**Dokumentáció**

Felhasználói kézikönyv:

Amikor megnyitjuk a programot, az automatikusan betölti az eddigi mappáinkat és szettjeinket.

Az ablak bal oldalán elhelyezkedő menüben láthatjuk a mappákat és a szetteket.

A „Create folder” gombbal létrehozhatunk új mappákat, amiket, ha kiválasztunk (bal egér gombbal rákattintva), akkor szetteket hozhatunk bennük létre a „Create set” gombbal. Ha úgy próbálunk szettet létrehozni, hogy semmi sincsen kiválasztva, hibát fogunk kapni, amire egy felugró ablak is figyelmeztetni fog.

Az egyes szetteknek a tartalma a jobb oldalon jelenik meg, ahol a „Create new element” gombokra kattintva tudunk új kifejezés-definíció párokat beszúrni a lista végére, illetve az „X” gombokra kattintva tudjuk törölni az egyes párokat.

A „Delete” gombbal elemeket törölhetünk a kijelzett struktúrából. Ha egy szett van kiválasztva, akkor csak a szettet, ha egy mappa, akkor a mappát és minden benne elhelyezkedő szettet is törölni fogunk.

A bal menü alján elhelyezkedő három gombbal különböző féleképpen mérhetjük fel a tudásunkat.

A „Flashcards” gombbal egy tanuló kártya jelenik meg amin egy kifejezés van. Ha rákattintunk a kártyára, akkor az felfedi a kifejezéshez tartozó definíciót. Ha jól tudtuk (ezt mindenki magának dönti el), akkor rányomhatunk az „I got it” gombra és a kifejezés bekerül egy listába, ahol azok a kifejezések vannak, amiket már egyszer eltaláltunk. Ezen kifejezésekkel még egyszer fogunk találkozni, és ha még egyszer eltaláljuk, akkor véglegesen megtanultuk azt. Ha netán nem találnánk el a definíciót, akkor a „Dunno” gombra kattintva fejezhetjük ki, hogy ne történjen semmi és tegye vissza a tanuló kártyát annak a listának a végére, amiből kiszedtük.

A „Learn” gombbal is hasonló módon kétszer kell eltalálnunk a kifejezéshez tartozó definíciót vagy éppen fordítva ahhoz, hogy teljesen megtanuljuk azt. Először egy feleletválasztós feladatban kell kiválasztanunk a megfelelő definíciót, azután pedig, egy definícióhoz kell leírnunk a kifejezést egy írásos feladatban. Mindkét esetben a „Validate” gombra kattintva tudjuk meg, hogy helyesen válaszoltunk-e.

