Legacy | Using Secure Strings in Selenium/Appium

Introduction

When applications require certain levels of secure input over the network, for example when sending a user's password to the bank server, the communication between the application (client) and the service provider (server) will usually employ some level of encryption for the information passed between these two entities. Similarly, when automating an application over the network, for example as performed when using Selenium/Appium in the Perfecto CQ Lab, there may be a requirement to only transfer encrypted strings between the automation script and the device at the other side of the network divide. This requirement may apply to certain sensitive input strings provided by the automation script, such as user names and passwords.

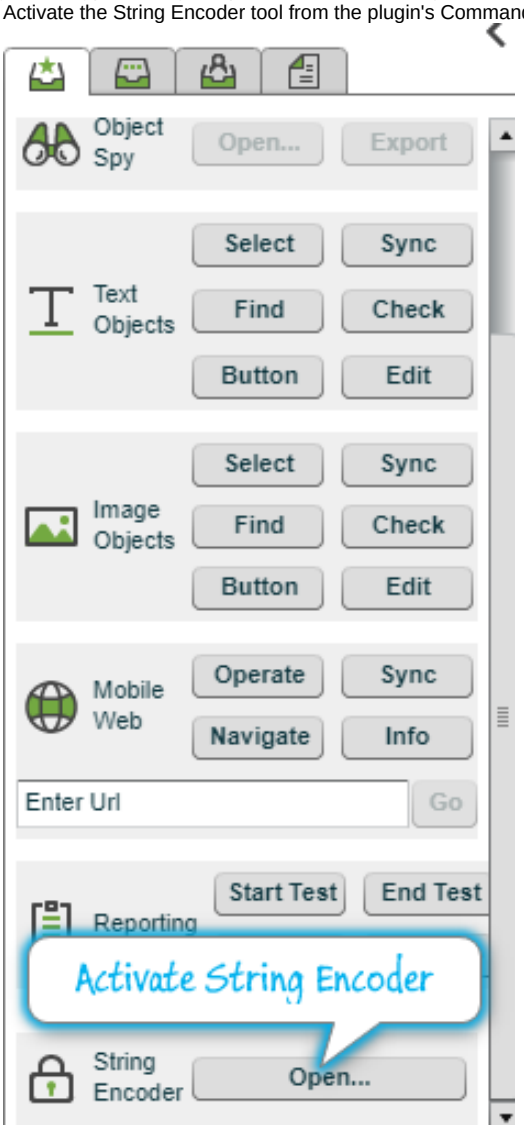
Legacy | Perfecto's String Encoder

Perfecto is very aware of the requirement to transfer encrypted text from the automation scripts to the controlled devices. Therefore, both the IDE and Selenium/Appium plugins provide a String Encoder tool. The tool allows the tester to generate encrypted string versions of the input strings, that can then be transmitted by the automation script to the application on the device, using the ***sendKeys()*** method of the UI Element.

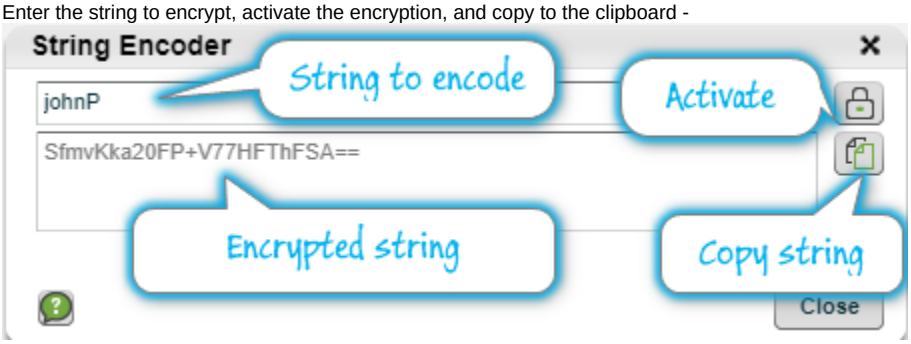
Legacy | Incorporating Secure Strings into Selenium/Appium Scripts

As a best practice:

1. Prepare your encrypted strings as private static final String variables -
    a. Activate the String Encoder tool from the plugin's Command Sidebar at the bottom



b. Enter the string to encrypt, activate the encryption, and copy to the clipboard -



c. Paste the string as the value of a static variable.
Important Note: Prefix the string with "**secured.**", this indicates to the Selenium server that the string in encrypted and the server will decrypt the string prior to transmitting it to the UI Text Field on device.

```
private static final String Secured_uname = "secured.SfmvKka20FP+V77HFThFSA==";
private static final String Secured_pw = "secured.cJiPYMXBxIx4WogfiHIjQg==";
```

