**Q1) Explain Appium Vs Calabash?**

|  |  |  |
| --- | --- | --- |
| **Appium Vs Selenium** | | |
| **Features** | **Appium** | **Calabash** |
| Language Support | Python, javascript, ruby, java & all major languages | Only language Ruby is supported |
| iOS mobile web page | We need custom safari app for automation | Not well supported |
| Hybrid iOS app | Custom ui commands & iOS uiautomator | Custom ui commands & iOS framework |
| Native iOS app | Only the iOS uiautomator | iOS instrumentation - calabash iOS |
| Android Mobile web app | Only automates chrome browser | Not well supported |
| Hybrid android app | Only the selendroid app | Android instrumentation - calabash android |
| Native android app | Both android uiautometer&selendroid | Android instrumentation - calabash android |
| Internal tools | Android, iOS uiautometers&selendroid | Android & iOS instrumentation framework |

**Q2) What are native Apps?Q: What is Appium?**

An open source tool that is required for mobile web, automating Native and hybrid application on Android and IOS platform is known as Appium which was in 2012. Appium is considered to be a cross platform that will low you to write tests which are on multiple platforms like Android and IOS. They do this using the same API. This facility will enable you to do reuse of codes between Android and IOS test sites.

Those Apps are written by using Android SDKs and IOS are known as Native Apps.

.

**Q5) Name the language that is used to write tests in Appium?**

The tests of Appium are written is any language and this is because appium is nothing but a HTTP server. It is also important that the test should be interfaced with Appium and it use HTTP libraries so that they can create HTTP sessions.  
In order to create the right commands in Appium then all you need to know Selenium protocol.   
The most sought after development frame works are .Net, C#, Java, Python, JavaScript and Ruby.

**Q6) Which are the pre- requisites to use Appium?**

The pre-requisites that are used in Appium. They are listed below.

1. Eclipse IDE  
2. Android SDK  
3. TestNG  
4. Web driver language binding library  
5. JS  
6. JDK  
7. APK App Info on Google play  
8. Selenium server jar  
9. Appium for Windows

**Q7) Write the advantages of using of Appium.**

The advantages of Appium are listed below:-

1. Using the same API, Appium will allow you to write tests that are against mobile platforms.  
2. By using any kind of test frame work or language you can write and run the tests.  
3. Appium is an open source platform so you can contribute to it easily.  
4. For the hybrid mobile applications and Native, Appium provides cross platform.  
5. Appium supports JSON wire protocol.   
6. Appium do not require recompilation of App.  
7. Appium also supports automation test on the physical devices and also for simulator or emulator both.  
8. Appium does not have any dependency on mobile devices.

**Q8) Name the test frame works that are supported by Appium.**

Test frame works are not supported by appium since there is no need to do it. All test frame works can be used by Appium. Some examples are .net unit test and NUnit. A test for Appium is written using one of the drivers so that the tests can interface with the appium in case of external dependency.  
     
**Q9) Write down the disadvantages of Appium?**

The disadvantages of Appium are listed below:-

1. The testing of those android that are lower than 4.2 is not allowed.   
2. Appium has limited support for hybrid app testing. You will not be able to test the action that allows switching of applications from native to web app and from web app to native.    
3. There is no support that will allow you to run Appium inspector on Microsoft windows.

**Q10) What are the requirements to write Appium tests?**

There are certain basic requirements when it comes to write Appium tests and they are:-

**1. Driver client**– Mobile applications are driven by Appium like that of a user. With the help of a client library Appium tests can be written and these will wrap the steps of a test and then send it Appium over the HTTP.  
**2. Appium Session** – Appium tests takes place within a session so it is important to initialize an appium session first. Once there is an end to the automation of a session it will be ended and wait again for the next session.  
**3. The desired capabilities** – In order to initialize an appium session it is very important to design some parameters which are known as desired parameters. These parameters are platform version, platform name, device name and many more. This also helps in specifying the type of automation that is required from the Appium server.   
**4. Driver Command**– In Appium you have the facility to write the tests by using a big and expressive collection of commands.

