Appium Desired Capabilities

Edit this Doc (https://github.com/appium/appium/edit/master/docs/en/writing-running-appium/caps.md)

Desired Capabilities are keys and values encoded in a JSON object, sent by Appium clients to the server when a new automation session is requested. They tell the Appium drivers all kinds of important things about how you want your test to work. Each Appium client builds capabilities in a way specific to the client's language, but at the end of the day, they are sent over to Appium as JSON objects.

Desired Capabilities can be scripted in the WebDriver test or set within the Appium Server GUI (via an Inspector Session)

Some important capabilities are demonstrated in the following example:

```
{
    "platformName": "iOS",
    "platformVersion": "11.0",
    "deviceName": "iPhone 7",
    "automationName": "XCUITest",
    "app": "/path/to/my.app"
}
```

This set of Desired Capabilities expresses the desire for Appium to begin an automation session on an iPhone 7 simulator with iOS 11, using the XCUITest Driver (/docs/en/drivers/ios-xcuitest/index.html), with `/path/to/my.app` as the app under test.

There are many, many Capabilities that Appium supports. Capabilities also differ by driver, though there are a standard set that most drivers pay attention to. What follows are a series of tables outlining the various Desired Capabilities available in general and for specific drivers.

General Capabilities

These Capabilities span multiple drivers.

Capability	Description	Values
`automationName`	Which automation engine to use	`Appium` (default), or `UiAutomator2` , `Espresso` , or `UiAutomator1` for Android, or `XCUITest` or `Instruments` for iOS, or `YouiEngine` for application built with You.i Engine
`platformName`	Which mobile OS platform to use	`iOS` , `Android` , or `FirefoxOS`
`platformVersion`	Mobile OS version	e.g., `7.1` , `4.4`
`deviceName`	The kind of mobile device or emulator to use	`iPhone Simulator` , `iPad Simulator` , `iPhone Retina 4-inch` , `Android Emulator` , `Galaxy S4` , etc.... On iOS, this should be one of the valid devices returned by instruments with `instruments -s devices` . On Android this capability is currently ignored, though it remains required.
`app`	The absolute local path *or* remote http URL to a `.ipa` file (IOS), `.app` folder (IOS Simulator), `.apk` file (Android) or `.apks` file (Android App Bundle), or a `.zip` file containing one of these. Appium will attempt to install this app binary on the appropriate device first. Note that this capability is not required for Android if you specify `appPackage` and `appActivity` capabilities (see below). `UiAutomator2` and `XCUITest` allow to start the session without `app` or `appPackage` . Incompatible with `browserName` . See here (/docs/en/writing-running-appium/android/android-appbundle/index.html) about `.apks` file.	`/abs/path/to/my.apk` or `http://myapp.com/app.ipa`
`otherApps`	App or list of apps (as a JSON array) to install prior to running tests. Note that it will not work with `automationName` of `Espresso` and iOS real devices	e.g., `"/path/to/app.apk"` , `https://www.example.com/url/to/app.apk` , `["http://appium.github.io/appium/assets/TestApp9.4.app.zip", "/path/to/app-b.app"]`
`browserName`	Name of mobile web browser to automate. Should be an empty string if automating an app instead.	'Safari' for iOS and 'Chrome', 'Chromium', or 'Browser' for Android
`newCommandTimeout`	How long (in seconds) Appium will wait for a new command from the client before assuming the client quit and ending the session	e.g. `60`
`language`	Language to set for iOS (XCUITest driver only) and Android.	e.g. `fr`

Capability	Description	Values
`locale`	Locale to set for iOS (XCUITest driver only) and Android. `fr_CA` format for iOS. `CA` format (country name abbreviation) for Android	e.g. `fr_CA` , `CA`
`udid`	Unique device identifier of the connected physical device	e.g. `1ae203187fc012g`
`orientation`	(Sim/Emu-only) start in a certain orientation	`LANDSCAPE` or `PORTRAIT`
`autoWebview`	Move directly into Webview context. Default `false`	`true` , `false`
`noReset`	Don't reset app state before this session. See here (/docs/en/writing-running-appium/other/reset-strategies/index.html) for more details	`true` , `false`
`fullReset`	Perform a complete reset. See here (/docs/en/writing-running-appium/other/reset-strategies/index.html) for more details	`true` , `false`
`eventTimings`	Enable or disable the reporting of the timings for various Appium-internal events (e.g., the start and end of each command, etc.). Defaults to `false` . To enable, use `true` . The timings are then reported as `events` property on response to querying the current session. See the event timing docs (/docs/en/advanced-concepts/event-timings/index.html) for the the structure of this response.	e.g., `true`
`enablePerformanceLogging`	(Web and webview only) Enable Chromedriver's (on Android) or Safari's (on iOS) performance logging (default `false`)	`true` , `false`
`printPageSourceOnFindFailure`	When a find operation fails, print the current page source. Defaults to `false` .	e.g., `true`
`clearSystemFiles`	Delete any generated files at the end of a session. Default to `false` .	`true` , `false`

