

Introduction to AWS Device Farm in 10 min

What Is AWS Device Farm?

PDF

Device Farm is an app testing service that you can use to test and interact with your Android, iOS, and web apps on real, physical phones and tablets that are hosted by Amazon Web Services (AWS).

There are two main ways to use Device Farm:

- Automated testing of apps using a variety of testing frameworks.
- Remote access of devices onto which you can load, run, and interact with apps in real time.

The screenshot shows the AWS Device Farm console. At the top, the AWS logo and 'Services' dropdown are visible. The main header area displays 'AWS Device Farm' in orange, with a description: 'AWS Device Farm enables you to improve the quality of your Android, iOS, and Fire OS apps by quickly and securely testing them on 100s of real smartphones, tablets, and other devices in the AWS cloud.' Below this, a modal window titled 'Run a test. Where do you want to run your tests?' is open. It has two radio buttons: 'Mobile Device Project' (selected) and 'Desktop Browser Project'. A text input field contains 'Walter_Project1', and a blue 'Create project' button is at the bottom. Below the modal, three numbered icons illustrate key features: 1. 'Test often' (icon of multiple devices) with text: 'Test your app on a device fleet comprised of hundreds of unique devices that scale with customer demand, ensuring fast and secure app testing. Test execution can be automated, allowing you to execute the same test cases on multiple devices in parallel, or manual, allowing you to swipe, gesture, and more.' 2. 'Find issues quickly' (icon of a device with a warning sign) with text: 'View comprehensive, actionable reports as tests complete on each device. In addition to containing test results, detailed logs, screenshots, video, and performance data, reports identify and group identical errors across multiple devices, allowing you to quickly and efficiently analyze data from potentially hundreds of devices.' 3. 'Control how you test' (icon of a device with a play button) with text: 'Run your own tests written in a number of popular, open-source test frameworks or run our built-in compatibility test suite to fuzz-test and crawl your app with no scripting necessary. You can also execute manual tests and reproduce customer issues via remote access. Fine-tune your test.' The footer contains 'Feedback', 'English (US)', copyright information '© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.', and links to 'Privacy Policy' and 'Terms of Use'.

Step1: create a project and new test run:

The screenshot displays the AWS Device Farm console. At the top, the 'AWS Device Farm' header is visible, along with a description: 'AWS Device Farm enables you to improve the quality of your Android, iOS, and Fire OS apps by quickly and securely testing them on 100s of real smartphones, tablets, and other devices in the AWS cloud.' Below this, a modal dialog titled 'Run a test. Where do you want to run your tests?' is shown. It has two radio buttons: 'Mobile Device Project' (selected) and 'Desktop Browser Project'. A text input field contains 'Walter_Project1', and a blue 'Create project' button is at the bottom. Below the dialog, three numbered icons illustrate the workflow: 1. 'Test often' (icon of a device with a refresh symbol), 2. 'Find issues quickly' (icon of a device with a warning symbol), and 3. 'Control how you test' (icon of a device with a checkmark). Each icon has a brief description of its function. At the bottom, the 'Automated tests' section is visible, showing a 'Create a new run' button and a table with columns: Run, Test results, Test Type, Created, and Total minutes. The table is currently empty, with a message 'You don't have any automated runs yet...'.

AWS Device Farm

AWS Device Farm enables you to improve the quality of your Android, iOS, and Fire OS apps by quickly and securely testing them on 100s of real smartphones, tablets, and other devices in the AWS cloud.

Run a test. Where do you want to run your tests?

☒ Mobile Device Project ☐ Desktop Browser Project

Walter_Project1

Create project

1 Test often

Test your app on a device fleet comprised of hundreds of unique devices that scale with customer demand, ensuring fast and secure app testing. Test execution can be automated, allowing you to execute the same test cases on multiple devices in parallel, or manual, allowing you to swipe, gesture,

2 Find issues quickly

View comprehensive, actionable reports as tests complete on each device. In addition to containing test results, detailed logs, screenshots, video, and performance data, reports identify and group identical errors across multiple devices, allowing you to quickly and efficiently analyze data from potentially hundreds of

3 Control how you test

Run your own tests written in a number of popular, open-source test frameworks or run our built-in compatibility test suite to fuzz-test and crawl your app with no scripting necessary. You can also execute manual tests and reproduce customer issues via remote access. Fine-tune your test

Feedback English (US) © 2008 - 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

AWS Services

Device Farm Mobile Device Project: Walter_Project1 Runs & sessions Learn more about unlimited testing

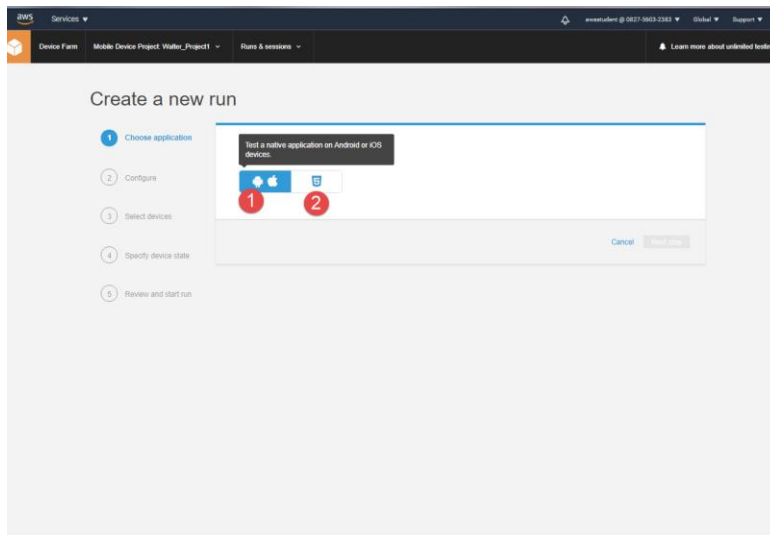
Automated tests Remote access Project settings

Automated runs allow you to execute built-in tests or your own scripts on one or more devices in parallel, generating a comprehensive report that includes high-level results, logs, screenshots, and performance data.

Create a new run

Run	Test results	Test Type	Created	Total minutes
You don't have any automated runs yet...				

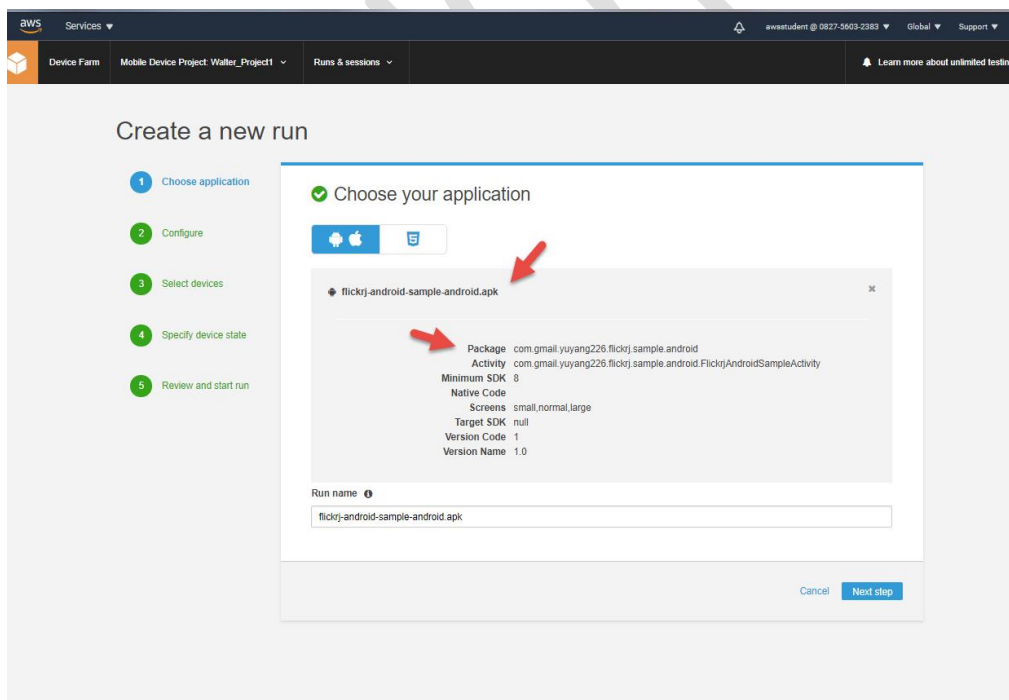
Select Android/Apple App or Mobile web app:



Configure the new run:

Upload your apk or sample app from :

[sample test app at https://code.google.com/archive/p/flickrj-android/downloads](https://code.google.com/archive/p/flickrj-android/downloads)

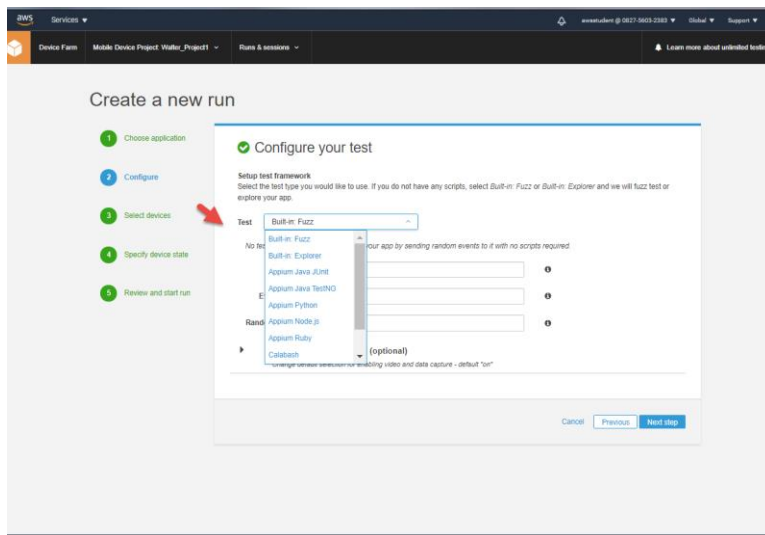


Select the test suites

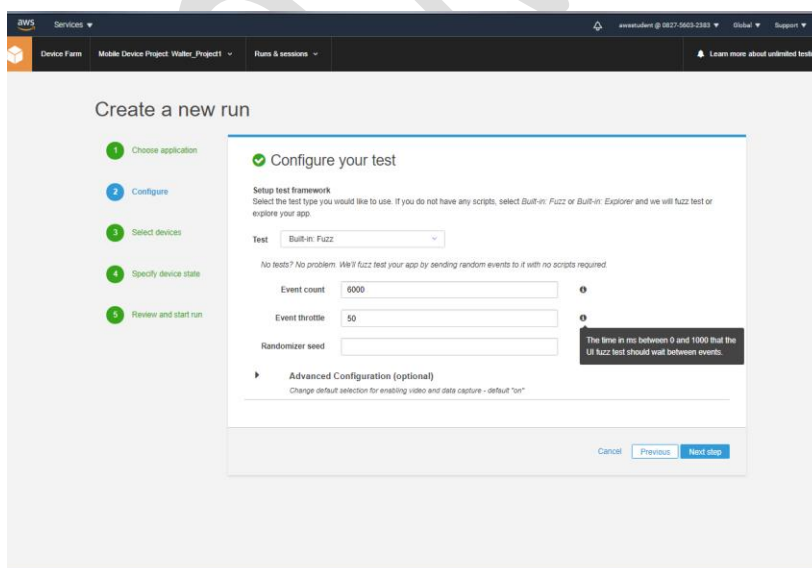
Many ready to use tests from the list:

For iOS:

- [Appium Java JUnit](#)
- [Appium Java TestNG](#)
- [Calabash](#)
- [UI Automation](#)
- [XCTest](#) (including KIF)



This hands on lab will demonstrate this feature. For more information on the tests implemented in Fuzz, see [Built-in: Fuzz \(Android and iOS\)](#).



You can enable Video Recording and App Perf data capture:

Create a new run

- Choose application
- Configure**
- Select devices
- Specify device state
- Review and start run

Configure your test

Setup test framework
Select the test type you would like to use. If you do not have any scripts, select *Built-in: Fuzz* or *Built-in: Explorer* and we will fuzz test or explore your app.

Test:

No tests? No problem. We'll fuzz test your app by sending random events to it with no scripts required.

Event count:

Event throttle:

Randomizer seed:

Advanced Configuration (optional)
Change default selection for enabling video and data capture - default "on"

Other Configurations

- ☒ Enable Video Recording
- ☒ Enable App Performance Data Capture

If checked, enables capture of performance data from the device.

Cancel Previous Next step

Select Devices

Create a new run

- Choose application
- Configure
- Select devices**
- Specify device state
- Review and start run

Select devices

Select from one of the available device pools or create a new device pool.

Device pool:

Curated pools:

100% Web Performance
Your app is compatible with 5 out of 5 devices in the selected pool.

Device	OS	Reason	Status
✓ Samsung Galaxy S9 (Unlocked)	9		AVAILABLE
✓ Samsung Galaxy Note 10	9		HIGHLY_AVAILABLE
✓ Samsung Galaxy S10+	9		HIGHLY_AVAILABLE
✓ Google Pixel 4 (Unlocked)	10		HIGHLY_AVAILABLE
✓ Samsung Galaxy Tab S6 (WiFi)	9		HIGHLY_AVAILABLE

Cancel Previous Next step

Specify Device state

The screenshot shows the 'Specify device state' configuration page in AWS IoT Device Studio. The left sidebar contains a progress indicator with five steps: 1. Choose application, 2. Configure, 3. Select devices, 4. Specify device state (current step), and 5. Review and start run. The main content area is titled 'Specify device state' and includes the following sections:

- Add extra data:** A button labeled 'Upload' and the text 'Or drop your file here'.
- Install other apps:** A button labeled 'Upload', the text 'Or drop your file here', and a dropdown menu labeled 'Select a recent upload'.
- Set radio states:** A list of checkboxes: ☒ WiFi, ☐ Bluetooth, ☒ GPS, and ☒ NFC.
- Device location:** Two input fields containing the coordinates '47.6204' and '-122.3491'.
- Paths to your files on the host machine and device:** Two input fields. The 'Host Machine' field contains '\$WORKING_DIRECTORY'.
- Android:** An empty input field.
- Device locale:** A dropdown menu showing 'English, US (en_US)'.
- Network profile:** A dropdown menu showing 'Full' and a button labeled 'Create a new network profile'.

A dark grey tooltip is visible on the right side of the page, stating: 'Upload an Android app as a .apk. No instrumentation or provisioning is required.'

You can select Network speed profile:

This screenshot shows the same 'Specify device state' configuration page, but with the 'Network profile' dropdown menu open. The menu lists the following options: 'Full ~', 'Curated profiles', '3G Average', '3G Good', '3G Lossy', 'Disabled', 'EDGE Average', and 'EDGE Good'. Two red arrows point to the '3G Good' and 'EDGE Average' options. At the bottom right of the configuration area, there are three buttons: 'Cancel', 'Previous', and 'Next step'.

Also, can select Device Languages/Locale, e.g. Arabic, Spanish and Chinese, etc.

Review and start Run:

The screenshot shows the 'Create a new run' wizard in the AWS Device Farm console. The 'Review and start run' step is active, indicated by a green checkmark and a red arrow. The left sidebar shows the progression: 1. Choose application, 2. Configure, 3. Select devices, 4. Specify device state, and 5. Review and start run. The main content area displays the 'Execution timeout' configuration, showing a value of 150 minutes per device for 5 devices, totaling 750 minutes. Below this is a slider ranging from 5m to 150m. A note explains that the timeout should be greater than the anticipated test duration. The configuration summary at the bottom lists: Application (flickrj-android-sample-android.apk), Test (Built-in: Fuzz), and Devices (Pool Top Devices). 'Edit' links are provided for each section. At the bottom right, there are 'Cancel' and 'Confirm and start run' buttons.

Then see the tests started:

The screenshot shows the 'Automated tests' page in the AWS Device Farm console. The 'Run' tab is selected, and a table lists the test runs. A red arrow points to the first run, which is 'flickrj-android...'. The table has columns for 'Run', 'Test results', 'Test Type', 'Created', and 'Total minutes'. The first run is 'flickrj-android...' with a test type of 'Built-in: Fuzz' and a creation time of '2020-10-10T22:28-0700'. The 'Run' column shows a circular refresh icon. The 'Test results' column is empty. The 'Total minutes' column is also empty. The 'Created' column shows the timestamp. The 'Test Type' column shows 'Built-in: Fuzz'. The 'Run' column shows a circular refresh icon. The 'Test results' column is empty. The 'Total minutes' column is also empty.

12 out of 15 tests done:

The screenshot shows the AWS Device Farm console for a project named 'Mobile Device Project: Walter_Project1'. Under the 'Automated tests' tab, a test run for 'flickjr-android-sample-android.apk' is shown. The progress bar indicates 12 out of 15 tests completed. Red arrows point to the refresh icon, the progress bar, and the total minutes.

Run	Test results	Test Type	Created	Total minutes
flickjr-android...	12	Built-In: Fuzz	2020-10-10T22:28-0700	00:13:18

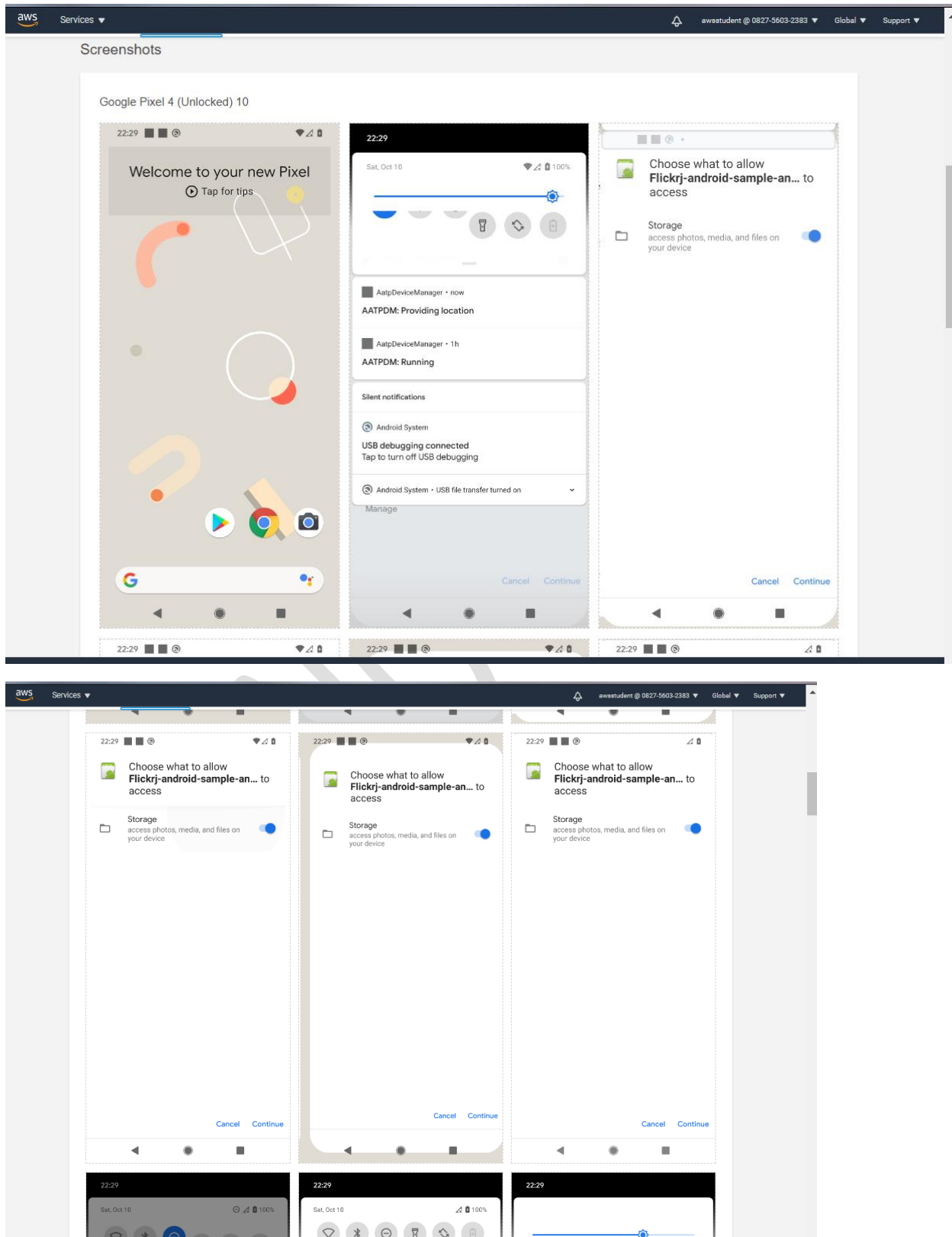
Finally all 15 tests done and about 2:45 min each:

The screenshot shows the AWS Device Farm console for a project named 'Mobile Device Project: Walter_Project1'. Under the 'Automated tests' tab, a test run for 'flickjr-android-sample-android.apk' is shown. The progress bar indicates 15 out of 15 tests completed. A green message box says 'Congratulations! All tests passed on all devices.' Below, a table lists the devices used.

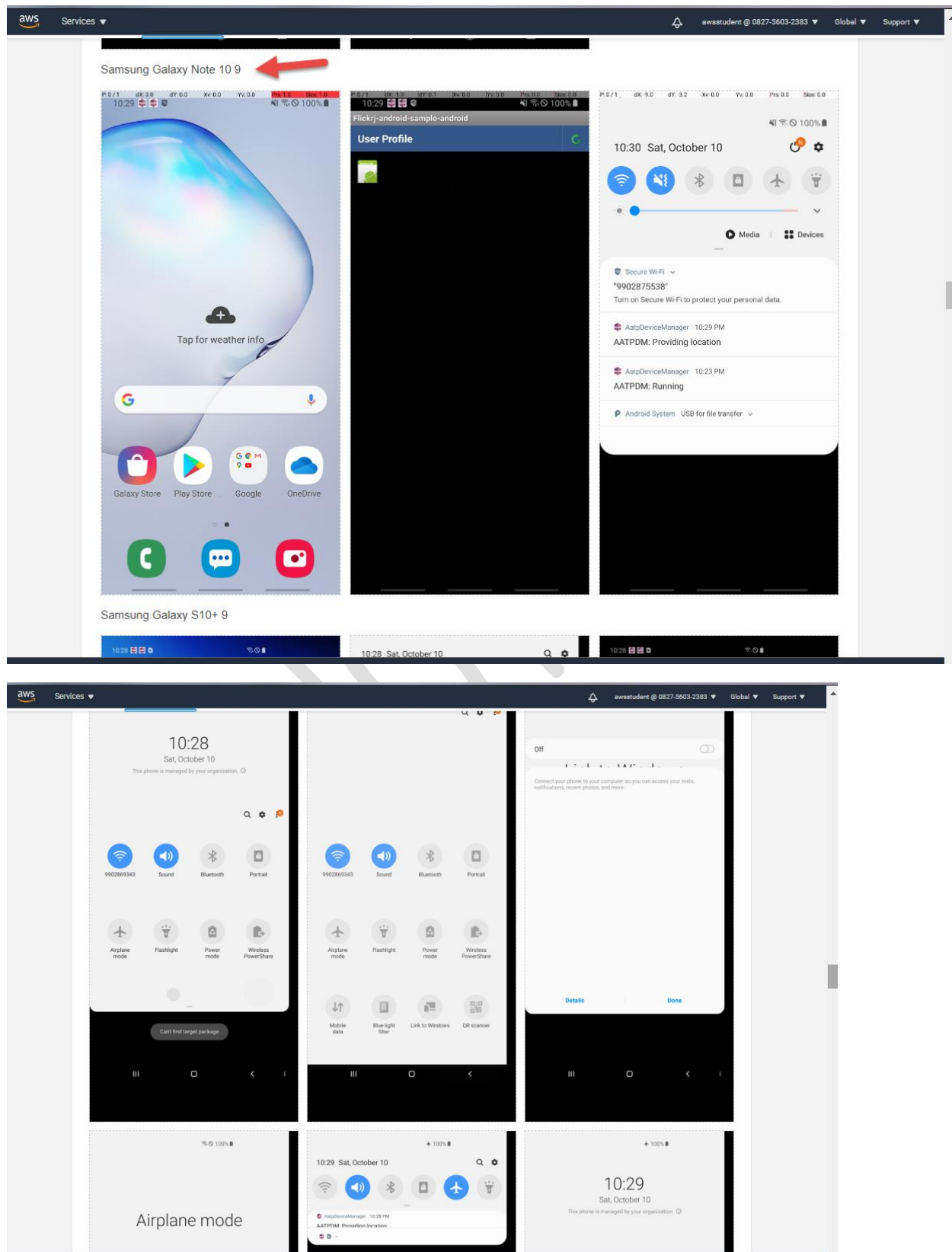
Device	OS	Test results	Total minutes ...
1 Google Pixel 4 (Unlocked)	10	3	00:02:28
2 Samsung Galaxy Note 10	9	3	00:02:45
3 Samsung Galaxy S10+	9	3	00:02:59
4 Samsung Galaxy S9 (Unlocked)	9	3	00:02:32
5 Samsung Galaxy Tab S6 (WiFi)	9	3	00:02:32

Each to review results with Screenshots:

a/ first with Pixel 4 device



b/ second with Samsung Galaxy Note 10 device



c/ third with Samsung Galaxy Tab S6 (Wifi) device

Samsung Galaxy Tab S6 (Wifi) 9

Parsing result

[Download parsing result](#)

Feedback English (US) © 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

parsingResult.txt Show all

Samsung Galaxy Tab S6 (Wifi) 9

Parsing result

[Download parsing result](#)

Feedback English (US) © 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

parsingResult.txt Show all

Easy to create a new remote access session

<https://docs.aws.amazon.com/devicefarm/latest/developer/welcome.html>

“Remote access allows you to swipe, gesture, and interact with a device through your web browser in real time. There are a number of situations where real-time interaction with a device is useful. For example, customer service representatives can guide customers through the use or setup of their device. They can also walk customers through the use of apps running on a specific device. You can install apps on a device running in a remote access session and then reproduce customer problems or reported bugs.

During a remote access session, Device Farm collects details about actions that take place as you interact with the device. Logs with these details and a video capture of the session are produced at the end of the session.”

The screenshot shows the AWS Device Farm console interface. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, and user information 'awsstudent @ 0827-5603-2383'. Below this is a secondary bar with 'Device Farm' and 'Mobile Device Projects' tabs. The main content area is titled 'Create a new remote access session'. Underneath, there's a section 'Choose a device' with a sub-instruction: 'Select a device for an interactive session. Interested in unlimited, unmetered testing? [Learn more](#)'. A checkbox option reads: 'Show available devices only (Note: When a device is "AVAILABLE", your session will start within 30 seconds)'. Below this is a table of available devices.

Name	Status	Platform	OS	Form factor
<input type="radio"/> Google Pixel 4 (Unlocked)	AVAILABLE	Android	10	Phone
<input type="radio"/> Google Pixel 4 XL (Unlocked)	AVAILABLE	Android	10	Phone
<input type="radio"/> Samsung Galaxy S20 (Unlocked)	AVAILABLE	Android	10	Phone
<input type="radio"/> Google Pixel 2	AVAILABLE	Android	9	Phone
<input type="radio"/> Google Pixel 2 XL	AVAILABLE	Android	9	Phone
<input type="radio"/> Google Pixel 3	AVAILABLE	Android	9	Phone
<input type="radio"/> Google Pixel 3 XL	AVAILABLE	Android	9	Phone
<input type="radio"/> Samsung Galaxy A40	AVAILABLE	Android	9	Phone
<input type="radio"/> Samsung Galaxy A50	AVAILABLE	Android	9	Phone
<input type="radio"/> Samsung Galaxy A70	AVAILABLE	Android	9	Phone
<input checked="" type="radio"/> Samsung Galaxy S10	AVAILABLE	Android	9	Phone
<input type="radio"/> Samsung Galaxy S10+	AVAILABLE	Android	9	Phone
<input type="radio"/> Samsung Galaxy S10e	AVAILABLE	Android	9	Phone
<input type="radio"/> Samsung Galaxy Tab S6 (WiFi)	AVAILABLE	Android	9	Tablet
<input type="radio"/> Samsung Galaxy Tab S4	AVAILABLE	Android	8.1.0	Tablet
<input type="radio"/> Samsung Galaxy A5	AVAILABLE	Android	8.0.0	Phone

At the bottom of the console, there's a footer with 'Feedback', 'English (US)', copyright '© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.', 'Privacy Policy', and 'Terms of Use'. A taskbar at the very bottom shows a file named 'parsingResult.txt' and a 'Show all' button.

aws Services

awstudent @ 0827-5603-2383 Global Support

☐ Show available devices only (Note: When a device is "AVAILABLE", your session will start within 30 seconds)

Name	Status	Platform	OS	Form factor
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="radio"/> Apple iPhone 11	AVAILABLE	iOS	14.0	Phone
<input type="radio"/> Apple iPad Air 2	AVAILABLE	iOS	13.6	Tablet
<input type="radio"/> Apple iPhone SE (2020)	AVAILABLE	iOS	13.6	Phone
<input type="radio"/> Apple iPhone 8	AVAILABLE	iOS	13.5.1	Phone
<input type="radio"/> Apple iPhone 11	AVAILABLE	iOS	13.1.3	Phone
<input type="radio"/> Apple iPhone 11 Pro	AVAILABLE	iOS	13.1.3	Phone
<input type="radio"/> Apple iPhone 11 Pro Max	AVAILABLE	iOS	13.1.3	Phone
<input type="radio"/> Apple iPad Pro 11"	AVAILABLE	iOS	12.1	Tablet
<input type="radio"/> Apple iPhone 8 Plus	AVAILABLE	iOS	12.1	Phone
<input type="radio"/> Apple iPhone XS Max	AVAILABLE	iOS	12.1	Phone
<input type="radio"/> Apple iPhone X	AVAILABLE	iOS	12.0	Phone
<input type="radio"/> Apple iPhone XR	AVAILABLE	iOS	12.0	Phone
<input type="radio"/> Apple iPhone XS	AVAILABLE	iOS	12.0	Phone
<input type="radio"/> Apple iPhone 6	AVAILABLE	iOS	11.4.1	Phone
<input type="radio"/> Apple iPhone 8	AVAILABLE	iOS	11.0.3	Phone
<input type="radio"/> Apple iPhone 7	AVAILABLE	iOS	10.3.3	Phone

Name remote access session

Give a name to the remote access session.

Session name

Feedback English (US)

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

parsingResult.txt

aws Services

awstudent @ 0827-5603-2383 Global Support

Device Farm Mobile Device Projects

Learn more about unlimited testing

Samsung Galaxy S10

Device requested

Please wait while we find an available device.

Waiting for Samsung Galaxy S10

If you would like to keep working while you wait for your session to start, you can continue in a [new tab](#).

Feedback English (US)

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

parsingResult.txt

Show all

Then get the same Samsung Galaxy S10 device which can do interactively test more.

Time left: 02:29:08 **Stop session**

Notice

To download an app from Play Store, add your Google Account to the device. Once you do that, and switch to your account in Play Store, you will be able to see all apps in the Play Store. Note that AWS Device Farm captures video and logs of activity taking place during Remote Access session. It is recommended that you avoid entering your personal accounts on the device (E.g., personal Google account) and instead use test accounts where possible.

Device information

Name: Samsung Galaxy S10
 OS: 9
 Manufacturer: Samsung
 Model: 9
 Chipset: arm64-v8a 1785.6MHz
 Memory: 12800000000
 Heap: 512000000
 Display: 1440 x 3040

Install applications

[Upload](#) Or drop your file here or [Select a recent upload](#)

Feedback English (US) © 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

parsingResult.txt **Show all**

Easy to upload your APK file to test:

File name: flickrj-android-sample-android.apk

All Files (*.*)

Open **Cancel**

1

Install applications

[Upload](#) Or drop your file here or [Select a recent upload](#)

Feedback English (US) © 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

parsingResult.txt **Show all**

The screenshot shows the AWS Device Farm console interface. On the left, a virtual device screen displays a home screen with a search bar, a 'Google' app icon, and a 'Play Store' icon. Below the screen are navigation buttons: 'Back', 'Home', 'Recent apps', and 'Landscape'. On the right, a 'Notice' box provides instructions on downloading apps from the Play Store. Below the notice, the 'Device information' section lists details for a Samsung Galaxy S10. The 'Install applications' section features an 'Upload' button and a list of recent uploads, including 'flickrj-android-sample-android.apk', which is highlighted with a red arrow.

aws Services

awsstudent @ 0827-5603-2383 Global Support

Notice

To download an app from Play Store, add your Google Account to the device. Once you do that, and switch to your account in Play Store, you will be able to see all apps in the Play Store. Note that AWS Device Farm captures video and logs of activity taking place during Remote Access session. It is recommended that you avoid entering your personal accounts on the device (E.g., personal Google account) and instead use test accounts where possible.

Device information

Name: Samsung Galaxy S10
OS: 9
Manufacturer: Samsung
Model: 9
Chipset: arm64-v8a 1785.6MHz
Memory: 12800000000
Heap: 512000000
Display: 1440 x 3040

Install applications

Upload Or drop your file here or

Select a recent upload ^

Just added flickrj-android-sample-android.apk

Just added flickrj-android-sample-android.apk

Manage uploads

https://us-west-2.console.aws.amazon.com/devicefarm/home?region=us-west-2

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

parsingResult.txt Show all

This screenshot shows the same AWS Device Farm console interface, but the virtual device screen now displays an update notification for the 'flickrj-android-sample-a...' app. A red arrow points to the notification, which states: 'This app was built for an older version of Android and may not work properly. Try checking for updates, or contact the developer.' Below the notification are 'Check for update' and 'OK' buttons. The rest of the console interface, including the 'Device information' and 'Install applications' sections, remains the same.

aws Services

awsstudent @ 0827-5603-2383 Global Support

Notice

To download an app from Play Store, add your Google Account to the device. Once you do that, and switch to your account in Play Store, you will be able to see all apps in the Play Store. Note that AWS Device Farm captures video and logs of activity taking place during Remote Access session. It is recommended that you avoid entering your personal accounts on the device (E.g., personal Google account) and instead use test accounts where possible.

Device information

Name: Samsung Galaxy S10
OS: 9
Manufacturer: Samsung
Model: 9
Chipset: arm64-v8a 1785.6MHz
Memory: 12800000000
Heap: 512000000
Display: 1440 x 3040

Install applications

Upload Or drop your file here or

Select a recent upload ^

Flickrj-android-sample-a...

This app was built for an older version of Android and may not work properly. Try checking for updates, or contact the developer.

Check for update OK

Back Home Recent apps Landscape

Feedback English (US)

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

parsingResult.txt Show all

Check results

The screenshot shows the AWS Device Farm console. The 'Mobile device testing projects' tab is selected. A project card for 'Walter_Project1' is visible, showing a 'Last run' status with 15 PASSED, 0 FAILED, and 0 ERRORED tests. A red arrow points from the '+ Create a new project' button to the project card.

Detail logs – a/ Harness

The screenshot shows the 'Samsung Galaxy S10' details page in the AWS Device Farm console. The 'Logs' tab is selected. A red arrow points to the 'Download logs' button. Another red arrow points to the 'Source' dropdown menu, which is set to 'Harness'. A third red arrow points to the 'Message' column header.

Source	Time	PID ...	Level	Tag	Message
Harness	00:00.0	2309	Info		Starting 00000 with device R38M7015L1B
Harness	00:00.0	2309	Info		Using test content version 0.1.0
Harness	00:00.0	2309	Info		Using image version ami-08af882d622b6f33b
Harness	00:00.0	2309	Info		Using kpc version None
Harness	00:00.0	2309	Info		Using kpl version None
Harness	00:00.1	2309	Info		Using ktc version None
Harness	00:00.1	2309	Info		Using krl version None
Harness	00:00.1	2309	Info		Using krcc version None
Harness	00:00.1	2309	Info		Using kp3 version None
Harness	00:00.1	2309	Info		Starting Setup Suite

Detail logs – b/ Device

Samsung Galaxy S10

Details | Files

Name: Samsung Galaxy S10
Created: 2020-10-10T22:42:0700
Total minutes: 00:04:37

Logs [Download logs](#)

Source	Time	PID ...	Level	Tag	Message
Device	00:02:46	1247	Debug	WifiPermissionsUtil	canAccessScanResults: pkgName = android, uid = 1000
Device	00:02:274	2670	Debug	io_stats	l@ 8.0 r 48793 4399212 w 53525 1364892 d 9106 504788 f 19438 2882...
Device	00:02:390	700	Debug	Netd	notify() code: 613, msg: IfaceClass active 1 2617048408479 1000
Device	00:02:677	817	Info	SurfaceFlinger	SFWD update time=2617336634781
Device	00:03:51	1247	Debug	WifiTrafficPoller	TrafficStats TxPkts=6528 RxPkts=10765 TxBytes=887362 RxBytes=129...
Device	00:03:52	1247	Debug	WifiPermissionsUtil	canAccessScanResults: pkgName = android, uid = 1000
Device	00:03:56	1865	Debug	SecStatusBarWifiView	updateState: WifiIconState(resId=2131232604, visible=true, activityId=21...
Device	00:03:456	1247	Debug		mAFPC_Read - w = 1440, h = 3040, s = 32, f = 4, s_size = 17899520, lu...
Device	00:03:768	1247	Debug	WifiPermissionsUtil	canAccessScanResults: pkgName = android, uid = 1000
Device	00:04:61	1247	Debug	WifiPermissionsUtil	canAccessScanResults: pkgName = android, uid = 1000

Feedback English (US) © 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

log.json ce97f067-5a29-4...mp4 parsingResult.txt Show all X

Easy download of all test videos, logs and tcpdumps:

Samsung Galaxy S10

Details | Files

Files

- 1 Video
- 2 Logcat
- 3 TCP dump log

easy download of all test videos, logs and tcpdumps

53c0ee0-eed5-469...t... 3a5bf040-1be0-...log... ce97f067-5a29-4...mp4 Show all X

Pricing

1/ Pay as you go:

The screenshot shows the AWS Device Farm Pricing page. The top navigation bar includes links for Products, Solutions, Pricing, Documentation, Learn, Partner Network, AWS Marketplace, Customer Enablement, Events, and Explore. A 'Sign In to the Console' button is also present. Below the navigation bar, the 'Pricing' tab is selected. The main content area features two tabs: 'Testing on Real Mobile Devices' and 'Testing on Desktop Browsers'. The 'Testing on Desktop Browsers' tab is active, displaying a 'Pay as you go' pricing model. A red arrow points to the '\$0.005 / instance minute' price. Below this, a text box explains that users can test on any of their desktop browser instances in parallel and pay only for the total duration of the test. A 'Desktop Browser Testing Pricing Example' section is also visible.

Pay as you go

\$0.005 / instance minute

Test on any of our desktop browser instances in parallel and pay just for the total duration it takes for your test to execute. Test duration is calculated in minutes, from the time your tests start executing on the browser instance until your test is terminated.

Desktop Browser Testing Pricing Example

2/ unlimited pricing:

The screenshot shows the AWS Device Farm Unlimited pricing page. The top navigation bar includes links for Services, Device Farm, and a 'Buy Now' button. The main content area features a 'Start today with unlimited testing' heading. Below this, four pricing options are displayed: 1. SINGLE PLATFORM (\$500/month), 2. CROSS PLATFORM (\$1000/month), 3. CONTINUOUS INTEGRATION (\$1500/month), and 4. Or, create your own custom package. A red arrow points to the 'CONTINUOUS INTEGRATION' option. Each option includes a 'Buy Now' button.

Start today with unlimited testing

1 SINGLE PLATFORM \$500/month

2 CROSS PLATFORM \$1000/month

3 CONTINUOUS INTEGRATION \$1500/month

4 Or, create your own custom package

Allow Automated testings with CI:

Start today with unlimited testing

★ Popular choice

Plan	Price	Automated testing	Remote access	Buy Now
SINGLE PLATFORM	\$500/month	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Buy Now
CROSS PLATFORM	\$1000/month	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Buy Now
CONT...	\$...	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Buy Now

Automated testing allows you to run built-in or your own tests against devices in parallel with the number of concurrent sessions equal to the number of slots you've purchased.

Or, create your own custom package

Feedback English (US) © 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

parsingResult.txt Show all X

Create your own package:

Or, create your own custom package

Automated testing

1 device slots 1 device slots

Remote access

1 device slots 1 device slots

\$1000/month

Buy Now

Unlimited testing FAQ

- 1** What is the unmetered plan and how do device slots work?

Unmetered plans allow unlimited testing and remote access starting at \$250 per month. Unmetered pricing is based on the number of device slots you purchase for each usage type (i.e. automated test or remote access) and device family (i.e. Android or iOS) and are priced at \$250 per slot per month. Device slots correspond to concurrency.

For instance, if you purchase ten automated test Android device slots and schedule a run on 100 Android devices, Device Farm will execute your tests on up to ten devices at a time until all tests are completed on your selected devices. Purchasing more slots would enable you to get your results faster. Regardless of how many tests or remote access sessions you have in a month, you are billed at the flat rate of \$250 per device slot per month. You can cancel your subscription for one or more device slots at any time and the cancellation will take effect at your next renewal date (the day of the month that you purchased your first active device slot). If you have any questions, please [contact us](#).
- 2** What if my testing needs change and I need to add or remove device slots?

You can add device slots at any time and they will be available to you immediately. You can also cancel your subscription for one or more unmetered device slots at any time and the cancellation will take effect at your next renewal date (the day of the month that you purchased your first active device slot). If you have any questions, please [contact us](#).
- 3** If I'm on an unmetered plan, can I still make use of metered billing?


Yes. When creating a run, you can choose to make use of your unmetered device slots or use metered device minutes instead. Because concurrency is not limited on metered billing, this gives you the flexibility of running tests faster than would otherwise happen using your device slots.

Feedback English (US) © 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

parsingResult.txt Show all X

Testing on Real Mobile Devices Testing on Desktop Browsers

Pay as you go




Test on any of our devices in parallel and pay just for what you use

DURATION OF FREE TIER ACCESS

Your first 1000 minutes are free*

Unlimited testing



Test as much as you want each month for a flat rate

Private devices

STARTS AT \$200/MONTH

Test on dedicated devices deployed exclusively for your account

[Contact us »](#)

Waiting for a0.awsstatic.com...

53c0eee0-ecd5-469-...t... 3a5bf040-1be0-...logc... ce97f067-5a29-4...mp4 Show all

My account setting to show Free trial minutes:

Account settings

You have **956 FREE TRIAL MINUTES** remaining.

Device slots Device instances Instance profiles

Changes to device slots apply to your entire account and will affect all projects.

Purchase and manage device slots

Automated testing

Automated testing allows you to run built-in or your own tests against devices in parallel with concurrency equal to the number of slots you've purchased.

[Learn More »](#)

You currently have

0 slots

From November 11, you will have

0 slots

Remote access

Remote access allows you to manually interact with devices through your browser with the number of concurrent sessions equal to the number of slots you've purchased.

[Learn More »](#)

You currently have

0 slots

From November 11, you will have

0 slots

Save

Feedback English (US) © 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

53c0eee0-ecd5-469-...t... 3a5bf040-1be0-...logc... ce97f067-5a29-4...mp4 Show all

Can request your own device instances:

The screenshot displays the AWS Device Farm console interface. At the top, the navigation bar includes the AWS logo, 'Services', and user information. The main header shows 'Device Farm' and 'Mobile Device Projects'. The 'Account settings' section indicates '956 FREE TRIAL MINUTES' remaining. The 'Device instances' tab is selected, showing a 'Request a new device instance' button highlighted with a red arrow. Below this is a table with columns: Instance Id, Device, Platform, OS, Profile, Labels, and Status. The table is currently empty, and a message below it states: 'You have no private devices. Add a new one. Not sure what private devices are? Click here'. The footer shows the URL 'https://aws.amazon.com/device-farm/pricing/#privateDevic...' and copyright information.

Instance Id	Device	Platform	OS	Profile	Labels	Status
-------------	--------	----------	----	---------	--------	--------

You have no private devices. Add a new one. Not sure what private devices are? [Click here](#)

Docs, Videos, Demos:

Docs:

<https://docs.aws.amazon.com/devicefarm/latest/developerguide/welcome.html>

Videos:

Getting Started with AWS Device Farm

<https://youtu.be/ZWAUAerFB6s>

Git :

<https://github.com/aws-samples/aws-device-farm-sample-app-for-android>

<https://github.com/aws-samples/aws-device-farm-sample-app-for-ios>

sample test app at <https://code.google.com/archive/p/flickrj-android/downloads>

Built-in Fuzz Test

<https://docs.aws.amazon.com/devicefarm/latest/developerguide/test-types-built-in-fuzz.html>

Lab:

<https://www.qwiklabs.com/focuses/10377>

Sample test suites:

```
[
  {
    "name": "Setup Suite",
    "tests": [
      {
        "name": "Setup Test"
      }
    ]
  },
  {
    "name": "Built-in Fuzz Suite",
    "tests": [
      {
        "name": "Built-in Fuzz Test"
      }
    ]
  },
  {
    "name": "Teardown Suite",
    "tests": [
      {
        "name": "Teardown Test"
      }
    ]
  }
]
```

Sample logs:

1/ harness logs

D:\Downloads>more log.json

```
[{"attachment_id": null, "parent_id": null, "timestamp": "2020-10-11T05:42:46.570250Z", "level": "Info",
"pid": 2309, "data": "Starting 00000 with device R38M7015L1B", "source": "Harness", "tag": null,
"subtype": "Text", "tid": 2314, "type": "Message", "id": 3}, {"attachment_id": null, "parent_id": null,
"timestamp": "2020-10-11T05:42:46.570456Z", "level": "Info", "pid": 2309, "data": "Using test content
version 0.1.0", "source": "Harness", "tag": null, "subtype": "Text", "tid": 2314, "type": "Message", "id":
4}, {"attachment_id": null, "parent_id": null, "timestamp": "2020-10-11T05:42:46.570594Z", "level":
"Info", "pid": 2309, "data": "Using image version ami-08af882d622b6f33b", "source": "Harness", "tag":
null, "subtype": "Text", "tid": 2314, "type": "Message", "id": 5}, {"attachment_id": null, "parent_id": null,
"timestamp": "2020-10-11T05:42:46.570765Z", "level": "Info", "pid": 2309, "data": "Using kpc version
None", "source": "Harness", "tag": null, "subtype": "Text", "tid": 2314, "type": "Message", "id": 6},
{"attachment_id": null, "parent_id": null, "timestamp": "2020-10-11T05:42:46.570894Z", "level": "Info",
"pid": 2309, "data": "Using kpl version None", "source": "Harness", "tag": null, "subtype": "Text", "tid":
2314, "type": "Message", "id": 7},...
```


