Dokumentácia k aplikácii Tréner slov

Tím SZTS =

Lukáš Slaninka,

Jakub Soviš,

Klaudia Turčeková,

Ján Zdarilek

2018/2019

Katalóg požiadaviek

1. Inštrukcie

1.1. Na čo a komu je určený systém

Tento dokument informuje o funkcionalite a využití systému Audio-vizuálny tréner slovnej zásoby. Je určený pre tých, ktorý s ním budú pracovať alebo ho vytvárať.

1.2. Funkcionalita v hrubých rysoch

Audio-vizuálny tréner slovnej zásoby plní jednu základnú funkciu. Precvičiť a naučiť sa slovnú zásobu cudzieho jazyka, pomocou rôznych metód. Slúži iba na precvičenie fráz, slov a slovných spojení. Nie je to náhrada nejakého jazykového kurzu, ale len doplnok na výučbu cudzieho jazyka. Učí v štyroch interaktívnych módoch.

1.3. Vysvetlenie pojmov

Mód – jeden zo štyroch spôsobov učenia a precvičovania v systéme (učenie, skúšanie, diktát a stacionárny bicykel)

Lekcia – spája skupiny do jedného učebného celku, na ktorom používateľ spúšťa módy

Skupina – každá skupina obsahuje niekoľko položiek

Položka – základná jednotka všetkých módov, reprezentovaná ako otázka a odpoveď

Prebraná skupina – používateľ odpovedal na všetky položky v skupine správne niekoľkokrát

Správne odpovedanie – znamená priradenie správneho obrázku, textu alebo zvuku k zodpovedajúcej otázke

Odpovedanie – označenie odpovede na správnu alebo nesprávnu

Stacionárny bicykel – jeden zo štyroch módov, prezentácia položiek v náhodnom poradí

1.4. Opis ďalších kapitol

V ďalších kapitolách je opísaná základná funkcionalita a požiadavky, ktoré musí systém spĺňať.

2. Základný opis

2.1. Systém v kontexte

Audio-vizuálny tréner slovnej zásoby je systém na učenie a precvičenie slovnej zásoby cudzieho jazyka. Môže sa použiť pre individuálne vzdelávanie jednotlivca.

2.2. Stručný opis celej funkcionality

Používateľ bude môcť systém používať na precvičenie slovnej zásoby v cudzom jazyku. Môže položky, skupiny, lekcie vytvárať , mazať, editovať. Môže určovať poradie skupín v lekcii. Vtedy pracuje v role učiteľ.

Používateľ vždy odpovedá na otázky tak, že si svoju odpoveď zaznačí niekam mimo systému. Systém sa potom už iba opýta, či používateľova zaznačená odpoveď bola správna alebo nie.

Používateľ sa môže učiť v jednom zo štyroch módov.

V móde učenia si používateľ zvolí poradie precvičovaných skupín, potom odpovedá na otázky a ak niekoľkokrát odpovie správne na otázku, tá sa už neprecvičuje.

V móde skúšanie používateľ odpovedá na otázky dovtedy, dokým na ňu neodpovie správne.

V móde diktát používateľ počúva nahrávky slovných spo22jení alebo slov, ktoré si zapisuje na papier. Sám si skontroluje správnosť, ktorá sa mu zobrazí po interakcii.

V móde stacionárny bicykel používateľ sleduje prezentáciu položky.

V každom móde sa využíva iný algoritmus na precvičenie alebo naučenie slov alebo slovných spojení presne vysvetlený v bode 3.1.

2.3. Typy používateľov

Audio-vizuálny tréner je systém, ktorý bude pracovať s jedným typom používateľa. Ten môže zastávať viaceré role:

i. Rola, kde je používateľ žiakom. To znamená, že sa učí. Používa módy učenia.

ii. Rola, kde je používateľ učiteľom. Vytvára vlastné lekcie, skupiny alebo položky.

2.4. Všeobecné obmedzenia

Systém bude dodržiavať platné pravidlá všetkých jazykov, ktoré bude obsahovať.

Bude fungovať na operačnom systéme Windows.

Obrázky a zvuky použité v systéme budú mať minimálne požiadavky (veľkosť, formát).

Obrázky sa budú prispôsobovať veľkosti okna, budú zaberať určité percento z plochy okna aplikácie.

Systém bude používať predvolené zvukové zariadenie operačného systému.

2.5. Rozhrania systému s jeho okolím a ich vlastnosti

Aplikácia bude slúžiť na výučbu, preto musí mať jednoduché, prehľadné používateľské rozhranie

3. Požiadavky

3.1. Funkcie

3.1.1.Systém bude bežať primárne na Windowse.

3.1.2.Lekcia bude obsahovať minimálne jednu skupinu

3.1.3.Skupina bude obsahovať aspoň tri položky

3.1.4.Základnou učebnou jednotkou bude položka

3.1.5. Položka bude reprezentovaná ako otázka a odpoveď. Kde otázka a odpoveď budú reprezentované ako obrázok, zvuk alebo text. V otázke aj odpovedi musí byť minimálne jedno (môžu sa použiť ľubovoľné kombinácie týchto troch prvkov v otázke aj odpovedi, aj všetky tri):

3.1.5.1. Obrázok

3.1.5.2. Zvuk

3.1.5.3. Text

3.1.6. Vždy sa bude precvičovať celá lekcia

3.1.7. Systém bude mať štyri módy učenia:

3.1.7.1. **Mód učenia bude mať vlastný algoritmus fungovania:**

3.1.7.1.1. Používateľ si zvolí lekciu, ktorú sa chce naučiť

3.1.7.1.2. Používateľ zadá počet opakovaní skupín v lekcii, po ktorých sa skupina označí ako prebraná

3.1.7.1.3. Vytvorí sa rad, do ktorého sa postupne budú zaraďovať skupiny v poradí určenom používateľom (používateľ si zvolí poradie skupín pri vytváraní lekcie alebo si upraví poradie skupín v lekcii pri vytváraní alebo editovaní lekcii) alebo v náhodnom poradí

3.1.7.1.4. Do radu sa zaradia prvé dve skupiny

3.1.7.1.5. Po prebraní radu sa na koniec zaradí ďalšia skupina v poradí

3.1.7.1.6. Rad sa preberá odznovu

3.1.7.1.7. Bod 3.1.7.1.5 a bod 3.1.7.1.6 sa opakujú dokým nie je každá skupina prebraná používateľom určený počet krát (používateľ si nastaví na začiatku koľko krát chce skupiny opakovať)

