What Kraken work with ?
- a file
- a terminal command

What are possibilities ?

For a file :
- open it
- execute it
- write in
- copy or cut and paste
- copy and paste it's content in a memory file
- copy and paste it's content in an entry field

Under-possibilities (if there is)

open :

-classic (open the file in his default app)

-in a text editor (open the file a text editor, like sublime text) ← work only on linux

write :
- classic , write **a sentence** at a **specific line** of **a file**
- copy and paste the content of an **'a' file** at a **specific line** of a **'b' file**
- complete a file : take the content of an **'a' file** (this file contains data , 1 element a line, ex : Bill, 20yo …), an other **'b' file** is incomplete , there is holes, these holes are not simply empty , there are some in they '{}' , that just precise where there is empty data (eg : Hey {} you are {} yo !) . Using data of the 'a' file the script complete the 'b' file (Important things : 1 line = 1 hole and 1 hole a line ! Eg : Hey **Bill** you are **20** yo !)

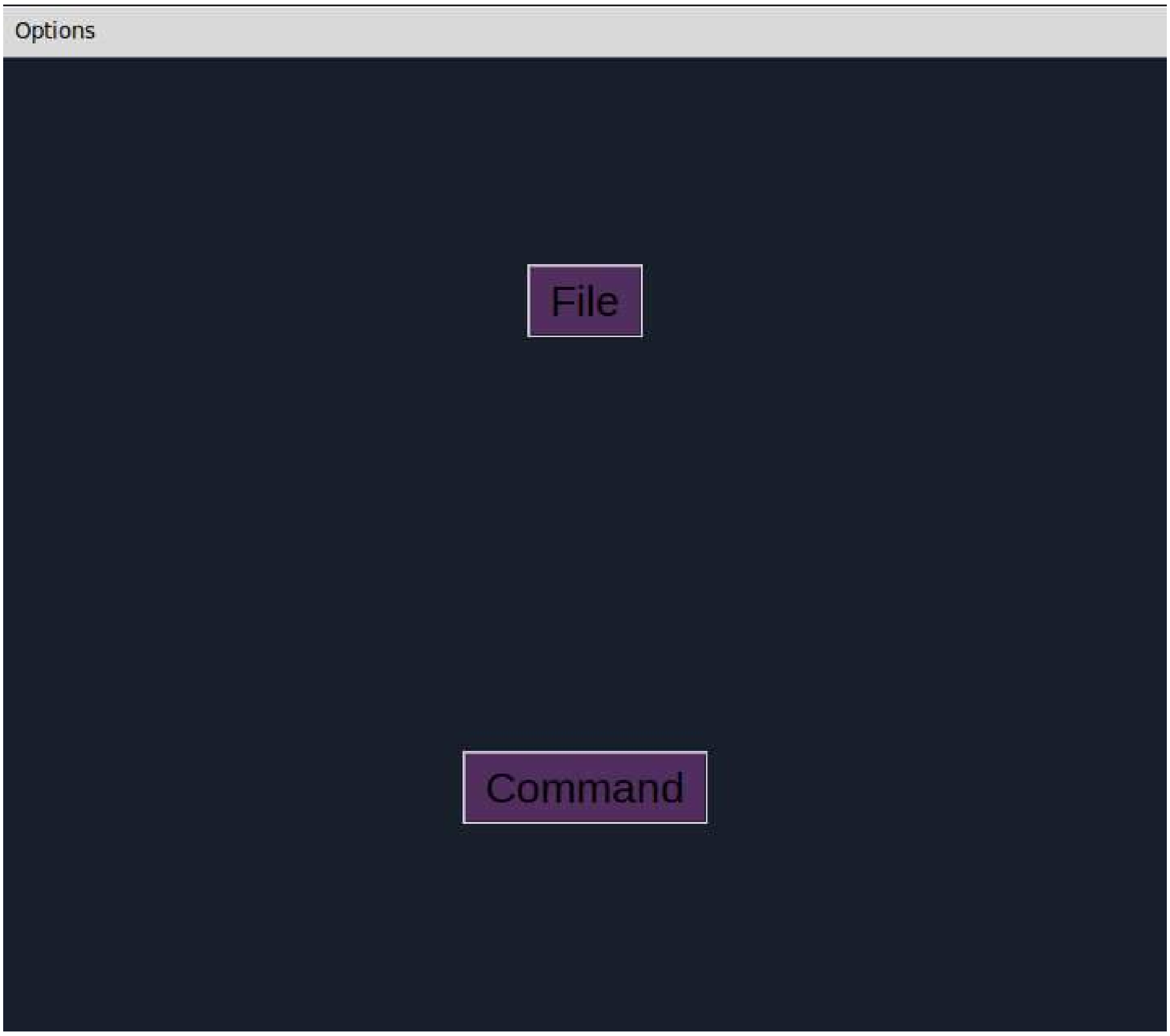
Content on Entry field :
- classic (copy and paste **all** lines **of the file** on entry field)
- more specific (copy and paste just one line **of the file** on entry field)

