aegis

Release 0.1

Jul 22, 2020

Contents

This is the documentation for the AEGIS (Academic Exam Generator for Interchange and Shuffe) package.

The purpose is to compose a collection of exams from a set of several versions of a number of problems. This is useful to generate several examns with a similar dificulty. It also can be used to generate alternative exams from a pool of exercises.

It requires a system with Latex and a pool of exercises written on separated Latex files.

Project by Marcelo Lares (IATE, UNC). Developed in 2020. Contact: marcelo.lares@unc.edu.ar

CHAPTER 1

Usage

1.1 Installation

1.1.1 Downloading AEGIS

The package can be installed either from source or from the pypi repososository.

AEGIS is publically available from a GitHub repository. It can be downloaded or cloned with:

```
git clone https://github.com/mlares/aegis.git
```

The code can be explored using GitHub, including development activity and documentation.

1.1.2 Requirements

AEGIS generates and compile Latex documents, so it need a working installation of Latex in the system.

1.1.3 Installing AEGIS

Once the virtualenvironment has been set (recommended), then install the required packages:

```
pip install -r requirements.txt
```

It is convenient to save the root directory of the installation. In bash, for example,:

```
export aegis_rootdir="$(pwd)"
```

AEGIS module can be used anywhere provided the following command is executed within the environment in the directory $aegis_rootdir:

```
pip install .
```

Alternatively, it can be installed via the python package index:

```
pip install aegis_latex
```

1.1.4 Testing

For testing purposes, an utility is provided in the :meth:`aegis.Exam.gen_examples` method.

```
X.gen_examples(N_problems=4, N_versions=[3, 3, 3, 3],
               dir_tex='dir_tex/', dir_pdf='dir_pdf/')
```

This will create two directories, and fill them with TEX and PDF files. Each tex file contains a short message indicating the problem and version numbers.

```
import aegis

problems, versions = aegis.gen_examples(dir_tex='exams', dir_pdf='exams')

print(problems)
print(versions)

X = aegis.Exam()
X.load_template('template.tex')

items_dir = 'exams'
items = problems
subitems = versions

X.load_items(items_dir, items, subitems)

X.generate(N=4, output_dir='exams', makepdfs=True)
# X.gen_excell(output_dir='exams/')
```

More tests can be found at `tests/test.py`.

1.2 API usage

This tools can be used as an API, from a python prompt or from a command line.

Some tasks that can be performed with the provided utilities include:

- Compile an exam from a chosen set of exercises
- Shuffle a set of versions of exercises to make a given number of different exams
- Obtain all possible combinations of exercise versions to compile exams
- Compile a complete set of exercises
- Generate filenames to be filled with text of exercises
- Generate documents that involve lists of text items (using the same model of an exam)
- Produce an excell file with the selected versions of the exercises.

1.2.1 Minimal example

The generation of the exams require three steps:

- load a latex template
- load the versions of the exercises
- generate the final latex files for the exams

For example, the code below uses a template file named `parcial_1.tex`, and loads the sources to compile exams with three exercises. Exercise 1 has three versions, exercise 2 has four versions, and exercise 3 has two versions.

The N option in `aegis.Exam.generate()` allows to produce N exams.

```
import aegis
X = aegis.Exam()
X.load_template('parcial_1.tex')

items_dir = 'exercises'
items = [1, 2, 3]
subitems = [[1, 2, 3], [1, 2, 3, 4], [1, 2]]

X.load_items(items_dir, items, subitems)
X.generate(N=4, output_dir='exams')
```

1.2.2 Optional parameters

Optional parameters can be used to generate exams in different ways. All optional parameters are described in the documentation of the methods. The most relevat options are:

Randomly combine exercise versions: In the :meth:`aegis.Exam.generate` method, use the `shuffle=True` option.

Compile PDF files: In the :meth:`aegis.Exam.generate` method, use the `makepdfs=True` option.

