УДК 004

Автоматизация взаимодействия с графическим интерфейсом с использованием Python

Болтовский Гавриил Александрович
Приамурский государственный университет им. Шолом-Алейхема
Студент

Аннотация
Целью данной статьи является описание создания программы на языке Python, осуществляющей автоматическое управление графическим интерфейсом операционной системы. Программа написана с использованием библиотек pyautogui и pyscreeze. Результатом исследования является готовая программа, осуществляющая автоматическое заполнение форм.
Ключевые слова: python, pyautogui, автоматизация

Automation interaction with graphical interface using Python

Boltovsky Gavriil Alexandrovich
Sholom-Aleichem Priamursky State University
Student

Abstract
The purpose of this article is to describe the creation of a program in the Python language that performs automatic control of the graphical interface of the operating system. The program is written using the pyautogui and pyscreeze library. The result of the research is a ready-made program that performs automatic form filling.
Keywords: python, pyautogui, automation

1 Введение

1.1 Актуальность исследования

В современном мире автоматизация рутинных задач становится все более востребованной. Управление графическим интерфейсом операционной системы вручную часто отнимает много времени и подвержено человеческим ошибкам. Использование библиотек, таких как pyautogui, позволяет значительно упростить выполнение подобных задач.

1.2 Обзор исследований

В процессе разработки программного обеспечения, способного автоматизировать взаимодействие с графическим интерфейсом операционной системы, значительное внимание уделяется использованию библиотек и методов, обеспечивающих высокую точность и эффективность выполнения задач. Одним из таких инструментов является библиотека

pyautogui, которая позволяет осуществлять автоматическое управление элементами интерфейса.

В статье И. С. Андриенко рассматривается применение компьютерного зрения и распознавания текста для автоматизации игры [1]. Автор демонстрирует возможности использования данных технологий для анализа и интерпретации визуальной информации, что может быть полезно при разработке программ для автоматизации интерфейса. Использование компьютерного зрения позволяет повысить точность взаимодействия с элементами интерфейса, идентифицируя их по визуальным характеристикам. Е. А. Комзалов в своей работе акцентирует внимание на автоматизации процесса предприятия с использованием языка программирования Python [2]. Автор подробно описывает различные библиотеки и инструменты, которые могут быть применены для создания автоматизированных систем управления. В частности, pyautogui рассматривается как один из ключевых инструментов для взаимодействия с графическим интерфейсом, что позволяет значительно сократить временные затраты на выполнение рутинных операций. М. Э. Нагорных, А. А. Антонов и С. А. Чернышёв исследуют программное обеспечение для дистанционного управления приложениями посредством жестов кистей рук [3]. Их работа демонстрирует альтернативный подход к взаимодействию с интерфейсом, основываясь на жестовом управлении, что может быть полезным для создания более интуитивных и удобных систем автоматизации. Хотя основной акцент сделан на жестовом управлении, принципы, изложенные в статье, могут быть адаптированы и для разработки решений на основе pyautogui. Е. Е. Ковшов и соавторы анализируют автоматизацию оценки эффективности взаимодействия конечного пользователя с обучающей информационной системой [4]. В их исследовании рассматриваются методы измерения и улучшения пользовательского опыта, что является важным аспектом при разработке автоматизированных систем. Понимание того, как пользователи взаимодействуют с системой, позволяет более точно настроить автоматизацию для достижения оптимальных результатов. С. Б. Крыльцов и М. А. Коробицына обсуждают применение Python для формирования шаблонов документов с целью автоматизации документооборота [5]. Авторы описывают практическое применение различных библиотек Python для автоматизации создания и управления документами, что является важным аспектом в контексте автоматизации рутинных задач. Pyautogui может быть использован в сочетании с другими библиотеками для создания комплексных решений, автоматизирующих весь процесс документооборота. Н. С. Рыженков и С. В. Борисова рассматривают применение компьютерного зрения в инженерном образовании [6]. Их работа подчеркивает важность использования компьютерного зрения для автоматизации различных процессов, что перекликается с применением pyautogui для автоматического управления графическим интерфейсом. Компьютерное зрение может служить дополнительным инструментом для повышения точности и эффективности автоматизации.

1.3 Цель исследования

Целью данного исследования является разработка программы на языке Python для автоматизации взаимодействия с графическим интерфейсом операционной системы с использованием библиотеки pyautogui.

1.4 Постановка задачи

Задачами исследования являются изучение возможностей библиотеки pyautogui для автоматизации действий пользователя, разработка программы, способной автоматически выполнять типичные действия, такие как навигация по меню и заполнение форм, а также тестирование и оценка эффективности разработанной программы.

2 Методы исследования

Для достижения поставленных целей были использованы следующие методы: анализ документации и примеров использования библиотеки pyautogui [7] и pyscreeze [8], разработка и отладка кода на языке Python с использованием pyautogui, а также экспериментальное тестирование программы в различных сценариях использования.