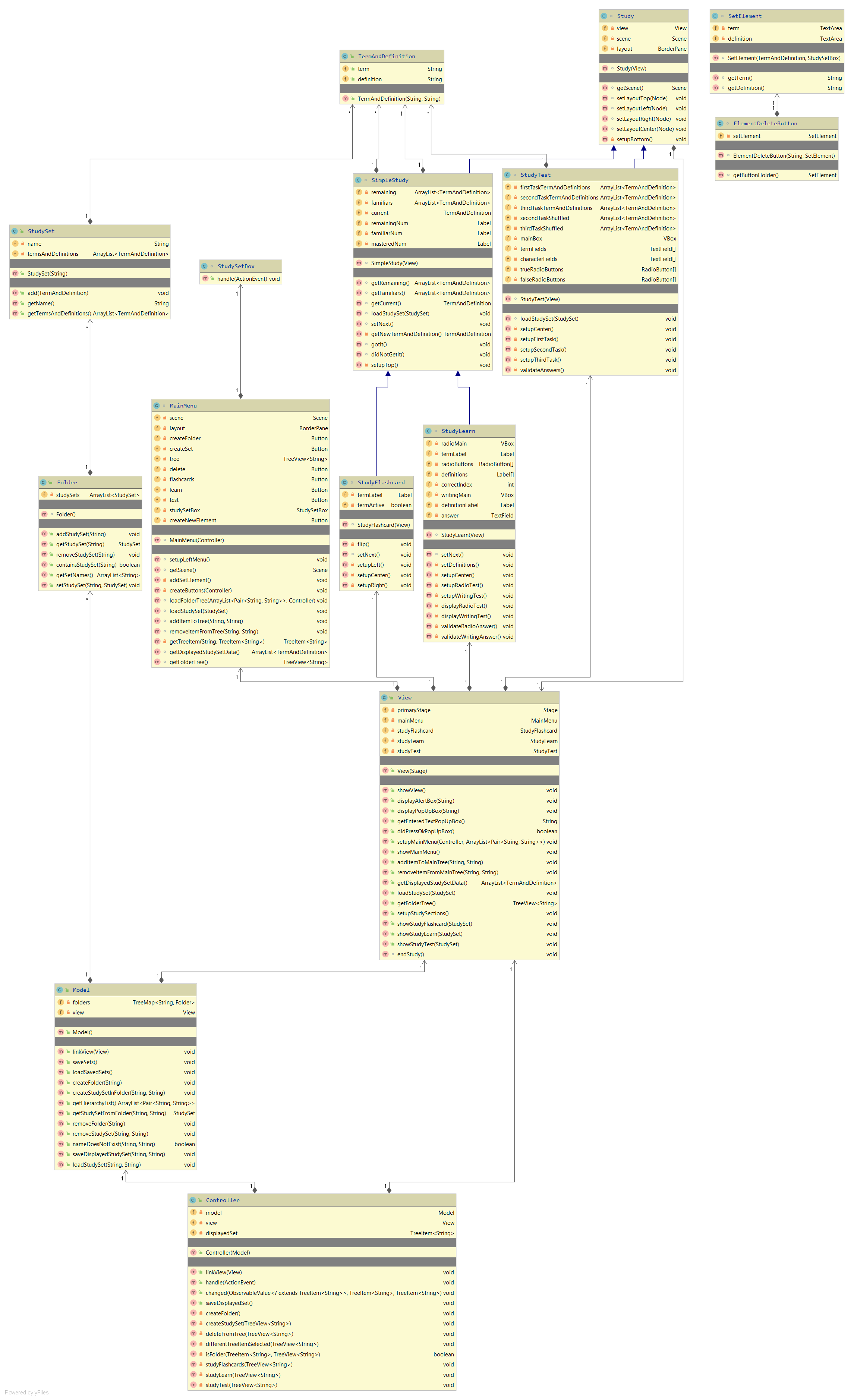
A „Test” gombbal egy tesztet generálhatunk magunknak, amiben van egy írásos, egy párosítós és egy igaz hamis feladat. Mindegy 5-5 kérdéssel. Mindezek végén a „Validate” gombra nyomva megnézhetjük, hogy mennyire tudjuk már az anyagot, hogy hány pontot kaptunk a 15-ből.

Mindegyik tanulási módszerből úgy tudunk kilépni, ha megnyomjuk az „I’m out” gombot az ablak alján.

Az egyes mappákat és szetteket a program bezáráskor menti le.

