishX, finishY), aktuálne skóre, či sú údaje uložené (zapísané), a nastavenia zvuku (sounds, soundPermision).

MazeInfoViewModel poskytuje funkcie na selektívnu aktualizáciu rôznych častí MazeInfo:

- updateALLMazeInfo nahradí celý stav novým objektom.
- updateMazeInfo aktualizuje rozmery bludiska.
- updateMazeFinish nastavuje pozíciu cieľa.
- updateMazePlayer nastavuje pozíciu hráča.
- updateMazeSkore aktualizuje skóre.
- updateMazeWrite a setZapisane spravujú stav uloženia bludiska.
- setSounds a allowSounds spravujú nastavenia zvuku a povolenia prehrávania.

Vďaka tomu ViewModel slúži ako centrálny zdroj pravdy pre všetky dáta súvisiace s bludiskom, ktoré sú potrebné medzi rôznymi obrazovkami a počas celej životnosti aplikácie, pričom zabezpečuje reaktívnu aktualizáciu UI podľa zmien stavu.

PlayerViewModel

PlayerViewModel slúži na správu a uchovávanie informácií o aktuálne vybranom hráčovi počas behu aplikácie. Obsahuje interný stav objektu Player, ktorý reprezentuje základné údaje hráča, ako sú jeho ID, meno, skóre a počet odohraných hier.
PlayerViewModel poskytuje funkciu updatePlayer na aktualizáciu aktuálneho hráča.

Vďaka tomu ViewModel slúži ako centrálny zdroj pravdy pre dáta hráča, ktoré sú potrebné medzi rôznymi obrazovkami a počas celej životnosti aplikácie, pričom zabezpečuje reaktívnu aktualizáciu používateľského rozhrania podľa zmien stavu.

GameViewModel

GameViewModel slúži na správu a uchovávanie stavu aktuálneho bludiska počas behu aplikácie. Obsahuje interný stav objektu Maze, ktorý reprezentuje samotné bludisko so svojimi rozmermi a štruktúrou buniek (Cell). Rozmery bludiska sú uložené ako premenné x a y a môžu byť aktualizované.

- GameViewModel poskytuje viacero funkcií na prácu s bludiskom:
- setAndResetMaze umožňuje nastaviť nové rozmery bludiska a zároveň ho resetovať na nový generovaný stav.
- updateSize mení rozmery bludiska, pričom zabezpečuje minimálne rozmery 2x2.
- updateMaze umožňuje priamo nastaviť nové bludisko.
- resetMaze generuje nové validné bludisko pomocou viacerých funkcií na tvorbu bludiska, umiestnenie hráča a cieľa, zatiaľ čo overuje jeho správnosť.
- saveMaze a loadMaze umožňujú uložiť a načítať stav bludiska vrátane ďalších údajov pomocou externých metód na kompresiu a dekompresiu dát.

Týmto spôsobom GameViewModel slúži ako centrálny správca logiky bludiska, ktorý udržuje jeho aktuálny stav, umožňuje jeho generovanie, ukladanie a načítanie počas celej životnosti aplikácie, pričom zabezpečuje reaktívnu aktualizáciu používateľského rozhrania podľa zmien.

ScreenViewModel

ScreenViewModel sa používa na určenie, ktoré štádium obrazovky sa má zobraziť – či konfigurácia alebo samotná hra. Uchováva interný stav vo forme celočíselnej hodnoty (Int), ktorá reprezentuje aktuálne štádium aplikácie. Tento stav potom jednotlivé prvky používateľského rozhrania čítajú a podľa neho sa automaticky aktualizujú a menia svoje správanie či zobrazenie.
ScreenViewModel poskytuje funkciu setStage na nastavenie alebo zmenu tohto štádia.

Dátové modely:

Maze

Dátová trieda Maze reprezentuje celé bludisko. Obsahuje:

- width: šírku bludiska (počet stĺpcov),
- height: výšku bludiska (počet riadkov),
- maze: dvojrozmerné pole objektov Cell, ktoré predstavujú jednotlivé bunky bludiska.

Trieda má aj **sekundárny konštruktor**, ktorý umožňuje vytvoriť prázdne bludisko so zadanými rozmermi, kde každá bunka je inicializovaná ako nová Cell.

Slúži ako hlavný kontajner pre logiku pohybu, generovanie bludiska a určovanie cieľov a pozícií hráča.

Cell

Trieda Cell reprezentuje jednu bunku v bludisku a obsahuje nasledujúce vlastnosti:

- visited: označuje, či bola bunka navštívená,
- top, bottom, left, right: určujú prítomnosť stien okolo bunky,
- finish: označuje, či je táto bunka cieľová,
- flood: číselná hodnota pre algoritmus zaplavovania na výpočet pozície hráča/zobrazenie hintu,
- player: označuje, či sa hráč práve nachádza v tejto bunke,
- hint: pomocná hodnota na zobrazovanie nápovedy alebo trasy.