- Update settings

Capability	Description	Values
`settings[settingsKey]`	Update Appium Settings (https://github.com/appium/appium/blob/master/docs/en/advanced-concepts/settings.md) on session creation.	e.g., `'settings[mjpegScalingFactor]': 10` , `'settings[shouldUseCompactResponses]': true`

Android Only

These Capabilities are available only on Android-based drivers (like UiAutomator2 (/docs/en/drivers/android-uiautomator2/index.html) for example).

Capability	Description	Values
`appActivity`	Activity name for the Android activity you want to launch from your package. This often needs to be preceded by a `.` (e.g., `.MainActivity` instead of `MainActivity`). By default this capability is received from the package manifest (action: android.intent.action.MAIN , category: android.intent.category.LAUNCHER)	`MainActivity` , `.Settings`
`appPackage`	Java package of the Android app you want to run. By default this capability is received from the package manifest (@package attribute value)	`com.example.android.myApp` , `com.android.settings`
`appWaitActivity`	Activity name/names, comma separated, for the Android activity you want to wait for. By default the value of this capability is the same as for `appActivity` . You must set it to the very first focused application activity name in case it is different from the one which is set as `appActivity` if your capability has `appActivity` and `appPackage` . You can also use wildcards (`*`).	`SplashActivity` , `SplashActivity,OtherActivity` , `*` , `*.SplashActivity`
`appWaitPackage`	Java package of the Android app you want to wait for. By default the value of this capability is the same as for `appActivity`	`com.example.android.myApp` , `com.android.settings`
`appWaitDuration`	Timeout in milliseconds used to wait for the appWaitActivity to launch (default `20000`)	`30000`
`deviceReadyTimeout`	Timeout in seconds while waiting for device to become ready	`5`
`allowTestPackages`	Allow to install a test package which has `android:testOnly="true"` in the manifest. `false` by default	`true` or `false`
`androidCoverage`	Fully qualified instrumentation class. Passed to -w in adb shell am instrument -e coverage true -w	`com.my.Pkg/com.my.Pkg.instrumentation.MyInstrumentation`

