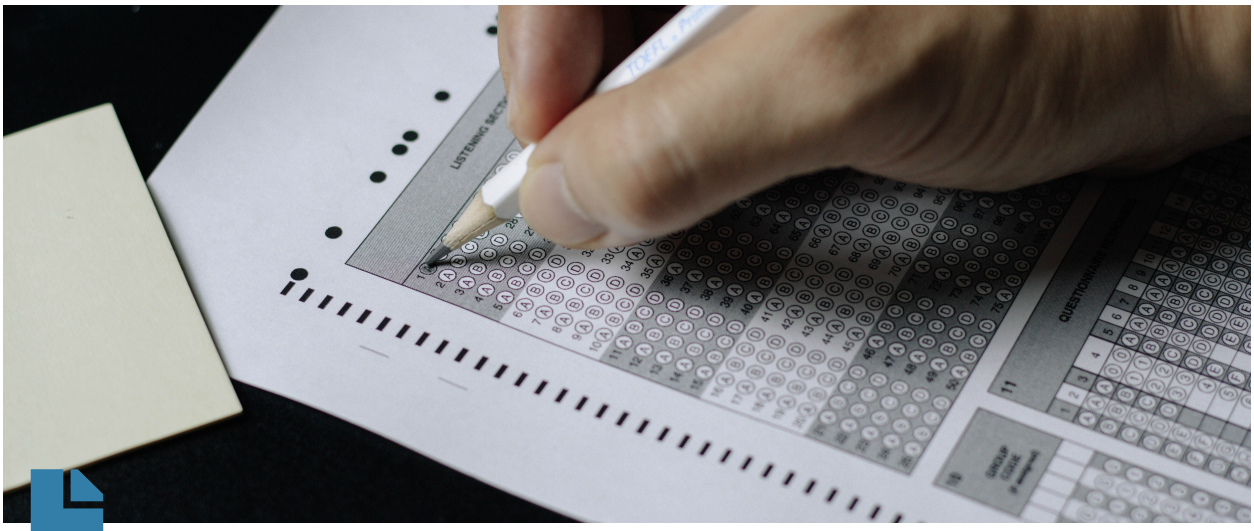

Quiz Generator Documentation

Contents

https://github.com/JaromirProchazka/quizGenerator

User Documentation

Notes file

The notes file used to create a new topic or quiz needs to be a .html file, Questions given in the quiz are then those strings in file in italics (in .html file those strings in `<em>❓</em>` tags). When these notes are used to create new topic, either the html title or the filename is used as name of the topic.

Topics Menu

At the Start of the App, you will be greeted by a list of your topics on top, and a `Create New Topic` button at the bottom. If you want to add a new topic, press the button and select the .html file on your device, with your notes.

After that a new topic is added to the topics list at the top, or a topic with the same name is updated. Than select topic in list, you can then either click

- `Open` button and start the quiz,
- or click `Edit` button and a edit window will appear. You can than **Rename** the topic or **Delete** it

Quiz Window

When the quiz is started, user is greeted on bottom with the question and answers window and on top with 4 buttons. Read the question you see and try to answer it with your own words.

- press `Answer` button, and read your notes, which are shown at the bottom.
  - If you answered correctly, press `Next 👍` which simply shows new question.
  - If you failed to answer correctly, press the `Next 👎` button. This one also shows new question, but it moves the incorrectly answered question ahead, so that it will be asked again and hopefully answered correctly.
- when you are done and want to start a different quiz, simply press the `Back` button, which closes this quiz.

There is also a counter, which says “`number of answered questions / number of question`”. If you answer incorrectly and press `Next 👎`, the counter doesn’t increment. If the user answers all questions and the numbers on each side of “/” are the same, the counter gets green showing, that the quiz is finished, but new set of questions is prepared.

Developer Documentation

Besides the System libraries, this project uses C# library `HtmlAgilityPack` for parsing and building .html files.

Overview

When notes file is given to the app, in hidden folder `.sources/`, a new folder is created with files used for quiz. The file structure of `.sources/` is as follows:

```
.sources\
        |---------- styles.css
        |
        |---------- *general topic folder\
        |                   |---------- notes.html
        |                   |---------- questions.html
        |
        |---------- ... more topic folders\
```

When quiz is than started in windows form window, the `questions.html` is opened and serves as visuals for the quiz manipulated by button in the form

Utility Functions

Most of our utility functions are located in `FileManager` namespace. This namespace implements methods for two main purposes:

- building files for new topics like quiz file used to display the quiz. These functions are implemented in `QuestionsFile` class
  - `CreateNewTopic( string notesPath )` is the most important method, which creates the new folder with the topic-relevant files in `.sources/` hidden folder
  - `CreateQuestionsFileText( string notesMarkdown )` is a method used also by `CreateNewTopic(string notesPath)` that converts the notes markdown into the questions markdown, which lacks useless text from notes and includes the question-answer pairs

- for the quiz itself, the most important is `sequenceOfQuestions` class, whose job is to give a random sequence of questions, which:
  - do not repeat;
  - are logically structured in such a way that any question A, that uses a concept from different question B is asked after the question B;
  - and from a set of all sequences following the first two rules, the probability of choosing a sequence is the same for each of them;
- this choice is achieved by using `Dag` data class from `FileManager` namespace, where first, a DAG with questions as Vertices and Edges as logical relationships between questions is built, and than the sequence is some randomly chosen topological order of the DAG.
  - DAG of size N is represented as a list of neighbors, where for each node, we also remember it's IN and OUT degree
  - There is a public function `insert( label , predicesors[] )`, that in time linear to the number of node predecessors add the node ensuring that the new graph is still DAG; if a predecessor referred to is not already in graph, an error is thrown
  - Than there is public function `randomTopologicalOrder()`, which goes throw N iteration, and on each iteration removes some random node with zero IN degree, updates IN degrees of all successors and puts this found node into the sequence, which creates a some topological order at the end
    - at the end, the graph is reset to state before call of `randomTopologicalOrder()`, so that it can be called again

Topics Menu

At the start of the app, a Main page with a list of topics is opened. The list is filled by topics by going throw the `.sources/` folder and adding links to the topics there. When new topic is added, the Topics list is updated as well, when the creation of new topic is finished.

Quiz Window

The question window first loads the contents of `questions.html` to a Windows Form Web Browser control and then the buttons invoke Javascript methods from `questions.html`, which manipulate the visibility of elements like questions and answers.