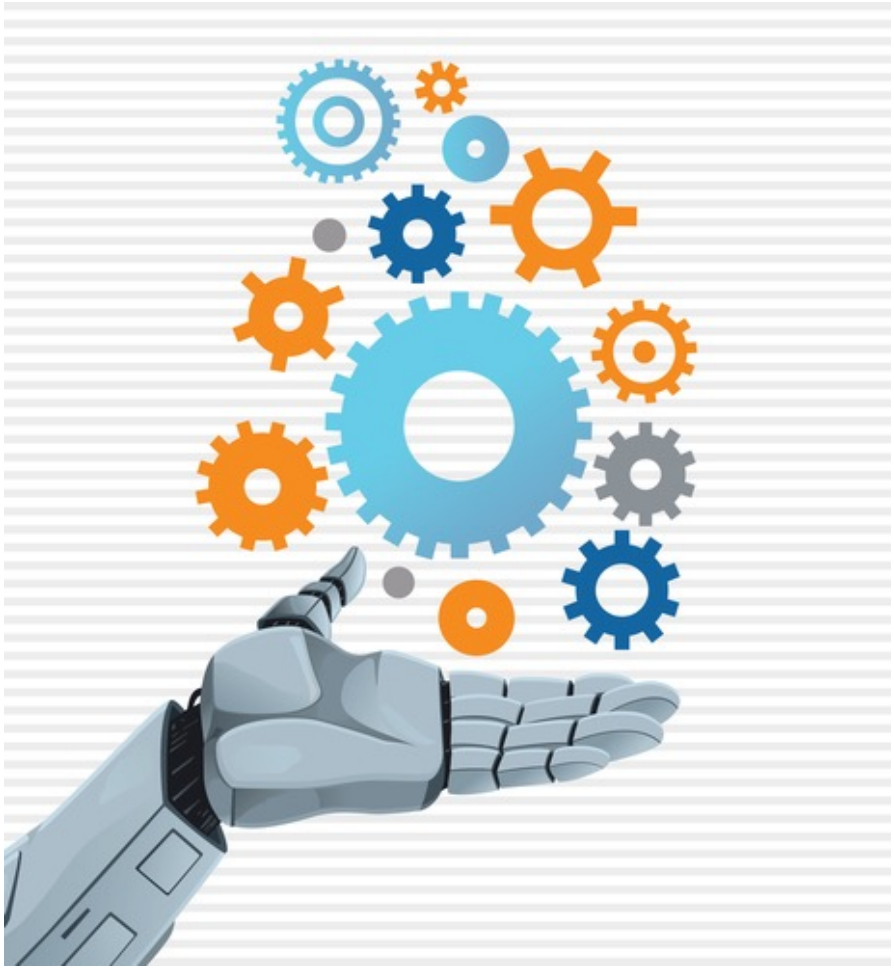
Introduction to automation testing

Join at

slido.com

#6940



What is automation testing?

- In software **testing, test automation** is the use of special software (separate from the software being **tested**) to control the execution of **tests** and the comparison of actual outcomes with predicted outcomes.

evozon

Why use automation testing?

- Automation testing is cheaper
- Automation testing is faster
- Automation testing is reliable
- Automation testing reduces human and technical risks
- Rapid feedback during development
- Automation testing reduces testing resources

evozon

When...

TO AUTOMATE

evozon

When to automate?

- Projects with long lifespan

When to automate?

- Projects with long lifespan
- Start early

evozon

When to automate?