3.1.7.1.8. Ak sa aspoň na jednu položku v skupine odpovie nesprávne, skupina sa zaraďuje do radu znova

3.1.7.1.9. Keď sú všetky skupiny v rade prebrané, z radu sa vyradí prvá skupina a rad sa preberá odznova

3.1.7.1.10. Bod 3.1.7.1.9 sa opakuje dokým nie je rad prázdny, potom je učenie lekcie ukončené

3.1.7.2. **Mód skúšania bude mať vlastný algoritmus fungovania:**

3.1.7.2.1. Lekcia sa bude spúšťať po položkách, rozdelenie do skupín tu nebude zohľadnené

3.1.7.2.2. Vytvorí sa rad položiek v náhodnom poradí

3.1.7.2.3. Používateľ odpovedá na otázky

3.1.7.2.4. Ak používateľ odpovie správne na otázku, položka sa z radu vyhodí

3.1.7.2.5. Ak používateľ odpovie nesprávne na otázku, položka sa zaradí na koniec radu

3.1.7.2.6. Takto používateľ odpovedá na otázky kým nie je rad prázdny

3.1.7.3. **Mód diktát bude mať vlastný algoritmus fungovania:**

3.1.7.3.1. Lekcia sa bude spúšťať po položkách, rozdelenie do skupín tu nebude zohľadnené

3.1.7.3.2. Vytvorí sa rad položiek (rad obsahuje len položky obsahujúce zvuk) v náhodnom poradí

3.1.7.3.3. Používateľ si na papier zapíše slovo alebo slovné spojenie ktoré počul alebo videl na obrázku. Potom dá aplikácii signál, aby zobrazila príslušnú odpoveď. Kliknutím signalizuje, či napísal odpoveď správne.

3.1.7.3.4. Ak používateľ napíše text správne, položka sa z radu vyhodí.

3.1.7.3.5. Ak používateľ napíše text nesprávne, položka sa zaradí na koniec radu

3.1.7.3.6. Takto používateľ prejde všetky položky kým nie je rad prázdny

3.1.7.4. **Mód stacionárny bicykel bude mať vlastný algoritmus fungovania:**

3.1.7.4.1. Používateľ si bude môcť pred spustením tohto módu zvoliť, koľkokrát sa prehrá zvuk z odpovede (ak odpoveď zvuk obsahuje). Prednastavená hodnota budú tri opakovania.

3.1.7.4.2. Lekcia sa bude spúšťať po položkách, rozdelenie do skupín tu nebude zohľadnené

3.1.7.4.3. Vytvorí sa rad položiek v náhodnom poradí

3.1.7.4.4. Položky v rade sa používateľovi zobrazujú postupne za sebou. Najskôr otázka a vzápätí aj odpoveď (dĺžku pauzy a veľkosť fontov si nastaví používateľ na začiatku).

3.1.7.4.5. Takto prejde celý rad (funguje to na princípe prezentácie)

3.1.7.4.6. Keď používateľ prejde celú lekciu, potom sa položky znova zamiešajú a pokračuje sa prezentácii

3.1.7.4.7. Daný mód beží až kým ho nezastaví používateľ, alebo keď uplynie čas, ktorý používateľ na beh tohto módu nastavil. Prednastavená hodnota je že, mód beží “večne“.

3.1.7.4.8. Používateľ v tomto móde len spustí a zastaví prezentáciu. (Ak nevyužije možnosť nastaviť čas bežania módu.)

3.1.7.5. Položky, skupiny, lekcie sa budú dať vytvoriť, zmazať, upravovať, importovať, exportovať

3.1.7.6. Bude sa dať vytvoriť z existujúcej položky nová, taká ktorá bude mať prehodenú otázku a odpoveď. To znamená, že otázka pôvodnej bude zodpovedať odpovedi novej a odpoveď pôvodnej bude zodpovedať otázke novej.

3.1.7.7. Systém bude po nainštalovaní obsahovať iba jednu lekciu, ktorá bude spĺňať minimálne požiadavky viď body od 3.1.2 do 3.1.5

3.1.7.8. Používateľ si bude môcť nastaviť veľkosť fontu.

3.2. Ostatné požiadavky

Dizajn aplikácie bude vizuálne estetický a podporujúci učenie sa.

3.3. Požiadavky rozhrania

Veľkosť okna sa bude prispôsobovať rozlíšeniu obrazovky.

Všetky texty použité v aplikácii budú zakódované v UTF-8.

Návrh aplikácie

1. Formáty súborov, s ktorými bude aplikácia pracovať:
   - Formát zvuku .mp3
     - Maximálna veľkosť zvuku v tomto formáte
   - Formát zvuku .wav
     - Maximálna veľkosť zvuku v tomto formáte
   - Formát obrázku .png
     - Maximálna veľkosť v pixeloch v tomto formáte
   - Formát obrázku .jpg
     - Maximálna veľkosť v pixeloch v tomto formáte
   - XML súbor
     - V tomto súbore budú údaje o jednotlivých lekciách, skupinách, položkách. Presnejšie aké lekcie sa nachádzajú v aplikácii, aké skupiny sú v jednotlivých lekciách, aké položky sú v jednotlivých lekciách a aký obrázok (iba 1.3 alebo 1.4 sú povolené formáty), zvuk (iba 1.1 alebo 1.2 sú povolené formáty) a text majú jednotlivé položky