Tento model definuje logickú štruktúru bludiska na úrovni jednotlivých políčok.

MazeInfo

Dátová trieda MazeInfo uchováva **stav a metadáta** bludiska. Obsahuje:

- x, y: rozmery bludiska,
- finishX, finishY: súradnice cieľa,
- playerX, playerY: súradnice hráča,
- skorenow: aktuálne skóre (počet zostávajúcich krokov),
- zapisane: označuje, či bol stav bludiska uložený,
- sounds: či je zvuk zapnutý,
- soundPermision: či sú povolené zvukové efekty zo systému.

Slúži ako zdroj stavu, ktorý je zdieľaný medzi obrazovkami a komponentmi, najmä prostredníctvom MazeInfoViewModel.

Player

Dátová trieda Player uchováva informácie o hráčovi:

- id: identifikátor hráča (napr. z databázy),
- name: meno hráča,
- skore: skóre hráča (napr. najlepší výsledok),
- games: počet odohraných hier.

Trieda sa používa najmä na správu profilov hráčov a na zaznamenávanie ich pokroku a výsledkov počas hry.

Screen: StartScreenVert1

Tento screen slúži na zobrazenie a výber hráča zo zoznamu hráčov, ktorí sú uložený v databáze. Dáta sa načítavajú zo ItemViewModel a po výbere hráča sa vybraný hráč uloží do PlayerViewModel ako objekt typu Player.

Okrem výberu hráča je možné vygenerovať nového hráča jednoduchým zadaním mena. Program automaticky priradí novému hráčovi ďalšie voľné ID v databáze (začínajúc od 0).

Tiež je možné vymazať hráča podľa jeho ID.

Všetky tieto akcie (pridanie a odstránenie hráča) sú riešené cez dialógy, ktoré umožňujú bezpečné zadanie údajov a potvrdenie operácie.

V tomto screene sa používa ConfirmExitOnBackHandler, ktorý slúži na potvrdenie ukončenia aplikácie pri stlačení tlačidla späť. Používateľ tak má možnosť zrušiť náhodné zatvorenie aplikácie a vybrať si, či naozaj chce aplikáciu opustiť.

Screen: StartScreenVert2

Tento screen zobrazuje informácie o aktuálne vybranom hráčovi načítanom z PlayerViewModel. Používateľ si môže vybrať veľkosť bludiska z niekoľkých preddefinovaných možností pomocou tlačidiel, ktoré zvýrazňujú aktuálny výber. Veľkosti bludiska sú definované v kóde cez premenné small = 5, medium = 25, large = 50 a huge = 100.

Pre zmenu týchto veľkostí je potrebné upraviť aj príslušné názvy v resources (string.xml), aby korešpondovali s novými hodnotami.

Po výbere veľkosti sa táto hodnota uloží do MazeInfoViewModel. Navigácia umožňuje vrátiť sa späť na domovskú obrazovku alebo pokračovať do hry so zvolenými nastaveniami.

Všetky interakcie sú spravované cez stav vybraného tlačidla a volania navigačných udalostí. UI je postavené na Jetpack Compose komponentoch s dôrazom na jednoduché a prehľadné ovládanie.

Screen: StartScreenHorz

StartScreenHorz slúži ako horizontálne orientovaná úvodná obrazovka, ktorá kombinuje funkcionalitu obrazoviek StartScreen1 a StartScreen2. Umožňuje používateľovi zvoliť si hráča, nastaviť veľkosť bludiska a spustiť hru – všetko v jednom rozhraní usporiadanom vedľa seba.
Táto obrazovka zobrazuje informácie o aktuálne vybranom hráčovi, načítanom z PlayerViewModel. Používateľ má možnosť vybrať si veľkosť bludiska z niekoľkých preddefinovaných možností pomocou tlačidiel, ktoré vizuálne zvýrazňujú aktuálny výber. Veľkosti sú definované ako:
small = 5
medium = 25
large = 50
huge = 100
Pre úpravu týchto veľkostí je potrebné zmeniť aj príslušné názvy v súbore strings.xml, aby korešpondovali s novými hodnotami.
Po výbere sa zvolená veľkosť uloží do MazeInfoViewModel, čím sa zabezpečí jej ďalšia dostupnosť počas navigácie. Pri stlačení tlačidla „Start“ sa aplikácia presunie na obrazovku s hrou, pričom je automaticky aktualizovaný PlayerViewModel a ScreenViewModel.

