Testing Progressive Web App on iOS

By Guy Heimann

Background

Progressive Web Apps (PWA) are creating waves in the mobile application community. Allowing a web application to act like a mobile application allows the user to know that his application is in-sync with the service provider's web-site services. Apple's Safari browser has introduced, in iOS 11.3, support for configuring any website, whose creator defined the ability to configure the site as a PWA, to install the corresponding PWA on the home page of your iOS device.

Perfecto Lab provides the tools to launch, test, and close the PWA on the iOS device. Whether you use Appium or Perfecto's Native Automation you can test the PWA as if it were a hybrid application without performing any instrumentation.

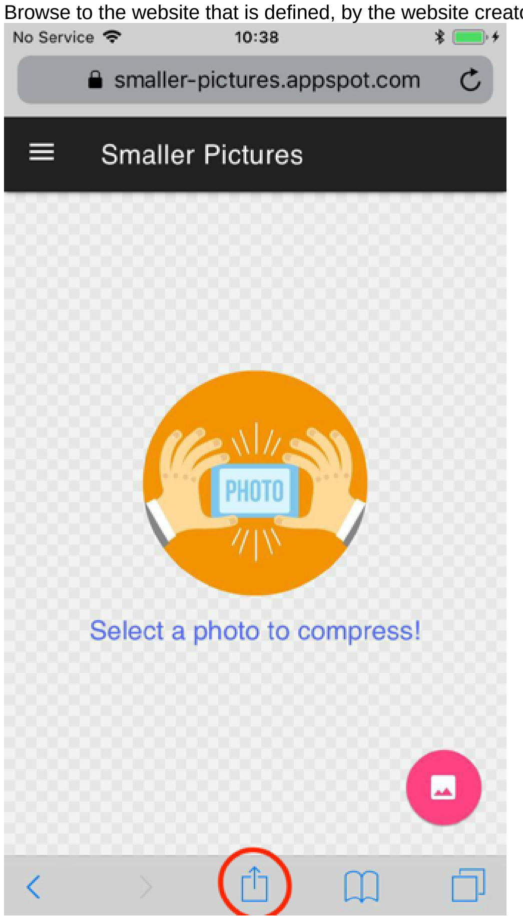
This article does not address how to create the PWA, but, rather with how one works on iOS and how you can automate it using the Perfecto Lab.

Safari Support for *Installing* PWA

When using the Safari browser, it’s possible to generate a special shortcut to the current open page, using the *share tool* that Safari supplies in the status line of the display (see below):

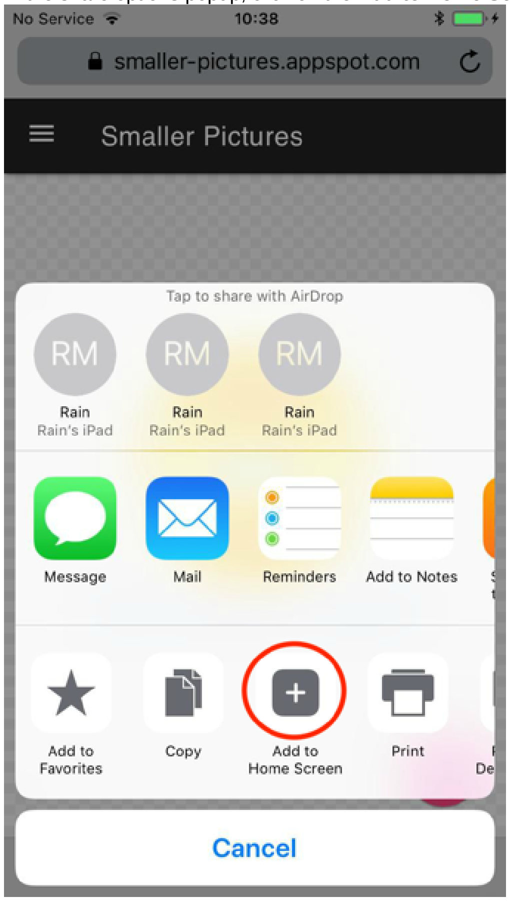
To *install* the PWA:

1. Browse to the website that is defined, by the website creator, to be PWA-ready:

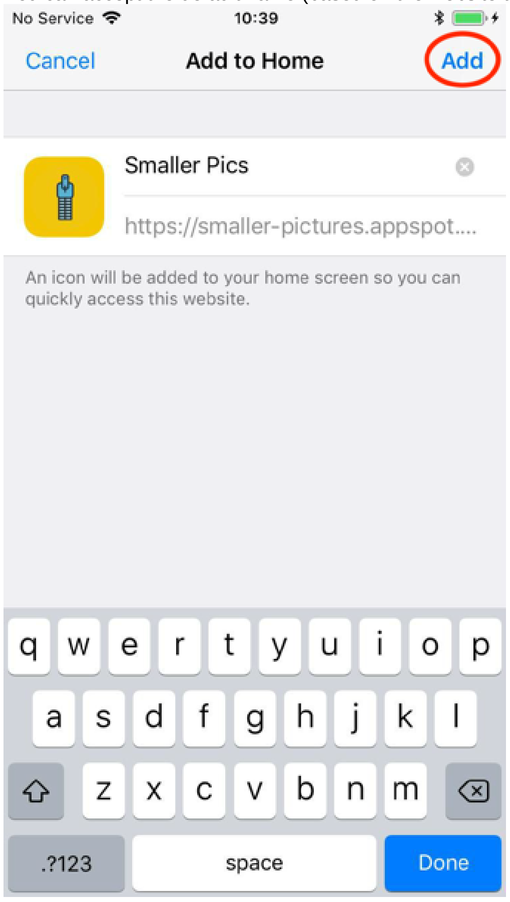


2. Click on the share icon (red circle, above).

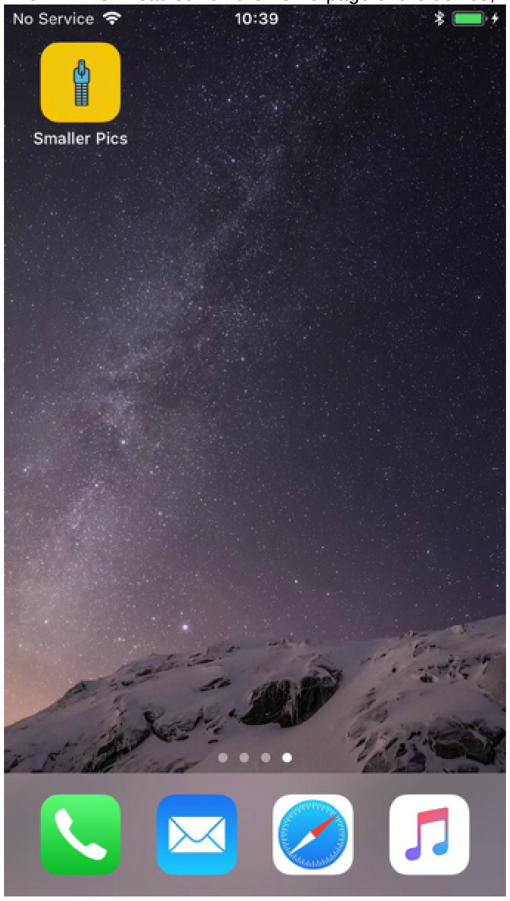
3. In the share options popup, click on the **Add to Home Screen** button.



4. You can accept the default name (based on the website title) or supply a new name. Click **Add**.


No Service
10:39
Cancel
Add to Home
Add
Smaller Pics
https://smaller-pictures.appspot....
An icon will be added to your home screen so you can quickly access this website.
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
.?123
space
Done

5. The PWA is "installed" on the home-page of the device, linked to the URL of the website, and ready to be activated.

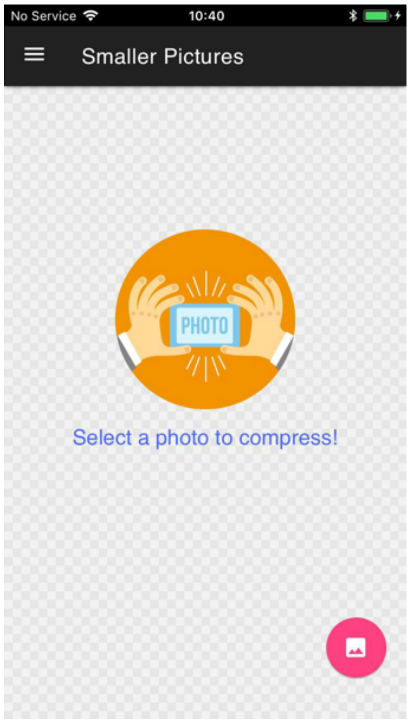


How it works

If the web developer configured the site to allow it to work as a PWA, once the PWA is launched from the shortcut- the URL will be open without the Safari native part. If the site is not configured to work as PWA, Safari will be launched with the URL (same as regular work with Safari web pages).