2. Dátový model perzistentných údajov
   - Dáta aplikácie budú uložené v priečinku data. Ten bude obsahovať:
     - súbor data.xml – v ňom budú lekcie a nastavenia aplikácie
     - priečinok files - v ňom budú priečinky images a sounds
       - ✓ Priečinok image bude obsahovať:
         - Obrázky
       - ✓ Priečinok sounds bude obsahovať:
         - Zvukové súbory
     - Schéma data.xml súboru bude nasledovná:

```
<xs:schema element FormDefault="qualified"xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="data">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="nastavenia">
     <xs:complexType>
      <xs:element name="font_size" type="xs:integer"/>
     </xs:complexType>
    </xs:element>
    <xs:element name="lekcie">
     <xs:complexType>
      <xs:element name="lekcia" maxOccurs="unbounded" minOccurs="1">
       <xs:complexType>
        <xs:sequence>
         <xs:element type="xs:string" name="nazov"/>
         <xs:element name="skupiny">
          <xs:complexType>
           <xs:element name="skupina" maxOccurs="unbounded" minOccurs="1">
            <xs:complexType>
             <xs:sequence>
              <xs:element type="xs:string" name="nazov"/>
              <xs:element type="xs:integer" name="poradie"/>
              <xs:element name="polozky">
               <xs:complexType>
                <xs:element name="polozka" maxOccurs="unbounded"minOccurs="3">
                 <xs:complexType>
                  <xs:choice>
                   <xs:element type="xs:string" name="text_otazky"/>
                   <xs:element type="xs:string" name="obrazok_otazky"/>
                   <xs:element type="xs:string" name="zvuk_otazky"/>
                  </xs:choice>
                  <xs:choice>
                   <xs:element type="xs:string" name="text_odpovede"/>
                   <xs:element type="xs:string" name="obrazok_odpovede"/>
                   <xs:element type="xs:string" name="zvuk_odpovede"/>
                  </xs:choice>
                 </xs:complexType>
                </xs:element>
               </xs:complexType>
              </xs:element>
             </xs:sequence>
            </xs:complexType>
           </xs:element>
          </xs:complexType>
         </xs:element>
        </xs:sequence>
       </xs:complexType>
      </xs:element>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

- Vysvetlenie:
  - ✓ Koreňový element bude element **data**.
  - ✓ Bude obsahovať elementy v poradí **nastavenia** a **lekcie**.
  - ✓ Element **nastavenia** bude obsahovať element **font_size**.
  - ✓ Element **lekcie** bude obsahovať aspoň jeden element **lekcia.**
  - ✓ **Lekcia** bude mať element **nazov** a element **skupiny** obsahujúci minimálne jednu **skupinu**.
  - ✓ **Skupina** bude mať elementy **nazov**, **poradie** a element **položky** obsahujúci minimálne 3 **položky**.
  - ✓ Položka bude obsahovať aspoň jeden z elementov **text_otazky**, **obrazok_otazky** a **zvuk_otazky** a aspoň jeden z elementov **text_odpovede**, **obrazok_odpovede** a **zvuk_odpovede**

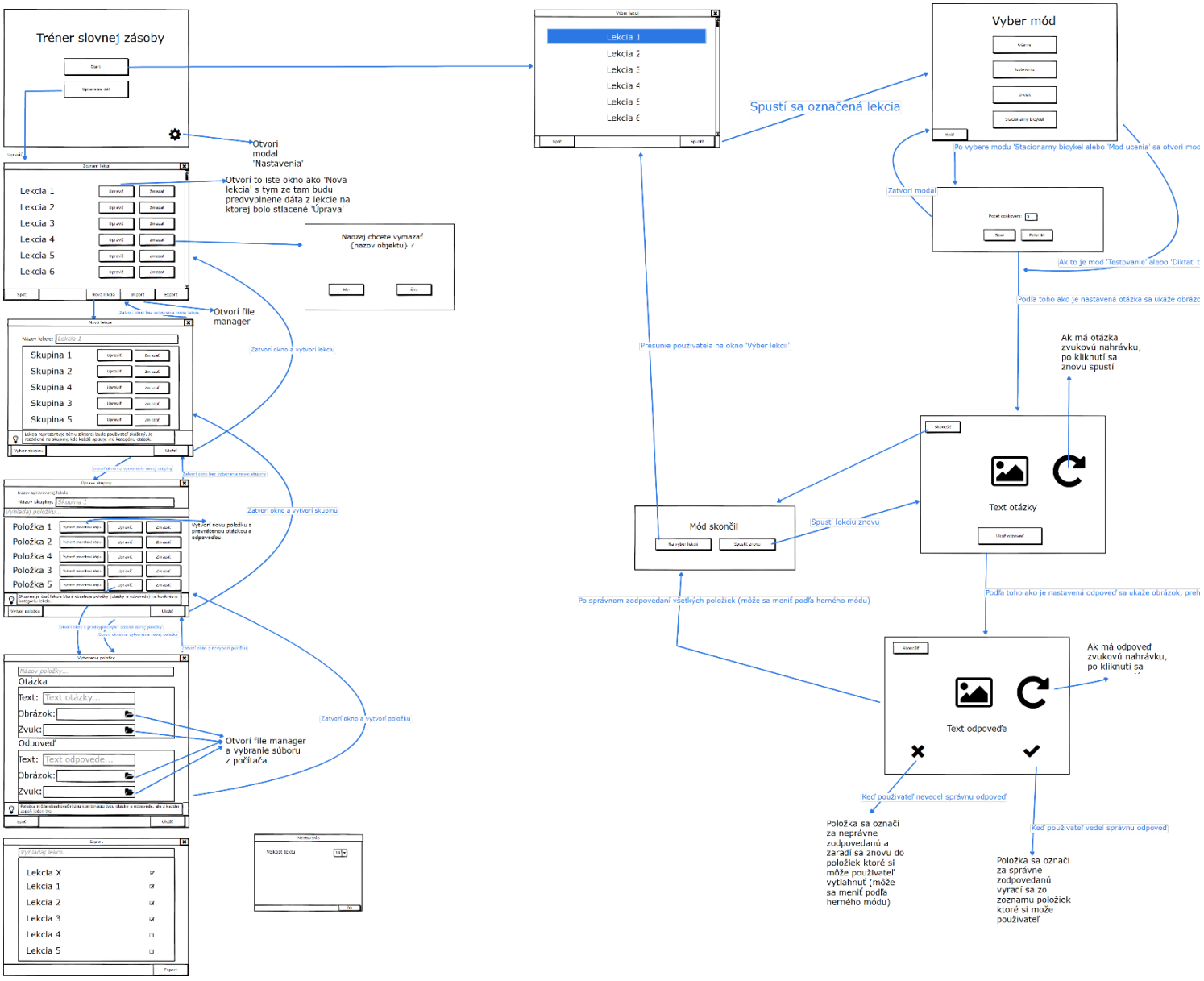
3. Návrh implementácie
   - **Moduly**
     - DataController
       - ✓ bude obsahovať metódy na vyberanie, vkladanie, updatovanie a mazanie údajov z dátového súboru.
     - FileManager
       - ✓ bude obsahovať metódy na premiestňovanie, pridávanie, mazanie súborov (obrázky, zvuky)
     - Export
       - ✓ Bude obsahovať metódu, ktorá z vybraných lekcií, skupín a položiek urobí balíček a ponúkne používateľovi, aby si ho uložil na vybrané miesto do zariadenia
       - ✓ Balíček (priečinok alebo zip) bude obsahovať súbor s dátami a priečinok s obrázkami a zvukmi