Capability	Description	Values
`androidCoverageEndIntent`	A broadcast action implemented by yourself which is used to dump coverage into file system. Passed to -a in adb shell am broadcast -a	`com.example.pkg.END_EMMA`
`androidDeviceReadyTimeout`	Timeout in seconds used to wait for a device to become ready after booting	e.g., `30`
`androidInstallTimeout`	Timeout in milliseconds used to wait for an apk to install to the device. Defaults to `90000`	e.g., `90000`
`androidInstallPath`	The name of the directory on the device in which the apk will be push before install. Defaults to `/data/local/tmp`	e.g. `/sdcard/Downloads/`
`adbPort`	Port used to connect to the ADB server (default `5037`)	`5037`
`systemPort`	`systemPort` used to connect to appium-uiautomator2-server (https://github.com/appium/appium-uiautomator2-server) or appium-espresso-driver (https://github.com/appium/appium-espresso-driver). The default is `8200` in general and selects one port from `8200` to `8299` for *appium-uiautomator2-server*, it is `8300` from `8300` to `8399` for *appium-espresso-driver*. When you run tests in parallel, you must adjust the port to avoid conflicts. Read Parallel Testing Setup Guide (https://github.com/appium/appium/blob/master/docs/en/advanced-concepts/parallel-tests.md#parallel-android-tests) for more details.	e.g., `8201`
`remoteAdbHost`	Optional remote ADB server host	e.g.: 192.168.0.101
`androidDeviceSocket`	Devtools socket name. Needed only when tested app is a Chromium embedding browser. The socket is open by the browser and Chromedriver connects to it as a devtools client.	e.g., `chrome_devtools_remote`
`avd`	Name of avd to launch	e.g., `api19`
`avdLaunchTimeout`	How long to wait in milliseconds for an avd to launch and connect to ADB (default `60000`)	`300000`
`avdReadyTimeout`	How long to wait in milliseconds for an avd to finish its boot animations (default `120000`)	`300000`
`avdArgs`	Additional emulator arguments used when launching an avd	e.g., `-netfast`
`useKeystore`	Use a custom keystore to sign apks, default `false`	`true` or `false`
`keystorePath`	Path to custom keystore, default ~/.android/debug.keystore	e.g., `/path/to.keystore`
`keystorePassword`	Password for custom keystore	e.g., `foo`
`keyAlias`	Alias for key	e.g., `androiddebugkey`
`keyPassword`	Password for key	e.g., `foo`
`chromedriverExecutable`	The absolute local path to webdriver executable (if Chromium embedder provides its own webdriver, it should be used instead of original chromedriver bundled with Appium)	`/abs/path/to/webdriver`
`chromedriverArgs`	An array of arguments to be passed to the chromedriver binary when it's run by Appium. By default no CLI args are added beyond what Appium uses internally (such as `--url-base` , `--port` , `--adb-port` , and `--log-path` .	e.g., `["--disable-gpu", "--disable-web-security"]`
`chromedriverExecutableDir`	The absolute path to a directory to look for Chromedriver executables in, for automatic discovery of compatible Chromedrivers. Ignored if `chromedriverUseSystemExecutable` is `true`	`/abs/path/to/chromedriver/directory`
`chromedriverChromeMappingFile`	The absolute path to a file which maps Chromedriver versions to the minimum Chrome that it supports. Ignored if `chromedriverUseSystemExecutable` is `true`	`/abs/path/to/mapping.json`
`chromedriverUseSystemExecutable`	If `true` , bypasses automatic Chromedriver configuration and uses the version that comes downloaded with Appium. Ignored if `chromedriverExecutable` is set. Defaults to `false`	e.g., `true`
`autoWebviewTimeout`	Amount of time to wait for Webview context to become active, in ms. Defaults to `2000`	e.g. `4`
`chromedriverPort`	Numeric port to start Chromedriver on. Note that use of this capability is discouraged as it will cause undefined behavior in case there are multiple webviews present. By default Appium will find a free port.	e.g. `8000`