When PWA is launched, it behaves like an independent app. Safari isn’t involved in the process and it looks to the user just like a new native iOS application, without the Safari wrapper - meaning no URL text-box at top and no status line at bottom:

In actuality, when “installing” the PWA, no new application is being added to the device. When clicking on the PWA icon, Apple’s Webapp is launched without Safari. This Webapp (*com.apple.webapp*) is a container that runs a web view, different PWAs will run in different Webapp instances.



Automating the PWA from Appium

As we outlined, the PWA is not coupled to Safari but, on the other hand, it is not a unique application. Therefore, to test it, user must:

- Use the Appium IOSDriver
- Explicitly launch the PWA from inside the script using the ***Start PWA*** command (see below).

Meaning that the application cannot be launched from the Desired Capabilities, like a native application, but should be considered a mobile hybrid application that runs in an iOS environment (without Safari).

Launching the application

Use the Perfecto Extension Command ***mobile:pwa:start*** to launch the PWA with the *displayName* as a parameter.

The display name is the label of the PWA icon on the home page:

```
HashMap<String, String> pwaParams = new HashMap<String, String>();

pwaParams.put("displayName", "Smaller Pics");
driver.executeScript("mobile:pwa:start", pwaParams);
```

Closing the application

Use Perfecto Extension Command ***mobile:pwa:stop*** to close the PWA (no parameters needed):

```
driver.executeScript("mobile:pwa:stop");
```

Note: The Stop PWA command will only work if the PWA is running in the foreground at time of command execution, otherwise the command will fail with “***Cannot close application. Application is not running.***”

Switch Context to Identify UI Elements

The Perfecto Lab considers the PWA as if it were a hybrid application, meaning:

- The website user interface elements are all DOM elements, and should be accessed (via XPath) using the **WEBVIEW** context.

```
driver.context("WEBVIEW");
driver.findElementByXPath("//*[text()='MyButton']").click();
```

- The application wrapper elements (for example, application header), and the XCUIElementTypeWebViewthat encapsulates the website are all native application elements, and should be accessed (via XPath) using the NATIVE_APP context.

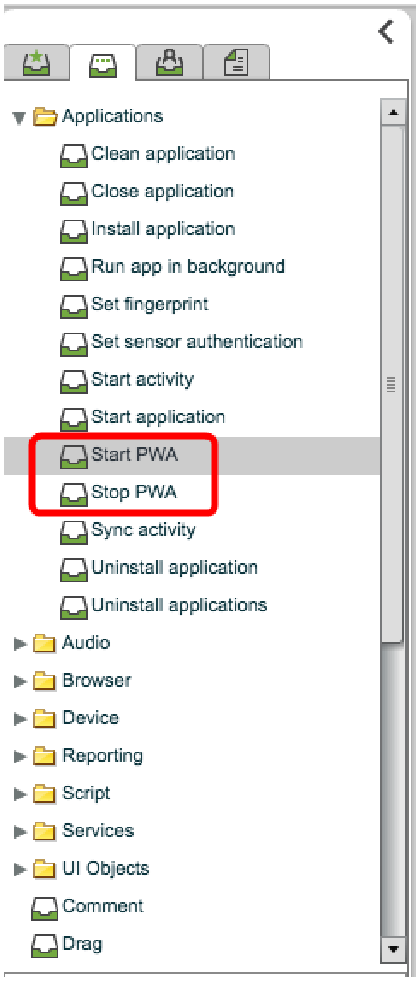
```
driver.context("NATIVE_APP");
RemoteWebElement element = (RemoteWebElement)driver.findElementByXPath("//XCUIElementTypeWebView");
String elementID = element.getId();

// Outer WebView element can be scrolled using the scroll object command
HashMap<String, String> scrollObject = new HashMap<String, String>();
scrollObject.put("element", elementID);
scrollObject.put("direction", "down");
driver.executeScript("mobile:scroll", scrollObject);
```