Všetky používateľské interakcie sú riadené stavom vybraného tlačidla a spracované pomocou Jetpack Compose komponentov, s dôrazom na jednoduchosť, prehľadnosť a optimalizáciu pre širšie obrazovky.

V tomto screene sa používa ConfirmExitOnBackHandler, ktorý slúži na potvrdenie ukončenia aplikácie pri stlačení tlačidla späť. Používateľ tak má možnosť zrušiť náhodné zatvorenie aplikácie a vybrať si, či naozaj chce aplikáciu opustiť.

Screen: Game

Tento screen predstavuje samotnú hru.
Na obrazovke sa vykresľuje bludisko, ktoré hráč prechádza. Bludisko obsahuje steny, ktoré vytvárajú cesty, kadiaľ môže hráč ísť, ako aj pole s hráčom (jeho aktuálna pozícia) a cieľom, ku ktorému sa hráč snaží dostať. Všetko je zobrazené prehľadne, aby hráč vždy vedel, kde sa nachádza a aké sú možné pohyby.
Mechanika hry a skóre:

- Hráč sa pohybuje po bludisku pomocou šípok, ktoré sú umiestnené na spodnej časti obrazovky.
- Každé stlačenie šípky spustí funkciu, ktorá spracuje pohyb hráča. Táto logika je implementovaná v súbore MazeGenerator.kt, kde sa rieši, či je pohyb možný, aktualizuje sa pozícia hráča a vykonávajú sa ďalšie herné kontroly.
- Ak sa hráč dostane na políčko s cieľom, systém overí, či vyhral. Po dosiahnutí cieľa sa spustí víťazná sekvencia.
- Hráč má k dispozícii skóre, ktoré sa odvíja od veľkosti bludiska. Výpočet je nasledovný: **X * Y rozmery bludiska** (napríklad 25x25 = 625 bodov na začiatku).
- Za každý úspešný pohyb sa skóre znižuje o 1 bod zo skóre.
- Ak hráč stratí všetky body (skóre dosiahne 0 alebo menej), prehráva a dostane penalizáciu -500 bodov.

Pomocné a akčné tlačidlá:

- **Nápoveda:** Po stlačení tlačidla nápovedy sa odhalí smer na ďalších 5 políčok v bludisku bližšie k cieľu, čím sa hráčovi uľahčí orientácia. Použitie nápovedy stojí 5 % z aktuálneho maximálneho skóre, čo hráča motivuje ju využívať rozumne.
- **Uloženie a načítanie:** Hráč má možnosť uložiť aktuálny stav bludiska a neskôr ho načítať pomocou tlačidiel Save a Load. To umožňuje pokračovať v hre neskôr alebo skúmať bludisko bez straty progresu.
- **Návrat:** Tlačidlo na návrat slúži na odchod späť k výberu hráča alebo k výberu veľkosti bludiska, čím sa hra preruší a používateľ môže meniť nastavenia.

Technické detaily a UI:

- Celý používateľský interface je vytvorený pomocou Jetpack Compose, čo zaručuje moderný, responzívny a plynulý zážitok.
- Stav hry a UI sa spravuje reaktívne, aby sa okamžite zobrazili zmeny v pozícii hráča, skóre či iných herných prvkoch.
- Interakcie so šípkami a tlačidlami sú implementované tak, aby boli intuitívne a bez oneskorenia.

V tomto screene sa používa ConfirmExitOnBackHandler, ktorý slúži na potvrdenie ukončenia aplikácie pri stlačení tlačidla späť. Používateľ tak má možnosť zrušiť náhodné zatvorenie aplikácie a vybrať si, či naozaj chce aplikáciu opustiť.

Screen: GameHorz

Tento screen je funkčne rovnaký ako základná herná obrazovka (Game Screen), avšak ide o **horizontálny layout**. Všetka herná logika, ovládanie, vykresľovanie bludiska, správa skóre a dialógy sú identické s pôvodnou hrou, líši sa len **rozloženie prvkov na obrazovke**, ktoré je prispôsobené na horizontálny režim zobrazenia.

V tomto screene sa používa ConfirmExitOnBackHandler, ktorý slúži na potvrdenie ukončenia aplikácie pri stlačení tlačidla späť. Používateľ tak má možnosť zrušiť náhodné zatvorenie aplikácie a vybrať si, či naozaj chce aplikáciu opustiť.

Súbor MazeGenerator.kt

V tejto časti je celá logika hry.