Capability	Description	Values
`chromedriverPorts`	A list of valid ports for Appium to use for communication with Chromedrivers. This capability supports multiple webview scenarios. The form of this capability is an array of numeric ports, where array items can themselves be arrays of length 2, where the first element is the start of an inclusive range and the second is the end. By default, Appium will use any free port.	e.g. `[8000, [9000, 9005]]`
`ensureWebviewsHavePages`	Whether or not Appium should augment its webview detection with page detection, guaranteeing that any webview contexts which show up in the context list have active pages. This can prevent an error if a context is selected where Chromedriver cannot find any pages. Defaults to `false`	e.g. `true`
`webviewDevtoolsPort`	To support the `ensureWebviewsHavePages` feature, it is necessary to open a TCP port for communication with the webview on the device under test. This capability allows overriding of the default port of `9222` , in case multiple sessions are running simultaneously (to avoid port clash), or in case the default port is not appropriate for your system.	e.g. `9543`
`enableWebviewDetailsCollection`	Enables collection of detailed WebView information via `/json/version` CDP (Chrome Developer Protocol) endpoint since Appium 1.18.0+. This helps to properly match Chromedriver version which supports the given WebView. Without this flag enabled, Appium tries to guess the version of the WebView based on the version of the corresponding installed package (which usually fails (https://github.com/appium/appium/issues/13918) for custom web views). Defaults to `false`	`true` or `false`
`dontStopAppOnReset`	Doesn't stop the process of the app under test, before starting the app using adb. If the app under test is created by another anchor app, setting this false, allows the process of the anchor app to be still alive, during the start of the test app using adb. In other words, with `dontStopAppOnReset` set to `true` , we will not include the `-S` flag in the `adb shell am start` call. With this capability omitted or set to `false` , we include the `-S` flag. Default `false`	`true` or `false`
`unicodeKeyboard`	Enable Unicode input, default `false`	`true` or `false`
`resetKeyboard`	Reset keyboard to its original state, after running Unicode tests with `unicodeKeyboard` capability. Ignored if used alone. Default `false`	`true` or `false`
`noSign`	Skip checking and signing of app with debug keys, will work only with UiAutomator, default `false`	`true` or `false`
`ignoreUnimportantViews`	Calls the `setCompressedLayoutHierarchy()` uiautomator function. This capability can speed up test execution, since Accessibility commands will run faster ignoring some elements. The ignored elements will not be findable, which is why this capability has also been implemented as a toggle-able *setting* as well as a capability. Defaults to `false`	`true` or `false`
`disableAndroidWatchers`	Disables android watchers that watch for application not responding and application crash, this will reduce cpu usage on android device/emulator. This capability will work only with UiAutomator, default `false`	`true` or `false`
`chromeOptions`	Allows passing chromeOptions capability for ChromeDriver. For more information see chromeOptions (https://sites.google.com/a/chromium.org/chromedriver/capabilities)	`chromeOptions: {args: ['--disable-popup-blocking']}`
`recreateChromeDriverSessions`	Kill ChromeDriver session when moving to a non-ChromeDriver webview. Defaults to `false`	`true` or `false`
`nativeWebScreenshot`	In a web context, use native (adb) method for taking a screenshot, rather than proxying to ChromeDriver. Defaults to `false`	`true` or `false`
`androidScreenshotPath`	The name of the directory on the device in which the screenshot will be put. Defaults to `/data/local/tmp`	e.g. `/sdcard/screenshots/`
`autoGrantPermissions`	Have Appium automatically determine which permissions your app requires and grant them to the app on install. Defaults to `false` . If `noReset` is `true` , this capability doesn't work.	`true` or `false`
`networkSpeed`	Set the network speed emulation. Specify the maximum network upload and download speeds. Defaults to `full`	`['full','gsm', 'edge', 'hscsd', 'gprs', 'umts', 'hsdpa', 'lte', 'evdo']` Check -netspeed option (https://developer.android.com/studio/run/emulator-commandline.html) more info about speed emulation for avds
`gpsEnabled`	Toggle gps location provider for emulators before starting the session. By default the emulator will have this option enabled or not according to how it has been provisioned.	`true` or `false`

