AutoTest IDE简介

[1 AutoTest IDE 基本结构](#_Toc530477166)

[2 被测设备查看，编辑及连接](#_Toc530477167)

[3 AutoTest IDE测试脚本的编写](#_Toc530477168)

[3.1 新建一个测试case](#_Toc530477169)

[3.2 操作录入](#_Toc530477170)

[3.2.1 Tbox事件输入](#_Toc530477171)

[3.2.2 keyevent事件输入](#_Toc530477172)

[3.2.3 控件事件输入](#_Toc530477173)

[3.3 断言的录入](#_Toc530477174)

[4 AutoTest脚本的运行及报告显示](#_Toc530477175)

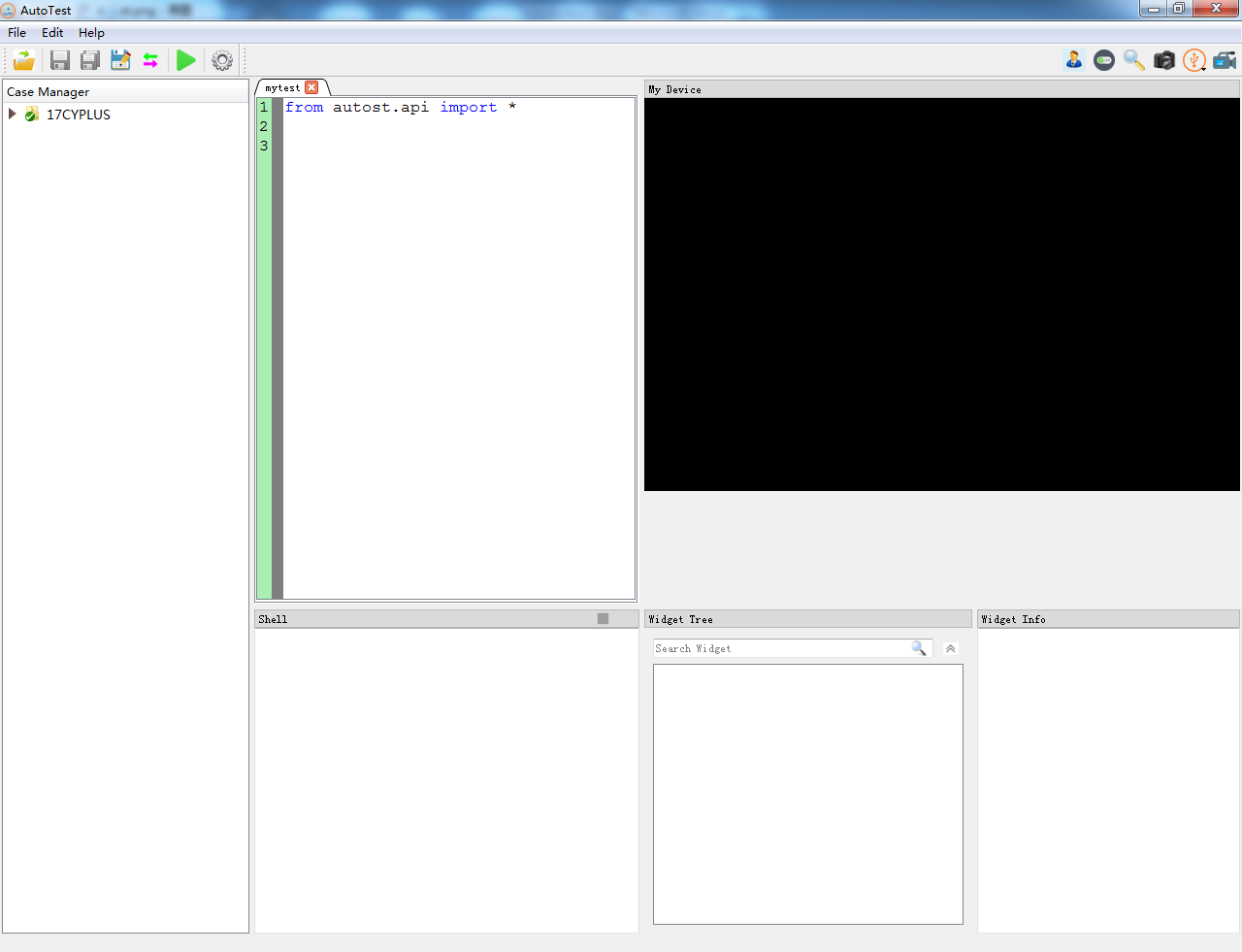
[4.1 运行一个测试case，及查看报告](#_Toc530477176)

[4.2 报告分析](#_Toc530477177)

[4.3 使用命令行运行脚本](#_Toc530477178)

# 1 AutoTest IDE 基本结构

AutoTest IDE 主要由代码编辑窗口、设备画面窗口、设备画面控件树窗口、控件信息展示窗口、case管理窗口及运行日志显示窗口组成。各窗口的功能简要介绍如下：