Save / Load mechanika

Uloženie bludiska podľa id hráča, bludisko sa skomprimuje aby zaberalo menej mesta, kde technicky vieme povedať, že vieme uložiť bludisko do 7xInt + X*Y*4/8 bajtov.

compressMaze(gameViewModel, mazeInfoViewModel, playerViewModel, mazeviewModel): Boolean

- **Popis:** Skontroluje, či sa v bludisku nachádza cieľ a hráč (pomocou findFinish a findPlayer), potom prevedie bludisko do binárneho reťazca a následne ho skompresuje do reťazca znakov. Nakoniec uloží komprimované bludisko spolu s informáciami o hráčovi a skóre do databázy.
- **Vracia:** true, ak sa kompresia a uloženie podarilo, inak false.

binaryStringToCompressedString(binaryString: String): String

- **Popis:** Prevedie binárny reťazec (reťazec '0' a '1') na komprimovaný reťazec znakov, kde každý znak reprezentuje 8 bitov.
- **Výstup:** Komprimovaný reťazec znakov.

compressedStringToBinaryString(compressed: String): String

- **Popis:** Prevod opačným smerom z komprimovaného reťazca znakov na binárny reťazec zložený z '0' a '1'.
- **Výstup:** Binárny reťazec.

mazeToString(maze: Maze): String

- **Popis:** Prejde každú bunku bludiska a vytvorí binárny reťazec, kde každá bunka je reprezentovaná 4 znakmi (top, bottom, left, right) s hodnotou '1' alebo '0' podľa toho, či je stena.
- **Výstup:** Binárny reťazec bludiska.

cellToString(cell: Cell): String

- **Popis:** Vytvorí reťazec 4 znakov '1' alebo '0' podľa toho, či má bunka stenu hore, dole, vľavo alebo vpravo.

stringToMaze(binary: String, cmaze: compMazes): Maze

- **Popis:** Prevedie binárny reťazec späť do objektu Maze, pričom obnoví steny každej bunky podľa binárnych hodnôt.
- **Výstup:** Objekt bludiska Maze.

decompressMaze(gameViewModel, mazeInfoViewModel, playerViewModel, mazeviewModel): Boolean

- **Popis:** Načíta uložené bludisko z databázy, rozbalí ho z komprimovaného formátu do binárneho, prevedie na Maze objekt, obnoví pozíciu hráča a cieľa, naplní informácie o bludisku a hráčovi a aktualizuje stav hry.
- **Vracia:** true, ak sa načítanie podarilo, inak false.

Herna Logika

winCheck(gameViewModel, mazeInfoViewModel): Boolean

- **Popis:** Skontroluje, či sa hráč nachádza na cieľovej pozícii, t.j. či vyhral hru.
- **Vracia:** true, ak je hráč na cieľovej bunke, inak false.

findFinish(gameViewModel, mazeInfoViewModel): Boolean

- **Popis:** Nájde pozíciu cieľa v bludisku. Najprv overí uloženú pozíciu, ak tam cieľ nie je, vyhľadá cieľ prechádzaním bludiska.
- **Vracia:** true, ak bol cieľ nájdený, inak false.

setSkore(skore: Int, mazeInfoViewModel)

- **Popis:** Aktualizuje aktuálne skóre v mazeInfoViewModel.

findPlayer(gameViewModel, mazeInfoViewModel): Boolean

- **Popis:** Nájde pozíciu hráča v bludisku podobne ako findFinish, najprv overí uloženú pozíciu, ak tam hráč nie je, vyhľadá hráča v bludisku.
- **Vracia:** true, ak hráč existuje v bludisku, inak false.

movePlayer(gameViewModel, mazeInfoViewModel, smer: Smer)

- **Popis:** Posunie hráča v zadanom smere, ak je to možné (bez steny). Aktualizuje pozíciu hráča, zníži skóre o 1, odstráni nápovedy a prehrá zvuk, ak sú povolené.
- **Poznámka:** Používa forceUpdateMaze na aktualizáciu stavu bludiska (bez toho neaktualizovalo bludisko v UI).

forceUpdateMaze(gameViewModel, maze)

- **Popis:** Vynúti aktualizáciu bludiska vytvorením novej kópie 2D poľa buniek, aby sa Compose správne prekreslil.

randomFinish(maze: Maze)

- **Popis:** Náhodne vyberie jednu bunku a označí ju ako cieľ (finish). Vymaže predchádzajúce označenia finish a resetuje flood hodnoty.

verifyMaze(maze: Maze): Boolean

- **Popis:** Overí, či bludisko obsahuje aspoň jednu bunku označenú ako finish a jednu bunku s hráčom.
- **Vracia:** true, ak sú obaja nájdení, inak false.

generatePlayer(maze: Maze)

- **Popis:** Vyberie bunku, ktorá má najvyššiu flood hodnotu (prednosť majú hodnoty >= 150) a označí ju ako pozíciu hráča. Všetky ostatné bunky resetuje.
- **Použitie flood hodnoty:** pre umiestnenie hráča čo najďalej od cieľa.