3 Результаты и обсуждения

Существуют два основных метода для автоматизации взаимодействия с графическим интерфейсом:

1. Использование координатных точек: Этот метод включает определение точных координат на экране, где расположены интересующие нас элементы (например, поля ввода или кнопки). После идентификации координаты передаются функциям автоматизации, таким как pyautogui.click() для выполнения кликов или pyautogui.typewrite() для ввода текста.
2. Поиск по изображениям: Этот подход основан на использовании библиотеки pyscreeze, которая позволяет искать элементы GUI на экране по их изображениям. Вместо фиксированных координат точек ищутся изображения, соответствующие полям ввода и кнопкам. Это делает скрипт более гибким, поскольку не зависит от конкретных координат и может адаптироваться к изменениям в интерфейсе.

Для демонстрации первого метода, рассмотрим пример кода на Python, который осуществляет автоматический ввод логина и пароля на странице входа в социальную сеть ВКонтакте, используя фиксированные координаты элементов (рис. 1).

```
1  import pyautogui
2  import time
3
4  time.sleep(5)
5  login_field_coords = (x1, y1)
6  password_field_coords = (x2, y2)
7  login_button_coords = (x3, y3)
8
9  login = "your_login"
10 password = "your_password"
11
12
13 pyautogui.click(login_field_coords)
14 pyautogui.typewrite(login)
15
16 time.sleep(1)
17
18 pyautogui.click(password_field_coords)
19 pyautogui.typewrite(password)
20
21 time.sleep(1)
22
23 pyautogui.click(login_button_coords)
```

Рисунок 1 – Код программы (метод 1)

Первым шагом является задержка выполнения программы с помощью функции `time.sleep(5)` (строка 4). Это дает пользователю время для переключения на окно браузера и открытия страницы входа в социальную сеть. Задержка в 5 секунд обычно достаточна для этого.

Далее в коде указываются координаты полей ввода логина, пароля и кнопки входа на странице ВКонтакте. Эти координаты (`login_field_coords`, `password_field_coords`, `login_button_coords`) представляют собой кортежи с числовыми значениями, которые указывают на точное положение каждого элемента на экране (строки 5-7). Координаты можно определить с помощью вспомогательных функций PyAutoGUI, которые позволяют выводить текущие координаты курсора или определять координаты на изображении.

Далее идет определение логина (`login`) и пароля (`password`), которые будут использоваться для автоматического ввода на странице входа (строки 9-10). Эти значения хранятся в переменных строкового типа и могут быть изменены пользователем в зависимости от необходимости.

После этого начинается непосредственно взаимодействие с элементами интерфейса. Сначала происходит клик по полю ввода логина с помощью функции `pyautogui.click(login_field_coords)`. После этого вызывается `pyautogui.typewrite(login)`, что позволяет автоматически ввести логин, сохраненный в переменной `login` (строки 13-23).

Затем делается небольшая задержка (`time.sleep(1)`), чтобы обеспечить стабильность работы программы и предотвратить возможные ошибки из-за несинхронности операций.

После задержки осуществляется клик по полю ввода пароля (`pyautogui.click(password_field_coords)`) и ввод самого пароля (`pyautogui.typewrite(password)`), хранящегося в переменной `password`.

После ввода пароля снова происходит небольшая задержка, а затем автоматически происходит клик по кнопке входа на странице ВКонтакте.

Этот код демонстрирует простую форму автоматизации взаимодействия с элементами веб-страницы, используя фиксированные координаты экрана для определения местоположения элементов интерфейса. Такой подход удобен для задач, где расположение элементов на странице постоянно и предсказуемо. Однако следует учитывать, что изменения в интерфейсе могут потребовать пересмотра координат для корректной работы программы.

Библиотека pyscreeze включает в себя функции для работы с изображениями, такие как pyautogui.locateOnScreen() для поиска изображения на экране и pyautogui.click() для клика по найденному изображению. Это позволяет автоматизировать процессы, требующие взаимодействия с различными элементами пользовательского интерфейса без необходимости предварительно задавать точные координаты.

Для демонстрации второго метода рассмотрим другой код. На рисунке приведен пример, использующий библиотеку pyscreeze для автоматического ввода логина и пароля на странице входа в социальную сеть (рис. 2).

```python
import pyautogui
import pyscreeze
import time

def find_image_on_screen(image_path, region=None):
    try:
        if region is None:
            loc = pyautogui.locateOnScreen(image_path)
        else:
            loc = pyautogui.locateOnScreen(image_path, region=region)
        if loc:
            return pyautogui.center(loc)
        else:
            return None
    except Exception as e:
        print(f"Error finding image: {e}")
        return None

def click_image(image_path, region=None):
    try:
        center = find_image_on_screen(image_path, region)
        if center:
            pyautogui.click(center)
            return True
        else:
            return False
    except Exception as e:
        print(f"Error clicking image: {e}")
        return False

time.sleep(5)
login_image_path = 'login_field.png'
password_image_path = 'password_field.png'
login_button_image_path = 'login_button.png'
if click_image(login_image_path):
    pyautogui.typewrite("your_login")
if click_image(password_image_path):
    pyautogui.typewrite("your_password")
click_image(login_button_image_path)
```