代码编辑窗口

1、编辑测试脚本

case管理窗口

1. case新建、配置

2. case运行

3. show report

4. 文件及目录级操作

运行日志显示窗口

1、脚本运行时，日志的显示

设备画面控件树窗口

1.显示当前设备画面上 的所有控件及对控件的相关操作

控件信息展示窗口

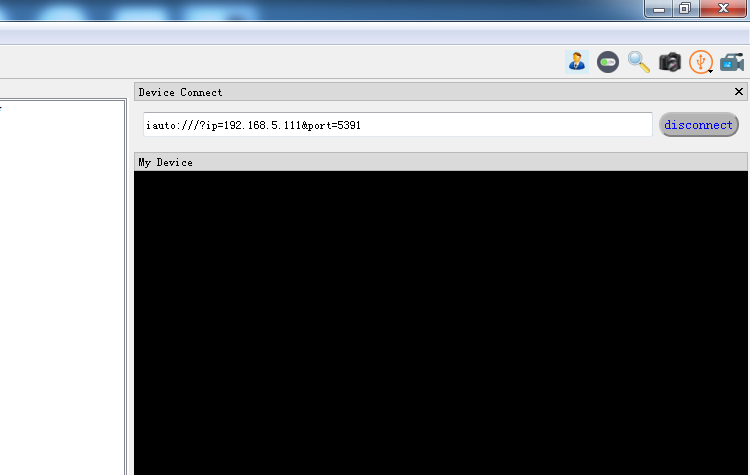
1、展示某个选中的控件信息（如控件大小、位置、名称、类别等）

**设备画面窗口**

**1、显示接入的设备画面**

# 2 被测设备查看，编辑及连接

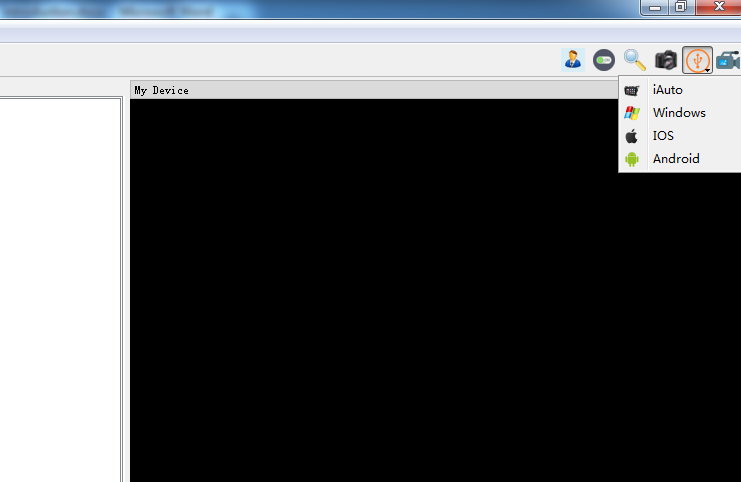
点击图标，可查看 默认连接设备。