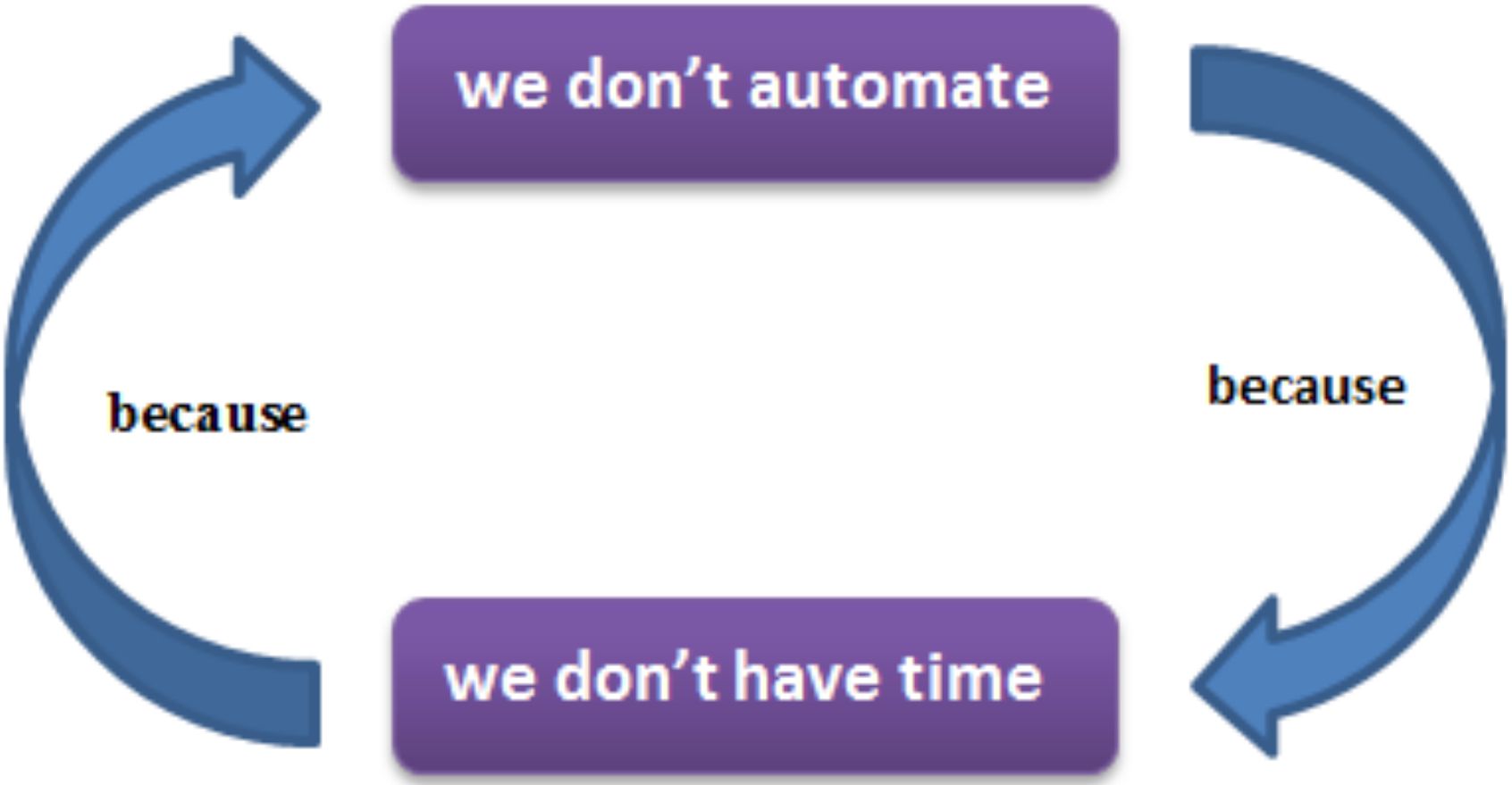
- Projects with long lifespan
- Start early
- Experienced and technical testers

evozon

When to automate?