**Q11) Name the Appium inspector.**

Just like a selenium IDE playback and record tool, Appium consist of an inspector that is used to record and playback. With the help of this, you can record and play native application behaviour which is achieved by inspecting DOM. It helps in generating the test scripts any language that is preferred. But Appium Inspector is not a good support for Windows and they use UIAutomator viewer in the option.

**Q12) Suppose you are testing App, how will the data be exchanged between your test and the App?**

In Appium the Web driver specification is not made for the purpose of exchanging data with the app. But it is not totally impossible to exchange data. It is achievable and it will require you to build more layers of testability.

**Q14) How can you find the DOM element or X path in case of a mobile application?**

In order to find the path between DOM elements or X path elements you can make use of "UIAutomateviewer" in case of Android application.

**Q15) What are the pros and cons of Appium?**

**Pros-**

Irrespective of the platform, programmer will be able to automate all the complexities which remain under a single Appium server. Also Appium helps in providing a cross platform mobile testing which signifies that the same test pattern will work on multiple platforms.   
With the help of Appium, there will be no need for extra components in the app to make it more users friendly. Appium can also automate hybrid, web and native mobile applications.

**Cons -**

It is not possible to run scripts on multiple IOS simulator simultaneously. With the help of Appium you are able to use UIAutomator in case of Android automation which only supports Android SDK platform and those API which are 16 more than that. For using older version of APIs you will have to use another open source library which is known as Selendroid.

**Q17) Is there a need for a server machine to run tests on Appium?**

In order to run tests on Appium there is no need of server machine. 2-tier architecture is facilitated with Appium. It is in this 2-tier architecture that the test machine gets connected to a test server which is running on Appium and also automating the whole thing. Appium can be run on the same machine where you are running the tests.

**Q18) Do you think it is possible to interact with the apps by using java script while you are testing the Appium?**

While you are testing the Appium it is possible for you to interact with the apps using Java script. During the time when the commands are being run on Appium, the server will then send the script to the app that is wrapped in an anonymous function which is to be executed.

**Q19) What are the most difficult scenarios that one might face while testing Appium?**

Data exchange is the most difficult scenario that one might face while testing Appium.

**Q20) Is it possible to run tests in a multithreaded environment while you are using Appium?**

It is indeed possible to run tests on multithreaded environment but you have to make sure that no two tests are running simultaneously on the same Appium server.

**Q21) In case of Android platform is it necessary to have an app’s .apk so that it can automate using Appium or do we also need app in workspace?**

In case of android platform, to automate using Appium you will require only .apk file.

**Q22) What is an appium package master? How can you create an Appium Package?**

A set of tools that is required to create and manage appium packages are defined as Appium Package master. In order to create package use this following code:-  
#using es7/babe1  
Gulp create-package –n  
#regular es5  
Gulp create-package ---nobabe1 –n   
The package will be created in the out/

**Q23) What is the underlying API that appium follows?**

The underlying selenium API is followed by Appium so that it can automate test cases. It is said that since all the selenium APIs are present in Appium as well so Appium is an extension to the selenium.

**Q24) How can you inspect elements that are present in Native Android App?**

With the help of the UIAutomator tool that in present in android SDK, you will be able to access those object locators that are part of the Android Native app.

**Q25) Mention the method with which you can scroll down in App?**

With the help of the scrollTo () method, you will be able to scroll down in App. Also such a method will help you to automatically scroll until the specific text is not match.

**Q26) Is it possible to start an appium server programmatically?**

It is possible to start an appium server programmatically. Generally the commands that are being entered to the command prompt are written in note pad and it is saved with .bat extension and you can click the bat file.

**Q27) How can you inspect elements on the IOS apps?**

With the help of using appium inspector that is a GUI based tool you can identify elements on IOS apps. These GUI based tools are quite similar to that of selenium IDE.

**Q28) How is it possible to identify Mobile browser objects?**

You can make use of User Agent in order to identify objects in Mobile browser. It is done by using the user agent and then changing the browser as the mobile proxy and thus gets an object.

**Q29) Write the command that will allow you to identify objects uniquely when there are multiple objects within the same class name using appium?**

With the help of the command driver.findElements (By.className) it is possible to identify the elements uniquely.

**Q30) Give the difference between emulator and simulator?**

The simulator is used for calling IOS virtual devices that will launch from Xcode in MAC. The emulator is used for calling Android virtual devices.

[Check Out Appium Tutorials](https://mindmajix.com/appium)

**Q31) Do I need Appium?**

The answer to such a question is always: “It depends on what you need!”. So the actual question becomes: “Which conditions make Appium suitable for me?”. The most important assumption is that you are developing apps (pretty obvious I know). If you are developing an app for a specific platform (and have no intention of supporting others in future), Appium is not really required and this is basically the answer you are looking for. Appium becomes meaningful only when you have apps targeting more than one platform (Windows, Android or iOS to cite some). Appium becomes essential if you have a webview-based app (necessarily) targeting many platforms out there.

**Q32) How difficult is it to set up a working environment?**

The assumption is that Appium comes with a not-so-tiny documentation, so users are not really left alone. However it is not so straightforward to set up Appium to work on a Windows or Mac machine (did not try on Unix so far). In my experience, instead of installing the GUI-based application, it is much better to install the command-line application (which is released more often). Also beware [sudo], as Appium will surely bite you back late in time if you installed it as a [superuser] (this is probably the clearest point in the documentation)

**Q33) Can Appium be used for all my tests?**

This is an implied question in this question. The answer is **No** (in general). As I said before Appium is not suitable for all types of tests you might want to write (this depends on the functionalities you need to cover). There are some scenarios that can be difficult to test and some of them are so platform specific that you will need to write some suites just for Android or iOS for example. Remember that you can always get to do something no matter how hard it is, so you can test all your difficult scenarios using Appium, but always keep in mind one question: is it worth the time and the pain? Having Appium testing some scenarios leaving a few tests to other approaches is fine too! World is not black and white!

**Q34) What is Appium’s most considerable limitation?**

Hand down my chin starting thinking and mumbling. If I had to provide one single thing you should be aware of about Appium before starting using it, it would surely be: **multiple session handling**. Since Appium is a server, it serves HTTP requests; you might have two different computers running a test each against the same Appium server: what happens? As for now, Appium does not support this scenario and the second test will be aborted. This is a considerable limitation, because no queuing system comes with Appium. If you need to support multiple sessions, you will need to implement this feature by yourself.

**Q35) How active is Appium?**

Appium is available on GITHUB and there you can find all you need. The Appium team is responsible for developing many different subsystems revolving around Appium (like APIs for different languages), thus I can tell you that this product is alive and very active. The team is also pretty well responsive and once you open an issue you will find a reply after no more than 36 hours (this ETA comes by my personal experience). The community around Appium is also pretty large and growing every month.

**Q36) Which approach is the best? Testing on real devices or simulators/emulators?**

This is a tough question because both options offer different levels of testability and flexibility when testing. There are also many problems associated with each. So my answer will be again: “It depends on your needs!”.  
Running test on a device is, always in my opinion, the best solution because it offers a testing environment completely aligned with the running environment: tests run on those devices where your apps will be used  once published on stores. However devices must be connected to the Appium server via USB at least, and this is not always a very nice thing. ADB has a known issue for which a device disconnects after a while (even though it remained plugged all the time): because of this your tests might fail after a while and Appium will report that a device could not be found! I had to write a component which resets ADB after some time so that devices will not disconnect.

**Q37) Tests on emulators or simulators?**

On the other hand emulators/simulators will never disconnect from Appium. They also offer nice options like the ability of choosing the orientation or other hardware-related configurations. However your tests will run much slower (sadly, my tests ran 3 times slower) and do expect some crazy behavior from the Android emulator which sometimes shuts down unexpectedly. Another problem is that emulators tend to allocate a lot of memory.

**Q38) I already have platform-specific tests for my app, what should I do to migrate to Appium?**

Unfortunately there is not a magic formula to translate your tests into Selenium tests. If you developed a test framework on different layers and observed good programming principles, you should be able to act on some components in your tests in order to migrate your suites to Appium. Your current tests are going to be easy to migrate if they are already using an automation framework or something close to a command-based interaction. Truth being told, you will probably need to write your tests from the beginning, what you can do is actually reusing your existing components.

**Q39) How much time does it take to write a test in Appium?**

Of course it depends by the test. If your test simply runs a scenario, it will take as many commands as the number of interactions needed to be performed (thus very few lines). If you are trying to exchange data, then your test will take more time for sure and the test will also become difficult to read.

**Q40) Any tips or tricks to speed up my test writing activity or my migration process?**

Here is **one piece of advice**. Since your tests will mostly consist in automation tasks (if this condition is not met, you might want to reconsider using Appium), make interactions reusable! Do not write the same sub-scenarios twice in your tests, make a diagram of what your scenarios are and split them in sub activities; you will get a graph where some nodes are reachable from more than one node. So make those tasks parametric and call them in your tests! This will make your test writing experience better even when you need to migrate from existing tests (hopefully you already did this activity for your existing suites).

**Q41) What test frameworks are supported by Appium?**

Appium does not support test frameworks because there is no need to support them! You can use Appium with **all test frameworks** you want. NUNIT and.NET UNIT TEST FRAMEWORK are just a few examples; you will write your tests using one of the drivers for Appium; thus your tests will interface with Appium just in terms of an external dependency. Use whatever test framework you want!

**Q42) Is it possible to interact using Javascript with my apps, while the appiumappiumtesing running?**

Yeah, It is possible**!** Selenium has commands to execute Javascript instructions on your app from your tests. Basically you can send a JS script from your test to your app; while the running commands on Appium, the server sends the scripting to the apps you used by wrapping into an anonymous function that have to be executed.

**Q43) Is it Returning the values?**

However your Javascript interaction can get more advanced as your script can return a value which will be delivered to your test when the HTTP response is sent back by Appium once your Javascript has finished running. However this scenario comes with a limitation: your Javascript can send back only primitive types (integers, strings), not complex objects. The limitation can be overtaken by passing objects as JSON strings or by modifying Appium’s or Selenium’s code to support specific objects.

**Q44)**

**Q45) What data exchange is?**

When I say “data exchange” I am not referring to scenarios like getting or setting the value of a textbox. I am also not referring to getting or setting the value of an element’s attribute. All these things are easy to achieve in Appium as Selenium provides commands just for those. By “data exchange” I mean exchanging information hosted by complex objects stored in different parts of your webview-based app like the window object. Consider when you dispatch and capture events, your app can possibly do many things and the ways data flows can be handled are many. Some objects might also have a state and the state machine behind some scenarios in your app can be large and articulated. For all these reasons you might experience problems when testing.

**Q46)**

**Q47) Is it Exchanging data through Javascript?**

Selenium provides commands do execute Javascript on the app, it is also possible to execute functions and have them return data (only basic types). If you exchange JSON strings it should be fine as JSON.stringify(str) will turn yourJSON string into an object on the app side, while on the test side (depending on the language you are using), you can rely on hundreds of libraries to parse the string you receive.

**Q48) What are the most difficult scenarios to test with Appium?**

Appium is not suitable for all types of tests. There is a particular scenario that will make your tests more difficult to write: **data exchange**. I already said it but I will repeat the same thing because it is very important: Appium and WebDriver are designed to automate stuff… not to exchange data with them. So what if we need to exchange information with our app during tests? Should we give up on Appium and write our tests manually for each platform? I am not saying this, but there are cases where you should consider this option (not nice I know, but if the effort of writing tests for Appium is higher than the benefits, than just throw Appium away).  
Appium is very nice because it will let you write tests once for all platofrms instead of writing as many tests as the numbers of platforms you need to support. So if you need to exchange data with your app while testing it and this data flow is the same for all platforms, then you should probably keep on using Appium and find a way to write a layer on top of it to handle data. Depending on your needs this might take time, but, in my experience, it is really worth it.