点击图标，

连接其他设备

**查看连接设备，编辑连接设备的手顺**



**连接其他设备的手顺**

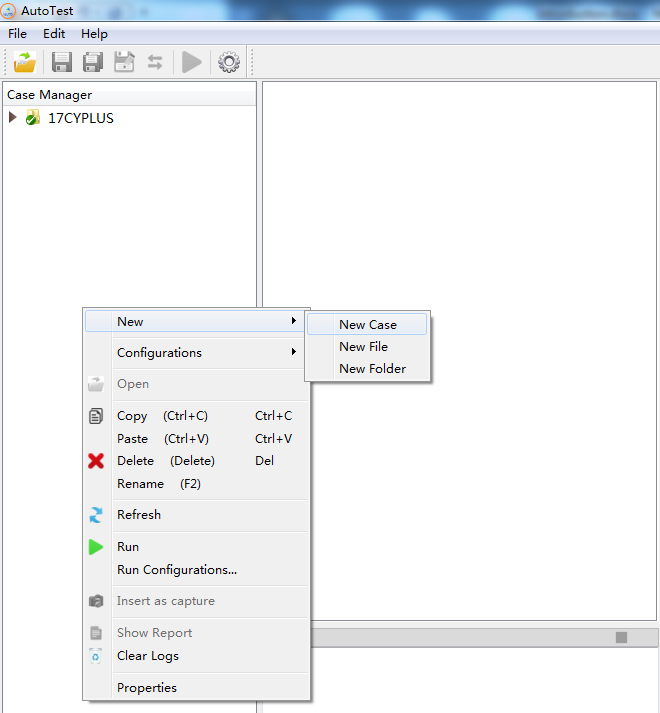
# 3 AutoTest IDE测试脚本的编写

AutoTest测试是一种以一系列车辆信号操作、车机按键操作、界面控件操作为输入，测试界面输出是否正确的一种测试方法。

AutoTest测试脚本编写的基本流程：

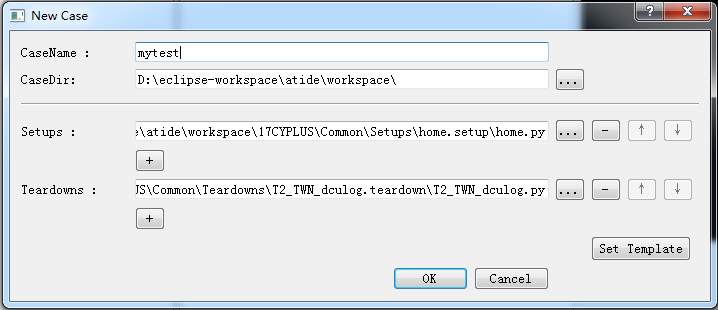
* 新建一个测试case
* 操作录入
* 断言的录入

## 3.1 新建一个测试case



**新建case的手顺（step1）**

1、case管理窗口中，鼠标右键New->New Case



