Module genassign

Generates Individualized Assignments and Solutions based on a LaTeX Template

Author: Colin Caprani, colin.caprani@monash.edu[1]

Overview

genassign is a wrapper script that does two main things:

1. It repeatedly calls a LaTeX-PythonTex template file and places the output in subfolders of the template file directory.
2. It substitutes student-specific data into each generated file.

When the template file has randomization embedded, this is called automatically at each compilation, resulting in individualized assignments. Further, using a specific sequence of LaTeX commands, the question paper and associated solution file are generated separtely and placed into separate subfolders of the template file directory.

genassign is written to allow indpendent compilation of the template file to facilitate deveopment and checking of the questions and solutions, including close control of the randomization.

It is not necessary for there to be PythonTex commands in the template.

Useage

Prepare a template LaTeX-PytonTex file with complete questions and solutions. Add the jinja templating variables to the document as necessary. Include the LaTeX commands, and wrap the solutions as shown above. Use PythonTex to randomize the problem variables upon each compilation.

Standard example useage:

```
python genassign.py main.tex students.csv -t "Test 1"
```

To debug:

```
!debugfile('genassign.py', args='"main.tex" "students.csv"')
```

Commands

```
genassign.py [-h] [-t TITLE] [-m MOODLE_STEM] [-s SOL_STEM]
                 [-p PAPER_STEM] [-a ANSDIR] [-q QUESTDIR]
                 template worksheet
```

optional arguments:

`-h`, `--help` show this help message and exit

`-t`, `--title` Test title filename prefix

`-m`, `--moodle_stem` Moodle assignment type folder stem, usually `onlinetext` or `file`

`-s`, `--sol_stem` Solutions filename stem, e.g. `'_sols'`

`-p`, `--paper_stem` Question paper filename stem, e.g. `'_paper'`

`-a`, `--ansdir` Directory name for solutions output, e.g. `'solutions'`

`-q`, `--questdir` Directory name for questions output, e.g. `'questions'`

required named arguments:

`template` LaTeX Template File with certain commands for jinja2 and hiding solutions, e.g. `main.tex`

`worksheet` Student Moodle worksheet of specific format from assignment grading, e.g. `students.csv`

[1] mailto://colin.caprani@monash.edu

Requirements

System requirements are working installations of Python, LaTeX, and PythonTex. More specifically, `genassign` requires:

1. A LaTeX-PythonTex template with certain specific commands
2. A Moodle grading worksheet for the assigment as input

Template

There are two commands required at a minimum in the LaTeX file.

Jinja2 Templating

The command for *jinja2* templating

```
\newcommand*{\VAR}[1]{}
```

which has no effect on the template other than to identify variables used for subsitution of student-specific information as defined in Moodle worksheet:

- Student's full name: `\VAR{FULL_NAME}`
- Student's ID: `\VAR{STUDENT_ID}`

LaTeX Commands

The LaTeX commands to wrap the solutions, so they can be toggled on and off must be placed in the document preamble. The following must appear:

```
\usepackage{comment}

\newif\ifhidden
% This defines whether to show the hidden content or not.
\hiddenfalse
\ifhidden   % if \ hiddentrue
    \excludecomment{hidden} % Exclude text within the "hidden" environment
\else       % \ hiddenfalse
    \includecomment{hidden} % Include text in the "hidden" environment
\fi
```

so that the solutions are wrapped in the document body as follows

```
\begin{hidden}
    ...LaTeX solution code, including PythonTex as necessary
\end{hidden}
```

Documentation

To use `pdoc` to generate this documentation, issue this:

```
pdoc --html --force --config show_source_code=False genassign
```

And to generate a PDF of the documentation, first generate the markdown to the standard output stream and pipe it to `doc.text`:

```
pdoc --pdf --force --config show_source_code=False genassign > doc.txt
```

then issue this command:

```
pandoc --metadata=title:"genassign Documentation" --toc --toc-depth=4 --from=markdown+abbreviations
```

Functions

Function `gen_assign`

```
def gen_assign(student, template, assign_strings)
```

Drives the rendering and compilation process for each student, and cleans up the files afterwards.