Annotation of all modes :
- classic open : 'o'
- open in a text editor : 'oe'
- classic write : 'w'
- write form an other file : 'wf'
- complete a file : 'w3f'
- execute : 'e'
- copy/cut paste : 'copy' ou 'cut'
- content in a memory file : 'Y'
- classic content in entry field : 'Y'
- more specific content in entry field : 'Y2' ('ce2' dans la suite du guide)

For commands in terminal :
- simple : execute 1 command line in terminal
- multiple : execute many commands in terminal

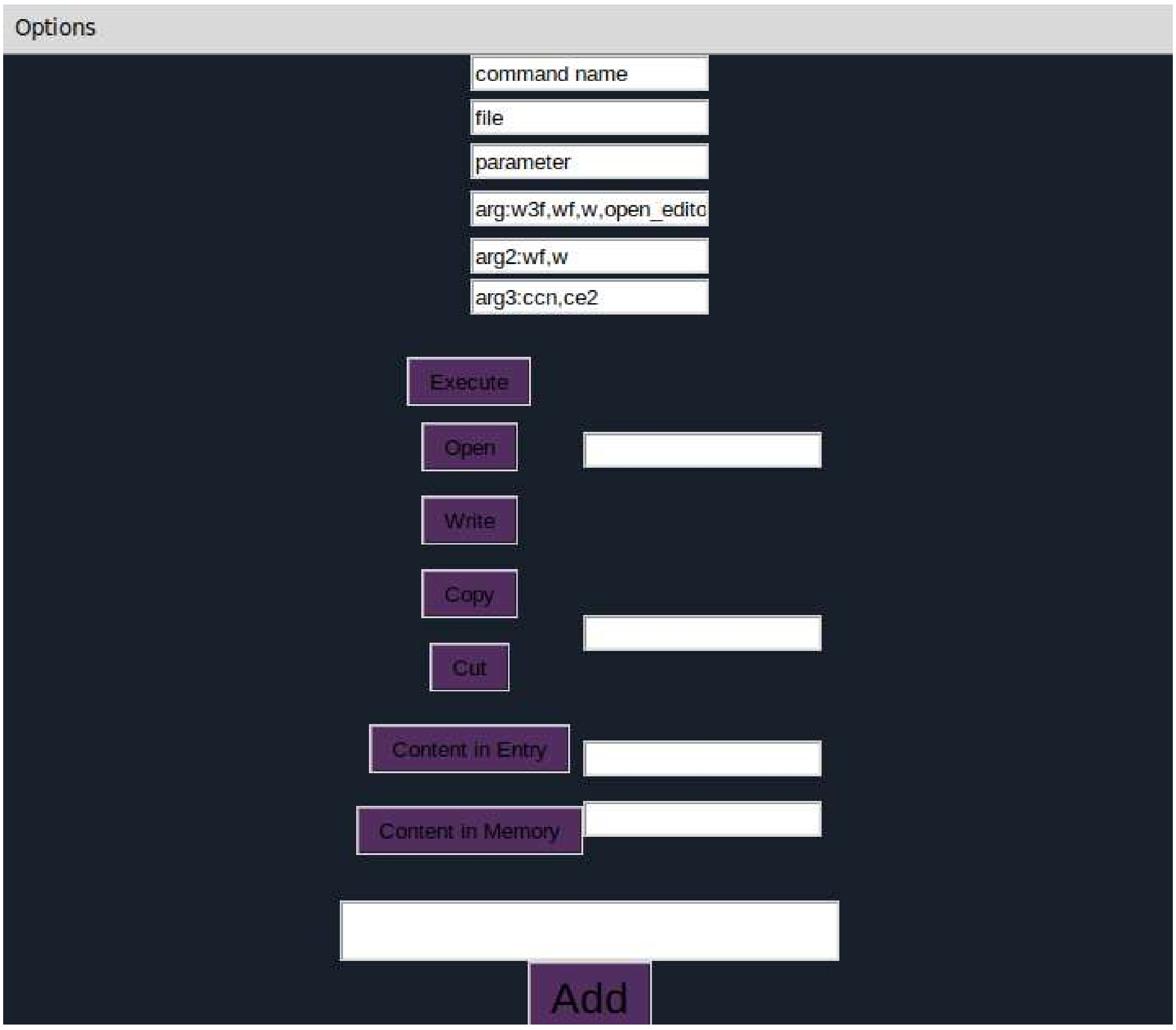
How Kraken work ?
1. you define a function that will done what you need between all possibilities
2. execute the command you defined on the interface