removeHint(gameViewModel)

- **Popis:** Vymaže všetky nápovedy (hint hodnoty) z bludiska a aktualizuje stav hry.

hint(gameViewModel, mazeInfoViewModel, x: Int = -1, y: Int = -1, counter: Int = 0)

- **Popis:** Rekurzívne vytvára nápovedu pre hráča z aktuálnej pozície (alebo zadanej pozície) v bludisku. Postupuje po bunkách s nižšou flood hodnotou a označuje ich hodnotou hint.
- **Použitie:** Nápoveda k správnej ceste v bludisku.

floodMaze(maze: Maze)

- **Popis:** Vykoná tzv. flood fill algoritmus od cieľa v bludisku. Označí vzdialenosť každej bunky od cieľa (flood hodnotou), ignorujúc steny.
- **Výsledok:** Pomocná informácia pre nápovedu a generovanie pozície hráča.

generateMaze(x: Int, y: Int, maze: Maze, randwall: Int)

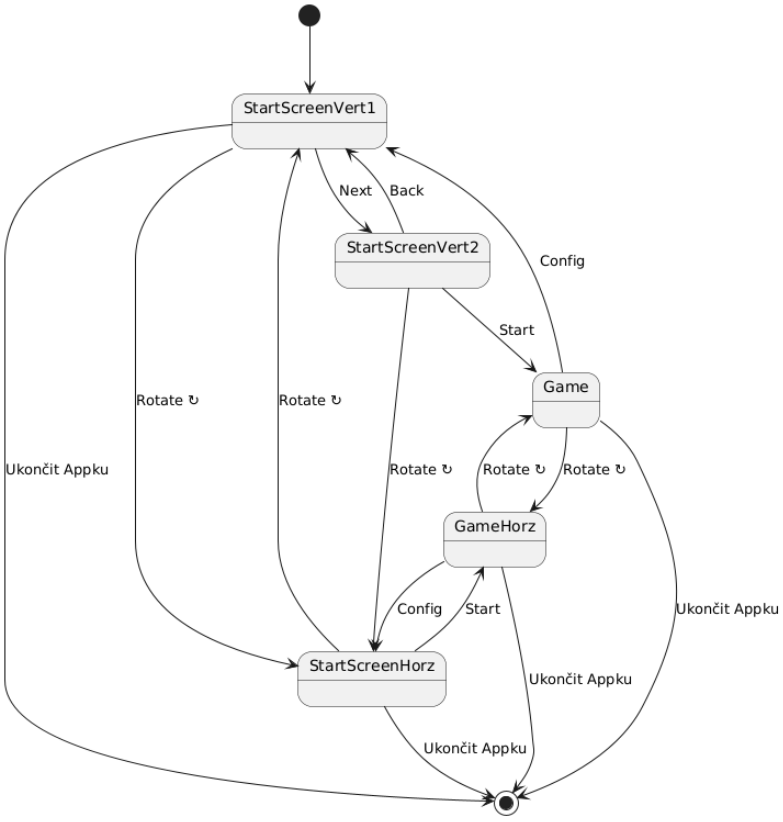
- **Popis:** Rekurzívny algoritmus na generovanie bludiska. Prechádza náhodne susedné bunky, odstraňuje steny medzi nimi a vytvára cesty. Parametr randwall určuje pravdepodobnosť odstránenia náhodnej steny medzi už navštívenými bunkami (pre viac otvorené bludisko).
- **Použitie:** Na generovanie nového bludiska.

Enum Class Smer

Tento enum class Smer udáva iba smer pohybu v bludisku.
Obsahuje štyri možné smery: hore (UP), dole (DOWN), doprava (RIGHT) a doľava (LEFT).

Grafické Znázornenie Aplikácie

UML diagram znázorňuje prechody medzi všetkými obrazovkami aplikácie. Zobrazuje možné cesty, ktorými sa používateľ môže pohybovať medzi jednotlivými obrazovkami (StartScreenVert1, StartScreenVert2, StartScreenHorz, Game, GameHorz), vrátane akcií ako **spustenie hry**, **rotácia obrazovky**, **návrat späť** alebo **ukončenie aplikácie**.

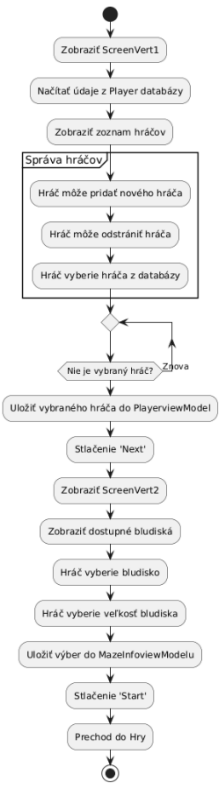


UML diagram znázorňuje proces získavania informácií potrebných pre spustenie hry.
Zobrazuje jednotlivé obrazovky aplikácie, cez ktoré používateľ postupne prechádza (ScreenVert1, ScreenVert2) a akcie, ktorými zadáva kľúčové vstupy pre hru.