2. Use these String variables as the strings that are transmitted as part of the automation.

a. When using the secured string with the *PerfectoMobile* value of the ***automationName*** capability

```
driver.findElementByName("username").sendKeys(Secured_uname);
driver.findElementByName("password").sendKeys(Secured_pw);
```

b. When using the secured string with the *Appium* or *XCUITest* value of the ***automationName*** capability

```
//declare the Map for script parameters
Map<String, Object> params = new HashMap<>();

params.put("text", Secured_uname);
params.put("by", "Name");
params.put("value", "username");
driver.executeScript("mobile:application.element:set", params);
```

c. When using a secured string for Desktop Web Selenium testing:

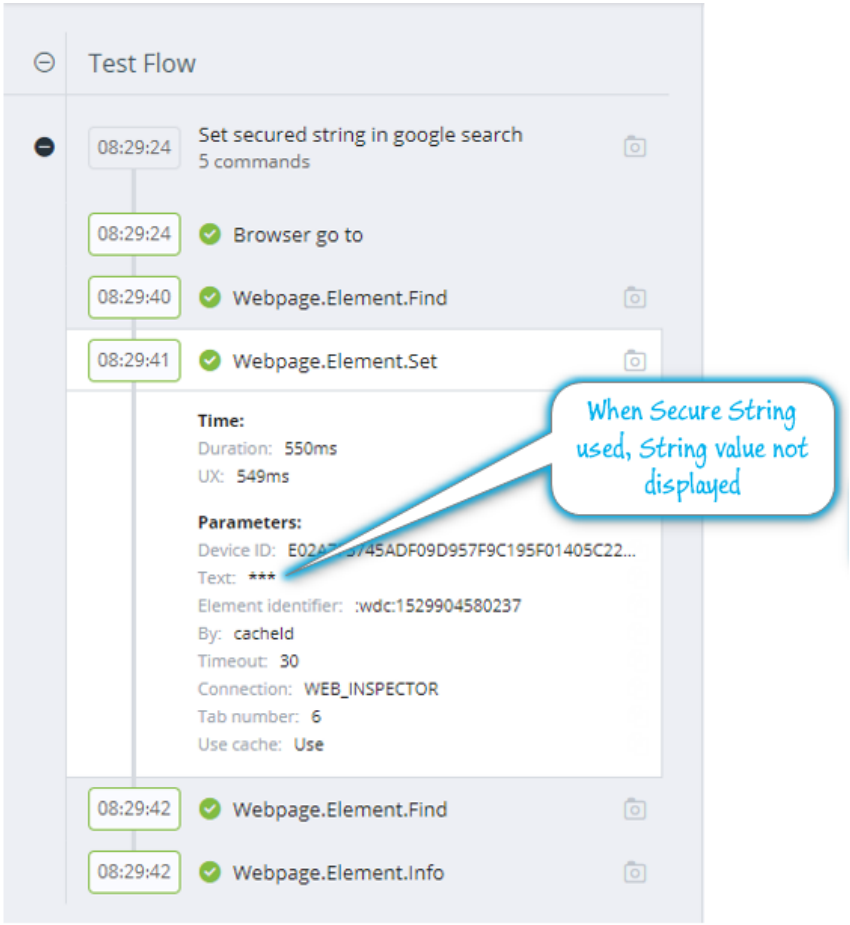
```
//declare the Map for script parameters
Map<String, Object> params = new HashMap<>();

params.put("value", Secured_uname);
params.put("label", "username");
driver.executeScript("mobile:edit-text:set", params);
```

3. Wrap the code within a Reporting Step to quickly identify the execution when looking at your automation's test report.
The complete snippet (based on option 2a above) would look something like this:

```
// encoded strings:
private static final String Secured_uname = "secured.SfmvKka20FP+V77HFThFSA==";
private static final String Secured_pw = "secured.cJiPYMXBxIx4WogfiHIjQg==";
...
    reportiumClient.stepStart("Send secured strings as username and password fields");
    driver.get("myApp URL");
    driver.findElementByName("username").sendKeys(Secured_uname);
    driver.findElementByName("password").sendKeys(Secured_pw);
    driver.findElementByName("loginBtn").click();
    reportiumClient.stepEnd();
    ...
```

Secure Strings in the Test Report

When viewing the Single Test Report (STR) of the automation execution any string values sent to the device as a Secured String will be displayed as "***":


Test Flow
08:29:24
Set secured string in google search
5 commands
08:29:24
Browser go to
08:29:40
Webpage.Element.Find
08:29:41
Webpage.Element.Set
Time:
Duration: 550ms
UX: 549ms
Parameters:
Device ID: E024
3745ADF09D957F9C195F01405C22...
Text: ***
Element identifier: :wdc:1529904580237
By: cacheId
Timeout: 30
Connection: WEB_INSPECTOR
Tab number: 6
Use cache: Use
When Secure String used, String value not displayed
08:29:42
Webpage.Element.Find
08:29:42
Webpage.Element.Info