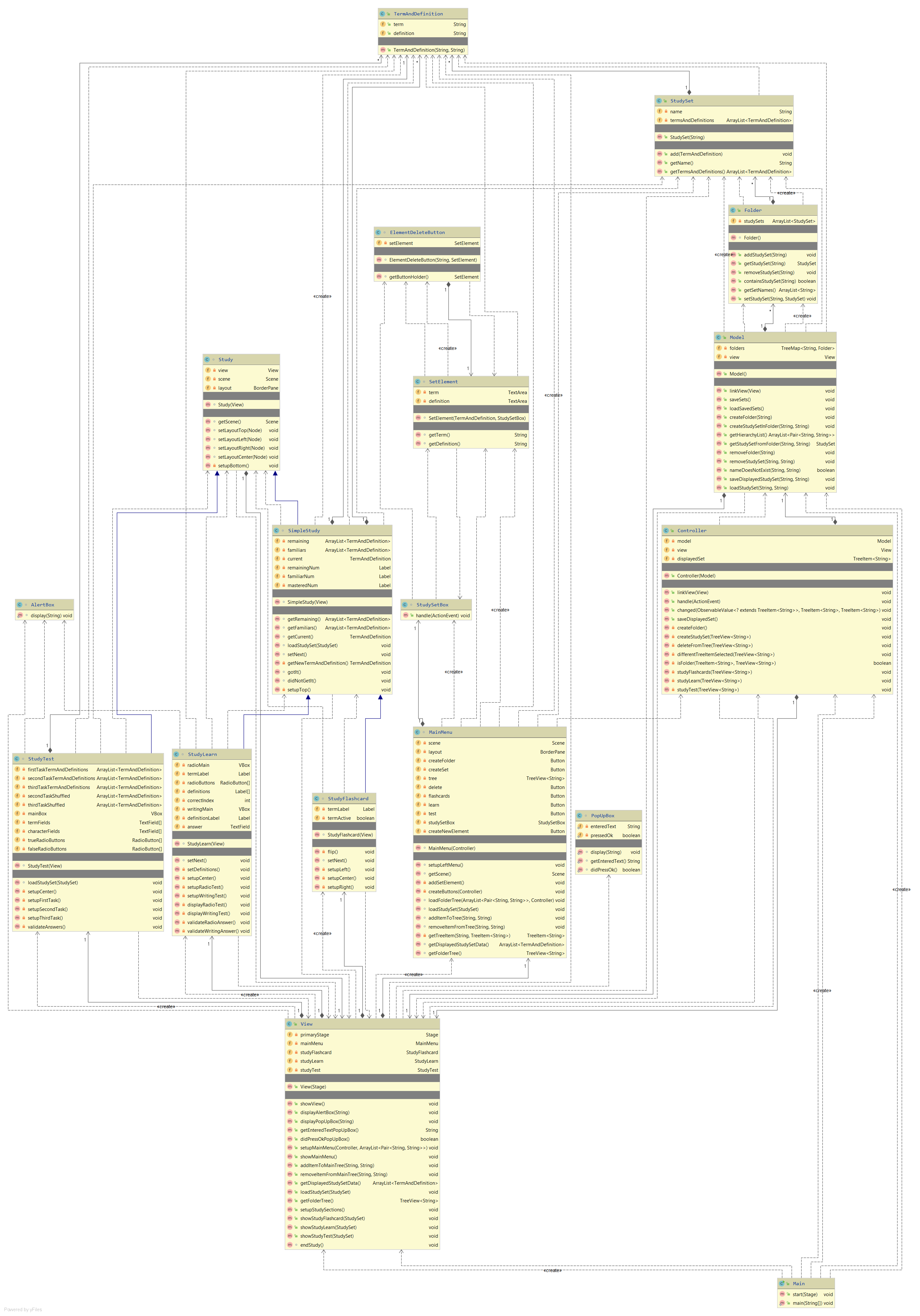
Programozói dokumentáció:

**Osztály diagramok:**

**Kisebb átlátható diagram hívások nélkül**

**(azon osztályok nincsenek rajta melyeknél nem volt él)**

**Teljes diagram**

****

**Osztályok és metódusaik:**

AlertBox

/\*\*

\* Megjelenit egy felugro ablakot a megadott szoveggel.

\* @param title megjelenitett szoveg

\*/

**static void display(String title)**

Controller

/\*\*

\* Beallitja a modell referenciat a kesobbi kommunikaciohoz.

\* @param model modell

\*/

**public Controller(Model model)**

/\*\*

\* Beallitja a nezet referenciat.

\* @param view nezet

\*/

**public void linkView(View view)**

/\*\*

\* Kezeli az egyes gombnyomasokat.

\* @param actionEvent esemeny

\*/

@Override

**public void handle(ActionEvent actionEvent)**

/\*\*

\* Kezeli az egyes valtozasokat amik a kijelzett mappa fan tortennek.

\* @param observableValue megfigyelheto ertek

\* @param old\_val regi ertek

\* @param new\_val uj ertek

\*/

@Override

**public void changed(ObservableValue<? extends TreeItem<String>> observableValue, TreeItem<String> old\_val, TreeItem<String> new\_val)**

/\*\*

\* Kezeli az egyes valtozasokat amik a kijelzett mappa fan tortennek.

\* @param observableValue megfigyelheto ertek

\* @param old\_val regi ertek

\* @param new\_val uj ertek

\*/

@Override

**public void changed(ObservableValue<? extends TreeItem<String>> observableValue, TreeItem<String> old\_val, TreeItem<String> new\_val)**

/\*\*

\* Az aktualisan kijelzett szettet elmenti a modellben.

\*/

**public void saveDisplayedSet()**

/\*\*

\* Letrehoz egy uj mappat a modellben, ami aztan frissiti a kijelzett fat is.

\*/

**private void createFolder()**

/\*\*

\* Letrehoz egy uj szettet a modellben, ami aztan frissiti a kijelzett fat is.

\* @param tree kijelzett fa struktura a mappakrol es szettekrol

\*/

**private void createStudySet(TreeView<String> tree)**

/\*\*

\*

\* @param tree kijelzett fa struktura a mappakrol es szettekrol

\*/

**private void deleteFromTree(TreeView<String> tree)**

/\*\*

\*

\* @param tree kijelzett fa struktura a mappakrol es szettekrol

\*/

**private void differentTreeItemSelected(TreeView<String> tree)**

/\*\*

\* Visszater azzal, hogy igaz-e, hogy a megadott elem egy mappa.

