

Автоматизация с помощью скриншотов

Виталий Шульга, EPAM Systems

О докладчике

Виталий Шульга

Software Test Automation Engineer
в EPAM Systems
Минск, Беларусь



www.linkedin.com/in/vitalliuss
vitalliuss@gmail.com

План доклада

1. Технология визуального поиска

- Что такое визуальный поиск?
- В чем отличие от стандартных решений?
- Что мне это даст?
- В каких случаях это выгодно?
- Чем можно воспользоваться?

2. Практические рекомендации

3. В чем подвох?

Что такое визуальный поиск?

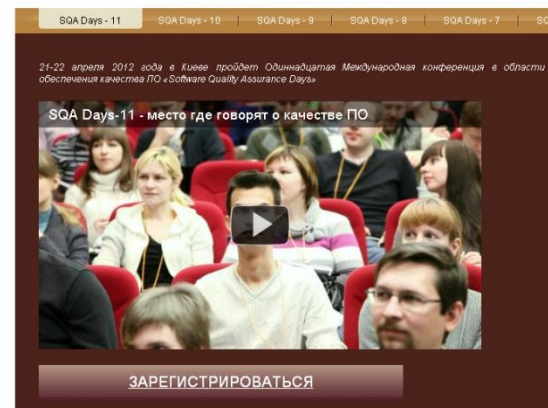
Сделать снимок экрана



Найти изображение на снимке



Выполнить необходимое действие



[ЗАРЕГИСТРИРОВАТЬСЯ](#)

[ЗАРЕГИСТРИРОВАТЬСЯ](#)

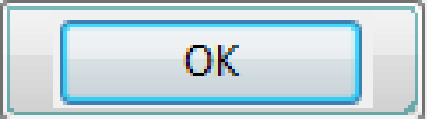


Пример скрипта с визуальным поиском

```
1 Settings.MoveMouseDelay = 0
```

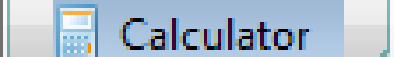
```
2 click (  )
```

```
3 click (  )
```

```
4 wait (  )
```

```
5 type ("calc")
```

```
6 click (  )
```

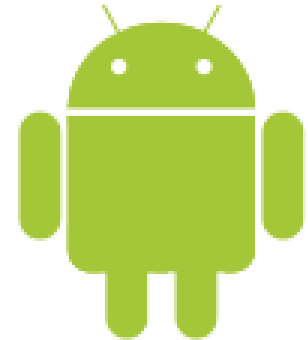
```
7 wait (  Calculator , 8.0 )
```

```
8 click (  )
```

```
9 click (  )
```

Что нам это даст?

✓ **Независимость от платформы и технологии**



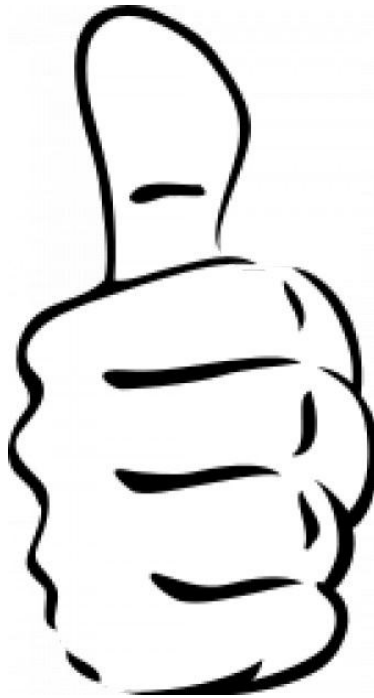
Что нам это даст?

✓ Простота реализации



Что нам это даст?

- ✓ **Сокращение затрат времени и усилий.**



В каких случаях это выгодно?

- Нет доступа к свойствам элементов
- Свойства есть, но они постоянно меняются
- Приложение больше не обновляют
- У нас недостаточно времени
- У нас недостаточно опыта
- Мы решили упростить себе жизнь 😊

Чем можно воспользоваться?

eggPlant

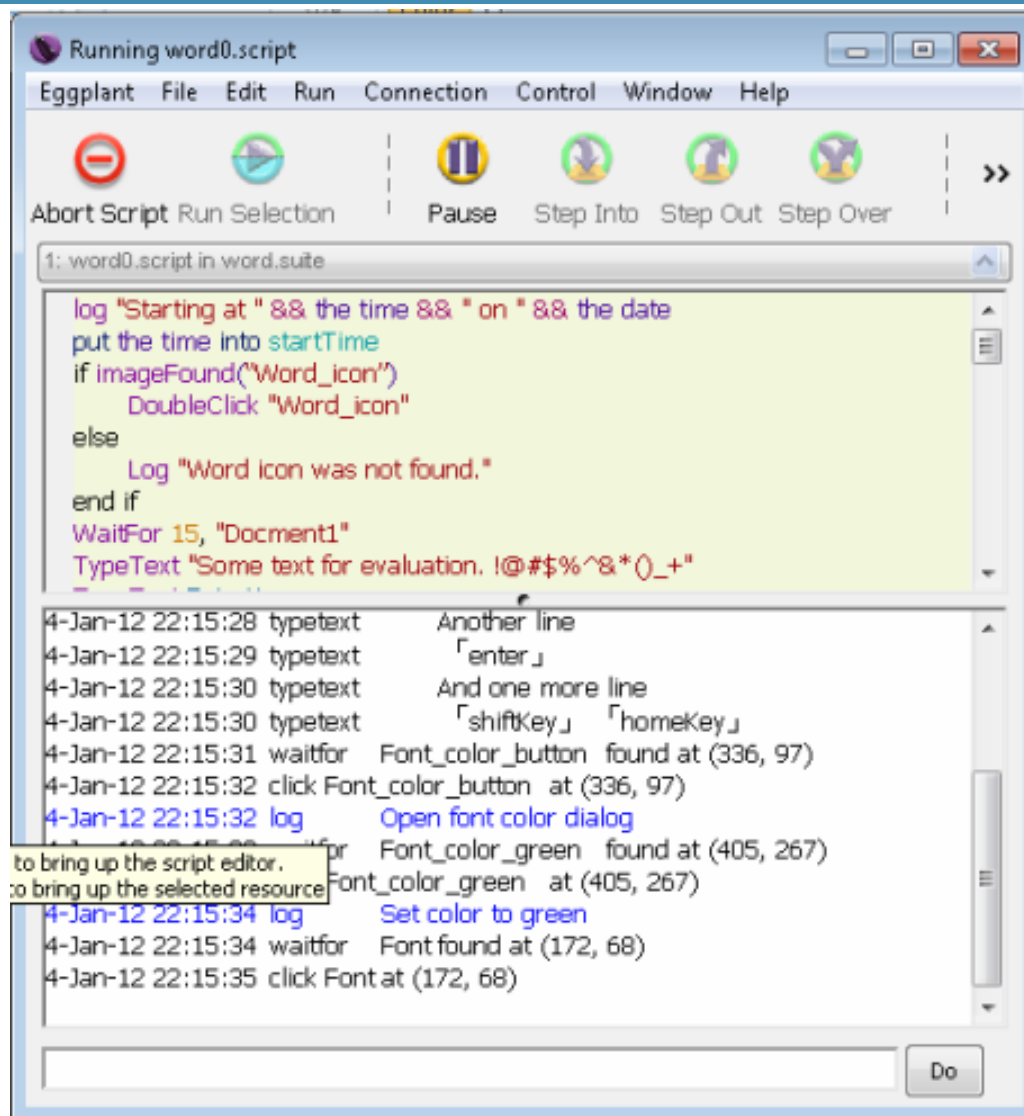


eggPlant

www.testplant.com

- Создан как инструмент тестирования
- Сильный модуль распознавания текста
- Высокая стабильность работы
- Хорошая система отчетов
- **Платный**
- **Работает только с удаленной машиной (VNC).**

EggPlant: выполнение сценария





www.sikuli.org

- **Инструмент бесплатный**
- **Прост в развертывании и использовании**
- **Не требует дополнительной машины**
- **Есть API для Java**
- **Возможность использовать Jython**
- **Слабый модуль распознавания текста**
- **Отсутствие подробного отчета.**

Sikuli: IDE

The screenshot displays the Sikuli IDE window titled "Sikuli X-1.0rc3 (r905) - calc.sikuli". The interface includes a menu bar (File, Edit, Run, View, Tools, Help), a toolbar with icons for "Take screenshot", "Insert image", "Create Region", "Run", and "Run in slow motion", and a search bar labeled "Find".

On the left side, there are three panels for action selection:

- Find:** contains actions like `exists()`, `find()`, `findAll()`, `wait()`, and `waitVanish()`.
- Mouse Actions:** contains actions like `click()`, `doubleClick()`, `rightClick()`, `hover()`, and `dragDrop()`.
- Keyboard Actions:** contains actions like `type()` and `paste()`.

The main editor area shows a script for opening a calculator, with visual overlays on the code indicating the actions performed:

```
1 Settings.MoveMouseDelay = 0
2 from datetime import *
3 timeformat = '%H:%M:%S'
4 start = datetime.now()
5 print "Start " + start.strftime(timeformat)
6 click( [Windows Logo] )
7 click( [Run...] )
8 wait( [OK] )
9 type("calc")
10 click( [OK] )
11 wait( [Calculator], 8.0 )
12 click( [1] )
13 click( [2] )
```

At the bottom, there is a "Message" and "Test Trace" panel.

Краткое сравнение

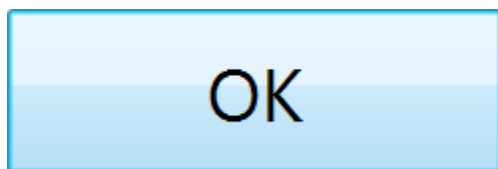
	eggPlant	sikuli
Лицензия	-	+
Язык программирования	-	+
Система отчетов	+	-
Распознавание текста	+	-
Дополнительная машина	-	+
API	-	+

- Используем красивые имена изображений



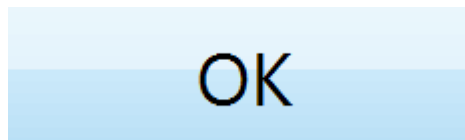
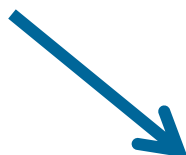
- **click(“1330030896672.png”)**
- **click(“button_close.png”)**

- Используем изображения многократно



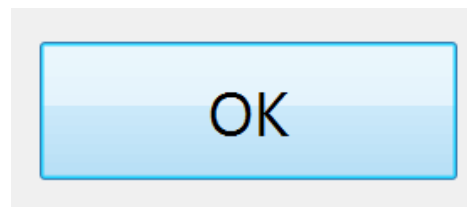
OK

“MainPage_OK”



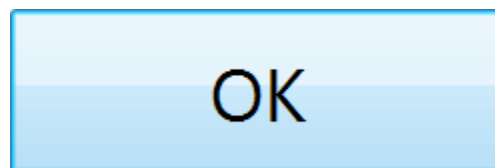
OK

“Popup_OK”



OK

“Button_OK”



OK

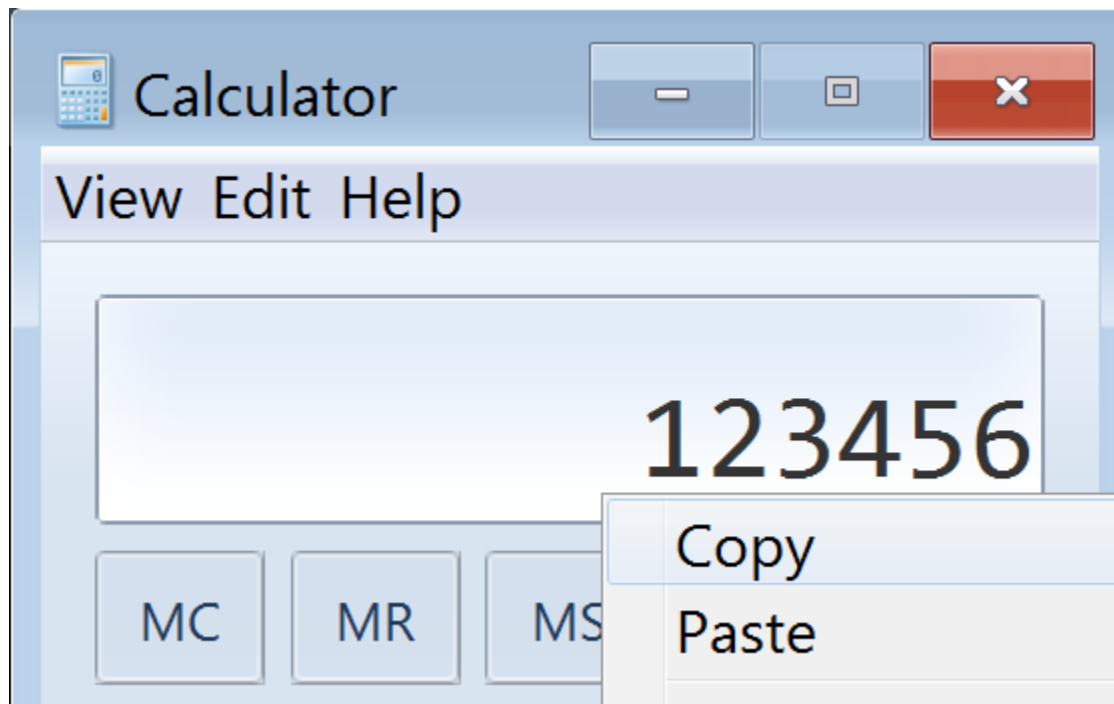
“OK”

- Доверяем только проверенным элементам

No. of digits after decimal:	<input type="text" value="2"/>
Digit grouping symbol:	<input type="text" value=","/>
Digit grouping:	<input type="text" value="123,456,789"/>
Negative sign symbol:	<input type="text" value="-"/>

```
digitsLabel = find(images/config_digits.png)  
digits = digitsLabel.targetOffset(300, 0)
```

- Сократим распознавание текста



- **Создаем удобный лог**

```
[log][15:12:54] >>> Test start
```

```
[log][15:12:54] >>> Assign: PauseOnFail=True
```

```
[log][15:12:54] >>> TestScenario: ParallelRunTest
```

```
[log][15:12:54] >>> =====
```

```
[log][15:12:54] >>> TestCase: Keying activities
```

```
[log][15:12:54] >>> LazyCure.IsOpenState
```

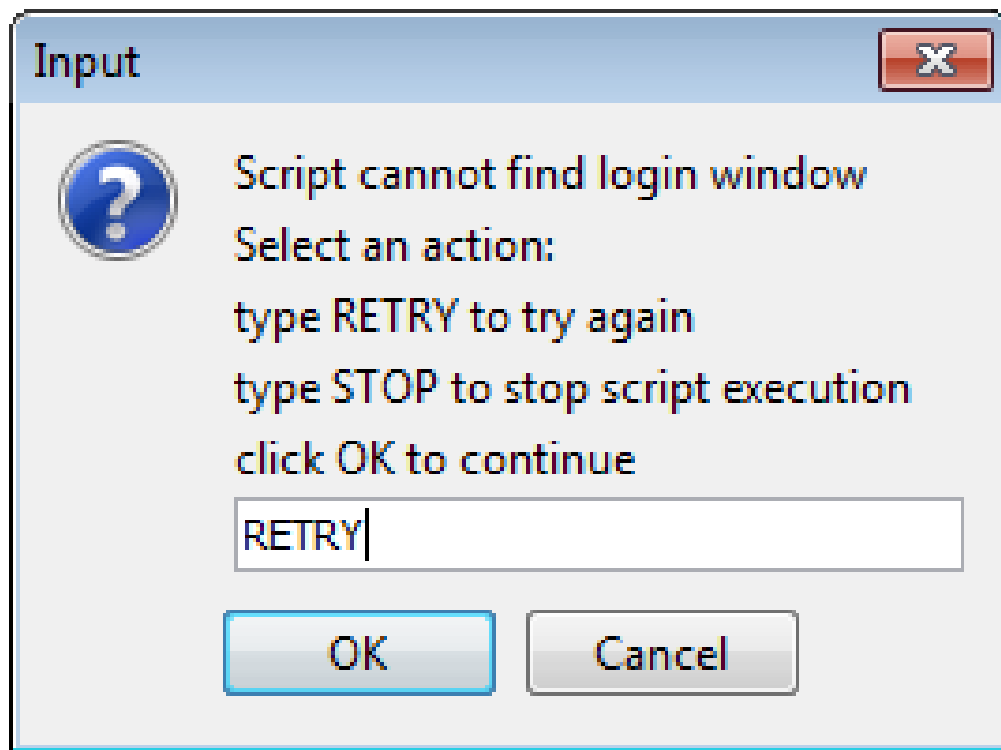
```
[log][15:12:54] >>> Call activity window
```

```
[log] RIGHT CLICK on (1139,927)
```

```
[log][15:13:02] >>> Search for activity window
```

```
[log] CLICK on (1052,736)
```

- Используем взаимодействие с пользователем



- Используем больше возможностей инструмента
 - Импорт jar
 - Единое хранилище изображений
 - Поддержка Jython
 - Встроенный режим unit-тестирования
 - Подсветка найденных элементов.

- Настроим инструмент для себя
 - `Settings.MoveMouseDelay`
 - `Settings.MinSimilarity`
 - Обработка `Find Failed`.



- Используем вызов через API

Java + Sikuli

В чем подвох?

- **Зависимость от графического интерфейса**
- **Уменьшение скорости выполнения сценария**
- **Необходимость поддержки скрипта в готовности**
- **Трудности с получением результатов работы.**

Вопросы

Краткий план доклада

- Принцип работы визуального поиска
- Отличие от стандартных решений
- Плюсы и минусы технологии
- Случаи рационального использования
- Обзор рынка инструментов
- Переиспользование изображений
- Взаимодействие с пользователем
- Использование API
- Настройка инструмента

О докладчике

Виталий Шульга

Software Test Automation Engineer
в EPAM Systems
Минск, Беларусь



www.linkedin.com/in/vitalliuss
vitalliuss@gmail.com