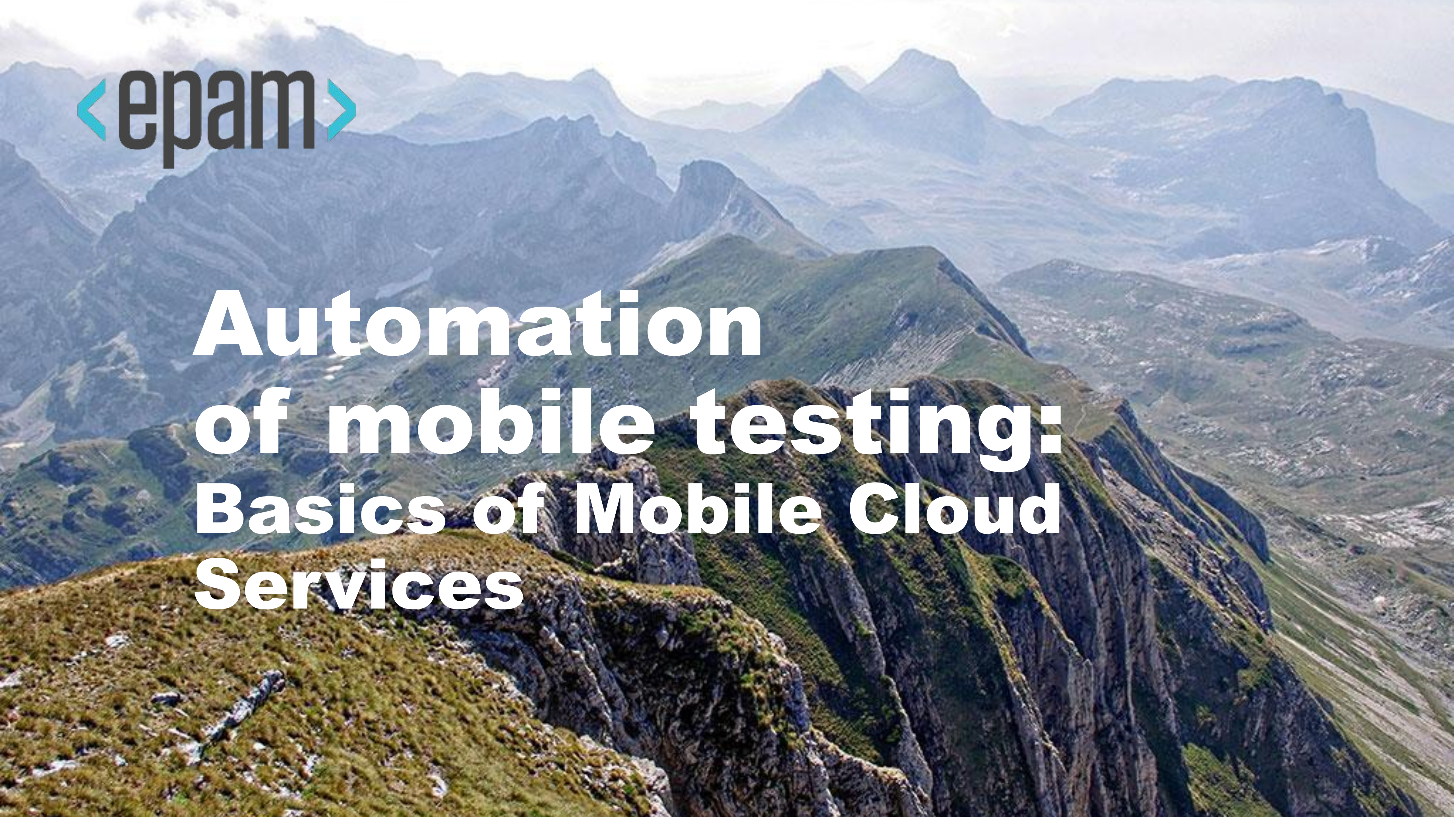


Automation of mobile testing: Basics of Mobile Cloud Services

Plan

- Basic tools
- How to write auto-tests for mobile apps
- Basics of Mobile Cloud Services
- Tips and tricks

форм фактор

Mobile Clouds

Cloud services of mobile devices (mobile farms) are the modern approach.