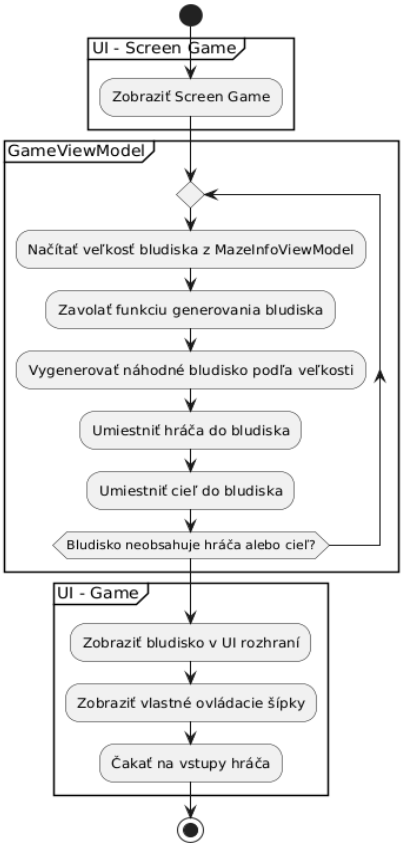
Diagram zachytáva:

- Načítanie zoznamu hráčov z databázy,
- Výber konkrétneho hráča a jeho uloženie do PlayerviewModel,
- Následný výber bludiska a jeho veľkosti,
- Uloženie výberu do MazeInfoviewModel,
- A finálne spustenie samotnej hry s použitím zvolených údajov.

Diagram tak ilustruje **kompletný tok používateľských vstupov** a ich spracovanie do podoby, ktorú herná logika dokáže využiť pri generovaní herného prostredia.



Tento UML activity diagram znázorňuje, ako sa získavajú a validujú informácie potrebné pre spustenie hry v aplikácii.

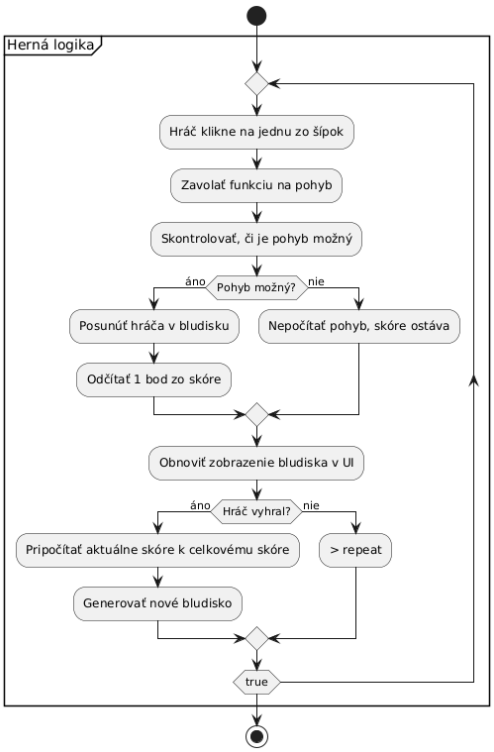


1. **UI - Screen Game**:
   Používateľ začne na obrazovke, kde sa zobrazuje základné herné rozhranie pred samotnou hrou.

2. **GameViewModel**:
   V tejto časti sa v cykle opakovane generuje bludisko podľa veľkosti, ktorá je načítaná z MazeInfoViewModel.
   Postup je nasledovný:

   - Načíta sa veľkosť bludiska, ktorú si používateľ zvolil.
   - Zavolá sa funkcia, ktorá náhodne vygeneruje bludisko podľa tejto veľkosti.
   - Do bludiska sa umiestni hráč aj cieľ.
   - Následne sa kontroluje, či bludisko obsahuje hráča aj cieľ.
   - Ak nie (bludisko nie je platné), celý proces generovania sa zopakuje (opakuje sa cyklus).
   - Ak je bludisko platné, pokračuje sa ďalej.

3. **UI - Game**:
   Po úspešnej validácii sa bludisko zobrazí v hernom používateľskom rozhraní.
   Hráč môže ovládať pohyb pomocou vlastných šípok zobrazených pod bludiskom.
   Systém čaká na vstupy hráča a hra pokračuje.