Legacy | Automating the PWA from Native Automation

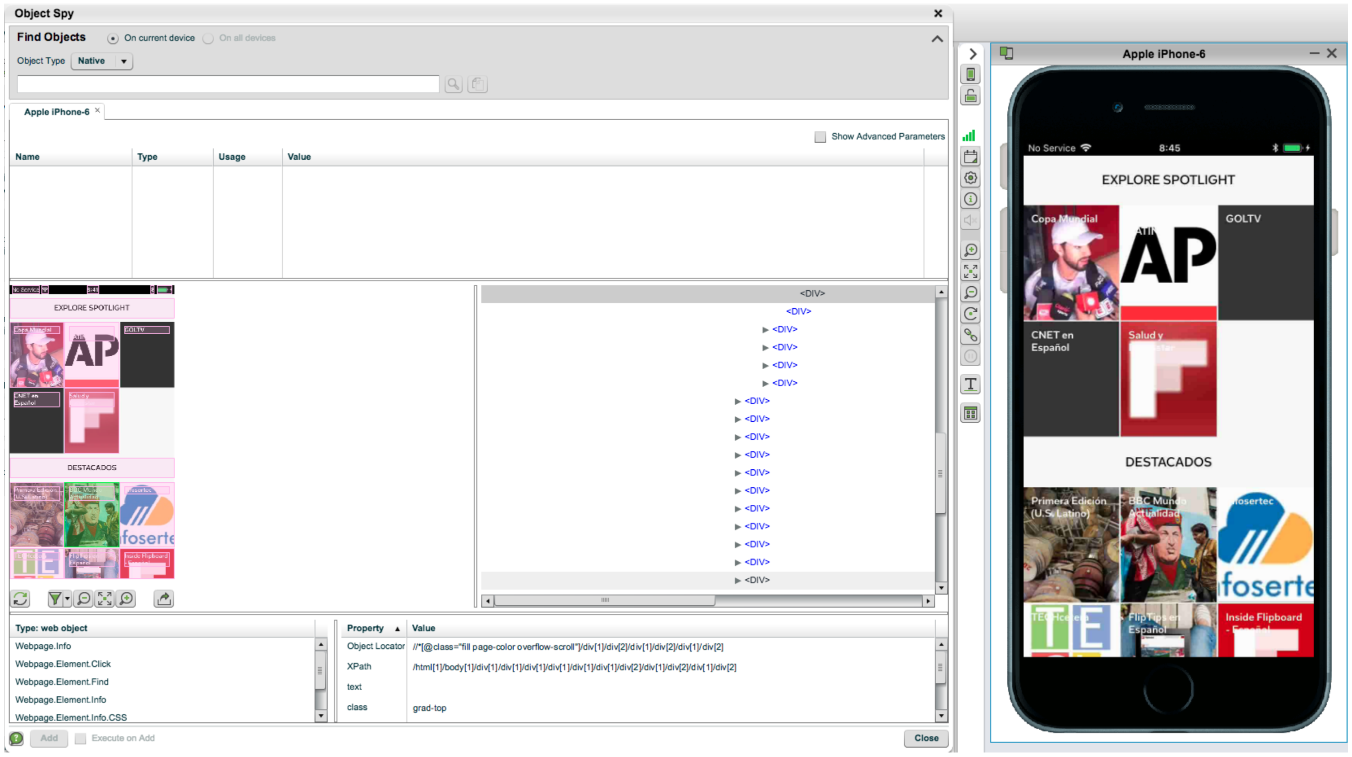
Using the Perfecto Lab IDE to automate the PWA, is possible using the new Applications functions:

- Use the ***Start PWA*** (under "Applications") function to activate the PWA and provide the *display name* of the PWA icon as a parameter.
- Use the ***Stop PWA*** (under "Applications") function to stop the PWA execution, no parameters needed.
  Note: Stop PWA function will only close the PWA if it is running in the foreground, otherwise it will throw an error.



Object Spy

The Object Spy will identify the UI Elements of the PWA and provide accurate placement of the elements to support using the Native Automation functions and commands to automate the application functionality.



Name	Type	Usage	Value

Type: web object

- Webpage.Info
- Webpage.Element.Click
- Webpage.Element.Find
- Webpage.Element.Info
- Webpage.Element.Info.CSS

Property	Value
Object Locator	//*[@class="fill page-color overflow-scroll"]/div[1]/div[2]/div[1]/div[2]/div[1]/div[2]
XPath	/html[1]/body[1]/div[1]/div[1]/div[1]/div[1]/div[1]/div[1]/div[2]/div[1]/div[2]/div[1]/div[2]
text	
class	grad-top

Limitations

The following limitations exist in the current version of the Perfecto support for PWA automation:

- Only applicable for devices running iOS 11.3 or later.
- Not supported for iPad devices in Landscape Mode.
- No instrumentation supported for PWA - this limits the use of the Sensor Instrumentation related functionality of Camera Image Injection, Fingerprint authentication, etc.