

Institiúid Teicneolaíochta Cheatharlach



INSTITUTE *of*  
TECHNOLOGY  

---

CARLOW

At the Heart of South Leinster

IT Carlow

Bachelor of Software Development

Year 4

# Portable GUI for ptpython shell

Functional Specification

Student Name: **Inga Melkerte**

Student ID: **C00184799**

Supervisor: **Paul Barry**

Date: **07/11/16**

# Table Of Contents

Table Of Contents	1
1.Introduction	2
2.Functionality	3
3.Ptpython description	6
References	9

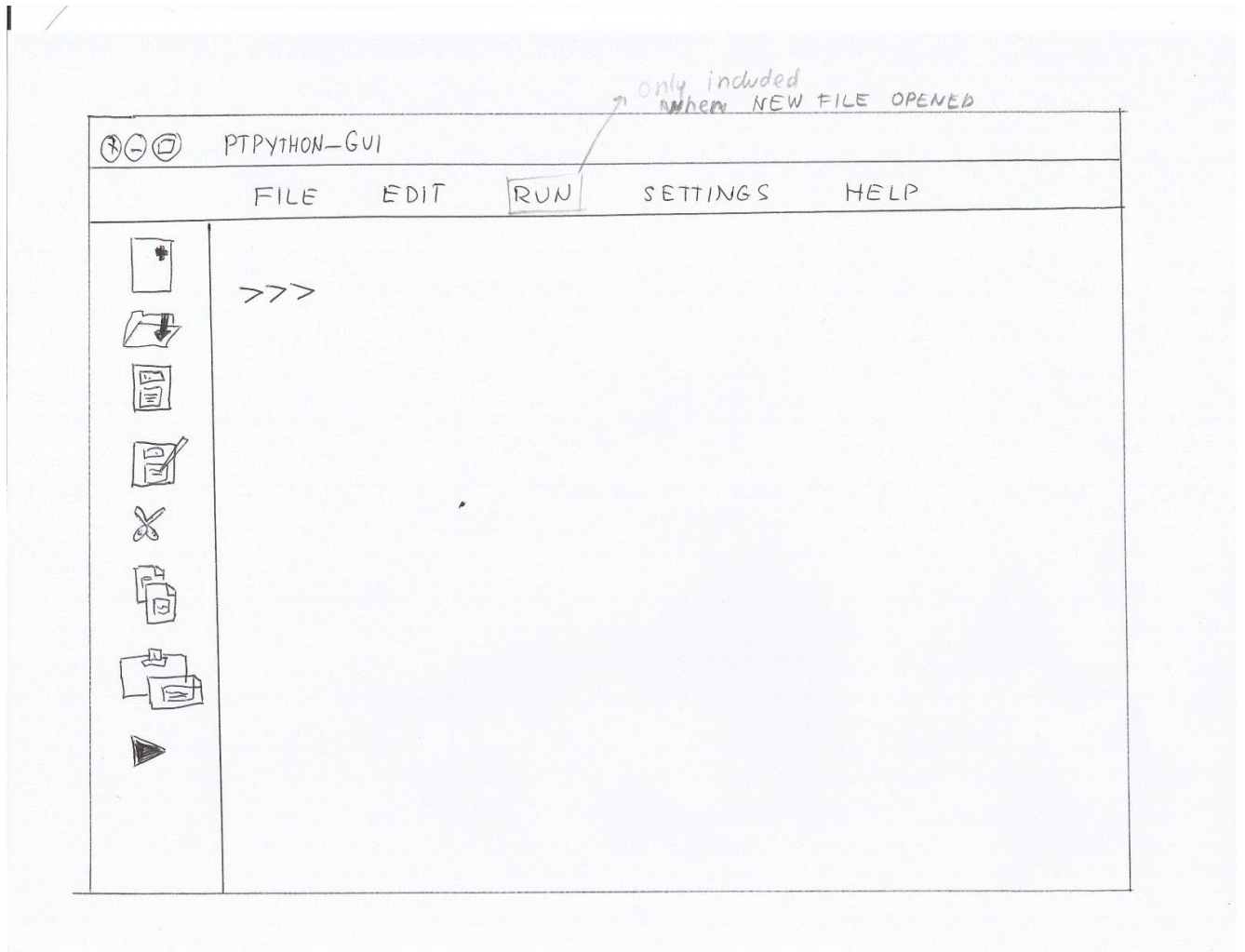
## 1.Introduction

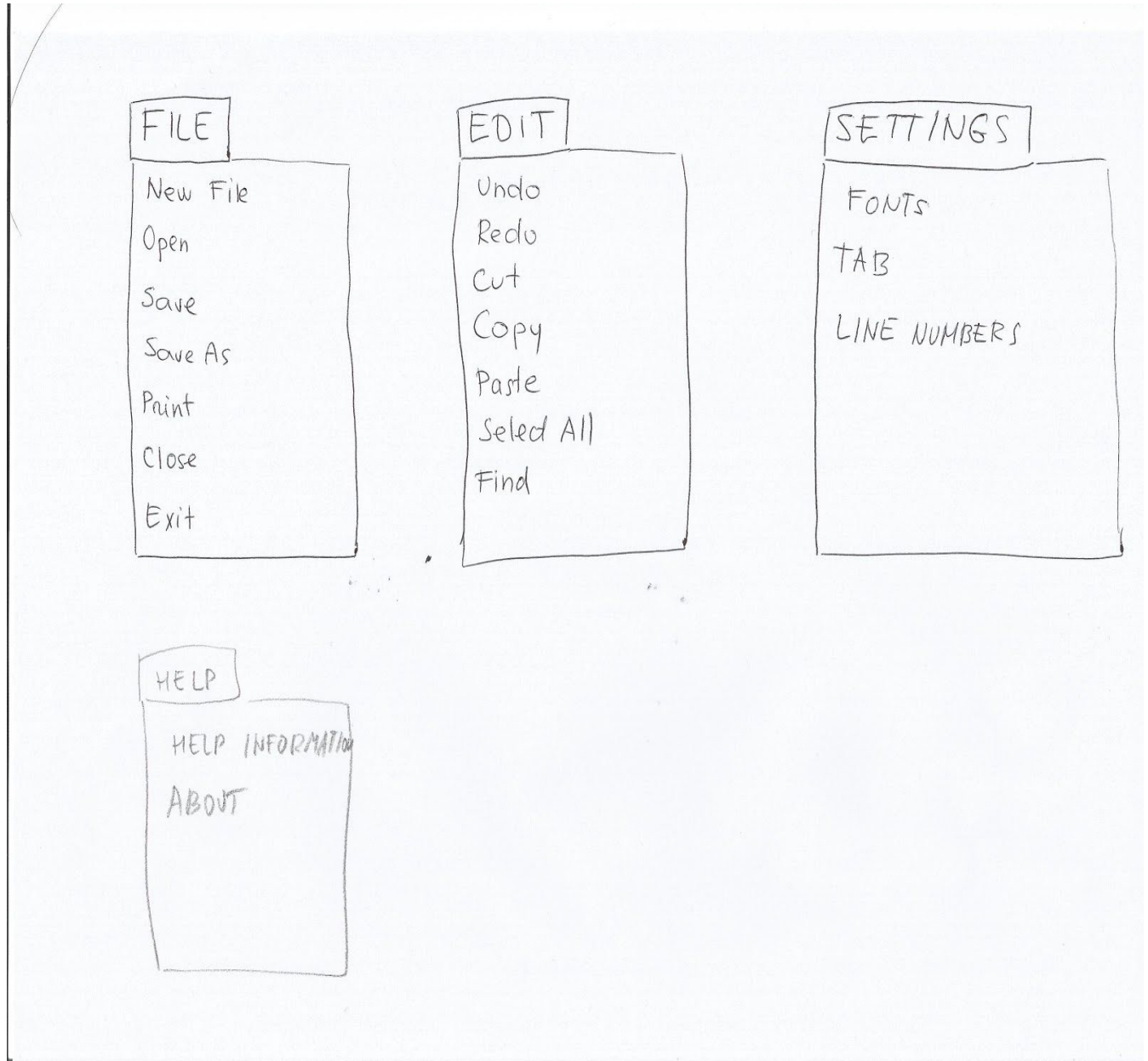
PtpythonGui is being developed to provide Graphical User Interface for ptpython shell. This tool will be developed for beginners as a learning tool. It would provide easy and user friendly interface with user familiar icons and names. PtpythonGui is being build on top of already existing command-line interface - ptpython. Such a tool would offer ptpython features in GUI.

## 2. Functionality

First of all the PtpythonGui tool has to be user friendly.

The PtpythonGui would have two windows - shell prompt window (start up) and code editor window (when clicked on File -> New).



**Menus -**

- File
  - New File - create new file (able to search )
  - Open File - open an existing file
  - Save
  - Save As
  - Print
  - Close

- Exit
- Edit
  - Undo
  - Redo
  - Cut
  - Copy
  - Paste
  - Select All
  - Find
- Run - not in the main window (Run option only included when new file is opened)
- Settings
  - Fonts
  - Tab
  - Line Numbers
- Help
  - Help information
  - About

### 3.Ptpython description

Ptpython is an advanced REPL which is developed by Belgium python developer, many open source project author - Jonathan Slenders.