Capability	Description	Values
`isHeadless`	Set this capability to `true` to run the Emulator headless when device display is not needed to be visible. `false` is the default value. *isHeadless* is also support for iOS, check XCUITest-specific capabilities.	e.g., `true`
`adbExecTimeout`	Timeout in milliseconds used to wait for adb command execution. Defaults to `20000`	e.g., `50000`
`localeScript`	Sets the locale script (https://developer.android.com/reference/java/util/Locale)	e.g., `"Cyrl"` (Cyrillic)
`skipDeviceInitialization`	Skip device initialization which includes i.a.: installation and running of Settings app or setting of permissions. Can be used to improve startup performance when the device was already used for automation and it's prepared for the next automation. Defaults to `false`	`true` or `false`
`chromedriverDisableBuildCheck`	Sets the chromedriver flag `--disable-build-check` for Chrome webview tests	`true` or `false`
`skipUnlock`	Skips unlock during session creation. Defaults to `false`	`true` or `false`
`unlockType`	Unlock the target device with particular lock pattern instead of just waking up the device with a helper app. It works with `unlockKey` capability. Defaults to undefined. `fingerprint` is available only for Android 6.0+ and emulators. Read unlock doc (https://github.com/appium/appium-android-driver/blob/master/docs/UNLOCK.md) in android driver.	`['pin', 'password', 'pattern', 'fingerprint']`
`unlockKey`	A key pattern to unlock used by `unlockType`.	e.g., '1111'
`autoLaunch`	Initializing the app under test automatically. Appium does not install/launch the app under test if this is `false`. Defaults to `true`	`true` or `false`
`skipLogcatCapture`	Skips to start capturing logcat. It might improve performance such as network. Log related commands will not work. Defaults to `false`.	`true` or `false`
`uninstallOtherPackages`	A package, list of packages or `*` to uninstall package/s before installing apks for test. `'*'` uninstall all of thrid-party packages except for packages which is necessary for Appium to test such as `io.appium.settings` or `io.appium.uiautomator2.server` since Appium already contains the logic to manage them.	e.g. `"io.appium.example"`, `["io.appium.example1", "io.appium.example2"]`, `'*'`
`disableWindowAnimation`	Set device animation scale zero if the value is `true`. After session is complete, Appium restores the animation scale to it's original value. Defaults to `false`	`true`, `false`
`remoteAppsCacheLimit`	Set the maximum number of remote cached apks (default is 10) which are pushed to the device-under-test's local storage. Caching apks remotely speeds up the execution of sequential test cases, when using the same set of apks, by avoiding the need to be push an apk to the remote file system every time a reinstall is needed. Set this capability to `0` to disable caching.	e.g. `0`, `5`, `20`
`buildToolsVersion`	Specify the Android `build-tools` version to be something different than the default, which is to use the most recent version. It is helpful to use a non-default version if your environment uses alpha/beta build tools.	e.g. `'28.0.3'`
`androidNaturalOrientation`	Allow for correct handling of orientation on landscape-oriented devices. Set to `true` to basically flip the meaning of `PORTRAIT` and `LANDSCAPE`. Defaults to `false`	`true`, `false`
`enforceAppInstall`	By default application installation is skipped if newer or the same version of this app is already present on the device under test. Setting this option to `true` will enforce Appium to always install the current application build independently of the currently installed version of it. Defaults to `false`.	`true` , `false`
`ignoreHiddenApiPolicyError`	Ignores `Security exception: Permission denial` alert and allows to continue the session creation process since Appium 1.18.0+. The error happens when Appium tries to relax hidden API policy (https://developer.android.com/distribute/best-practices/develop/restrictions-non-sdk-interfaces#how_can_i_enable_access_to_non-sdk_interfaces), although some devices with a customized firmware deny such requests. Defaults to `false`.	`true`, `false`
`mockLocationApp`	Sets the package identifier of the app, which is used as a system mock location provider since Appium 1.18.0+. This capability has no effect on emulators. If the value is set to `null` or an empty string, then Appium will skip the mocked location provider setup procedure. Defaults to Appium Setting package identifier (`io.appium.settings`).	e.g., `null`, `io.appium.settings`, `example.your.app`

Capability	Description	Values
logcatFormat	Set the output format for logcat messages since Appium 1.18.0. Supported formats are listed in here (https://github.com/appium/appium-adb/blob/master/lib/logcat.js). Please read logcat#outputFormat (https://developer.android.com/studio/command-line/logcat#outputFormat) for more details about each format. Defaults to `threadtime` .	e.g., `process`
logcatFilterSpecs	Set the output filter rule for logcat messages since Appium 1.18.0. Please read logcat#filteringOutput (https://developer.android.com/studio/command-line/logcat#filteringOutput) for more details about the rule. Write and View Logs with Logcat (https://developer.android.com/studio/debug/am-logcat) is also helpful.	e.g., `['*:W', 'MyActivity:D']` (`MyActivity` is a tag)
allowDelayAdb	Whether enable `-delay-adb` on emulator startup. Defaults to `true`	`true` , `false`

UIAutomator (1 & 2)

These Capabilities are available on UIA 1 and 2

Capability	Description	Values
`intentAction`	Intent action which will be used to start activity (default `android.intent.action.MAIN`)	e.g. `android.intent.action.MAIN` , `android.intent.action.VIEW`
`intentCategory`	Intent category which will be used to start activity (default `android.intent.category.LAUNCHER`)	e.g. `android.intent.category.LAUNCHER` , `android.intent.category.APP_CONTACTS`
`intentFlags`	Flags that will be used to start activity (default `0x10200000`)	e.g. `0x10200000`
`optionalIntentArguments`	Additional intent arguments that will be used to start activity. See Intent arguments (http://developer.android.com/reference/android/content/Intent.html)	e.g. `--esn <EXTRA_KEY>` , `--ez <EXTRA_KEY> <EXTRA_BOOLEAN_VALUE>` , etc.

