**РАЗРАБОТКА КОМПЬЮТЕРНОГО ПРИЛОЖЕНИЯ НА ЯЗЫКЕ PYTHON ДЛЯ СЧИТЫВАНИЯ ТЕКСТА С ИЗОБРАЖЕНИЯ.**

***Владимир Аленгоз***

*студент группы НТ-301оПИУ, факультета естествознания, математики и информатики Нижнетагильского государственного социально-педагогического института г. Нижний Тагил*

*E-mail: Vlalengoz@yandex.ru*

**DEVELOPMENT OF A PYTHON COMPUTER APPLICATION FOR READING TEXT FROM AN IMAGE.**

**Alengoz V.S.**

**Annotation**

**"In this article, the author discusses the creation of an application for reading text from the screen in a user-friendly form."**

**Keywords: library, development environment, PyTesseract , PySide2 , OpenCV , OpenCV .**

**1. Relevance of the work**

The relevance of this application lies in the fact that it is quite inconvenient for many users to rewrite the code during training from videos or other media, and for more comfortable work they try to find information in a format convenient for them.

**2. Description of the problem**

It is no longer a secret that in the modern world computer technology is an integral part of our lives. There are many applications that help us make our work easier.

In our case, we are interested in the possibility of using means of reading data from an image to implement the output of textual information in a format convenient for us.

**3. Description of the technologies used and justification of the expediency of the choice.**

First of all, before starting to develop an application, we will analyze the existing tools that are optimal for these purposes. The necessary tools are highlighted, without which it is simply impossible.

PyTesseract is a python package for OCR development. OCR or Optical Character Recognition (OCR) is a technology that allows you to convert various types of documents, such as scanned documents, PDF files or photos from a digital camera, into editable searchable formats.

PySide2 is a Python binding from the Qt cross-platform GUI Toolkit, currently being developed by the Qt Company under the Qt for Python project to port PySide to work with Qt 5 instead of Qt 4. This is one of the alternatives to the standard Tkinter library package . Like Qt, PySide2 is free software.

OpenCV is a cross-platform library with which we can develop real-time computer vision applications.

PyAutoGui is a library that allows you to emulate user actions with both a keyboard and a mouse.

**4. Justification of the choice of software tools for implementation. A comparative analysis is given with a description of the comparison criteria.**

To develop this application, you need to take into account many factors that affect the quality of development. Therefore, you need to choose the most suitable development environment, which includes all the amenities necessary to create a program.

**Sublime Text**

Sublime Text, written by a Google engineer in pursuit of the dream of the best text editor, is an extremely popular code editor. Sublime Text, supported on all platforms, has built-in Python code editing support and a rich set of extensions (called packages) with which syntax and editing capabilities are expanded. Installing additional Python packages will seem complicated: Sublime Text packages are written in Python, and installing the editor often requires executing Python scripts directly in Sublime Text.

Positive:

Sublime Text is popular in the developer community. As a code editor, Sublime Text is fast, lightweight and well supported.

Minuses:

Sublime Text is not free, although you can use the trial version for an indefinite period of time. Installing extensions will require effort, as well as the lack of direct support for executing or debugging code from the editor is a minus.

**Atom**

Available on all platforms, Atom is considered the "hacker text editor of the 21st century". A well-designed interface, working with the file system and an extension store. Free-to-access Atom is created using Electron, a framework for creating computer applications using JavaScript, HTML and CSS. Python language support is provided by an extension that can be installed when Atom is launched.

Positive:

Supported on all platforms, thanks to Electron. Atom is small in volume and loads quickly.

Minuses:

Support for working with code and debugging is not built in, but there are additions to the editor. Since Atom is built on Electron, it works in as a JavaScript process, not as an application.

**Visual Studio**

Visual Studio is a full-featured IDE that is similar to Eclipse in many ways. Built exclusively for Windows and Mac OS, VS comes with both free (Community) and paid (Professional and Enterprise) versions. Visual Studio allows you to develop for various platforms and comes with its own extension store. Python Tools for Visual Studio (PTVS) allows you to program in python in Visual Studio, and Intellisense for Python, debug.

Positive:

If you already have Visual Studio installed for other developments, adding PTVS is faster and easier.

Minuses:

Visual Studio is heavy to use for Python alone. Also, there is no Visual Studio installation for Linux.

**Visual Studio Code**

Small and lightweight, but fully functional VS-Code is open source, scalable and customizable for most tasks. Like Atom, VS Code is built on Electron, so it has the same advantages and disadvantages. Installing Python support in VS Code is simple and accessible. Look for Python in the add-ons store, click "Install" and restart if necessary. VS Code will automatically see the installation and Python libraries.

