



# Automatización de pruebas en móviles

Sesión 1

06 OCT 2021

# Temas de la sesión 1:

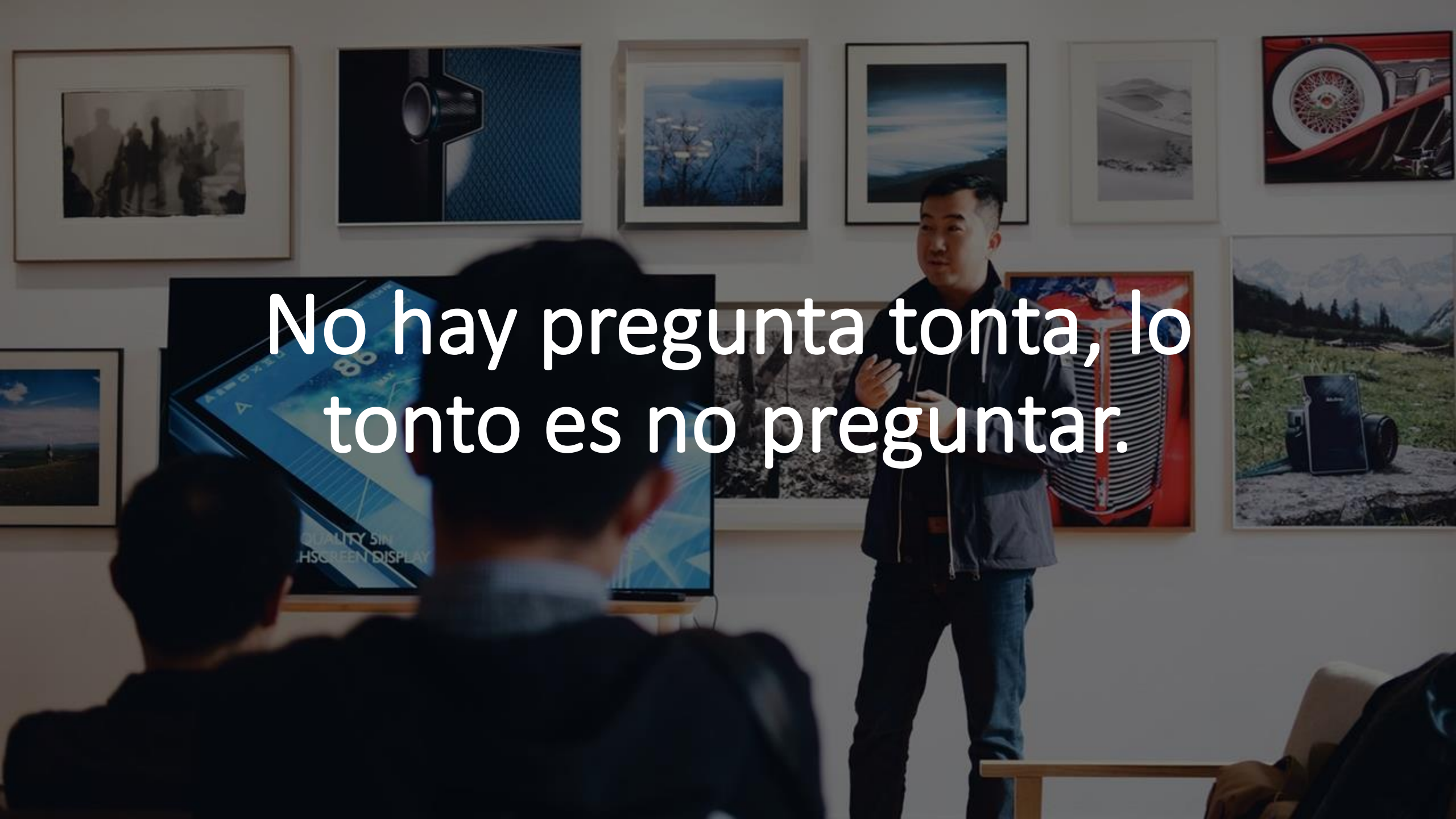
- Presentación del curso
- Introducción a la automatización de pruebas móviles
- Instalación y configuración del entorno de trabajo

*Ningún mar en calma hizo experto a un marinero*

# Reglas de las sesiones

- La clase inicia a las 7:00am y termina 9:00 am
  - Primera llamada 7:00am
  - Segunda llamada 7:05am
- Tendremos un break a las 08:00 por 10 minutos.
- La participación en clase será levantando la mano o escribiendo en el chat y luego de cerrar un tema.

No hay pregunta tonta, lo  
tonto es no preguntar.