They provide developers and testers with remote access to sets of physical devices *for fixed prices*
and some tools for using them

Remote access to set of emulators can be provided as well for less prices

Advantages

Utilize advantages, avoid weaknesses
(of real devices and emulators):

- Doesn’t take space on your desk
- No power needs
- A range of devices of different capabilities
- Do not need to take care about devices
- Pay for access only, no Total Cost of Ownership

Problems

Not a fairy tale:

- the set of models is defined by provider of mobile cloud
- availability of items
- communications could be a problem:
    - connection could be lost
    - devices react with a delay

Examples

Public mobile test clouds: SauceLabs, Browserstack

EPAM own mobile cloud solution (now located in Minsk and SPb)

- Free access for EPAM employees by their credentials
- Sites of this cloud can be located at any of EPAM offices

EPAM mobile cloud

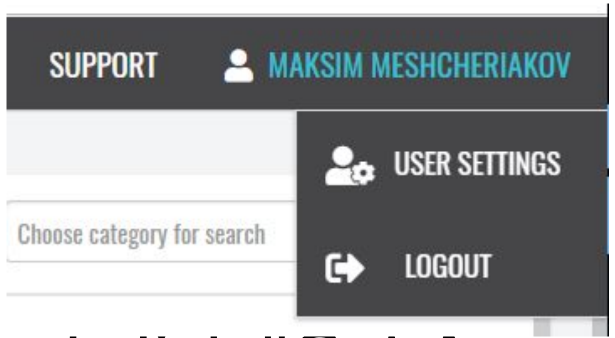
1. Located here
2. Access: by EPAM account for all employee
3. Features:
   a. ***Physical*** Android and iOS devices (up to 25 of each type)
   b. Inspector.
   c. Use your desktop Chrome for debugging remote devices
   d. Appium support
   e. CI integration
   f. Some device can be reserved for a certain time
   g. REST interface for booking and application setup
   h. How to start working?

Modify our tests for m-cloud

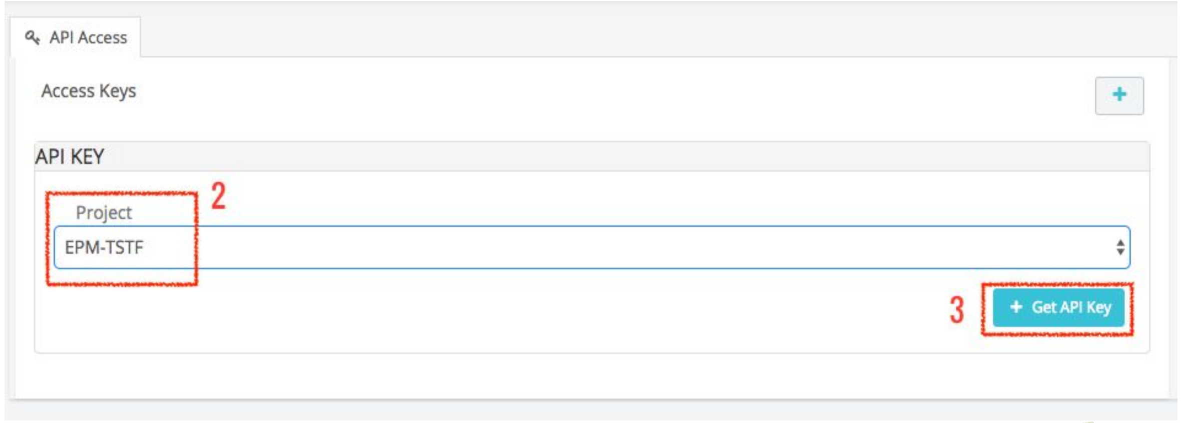
1. Install security certificates to Java SDK
2. Get a token and integrate it to Appium driver URL
3. Check driver type and change it if required
4. Select a certain device (Android or iOS?)
5. Build, run, …
6. Enjoy?????