case目录，必填

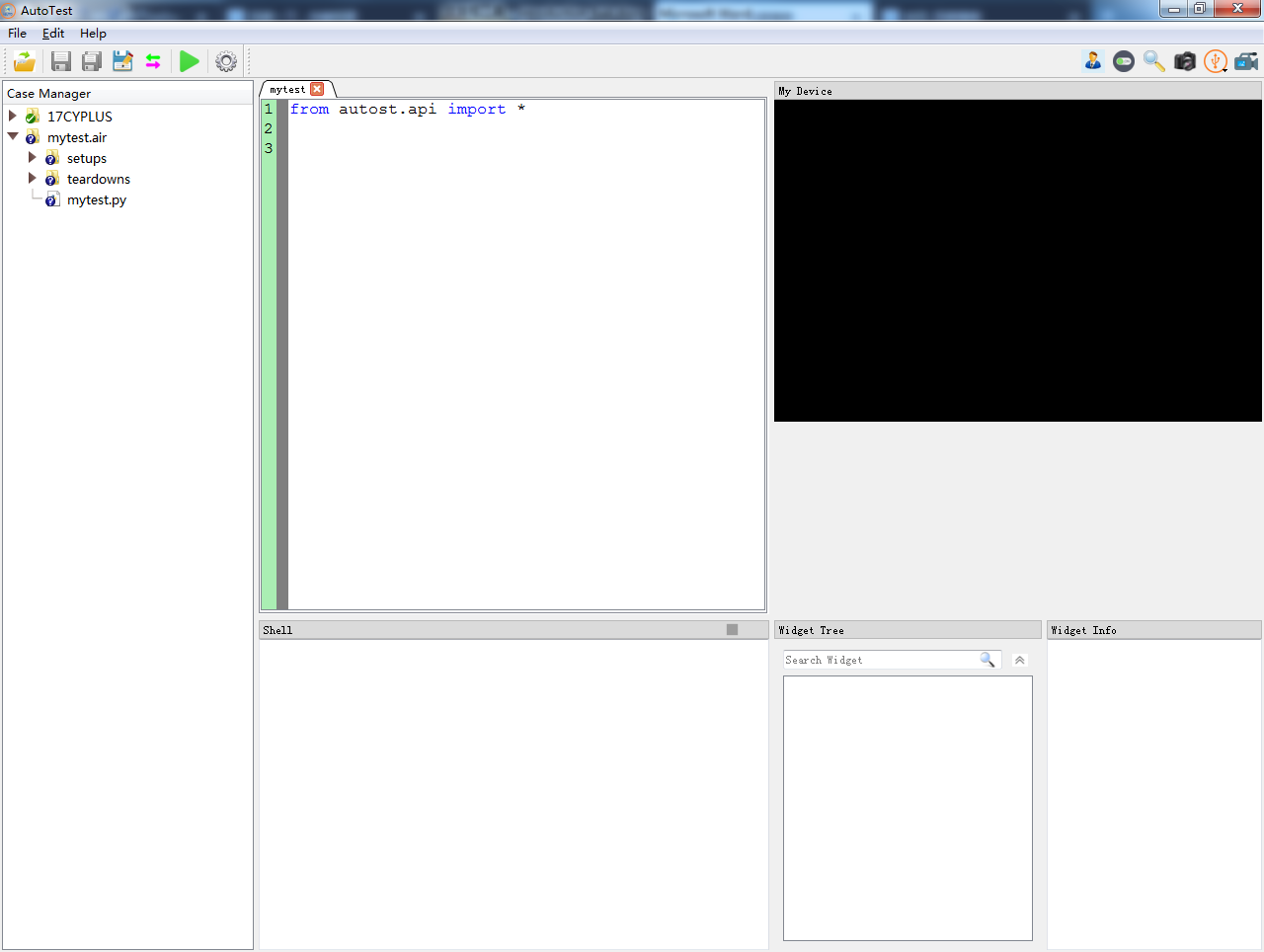
确定

teardowns，选填

setups，选填

case名称，必填

**新建case的手顺（step2）**



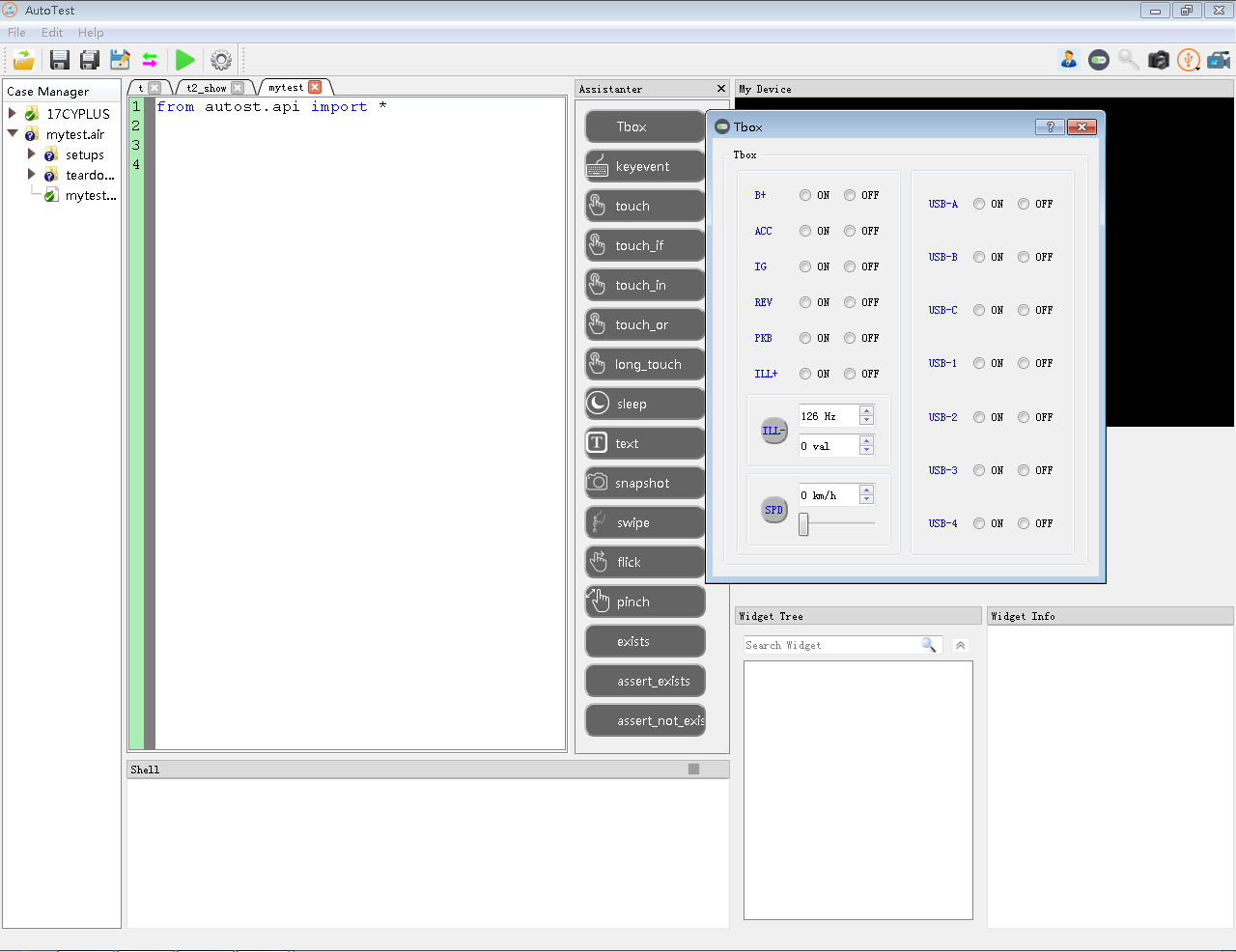
名称为mytest.air的case创建完毕

## 3.2 单个事件的代码录入

AutoTest使用Assistanter面板，定义了三类事件的代码录入：Tbox事件（车辆信号）的代码录入、控件操作(触摸屏的按钮等控件)的代码录入、keyevent操作(键盘操作)的代码录入。