- Projects with long lifespan
- Start early
- Experienced and technical testers
- Frequent regression testing

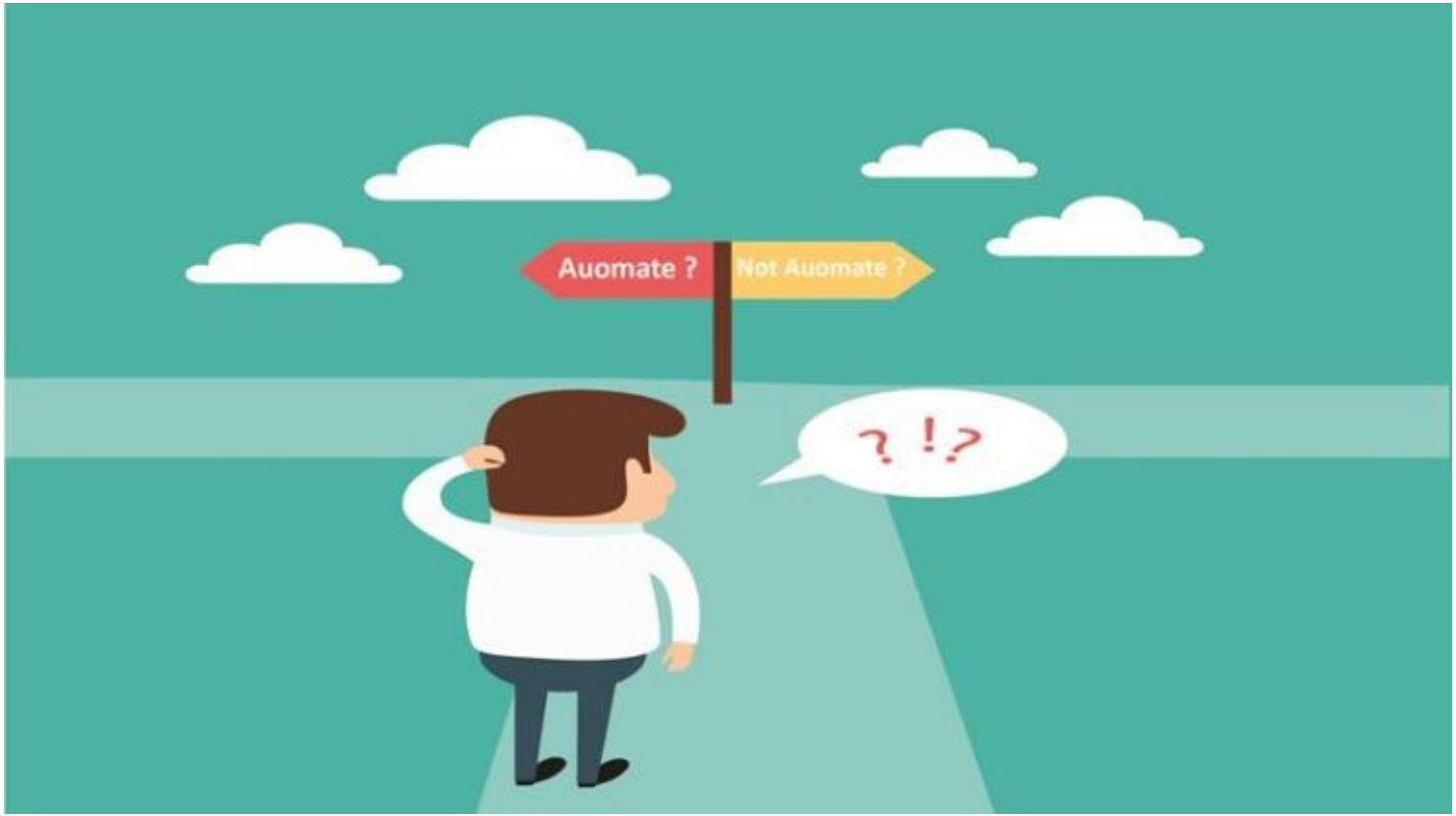
evozon


we don't automate
because
we don't have time
because

When NOT to automate?

- Projects with short lifespan
- Unstable design
- Inexperienced testers
- Insufficient time, resources

What to automate?



evozon

What to automate?

