# scrcpy\_ui\_launcher.py 가이드 문서

## 패시전 조건

1. \*\*Python 3.x\*\* 설치

2. \*\*Python 패키지 설치\*\*

```bash

pip install pillow opencv-python pyautogui pandas numpy

```

3. \*\*ADB (Android Debug Bridge)\*\* 설치 및 환경변수 등록

4. \*\*scrcpy 설치\*\*

- https://github.com/Genymobile/scrcpy/releases 에서 Windows 버전 다운로드

- `scrcpy.exe` 가 환경변수 PATH 에 포함되어야 합니다.

5. \*\*이미지 기반 테스트 스크립트\*\*

- 버튼 누르면 `python script.py --device-id <device>` 형식으로 실행되는 것

## 실행 방법

1. 스크립트 실행

```bash

python scrcpy\_ui\_launcher.py

```

2. 디버이스 목록 새로고침

- `↻ Refresh Devices`

3. 모든 디바이스 scrcpy 열기

- `① Run scrcpy on all devices`

4. APK 선택

- `② Select APK`

5. APK 설치 및 실행

- `③ Install && Launch APK`

6. 테스트 스크립트 선택

- `④ Select Test Script`

7. 테스트 시작

- `⑥ Run Selected Test`

8. 결과 리포트 저장

- `⑦ Save Report`

9. scrcpy 강제 종료

- `✖ Stop All scrcpy`

10. 특정 디바이스만 열기

- 목록에서 선택 후 `▶ Run Selected Device`

## 결과 파일

- 리포트 .txt 파일을 저장합니다.

## 주의 사항

- scrcpy 는 `--no-audio` 옵션으로 실행

- 디바이스 수가 많을 경우 scrcpy 창이 두둥두둥 떨어진다고 거의로 확인

- 테스트 시 scrcpy에서 터미널만 파견해 종료

- 테스트 스크립트는 `--device-id` 파라미터 받는 형식이어야 합니다.

## 테스트 스크립트 작성 과 사전 준비 가이드

### 작성 방식 (excel\_image\_automation.py 기준)

1. \*\*파로세스 인식 기능\*\*을 위한 `cv2.matchTemplate` 구현

2. 매 테스트 요소는 Excel 파일에 기록

- 값: `image\_path`, `action`, `offset\_x`, `offset\_y`, `wait\_sec`, `write`

3. 터미널 촬영 (가장 위) 구경이나 `scrcpy` 모델의 모델화 필요

4. 스크립트는 `--device-id` 파라미터 필요

5. 발생 리케스트 결과는 HTML 형식으로 저장

### 사전 준비

- `adb devices` 로 디바이스 연결 확인

- `scrcpy` 연결 후 화면 보이는 상황이어야 함

- Excel 파일에 이미지 경로가 정확히 입력되어 있어야 함

### 실행 예

```bash

python excel\_image\_automation.py --device-id emulator-5554

```

---

**자동화 스크립트 작성 요령**

1. **동작 목적부터 명확히 정의하기**
   * 예: 로그인 자동화, 특정 UI 검사, 버튼 반복 클릭 등
2. **입력 데이터는 외부 파일(예: Excel)로 분리**
   * 유지 보수가 쉬움
   * 여러 기기나 시나리오에 확장 가능
3. **단계별 로그 출력하기**
   * 어떤 이미지가 인식되었는지, 어떤 동작이 수행되었는지 명시적으로 출력
   * 실패 시 원인 추적이 용이함
4. **이미지 매칭 실패에 대비한 예외 처리**
   * None 리턴 시 대기 후 재시도 또는 다음 항목으로 스킵
5. **스크립트는 항상 파라미터 기반으로 설계**
   * 예: --device-id 외에도 --excel-path, --headless 등 확장 고려
6. **테스트 실행 시 GUI 피드백 제공**
   * cv2.rectangle 등으로 인식 영역을 하이라이트
   * 시각적 디버깅에 매우 유용함
7. **스크립트는 최대한 모듈화할 것**
   * 반복되는 기능(예: 이미지 인식, 클릭, 타이핑 등)은 함수로 분리
8. **리포트는 HTML 또는 CSV 등 구조화된 형식으로 출력**
   * 테스트 관리 및 분석을 위한 기반 데이터로 활용