Overview

Start Your Application

Find UI elements

Validate UI elements

Implement common UI test patterns

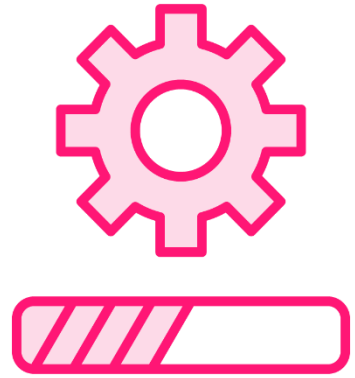
Understand how to wait for elements to appear and disappear

Summary

Start Your Applicaition

Starting Your Application

Send POST Session command with the required capabilities



Appium will find the corresponding driver and uses the driver to create a new session

```
var capabilities = new AppiumOptions();
// specify options here ...

driver = new WindowsDriver(new Uri("http://127.0.0.1:4723"), capabilities);

var appiumLocalService = new
            AppiumServiceBuilder().UsingAnyFreePort().Build();
appiumLocalService.Start();

driver = new WindowsDriver(appiumLocalService, capabilities);
```

Starting the Appium Server

Assume a server is already running

Start your own self hosted server

- Requires NodeJs installed on your system
- Environment variable APPIUM_HOME must point to the driver you are going to use

Find UI Elements

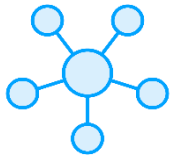
How to Find a Mobile UI Element

Class Name
e.g. android.widget.ListView, or ListView (on windows)

Accessibility id
for iOS the accessibility identifier, for Android the content-description, and for windows, the AutomationId

Xpath
a valid xpath string applied to the XML document that would be retrieved using the page source command

How to Find a Mobile UI Element

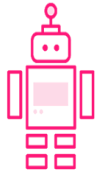


Id

Platform specific way to find an element. On Android e.g. the resource identifier. E.g. **<appPackagename>:id/toolbar**

Name

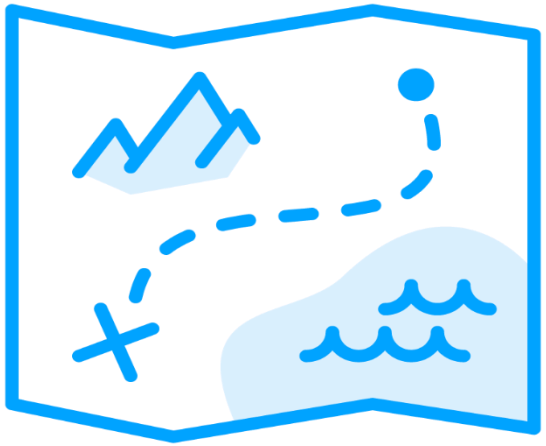
Platform specific. Throws exception on Android and maps to control name in Windows



-android uiautomation, -ios predicate string, -ios class chain

Platform specific implementation of a locator strategy using a script

Use PageSource



Driver.PageSource;

Retrieve the document representing the UI

Used to determine how to find a specific element

Use Appium Inspector

Part of the Appium UI version

- Start the Appium server
- Set Appium Options
- Start the inspector

Visual inspection to find locator strategy

```
public static By ClassName(string classNameToFind);
public static By Id(string idToFind);
public static By Name(string nameToFind);
public static By XPath(string xpathToFind);
```

Using the FindElement(By.XXX) operations

Only items listed here are usable with mobile applications

CssSelector, LinkText, PartialLinkText and TagName are not supported

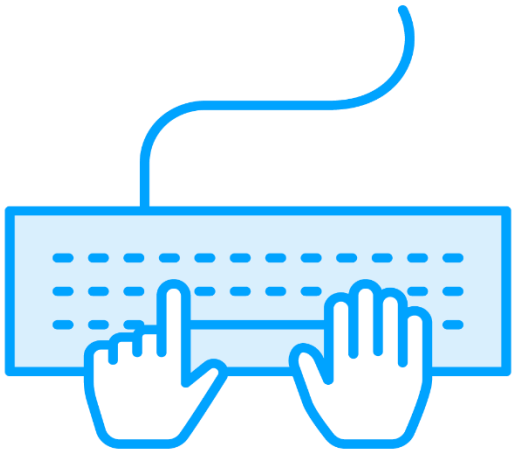
```
public static MobileBy Name(string nameToFind);
public static MobileBy AccessibilityId(string selector);


public static MobileBy AndroidUIAutomator(string selector);
public static MobileBy IosClassChain(string selector);
public static MobileBy IosNSPredicate(string selector);
public static MobileBy IosUIAutomation(string selector);
public static MobileBy TizenAutomation(string selector);
public static MobileBy WindowsAutomation(string selector);
```