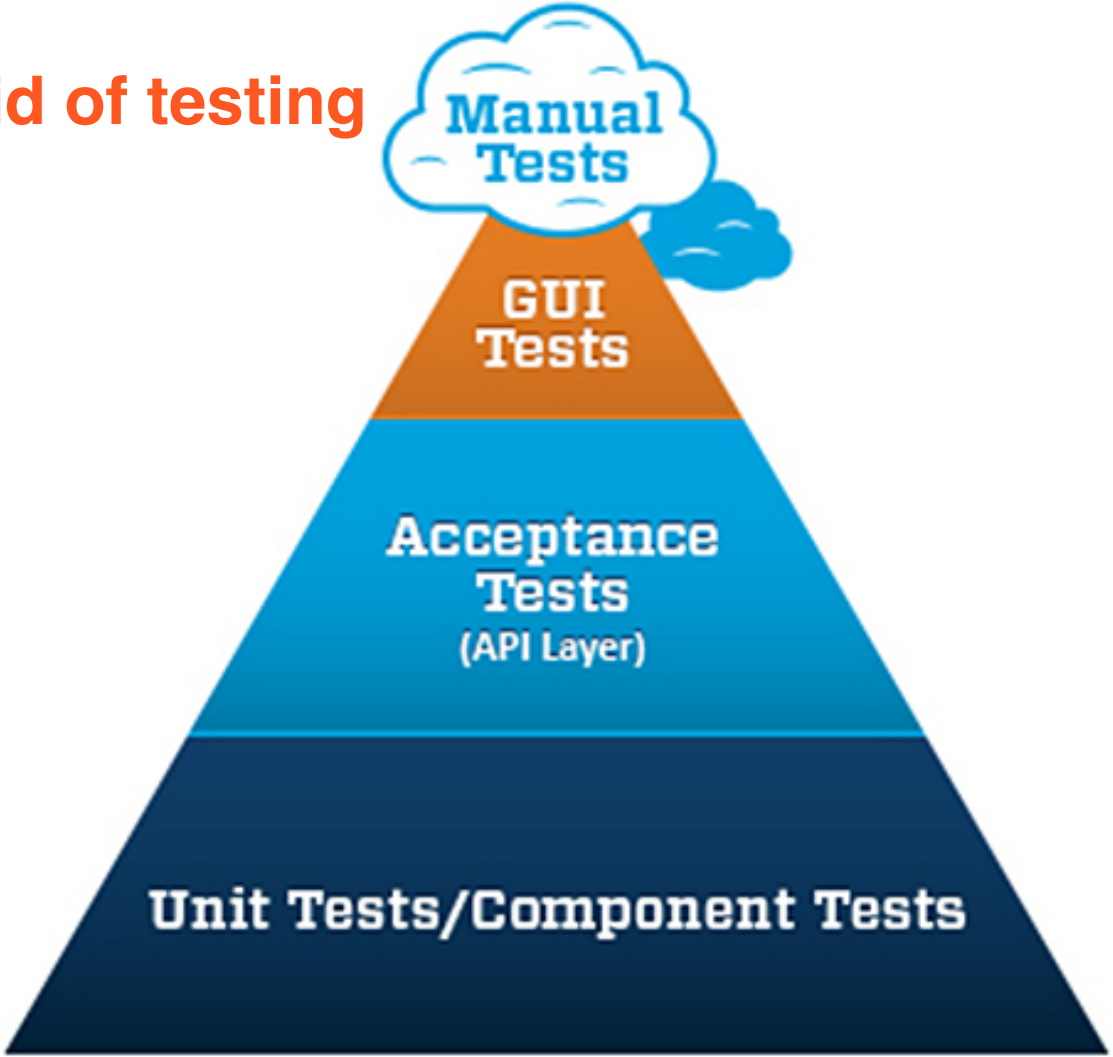
- Core application features
- Sanity/Smoke tests
- Regression tests
- Tests you need to run several times
- Areas of the app which are more “problematic”
- Things that are easy to automate and provide a “quick win”


ONE DOES NOT SIMPLY
AUTOMATE ALL OF THE APPLICATION

What NOT to automate?