**Q49) I don’t want to set up a whole infrastructure for my tests and I don’t want to spend money on HW. Can Appium help me?**

If you think about it, what really is required from you is writing tests. Then the fact that you must deploy an Appium server somewhere is something more. If you want to skip this part, you can rely on some web services that already deployed a whole architecture of Appium servers for your tests. Most of them are online labs and they support Selenium and Appium.

**Q51) I build my apps with CORDOVA, is it supported by Appium?**

CORDOVA is a very famous system that enables you to develop webview-based apps for all platforms in short time. Appium does not explicitely say that Cordova is supported, even though they do it implicitely as some examples using apps built with Cordova are provided on Appium’s website. So the answer is that **Cordova should not be a problem**. Why am I being so shy about it? Because anything can happen and it actually happened to me!  
Cordova and Appium are two different projects that are growing up separately and independently, of course a mutual acknowledgement is present, but both teams do not really talk to each other when pushing features. So problems can occur (I am currently dealing with a problem concerning Cordova’s new version which is causing my tests to fail).

**Q52) What are the basic commands that I can use in the SELENIUM protocol?**

Google’s SELENIUM provides a collection of commands to automate your app. With those commands you can basically do the following:  
Locate web elements in your webview-based app’s pages by using their ids or class names.  
Raise events on located elements like Click().  
Type inside textboxes.  
Get or set located element’s attributes.  
Execute some Javascript code.  
Change the context in order to test the native part of your app, or the webview. If your app uses more webviews, you can switch the context to the webview you desire. If your webview has frames or iframes inside, you can change context to one of them.  
Detect alert boxes and dismiss or accept them. Be careful about this functionality, I experienced some problems.

**Q53) I want to run my tests in a multithreaded environment, any problems with that?**

Yes! You need some special care when using Appium in a multithreaded environment. The problem does not really rely on the fact of using threads in your tests: you can use them but you must ensure that no more than one test runs at the same time against the same Appium server. As I mentioned, Appium does not support multiple sessions, and unless you implemented an additional layer on top of it to handle this case, some tests might fail.

**Q54) How can I run Android tests without Appium?**

For older versions of Android Appium might not be supported. For instance, Appium is only supported in Android versions 4.4 or later for MOBILE WEB APPLICATION tests, and Android versions 2.3, 4.0 and later for MOBILE NATIVE APPLICATION and MOBILE HYBRID APPLICATION tests.  
For those versions in which Appium is not supported you can request an emulator driven by Webdriver + Selendroid. All you need to do is use our PLATFORMS CONFIGURATOR and select Selenium for the API instead of Appium.  
In the Sauce Labs test you will notice that the top of the emulator says “AndroidDriverWebview App”. In addition, you will notice that you will get a “Selenium Log” tab which has the output of the Selendroid driver.  
With an emulator driven by Webdriver + Selendroid you will be able to testMOBILE WEB APPLICATION only. You should be able to select any Android emulator version from 4.0 to the latest version and any Android emulator skin (e.g “deviceName”:”Samsung Galaxy Tab 3 Emulator”).

**Q55) How can I run iOS tests without Appium?**

For older versions of iOS Appium might not be supported. For instance, Appium is supported in iOS versions 6.1 and later. For earlier versions of iOS the tool or driver used to drive your mobile applications automated test is called iWebdriver.  
To obtain a simulator driven by iWebdriver use our PLATFORMS CONFIGURATOR and select Selenium for the API instead of Appium. With an emulator driven by iWebdriver you will be able to test MOBILE WEB APPLICATION only. In addition, in the Sauce Labs test you will notice a “Selenium Log” tab which has the output of iWebdriver.

**Q56) What mobile web browsers can I automate in the Android emulator?**

Currently the only browser that can be automated in our Android emulators is the stock browser (i.e Browser). The Android stock browser is an Android flavor of ‘chromium’ which presumably implies that its behavior is closer to that of Google Chrome.