No solo mires la partida;  
juega, pierde y aprende

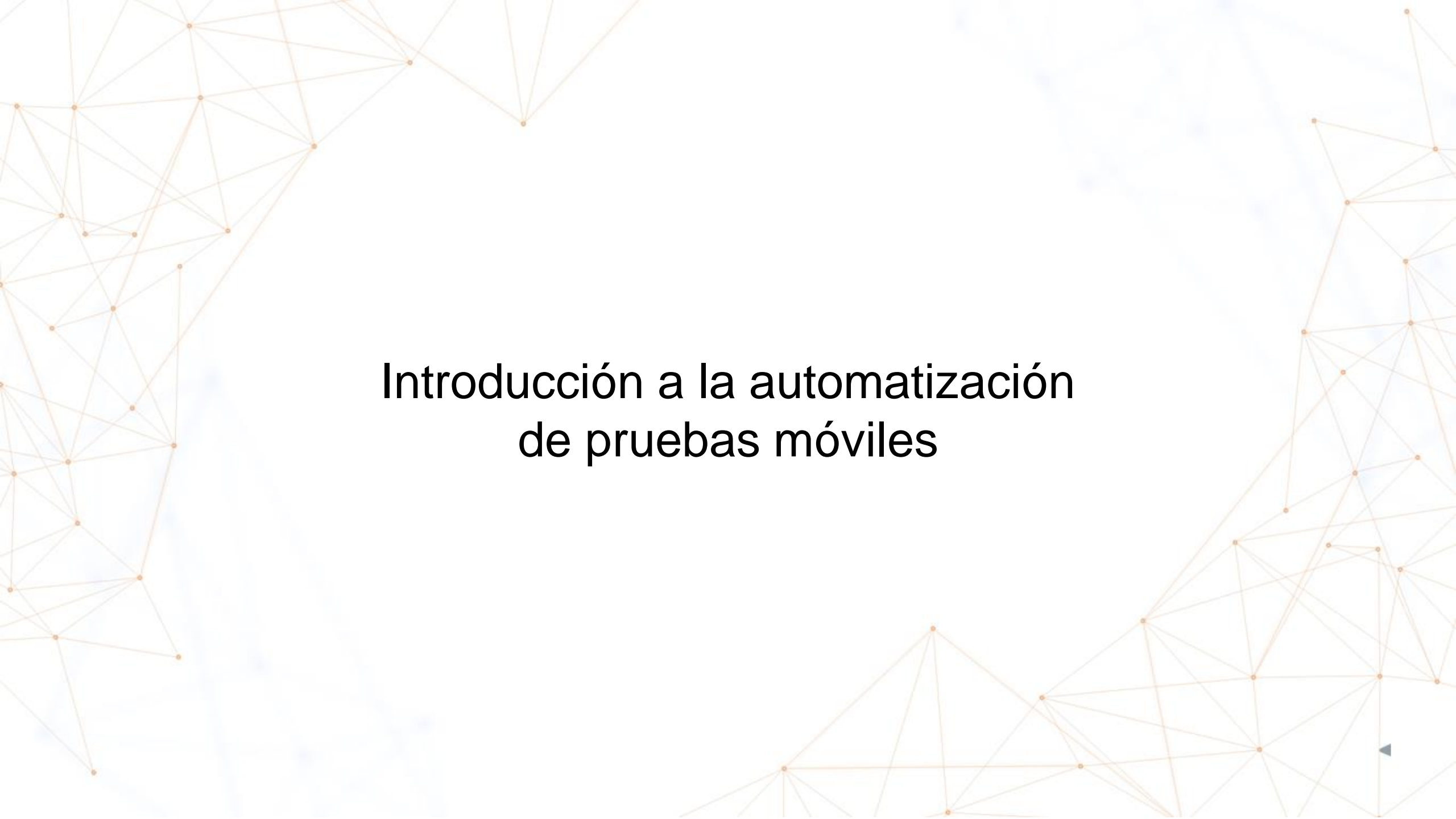
# Objetivos

## GENERAL

Este curso está orientado a potenciar la capacidad del analista de calidad haciendo el uso adecuado de herramientas que le permitan automatizar pruebas en móviles.

## AL FINALIZAR EL CURSO

- Aprender el uso adecuado de locators.
- Comprender el lenguaje Gherkin
- Entender el proceso de automatización de pruebas.
- Automatizar casos de prueba usando JAVA.



# Introducción a la automatización de pruebas móviles

# Objetivo

Incrementar el **nivel de calidad** del software, optimizando los **tiempos de pruebas** para obtener **información a tiempo**.