UIAutomator2 Only

These Capabilities are available only on the UiAutomator2 Driver (/docs/en/drivers/android-uiautomator2/index.html)

Capability	Description	Values
`appWaitForLaunch`	Tries to launch the app under test without -W (https://developer.android.com/studio/command-line/adb#am) option in session creation. It might help when the session creation does not proceed since `shell am start` does not respond. Defaults to `true` .	`false` or `true`
`disableSuppressAccessibilityService`	Set FLAG_DONT_SUPPRESS_ACCESSIBILITY_SERVICES (https://developer.android.com/reference/android/app/UiAutomation#FLAG_DONT_SUPPRESS_ACCESSIBILITY_SERVICES) to allow existing accessibility service continue to run, and a new one may start for Appium. It helps to test the app under test which has accessibility feature such as TalkBack. Appium will not specify the flag if nothing is provided. The flag requires Android API Level 24+.	`false` or `true`
`mjpegServerPort`	If specified this is the local port that will be bound to the `appium-uiautomator2-server` 's MJPEG screenshot stream. This can be used in conjunction with `mjpegScreenshotUrl` . It should be a valid integer in range *1025..65535*. Defaults to `null` . e.g. `mjpegScreenshotUrl = 'http://localhost:9200', mjpegServerPort = 9200`	any `Integer` , recommended: `9200..9299` for consistency w/ `serverPort` range
`skipServerInstallation`	Skip uiAutomator2 server installation and use uiAutomator2 server from the device. Can be used to improve startup performance when an uiAutomator2 server in proper version is already installed on the device. Defaults to `false` .	`false` or `true`
`uiautomator2ServerInstallTimeout`	Timeout in milliseconds used to wait for an uiAutomator2 server to be installed. Defaults to `20000`	e.g., `20000`
`uiautomator2ServerLaunchTimeout`	Timeout in milliseconds used to wait for an uiAutomator2 server to launch. Defaults to `20000`	e.g., `20000`
`userProfile`	Enforce user profile as the given parameter if the value was provided. It should be an integer.	e.g., `11`

Espresso Only

These Capabilities are available only on the Espresso Driver (/docs/en/drivers/android-espresso/index.html)

Capability	Description	Values
`espressoServerLaunchTimeout`	Timeout in milliseconds used to wait for the Espresso server to launch. Defaults to `30000`	e.g., `50000`
`espressoBuildConfig`	Path to the Espresso server build configuration JSON (see below) or a stringified JSON itself (only supported since server version 1.19)	e.g., `/projects/myapp-tests/bu`
`showGradleLog`	Whether to pipe Gradle build log for the Espresso server to the Appium log. Defaults to `false`	e.g., `true`

Capability	Description	Values
`skipServerInstallation`	Skip Espresso server build and apk installation. This option could break proper Espresso server setup for the particular Appium version, but it can improve startup performance when the proper Espresso server and the proper app under test are already installed on the device. Please, make sure not to enable this option if the Espresso server or the application under test needs an update. Defaults to `false`	`true` or `false`
`intentOptions`	Intent options which will be used to start the application under test. It can set intent options such as `action`, `categories` and `component` as JSON format. Please read #538 (https://github.com/appium/appium-espresso-driver/issues/538) as an example usage.	e.g. `{"action": "android.intent.ac`
`disableSuppressAccessibilityService`	Set FLAG_DONT_SUPPRESS_ACCESSIBILITY_SERVICES (https://developer.android.com/reference/android/app/UiAutomation#FLAG_DONT_SUPPRESS_ACCESSIBILITY_SERVICES) to allow existing accessibility service continue to run, and a new one may start for Appium. It helps to test the app under test which has accessibility feature such as TalkBack. Appium will not specify the flag if nothing is provided. The flag requires Android API Level 24+.	`true`, `false`
`appLocale`	Set Locale (https://developer.android.com/reference/java/util/Locale) for the target context (https://developer.android.com/reference/androidx/test/core/app/ApplicationProvider#getApplicationContext()) since Appium 1.18.0. `language`, `country` and `variant` are available as same as the Locale class. `language` is mandatory. Please check the locale page (https://developer.android.com/reference/java/util/Locale) for more details. Setting `appLocale` only affects the locale of the application under test and does not influence other system views, such as the status bar. Consider setting `language`, `locale` and/or `localeScript` capabilities if you would like to change the locale for the entire the system.	e.g., `{"language": "ja", "coun`

Espresso server build configuration JSON

Passing this configuration file using `espressoBuildConfig` desired capability allows to fine-tune the build process of the Espresso server. It is mostly useful in cases where the default Espresso server settings are not compatible with your application under test. One example of such a case is tests crashing due to `Resource <name> is not a Drawable` error (see https://github.com/appium/appium-espresso-driver/issues/449 for discussion).