To define a function :
→ *Open graph_import.py*



File:
- You will access to possibilities for files

Command:
- You will access to possibilities for terminal commands



for file :
1. choose the main command
2. choose a file (type the full location and not just the name of the file)
3. ignore args for moment…
4. type on buttons with functionalities you need

<u>Before going to next step we should explain some thing:</u>

w : -the sentence = arg2
- the line = arg
- the file = file

wf : - the line number = arg
- b file = file
- a file = arg2

w3f :- a file = file
        - b file = arg
ce2 : - the file = file

- the line = arg3

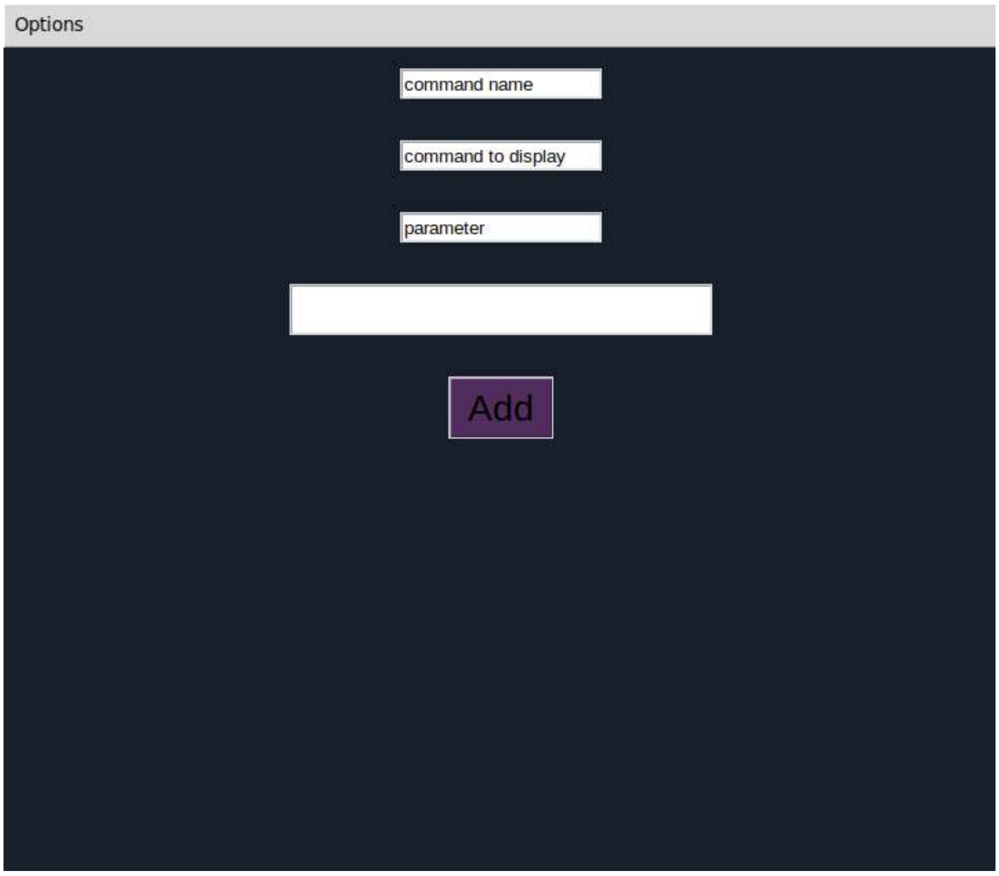
ccn : - the file = file
- the target location = arg3