Automation •  
tools •



Automation Testing

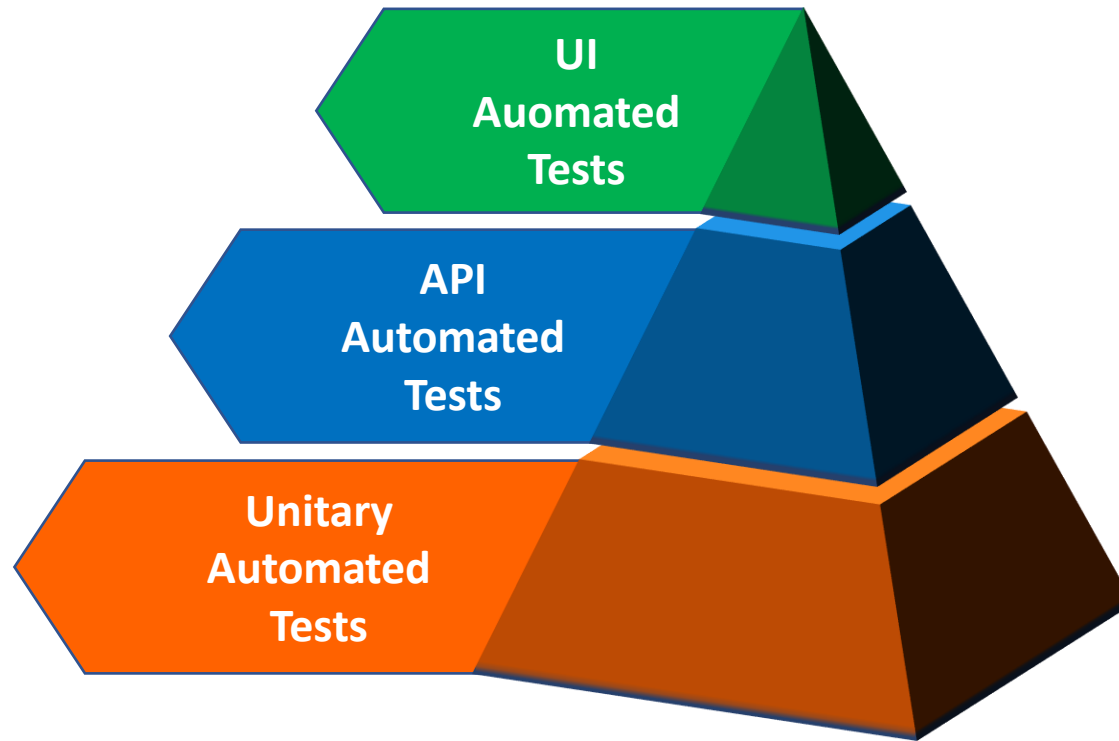


# Beneficios de la Automatización

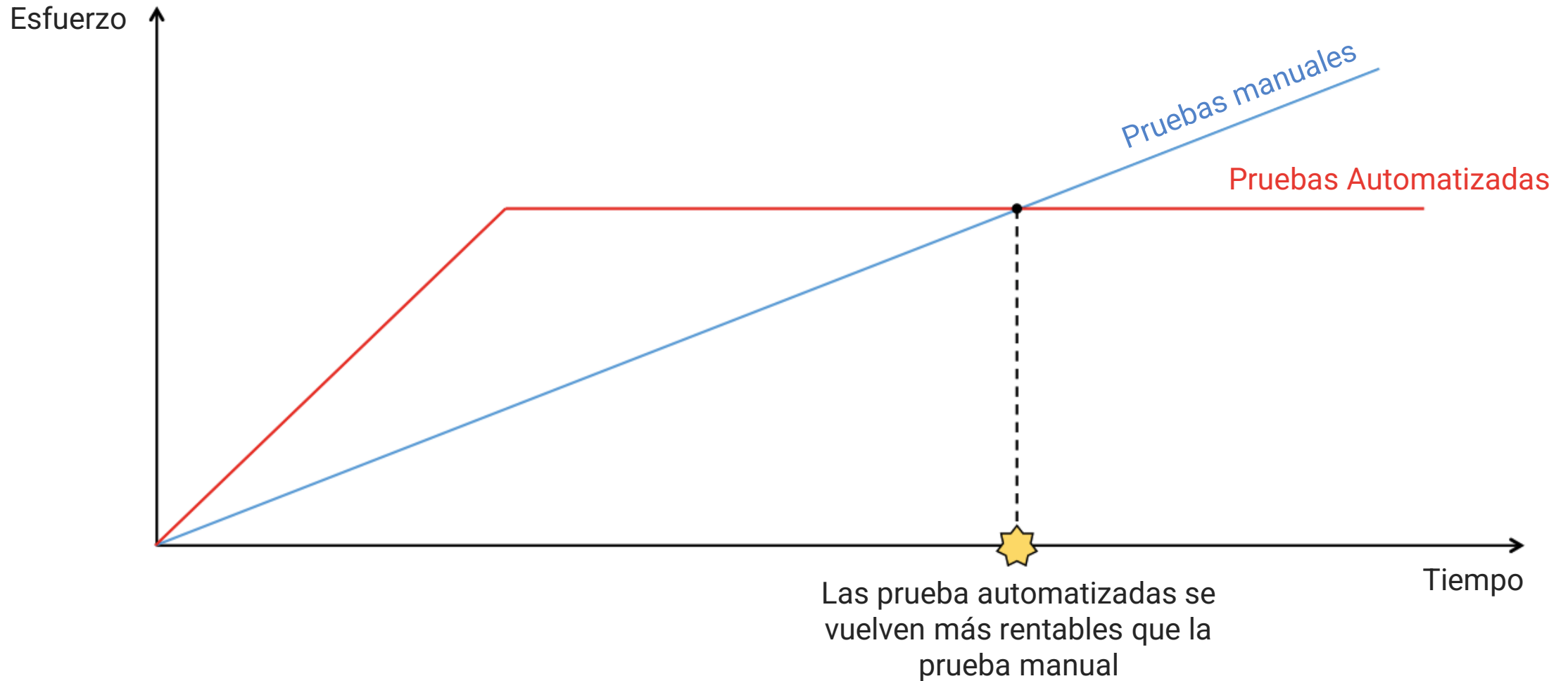


- ✓ Incrementa la cobertura de pruebas.
- ✓ Reduce el trabajo manual repetitivo.
- ✓ Incrementa confiabilidad y consistencia de las pruebas.
- ✓ Evaluación más objetiva.
- ✓ Acceso oportuno sobre las pruebas.
- ✓ Pluralidad de datos de prueba.
- ✓ Reducción de errores en funcionalidades de alto riesgo.
- ✓ Se adapta a modelos CI/CD.

# Pruebas automatizadas



# Comparación tiempo vs esfuerzo



# ¿Qué automatizar?

- Procesos de negocio que sean críticos y repetitivos
- Casos de regresión críticos para el funcionamiento de la aplicación.
- Pruebas end to end que garanticen el funcionamiento de la aplicación (funcionamiento transversal) validando que las funcionalidades que fueron probadas individualmente, funcionen en conjunto.
- Casos de prueba tediosos o difíciles de realizar manualmente o que nos toman mucho tiempo.

# ¿Qué no automatizar?

- Casos de prueba que se han creado y no se han probado manualmente al menos una vez.
- Casos de prueba donde los requerimientos cambian constantemente.
- Casos de prueba para una solución ad-hoc.
- Prueba dependientes. No obstante, se debe analizar si la dependencia puede ser “bypaseada” o reemplazada por un servicio.



# Grandes mentiras

- Las pruebas manuales desaparecen.
- Veré resultados desde el primer día.
- Un automatizador es mejor que un tester manual.
- Debo automatizar todos los casos de prueba.
- Ahora sí se encontrarán todos los errores.



