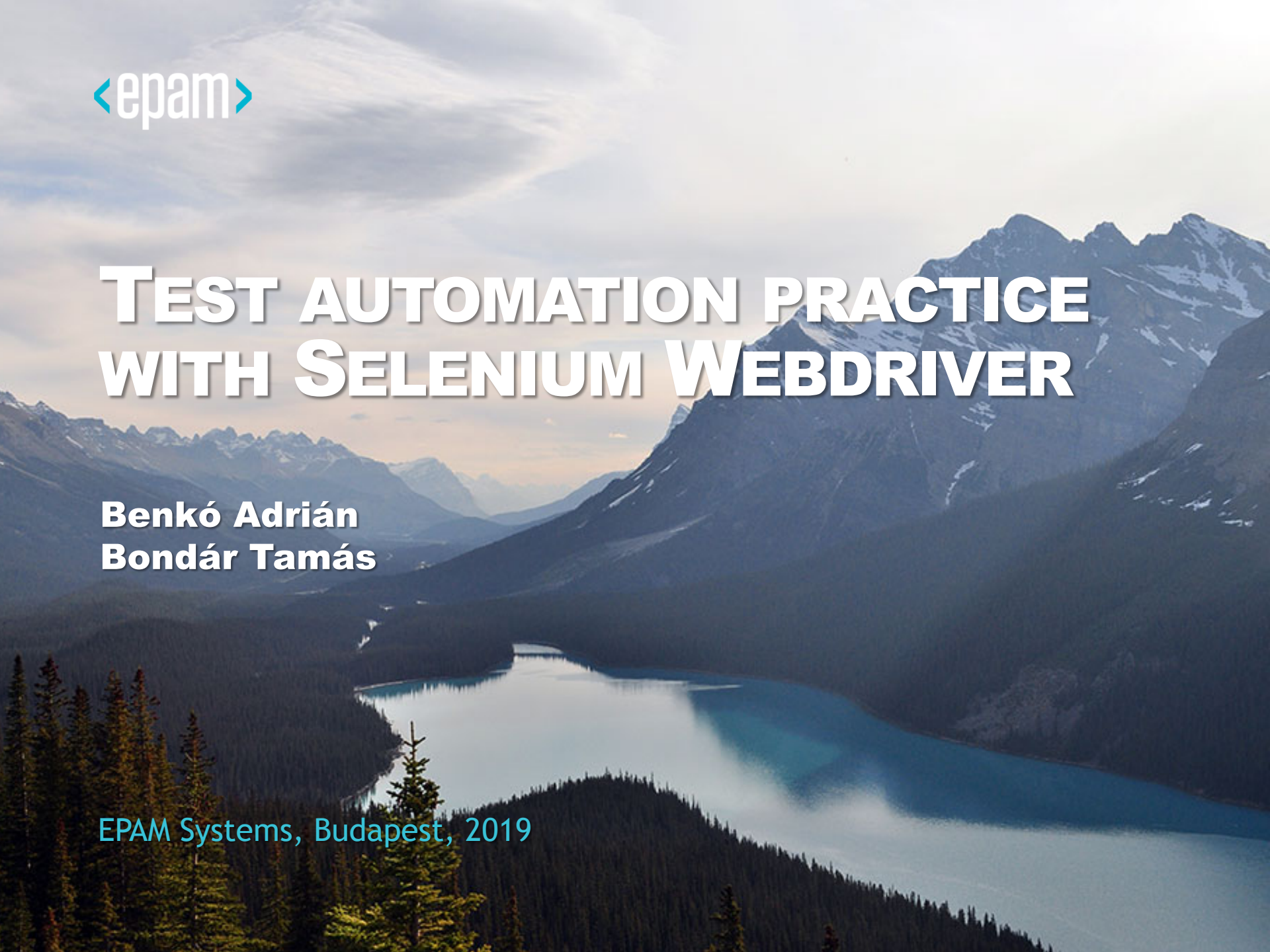


Test automation practice with Selenium Webdriver

Benkó Adrián
Bondár Tamás

EPAM Systems, Budapest, 2019

Homework

https://sites.google.com/site/elte2019oszta/

- Send the ZIP file to the tutor
- **Deadline**: usually 2 weeks after the topic is finished

Advanced Agenda

1. Synchronization
2. Cookies
3. Data Driven Testing

CHAPTER 1

SYNCHRONIZATION

Synchronization

Page Load Timeout

- Sets the amount of time to wait for a page load to complete
- Global setting of the Webdriver object
- Negative value means indefinite wait time

Example

- `driver.manage().timeouts(). pageLoadTimeout(30, TimeUnit.SECONDS);`

Synchronization

Implicit Wait

- Specifies the waiting time for element not immediately visible
- Global setting of the Webdriver object
- 0 by default