Parameters

`student : tuple of string` Contains student's data: Moodle ID, Full Name, Student ID.
`template : jinja2 template` set to render the LaTeX file.
`assign_strings : tuple of strings` Contains prgram string variables in the order: tmpfile = temporary file name moodle_str = older name stem as appropriate to the type of Moodle assigment. Usually 'file' or 'onlinetext' filename_prefix = title of test typically solutions_stem = filename postfix for solutions pdf questions_stem = filename postfix for questions pdf solutions_dir = name of solutions directory questions_dir = name of questons directory

Returns

None.

Function `gen_q_and_a`

```
def gen_q_and_a(student, assign_strings)
```

Generates the Questions and Answers documents for a student

Parameters

`student : tuple of string` Contains student's data: Moodle ID, Full Name, Student ID.
`assign_strings : tuple of strings` Contains prgram string variables in the order: tmpfile = temporary file name moodle_str = older name stem as appropriate to the type of Moodle assigment. Usually 'file' or 'onlinetext' filename_prefix = title of test typically solutions_stem = filename postfix for solutions pdf questions_stem = filename postfix for questions pdf solutions_dir = name of solutions directory questions_dir = name of questons directory

Returns

None.

Function `main`

```
def main(args)
```

The main function, called when file is run as script, allowing the other functions to be used from this script through a module interface

Parameters

`args : argparse arguments` The command line arguments parsed using argparse

Returns

None.

Function `make_template`

```
def make_template(texfile, tmpfile)
```

Creates the jinja2 template using a redefined template structure that plays nicely with LaTeX https://web.archive.org/web/20121024021221/http://e6h.de/post/11/

Parameters

`texfile : string` The template LaTeX file containing jinja template variables.
`tmpfile : string` The name of the temporary files that will be used.

Returns

`jinja2 template` jinja2 template used to render the documents.

Function `move_pdf`

```
def move_pdf(student, tmpfile, root, file_stem, moodle_str, prefix)
```

Moves the compiled PDF to the appropriate folder

Parameters

`student : tuple of string` Contains student's data: Moodle ID, Full Name, Student ID.
`tmpfile : string` Name of the temporary files.
`root : string` The name of the subfodler in which the student's folder will be placed. Usually one of 'solutions' or 'questions'
`file_stem : string` Postfix to be applied to student-specific string to distinguish questions from solutions
`moodle_str : string` Folder name stem as appropriate to the type of Moodle assigment. Usually 'file' or 'onlinetext'
`prefix : string` Prefix to be applied to the student-specific filenames. Usually the title of the test.

Returns

None.

Function `remove_readonly`

```
def remove_readonly(func, path, excinfo)
```

Attempts to remove a read-only file by changing the permissions

Function `render_student_tex_file`

```
def render_student_tex_file(student, template, tmpfile)
```

Renders the tex file for compilation for a specific student

Parameters

`student : tuple of string` Contains student's data: Moodle ID, Full Name, Student ID.
`template : jinja2 template` sed to render the LaTeX file.
`tmpfile : string` Name of the temporary files.

Returns

None.

Function `set_hidden`

```
def set_hidden(texfile, hidden=True)
```

Toggles the solutions visbility in the student's LaTeX file

Parameters

`texfile : string` The template LaTeX file containing jinja template variables.
`hidden : bool`, optional Whether or not the solutions are to be hidden. The default is True.

Returns

None.

Function `student_strings`

```
def student_strings(student, moodle_str)
```

Creates the file and folder name strings for a student

Parameters

`student : tuple of string` Contains student's data: Moodle ID, Full Name, Student ID.

`moodle_str : string` Folder name stem as appropriate to the type of Moodle assigment. Usually 'file' or 'onlinetext'

Returns

`student_file : string` The student-specific part of the pdf filename
`student_folder : string` The student-specific part of the folder name

Generated by *pdoc* 0.7.2 (https://pdoc3.github.io).