\* @param item a kerdeses elem

\* @param tree kijelzett fa struktura a mappakrol es szettekrol

\* @return igaz-e, hogy mappa a fenti elem

\*/

**private boolean isFolder(TreeItem<String> item, TreeView<String> tree)**

/\*\*

\* Elmenti az aktualisan kijelzett szettet, ezutan megjeleniti a StudyFlashcard reszt.

\* @param tree kijelzett fa struktura a mappakrol es szettekrol

\*/

**private void studyFlashcards(TreeView<String> tree)**

/\*\*

\* Elmenti az aktualisan kijelzett szettet, ezutan megjeleniti a StudyLearn reszt.

\* @param tree kijelzett fa struktura a mappakrol es szettekrol

\*/

**private void studyLearn(TreeView<String> tree)**

/\*\*

\* Elmenti az aktualisan kijelzett szettet, ezutan megjeleniti a StudyTest reszt.

\* @param tree kijelzett fa struktura a mappakrol es szettekrol

\*/

**private void studyTest(TreeView<String> tree)**

ElementDeleteButton

/\*\*

\* Beallitja a gomb szoveget, es, hogy melyik szett elemben van benne.

\* @param text szoveg

\* @param setElement szett elem

\*/

**ElementDeleteButton(String text, SetElement setElement)**

/\*\*

\* Visszater azzal a szett elemmel amelyikben a gomb elhelyezkedik.

\* @return szett elem

\*/

**SetElement getButtonHolder()**

Folder

/\*\*

\* Letrehozza az osztaly valtozoit.

\*/

**Folder()**

/\*\*

\* Hozzaad egy szettet a mappahoz.

\* @param nameOfSet szett neve

\*/

**public void addStudySet(String nameOfSet)**

/\*\*

\* Visszater a megadott nevu szettel, ha van olyan a mappaban.

\* @param setName szett neve

\* @return szett, ha ne letezik akkor null

\*/

**public StudySet getStudySet(String setName)**

/\*\*

\* Eltavolitja a megadott nevu szettet a mappabol.

\* @param setName szett neve

\*/

**public void removeStudySet(String setName)**

/\*\*

\* Visszater, hogy letezik-e a megadott szett a mappaban, vagy sem.

\* @param setName szett neve

\* @return igaz, ha benne van, kulonben hamis

\*/

**public boolean containsStudySet(String setName)**

/\*\*

\* Visszater egy listaval a mappaban levo szettek neveirol

\* @return lista

\*/

**public ArrayList<String> getSetNames()**

/\*\*

\* A megadott szettnevvel beteszi a megadott szettet a mappaba.

\* @param setName szett neve

\* @param studySet szett

\*/

**public void setStudySet(String setName, StudySet studySet)**

Main

/\*\*

\* Beallitja a Controller, Model, View harmast.

\* @param primaryStage szinter

\* @throws Exception kivetel (Sosem dob kivetelt, de benne hagyjuk, a szokas kedveert)

\*/

**@Override**

**public void start(Stage primaryStage) throws Exception**

/\*\*

\* Main fuggveny. Inicializalja a futtatasi kornyezetet.

\* @param args futtatasi argumentumok

\*/

**public static void main(String[] args)**

MainMenu

/\*\*

\* Beallitja az osztaly valtozoit es a megadott kontroller referenciat elmenti a kommunikaciohoz.

\* @param controller kontroller

\*/

**MainMenu(Controller controller)**

/\*\*

\* Beallitja a bal oldali menut.

\*/

**void setupLeftMenu()**

/\*\*

\* Getter a scene-re

\* @return scene

\*/

**Scene getScene()**

/\*\*

\* Hozzaad egy szett elemet az aktualisan kijelzett szetthez.

\*/

**private void addSetElement()**

/\*\*

\* Beallitja a gombok kinezetet es futtatott funkcioit.

\* @param controller a kontroller ami kezeli a gombok funkcioit

\*/

**private void createButtons(Controller controller)**

/\*\*

\* A megadott fa hierarchiabol letrehozza a kijelzett mappa fa strukturat es a fahoz hozzaadja a megadott kontrollert

\* mint figyelo.

\* @param treeHierarchy fa struktura

\* @param controller kontroller

\*/

**void loadFolderTree(ArrayList<Pair<String, String>> treeHierarchy, Controller controller)**

/\*\*

\* Betolti a megadott szettet a kepernyore.