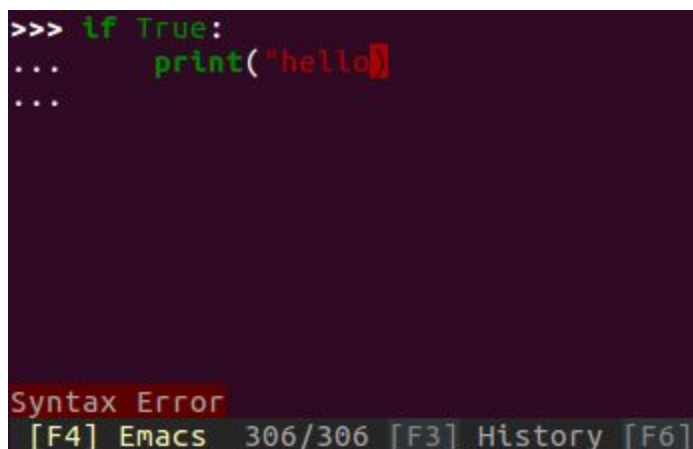
A **read-eval-print** loop (**REPL**), also known as an interactive toplevel or language shell, is a simple, interactive computer programming environment that takes single user inputs (i.e. single expressions), evaluates them, and returns the result to the user; a program written in a REPL environment is executed piecewise. (Wikipedia, 2015)

Jonathan has developed python-prompt-toolkit also and ptpython is built on top of the python-prompt-toolkit library. Prompt-toolkit is a library for building interactive command lines and terminal application in Python. Jonathan has developed bpython and ipython as well and they are based on the same library.

Ptpython works on all Python versions from 2.6 up to 3.5 and works on cross platform (Linux, BSD, OS X and Windows).

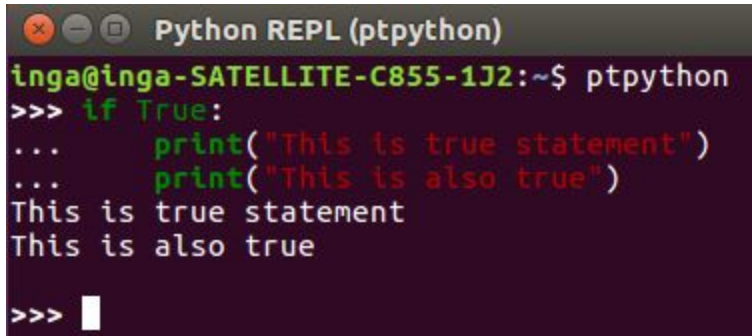
Ptpython is great as learning tool because it provides many **features**:

#### - Syntax highlighting:



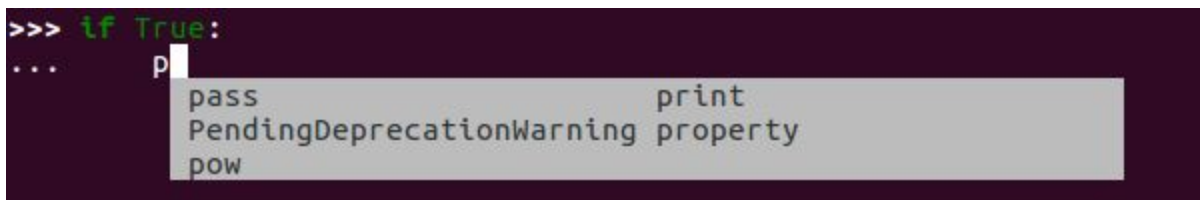
```
>>> if True:
...     print('hello')
...
Syntax Error
[F4] Emacs 306/306 [F3] History [F6]
```

- **Multiline editing** (the up arrow works). Multi-line editing mode will automatically turn on when you press enter after a colon:



```
Python REPL (ptpython)
inga@inga-SATELLITE-C855-1J2:~$ ptpython
>>> if True:
...     print("This is true statement")
...     print("This is also true")
This is true statement
This is also true
>>> █
```

- **Autocompletion:** - Uriegas (Uriegas, 2016) comments: "The idea behind popup autocompletion is amazing. It allows you to see all of your options without you having to hit the TAB multiple times."



```
>>> if True:
...     p█
      pass                print
      PendingDeprecationWarning property
      pow
```

