Návrh

Tím SZTS

2018/2019

28.10. 2018

1. Formáty súborov, s ktorými bude aplikácia pracovať:
    1.1. Formát zvuku .mp3
        1.1.1. Maximálna veľkosť zvuku v tomto formáte
    1.2. Formát zvuku .wav
        1.2.1. Maximálna veľkosť zvuku v tomto formáte
    1.3. Formát obrázku .png
        1.3.1. Maximálna veľkosť v pixeloch v tomto formáte
    1.4. Formát obrázku .jpg
        1.4.1. Maximálna veľkosť v pixeloch v tomto formáte
    1.5. XML súbor
        1.5.1. 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
   2.1. Dáta aplikácie budú uložené v priečinku data. Ten bude obsahovať:
      2.1.1. súbor data.xml – v ňom budú lekcie a nastavenia aplikácie
      2.1.2. priečinok files - v ňom budú priečinky images a sounds
         2.1.2.1. Priečinok image bude obsahovať:
            2.1.2.1.1. Obrázky
         2.1.2.2. Priečinok sounds bude obsahovať:
            2.1.2.2.1. Zvukové súbory
      2.1.3. Schéma 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>
```

2.1.4. Vysvetlenie:

2.1.4.1. Koreňový element bude element **data**.

2.1.4.2. Bude obsahovať elementy v poradí **nastavenia** a **lekcie**.

2.1.4.3. Element **nastavenia** bude obsahovať element **font_size**.

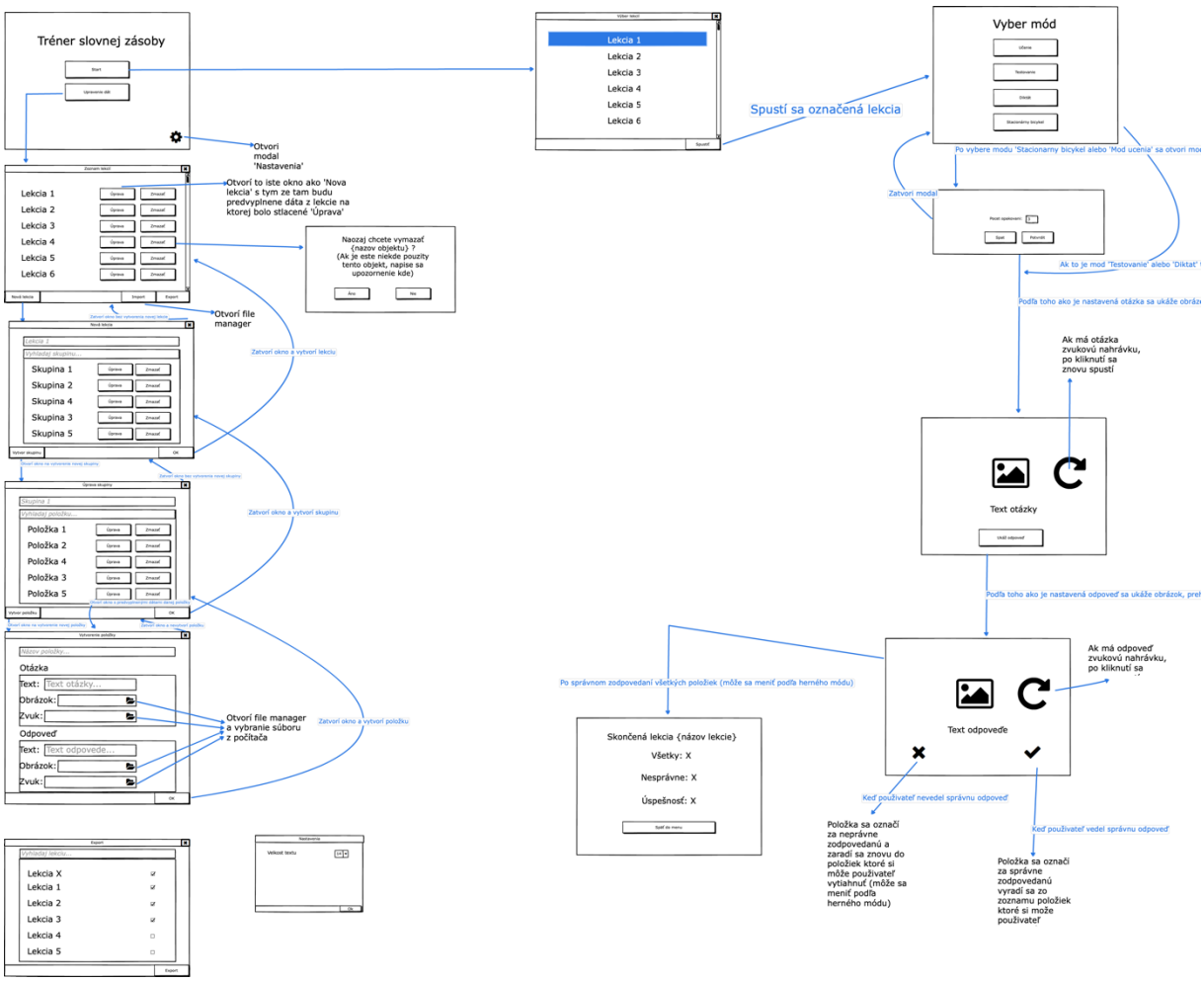
2.1.4.4. Element **lekcie** bude obsahovať aspoň jeden element **lekcia.**

2.1.4.5. **Lekcia** bude mať element **nazov** a element **skupiny** obsahujúci minimálne jednu **skupinu**.

2.1.4.6. **Skupina** bude mať elementy **nazov**, **poradie, aktivna** a element **položky** obsahujúci minimálne 3 **položky**.