\* @param studySet szett

\*/

**void loadStudySet(StudySet studySet)**

/\*\*

\* Hozzaadja a megadott elemet a megadott mappaba a kijelzett fa mappa strukturaba.

\* @param treeItemValue elem neve

\* @param folderName mappa neve

\* @throws NullPointerException ha egy szettet probalunk letrehozni ugy, hogy egyik elem sincsen kivalasztva

\*/

**void addItemToTree(String treeItemValue, String folderName) throws NullPointerException**

/\*\*

\* A megadott elemet a megadott mappabol eltavolitja a kifelzett fa mappa strukturabol.

\* @param treeItemValue elem neve

\* @param folderName mappa neve

\* @throws NullPointerException ha egy szettet probalunk eltavolitani ugy, hogy egyik elem sincsen kivalasztva

\*/

**void removeItemFromTree(String treeItemValue, String folderName) throws NullPointerException**

/\*\*

\* Visszater a megadott nevu elemmel a megadott mappabol a kijelzett fa mappa strukturabol.

\* @param treeItemValue elem neve

\* @param treeFolder mappa neve

\* @return elem

\*/

**private TreeItem<String> getTreeItem(String treeItemValue, TreeItem<String> treeFolder)**

/\*\*

\* Visszater az aktualisan kijelzett szett kifejezes-definicio parjaival.

\* @return kifejezes-definicio parok

\*/

**ArrayList<TermAndDefinition> getDisplayedStudySetData()**

/\*\*

\* Getter a kijelzett fara.

\* @return kijelzett fa

\*/

**TreeView<String> getFolderTree()**

Model

/\*\*

\* Beallitja az osztaly valtozoit.

\*/

**public Model()**

/\*\*

\* Beallitja a nezet referenciat.

\* @paramview nezet

\*/

**public void linkView(View view)**

/\*\*

\* Fajlba menti a szetteket es a mappakat.

\*/

**public void saveSets()**

/\*\*

\* Kiolvassa fajlbol az elmentett szettekeet es mappakat.

\*/

**public void loadSavedSets()**

/\*\*

\* Letrehoz egy magadott nevu mappat a strukturaban.

\* @param folderName mappa neve

\*/

**public void createFolder(String folderName)**

/\*\*

\* Letrehoz egy megadott nevu szettet a megadott nevu mappaban.

\* @param setName szett neve

\* @param folderName mappa neve

\*/

**public void createStudySetInFolder(String setName, String folderName)**

/\*\*

\* Letrehoz egy hierarchia listat a mappakbol es szettekbol.

\* @return hiererhia lista

\*/

**public ArrayList<Pair<String, String>> getHierarchyList()**

/\*\*

\* Visszater a megadott nevu szettel a megadott nevu mappabol.

\* @param setName szett neve

\* @param folderName mappa neve

\* @return szett

\*/

**public StudySet getStudySetFromFolder(String setName, String folderName)**

/\*\*

\* Eltavolitja a megadott nevu mappat es annak az osszes elemet a modelbol.

\* @param folderName mappa neve

\*/

**public void removeFolder(String folderName)**

/\*\*

\* Eltavolitja a megadott nevu szettet a megadott nevu mappabol a modelbol

\* @param setName szett neve

\* @param folderName mappa neve

\*/

**public void removeStudySet(String setName, String folderName)**

/\*\*

\* Megadja, hogy a megadott nevu elem letezik-e a megadott nevu mappaban.

\* @param name elem neve

\* @param folderName mappa neve

\* @return letezik-e

\*/

**public boolean nameDoesNotExist(String name, String folderName)**

/\*\*

\* Elmenti a modellben az eppen kijelzett szettet a megadott nevvel a megadott mappaba.

\* @param setName szett neve

\* @param folderName mappa neve

\*/

**public void saveDisplayedStudySet(String setName, String folderName)**

/\*\*

\* Betolti a nezetbe a megadott nevu szettet a megadott mappabol.

\* @param setName szett neve

\* @param folderName mappa neve

\*/

**public void loadStudySet(String setName, String folderName)**

PopUpBox

/\*\*

\* Megjelenit egy felugro ablakot a megadott szoveggel és egy bemeneti mezovel.

\* @param title megjelenitett szoveg

\*/

**static void display(String title)**

/\*\*

\* Getter a beirt szovegre.

\* @return beirt szoveg

\*/

**static String getEnteredText()**

/\*\*

\* Visszater, hogy meg lett-e nyomva az ok gomb.

