MarkdownToGuide implements Function<Path, String>

// Extracts all headings and bracketed info from the
// markdown file at the path (including brackets and
// hashtags)
public String apply(Path md);

AccMarkdowns implements FileVisitor<Path>

private ArrayList<Path> markdowns;

// default constructor
public AccMarkdowns(ArrayList<Path> list);

// Invoked before entries in the directory are visited
public FileVisitResult preVisitDirectory(Path root, BasicFileAttributes attrs);

// Invoked after entries in the directory, and all their descendants, have been visited
public FileVisitResult postVisitDirectory(Path root, BasicFileAttributes attrs);

// Throws the exception that prevented the file from being visited
public FileVisitResult visitFileFailed(Path root, IOException exc);

// Adds the current file to the list of markdowns if it is a markdown file
public FileVisitResult visitFile(Path root, BasicFileAttributes attrs);

CreateGuideProblems

//Writes a StudyGuide and question list to a
//file
public void CreateGuideProblems();

Driver

// Either creates a study guide and problem list, or
// begins a study session
public static void main(String[] args);

SRToALProblems

// Gets an ArrayList of Problems
// from anSR file
public ArrayList<Problem>
apply(Path questionSR);

SessionController implements Iterator<Problem>

private Path toGuide;
private ArrayList<Problem> allProblems;
int curIndex;
int changedToEasy;
int changedToHard;
private ProblemSet problemSet;
private int maxProblems
private final Appendable appender;
private final Scanner sc;

public void startSession();
public void setMaxProblems();
public void ControlInput();
public void cycleProblems();
publi void showAnswerOrExit();
public void endGame();
public void changeDifficulty();

MarkdownToQuestions implements Function<Path, String>

// Extracts all questions from the given markdown file and formats as
a string
public String apply(Path md);

ProblemSet

private ArrayList<Problem> allProblems;
private Random random;
private int numEasy;
private int numHrad;
private Path tosr;

ArrayList<Problem> getAllProblems();
public void changeToHard(int index);
public void changeToEasy(int index);
public String toString();
public void writeToPath(Path tosr);

WriteGuide

private Path path;

// Writes the given string to the path
public void writeToFile(String str);

StudyGuide

ArrayList<Path> markdowns;

// default constructor
StudyGuide(ArrayList<Path> markdowns);
// Sorts the markdown path by the given
// comparator
public void sort(Comparator<Path> sortBy);
// Maps each markdown file to a String, then folds them together
public String toString();
// Accumulates all markdowns files contained within the given root path
public void addToFile(Path root);

Enum: Difficulty

HARD
EASY

Problem

String question;
String answer;
Enum difficulty;

String getAnswer();
void changeDifficulty(Enum difficulty);
String toString();

CompareName implements Comparator<Path>

// Determines which Markdown file's
// title comes first lexographically
public int compare(Path p1, Path p2);

CompareCreated implements Comparator<Path>

// Determines which Markdown file's
// title comes first in terms of what
// was created first
public int compare(Path p1, Path p2);

CompareModified implements Comparator<Path>

// Determines which Markdown file's
// title comes first in terms of what
// was last modified
public int compare(Path p1, Path p2);