Using the MobileBy helper class

Conform the Appium mobile WebDriver protocol extensions

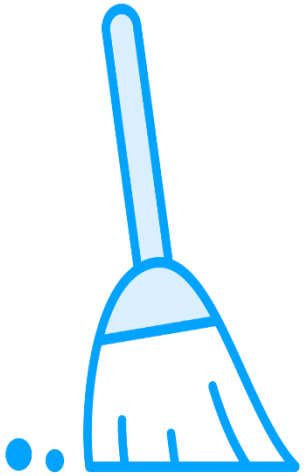
Interact With UI Elements

Basic Interactions Text Fields



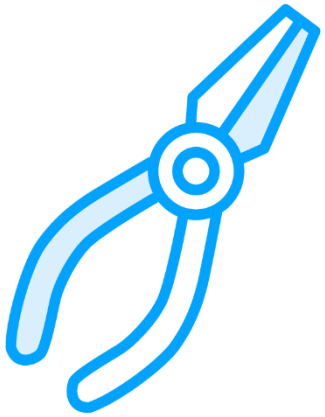
Enter Keystrokes

SendKeys()



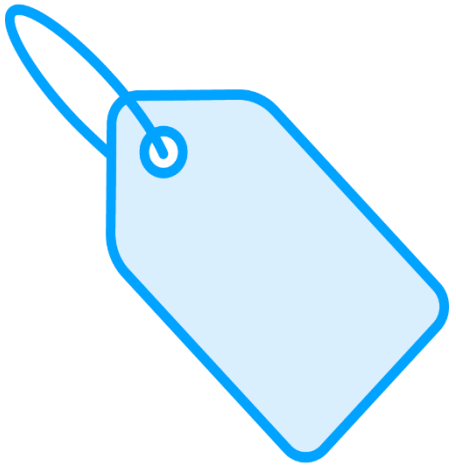
Clear an input field

Clear()



Retrieve the text on a field or label

GetText()



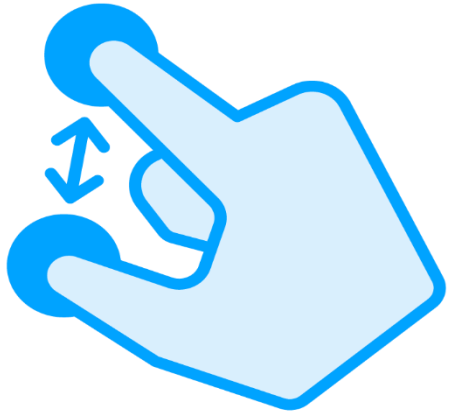
Get any platform specific value

GetAttribute()

Using Pointer Input

Click/Tap

Touch



Multitouch

```
var listView = driver.FindElement(MobileBy.Name("Second item"));
listView.Click();
```