\* @return meg lett-e nyomva

\*/

**static boolean didPressOk()**

SetElement

/\*\*

\* Létrehoz egy szett elemet a megadott kifejezes-definicio parral a megadott kontenerben.

\* @param tad kifejezes-definicio par

\* @param studySetBox kontener

\*/

**SetElement(TermAndDefinition tad, StudySetBox studySetBox)**

/\*\*

\* Getter a kifejezesre.

\* @return kifejezes

\*/

**String getTerm()**

/\*\*

\* Getter a definiciora.

\* @return definicio

\*/

**String getDefinition()**

SimpleStudy

/\*\*

\* Beallitja a megadott nezetet az osben es beallitja a felso menu savot.

\* @param view nezet

\*/

**SimpleStudy(View view)**

/\*\*

\* Getter a maradek listara.

\* @return maradek lista

\*/

**ArrayList<TermAndDefinition> getRemaining()**

/\*\*

\* Getter az ismeros listara.

\* @return ismeros lista

\*/

**ArrayList<TermAndDefinition> getFamiliars()**

/\*\*

\* Getter az aktualis kifejezes-definicio parra.

\* @return kifejezes-definicio par

\*/

**TermAndDefinition getCurrent()**

/\*\*

\* Betolti a megadott szettet a listakba es betolt egy kezdo part

\* @param studySet szett

\*/

**void loadStudySet(StudySet studySet)**

/\*\*

\* Beallitja a kovetkezo kifejezes-definicio part

\*/

**void setNext()**

/\*\*

\* Visszater egy algoritmus alapjan vagy a maradek listabol vagy az ismeros listabol egy parral

\* @return kifejezes-definicio par

\*/

**private TermAndDefinition getNewTermAndDefinition()**

/\*\*

\* Athelyezi az aktualis part a megfelelo listaba, beallitja a szamlalokat, ezutan beallitja a kovetkezo part.

\*/

**void gotIt()**

/\*\*

\* Athelyezi az aktualis part a megfelelo listaban a lista vegere, ezutan beallitja a kovetkezo part.

\*/

**void didNotGetIt()**

/\*\*

\* Beallitja a felso menusavot.

\*/

**private void setupTop()**

Study

/\*\*

\* Beallitja a nezetet a kommunikaciohoz és beallitja az also menu savot.

\* @param view nezet

\*/

**Study(View view)**

/\*\*

\* Getter a scene-re

\* @return scene

\*/

**Scene getScene()**

/\*\*

\* Setter az ablak felso reszere.

\* @param node elrendezes

\*/

**void setLayoutTop(Node node)**

/\*\*

\* Setter az ablak bal reszere.

\* @param node elrendezes

\*/

**void setLayoutLeft(Node node)**

/\*\*

\* Setter az ablak jobb reszere.

\* @param node elrendezes

\*/

**void setLayoutRight(Node node)**

/\*\*

\* Setter az ablak kozepso reszere.

\* @param node elrendezes

\*/

**void setLayoutCenter(Node node)**

/\*\*

\* Beallitja az ablak also reszet.

\*/

**private void setupBottom()**

StudyFlashcard

/\*\*

\* Beallitja az ost a megadott nezettel és letrehozza az ablak bal, kozepso és jobb reszet.

\* @param view nezet

\*/

**StudyFlashcard(View view)**

/\*\*

\* Kifejezesrol definiciora, definiciorol kifejezesre allitja a kifelzett szoveget.

\*/

**private void flip()**

/\*\*

\* Beallitja a kovetkezo kifejezes-definicio part.

\*/

**@Override**

**void setNext()**

/\*\*

\* Beallitja az ablak bal reszet

\*/

**private void setupLeft()**

/\*\*

\* Beallitja az abalak kozepso reszet.

\*/

**private void setupCenter()**

/\*\*

\* Beallitja az ablak jobb reszet

\*/

**private void setupRight()**

StudyLearn

/\*\*

\* Beallitja azt ost a megadott nezettel egy beallitja az ablak kozepso reszet

\* @param view nezet

\*/

**StudyLearn(View view)**

/\*\*

\* Bealltja a kovetkezo kifejezes-definicio part

\*/

**@Override**

**void setNext()**

/\*\*

\* A feleletvalasztos resznel beallitja a definiciokat osszekeverve.

\*/

**private void setDefinitions()**

/\*\*

\* Beallitja az ablak kozepso reszen elhelyezkedo feladat reszeket

\*/

**private void setupCenter()**

/\*\*

\* Beallitja a feleletvalasztos feladat kinezetet.