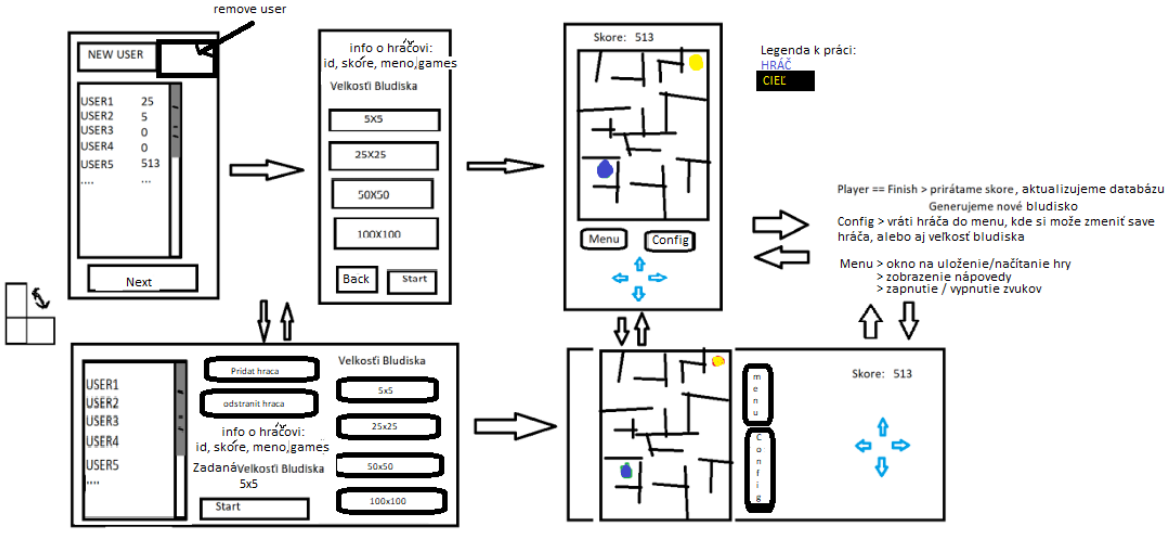


4. **Herná logika:**
   Hráč klikne na šípku a funkcia skontroluje, či je pohyb možný. Ak áno, hráč sa posunie a skóre sa zníži o jeden bod. UI sa aktualizuje, potom sa kontroluje, či hráč vyhral. Ak áno, skóre sa pripočíta k celkovému a vygeneruje sa nové bludisko. Ak nie, cyklus pokračuje odznova.

5. **Reset Bludiska:**
   Program sa presunie do bodu **2. GameViewModel**

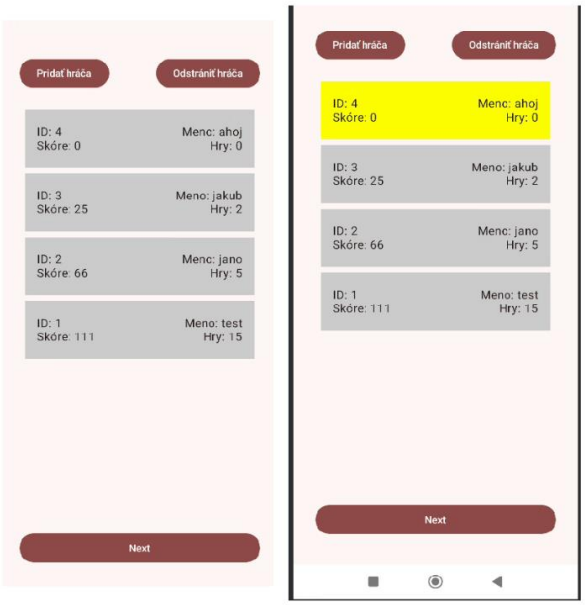
Návrh vzhľadu obrazoviek

Rýchly pohľad na obrazovky



Podrobný pohľad na obrazovky:

Po spustení aplikácie sa zobrazí táto obrazovka, kde si hráč vyberie uložený save a klikne na „Next“, čím sa presunie na ďalšiu obrazovku.



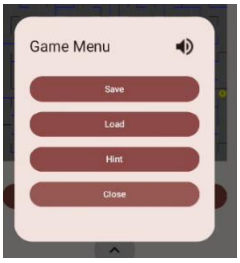
Obrazovka č.2 je na zadanie veľkosti bludiska a ovláda sa rovnako ako predošlá:

Obrazovka č.3 je samotná hra:

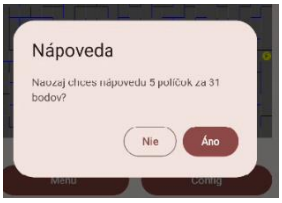
Hráč sa pohybuje pomocou šípok na spodku obrazovky a jeho cieľom sa je dostať na žlté políčko s F(finish), (hráč je modrý (P - player)).