Get a token

This token gives you access to REST API and Appium Automation



1. Open your account -> User Setting
2. Click " + ", select EPM-TSTF Project and click "Get Access Token“. If you has an expired key, you need remove it before (use red REMOVE link at the right)
3. Write down the key value and keep it safe
4. This access token is used for both REST API and Appium connection
5. TTL of this token is 1 month



Change Appium server URL

```
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-surefire-plugin</artifactId>
            <version>2.22.2</version>
            <configuration>
                <suiteXmlFiles>
                    <suiteXmlFile>${suite.file}</suiteXmlFile>
                </suiteXmlFiles>
                <systemPropertyVariables>
                    <!--ts.appium>http://localhost:4723/wd/hub</ts.appium-->
                    <ts.appium>https://EPM-TSTF:${token}@mobilecloud.epam.com/wd/hub</ts.appium>
                </systemPropertyVariables>
            </configuration>
        </plugin>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <configuration>
                <source>1.8</source>
                <target>1.8</target>
            </configuration>
        </plugin>
    </plugins>
</build>

<properties>
    <token>de669071-644c-4fdb-9fad-293f8decc3d2</token>
</properties>
```

Cloud Web test run

Dedicated Maven profile

```xml
<profile>
    <id>cloudWeb</id>
    <activation>
        <activeByDefault>false</activeByDefault>
    </activation>
    <properties>
        <suite.file>./src/test/resources/cloudWebTNG.xml</suite.file>
    </properties>
</profile>
```

Dedicated TestNG .xml

```xml
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="EPAM mobile cloud Web testing Suite">
    <parameter name="appType" value="web"></parameter>
    <!--parameter name="deviceName" value="SAMSUNG SM-G960F"></parameter-->
    <!--parameter name="udid" value="802KPFX1550899"></parameter-->
    <parameter name="platformName" value="Android"></parameter>
    <parameter name="browserName" value="Chrome"></parameter>
    <test name="Cloud Web tests">
        <groups>
            <run>
                <include name="web"/>
            </run>
        </groups>
        <packages>
            <package name="scenarios"/>
        </packages>
    </test>
</suite>
```

Modify Appium driver

1. “Agile way”:
   simple proto => check => fix obvious => commit
2. Modify parameters instead source code (sic!):
   sut=www.iana.org
   platform=Android
   devicename=SAMSUNG SM-J320F
   driver=https://EPM-TSTF:6a3e096f-0ad9-4cf4-bdff-7016e684ed05
   @mobilecloud.epam.com/wd/hub
3. Sorry, but we have to modify source code slightly…

App accessibility

“app” capability not useful more

- Use instead:
  - Android: appPackage & appActivity
  - iOS: bundleId
- Install app to target device
  - Manually via cloud UI
  - Automatically via cloud API

Device identification alternatives

- You may need to use *UDID* instead of *deviceName*. It depends on current implementation of mobile cloud software
- For EPAM mobile cloud solution you can use *Serial Number* as a UDID
- *DeviceName* = Manufacturer + Product