Maven™



appium



cucumber

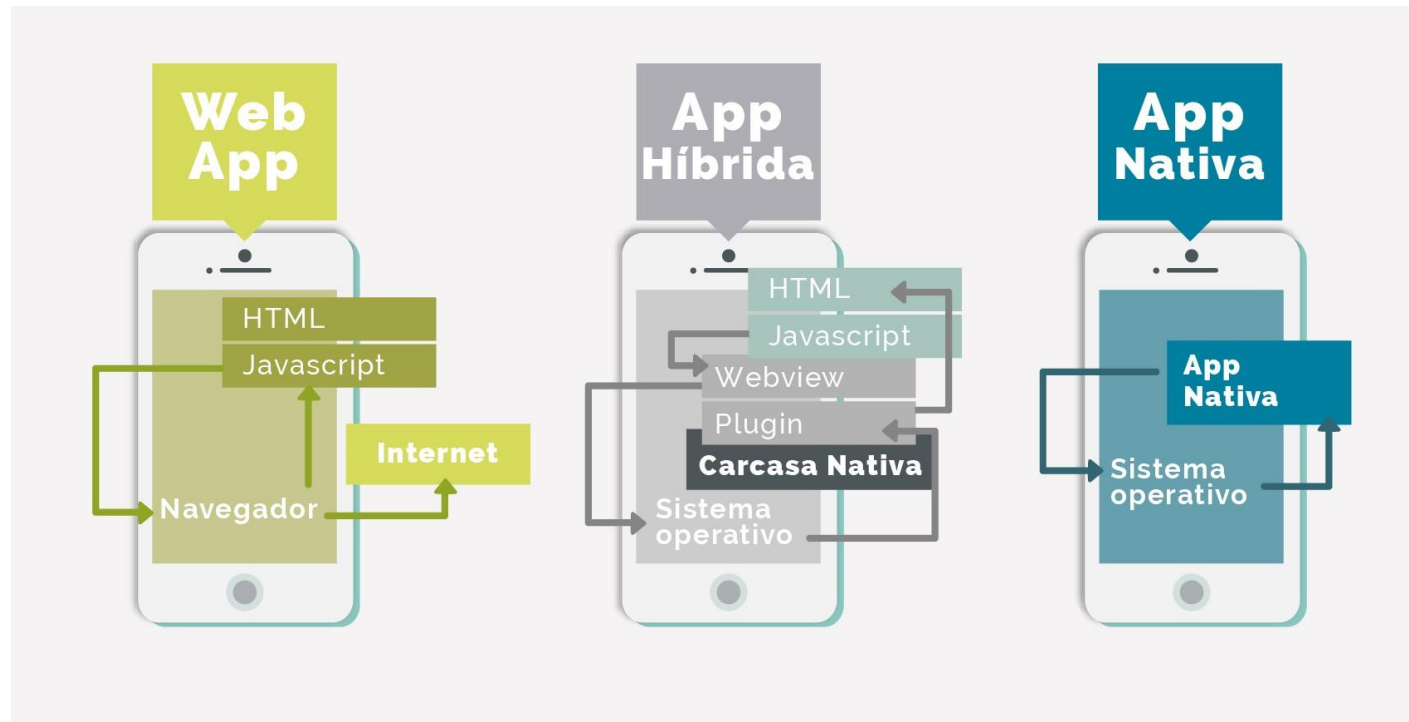


Java

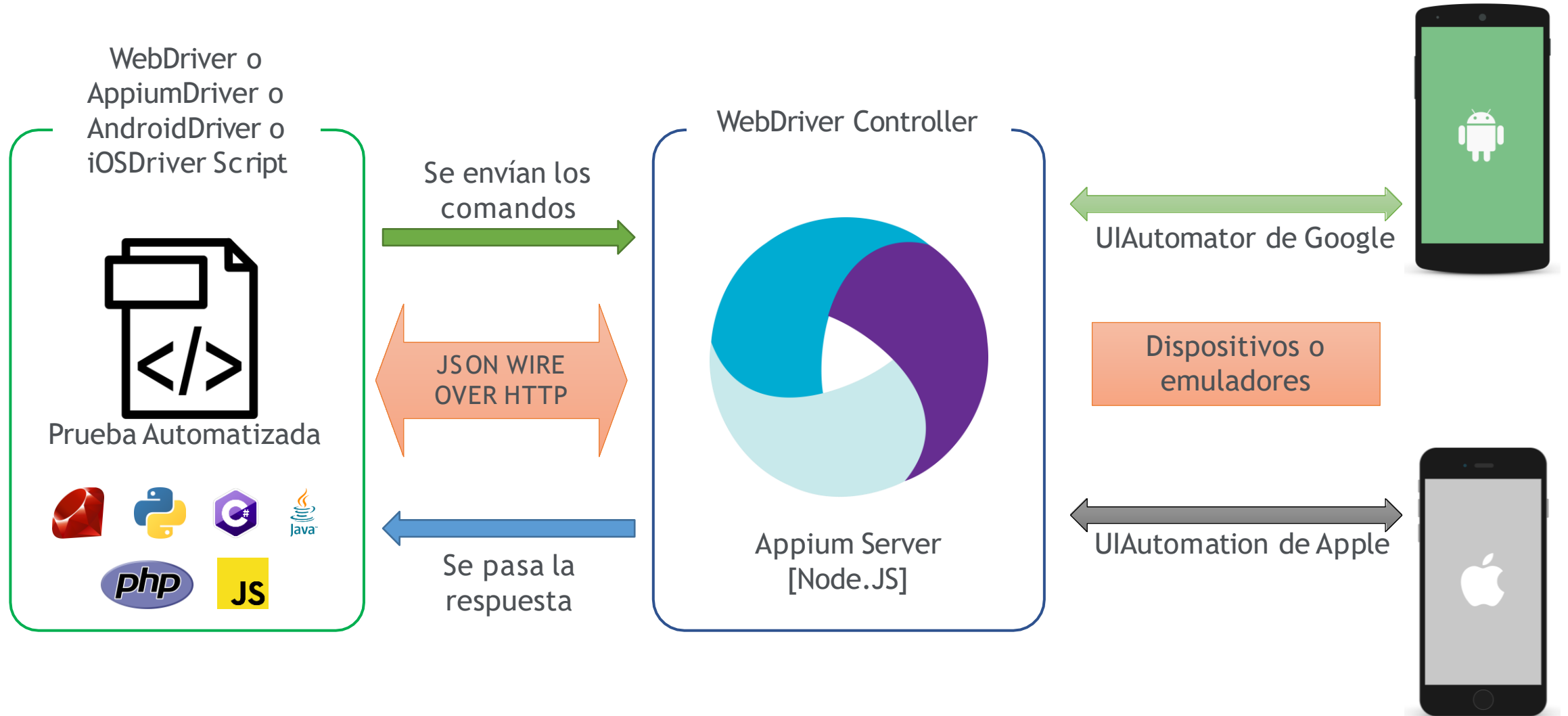
# Herramientas a utilizar


# Appium: Definición

Appium es una herramienta de automatización de código abierto para ejecutar scripts y probar aplicaciones nativas, aplicaciones web móviles y aplicaciones híbridas en Android o iOS **utilizando un controlador web.**