2.1.4.7. **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 používateľského rozhrania



Existuje ešte aj .json a .bmpr (Balsamiq 3) verzia tohto návrhu

4. Návrh implementácie

4.1. Moduly

4.1.1. DataController

4.1.1.1. bude obsahovať metódy na vyberanie, vkladanie, updatovanie a mazanie údajov z dátového súboru.

4.1.2. FileManager

4.1.2.1. bude obsahovať metódy na premiestňovanie, pridávanie, mazanie súborov (obrázky, zvuky)

4.1.3. Export

4.1.3.1. Bude obsahovať metódu, ktorá z vybraných lekcií, skupín a položiek urobí balíček a ponúkne používateľovi, aby si ho uložil na vybrané miesto do zariadenia

4.1.3.2. Balíček (priečinok alebo zip) bude obsahovať súbor s dátami a priečinok s obrázkami a zvukmi

4.1.3.3. Bude používať DataController a FileManager

4.1.4. Import

4.1.4.1. Bude obsahovať metódu, ktorá uloží lekcie, skupiny a položky z balíčka

4.1.4.2. Dáta uloží do dátového súboru a súbory (obrázky, zvuky) na správne miesto

4.1.4.3. Balíček vznikol exportom

4.1.4.4. Balíček užívateľ vyberie pomocou FileChoosera

4.1.4.5. Bude používať moduly DbController a FileManager

4.1.5. MainController

4.1.5.1. Bude obsahovať všetky hlavné funkcie programu – pridávanie, mazanie, editovanie lekcii, skupín, položiek, prispôsobenie grafického rozhrania, ...

4.1.5.2. Bude používať moduly DbController, FileManager, Export, Import

4.1.6. GUI

4.1.6.1. Ponuka možností, zobrazuje dáta

4.1.6.2. Komponentami (buttony, checkboxy, ...) sa budú spúšťať funkcie z MainControllera

4.1.7. ModesController

4.1.7.1. Bude obsahovať algoritmy jednotlivých módov