Source code example

Android native app

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Native mobile testing Suite for EPAM Mobile Cloud">
    <parameter name="appType" value="native"></parameter>
    <parameter name="deviceName" value="GOOGLE Pixel 3 XL"></parameter>
    <parameter name="platformName" value="Android"></parameter>
    <parameter name="appPackage"
value="platkovsky.alexey.epamtestapp"></parameter>
    <parameter name="appActivity"
value="activities.LoginActivity"></parameter>
    <test name="Cloud Native tests">
        <groups>
            <run>
                <include name="native"/>
            </run>
        </groups>
        <packages>
            <package name="scenarios"/>
        </packages>
    </test>
</suite>
```

iOS native app

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="iOS Native mobile testing Suite for EPAM Mobile Cloud">
    <parameter name="appType" value="native"></parameter>
    <parameter name="udid" value="00008020-001429081A43002E"></parameter>
    <parameter name="platformName" value="iOS"></parameter>
    <parameter name="bundleId" value="com.epam.EPAMTestApp.ii"></parameter>
<test name="iOS Native tests">
        <groups>
            <run>
                <include name="native"/>
            </run>
        </groups>
        <packages>
            <package name="scenarios"/>
        </packages>
    </test>
</suite>
```

App processing

```
@Parameters({"platformName", "appType", "deviceName", "udid", "browserName","app","appPackage","appActivity","bundleId"})
@BeforeSuite(alwaysRun = true)
public void setUp(String platformName,
                  String appType,
                  @Optional("") String deviceName,
                  @Optional("") String udid,
                  @Optional("") String browserName,
                  @Optional("") String app,
                  @Optional("") String appPackage,
                  @Optional("") String appActivity,
                  @Optional("") String bundleId
) throws Exception {
    System.out.println("Before: app type - "+appType);
    setAppiumDriver(platformName, deviceName, udid, browserName, app, appPackage, appActivity, bundleId);
    setPageObject(appType, appiumDriver);

}
```

App processing

```
private void setAppiumDriver(String platformName, String deviceName, String udid, String browserName,
                             String app, String appPackage, String appActivity, String bundleId){
    DesiredCapabilities capabilities = new DesiredCapabilities();
    // mandatory Android capabilities
    capabilities.setCapability("platformName",platformName);
    capabilities.setCapability("deviceName", deviceName);
    capabilities.setCapability("udid", udid);

    if(app.endsWith(".apk")) capabilities.setCapability("app", (new File(app)).getAbsolutePath());

    capabilities.setCapability("browserName", browserName);
    capabilities.setCapability("chromedriverDisableBuildCheck","true");

    // Capabilities for test of Android native app on EPAM Mobile Cloud
    capabilities.setCapability("appPackage",appPackage);
    capabilities.setCapability("appActivity",appActivity);

    // Capabilities for test of iOS native app on EPAM Mobile Cloud
    capabilities.setCapability("bundleId",bundleId);
    //if(platformName.equals("iOS")) capabilities.setCapability("automationName","XCUITest");


    try {
        appiumDriver = new AppiumDriver(new URL(System.getProperty("ts.appium")), capabilities);
    } catch (MalformedURLException e) {
        e.printStackTrace();
    }

    // Timeouts tuning
    appiumDriver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

}
```

Build and run!

- Pray or enjoy?
- Try existent (web) test on iOS w/o changes in code.

Can we hope for cross-platform auto tests?

Cross-platform troubles

Why it may not work for iOS native apps?

- No the same iOS app
- Different GUI design & guides
- Different libs
- Different target audience for Android and iOS platforms/apps
- In particular: locators strategy for iOS differs from Android one (xpath mostly)

But methodology itself is cross-platform

Cross-platform PO

```
public class NativePageObject  {

    @AndroidFindBy(id = "platkovsky.alexey.epamtestapp:id/email_sign_in_button")
    @iOSXCUITFindBy(xpath = "//XCUIElementTypeStaticText[@label='Sign In']")
    WebElement signInBtn;

    public NativePageObject(AppiumDriver appiumDriver) {
        PageFactory.initElements( new AppiumFieldDecorator(appiumDriver), this);
    }

}
```

Plan

- Basic tools
- How to write auto-tests for mobile apps
- Basics of Mobile Cloud Services
- Tips and tricks

QUESTIONS

maxim.mescheryakov
maksim_meshcheriakov@epam.com