oe : - the text editor name = arg
- the file = file

<u>And now …</u>
5. complete args depending of your choices
6. tap on ‘add’ button
7. you want to create the next option for a terminal command go to ‘Options’ → ‘First step’

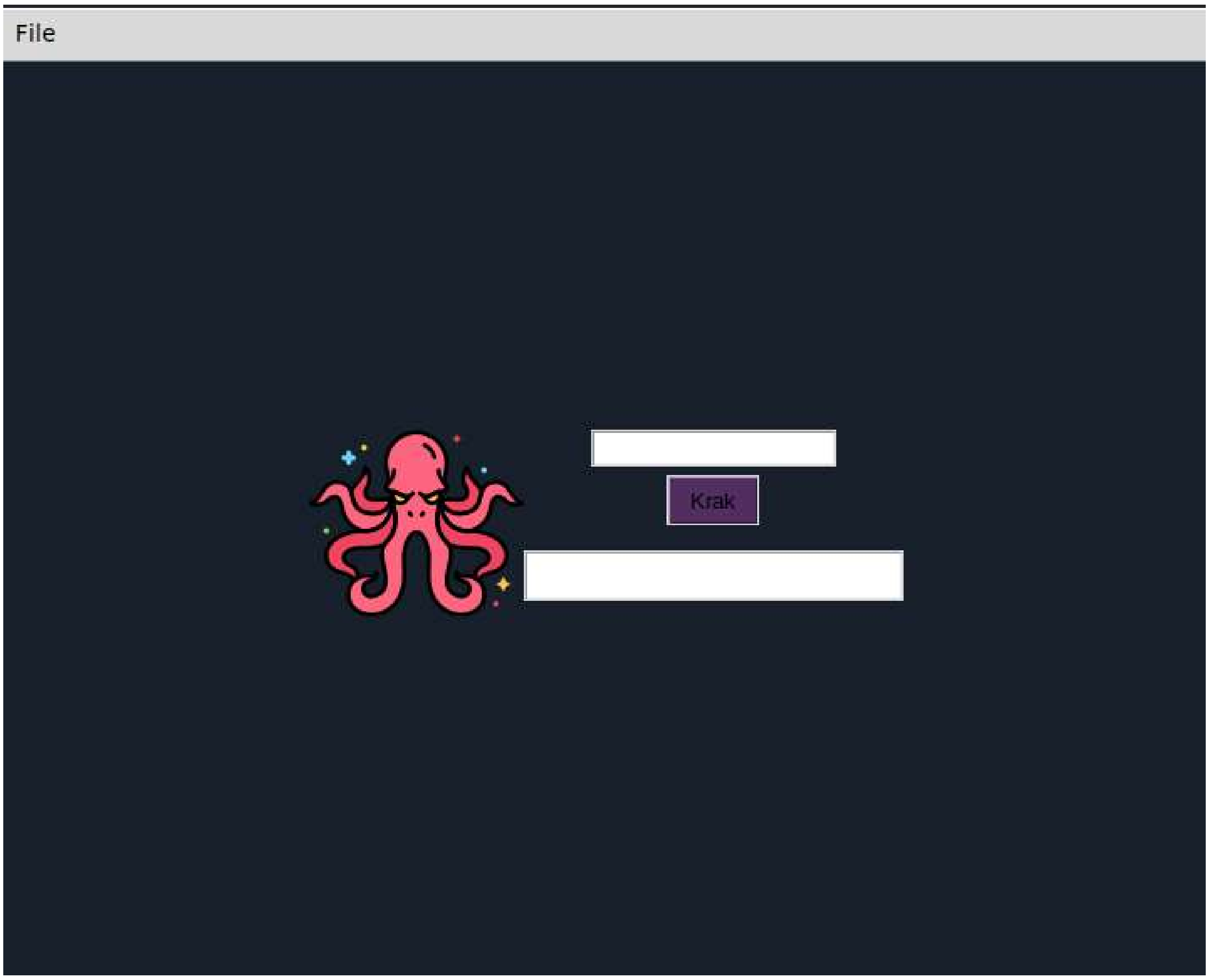
8. when you will done, just close the window