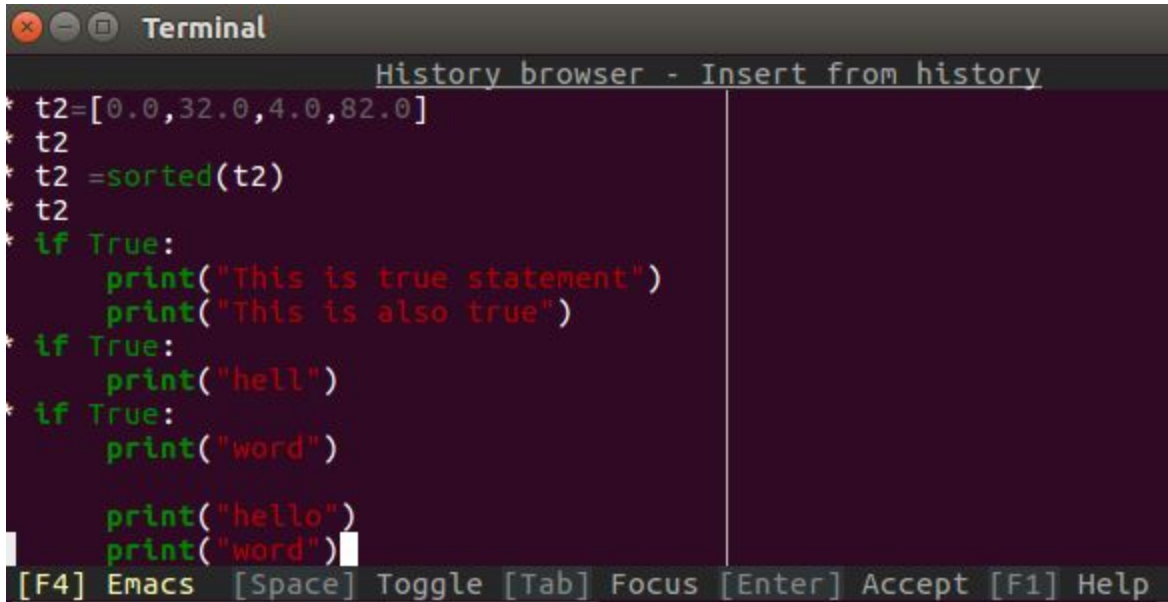
- **Mouse support:**



```
>>> if True:█
...     print("hello")
...     print("word")
```

- **Support for color schemes.**
- **Support for 'bracketed paste** <<https://cirw.in/blog/bracketed-paste>>
- **Both Vi and Emacs key bindings.**
- **Support for double width** (Chinese) characters. (Slenders, 2016)
- **The history browser** is a killer feature that I use everyday. It can be used to pick and choose code you have entered before to include into your current session. It is also searchable and, with recent updates, is very fast





A terminal window titled "Terminal" with a dark background. The terminal shows Python code with syntax highlighting. A "History browser - Insert from history" overlay is visible on the right side of the terminal. The code includes list creation, sorting, and conditional print statements. The terminal prompt is [F4] Emacs [Space] Toggle [Tab] Focus [Enter] Accept [F1] Help.

```
History browser - Insert from history
* t2=[0.0,32.0,4.0,82.0]
* t2
* t2 =sorted(t2)
* t2
* if True:
    print("This is true statement")
    print("This is also true")
* if True:
    print("hell")
* if True:
    print("word")

    print("hello")
    print("word")
[F4] Emacs [Space] Toggle [Tab] Focus [Enter] Accept [F1] Help
```

Ptpython was developed using some **dependencies** (Vinet, Griffin, 2016)

- Python-docopt:

- Python-jedi: Autocompletion library

- Python-prompt-toolkit: Interface

- Python-pygments: Syntax highlighter

- Python-setuptools

- Python2-setuptools

- ipython

## References

1. Slenders, J. (2016) Github. Available at:  
<https://github.com/jonathanslenders/ptpython/blob/master/README.rst>  
(Accessed: 25 October 2016)
2. Slenders, J. (2016) Available at:  
<https://media.readthedocs.org/pdf/python-prompt-toolkit/stable/python-prompt-toolkit.pdf> (Accessed 1 November 2016)
3. Uriegas, E. (2016) *Why ptpython is the only REPL you will ever need*. Available at: <http://terriblecode.com/why-ptpython-is-the-only-repl-you-will-ever-need-2/>  
(Accessed: 1 November 2016)
4. Vinet, J., Griffin, A. (2016) Available at:  
<https://aur.archlinux.org/packages/ptpython/> (Accessed 1 November 2016)
5. Wikipedia (2015) 'Read–eval–print loop', in *Wikipedia*. Available at:  
[https://en.wikipedia.org/wiki/Read%E2%80%93eval%E2%80%93print\\_loop](https://en.wikipedia.org/wiki/Read%E2%80%93eval%E2%80%93print_loop)  
(Accessed: 26 October 2016).