Configuration example:

```
{
  "toolsVersions": {
    "gradle": "5.1.1",
    "androidGradlePlugin": "3.4.2",
    "compileSdk": 28,
    "buildTools": "28.0.3",
    "minSdk": 18,
    "targetSdk": 28,
    "kotlin": "1.3.31"
  },
  "additionalAppDependencies": [
    "com.google.android.material:material:1.0.0"
  ]
}
```

Version settings

`toolsVersion` specifies versions of various tools and SDKs used during the building process of the Espresso server. Default versions are the versions used to build the Espresso driver without build configuration JSON.

The module versions enumerated under `toolsVersion` are only used to build the server APK. They don't affect the manifest of your application under test or the Espresso server manifest (that is still generated from the manifest of your application under test).

Setting	Description	Values
`gradle`	Gradle version	e.g., `5.1.1`
`androidGradlePlugin`	Android Gradle Plugin version	e.g., `3.4.2`
`buildTools`	Version of the Android SDK build tools that should be used to compile the Espresso server (corresponds to `buildToolsVersion` in Gradle build files)	e.g., `28.0.3`
`compileSdk`	Android API level that should be used to compile the Espresso server (corresponds to `compileSdkVersion` in Gradle build files)	e.g., `28`
`minSdk`	Minimum Android API level required to run the application under tests, affects compatibility libraries used to build the Espresso server (corresponds to `minSdk` in Gradle build files)	e.g., `18`
`targetSdk`	Android API level used to test the app (corresponds to `targetSdk` in Gradle build files)	e.g., `28`
`kotlin`	Version of Kotlin compiler and official libraries	e.g., `1.3.311`

Application dependencies

`additionalAppDependencies` and `additionalAndroidTestDependencies` array specify additional dependencies of the application under test that build tools should know about when building the Espresso server. For example: `"additionalAndroidTestDependencies": [ "com.google.android.material:material:1.0.0" ]`.

Items belonging to `additionalAppDependencies` array are translated to `implementation` lines in Gradle build files of the Espresso server. `additionalAndroidTestDependencies` are translated to `androidTestImplementation`.

iOS Only

These Capabilities are available only on the XCUITest Driver (/docs/en/drivers/ios-xcuitest/index.html) and the deprecated UIAutomation Driver (/docs/en/drivers/ios-uiautomation/index.html).

Capability	Description	Values
`calendarFormat`	(Sim-only) Calendar format to set for the iOS Simulator	e.g. `gregorian`
`bundleId`	Bundle ID of the app under test. Useful for starting an app on a real device or for using other caps which require the bundle ID during test startup. To run a test on a real device using the bundle ID, you may omit the 'app' capability, but you must provide 'udid'.	e.g. `io.appium.TestApp`
`udid`	Unique device identifier of the connected physical device	e.g. `1ae203187fc012g`
`launchTimeout`	Amount of time in ms to wait for instruments before assuming it hung and failing the session	e.g. `20000`
`locationServicesEnabled`	(Sim-only) Force location services to be either on or off. Default is to keep current sim setting.	`true` or `false`
`locationServicesAuthorized`	(Sim-only) Set location services to be authorized or not authorized for app via plist, so that location services alert doesn't pop up. Default is to keep current sim setting. Note that if you use this setting you MUST also use the `bundleId` capability to send in your app's bundle ID.	`true` or `false`
`autoAcceptAlerts`	Accept all iOS alerts automatically if they pop up. This includes privacy access permission alerts (e.g., location, contacts, photos). Default is false.	`true` or `false`
`autoDismissAlerts`	Dismiss all iOS alerts automatically if they pop up. This includes privacy access permission alerts (e.g., location, contacts, photos). Default is false.	`true` or `false`
`nativeInstrumentsLib`	Use native intruments lib (ie disable instruments-without-delay).	`true` or `false`
`nativeWebTap`	Enable "real", non-javascript-based web taps in Safari. Default: `false`. Warning: depending on viewport size/ratio; this might not accurately tap an element	`true` or `false`
`safariInitialUrl`	Initial safari url, default is a local welcome page	e.g. `https://www.github.com`
`safariAllowPopups`	(Sim-only) Allow javascript to open new windows in Safari. Default keeps current sim setting	`true` or `false`
`safariIgnoreFraudWarning`	(Sim-only) Prevent Safari from showing a fraudulent website warning. Default keeps current sim setting.	`true` or `false`
`safariOpenLinksInBackground`	(Sim-only) Whether Safari should allow links to open in new windows. Default keeps current sim setting.	`true` or `false`
`keepKeyChains`	(Sim-only) Whether to keep keychains (Library/Keychains) when appium session is started/finished	`true` or `false`
`localizableStringsDir`	Where to look for localizable strings. Default `en.lproj`	`en.lproj`
`processArguments`	Arguments to pass to the AUT using instruments	e.g., `-myflag`
`interKeyDelay`	The delay, in ms, between keystrokes sent to an element when typing.	e.g., `100`
`showIOSLog`	Whether to show any logs captured from a device in the appium logs. Default `false`	`true` or `false`
`sendKeyStrategy`	strategy to use to type test into a test field. Simulator default: `oneByOne`. Real device default: `grouped`	`oneByOne`, `grouped` or `setValue`
`screenshotWaitTimeout`	Max timeout in sec to wait for a screenshot to be generated. default: 10	e.g., `5`
`waitForAppScript`	The ios automation script used to determined if the app has been launched, by default the system wait for the page source not to be empty. The result must be a boolean	e.g. `true;`, `target.elements().length > 0;`, `$.delay(5000); true;`
`webviewConnectRetries`	Number of times to send connection message to remote debugger, to get webview. Default: `8`	e.g., `12`
`appName`	The display name of the application under test. Used to automate backgrounding the app in iOS 9+.	e.g., `UICatalog`
`customSSLCert`	(Sim only) Add an SSL certificate to IOS Simulator.	e.g. `-----BEGIN CERTIFICATE-----MIIFWjCCBEKg... -----END CERTIFICATE-----`
`webkitResponseTimeout`	(Real device only) Set the time, in ms, to wait for a response from WebKit in a Safari session. Defaults to `5000`	e.g., `10000`