- Areas that will change
- Tests that are run only a few times
- Data validation

The pyramid of testing



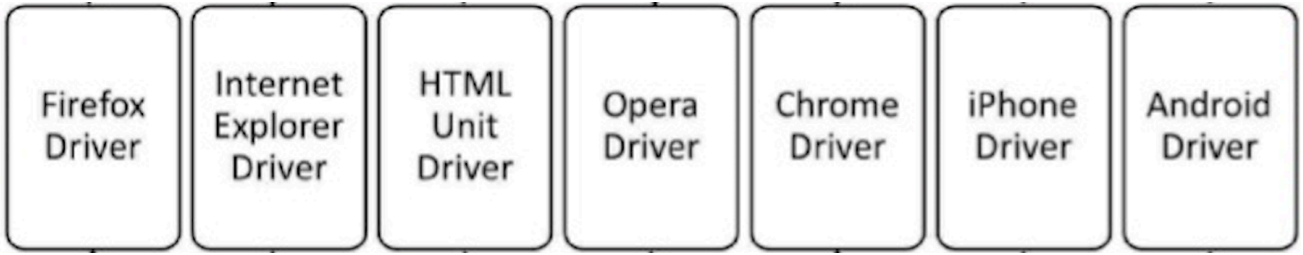
How to automate?

- Automation is more than a record and play tool
- Automation is more than test execution
- Automation needs a framework
- Good automation needs code review and refactoring
- Automation needs maintenance

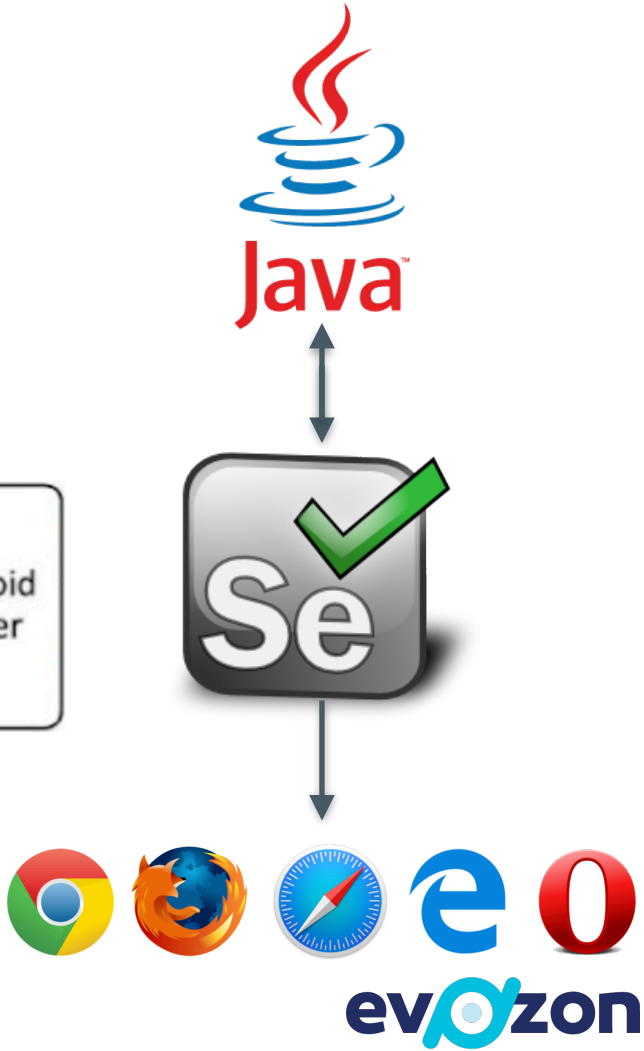
What to use to automate?

What do we need

- Programming language
- Selenium WebDriver



- Browser



evozon

Selenium WebDriver

- **WebDriver** is a tool for automating web application testing, and in particular to verify that they work as expected.