4.1.7.2. Používa funkcie GUI na zobrazenie položiek, lekcií, ...

4.2. Technológie

4.2.1. Java

4.2.1.1. Výsledný program bude buildnutý do executable jar súboru

4.2.1.2. Obahuje všetky zdrojové súbory a knižnice projektu

4.2.1.3. Spustením celého programu sa spustí main

4.2.2. JavaFX Scene Builder

4.2.2.1. Týmto nástrojom bude realizované celé GUI

4.2.2.2. Vytvorí .fxml súbor, ktorý popisuje rozloženie grafických componentov

4.3. Komponent Diagram

4.3.1. Jednotlivé komponenty predstavujú základné triedy, z ktorých bude celý systém zostavený.

4.3.2. Komponenty DataController, FileManager, Import, Export budú spracovávať perzistentné dáta (xml, mp3, png).

4.3.3. Export a Import budú pre svoj chod využívať komponenty DataController a FileManager.

4.3.4. MainController bude riadiť všetky komponenty

4.3.5. 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)

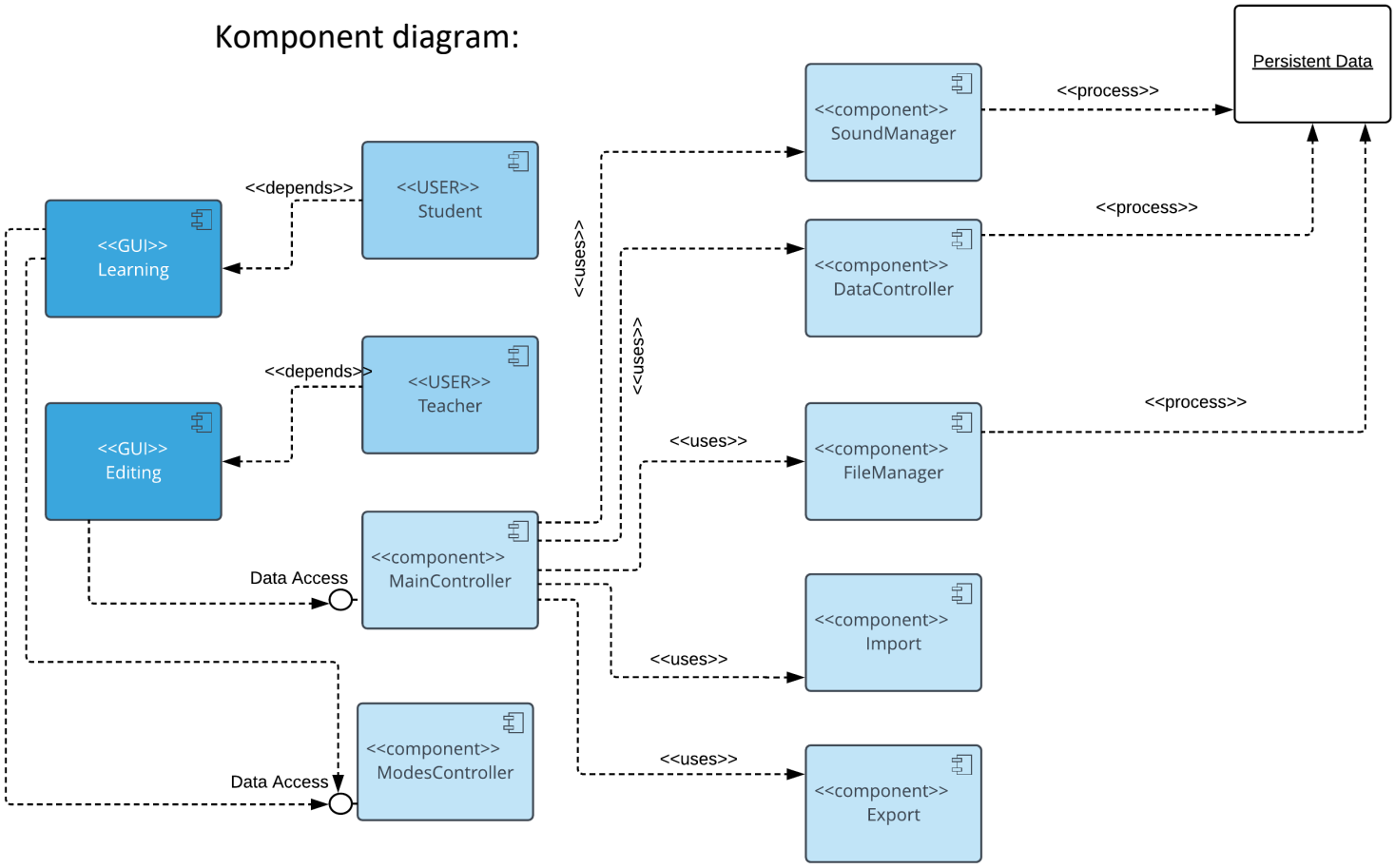
4.3.6. V systéme bude celé používateľské rozhranie ako jedna časť

4.3.7. Editovanie a Učenie plnia pokyny, ktoré mu zadáva používateľ, posiela dáta do MainControllera, ktorý ďalej spracováva pokyny

4.3.8. ModesController zobrazuje jednotlivé módy učenia v používateľskom rozhraní (Učenie)

4.3.9. SoundManager komunikuje s perzistentnými dátami, pokyny mu bude posielať MainController

Komponent diagram:



4.4. Triedny diagram

4.4.1. Trieda Main

4.4.1.1. Je to hlavná trieda programu

4.4.1.2. Obsahuje metódu main(String[] args), ktorá sa spustí ako prvá

4.4.2. Trieda GUI

4.4.2.1. Je to symbolická trieda pre všetky Gui controllery

4.4.2.2. Bude vytvorený Gui controller pre každú scénu v aplikácii

4.4.2.3. Bude riadiť dáta a grafické komponenty pre svoju scénu

4.4.3. Trieda Mode

4.4.3.1. Abstraktná trieda pre všetky štyri módy

4.4.3.2. Obsahuje atribút lesson - je to lekcia, nad ktorou je mód spustený

4.4.3.3. Metódy

4.4.3.3.1. next(answer) - Tato metóda bude implementovaná v každom móde inak

4.4.3.3.1.1. Dostane odpoveď na predchádzajúcu otázku (true - dobre, false - zle) a podľa algoritmu daného módu vráti ďalšiu položku

4.4.3.3.2. randomize() - Zamieša obsah preberanej lekcie

4.4.4. Trieda MainController

4.4.4.1. Táto trieda je hlavné prepojenie medzi GUI controllermi a triedami určenými na prácu s dátami