Generate an Excell file with the summary of exercise versions: Call the :meth:`aegis.Exam.gen_excell` method

1.2.3 Format for latex files

The default format for latex files is `e01_v01.tex`. The method `Exam.gen_examples()` produces example files with this format:

```
X = aegis.Exam()
X.gen_examples()
```

It is possible to change the default behaviour with the method *aegis.aegis.Exam.name_pattern()*. For example, in order to use a suite of exercises of the form:

- problem_001-version_01.tex
- problem_001-version_02.tex
- problem_001-version_03.tex
- etc

we can call the function as follows:

```
X = aegis.Exam()
X.name_pattern('problem_', 'version_', '-', 3, 2)
```

1.2.4 Excell file with the list of versions

```
X.gen_excell()
```

1.3 Exercises

Individual exercises must be set following a name convention, where the exercise number and version are explicitly contained in the name. For example,

for the version 3 of the exercise 1.

All exercises can be set on a separate directory.

1.4 Templates

Any latex file can be used as a template for the exams. The part with the exercises must contain the following:

```
\newenvironment{ejj}[1]%
{\addtocounter{ejnro}{1}\hspace{-1.2cm}$\blacktriangleright$\hspace{.3cm}{\textb
f{\arabic{ejnro}.}}\hspace{.3cm}}%
{\vspace{5pt}}
```

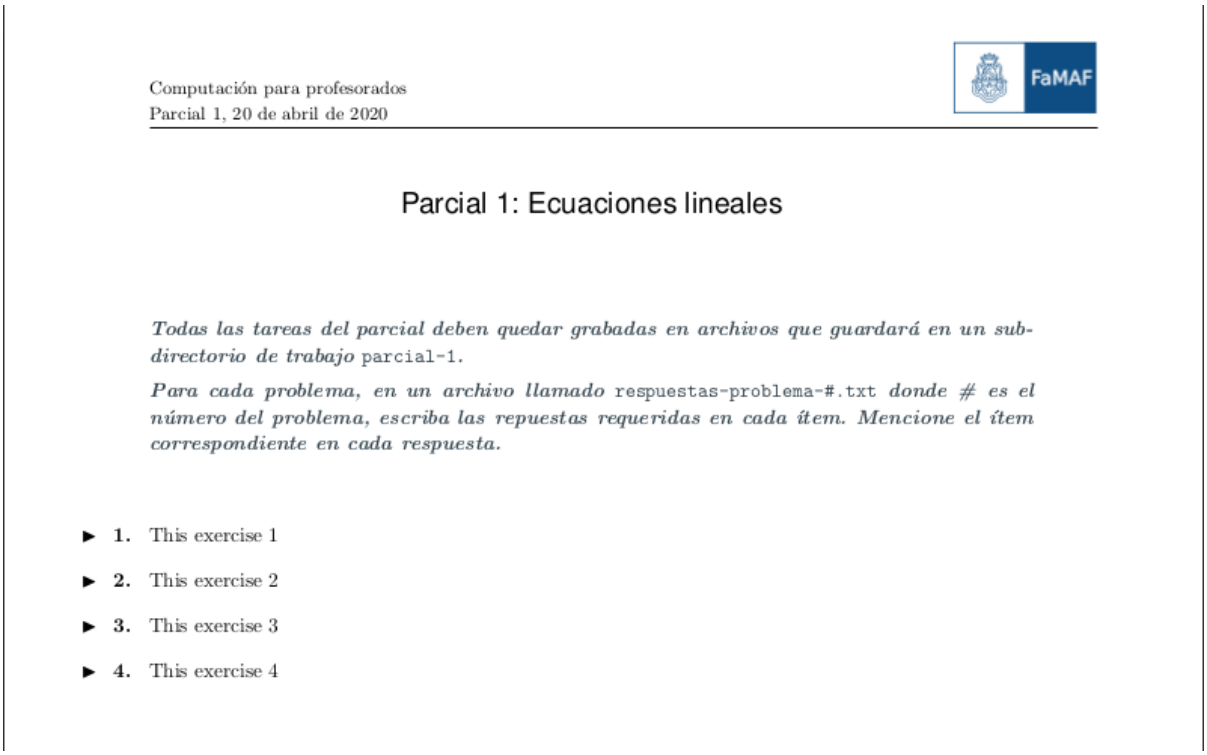
in the preamble, and

```
\BLOCK{ for ej in exs }
   \begin{ejj}
      \VAR{ej}
   \end{ejj}
\BLOCK{ endfor }
```