- Open source framework that can be used for writing and running automated tests that can be integrated with Selenium to produce reliable automation tests

The structure of an automated test

- Navigation
- Interactions (click, type, read text, select dropdown, etc)
- Verification (a test is not a test, unless it checks something)

evozon

Serenity BDD

What is Serenity BDD

- Test automation framework

- Helps you write better, more effective automated acceptance tests, and use these acceptance tests to produce world-class test reports and living documentation

- Previously know as thucydides

Why use

Serenity BDD

- Already an established framework
- Easy to use
- Great reporting
- Is based on Selenium WebDriver
- A lot of already implemented methods (helper methods)

evozon

Automated test

This is a small script. Script maintenance looks easy. But with time test suite will grow. As you add more and more lines to your code, things become tough.

The chief problem with script maintenance is that if 10 different scripts are using the same page element, with any change in that element, you need to change all 10 scripts. This is time consuming and error prone.

```
public class NoPOMTest99GuruLogin {

    /**
     * This test case will login in http://demo.gu
     * Verify login page title as guru99
     * Login to application
     * Verify the home page using Dashboa
     */
    @Test(priority=0)
    public void test_Home_Page_Appear_Cor
        WebDriver driver = new FirefoxDri
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
        driver.get("http://demo.guru99.com/V4/");
        //Find user name and fill user name
        driver.findElement(By.name("uid")).sendKeys("demo");
        //find password and fill it
        driver.findElement(By.name("password")).sendKeys("password");
        //click login button
        driver.findElement(By.name("btnLogin")).click();
        String homeText = driver.findElement(By.xpath("//table//tr[@class='heading3']")).getText();
        //verify login success
        Assert.assertTrue(homeText.toLowerCase().contains("guru99 bank"));
    }
}
```

1. Find user name and fill it
2. Find password and fill it
3. Find Login button and click it
4. Find home page text and get it
5. Verify home page has text 'Guru99 Bank'

evozon

Automated test

- A better approach to script maintenance is to create a separate class file which would find web elements, fill them or verify them.

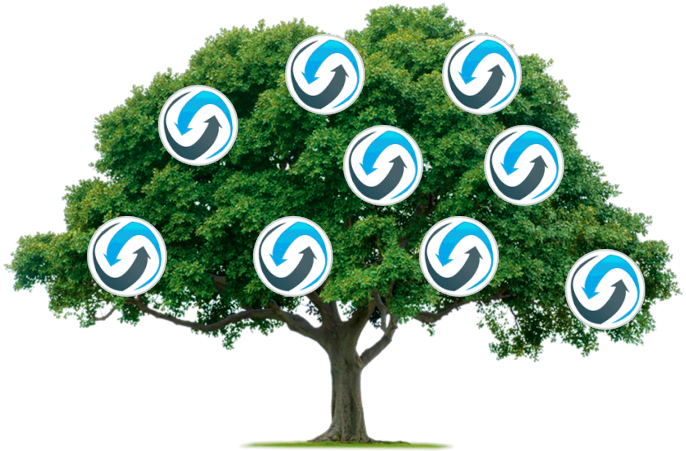
- This class can be reused in all the scripts using that element.

- In future, if there is a change in the web element, we need to make the change in just 1 class file and not 10 different scripts. Interactions (click, type, read text, select dropdown, etc)

What is POM?

- **Page Object Model** is a design pattern to create **Object Repository** for web UI elements.
- Under this model, for each web page in the application, there should be corresponding page class.
- This Page class will find the WebElements of that web page and also contains Page methods which perform operations on those WebElements.
- Name of these methods should be given as per the task they are performing, i.e., if a loader is waiting for the payment gateway to appear, POM method name can be waitForPaymentScreenDisplay().