4.4.4.2. Metódy

4.4.4.2.1. 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

4.4.4.2.2. 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

4.4.4.2.3. removeLesson(lesson) - pomocou DataControllera odstráni lekciu z načítaných lekcií

4.4.4.2.4. addLesson(lesson) - pomocou DataControllera pridá lekciu do načitaných lekcií

4.4.4.2.5. getLessons(search) - pomocou DataControllera získa zoznam lekcií zodpovedajúcich vyhľadávajúcemu reťazcu

4.4.4.2.6. saveData() - pomocou DataControllera uloží načítané (a v aplikácii upravované) dáta (lekcie a nastavenia) do dátového súboru aplikácie

4.4.4.2.7. playSound(soundFilePath) - použitím triedy SoundManager prehrá zvukový súbor

4.4.4.2.8. getFontSize() - z DataControllera získa veľkosť fontu a vráti ho

4.4.4.2.9. setFontSize(fontSize) - pomocou DataControllera nastaví veľkosť fontu v aplikácii

4.4.5. Trieda Import

4.4.5.1. Trieda predstavuje importovaný package

4.4.5.2. V konštruktore triedy sa vyberie vstupný package

4.4.5.3. Atribút packagePath je cesta k tomuto vstupnému súboru

4.4.5.4. Metódy

4.4.5.4.1. choosePackagePath() - otvorí pre používateľa FileChooser, kde úlohou používateľa je vybrať vstupný súbor; cesta k tomuto súboru sa uloží do triedneho atribútu packagePath

4.4.5.4.2. isValidPackage() - overí, či štruktúra vstupného súboru je správna, je zadefinovaná v návrhu perzistentných dát; metóda vráti true, ak je správna, inak false

4.4.5.4.3. getDataFilePath() - metóda vráti cestu k .xml súboru

4.4.5.4.4. getFilesDirPath() - metóda vráti cestu k priečinku, ktorý obsahuje obrázky a zvukové súbory

4.4.6. Trieda Export

4.4.6.1. Predstavuje exportovaný package

4.4.6.2. V konštruktore triedy sa vyberie cesta, na ktorú uloží výstupný package

4.4.6.3. Atribút packetPath je cesta k výstupnému súboru

4.4.6.4. Metódy

4.4.6.4.1. 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

4.4.6.4.2. hasWriteAccess() – Vráti true, ak v cieľovom priečinku má program právo na písanie, vráti true, ak v cieľovom priečinku má program právo na písanie, inak vráti false

4.4.6.4.3. getDataFilePath() – metóda vráti cestu k .xml súboru, do ktorého sa bude písať