# Appium: Funcionamiento





# Instalación y configuración del entorno de trabajo.



# Requerimientos

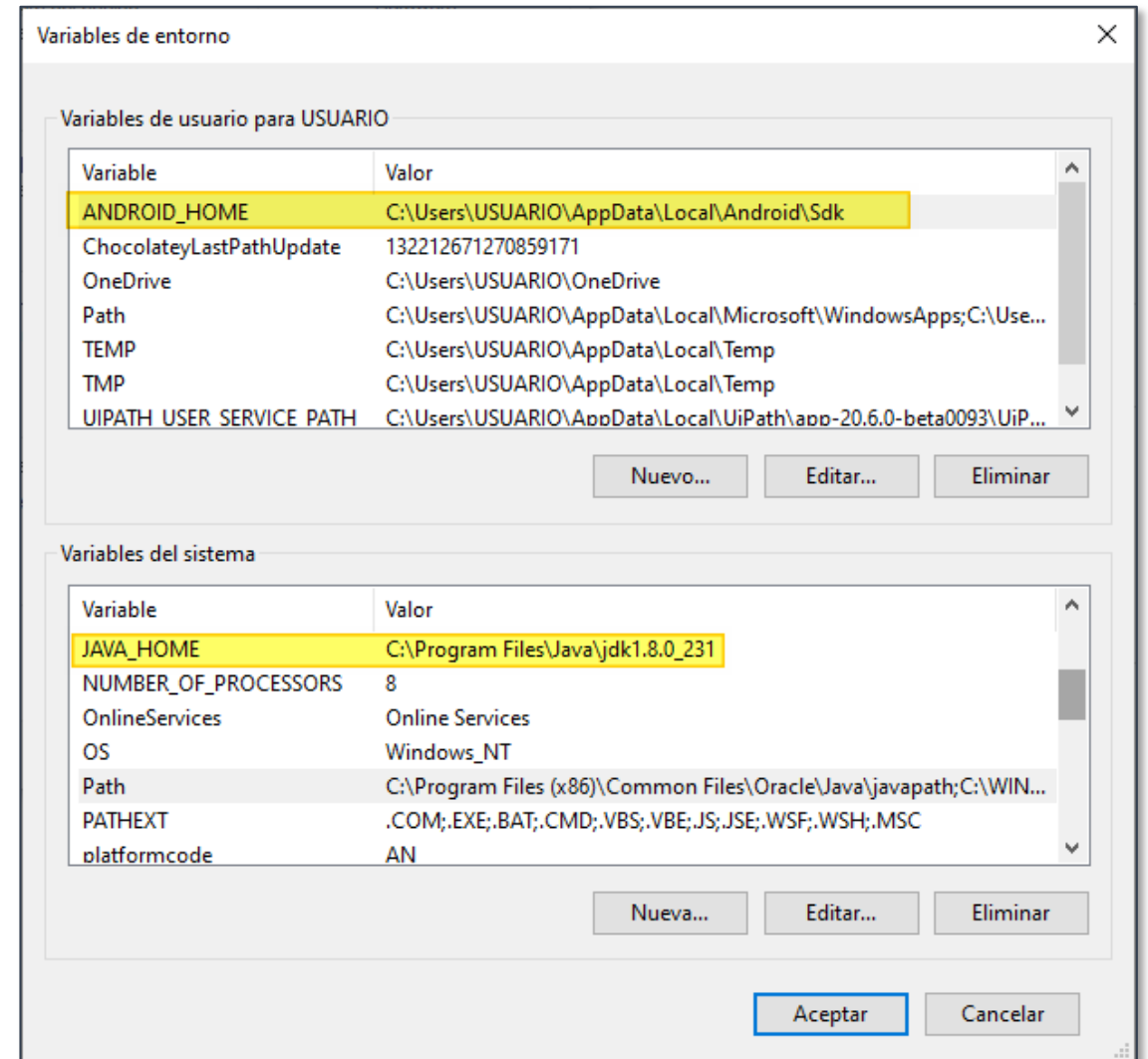
## Páginas oficiales

- IntelliJ (Community): <https://www.jetbrains.com/es-es/idea/download>
- Java SE (JDK 8 o 11): <https://www.oracle.com/java/technologies/javase-downloads.html>
- NodeJS (versión LTS): <https://nodejs.org/es/>
- Android Studio: <https://developer.android.com/studio>
- Appium: <http://appium.io/>

# Variables de entorno

Configurar las siguientes variables de entorno:

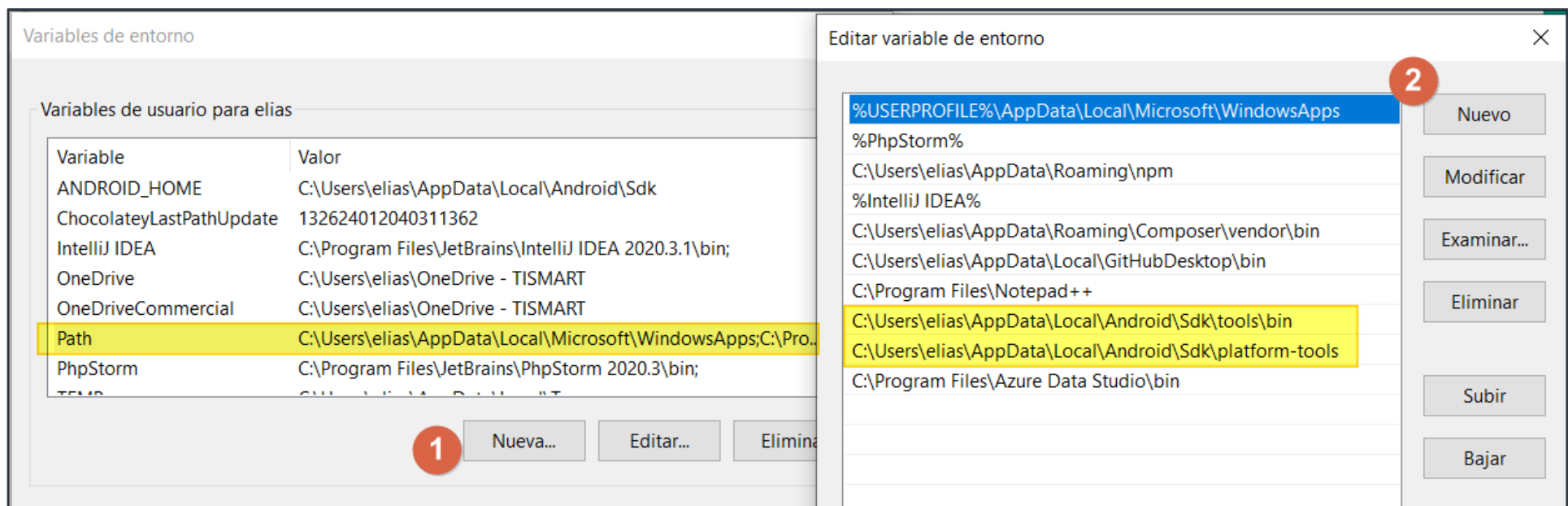
1. JAVA\_HOME con la ruta: C:\ruta\a\java\jdk1.8
2. ANDROID\_HOME con la ruta al SDK



