	Document Title	Function library guide	
	Application Name	Smart Console	
	Document Version	1.0	
	Written By	Vlad Feldfix	**Page 1 of 6**

1. Overview

1.1. The Smart-Console is a Python 3 library designed to make console applications for various purposes.

1.2. Smart-Console offers a variety of functions that can save the developer a lot of code writing.

2. How to use

2.1. Download Smart-Console.py from https://github.com/VladFeldfix/Smart-Console

2.2. Place the file in your Python folder.
(typically: C:\Python\Python313\Lib\SmartConsole.py)

2.3. Once the SmartConsole.py is among your other libraries, you can import it.

2.4. Below is a template for a program using the smart console.

```
from SmartConsole import *
class main:
    # constructor
    def __init__(self):
        # load smart console
        self.sc = SmartConsole("Sample program", "1.0")

        # set-up main menu
        self.sc.add_main_menu_item("FUNCTION1", self.fun1)
        self.sc.add_main_menu_item("FUNCTION2", self.fun2)

        # get settings
        self.path_main = self.sc.get_setting("My folder")

        # test all paths
        self.sc.test_path(self.path_main)

        # start
        self.sc.start()

    def fun1(self):
        # insert your code here and also make fun2 in the same way

        # restart
        self.sc.restart()
main()
```

Code 1 - import smart console

2.5. Once Smart-Console is implemented as an object as presented in Code1, the developer can use the smart console functions as described in section 3

3. Smart-Console Functions:

3.1. CONSTRUCTOR

3.1.1. `__init__(name, version)` - this is the constructor function and it requires the following arguments:

3.1.1.1. **Name** - the name of your application

3.1.1.2. **Version** - the version of your application

3.2. FLOW

3.2.1. `start()` - Start the application displaying the application header, version, and main menu

3.2.2. `restart()` - Restarts the application

3.2.3. `exit()` - Exits the application

3.3. MAIN MENU

3.3.1. `add_main_menu_item(name, function)` - creates a new item in the main menu

3.3.1.1. **Name** - The name of the function for example: "RUN"

3.3.1.2. **Function** - The function to call when this menu item is selected. For example: self.run

3.4. INPUT

3.4.1. `input(text)`

3.4.1.1. **text** - The prompt of the input

3.4.1.2. **RETURN VALUE**: The inserted text

3.4.2. `question(text)`

3.4.2.1. **text** - The prompt of the question

3.4.2.2. **RETURN VALUE**: True or False

3.4.3. `choose(text, options)`

3.4.3.1. **text** - The prompt of the displayed menu

3.4.3.2. **options** - A list of possible options

3.4.3.3. **RETURN VALUE**: The text of the selected option

3.5. OUTPUT

3.5.1. print(text)

3.5.1.1. **text** - The text to display

3.5.2. good(text)

3.5.2.1. **text** - The text of the good message to display

3.5.3. warning(text)

3.5.3.1. **text** - The text of the warning message to display

3.5.4. error(text)

3.5.4.1. **text** - The text of the error to display

3.5.5. fatal_error(text)

3.5.5.1. **text** - The text of the fatal error to display before exiting the application

3.5.6. hr() - Draws a horizontal line

3.6. SETTINGS

3.6.1. open_settings() - Opens "settings.txt" to edit

3.6.2. get_setting(var)

3.6.2.1. **var** - the name of the setting

3.6.2.2. **RETURN VALUE**: The value of the given setting

3.7. INFO

3.7.1. help() - Displays the help file "help.pdf"

3.7.2. svh() - Displays the file: “software software history.pdf”

3.8. FILE HANDLER

3.8.1. test_path(path)

3.8.1.1. **path** - Throws a fatal error if the given path is missing

3.8.2. open_folder(path)

3.8.2.1. **path** - Open a given folder with Windows File Explorer

3.8.3. load_csv(path)

3.8.3.1. **path** - The location of the .csv file

3.8.3.2. **RETURN VALUE**: the content of the given .csv file as a list

3.8.4. save_csv(path, data)

3.8.4.1. **path** - The location of the .csv file to save

3.8.4.2. **data** - a list to be saved as a .csv file

3.9. SCRIPT

3.9.1. run_script(path, functions)

3.9.1.1. **path** - The location of the script file to read

3.9.1.2. **functions** - a directory of functions to activate. For example:
{"START":(self.start,("Argument0","Argument1","Argument3"))
}

3.10. DATABASES

3.10.1. save_database(path, data)

3.10.1.1. **path** - The location of the file to save

3.10.1.2. **data** - a directory

3.10.2. load_database(path, headers)

3.10.2.1. **path** - The location of the file to load

3.10.2.2. **headers** - the headers of the directory

3.10.3. invert_database(database)

3.10.3.1. **database** - a directory to invert

3.10.3.2. **RETURN VALUE**: the inverted version of the given directory

3.11. DATE

3.11.1. today() - Returns today's date

3.11.1.1. **RETURN VALUE**: current date in format: YYYY-MM-DD

3.11.2. current_year()

3.11.2.1. **RETURN VALUE**: current year in format: YYYY

3.11.3. current_month()

3.11.3.1. **RETURN VALUE**: current month in format: MM

3.11.4. current_day()

3.11.4.1. **RETURN VALUE**: current day in format: DD

3.11.5. current_week()

3.11.5.1. **RETURN VALUE**: current week number in format: WW

3.11.6. current_weekday()

3.11.6.1. **RETURN VALUE**: current weekday number: 1-Sunday, 2-Monday, 3-Tuesday, 4-Wednesday, 5-Thursday, 6-Friday, 7-Saturday, in format: 0

3.11.7. now()

3.11.7.1. **RETURN VALUE**: current time in format: HH:MM

3.11.8. right_now()

3.11.8.1. **RETURN VALUE**: current time in format: HH:MM:SS

3.11.9. current_hour()

3.11.9.1. **RETURN VALUE**: current hour number in format: HH

3.11.10. current_minute()

3.11.10.1. **RETURN VALUE**: current minute number in format: MM

3.11.11. current_second()

3.11.11.1. **RETURN VALUE**: current second number in format: SS

3.11.12. current_millisecond()

3.11.12.1. **RETURN VALUE**: current millisecond number in format: MS

3.11.13. current_time()

3.11.13.1. **RETURN VALUE**: current time in format: HH:MM:SS:MS

3.11.14. test_date(givenDate)

3.11.14.1. **givenDate** - a text in the format YYYY-MM-DD

3.11.14.2. **RETURN VALUE**: True or False that the given text is in the following date format: YYYY-MM-DD

3.11.15. compare_dates(firstDate, secondDate)

3.11.15.1. **firstDate** - a text in the format YYYY-MM-DD

3.11.15.2. **secondDate** - a text in the format YYYY-MM-DD

3.11.15.3. **RETURN VALUE**: The number of days between firstDate to secondDate

4. Document version history

4.1. v1.0 Created on 2025-03-17