       - ✓ Bude používať DataController a FileManager
     - Import
       - ✓ Bude obsahovať metódu, ktorá uloží lekcie, skupiny a položky z balíčka
       - ✓ Dáta uloží do dátového súboru a súbory (obrázky, zvuky) na správne miesto
       - ✓ Balíček vznikol exportom
       - ✓ Balíček užívateľ vyberie pomocou FileChoosera
       - ✓ Bude používať moduly DbController a FileManager
     - MainController
       - ✓ Bude obsahovať všetky hlavné funkcie programu – pridávanie, mazanie, editovanie lekcii, skupín, položiek, prispôsobenie grafického rozhrania, ...
       - ✓ Bude používať moduly DbController, FileManager, Export, Import
     - GUI
       - ✓ Ponuka možností, zobrazuje dáta

- ✓ Komponentami (buttony, checkboxy, ...) sa budú spúšťať funkcie z MainControllera
- ModesController
  - ✓ Bude obsahovať algoritmy jednotlivých módov
  - ✓ Používa funkcie GUI na zobrazenie položiek, lekcií

- **Technológie**
  - Java
    - ✓ Výsledný program bude buildnutý do executable jar súboru
    - ✓ Obsahuje všetky zdrojové súbory a knižnice projektu
    - ✓ Spustením celého programu sa spustí main
  - JavaFX Scene Builder
    - ✓ Týmto nástrojom bude realizované celé GUI
    - ✓ vytvorí .fxml súbor, ktorý popisuje rozloženie grafických componentov

Diagramy

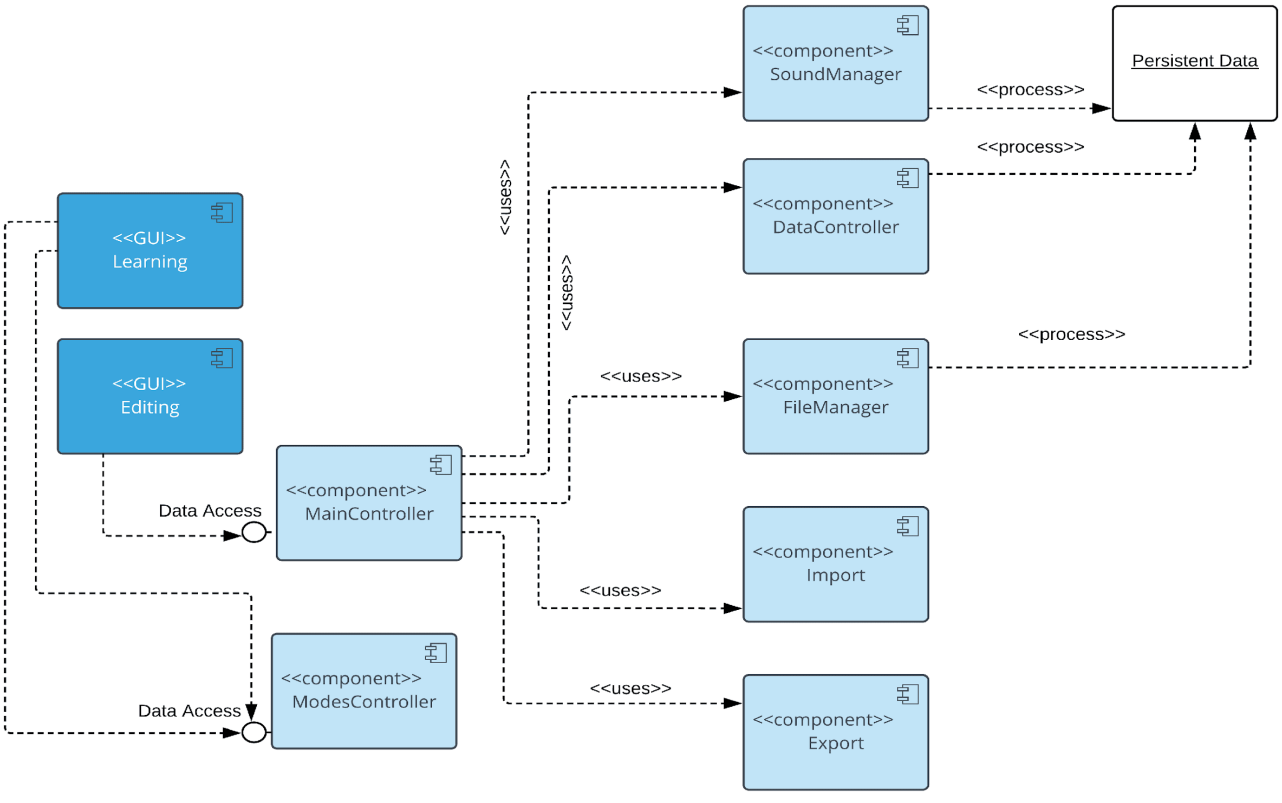
Návrh používateľského rozhrania



Komponent Diagram

- Jednotlivé komponenty predstavujú základné triedy, z ktorých bude celý systém zostavený.
- Komponenty DataController, FileManager, DataController budú spracovávať perzistentné dáta (xml, mp3, png, jpg).
- MainController bude riadiť a používať všetky komponenty
- Používateľské rozhranie je v diagrame logicky rozdelené na 2 časti, aby sa dalo vidieť, že používateľ (Študent) riadi len časť Učenie (Learning) používateľského rozhrania, tak isto používateľ (Učiteľ) riadi len časť Upravovanie (Editing)
- V systéme bude celé používateľské rozhranie ako jedna časť
- Editovanie a Učenie plnia pokyny, ktoré mu zadáva používateľ, posiela dáta do MainControllera, ktorý ďalej spracováva pokyny
- ModesController zobrazuje jednotlivé módy učenia v používateľskom rozhraní (Učenie)
- SoundManager komunikuje s perzistentnými dátami, pokyny mu bude posielať MainController

Komponent diagram:



Triedny diagram

- Trieda Main
  - ✓ Je to hlavná trieda programu
  - ✓ Obsahuje metódu main(String[] args), ktorá sa spustí ako prvá
  - ✓ Metóda start(primaryStage) inicializuje grafické rozhranie.
- Trieda GUI
  - ✓ Je to symbolická trieda pre všetky Gui controllery
  - ✓ Bude vytvorený Gui controller pre každú scénu v aplikácii
  - ✓ Bude riadiť dáta a grafické komponenty pre svoju scénu
- Trieda GameMode
  - ✓ Abstraktná trieda pre všetky štyri módy
  - ✓ Obsahuje atribút lesson - je to lekcia, nad ktorou je mód spustený
  - ✓ Metódy
    - next(answer) - Tato metóda bude implementovaná v každom móde inak
      - Dostane odpoveď na predchádzajúcu otázku (true - dobre, false - zle) a podľa algoritmu daného módu vráti ďalšiu položku
    - randomize() - Zamieša obsah preberanej lekcie
    - reinitialize() - pripravý mód na spustenie
- Trieda MainController
  - ✓ Táto trieda je hlavné prepojenie medzi GUI controllermi a triedami určenými na prácu s dátami
  - ✓ Metódy
    - importLessons() - vytvorí objekt triedy Import a ak ma importovaný súbor správny formát, tak pomocou DataControllera načíta lekcie zo súboru a uloží k lekciám v aplikácii
    - exportLesson(lesson) - vytvorí objekt triedy Export, overí, či je v cieľovej lokácii exportu právo na zápis a ak áno,

pomocou DataControllera vytvorí balík s exportovanými lekciami
    - removeLesson(lesson) - pomocou DataControllera odstráni lekciu z načítaných lekcií
    - addLesson(lesson) - pomocou DataControllera pridá lekciu do načitaných lekcií
    - getLessons(search) - pomocou DataControllera získa zoznam lekcií zodpovedajúcich vyhľadávajúcemu reťazcu
        - saveData() - pomocou DataControllera uloží načítané (a v aplikácii upravované) dáta (lekcie a nastavenia) do dátového súboru aplikácie
        - loadData() -načíta dáta z dátového súboru
        - playSound(soundFilePath) - použitím triedy SoundManager prehrá zvukový súbor
        - getSoundDuration(soundFilePath) - vráti dĺžku trvania zvuku
    - getFontSize() - z DataControllera získa veľkosť fontu a vráti ho
        - setFontSize(fontSize) - pomocou DataControllera nastaví veľkosť fontu v aplikácii
        - saveFilesInItem(item, newQImage, newQSound, newAImage, newASound) - v položke item uloží nové súbory
- Trieda Import
    - ✓ Trieda sa stará o importovanie lekciíi do aplikácie
    - ✓ Metódy
        - choosePackagePath() - otvorí pre používateľa FileChooser, kde úlohou používateľa je vybrať vstupný súbor; cesta k tomuto súboru sa uloží do triedneho atribútu packagePath
        - isValidPackage() - overí, či štruktúra vstupného súboru je správna, je zadefinovaná v návrhu

perzistentných dát; metóda vráti true, ak je správna, inak false
    - unzip(path, save_path) - rozzipuje vstupný súbor
    - newFile() - vytvorí nový súbor
    - loadLessonsFromFile(path) - načíta lekcie zo vstupného súboru

- Trieda Export
  - ✓ Trieda sa stará o exportovanie vybraných lekcií
  - ✓ Metódy
    - chooseTargetPath() - otvorí pre používateľa FileChooser, kde úlohou používateľa je vybrať miesto, na ktoré chce súbor uložiť; cesta k tomuto súboru sa uloží do triedneho atribútu targerPath
    - hasWriteAccess() – Vráti true, ak v cieľovom priečinku má program právo na písanie, inak vráti false
    - zipDirectory(sourceDirPath, zipPath, type) - zazipuje priečinok a uloží ho cieľové miesto
    - saveLessonsToFile(lessons) - uloží zvolené lekcie do súboru
- Trieda DataController
  - ✓ Bude slúžiť manipuláciu s perzistentnými dátami
  - ✓ Bude ich načítavať, zapisovať a načítané dáta bude u seba držať
  - ✓ Atribút dataFilePath je konštanta obsahujúca cestu k dátovému priečinku
  - ✓ Metódy
    - unitLessonsSets(lessons1, lessons2) - zjednotí 2 kolekcie lekcií a vráti výsledok; bude potrebná pri importovaní lekcii, keď aplikácia už bude mať nejaké importované lekcie obsahovať
    - loadData() - načíta perzistentné dáta; bude volaná v konštruktore

- saveData() - uloží údaje načítané v aplikácii do perzistentných dát; bude používaná pri editácii
- getLessons(search) - vráti lekcie zodpovedajúce vyhľadávajúcemu reťazcu
- addLesson(lesson) - pridá lekciu
- removeLesson(lesson) - odstráni lekciu
- getFondSize() - vráti veľkosť fontu, ktorá je nastavená v aplikácii
- setFondSize(fontSize) - nastaví aplikácii novú hodnotu veľkosti fontu
- CreateDataControllerFromXml(path) - vytvorí objekt DataControllera obsahujúci údaje z xml súboru
- saveFilesInItem(item, newQImage, newQSound, newAImage, newASound) - v položke item uloží nové súbory

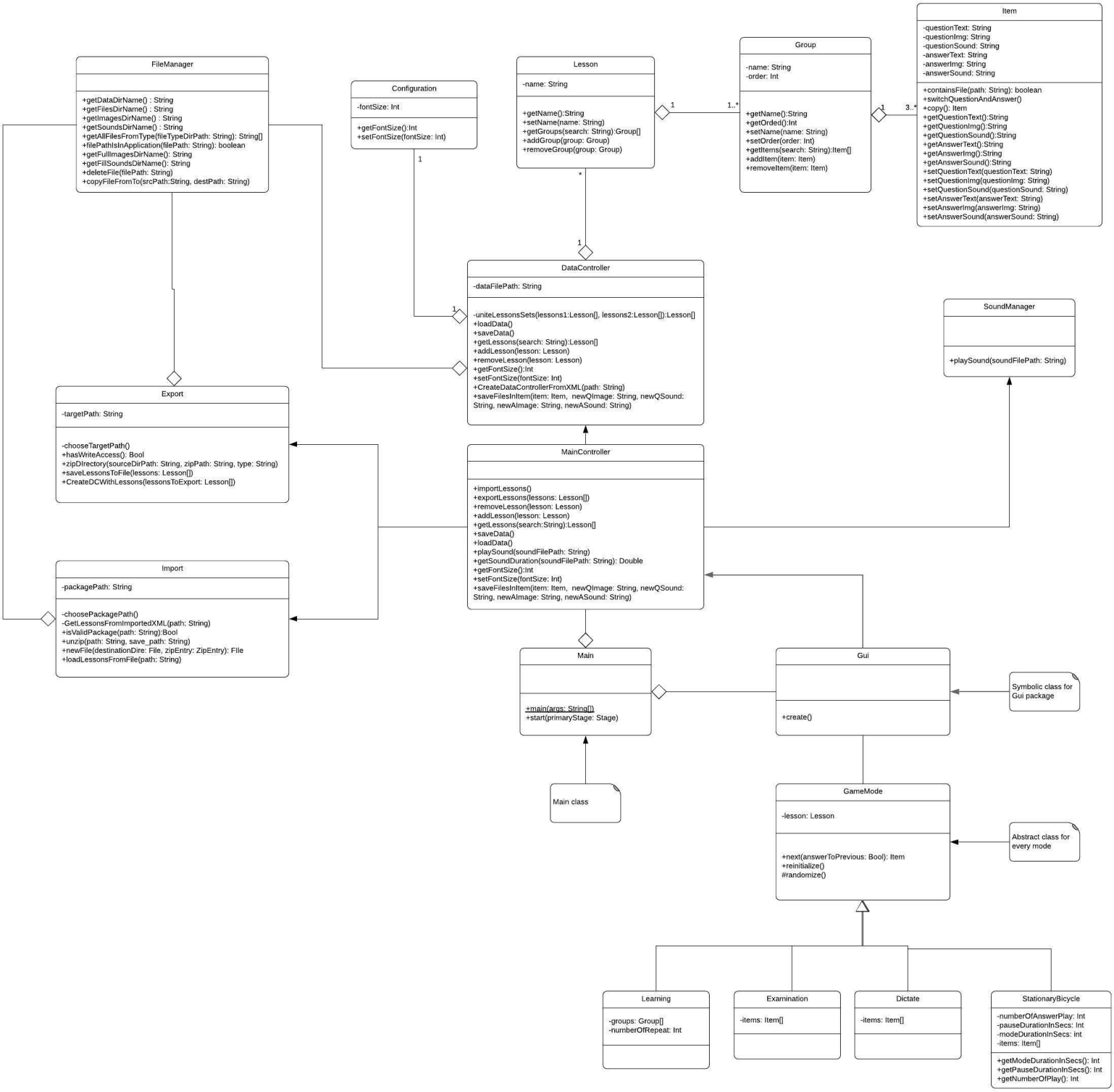
- Trieda FileManager:
  - ✓ Bude obsahovať metódy na manipuláciu so súbormi.
  - ✓ Metódy:
    - getDataDirName() - vráti názov priečinku obsahujúceho dáta
    - getFilesDirName() - vráti názov priečinku obsahujúceho súbory
    - getImagesDirName() - vráti názov priečinku obsahujúceho obrázky
    - getSoundsDirName() - vráti názov priečinku obsahujúceho zvukové súbory
    - getAllFilesFromType(fileTypeDirPath) - vráti všetky súbory zvoleného typu
    - pathFileIsInApplication(filePath) - overí, či sa súbor nachádza v aplikácii
    - getFullImagesDirName() - vráti plnú cestu k priečinku s obrázkami

- getFullSoundsDirName() - vráti plnú cestu k priečinku so zvukmi
- deleteFile(filePath) - odstráni zvolený súbor
- copyFileFromTo(srcPath, destPath) - skopíruje zvolený súbor na vybranú destináciu

- SoundManager
  - ✓ Trieda bude slúžiť na prehrávanie zvukov na zvlášť threade.
  - ✓ Bude mať metódu playSound(soundFilePath), ktorá prehrá zadaný zvukový súbor.
  - ✓ Ďalej bude mať metódu sound_time(soundFilepath), ktorá vráti dĺžku trvania zvuku
- Trieda Config:
  - ✓ Trieda obsahuje údaje o nastavení aplikácii.
  - ✓ Obsahuje atribút fontSize - definuje veľkosť písma v celej aplikácii; obsahuje getter a setter pre tento atribút
- Trieda Lesson:
  - ✓ Trieda zodpovedá lekcii.
  - ✓ Obsahuje atribút name a minimálne jednu skupinu.
  - ✓ Obsahuje getter a setter pre atribút name
  - ✓ Ďalej obsahuje metódu na hľadanie, pridanie a vymazanie skupiny.
- Trieda Group.
  - ✓ Trieda zodpovedá skupine
  - ✓ Obsahuje atribúty name - názov, order - poradie v lekcii a minimalne tri položky
  - ✓ Obsahuje gettery, settery a metódy na vyhľadanie, vymazanie a pridanie položiek
- Trieda Item:
  - ✓ Zodpovedá položke
  - ✓ Obsahuje atribúty:
    - questionText - text otázky

- questionImg - obrázok otázky
- questionSound - zvuk otázky
- answerText - text odpovede
- answerImg - obrázok odpovede
- answerSound - zvuk odpovede

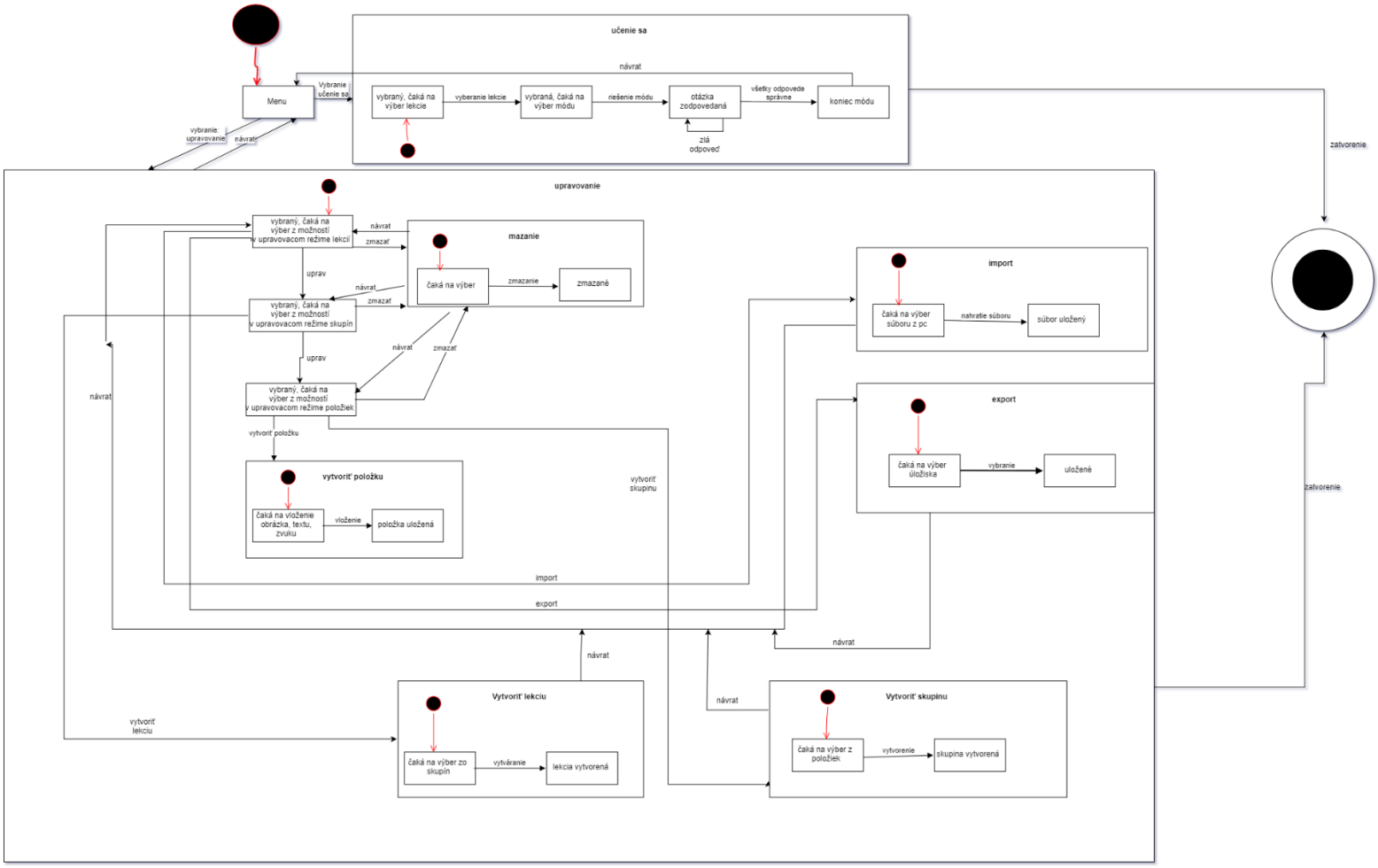
✓ Obsahuje gettery a settery pre tieto atribúty, metódu switchQuestionAndAnswer(), ktorá prehodí otázku a odpoveď v položke, metódu containsFile(path), ktorá overí, či položka obsahuje súbor, a metódu copy, ktorá vráti kópiu samej seba.


FileManager
+getDataDirName() : String
+getFilesDirName() : String
+getImagesDirName() : String
+getSoundsDirName() : String
+getAllFilesFromType(fileTypeDirPath: String): String[]
+filePathIsInApplication(filePath: String): boolean
+getFullImagesDirName(): String
+getFillSoundsDirName(): String
+deleteFile(filePath: String)
+copyFileFromTo(srcPath:String, destPath: String)
Configuration
-fontSize: Int
+getFontSize():Int
+setFontSize(fontSize: Int)
Lesson
-name: String
+getName():String
+setName(name: String)
+getGroups(search: String):Group[]
+addGroup(group: Group)
+removeGroup(group: Group)
Group
-name: String
-order: Int
+getName():String
+getOrded():Int
+setName(name: String)
+setOrder(order: Int)
+getItems(search: String):Item[]
+addItem(item: Item)
+removeItem(item: Item)
Item
-questionText: String
-questionImg: String
-questionSound: String
-answerText: String
-answerImg: String
-answerSound: String
+containsFile(path: String): boolean
+switchQuestionAndAnswer()
+copy(): Item
+getQuestionText():String
+getQuestionImg():String
+getQuestionSound():String
+getAnswerText():String
+getAnswerImg():String
+getAnswerSound():String
+setQuestionText(questionText: String)
+setQuestionImg(questionImg: String)
+setQuestionSound(questionSound: String)
+setAnswerText(answerText: String)
+setAnswerImg(answerImg: String)
+setAnswerSound(answerSound: String)
DataController
-dataFilePath: String
-uniteLessonsSets(lessons1:Lesson[], lessons2:Lesson[]):Lesson[]
+loadData()
+saveData()
+getLessons(search: String):Lesson[]
+addLesson(lesson: Lesson)
+removeLesson(lesson: Lesson)
+getFontSize():Int
+setFontSize(fontSize: Int)
+CreateDataControllerFromXML(path: String)
+saveFilesInItem(item: Item, newQImage: String, newQSound: String, newAImage: String, newASound: String)
SoundManager
+playSound(soundFilePath: String)
Export
-targetPath: String
-chooseTargetPath()
+hasWriteAccess(): Bool
+zipDirectory(sourceDirPath: String, zipPath: String, type: String)
+saveLessonsToFile(lessons: Lesson[])
+CreateDCWithLessons(lessonsToExport: Lesson[])
MainController
+importLessons()
+exportLessons(lessons: Lesson[])
+removeLesson(lesson: Lesson)
+addLesson(lesson: Lesson)
+getLessons(search:String):Lesson[]
+saveData()
+loadData()
+playSound(soundFilePath: String)
+getSoundDuration(soundFilePath: String): Double
+getFontSize():Int
+setFontSize(fontSize: Int)
+saveFilesInItem(item: Item, newQImage: String, newQSound: String, newAImage: String, newASound: String)
Import
-packagePath: String
-choosePackagePath()
-GetLessonsFromImportedXML(path: String)
+isValidPackage(path: String):Bool
+unzip(path: String, save_path: String)
+newFile(destinationDire: File, zipEntry: ZipEntry): File
+loadLessonsFromFile(path: String)
Main
+main(args: String[])
+start(primaryStage: Stage)
Gui
+create()
Symbolic class for Gui package
Main class
GameMode
-lesson: Lesson
+next(answerToPrevious: Bool): Item
+reinitialize()
#randomize()
Abstract class for every mode
Learning
-groups: Group[]
-numberOfRepeat: Int
Examination
-items: Item[]
Dictate
-items: Item[]
StationaryBicycle
-numberOfAnswerPlay: Int
-pauseDurationInSecs: Int
-modeDurationInSecs: int
-items: Item[]
+getModeDurationInSecs(): Int
+getPauseDurationInSecs(): Int
+getNumberOfPlay(): Int

Stavový diagram

- Diagram je nad entitou celej aplikácie, aby sme mohli ukázať všetky stavy, do ktorých sa aplikácia môže dostať
- Jednotlivé štvorce predstavujú nejaký proces, ktorý aplikácia vykonáva – štvorce sú iba na sprehľadnenie, aby sa dalo vidieť jeden samostatný proces
- Proces sa vždy začína stavom, do ktorého smeruje červená šípka
- Začiatok je vždy označený čiernym plným kruhom
- Koniec je označený čiernym kruhom s čiernou bodkou v strede
- Činnosti popisujúce prechody medzi stavmi vykonáva používateľ prostredníctvom používateľského rozhrania

Stavový diagram:



Testovanie jednotlivých častí aplikácie

a) Jednotlivé módy

Každý jeden mód sa bude testovať samostatne a bude musieť spĺňať požiadavky uvedené v katalógu. Tie otestujem tak, že každý mód spustím nad nejakou množinou testovacích dát a budem sledovať, čo sa deje. Budem sledovať najmä správne poradie otázok (ak nejde o náhodné poradie) a či algoritmus vykonáva presne to, čo má a čo je uvedené v katalógu požiadaviek.

b) Používateľské rozhranie

Upravenie fontu:

Po spustení aplikácie sa zobrazí hlavné menu. Keď sa v hlavnom menu klikne na ozubené koliesko, zobrazí sa vyskakovacie okno, v ktorom bude možné zmeniť veľkosť fontu. Po zmene a po potvrdení bude veľkosť fontu zmenená.

Pridanie lekcie s jednou skupinou a jednou položkou:

Keď sa v hlavnom menu klikne na upravovanie dát, otvorí sa zoznam všetkých lekcií. Po kliknutí na tlačidlo pridaj lekciu sa otvorí okno na pridanie lekcie. Vyplní sa názov lekcie a klikne sa na tlačidlo vytvor skupinu. Otvorí sa okno na vytváranie skupiny, zadá sa meno skupiny. Klikne sa na tlačidlo vytvor položku, otvorí sa okno s vytváraním položky. Do vstupu text otázky sa zadá text a klikne sa na priečinok vedľa vstupu na zvuk odpovede. Otvorí sa adresár, v tom sa vyberie vhodný zvukový súbor. Klikne sa na tlačidlo Ok, otvorí sa okno so skupinou, do ktorej sa vytvorila položka. Klikne sa na tlačidlo Ok, otvorí sa lekcia, do ktorej sa vytvorila skupina. Klikne sa na tlačidlo Ok a v zozname lekcií bude nová lekcia.

Editácie lekcie zmenou názvu a pridaním skupiny

Keď sa v hlavnom menu klikne na upravovanie dát, otvorí sa zoznam všetkých lekcií. Pri jednej z lekcií je tlačidlo uprav. Po kliknutí na toto

tlačidlo sa otvorí upravovacie okno tejto lekcie obsahujúce jej názov a skupiny. Prepíše sa názov, pridá sa nová skupina. Lekcia bude mať zmenený názov a o jednu skupinu viac.

Mazanie lekcie

Keď sa kline na tlačidlo vymazať pri lekcii v zozname všetkých lekcií v upravovaní dát, vyskočí okno ktoré sa spýta, či naozaj chceme túto lekciu vymazať. Po potvrdení lekcia už nebude v zozname lekcií.

Export

Keď sa kline na tlačidlo export pod zoznamom všetkých lekcií v upravovaní dát, otvorí sa okno ktoré bude obsahovať zoznam lekcií a pri každej bude checkbox. Označí sa jedna alebo viacej lekcii a klikne sa na export. Otvorí sa adresár a v ňom sa vyberie cieľový priečinok, do ktorého chcem uložiť exportované dáta. Po potvrdení bude v tomto priečinku exportovaný súbor s vybranými lekciami.

Import

Keď sa kline na tlačidlo import pod zoznamom všetkých lekcií v upravovaní dát, otvorí sa adresár. Vyberie sa súbor, ktorý sa ide importovať. Ak je tento súbor správny (má správnu štruktúru), v zozname lekcii pribudne jedna alebo viac nových lekcií. Keď všetky lekcie, ktoré boli importované už aplikácie obsahuje, nepribudne žiadna. Ak má importovaný súbor zlý formát, import neprebehne a na upozorní sa na to alertom.

Spustenie módu na lekcii

Keď sa v hlavnom menu klikne na tlačidlo štart, otvorí sa zoznam všetkých lekcií. Po vybratí jednej z nich sa ukáže okno s výberom štyroch módov. Ak sa vyberie mód učenie, vyskočí okno, v ktorom sa bude dať nastaviť počet opakovaní skupín v lekcii, po ktorom sa skupina označí ako prebraná. Ak sa vyberie mód stacionárny bicykel, vyskočí okno, v ktorom sa bude dať nastaviť dĺžka trvania módu v sekundách, počet prehraní zvuku odpovede (ak má odpoveď zvuk) a

dĺžka trvania zobrazenia otázky v sekundách. V každom móde sa zobrazujú položky a používateľ zadáva, či vedel alebo nevedel odpoveď na otázku. Keď mód skončí, zobrazia sa štatistiky. Po kliknutí na tlačidlo späť do menu sa otvorí hlavné menu.

c) MainController a ďalšie triedy

- Položka môže obsahovať text, obrázok alebo zvuk v otázke/odpovedi
- Skupina si vie udržiavať svoje meno a položky ktoré sú v nej
- Lekcia si vie udržiavať svoje meno a položky ktoré sú v nej
- Aplikácia vie pridávať a vymazávať lekcie/skupiny/položky
- Aplikácia si udržiava všetky hráčove lekcie a zároveň si každú zmenu v nich ukladá do svojho súboru v podobe XML
- Pri spustení si aplikácia načíta posledné uložené XML dáta do svojho súboru a vytvorí si z nej dátovú štruktúru s ktorou neskôr môže pracovať.
- Obrázky a zvuky ktoré aplikácia nemá vo svojich položkách si prekopíruje do „sounds“ priečinku a obrázky do „images“ priečinku
- Ak sú novo-nakopírovane súbory v konflikte(napr. názvom) so súbormi ktoré už aplikácia obsahuje, tak k nim pridá podtržník a zaň posledné poradové číslo súborov s rovnakým názvom, napr.: „stolicka_3“

d) Export, import a Sound Manager

<u>Export</u>

Budem testovať funkcie, ktoré budú slúžiť k exportu údajov z aplikácie. Čiže budem kontrolovať, či zadaná cesta, kde chce užívateľ uložiť údaje je možná na použitie, popr. či je vôbec správna. Ďalej budem kontrolovať, či si užívateľ vybral lekcie, ktoré chce exportovať. Potom, či sa daný súbor vytvoril a či nie je prázdny.

<u>Import</u>

Budem testovať funkcie, ktoré budú slúžiť k importu údajov z aplikácie. Užívateľ vyberie cestu odkiaľ chce importovať súbor. Najskôr skontrolujem, či daná cesta existuje. Potom skontrolujem, či daný súbor nie je prázdny a či obsahuje súbory potrebné pre funkčnosť v aplikácií. Dané súbory sa uložia do priečinku aplikácie. Skontrolujem vytvorenie xml súboru.

Sound Manager

Danou triedou spúšťam zvuk. Otestujem spúšťanie zvuku a jeho vyberanie z priečinka aplikácie. Budem kontrolovať, či zadaná cesta k zvuku je správna. A otestujem spúšťanie zvuku v aplikácii.