in the main document.

An example template file, `template.tex` is provided, which can be easily modified. This template uses a logo (for Famaf, UNC), which can be replaced.

The result for the sample template file is as follows:



Computación para profesorados
Parcial 1, 20 de abril de 2020

FaMAF

Parcial 1: Ecuaciones lineales

Todas las tareas del parcial deben quedar grabadas en archivos que guardará en un subdirectorio de trabajo `parcial-1`.

Para cada problema, en un archivo llamado `respuestas-problema-#.txt` *donde # es el número del problema, escriba las repuestas requeridas en cada ítem. Mencione el ítem correspondiente en cada respuesta.*

- **1.** This exercise 1
- **2.** This exercise 2
- **3.** This exercise 3
- **4.** This exercise 4

1.5 aegis module

AEGIS: Academic Exam Generator for Interchange and Shuffe.

The purpose is to compose a collection of exams from a set of several versions of a number of problems. This is useful to generate several examns with a similar dificulty. It also can be used to generate alternative exams from a pool of exercises.

class aegis.aegis.**Exam**

Bases: object

Exam (class): tools to generate exams with random exercises.

load_template : load template

gen_examples(*N_problems=1, N_versions=[[1]], dir_exams='exams/'*)

Generate exams example files.

N_problems: int Numbers of the problems in the exams

N_versions: list of lists Numbers of the versions to be used in the problems

dir_exams: str Directory where latex files are stored

problems: list List of the numbers of the problems

versions: list of lists Lists with the numbers of the versions

gen_excell(*output_dir='./', fname_xlsx='exams_versions.xlsx'*)

Generate an Excell file with the contents of the exams.

output_dir: directory where to put the excell file

None

generate(*N=0*, *output_dir='./'*, *shuffle=True*, *all_permutations=False*, *makepdfs=False*, *interactive=False*)

Generate exams suffling and randomly chosing items.

N [int (optional)] The number of exams to generate. If N is greater than the number of iterations, some exams will be repeated (by the Pigeon-hole theorem.)

shuffle [boolean (optional)] If True, shuffle the versions of the exercises to generate random versions of the exams. Dafault: True

all_permutations [boolean (optional)] If True, generate the complete list of possible combinations of the versions of the exercises. If N is present, it will be ignored. Default: False.

makepdfs [boolean] If True, compile PDF files from latex files. Default: False.

interactive: boolean If True, return a list with the version used on the exams.

ex_list [list] A list containing the versions of the exercises. Only returned if "interactive=True".

load_items(*idir*, *items*, *subitems*)

Load exercises to compile exams.

idir: str Directory where latex files are stored

items: list List of numbers representing exercises

subitems: list List of lists that contain the versions for each one of the exercises in "items".

load_template(*template_file*)

Load a template.

template_file: str Name of the file containing the latex template

Updates the self.template variable.

make_latex_filename(*p*, *v*)

Get the name of latex filenames from problem and version numbers.

If name_pattern was not set, the default convention is used.

name_pattern(*problem_part*, *version_part*, *parts_separation='_'*, *problem_format=2*, *version_format=2*)

Set the convention for latex filenames.

This convention must include two parts, the "problem_part" and the "version_part". If not set, the default is used: problem_part = 'e' version_part = 'v'

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index