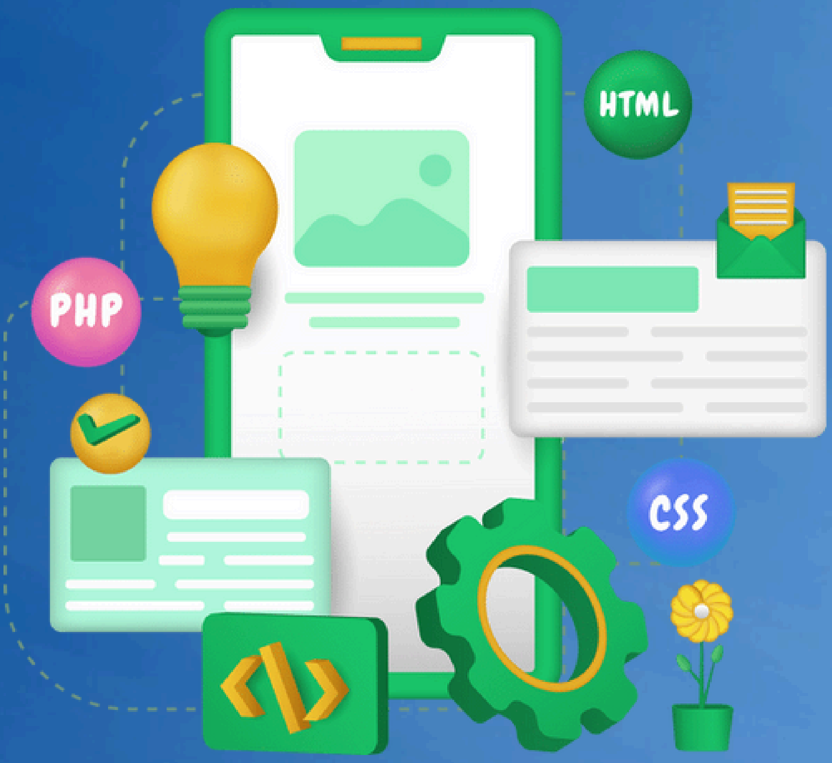
Guide to Automating Mobile Applications with Appium and Python



Step 1: Environment Setup

1. Install Appium and Set Up Its Environment

1. Install Node.js (required for Appium):
   - Download and install Node.js from https://nodejs.org/
   - Verify installation:

```
node -v
npm -v
```

2. Install Appium globally using npm:

```
npm install -g appium
```

3. Verify Appium installation:

```
appium -v
```

4. Install Appium Inspector (optional, for identifying app elements):
   - Download Appium Inspector from Appium Inspector GitHub.

2. Install Android Studio and Configure the Android SDK

1. Download and install Android Studio from https://developer.android.com/studio.
2. Open Android Studio and install the Android SDK, SDK tools, and platform tools.
3. Set up environment variables:
   - ANDROID_HOME:
     - Path to the Android SDK directory (e.g., C:\Users\YourUsername\AppData\Local\Android\Sdk).
   - Add the following paths to the PATH variable:
     - ANDROID_HOME\platform-tools
     - ANDROID_HOME\tools
     - ANDROID_HOME\tools\bin
   - Example (Windows):

```
setx ANDROID_HOME "C:\Users\YourUsername\AppData\Local\Android\Sdk"
setx PATH "%PATH%;%ANDROID_HOME%\platform-tools;%ANDROID_HOME%\tools;%ANDROID_HOME%\tools\bin"
```

Example (macOS/Linux): Add this to ~/.bashrc or ~/.zshrc:

```
export ANDROID_HOME=$HOME/Library/Android/sdk
export PATH=$PATH:$ANDROID_HOME/platform-tools:$ANDROID_HOME/tools:$ANDROID_HOME/tools/bin
```

3. Install Appium Python Client

Install the Python client library for Appium:

```
pip install Appium-Python-Client
```

Step 2: Identify App Details

1. Using Command-Line (ADB)

1. Connect your Android device or emulator to your computer.
2. Use the following command to list connected devices:

```
adbdevices
```

3. Start the app and find its package and activity name:

```
adb shell dumpsys window | grep -E "mCurrentFocus"
```

Example output:

```
mCurrentFocus=Window{d0a3f1b u0 com.example.app/.MainActivity}
```

- Package Name: com.example.app
- Activity Name: .MainActivity

2. Using Appium Inspector

1. Launch Appium Inspector.
2. Enter the following details:
   - Remote server address (e.g., http://127.0.0.1:4723)
   - Desired capabilities, such as:

```json
{
  "platformName": "Android",
  "deviceName": "emulator-5554",
  "app": "/path/to/your/app.apk",
  "appPackage": "com.example.app",
  "appActivity": ".MainActivity"
}
```

Step 3: Installing and Activating Appium Plugins

1. Install Appium Plugins

To install a plugin, use the following command:

```
appium plugin install --source=npm appium-plugin-name
```

2. Activate Installed Plugins

Activate a plugin using the following command:

```
appium plugin activate appium-plugin-name
```

3. Start Appium Server with Plugins

```
appium --use-plugins=plugin1,plugin2
```

Step 4: Python Automation Script

Here is a Python script to scrape and interact with a mobile app using Appium:

```python
from appium import webdriver
from appium.webdriver.common.touch_action import TouchAction

# Desired Capabilities
caps = {
    "platformName": "Android",
    "deviceName": "emulator-5554",
    "app": "/path/to/your/app.apk",
    "appPackage": "com.example.app",
    "appActivity": ".MainActivity"
}

# Initialize Appium Driver
driver =
webdriver.Remote('127.0.0.1:4723',options=UiAutomator2Options().load_capabilities(desired_caps))

# Function to Swipe
def swipe(driver, direction, duration=800):
    size = driver.get_window_size()
    width = size['width']
    height = size['height']

    if direction == 'up':
        driver.swipe(width // 2, height // 2, width // 2, height // 4, duration)
    elif direction == 'down':
        driver.swipe(width // 2, height // 4, width // 2, height // 2, duration)
    elif direction == 'left':
        driver.swipe(width // 4, height // 2, width // 2, height // 2, duration)
    elif direction == 'right':
        driver.swipe(width // 2, height // 2, width // 4, height // 2, duration)

# Launch App
def launch_app():
    driver.launch_app()
```

Step 4: Python Automation Script

Here is a Python script to scrape and interact with a mobile app using Appium:

```python
# Fetch and Print UI Elements
def fetch_elements():
    elements = driver.find_elements_by_xpath("//android.widget.TextView")
    for element in elements:
        print(element.text)

# Example Usage
launch_app()
swipe(driver, 'up')
fetch_elements()

# Quit Driver
driver.quit()
```

Code Explanation

- Desired Capabilities: Specifies the app and device details.
- Swipe Function: Performs swipe gestures (up, down, left, right).
- Fetch Elements: Dynamically retrieves and prints all text elements on the screen.
- Driver Quit: Closes the Appium session after execution.

Start automating your first app today and explore the endless possibilities of mobile automation with Appium. Remember, practice makes perfect!

Follow me for more updates, tips, and insights on Data 💻🌟