Example

- `driver.manage().timeouts().implicitlyWait(5, TimeUnit.SECONDS);`

Synchronization

Explicit Wait

- Waiting for a certain condition
- Poor alternative
  - Thread.sleep(1000);
- Recommended
  - WebDriverWait class

Example

- `WebDriverWait wait = new WebDriverWait(driver, TIME_OUT_IN_SECONDS);`
- `wait.until(ExpectedConditions.method);`

Synchronization

ExpectedConditions class

- `presenceOfElementLocated(By locator)`
- `textToBePresentInElement(WebElement element, java.lang.String text)`
- `titleContains(java.lang.String title)`
- `visibilityOf(WebElement element)`
- `invisibilityOfElementLocated(By locator)`
- `elementToBeSelected(WebElement element)`
- `elementToBeClickable(By locator)`

Click here for more ExpectedConditions

Synchronization

Fluent Wait

- Provides more control than ExplicitWait:
  - Specify polling time
  - Ignore exception classes

Example

```
Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)
    .withTimeout(5, TimeUnit.SECONDS).pollingEvery(1, TimeUnit.SECONDS)
    .ignoring(NoSuchElementException.class);
wait.until(ExpectedConditions.alertIsPresent());
```

CHAPTER 2

COOKIES

Cookies

Introduction

- Useful for testing the login feature
- Getting or setting session IDs
- Cookie attributes
  - Name
  - Value
  - Domain
  - Path
  - Expiry
  - Secure
  - Http only

Cookies

Interrogation

- Get all cookies from the current session
  - `driver.manage().getCookies();`
- Get cookie with a given name
  - `driver.manage().getCookieNamed(cookieToTest);`

Cookies

Manipulation

- Delete all cookies from the current session
  - `driver.manage().deleteAllCookies()`
- Delete a specific cookie
  - `driver.manage().deleteCookie(TestCookie);`
- Delete cookie with a given name
  - `driver.manage().deleteCookieNamed(cookieToTest);`

Cookies

Manipulation

- Add a specific cookie
  - `Cookie cookie = new Cookie("mycookie", "123456");`
  - `driver.manage().addCookie(cookie);`
- Domain attribute is the current document by default

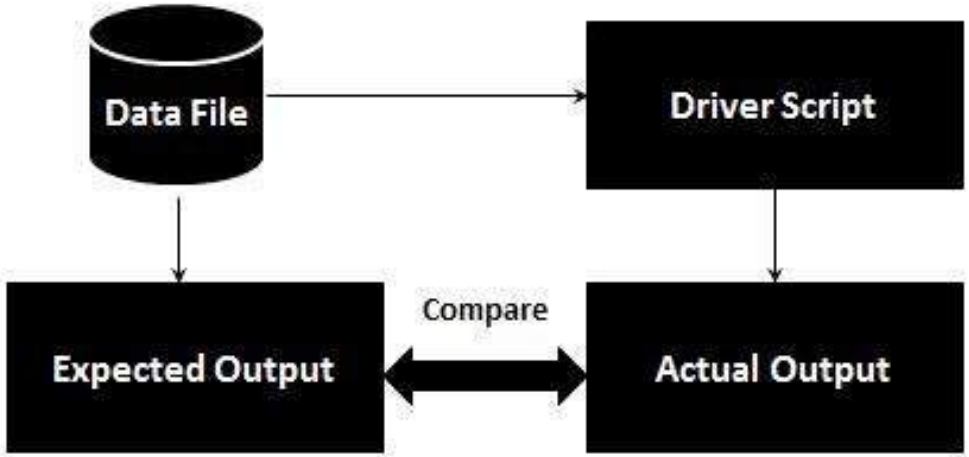
CHAPTER 3

DATA DRIVEN TESTING

Data Driven Testing

Concept

- Use pre-stored data as input and expected output
- Run your script to get actual output and compare them
- Continue testing with the next set of data



Data Driven Testing

Possible data sources

- Database
- XML file
- Property file
- Etc.

Pros of Data Driven Testing

- Repeatability and reusability
- Separation of test code and data
- Reduction in number of tests

Data Driven Testing

Where to use

- Testing different localizations of a site
  - `<localizatioinData lang="en" submit=„Timetable" />`
  - `<localizationData lang="hu" submit=„Menetrend" />`

Data Driven Testing

How to use

- Use JUnitParamsRunner class
  - `@RunWith(JUnitParamsRunner.class)`
  - `public class Testclass { … }`
- Add test parameters method #1
  - `@Parameters(method = „testDataProviderMethod")`
  - `public void testCase(String param1, String param2) { … }`

Data Driven Testing

How to use

- Add test parameters method #2

```
– @FileParameters("path/to/file")

– public void testCase(String param1, String param2)

  { … }
```

Questions