Capability	Description	Values
`remoteDebugProxy`	(Sim only, <= 11.2) If set, Appium sends and receives remote debugging messages through a proxy on either the local port (Sim only, <= 11.2) or a proxy on this unix socket (Sim only >= 11.3) instead of communicating with the iOS remote debugger directly.	e.g. `12000` or `"/tmp/my.proxy.socket"`
`enableAsyncExecuteFromHttps`	capability to allow simulators to execute asynchronous JavaScript on pages using HTTPS. Defaults to `false`	`true` or `false`
`skipLogCapture`	Skips to start capturing logs such as crash, system, safari console and safari network. It might improve performance such as network. Log related commands will not work. Defaults to `false` .	`true` or `false`
`webkitDebugProxyPort`	(Real device only) Port to which `ios-webkit-debug-proxy` is connected, during real device tests. Default is `27753` .	`12021`
`fullContextList`	Returns the detailed information on contexts for the get available context (/docs/en/commands/context/get-contexts/index.html) command. If this capability is enabled, then each item in the returned contexts list would additionally include WebView title, full URL and the bundle identifier. Defaults to `false` .	`true` or `false`

iOS Only, using XCUITest

(For XCUITest-specific capabilities, please refer to the documentation on the XCUITest Driver repo (https://github.com/appium/appium-xcuitest-driver#desired-capabilities) itself.)

MacDriver Only

(For MacDriver capabilities, please refer to the documentation on the Appium MacDriver repo (https://github.com/appium/appium-mac-driver#desired-capabilities) itself.)

Mac2Driver Only

(For Mac2 Driver capabilities, please refer to the documentation on the Appium Mac2Driver repo (https://github.com/appium/appium-mac2-driver#capabilities) itself.)

You.i Engine Only

(For You.i Engine-specific capabilities, please refer to the documentation on the You.i Engine driver (https://github.com/YOU-i-Labs/appium-youiengine-driver#desired-capabilities) itself.)

WinAppDriver Only

(For WinAppDriver specific capabilities, please refer to the documentation on the Appium Windows Driver repo (https://github.com/appium/appium-windows-driver#windowsdriver-specific-capabilities) itself.)

Flutter driver only

(For FlutterDriver specific capabilities, examples please refer to the documentation on the Flutter Driver repo (https://github.com/truongsinh/appium-flutter-driver#desired-capabilities-for-flutter-driver-only) itself.)