Рисунок 2 – Код программы (метод 2)

Программа начинает выполнение с задержки в 5 секунд (строка 31), что дает пользователю время для переключения на окно браузера или приложения, где требуется автоматизация ввода логина и пароля. Эта задержка необходима для обеспечения стабильной работы скрипта и корректного определения изображений на экране.

Далее определены три ключевые функции: `find_image_on_screen` (строка 5), `click_image` (строка 19) и основной блок выполнения кода (строка 35).

Функция `find_image_on_screen (image_path, region=None)` отвечает за поиск изображения на экране. Она использует библиотеку `pyscreeze`, которая в свою очередь использует алгоритмы компьютерного зрения для нахождения совпадений изображений. В первую очередь функция пытается найти изображение `image_path` на весь экран или в указанной области `region`, если она задана. Если совпадение найдено, функция возвращает центральные координаты найденного изображения в виде кортежа `(x, y)`. В случае ошибки или если изображение не было найдено, функция возвращает `None`.

Функция `click_image(image_path, region=None)` использует `find_image_on_screen` для поиска изображения `image_path` и, в случае успешного нахождения, кликает по центру найденного изображения с помощью `pyautogui.click()`. Функция возвращает `True`, если клик был выполнен успешно, и `False` в случае ошибки или если изображение не было найдено.

Основной блок кода начинается с задания путей к изображениям, представляющим поля логина, пароля и кнопку входа. Эти пути `login_image_path`, `password_image_path` и `login_button_image_path` указывают на соответствующие файлы изображений, которые должны быть предварительно подготовлены и сохранены в формате PNG.

Далее последовательно вызываются функции `click_image` для каждого из изображений:

1. Для изображения поля логина, что позволяет ввести логин с помощью `pyautogui.typewrite("your_login")`.
2. Для изображения поля пароля, что позволяет ввести пароль аналогично.
3. И наконец, для изображения кнопки входа, что приводит к выполнению клика по кнопке для входа в систему.

Этот подход основывается на предварительно подготовленных изображениях элементов интерфейса, что делает код более гибким и менее зависимым от конкретных координат на экране. Он также улучшает надежность автоматизации, поскольку позволяет быстрее адаптироваться к изменениям в интерфейсе программного обеспечения.

Таким образом, приведенный код демонстрирует эффективный метод автоматизации взаимодействия с GUI на основе поиска и кликов по изображениям, что полезно в задачах тестирования программного

обеспечения, автоматизации рутинных действий и других сценариях, где требуется автоматическое управление пользовательским интерфейсом.

Разработанная программа успешно выполняет функции автоматического заполнения форм с использованием предопределенных данных, навигации по меню программного обеспечения и операционной системы, а также кликанья на определенные кнопки и выполнения скриптов по заданному сценарию. Программа была протестирована на Windows.

3. Выводы

В результате проведенного исследования была разработана и протестирована программа на языке Python, использующая библиотеку pyautogui для автоматизации взаимодействия с графическим интерфейсом операционной системы.

Библиографический список

1. Андриенко И. С. Автоматизация игры с использованием компьютерного зрения и распознавания текста //Постулат. 2023. №. 7.
2. Комзалов Е. А. Автоматизация процесса предприятия используя язык программирования Python // Внедрение передового опыта и практическое применение результатов инновационных исследований. 2021. С. 68-71.
3. Нагорных М. Э., Антонов А. А., Чернышёв С. А. Программное обеспечение для дистанционного управления приложениями посредством жестов кистей рук // Международный журнал информационных технологий и энергоэффективности. 2021. Т. 6. №. 1. С. 34-41.
4. Ковшов Е. Е. и др. Автоматизация оценки эффективности взаимодействия конечного пользователя с обучающей информационной системой // Открытое образование. 2010. №. 1. С. 37-43.
5. Крыльцов С. Б., Коробицына М. А. Применение Python для формирования шаблонов документов с целью автоматизации документооборота // Современные образовательные технологии в подготовке специалистов для минерально-сырь. 2020. Т. 5. С. 197.
6. Рыженков Н. С., Борисова С. В. Применение компьютерного зрения в инженерном образовании // Информатизация инженерного образования. 2020. С. 134-138.
7. PyAutoGUI. URL: https://www.pyautogui.readthedocs.io/ (дата обращения: 30.06.2024).
8. PyAutoGUI. URL: https:// www.github.com/asweigart/pyscreeze (дата обращения: 30.06.2024).