Tap an element we find

Click method will do the tap

Tap is done at the center of the rectangle of the element found

Using Pointer Input Device

Touch, Mouse, Pen

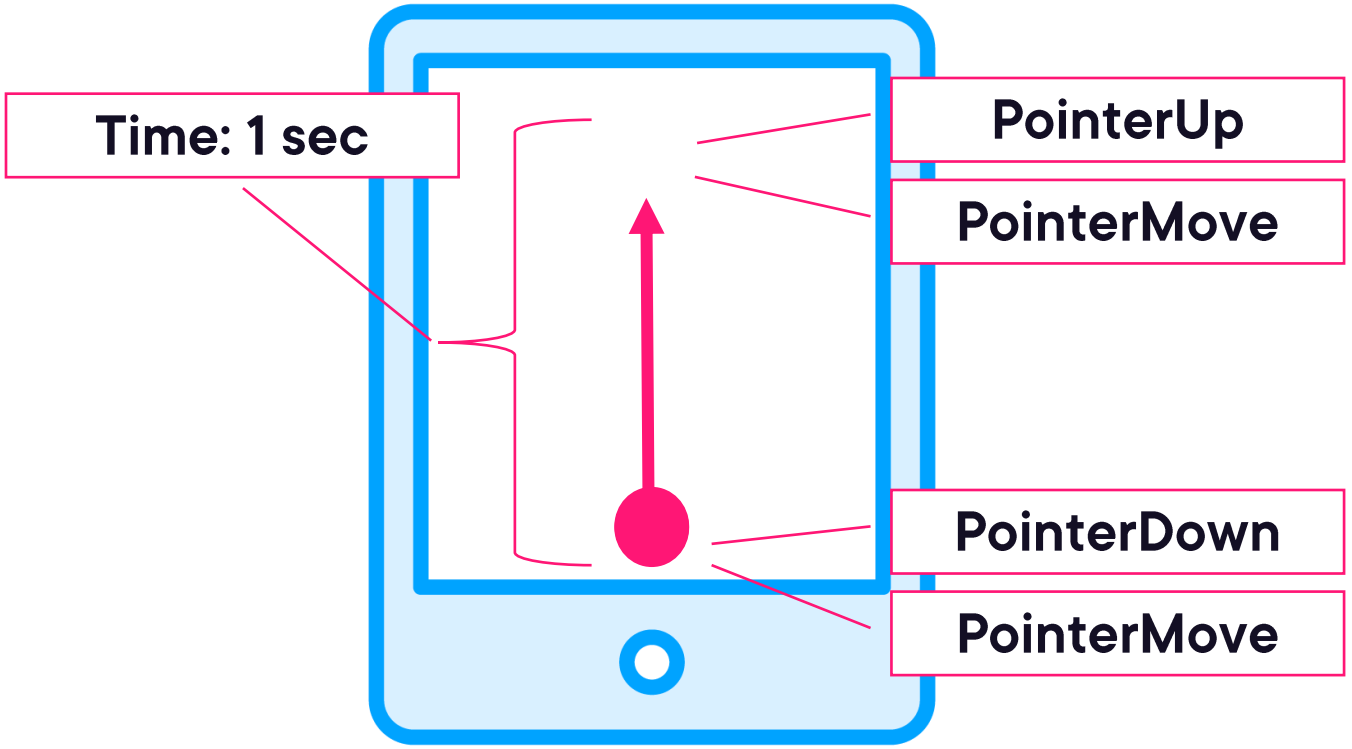
Pointer input types

- Touch, Mouse, Pen

Pointer name

Create sequence of actions

Perform the sequence



```
void FlickUp(AndroidDriver driver, AppiumElement element)
{
    var input = new PointerInputDevice(PointerKind.Touch);
    ActionSequence flickUp = new ActionSequence(input);
    flickUp.AddAction(input.CreatePointerMove(element, 0, 0,TimeSpan.Zero));
    flickUp.AddAction(input.CreatePointerDown(MouseButton.Left));
    flickUp.AddAction(input.CreatePointerMove(element,0, -200,
                                              TimeSpan.FromMilliseconds(200)));
    flickUp.AddAction(input.CreatePointerUp(MouseButton.Left));

    driver.PerformActions(new List<ActionSequence>() { flickUp });
}
```

Creating a Touch Flick

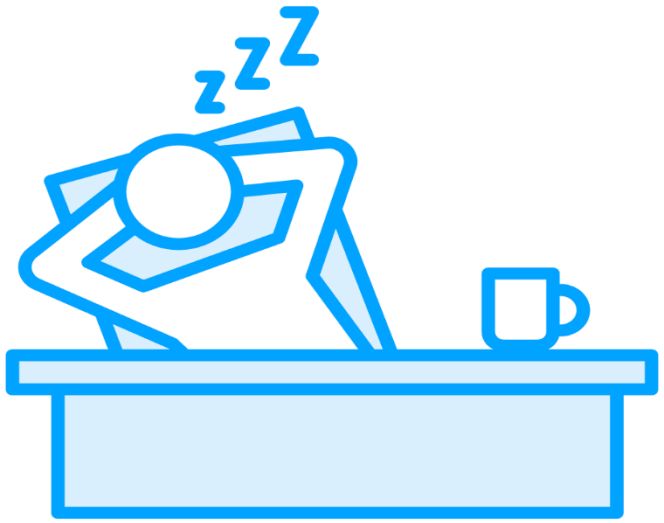
Use MouseButton.Left

Good value typically goes around 150 - 200 pixels/sec move

Use time window of 200-400 millisecond

Wait for Elements to Appear and Disappear

How To Wait



Sometimes you need to wait for elements to either appear or disappear

Never use Thread.Sleep()!

- Unnecessary fixed wait times

Client libraries offer better support for this

```
private void WaitForElementToAppear(AndroidDriver driver)
{
    var wait = new DefaultWait<AndroidDriver>(driver)
    {
        Timeout = TimeSpan.FromSeconds(60),
        PollingInterval = TimeSpan.FromMilliseconds(500)
    };
    wait.IgnoreExceptionTypes(typeof(NoSuchElementException));
    wait.Until(d => d.FindElement(MobileBy.AccessibilityId("<some id>")));
}
```

Waiting On Elements

- **Client library provides option to wait on elements**
- **Poling every 500 millisecond**
- **Executing function FindElement every iteration**
- **Until Timeout**

Summary

- Start Your Application
- Find UI elements
- Validate UI elements
- Implement common UI test patterns
- Understand how to wait for elements to appear and disappear