# Variables de entorno

Opcionalmente, puede configurar las rutas de **tools/bin** y **platform-tools** para no tener que cambiar siempre a la ruta de **ANDROID\_HOME** antes de ejecutar ADB.

Para ello, agregue dos variables de entorno en la variable de usuario **Path**. Estas deben hacer referencia a la ruta donde se encuentra su **Android SDK**.



# ADB, UIAutomator Viewer y Modo Desarrollador

**ADB (Android Debug Bridge):** Es una herramienta mediante la cual la consola de comandos de nuestro PC hará de puente entre este y el teléfono. Gracias a ella, podemos enviar órdenes al smartphone, así como cargar archivos entre una y otra plataforma.

**UIAutomator Viewer:** Es una herramienta que proporciona una GUI conveniente para escanear y analizar los componentes de la interfaz de usuario que se muestran actualmente en un dispositivo Android.

**Modo Desarrollador:** Es una de las opciones que pueden ser accedidas desde nuestro dispositivo para trabajar en el debug de nuestra app y otros aspectos.

# Activar el Modo Desarrollador

Dependiendo del dispositivo Android que tengamos los pasos podrían variar. Estos pasos aplican para un dispositivo Huawei

1. Abrir el menú de ajustes o configuración del teléfono
2. Busque la opción **Acerca del Teléfono**
3. Ubicamos la opción donde muestra la compilación de nuestro dispositivo y hacemos tap 7 veces para activar el modo desarrollador.
4. Regresamos un nivel atrás y veremos un nuevo menú **OPCIONES DEL DESARROLLADOR**.
5. Según el dispositivo, activaremos estas opciones:
  1. Depuración USB
  2. Permitir depuración ADB en modo solo carga.





# ADB: Detectar dispositivo

Abrir una consola de Windows (CMD) y ejecutar:

```
> cd %android_home%\platform-tools\  
> adb devices -l
```

Debe conectar su dispositivo a la PC y haber activado previamente el modo desarrollador

ADB nos listará todos los dispositivos encontrados

```
C:\WINDOWS\system32\cmd.exe  
Microsoft Windows [Versión 10.0.18363.959]  
(c) 2019 Microsoft Corporation. Todos los derechos reservados.  
C:\Users\USUARIO>cd %android_home%\platform-tools\  
C:\Users\USUARIO\AppData\Local\Android\Sdk\platform-tools>adb devices -l  
* daemon not running; starting now at tcp:5037  
* daemon started successfully  
List of devices attached  
A4NDU19A31001505    device product:MAR-LX3A model:MAR_LX3A device:HWMAR transport_id:1  
C:\Users\USUARIO\AppData\Local\Android\Sdk\platform-tools>
```

1

2

Nombre del dispositivo

UDID del dispositivo

# UIAutomator Viewer

Para ejecutar el UIAutomator Viewer ejecutaremos este comando desde la consola de Windows (CMD):

```
> %android_home%\tools\bin\uiautomatorviewer
```

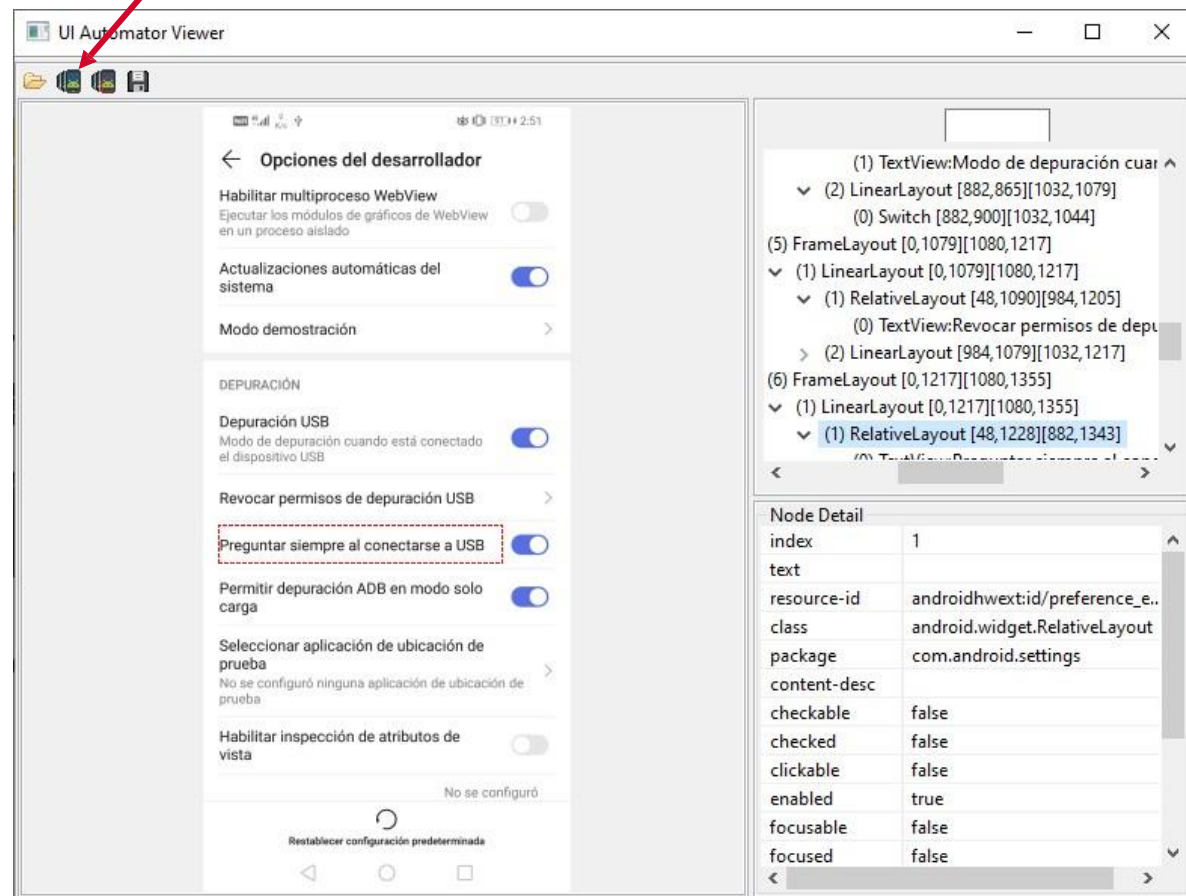
```
C:\WINDOWS\system32\cmd.exe - C:\Users\USUARIO\AppData\Local\Android\Sdk\tools\bin\uiautomatorviewer
Microsoft Windows [Versión 10.0.18363.959]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.
C:\Users\USUARIO>%android_home%\tools\bin\uiautomatorviewer
```