4.4.6.4.4. .getFilesDirPath() - metóda vráti cestu k priečinku, ktorý bude obsahovať obrázky a zvukové súbory

4.4.7. Trieda DataController

4.4.7.1. Bude slúžiť manipuláciu s perzistentnými dátami

4.4.7.2. Bude ich načítavať, zapisovať a načítané dáta bude u seba držať

4.4.7.3. Atribút dataFilePath je konštanta obsahujúca cestu k dátovému priečinku

4.4.7.4. Metódy

4.4.7.4.1. uniteLessonsSets(lessons1, lessons2) - zjednotí 2 kolekcie lekcií a vráti výsledok; bude potrebná pri importovaní lekcii, keď aplikácia už bude mať nejaké importované lekcie obsahovať

4.4.7.4.2. loadData() - načíta perzistentné dáta; bude volaná v konštruktore

4.4.7.4.3. loadLessonsFromFile(path) - načíta a vráti lekcie zo súboru; bude použitá v metóde loadData() a pri importe

4.4.7.4.4. saveData() - uloží údaje načítané v aplikácii do perzistentných dát; bude používaná pri editácii

4.4.7.4.5. saveLessonsToFile(path, lessons) - uloží lekcie do súboru; bude požitá v metóde saveData() a pri exporte

4.4.7.4.6. getLessons(search) - vráti lekcie zodpovedajúce vyhľadávajúcemu reťazcu

4.4.7.4.7. addLesson(lesson) - pridá lekciu

4.4.7.4.8. removeLesson(lesson) - odstráni lekciu

4.4.7.4.9. getFondSize() - vráti veľkosť fontu, ktorá je nastavená v aplikácii

4.4.7.4.10. setFondSize(fontSize) - nastaví aplikácii novú hodnotu veľkosti fontu

4.4.8. Trieda FileManager:

4.4.8.1. Bude obsahovať metódy na manipuláciu so súbormi.

4.4.8.2. Metódy:

4.4.8.2.1. moveFile(filePath, targetDir) - premiestni súbor do cieľového priečinka
copyFile(filePath, targetDir) - skopíruje a premiestni súbor do cieľového priečinkadeleteFile(filePath) - odstráni súbor

4.4.9. SoundManager

4.4.9.1. Trieda bude slúžiť na prehrávanie zvukov na zvlášť threade.

4.4.9.2. Bude mať metódu playSound(soundFilePath), ktorá prehrá zadaný zvukový súbor.

4.4.10. Trieda Config:

4.4.10.1. Trieda obsahuje údaje o nastavení aplikácii.

4.4.10.2. Obsahuje atribút fontSize - definuje veľkosť písma v celej aplikácii; obsahuje getter a setter pre tento atribút

4.4.11. Trieda Lesson:

4.4.11.1. Trieda zodpovedá lekcii.

4.4.11.2. Obsahuje atribút name a minimálne jednu skupinu.

4.4.11.3. Obsahuje getter a setter pre atribút name

4.4.11.4. Ďalej obsahuje metódu na hľadanie, pridanie a vymazanie skupiny.