### 3.2.1 Tbox事件的代码录入

step2：点击Assistanter按钮



step5：代码编辑窗口上自动添加‘bup\_on()’字样，B+ON事件的代码录入完成

step4：选中B+ ON

**Tbox事件代码录入的手顺**

**（以B+ON事件为例）**

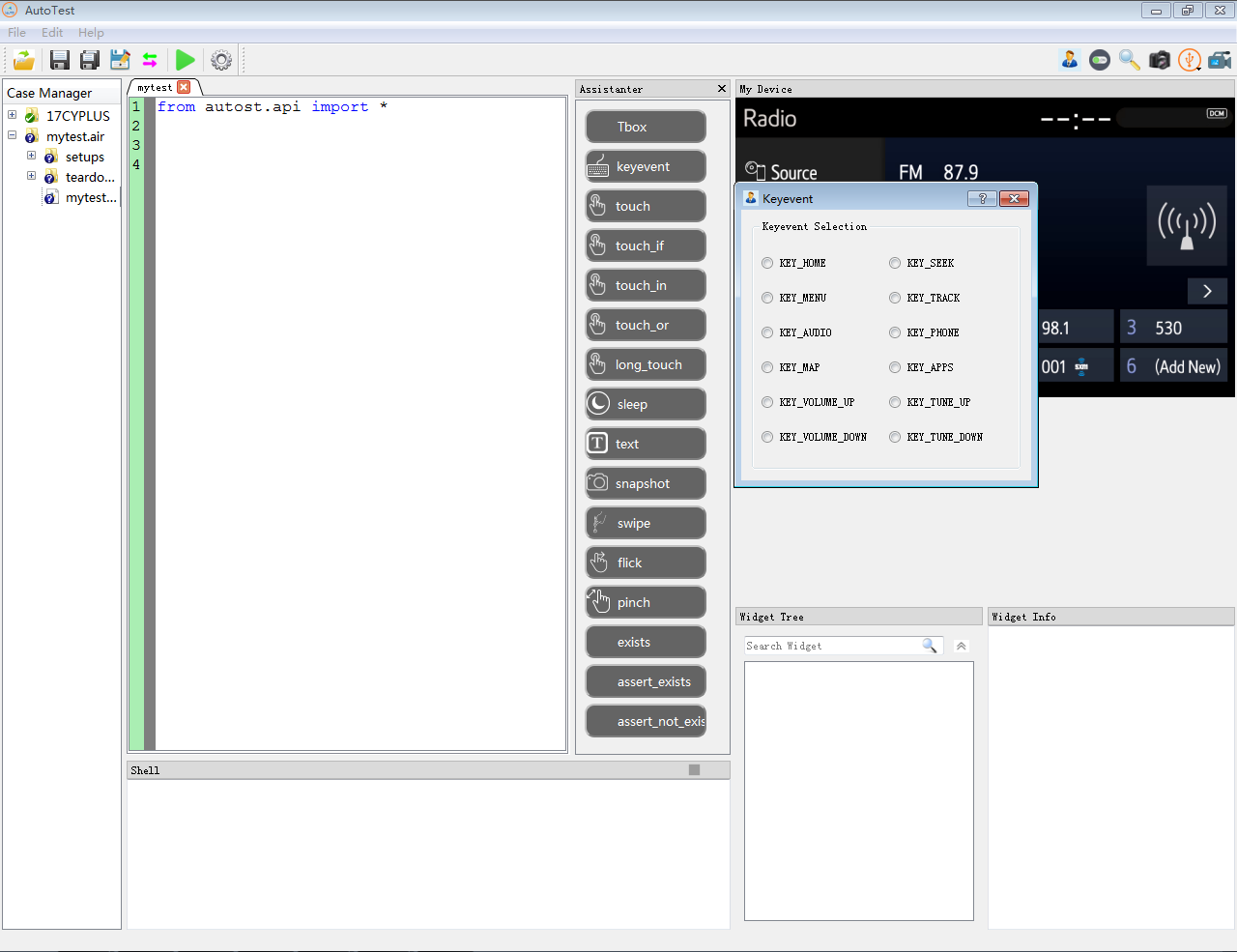
step3：点击Tbox按钮

step1：新建测试case成功后，在代码编辑窗口，光标放置在现有代码下另起一行的位置

### 3.2.2 keyevent事件输入

step2：点击Assistanter按钮