Positive:

Available on every platform.

Minuses:

Electron means that VS Code is not a separate application. In addition, for some people, the principle is not to use Microsoft products.

**PyCharm**

PyCharm supports Python development directly. Just open a new file and write the code. Run and debug Python code directly inside PyCharm. Plus it supports versioning and project management.

Positive:

It's a de facto Python IDE, with a ton of support in general and community support. Edits, runs and debugs Python out of the box.

Minuses:

PyCharm can load slowly, and the default settings need to be configured for existing projects.

I chose the VS Code development environment because this application is publicly available and less demanding on a PC.

**5. Description of the development (algorithm and code with comments).**

As resources for my development, I chose a number of libraries: PySide2, OpenCV, pytesseract, pyautogui, pynput.

Code listing:

# List for mouse coordinates

s = []

# Click processing

def on\_click(x, y, button, pressed):

if pressed:

s.append(x)

s.append(y)

else:

s.append(x)

s.append(y)

return False

# Collecting events until the stream ends

with mouse.Listener(on\_click=on\_click) as listener:

listener.join()

# Running the method to track the mouse

listener = mouse.Listener(on\_click=on\_click)

listener.start()

# Coordinates of the screen

if (s[2] > s[0]):

left = s[0]

else:

left = s[2]

if (s[3] > s[1]):

top = s[1]

else:

top = s[3]

width = s[2] - s[0]

height = s[3] - s[1]

# Screen of the area

screenshot = pyautogui.screenshot('screenshot.png', region=(left, top, abs(width), abs(height)))

# Path to connect tesseract

pytesseract.pytesseract.tesseract\_cmd = 'C:\\Program Files\\Tesseract-OCR\\tesseract.exe'

# Connecting photos

img = cv2.imread('screenshot.png')

img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

# The entire text from the image will be displayed

config = r'--oem 3 --psm 6'

print(pytesseract.image\_to\_string(img, config=config))

a = (pytesseract.image\_to\_string(img, config=config))

b = a

a = a[:len(a)//2]

data = pytesseract.image\_to\_data(img, config=config)

# Sorting through the data about text labels

for i, el in enumerate(data.splitlines()):

if i == 0:

continue

el = el.split()

try:

# Creating captions on the picture

x, y, w, h = int(el[6]), int(el[7]), int(el[8]), int(el[9])

cv2.rectangle(img, (x, y), (w + x, h + y), (0, 0, 255), 1)

cv2.putText(img, el[11], (x, y), cv2.FONT\_HERSHEY\_COMPLEX, 1, (0, 128, 0), 1)

except IndexError:

q = 0

# Displaying photos

cv2.imshow('Result', img)

cv2.waitKey(0)

self.textEdit.setText(b)

After the main code was written, I moved on to the design of the graphical interface in designer.exe .

**6. Conclusions describing the advantages of your development.**

As a result of the work , the following tasks were solved:

\* The analysis of libraries for the creation of this application has been carried out. A number of libraries were selected as development resources: PySide2, OpenCV, pytesseract, pyautogui, pynput.

\* The Python installation method is described step by step. With all the warnings.

\* IDE for Python have been disassembled.

• The program code is described in detail.

\* The results of the work are listed OCSsc.exe

This development is distinguished by the small size of the occupied memory, the possibility of using third-party storage devices without the need for an Internet connection, as well as an intuitive interface.

**7.Список использованной литературы.**

1. Маккинни, У. Python и анализ данных / У. Маккинни ; перевод с английского А. А. Слинкина. — 2-ое изд., испр. и доп. — Москва : ДМК Пресс, 2020. — 540 с. — ISBN 978-5-97060-590-5. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/131721 (дата обращения: 14.09.2021). — Режим доступа: для авториз. пользователей.
2. Косицин, Д. Ю. Язык программирования Python : учебно-методическое пособие / Д. Ю. Косицин. — Минск : БГУ, 2019. — 136 с. — ISBN 978-985-566-746-0. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/180546 (дата обращения: 14.09.2021). — Режим доступа: для авториз. пользователей.
3. Саммерфилд, М. Python на практике : учебное пособие / М. Саммерфилд ; перевод с английского А. А. Слинкин. — Москва : ДМК Пресс, 2014. — 338 с. — ISBN 978-5-97060-095-5. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/66480 (дата обращения: 14.09.2021). — Режим доступа: для авториз. пользователей.