Debe conectar su dispositivo a la PC y haber activado previamente el modo desarrollador

En el UI Automator Viewer haga clic en el primer icono y el programa cargará la vista actual de su dispositivo.

Con este programa podremos analizar la posición y datos de todos los elementos actualmente visibles en el dispositivo.

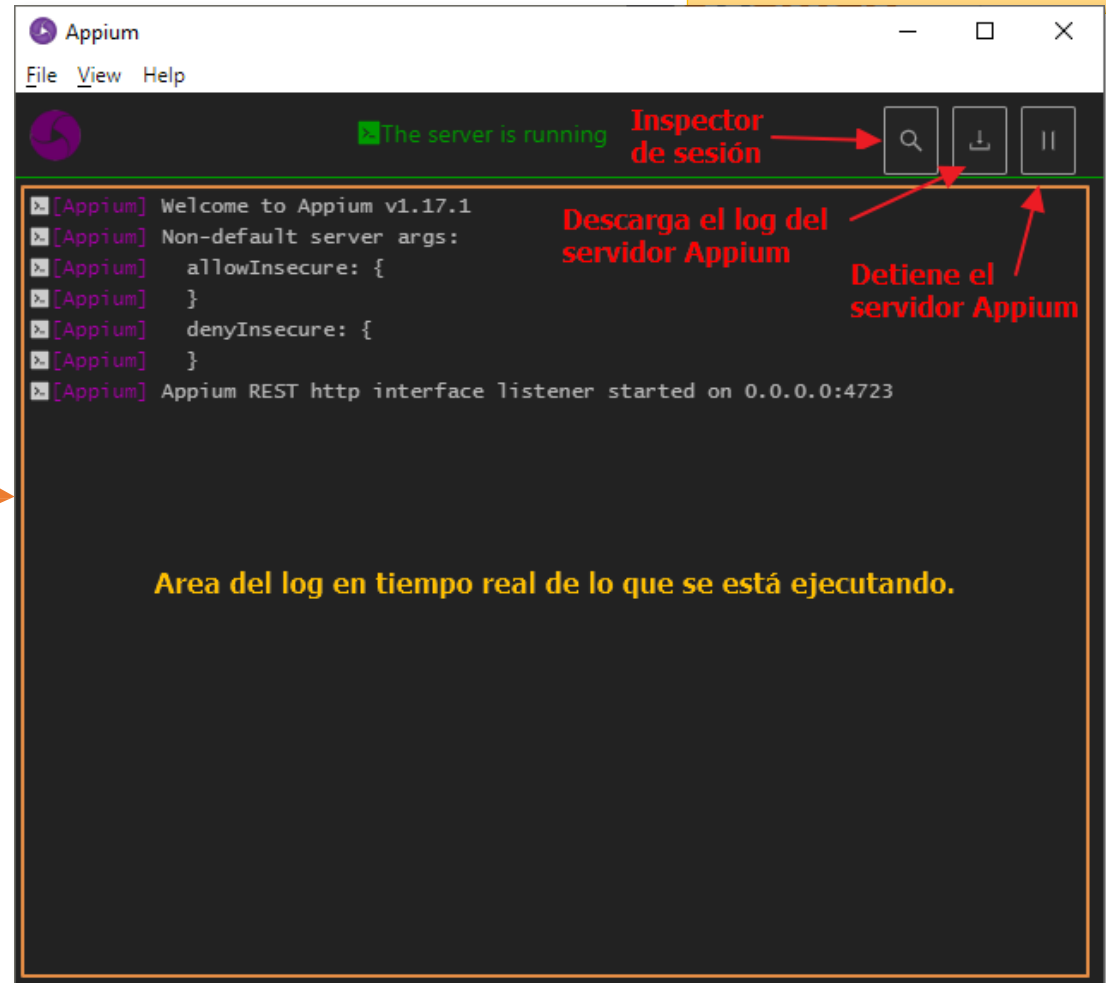
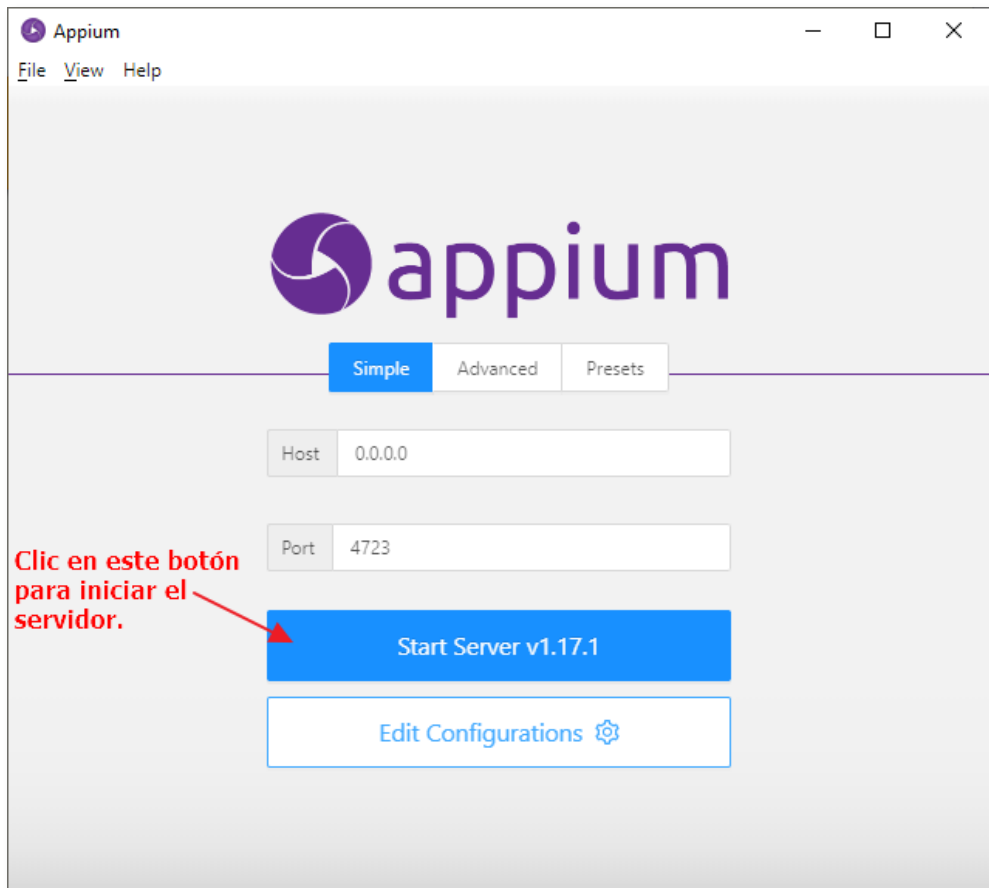
**Nota:** Esto es sólo una “captura” de lo que se encontró en el dispositivo. No es una vista en tiempo real.