Config tlačidlo je na návrat na obrazovku č.1 a teda znovu výber nastavenia hráčovho save a veľkosti bludiska.

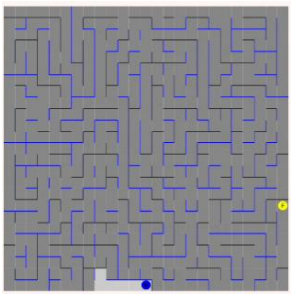
Menu tlačidlo otvorí pop up okno s ďalšími nastaveniami:



Mute Ikonka: zapnutie/vypnutie zvukou
Save: Uloženie hry.
Load: Načítanie hry.
Hint: Zobrazenie 5 políčok na najkratšiu cestu ku cieľu za cenu 5% z max získaných bodov za tento level.
Close: Zatvorenie menu.



Po stlačení áno, sa zobrazí bludisko takto:

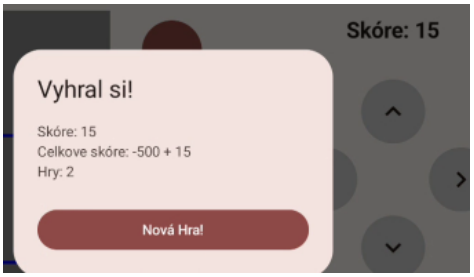


Výhra/Prehra:

Prehra nastane, keď hráč bude mať aktuálne skóre menšie ako 0. Hra sa vtedy ukončí a hráč dostane penalizáciu –500 bodov.

Výhra nastane, keď sa hráč dostane na rovnakú pozíciu ako cieľ. K jeho profilu sa pripočíta aktuálne skóre a hra sa ukončí.

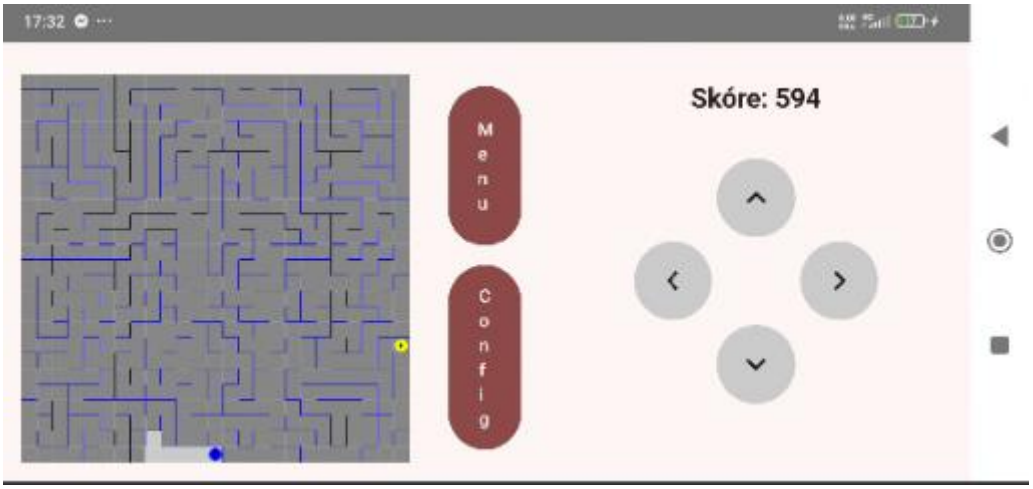
Po ukončení sa hra dá znova pregenerovať a hrať.



Obrazovka č.1.2->Horizontálna:
Rovnaké funkcie ako obrazovka č. 1 a 2

Obrazovka č.3.2->Horizontálna:

Rovnaké funkcie ako obrazovka č. 3



Zoznam zdrojov

Obchod play

Cvičenia zo školy

Package sql -> + všetky kódy pod týmto packagom, sú kódy prevzaté a následne upravené zo cvičení VAPMZ.

Chatgpt.com -> riešenie chýb v kóde, drobné optimalizačné časti, zdroj informácii pre špecifické funkcie

https://editor.plantuml.com/uml/SoWkIImgAStDuUMArefLqDMrKmW6YGGd5wMcveMb5fKOydBnSg6BOLDef62bu9TQKK9OjKT-KHLGSL1-IM99giAZjjquj81S10HM2g7P-Nb7UZDB51mB2h9BiJZ82AlBWJgT4fEp2Qicx2285lV4t5IWEh0ScYl0Hi3KGCs3giEc2ejJSTWErm893Mo10NOMcjt8fi7biTZgd9-NbfcEf0Pi3IvRK7o9nHPi_eCw2DaYD3zXiSuXDIy5d2e1

https://pixabay.com/sound-effects/search/tick/ - zvuk pri pohybe