\*/

**private void setupRadioTest()**

/\*\*

\* Beallitja az irasos feladat kinezetet.

\*/

**private void setupWritingTest()**

/\*\*

\* A feleletvalaszot feladatot allitja be az ablak kozepso reszere.

\*/

**private void displayRadioTest()**

/\*\*

\* Az irasos feladatot allitja be az ablak kozepso reszere.

\*/

**private void displayWritingTest()**

/\*\*

\* Ellenorzi a feleletvalasztos feladat megoldasat.

\*/

**private void validateRadioAnswer()**

/\*\*

\* Ellenorzi az irasos feladat megoldasat.

\*/

**private void validateWritingAnswer()**

StudySet

/\*\*

\* Letrehozza és beallitja a szett nevet.

\* @param name szett neve

\*/

**public StudySet(String name)**

/\*\*

\* Hozzaad egy kifejezes-definicio part a szetthez.

\* @param tad kifejezes-definicio par

\*/

**public void add(TermAndDefinition tad)**

/\*\*

\* Gette a szett nevére.

\* @return szett neve

\*/

**public String getName()**

/\*\*

\* Visszater egy listaval amiben benne van a szettben levo osszes kifejezes-definicio par

\* @return lista a kifejezes-definicio parokkal

\*/

**public ArrayList<TermAndDefinition> getTermsAndDefinitions()**

StudySetBox

/\*\*

\* Torli azt a szett elemet amibol hivas tortent.

\* @param actionEvent gomb esemény

\*/

**@Override**

**public void handle(ActionEvent actionEvent)**

StudyTest

/\*\*

\* Beallitja az ost a megadott nezettel es beallitja a osztaly valtozoit.

\* @param view nezet

\*/

**StudyTest(View view)**

/\*\*

\* Beolti a megadott szettet es beallitja az egyes feladatot listait.

\* @param studySet szett

\*/

**void loadStudySet(StudySet studySet)**

/\*\*

\* Beallitja az ablak kozepso reszet.

\*/

**private void setupCenter()**

/\*\*

\* Letrehozza az elso feladat kinezetet

\*/

**private void setupFirstTask()**

/\*\*

\* Letrehozza a masodik feladat kinezetet

\*/

**private void setupSecondTask()**

/\*\*

\* Letrehozza a harmadik feladat kinezetet.

\*/

**private void setupThirdTask()**

/\*\*

\* Lepontozza a harom feladatot es egy kulon ablakban kijelzi az eredmenyt.

\*/

**private void validateAnswers()**

TermAndDefinition

/\*\*

\* Beallitja az osztalyban tarolt kifejezest es definiciot.

\* @param term kifejezes

\* @param definition definicio

\*/

**public TermAndDefinition(String term, String definition)**

View

/\*\*

\* Beallitja a nezetben a szintert es beallitja az ablak cimet.

\* @param primaryStage szinter.

\*/

**public View(Stage primaryStage)**

/\*\*

\* Kijelzi a nezetet.

\*/

**void showView()**

/\*\*

\* Megjelenit egy felugro ablakot a megadott szoveggel.

\* @param text megjelenitett szoveg

\*/

**public void displayAlertBox(String text)**

/\*\*

\* Megjelenit egy felugro ablakot a megadott szoveggel és egy bemeneti mezovel.

\* @param text megjelenitett szoveg

\*/

**public void displayPopUpBox(String text)**

/\*\*

\* Visszater a megjelenitett ablakban beirt szoveggel.

\* @return beirt szoveg

\*/

**public String getEnteredTextPopUpBox()**

/\*\*

\* Visszater, hogy meg lett-e nyomva a felugro ablakban az ok gomb.

\* @return meg lett-e nyomva

\*/

**public boolean didPressOkPopUpBox()**

/\*\*

\* Beallitja a fomenut a megadott hierarchia lista alapjan es megadja a fomenunek a kontrollert a gomb nyomasok

\* lekezelesehez.

\* @param controller kontroller

\* @param hierarchyList hierarchia lista

\*/

**public void setupMainMenu(Controller controller, ArrayList<Pair<String, String>> hierarchyList)**

/\*\*

\* Beallitja a szinterre a fomennut.