4.4.12. Trieda Group.

4.4.12.1. Trieda zodpovedá skupine

4.4.12.2. Obsahuje atribúty name - názov, order - poradie v lekcii a minimalne tri položky

4.4.12.3. Obsahuje gettery, settery a metódy na vyhľadanie, vymazanie a pridanie položiek

4.4.13. Trieda Item:

4.4.13.1. Zodpovedá položke

4.4.13.2. Obsahuje atribúty:

4.4.13.2.1. questionText - text otázky

4.4.13.2.2. questionImg - obrázok otázky

4.4.13.2.3. questionSound - zvuk otázky

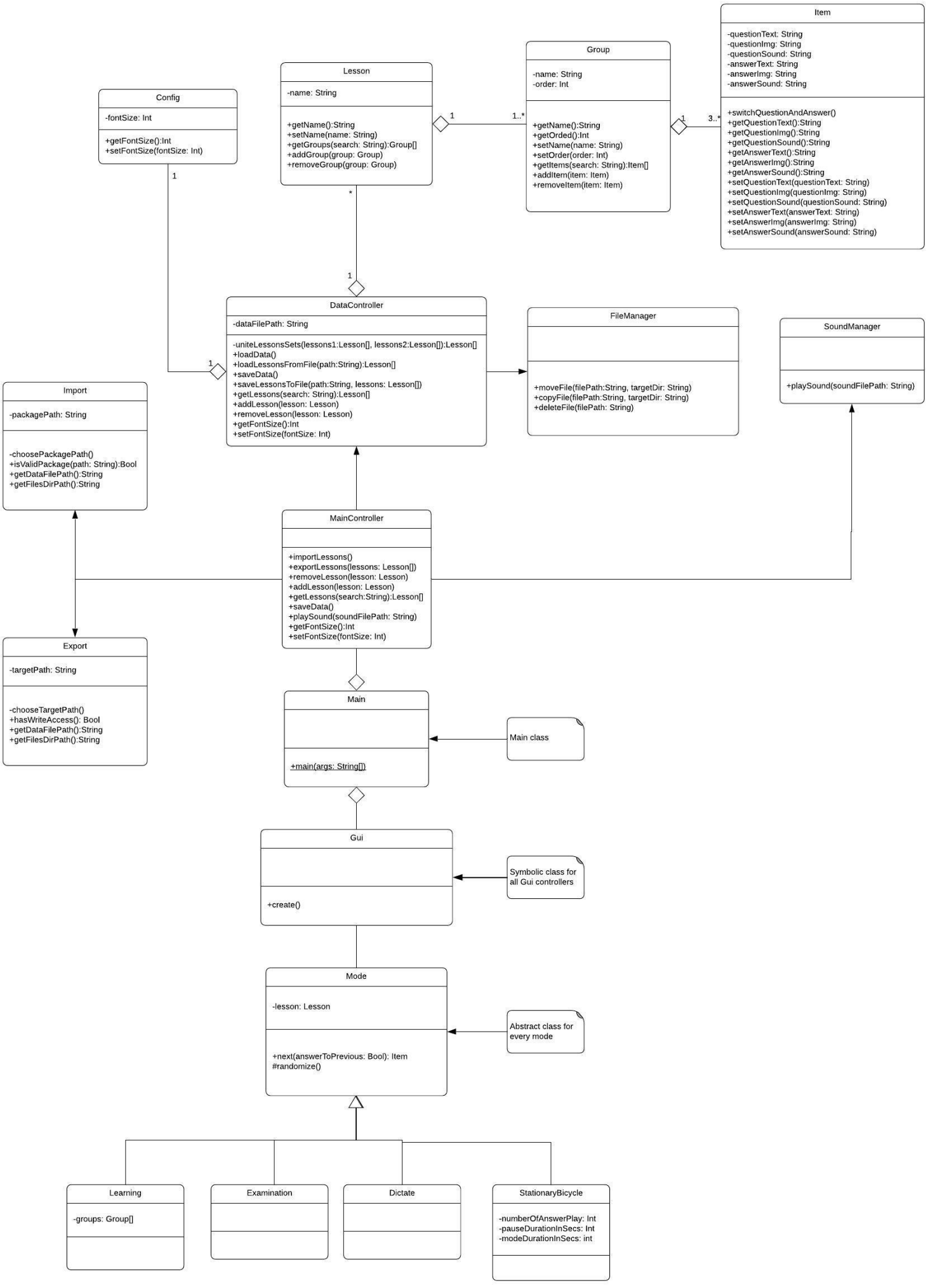
4.4.13.2.4. answerText - text odpovede

4.4.13.2.5. answerImg - obrázok odpovede

4.4.13.2.6. answerSound - zvuk odpovede

4.4.13.3. Obsahuje gettery a settery pre tieto atribúty a metódu switchQuestionAndAnswer(), ktorá prehodí otázku a odpoveď v položke

Na nasledujúcej strane je Triedny diagram:


Item
-questionText: String
-questionImg: String
-questionSound: String
-answerText: String
-answerImg: String
-answerSound: String
+switchQuestionAndAnswer()
+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)
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)
Lesson
-name: String
+getName():String
+setName(name: String)
+getGroups(search: String):Group[]
+addGroup(group: Group)
+removeGroup(group: Group)
Config
-fontSize: Int
+getFontSize():Int
+setFontSize(fontSize: Int)
DataController
-dataFilePath: String
-uniteLessonsSets(lessons1:Lesson[], lessons2:Lesson[]):Lesson[]
+loadData()
+loadLessonsFromFile(path:String):Lesson[]
+saveData()
+saveLessonsToFile(path:String, lessons: Lesson[])
+getLessons(search: String):Lesson[]
+addLesson(lesson: Lesson)
+removeLesson(lesson: Lesson)
+getFontSize():Int
+setFontSize(fontSize: Int)
FileManager