# Appium: Iniciando el servidor

Abriremos el programa Appium y haremos clic sobre el botón Start Server.

Para que Appium inicie y ejecute satisfactoriamente deberá haber configurado correctamente las variables de entorno JAVA\_HOME y ANDROID\_HOME



# Desired Capabilities

Son configuraciones que nos **ayudan a modificar el comportamiento del servidor** (Appium) durante el tiempo de ejecución de los scripts de automatización. La configuración en **Appium** se realiza como hashmap o par **clave-valor**.

- **Principales funciones:**

1. Ayuda al usuario a controlar la *'session request'* con el servidor Appium.
2. Se utilizan para configurar la instancia de Webdriver, por ejemplo: FirefoxDriver, ChromeDriver, InternetExplorerDriver.

- Para trabajar con **Desired Capabilities** en java será necesario importar la librería:

- **org.openqa.selenium.remote.DesiredCapabilities.**

- Aquí puedes conocer todas las capabilities disponibles: <https://appium.io/docs/en/writing-running-appium/caps/>

```
{
  "platformName": "android",
  "platformVersion": "9",
  "deviceName": "HWMAR",
  "automationName": "UiAutomator2",
  "browserName": "chrome",
  "appPackage": "com.android.calculator2"
}
```



```
import io.appium.java_client.AppiumDriver;
import org.openqa.selenium.remote.DesiredCapabilities;

{
    DesiredCapabilities capabilities = new DesiredCapabilities();
    capabilities.setCapability("deviceName", "Android Emulator");
    capabilities.setCapability("platformVersion", "4.4");
}
```



# Configurando el Inspector de sesión

Cuando el servidor Appium inicie, abriremos el inspector de sesión ([icono de lupa](#)).

Configure las siguientes **capabilities**, guarde (Save) y luego inicie la sesión (Start Session)

Capability	Descripción
automationName	Define el motor de automatización a utilizar. Los valores posibles son: <b>Para Android:</b> Appium, UiAutomator2, Espresso, UiAutomator1. <b>Para iOS:</b> XCUITest o Instruments <b>You.i Engine:</b> YouiEngine
platformName	Que sistema operativo del dispositivo usaremos. Valores posibles: Android, iOS, FirefoxOS
platformVersion	Versión del sistema operativo del dispositivo móvil.
deviceName	El nombre del dispositivo móvil o emulador a usar. Vea la <a href="#">diapositiva 13</a> para conocer cómo encontrar este nombre.

The screenshot shows the Appium Desktop application window. The 'Automatic Server' tab is selected, showing the current server URL as `http://localhost:4723`. Below this, the 'Desired Capabilities' section is active, displaying a table of configuration options:

Capability	Type	Value	Action
automationName	text	UiAutomator2	Remove
platformName	text	android	Remove
platformVersion	text	9	Remove
deviceName	text	HWMAR	Remove

Annotations with red arrows point to various elements:

- Configura un nuevo arreglo de configuración de capabilities.** Points to the 'Advanced Settings' link.
- Arreglos de capabilities guardados.** Points to the 'Saved Capability Sets 1' tab.
- Editar directamente el arreglo en formato JSON** Points to the 'JSON Representation' section, which shows the following JSON:

```
{  "automationName": "UiAutomator2",  "platformName": "android",  "platformVersion": "9",  "deviceName": "HWMAR"}
```

- Agrega una nueva línea para configurar una capability** Points to the '+' button at the bottom of the capabilities table.
- Graba el arreglo de configuraciones. Al hacer clic aquí deberá proporcionar un nombre. Por ejemplo: "SamsungNote10"** Points to the 'Save' button.
- Inicia la sesión y ejecuta el comando en nuestro dispositivo.** Points to the 'Start Session' button.

At the bottom left, there is a link for 'Desired Capabilities Documentation'.





Gracias