AUTOMATYZACJA TESTÓW APLIKACJI OKIENKOWYCH Z UŻYCIEM UIAUTOMATION

Milena Sobolewska, Paweł Maciejewski, Rule Financial

PLAN PREZENTACJI

- O nas
- O projekcie
- Dlaczego UIAutomation
  - O UIAutomation ogólnie
  - O prostocie użycia
  - O elastyczności
- Testowalny interfejs
- Zdalne uruchamianie
- Architektura i demo przykładowego projektu
- Pytania

O NAS

Milena Sobolewska

sob.milena@gmail.com

- Ponad cztery lata doświadczenia w testowaniu oprogramowania, w tym dwa lata w automatyzacji
- C#, Specflow, Java, JMeter, Sikuli, TCL/Expect
- ISTQB Foundation
- Microsoft Programming in C#

Paweł Maciejewski

pawel@maciejewski.in

- Ponad dwa lata doświadczenia w testowaniu oprogramowania, głównie w automatyzacji
- C#, Specflow, Ruby, Capybara, RSpec
- ISTQB Foundation
- Microsoft Programming in C#

O PROJEKCIE

- **Klient:** duży brytyjski bank inwestycyjny
- **Projekt:** okienkowa aplikacja wewnętrzna do edycji i wizualizacji danych bankowych
- **Zastane:** niskiej jakości framework oparty o TestComplete wykonujący testy półautomatycznie
- **Obecnie:** framework oparty o UIAutomation wraz z pełnym rozwiązaniem zdalnego wykonania

DLACZEGO UIAUTOMATION

- Standardowy element biblioteki .Net (od wersji 3.0)
- Zaprojektowany w celu automatyzacji użycia interfejsów tworzonych przy użyciu WPF
- Dodawany do projektu w postaci referencji (biblioteki)
- Pozwala na pełną kontrole kodu testów
- **Prosty w użyciu i elastyczny**

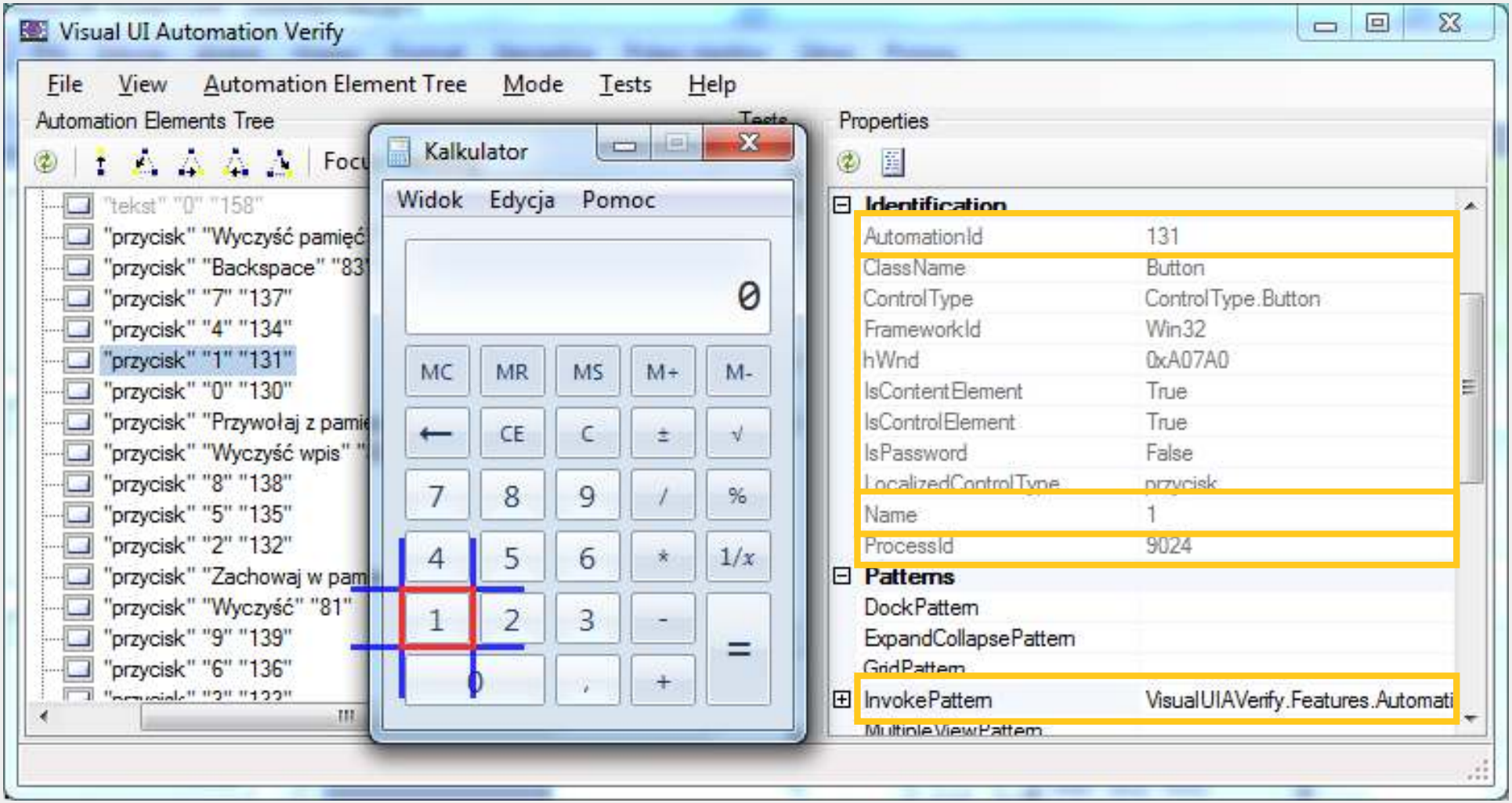
PROSTY W UŻYCIU

PROSTY W UŻYCIU C.D.

```
AutomationElement MyDesktop =
                    AutomationElement.RootElement;

AutomationElement Calc = MyDesktop.FindFirst(
    TreeScope.Children,
    new PropertyCondition(
        AutomationElement.AutomationIdProperty,
        "Calc"));
```

PROSTY W UŻYCIU C.D.



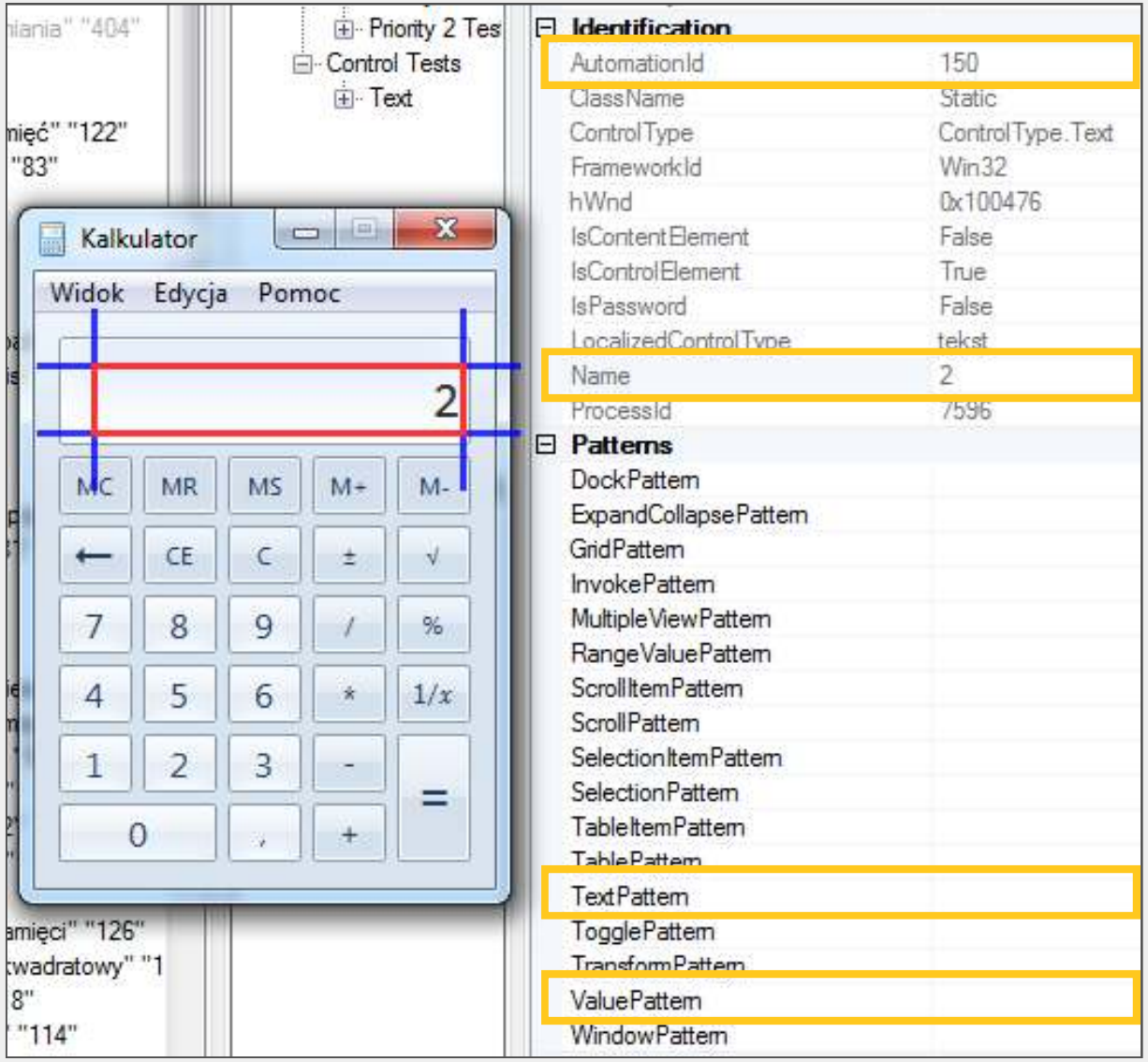
ELASTYCZNY

```
PropertyCondition TypeButton =
    new PropertyCondition(
        AutomationElement.ControlTypeProperty,
        ControlType.Button);

PropertyCondition ElName =
    new PropertyCondition(
        AutomationElement.NameProperty, "1");

AutomationElement ButtonOne = Calc.FindFirst(
    TreeScope.Children,
    new AndCondition(TypeButton, ElName));
```

ELASTYCZNY C.D.



PROSTY + ELASTYCZNY


```
Condition AllListEls = new
    PropertyCondition(
        AutomationElement.ControlTypeProperty,
        ControlType.ListItem);

Condition NotPleaseChoose = new
    NotCondition(new
        PropertyCondition(
        AutomationElement. AutomationIdProperty,
        "PleaseChoose"));

AutomationElementCollection Countries =
Calc.FindAll(TreeScope.Children, new
AndCondition(AllListEls, NotPleaseChoose));
```

TESTOWALNY INTERFEJS

- Charakterystyka testowalnych automatycznie interfejsów
- Dostępne właściwości oraz akcje dla elementów aplikacji
- Niestandardowe metody pracy z trudno testowalnymi interfejsami aplikacji

ZDALNIE URUCHAMIANIE

- Uruchamianie testów poprzez **połączenie zdalnego pulpitu** i ograniczenia tego rozwiązania
- Uruchamianie testów poprzez **serwis systemowy** i ograniczenia tego rozwiązania
- Uruchamianie testów poprzez **połączenie zdalnego pulpitu i serwis systemowy**

ARCHITEKTURA I DEMO

PRZYKŁADOWEGO PROJEKTU

PYTANIA?

Kontakt do nas:

sob.milena@gmail.com

pawel@maciejewski.in