+moveFile(filePath:String, targetDir: String)
+copyFile(filePath:String, targetDir: String)
+deleteFile(filePath: String)
SoundManager
+playSound(soundFilePath: String)
Import
-packagePath: String
-choosePackagePath()
+isValidPackage(path: String):Bool
+getDataFilePath():String
+getFilesDirPath():String
MainController
+importLessons()
+exportLessons(lessons: Lesson[])
+removeLesson(lesson: Lesson)
+addLesson(lesson: Lesson)
+getLessons(search:String):Lesson[]
+saveData()
+playSound(soundFilePath: String)
+getFontSize():Int
+setFontSize(fontSize: Int)
Export
-targetPath: String
-chooseTargetPath()
+hasWriteAccess(): Bool
+getDataFilePath():String
+getFilesDirPath():String
Main
+main(args: String[])
Main class
Gui
+create()
Symbolic class for all Gui controllers
Mode
-lesson: Lesson
+next(answerToPrevious: Bool): Item
#randomize()
Abstract class for every mode
Learning
-groups: Group[]
Examination
Dictate
StationaryBicycle
-numberOfAnswerPlay: Int
-pauseDurationInSecs: Int
-modeDurationInSecs: int
1
1..*
3..*
*

4.5. Stavový diagram

4.5.1. Diagram je nad entitou celej aplikácie, aby sme mohli ukázať všetky stavy, do ktorých sa aplikácia môže dostať

4.5.2. Jednotlivé štvorce predstavujú nejaký proces, ktorý aplikácia vykonáva – štvorce sú iba na sprehľadnenie, aby sa dalo vidieť jeden samostatný proces

4.5.3. Proces sa vždy začína stavom, do ktorého smeruje červená šípka

4.5.4. Začiatok je vždy označený čiernym plným kruhom

4.5.5. Koniec je označený čiernym kruhom s čiernou bodkou v strede

4.5.6. Činnosti popisujúce prechody medzi stavmi vykonáva používateľ prostredníctvom používateľského rozhrania

Stavový diagram: