UI Automator

UI Automator is a UI testing framework suitable for cross-app functional UI testing across system and installed apps.

★ **Note:** This framework requires Android 4.3 (API level 18) or higher.

The UI Automator testing framework provides a set of APIs to build UI tests that perform interactions on user apps and system apps. The UI Automator APIs allows you to perform operations such as opening the Settings menu or the app launcher in a test device. The UI Automator testing framework is well-suited for writing black box-style automated tests, where the test code does not rely on internal implementation details of the target app.

The key features of the UI Automator testing framework include the following:

- A viewer to inspect layout hierarchy. For more information, see UI Automator Viewer (#ui-automator-viewer).
- An API to retrieve state information and perform operations on the target device. For more information, see Accessing device state (#accessing-device-state).
- APIs that support cross-app UI testing. For more information, see UI Automator APIs (#ui-automator-apis).

UI Automator viewer

The `uiautomatorviewer` tool provides a convenient GUI to scan and analyze the UI components currently displayed on an Android device. You can use this tool to inspect the layout hierarchy and view the properties of UI components that are visible on the foreground of the device. This information lets you create more fine-grained tests using UI Automator, for example by creating a UI selector that matches a specific visible property.

The `uiautomatorviewer` tool is located in the `<android-sdk>/tools/bin` directory.

Accessing device state

The UI Automator testing framework provides a `UiDevice` (https://developer.android.com/reference/androidx/test/uiautomator/UiDevice) class to access and perform operations on the device on which the target app is running. You can call its methods to access device properties such as current orientation or display size. The `UiDevice` (https://developer.android.com/reference/androidx/test/uiautomator/UiDevice) class also let you perform actions such as:

- Change the device rotation.
- Press a hardware key, such as "volume up".
- Press the Back, Home, or Menu buttons.
- Open the notification shade.
- Take a screenshot of the current window.

For example, to simulate a Home button press, call the `UiDevice.pressHome()` method.

UI Automator APIs

The UI Automator APIs allow you to write robust tests without needing to know about the implementation details of the app that you are targeting. You can use these APIs to capture and manipulate UI components across multiple apps:

- **UiCollection** (https://developer.android.com/reference/androidx/test/uiautomator/UiCollection): Enumerates a container's UI elements for the purpose of counting, or targeting sub-elements by their visible text or content-description property.
- **UiObject** (https://developer.android.com/reference/androidx/test/uiautomator/UiObject.html): Represents a UI element that is visible on the device.
- **UiScrollable** (https://developer.android.com/reference/androidx/test/uiautomator/UiScrollable.html): Provides support for searching for items in a scrollable UI container.
- **UiSelector** (https://developer.android.com/reference/androidx/test/uiautomator/UiSelector.html): Represents a query for one or more target UI elements on a device.
- **Configurator** (https://developer.android.com/reference/androidx/test/uiautomator/Configurator.html): Allows you to set key parameters for running UI Automator tests.

For example, the following code shows how you can write a test script that displays the default app launcher in the device:

KOTLIN (#KOTLIN)JAVA

```
device = UiDevice.getInstance(getInstrumentation());
device.pressHome();

// Bring up the default launcher by searching for a UI component
// that matches the content description for the launcher button.
UiObject allAppsButton = device
        .findObject(new UiSelector().description("Apps"));

// Perform a click on the button to load the launcher.
allAppsButton.clickAndWaitForNewWindow();
```

To learn more about using UI Automator, see the API reference (https://developer.android.com/reference/androidx/test/package-summary.html) and the Test UI for multiple apps (https://developer.android.com/training/testing/ui-testing/uiautomator-testing.html) guide.

Additional resources

For more information about using UI Automator in Android tests, consult the following resources.

Samples

- BasicSample (https://github.com/googlesamples/android-testing/tree/master/ui/uiautomator/BasicSample): Basic UI Automator sample.