For Command :



…….
1.Choose the main command
2. Choose the terminal command to display (If you want more that one command, you have just to do like that : first command…..second command…..third command (1 command + 5 ‘.’ + 1 command))
3. tap on ‘add’ button

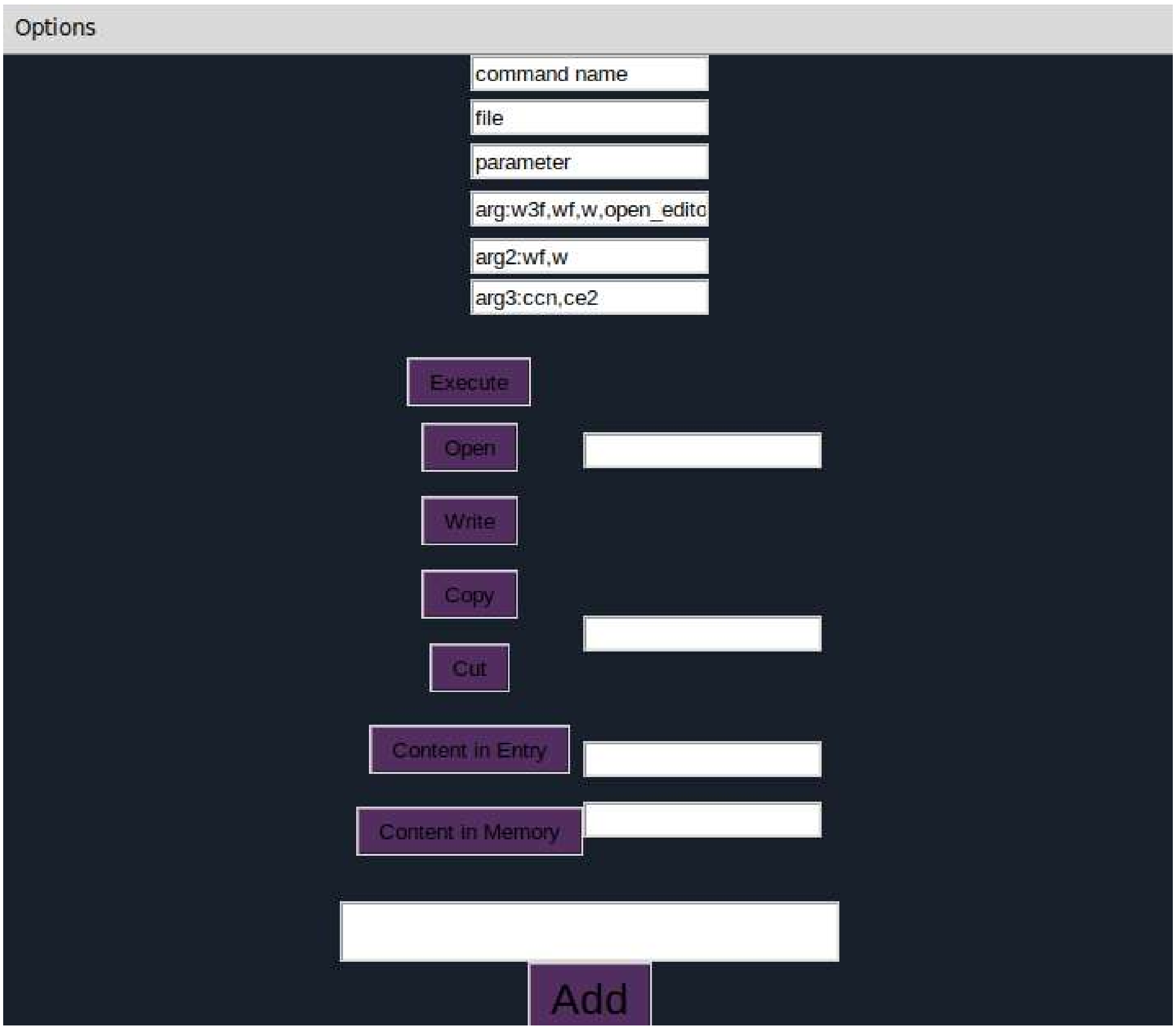
To execute the command :
→ *Open ‘graphic_interface.py’*



Until you openned this window, all your previously defined functions have been imported here, now, you just have to type the main command of one of the function and tap on the button, it will done what you defined it should…

But what is the ‘parameter’ entry field in ‘graph_import.py’ ?
It is some thing you don’t see every days, except if you are a programmer or if you do mathematics …
It’s a parameter (like ‘x ’ or ‘y’ in mathematics) .
Imagine, you defined a function that (for example) open a file, copy and paste it, … you defined it for this specific file, and, now, you want to do the exactly same thing but, with an other file… By using parameters you will not have to define an other function, you just specify the name of the file and it will done the same thing but with the new file…