2/ logcat:

----- beginning of main

10-11 16:43:04.880 1247 1653 D WifiPermissionsUtil: canAccessScanResults: pkgName = android, uid = 1000

10-11 16:43:05.412 27141 27155 I android.dqagen: Waiting for a blocking GC ProfileSaver

10-11 16:43:05.420 27141 27155 I android.dqagen: WaitForGcToComplete blocked ProfileSaver on ClassLinker for 7.703ms

10-11 16:43:05.767 24467 27193 D Aatp : /default-rotation

10-11 16:43:05.774 24467 27193 D Aatp : sha1: afbe92eb910a5b8c2b798f35cbfa6dbc1f2580ce

10-11 16:43:05.783 24467 24467 D AatpDeviceManager: taskStatus update
(afbe92eb910a5b8c2b798f35cbfa6dbc1f2580ce: {"state":1})

10-11 16:43:05.788 24467 24467 D AatpDeviceManager: taskStatus update
(afbe92eb910a5b8c2b798f35cbfa6dbc1f2580ce: {"result":{"rotation":0},"state":2})

10-11 16:43:05.793 24467 27193 D Aatp : /task/afbe92eb910a5b8c2b798f35cbfa6dbc1f2580ce

10-11 16:43:05.884 1247 1653 D WifiPermissionsUtil: canAccessScanResults: pkgName = android, uid = 1000

10-11 16:43:05.929 24467 27195 D Aatp : /task/afbe92eb910a5b8c2b798f35cbfa6dbc1f2580ce/status

10-11 16:43:06.263 24467 27200 D Aatp : /default-rotation

10-11 16:43:06.269 24467 27200 D Aatp : sha1: 60b7b0b120c87106df913cee5f41cec9b55d3242

10-11 16:43:06.279 24467 24467 D AatpDeviceManager: taskStatus update
(60b7b0b120c87106df913cee5f41cec9b55d3242: {"state":1})

10-11 16:43:06.281 24467 24467 D AatpDeviceManager: taskStatus update
(60b7b0b120c87106df913cee5f41cec9b55d3242: {"result":{"rotation":0},"state":2})

10-11 16:43:06.291 24467 27200 D Aatp : /task/60b7b0b120c87106df913cee5f41cec9b55d3242

10-11 16:43:06.425 24467 27202 D Aatp :
/task/60b7b0b120c87106df913cee5f41cec9b55d3242/status

10-11 16:43:06.524 1247 1654 D WifiStateMachine: enter getWifiLinkLayerStats

10-11 16:43:06.525 1247 1654 I WifiVendorHal: getWifiLinkLayerStats(l.2973) before calling
iface.getLinkLayerStats

10-11 16:43:06.561 1247 1654 I WifiVendorHal: getWifiLinkLayerStats(l.2973) after calling
iface.getLinkLayerStats

10-11 16:43:06.580 27203 27203 D AndroidRuntime: >>>>> START com.android.internal.os.RuntimeInit
uid 2000 <<<<<

3/ tcpdumps:

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on wlan0, link-type EN10MB (Ethernet), capture size 262144 bytes

05:44:42.277695 IP ip-192-168-1-100.us-west-2.compute.internal.53088 > ec2-52-45-155-205.compute-1.amazonaws.com.https: Flags [S], seq 2127107027, win 65535, options [mss 1460,sackOK,TS val 377166308 ecr 0,nop,wscale 8], length 0

05:44:42.366521 IP ec2-52-45-155-205.compute-1.amazonaws.com.https > ip-192-168-1-100.us-west-2.compute.internal.53088: Flags [S.], seq 1687700965, ack 2127107028, win 26847, options [mss 1360,sackOK,TS val 1768108349 ecr 377166308,nop,wscale 8], length 0

05:44:42.369072 IP ip-192-168-1-100.us-west-2.compute.internal.53088 > ec2-52-45-155-205.compute-1.amazonaws.com.https: Flags [.], ack 1, win 343, options [nop,nop,TS val 377166399 ecr 1768108349], length 0

05:44:42.372104 IP ip-192-168-1-100.us-west-2.compute.internal.53088 > ec2-52-45-155-205.compute-1.amazonaws.com.https: Flags [P.], seq 1:187, ack 1, win 343, options [nop,nop,TS val 377166402 ecr 1768108349], length 186

05:44:42.424318 IP ip-192-168-1-100.us-west-2.compute.internal.57496 > sea15s11-in-f3.1e100.net.https: Flags [S], seq 972138972, win 65535, options [mss 1460,sackOK,TS val 1559468470 ecr 0,nop,wscale 8], length 0

05:44:42.439219 IP sea15s11-in-f3.1e100.net.https > ip-192-168-1-100.us-west-2.compute.internal.57496: Flags [S.], seq 2618646787, ack 972138973, win 60192, options [mss 1360,sackOK,TS val 328146633 ecr 1559468470,nop,wscale 8], length 0

05:44:42.441664 IP ip-192-168-1-100.us-west-2.compute.internal.57496 > sea15s11-in-f3.1e100.net.https: Flags [.], ack 1, win 343, options [nop,nop,TS val 1559468487 ecr 328146633], length 0

05:44:42.444427 IP ip-192-168-1-100.us-west-2.compute.internal.57496 > sea15s11-in-f3.1e100.net.https: Flags [P.], seq 1:518, ack 1, win 343, options [nop,nop,TS val 1559468490 ecr 328146633], length 517

05:44:42.459166 IP sea15s11-in-f3.1e100.net.https > ip-192-168-1-100.us-west-2.compute.internal.57496: Flags [.], ack 518, win 240, options [nop,nop,TS val 328146654 ecr 1559468490], length 0

05:44:42.460771 IP ec2-52-45-155-205.compute-1.amazonaws.com.https >

ip-192-168-1-100.us-west-2.compute.internal.53088: Flags [.], ack 187, win 110, options [nop,nop,TS val 1768108443 ecr 377166402], length 0

05:44:42.461079 IP ec2-52-45-155-205.compute-1.amazonaws.com.https > ip-192-168-1-100.us-west-2.compute.internal.53088: Flags [.], seq 1:1349, ack 187, win 110, options [nop,nop,TS val 1768108443 ecr 377166402], length 1348