Building a Simple Automation Framework (Beginner Guide)

What is an Automation Framework?

An automation framework is a structured set of guidelines that help in creating and designing test cases efficiently. It improves test efficiency, reusability, and scalability.

Types of Automation Frameworks (Overview)

1. Linear Scripting
2. Modular Testing Framework
3. Data-Driven Framework
4. Keyword-Driven Framework
5. Hybrid Framework

Tools Used in This Guide

- Selenium WebDriver
- Java
- TestNG
- Maven (for project management)
- IDE: IntelliJ IDEA or Eclipse

Project Structure

Simple Maven project structure:

```
project-root/
|
 -- src/test/java
```

Building a Simple Automation Framework (Beginner Guide)

| -- tests

| -- pages

|

-- testng.xml

-- pom.xml

Step-by-Step Guide

1. Create a Maven project

2. Add dependencies (Selenium, TestNG) in pom.xml

3. Create Page classes for UI elements

4. Create Test classes for test logic

5. Add testng.xml to define test suite

Sample Code Snippets

Sample Page (LoginPage.java):

```
public class LoginPage {
  WebDriver driver;
  By username = By.id("user");
  By password = By.id("pass");
  By loginBtn = By.id("login");

  public void login(String user, String pass) {
    driver.findElement(username).sendKeys(user);
```

```
    driver.findElement(password).sendKeys(pass);
    driver.findElement(loginBtn).click();
  }
}
```

Sample Test (LoginTest.java):

```
@Test
public void validLoginTest() {
  LoginPage lp = new LoginPage(driver);
  lp.login("admin", "password123");
}
```

Running the Tests

Use testng.xml or run from the IDE using right-click Run as TestNG Test.

Generating Basic HTML Report

TestNG automatically generates an HTML report in the 'test-output' folder after running tests.

Best Practices

- Keep locators in Page classes
- Follow naming conventions
- Reuse code via utility methods
- Use assertions for validations

Building a Simple Automation Framework (Beginner Guide)

- Keep your framework modular and scalable