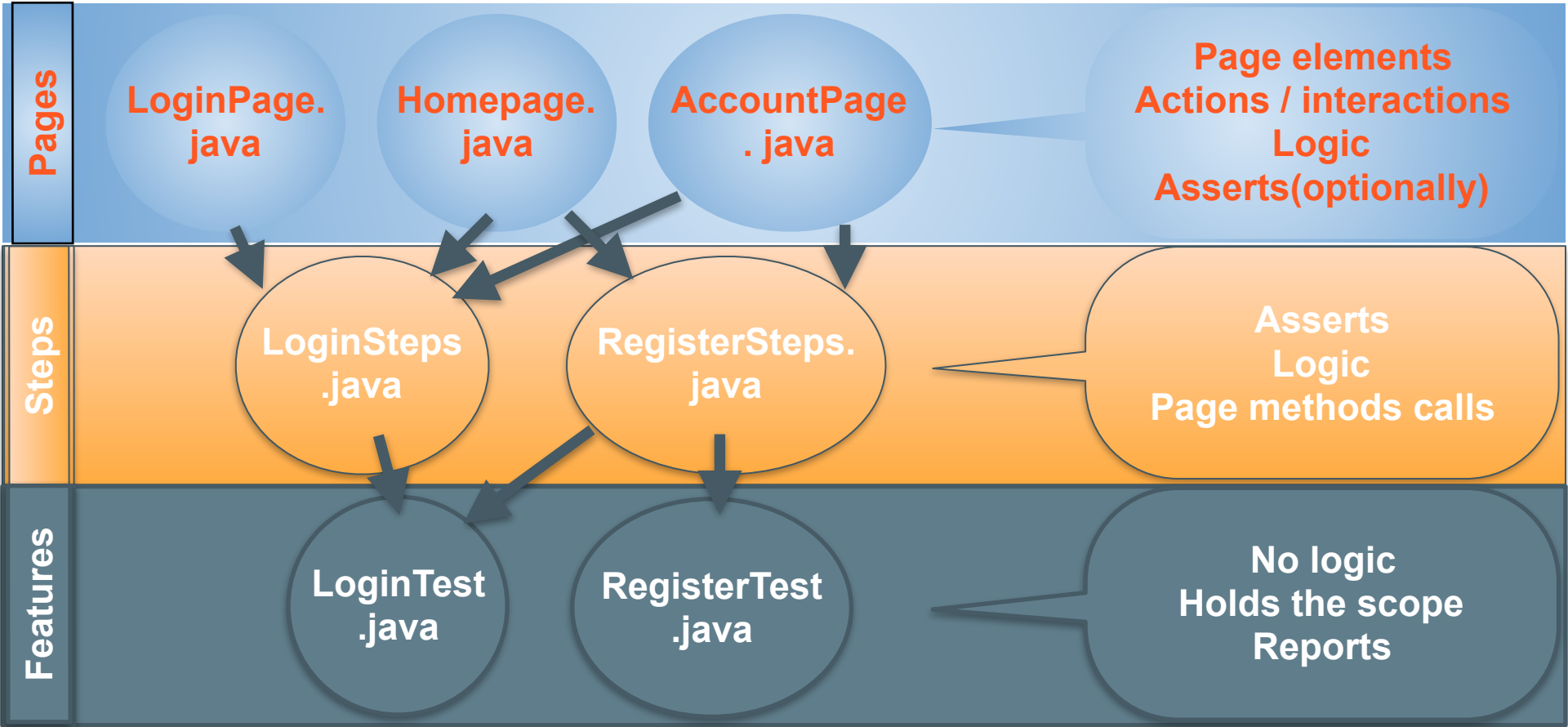
POM and Serenity?



- **Serenity** builds on the **POM** structure
- It adds a new layer between the classic **PAGE - TEST**
- Between these two, Serenity adds a new class, called **STEPS**
- The role of the **STEPS** class is to combine specific methods from the **PAGE** class, to create more complex functions

evozon

Pages - Steps - Tests



Demo project on GIT

https://github.com/cosminevo/TrelloDemoApp

- Read the README.md file
- In order to run the tests you need to have an account, and generate the KEY and TOKEN for that user

Thank you!