step1：新建测试case成功后，在代码编辑窗口，光标放置在现有代码下另起一行的位置



step5：代码编辑窗口上自动添加‘keyevent(KEY\_MENU)’字样，KEY\_MENU事件的代码录入完成

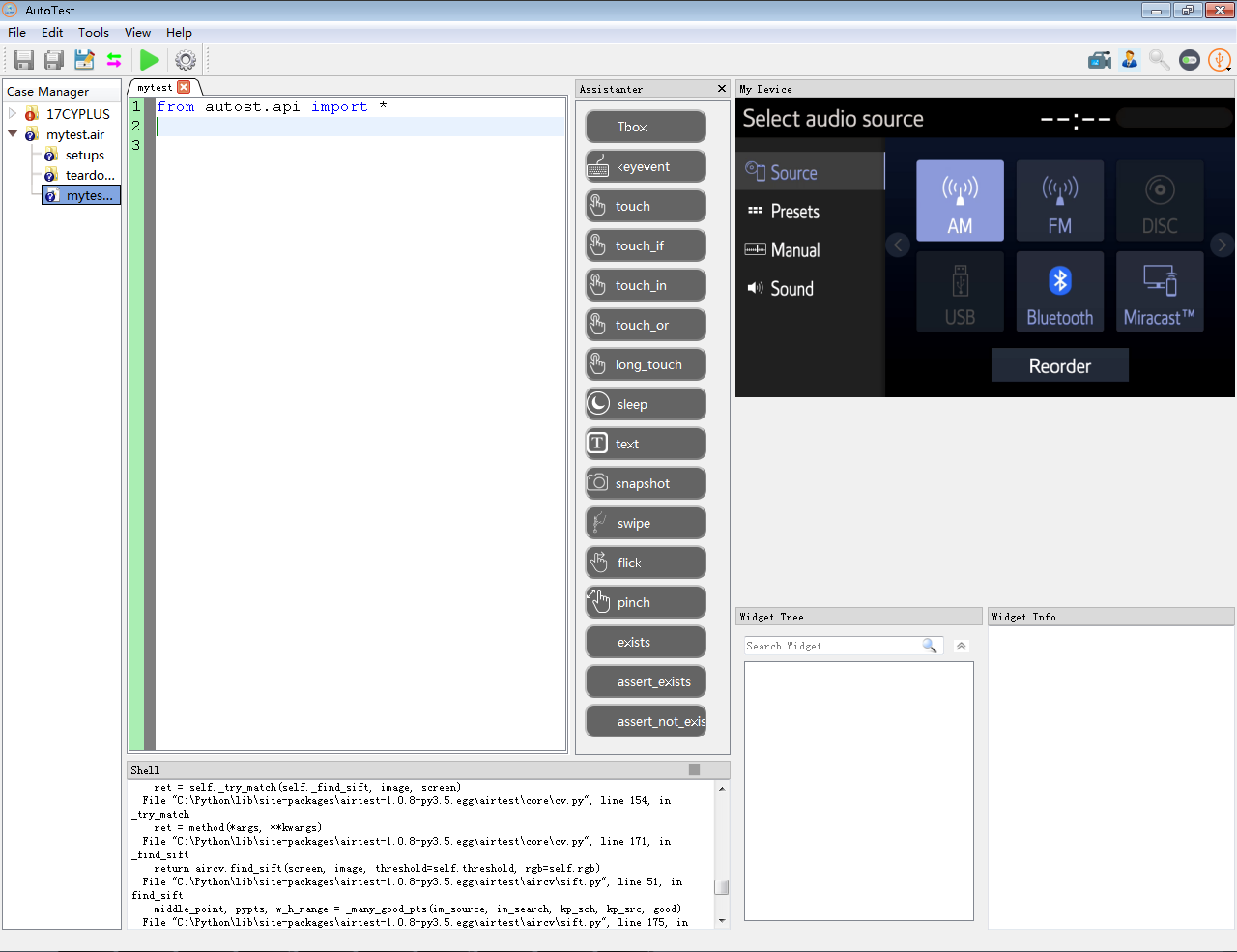
**Keyevent事件代码录入的手顺（以KEY\_MENU事件为例）**

step4：选中KEY\_MENU

step3:点击keyevent按钮

### 3.2.3 控件事件输入

step2：点击Assistanter按钮



