Introduction to AWS Device Farm

**SPL-27 - Version 1.5.13**

© 2023 Amazon Web Services, Inc. or its affiliates. All rights reserved. This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited. All trademarks are the property of their owners.

Note: Do not include any personal, identifying, or confidential information into the lab environment. Information entered may be visible to others.

Corrections, feedback, or other questions? Contact us at [*AWS Training and Certification*](https://support.aws.amazon.com/#/contacts/aws-training).

**Overview**

This lab shows you how to use AWS Device Farm to test a sample Android mobile app. In this lab, you will use the AWS Device Farm console to create a project, upload a sample .apk file, run a suite of standard tests, and then view the results.

**Topics covered**

By the end of this lab you will be able to:

* Create an AWS Device Farm Project
* Upload a sample Android mobile app into the AWS Device Farm Project
* Test the mobile app using a basic suite of tests in AWS Device Farm
* View the results of the test suite run

**Prerequisites**

Some familiarity with mobile application development and deployment platforms, and concepts of application testing, will be helpful, but not necessary to run the lab.

**AWS Device Farm Introduction**

AWS Device Farm is an app testing service that enables you to test your iOS, Android and Fire OS apps on real, physical phones and tablets that are hosted by AWS. The service allows you to upload your own tests or use built-in, script-free compatibility tests. A test report containing high-level results, low-level logs, pixel-to-pixel screenshots, and performance data is updated as tests are completed.

Device Farm supports native and hybrid Android, iOS, and Fire OS apps, including those created with PhoneGap, Titanium, Xamarin, Unity, and other frameworks.

**AWS Device Farm Terminology**

Device Farm introduces the following terms that define the way information is organized:

* **project:** A logical workspace that contains runs, one run for each test of a single app against one or more devices. Projects enable you to organize workspaces in whatever way you choose. For example, there can be one project per app title, or there can be one project per platform. You can create as many projects as you need.
* **run:** A specific build of your app, with a specific set of tests, to be run on a specific set of devices. A run produces a report that contains information about the results of the run. A run contains one or more jobs. For more information, see the AWS Documentation for AWS Device Farm [Runs](http://docs.aws.amazon.com/devicefarm/latest/developerguide/test-runs.html).
* **report:** Contains information about a run, which is a request for Device Farm to test a single app against one or more devices. For more information, see the AWS Documentation for AWS Device Farm [Reports](http://docs.aws.amazon.com/devicefarm/latest/developerguide/reports.html).
* **job:** A request for Device Farm to test a single app against a single device. A job contains one or more suites.
* **suite:** The hierarchical organization of tests in a test package. A suite contains one or more tests.
* **test:** An individual test within a test package.

**AWS Device Farm Test Types**

Device Farm currently provides support for the following test types:

**For Android:**

* [Appium Java JUnit](https://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-appium.html)
* [Appium Java TestNG](https://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-appium.html)
* [Calabash](https://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-calabash.html)
* [Instrumentation](http://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-android-instrumentation.html) (JUnit, Espresso, Robotium, or any instrumentation-based tests)
* [UI Automator](http://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-android-uiautomator.html)
* [Explorer](http://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-built-in-explorer.html)

**For iOS:**

* [Appium Java JUnit](https://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-appium.html)
* [Appium Java TestNG](https://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-appium.html)
* [Calabash](https://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-calabash.html)
* [UI Automation](http://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-ios-ui-automation.html)
* [XCTest](http://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-ios-xctest.html) (including KIF)

If you do not have your own tests, you can use a built-in fuzz test. The built-in fuzz test randomly sends user interface events to devices and then reports results.

This hands on lab will demonstrate this feature. For more information on the tests implemented in Fuzz, see [Built-in: Fuzz (Android and iOS)](http://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-built-in-fuzz.html).

**Start lab**

1. To launch the lab, at the top of the page, choose **Start lab**.

 You must wait for the provisioned AWS services to be ready before you can continue.

1. To open the lab, choose **Open Console**.

You are automatically signed in to the AWS Management Console in a new web browser tab.

**Do not change the Region unless instructed.**

COMMON SIGN-IN ERRORS

**Error: You must first sign out**



If you see the message, **You must first log out before logging into a different AWS account:**

* Choose the **click here** link.
* Close your **Amazon Web Services Sign In** web browser tab and return to your initial lab page.
* Choose **Open Console** again.

**Error: Choosing Start Lab has no effect**

In some cases, certain pop-up or script blocker web browser extensions might prevent the **Start Lab** button from working as intended. If you experience an issue starting the lab:

* Add the lab domain name to your pop-up or script blocker’s allow list or turn it off.
* Refresh the page and try again.

**Task 1: Locate or Download an Example Android \*.apk or iOS \*.ipa File**

If you are a mobile developer, and you have a compiled Android \*.apk or iOS \*.ipa file that you would like to use for this lab to test, locate the compiled \*.apk or \*.ipa at this time.