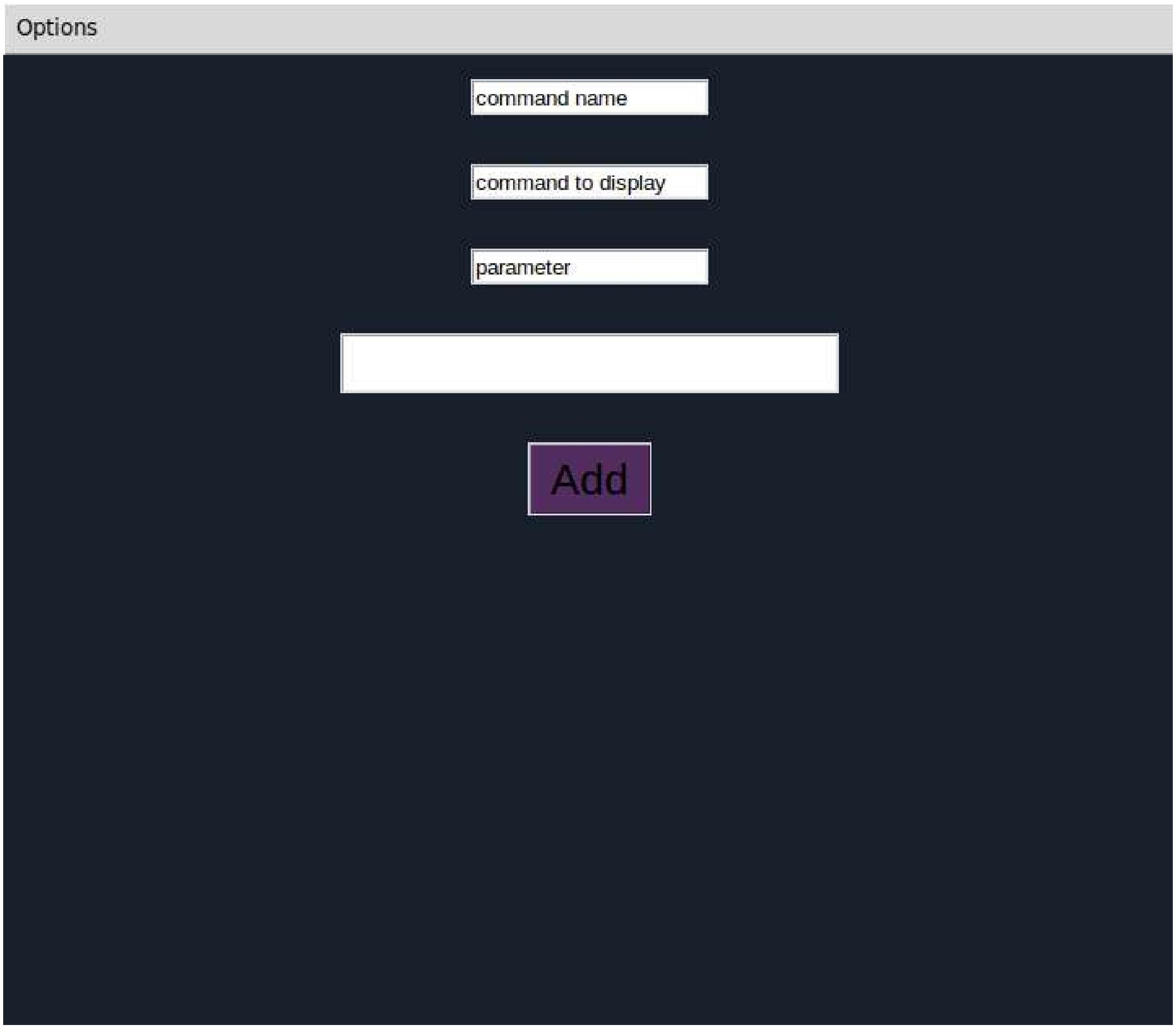
To create a parameter : (for file)



1. specify the fact that you are using parameters by typing 'Y' in 'parameter' entry field
2. put '{}' where you want parameters