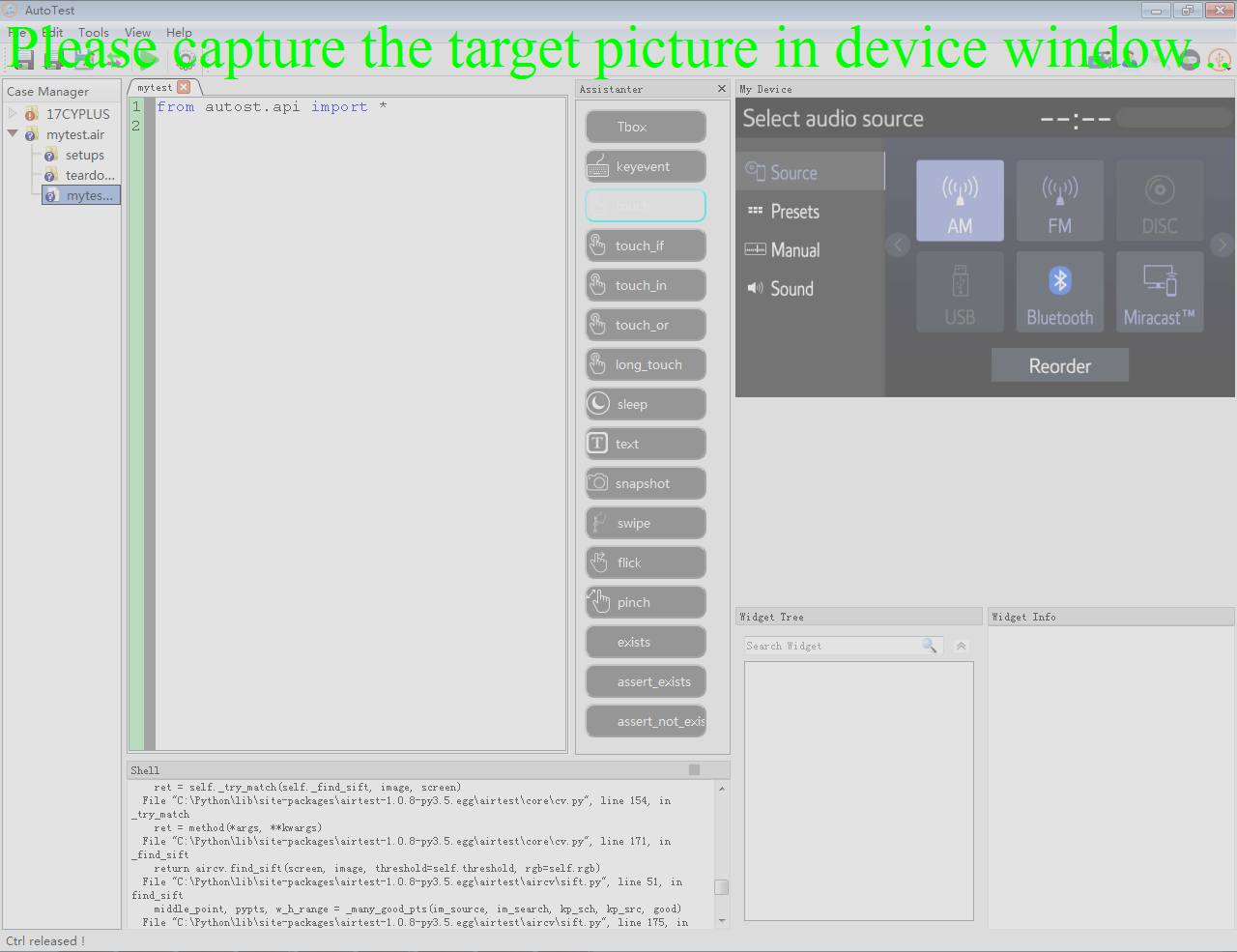
step3:点击touch按钮

**控件事件代码录入的手顺**

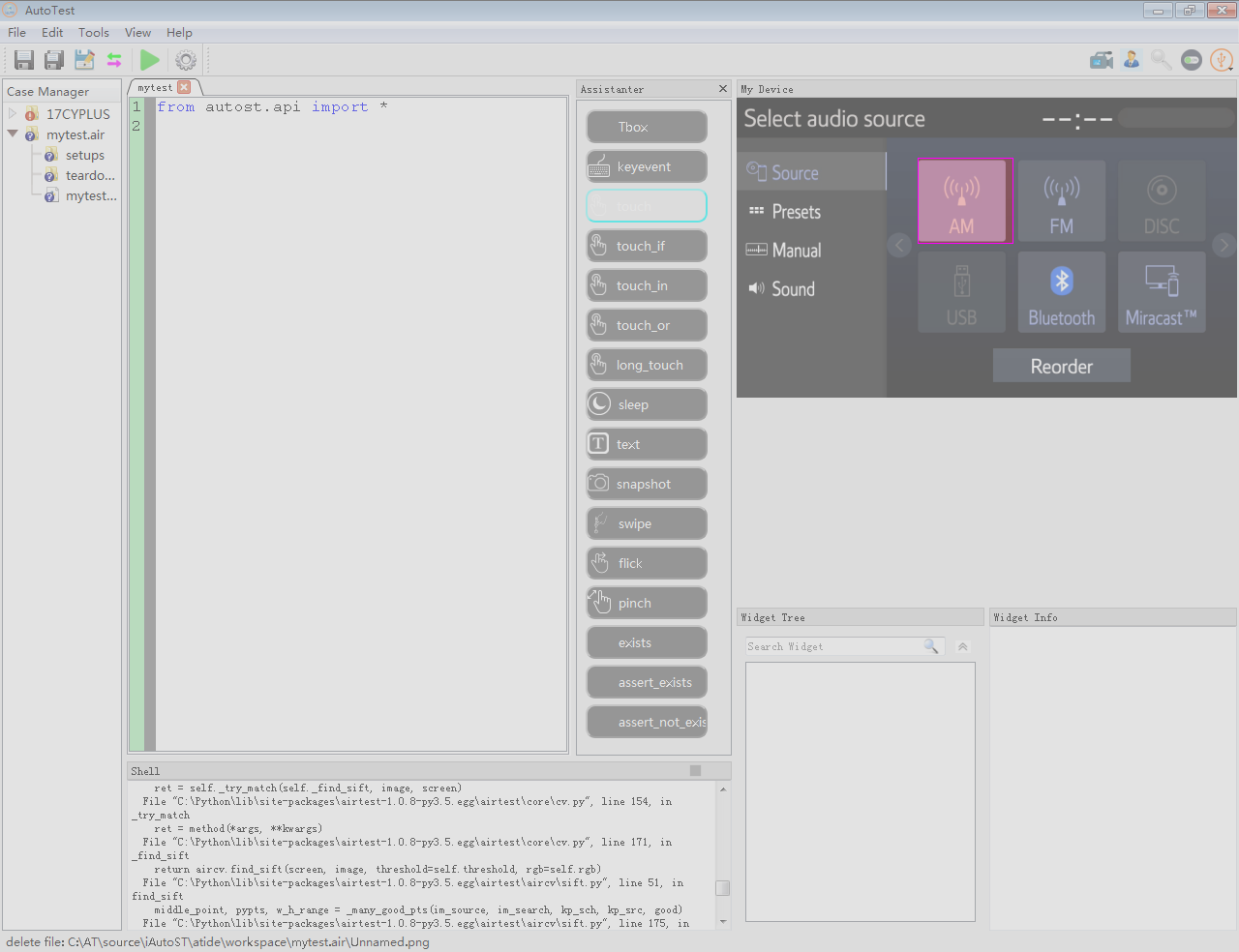
**（以touch AM事件为例）**

**(step1-3)**

step1：新建测试case成功后， 在代码编辑窗口，光标放置在现有代码下另起一行的位置



接上图，step1-3操作成功后，界面如图显示。

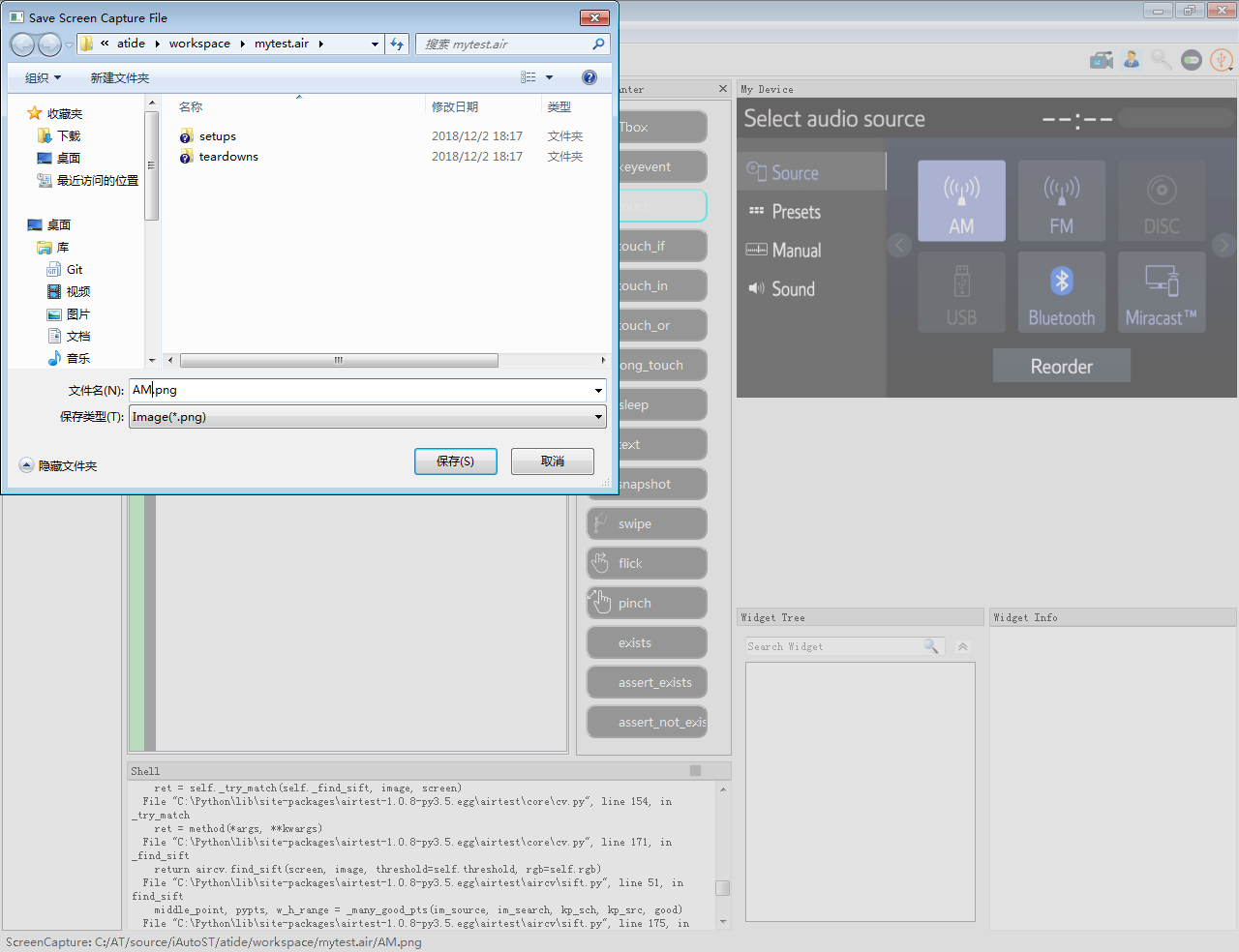