If you do not have your own mobile app to test, below are some examples available publicly to test in this lab. These examples may move or change from time to time. The test results may vary and may not all pass or fail. Each will provide interesting test results nonetheless.

1. Please download one of the following files to your computer:

* [flickrj-android-sample-android.apk](https://code.google.com/p/flickrj-android/downloads/detail?name=flickrj-android-sample-android.apk)
* [mixarev0.9.2.apk](https://code.google.com/p/mixare/downloads/detail?name=mixarev0.9.2.apk)
* [github.com/mehtank/androminion/releases](https://github.com/mehtank/androminion/releases)

1. At the top of the AWS Management Console, to the right of **Services** menu, in the search bar, search for

**Device Farm**

 and then choose **Device Farm** from the list.

1. If you see **Next**, click it.
2. At the **AWS Device Farm** window, configure:

* Project Name:

myProject

* Click **Create**

If you type a project name other than **myProject**, be sure to use it consistently throughout this lab.

**Task 2: Upload and Test the Example Application**

1. Click **Create a new run**
2. On **Step 1 - Choose application**, configure the following:

* Click  **Upload**
* Browse to and select the application that you downloaded.

AWS Device farm processes and analyzes the mobile app. A progress bar is displayed, and when complete, a summary of the analysis is presented.

1. At the bottom of the screen, click **Next**
2. Ensure the following values set:

* **Test:** *Built-in Fuzz*
* **Event count:**

6000

* **Event throttle:**

50

1. Click **Next**
2. On the **Select devices** page, click **Next**

You will use the **Top Devices** pool.

1. On the **Specify device state** page, click **Next**

A device pool in Device Farm represents a collection of devices that typically share similar characteristics such as platform, manufacturer, or model. For more information, see [Devices](http://docs.aws.amazon.com/devicefarm/latest/developerguide/devices.html).

You will skip **specifying the device state**. However, if you’d like, you can adjust the parameters on the **Specify device state** page. On the **Specify device state**, you can do any of the following:

* Provide additional data for Device Farm to use during the run, next to Add extra data, choose Upload, and then browse to and choose the .zip file
* Specify whether Wi-Fi, Bluetooth, GPS, or NFC will be enabled during the run, next to Set radio states, select the appropriate boxes
* Preset the device latitude and longitude for the run, next to Device location, type the coordinates
* Reset the device locale for the run, choose the locale in Device Locale

**Task 3: Run Test and View the Run’s Results**

1. On the **Review and start run** page:

* Review your settings
* Click **Confirm and start run**

Device Farm should start the run as soon as the requested devices are provisioned and available, typically within a few minutes. Until the run starts, Device Farm will display a calendar icon. After the run starts, results will appear as tests are completed. During this time, Device Farm will display a progress icon .

You’ll know the run is complete when the progress icon changes to a result icon. Depending on the complexity of the application, a full test run can take from less than a minute to ten minutes or more.

1. Click the test result to view the run’s results.

A summary page that includes the following information is displayed.

* A list of devices and test results for each
* The total number of suites, by outcome
* Lists of tests with unique warnings or failures
* Screenshots captured during the run, grouped by device

**Conclusion**

 Congratulations! You now know how to:

* Create an AWS Device Farm Project
* Upload a sample Android mobile app into the AWS Device Farm Project
* Test the mobile app using a basic suite of tests in AWS Device Farm
* View the results of the test suite run

**End lab**

Follow these steps to close the console and end your lab.

1. Return to the **AWS Management Console**.
2. At the upper-right corner of the page, choose **AWSLabsUser**, and then choose **Sign out**.
3. Choose **End lab** and then confirm that you want to end your lab.

**Additional Resources**

[Logging AWS Device Farm API Calls by Using AWS CloudTrail](https://docs.aws.amazon.com/devicefarm/latest/developerguide/logging-using-cloudtrail.html)

Device Farm is integrated with CloudTrail, a service that captures API calls made by or on behalf of Device Farm in your AWS account and delivers the log files to an Amazon S3 bucket you specify. Examples of these API calls include creating a new project or run in Device Farm. CloudTrail captures API calls from the Device Farm console or the Device Farm APIs. Using the information collected by CloudTrail, you can determine which request was made to Device Farm, the source IP address from which the request was made, who made the request, when it was made, and so on.

* [Information about Device Farm](http://docs.aws.amazon.com/devicefarm/latest/developerguide/welcome.html)
* [AWS Mobile Hub](https://aws.amazon.com/mobile/)
* [AWS Mobile SDK](https://aws.amazon.com/mobile/sdk/)

For more information about AWS Training and Certification, see [*https://aws.amazon.com/training/*](https://aws.amazon.com/training/).

*Your feedback is welcome and appreciated.*  
If you would like to share any feedback, suggestions, or corrections, please provide the details in our [*AWS Training and Certification Contact Form*](https://support.aws.amazon.com/#/contacts/aws-training).