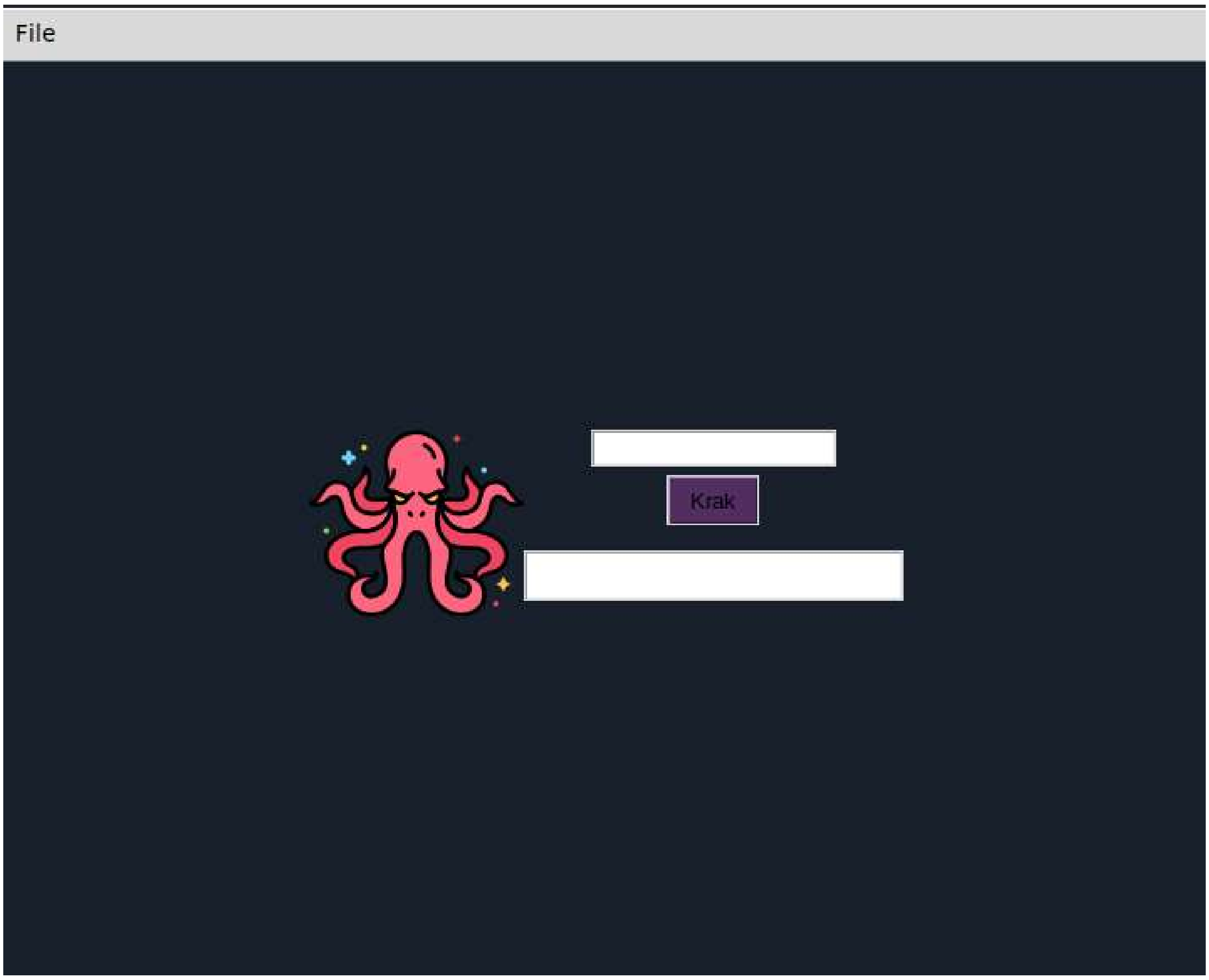
What can have a parameter ?
- file
-arg/arg2
-arg3

For command :



1. specify the using of parameters by typing ‘Y’ on ‘parameter’ entry field
2.put ‘{}’ where you need parameters

Them open ‘graphic_interface.py’ :



For file :
type :
- the main command
- **
- the first parameter (it can be : file, arg, arg2 or arg3)
- **
- the second parameter (it can be : arg, arg2, arg3)
- **
- the third parmaeter (it can be : arg3 only)

For command :
type:
- the main command
- **
- the parameter (this can only have one)

Executelist :
What is it ?
It just a list (you define this list) of commands (main commands you defined in ‘graph_import.py’) that will be displayed until you will open ‘graphic_interface.py’ .

How to define this list ?

Open ‘executelist.txt’ and here, type the main commands to display (with parameters if there is , use the same syntax used in ‘graphic_interface.py’) , **1 command = 1 line**
Then, save the file. Finally, open ‘graphic_interface.py’ and the list will be executed instantly…

Thedarkhorse-source