\*/

**public void showMainMenu()**

/\*\*

\* Hozzzaadja a megadott nevu elemet a megadott nevu mappaba

\* @param treeItemValue elem neve

\* @param folderName mappa neve

\* @throws NullPointerException ha egy szettet probalunk letrehozni ugy, hogy egyik elem sincsen kivalasztva

\*/

**public void addItemToMainTree(String treeItemValue, String folderName) throws NullPointerException**

/\*\*

\* Eltavolitja a megadott nevu elemet a megadott nevu mappabol

\* @param treeItemValue elem neve

\* @param folderName mappa neve

\* @throws NullPointerException ha egy szettet probalunk eltavolitani ugy, hogy egyik elem sincsen kivalasztva

\*/

**public void removeItemFromMainTree(String treeItemValue, String folderName) throws NullPointerException**

/\*\*

\* Visszater a fomenuben aktualisan megjelenitett szettnek a kifeljezes-definicio listajaval.

\* @return kifejezes-definicio lista

\*/

**public ArrayList<TermAndDefinition> getDisplayedStudySetData()**

/\*\*

\* Betolti a fomenube a megadott szettet.

\* @param studySet szett

\*/

**public void loadStudySet(StudySet studySet)**

/\*\*

\* Visszater a fomenuben megjelenitett fa mappa strukturaval.

\* @return fa mappa struktura

\*/

**public TreeView<String> getFolderTree()**

/\*\*

\* Letrehozza az egyes tanulo ablakokat.

\*/

**public void setupStudySections()**

/\*\*

\* Az ablakban a megadott szettel a StudyFlashcard reszt jeleniti meg.

\* @param studySet szett

\*/

**public void showStudyFlashcard(StudySet studySet)**

/\*\*

\* Az ablakban a megadott szettel a StudyLearn reszt jeleniti meg.

\* @param studySet szett

\*/

**public void showStudyLearn(StudySet studySet)**

/\*\*

\* Az ablakban a megadott szettel a StudyTest reszt jeleniti meg.

\* @param studySet szett

\*/

**public void showStudyTest(StudySet studySet)**

/\*\*

\* A tanulas befejezese es ujra a fomenu megjelenitese.

\*/

**void endStudy()**

**Tesztek és metódusaik:**

FolderTest

/\*\*

\* Felallitja a teszteloi kornyezetet.

\*/

@BeforeEach

**void setUp()**

/\*\*

\* Teszteli a getStudySet() fuggveny megfelelo mukodeset.

\*/

@Test

**void getStudySet()**

/\*\*

\* Teszteli az addStudySet() fuggveny megfelelo mukodeset.

\*/

@Test

**void testAddStudySet()**

/\*\*

\* Teszteli a containsStudySet() fuggveny megfelelo mukodeset.

\*/

@Test

**void testContainsStudySet()**

/\*\*

\* Teszteli a removeStudySet() fuggveny megfelelo mukodeset.

\*/

@Test

**void removeStudySet()**

/\*\*

\* Teszteli a getSetNames() fuggveny megfelelo mukodeset.

\*/

@Test

**void testGetSetNames()**

/\*\*

\* Teszteli a setStudySet() fuggveny megfelelo mukodeset.

\*/

@Test

**void testSetStudySet()**

ModelTest

/\*\*

\* Felallitja a teszteloi kornyezetet.

\*/

@BeforeEach

**void setUp()**

/\*\*

\* Teszteli a nameDoesNotExist() fuggveny megfelelo mukodeset.

\*/

@Test

**public void testNameDoesNotExist()**

/\*\*

\* Teszteli a createFolder() fuggveny megfelelo mukodeset.

\*/

@Test

**public void testCreateFolder()**

/\*\*

\* Teszteli a createStudySet() fuggveny megfelelo mukodeset.

\*/

@Test

**public void testCreateStudySet()**

/\*\*

\* Teszteli a removeStudySet() fuggveny megfelelo mukodeset.

\*/

@Test

**public void testRemoveStudySet()**

/\*\*

\* Teszteli a removeFolder() fuggveny megfelelo mukodeset.

\*/

@Test

**public void testRemoveFolder()**

StudySetTest

/\*\*

\* Felallitja a teszteloi kornyezetet.

\*/

@BeforeEach

**void setUp()**

/\*\*

\* Teszteli az add() fuggveny megfelelo mukodeset.

\*/

@Test

**void testAdd()**

/\*\*

\* Teszteli a getName() fuggveny megfelelo mukodeset.

\*/

@Test

**void testGetName()**

/\*\*

\* Teszteli a getTermsAndDefinitions() fuggveny megfelelo mukodeset.

\*/

@Test

**void testGetTermsAndDefinitions()**

TermAndDefinitionTest

/\*\*

\* Teszteli a konstruktor megfelelo mukodeset.

\*/

@Test

**void testConstructor()**