step4:鼠标从AM图标的左上角向右下角拖动，选中的矩形区域变成粉红色。

**控件事件代码录入的手顺**

**（以touch AM事件为例）**

**(step4)**

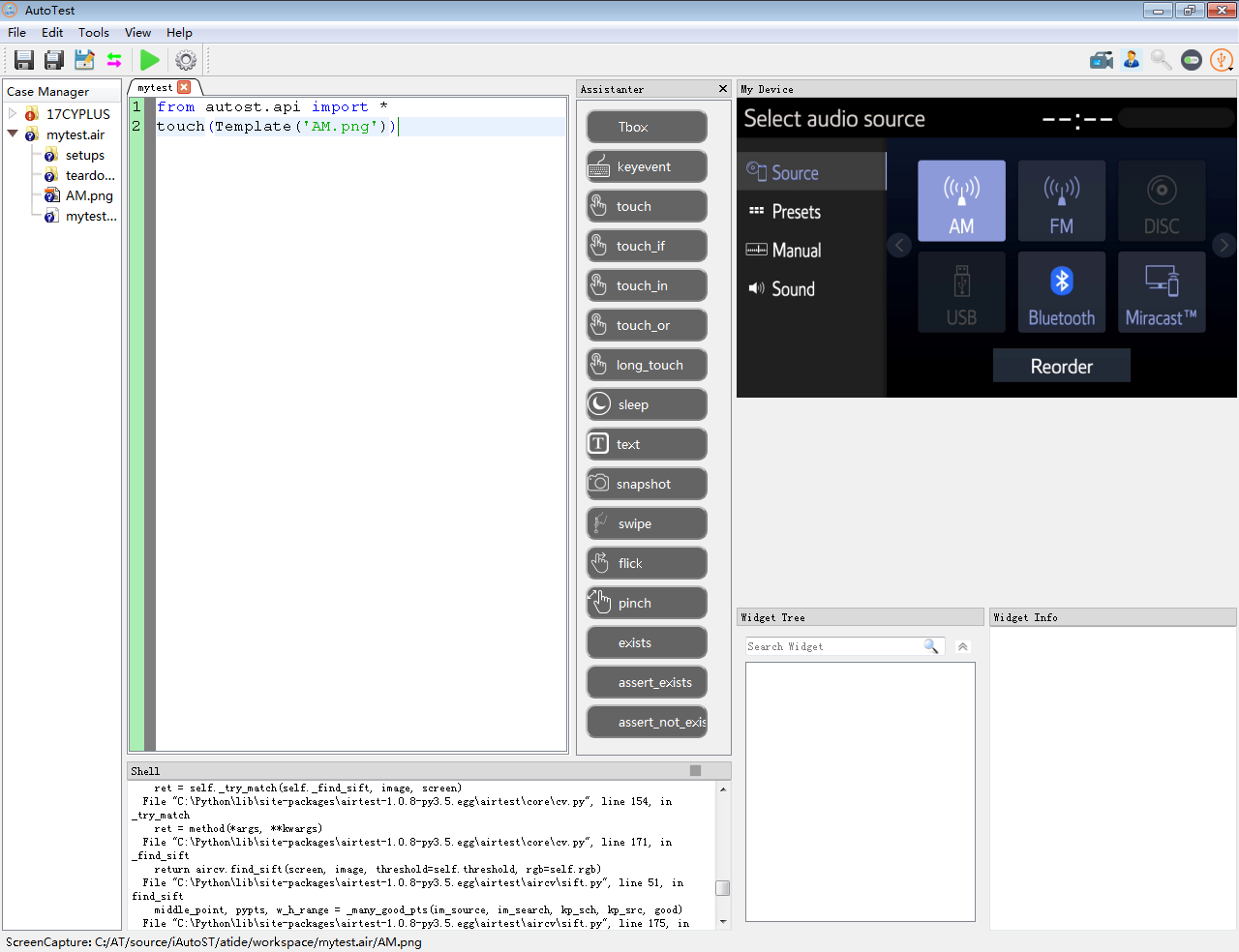


step5: 接step4,松开鼠标后,出现"Save Screen Capture File"对话框,输入文件名,保存。

**控件事件代码录入的手顺**

**（以touch AM事件为例）**

**(step5)**



完成后，代码编辑窗口将自动出现

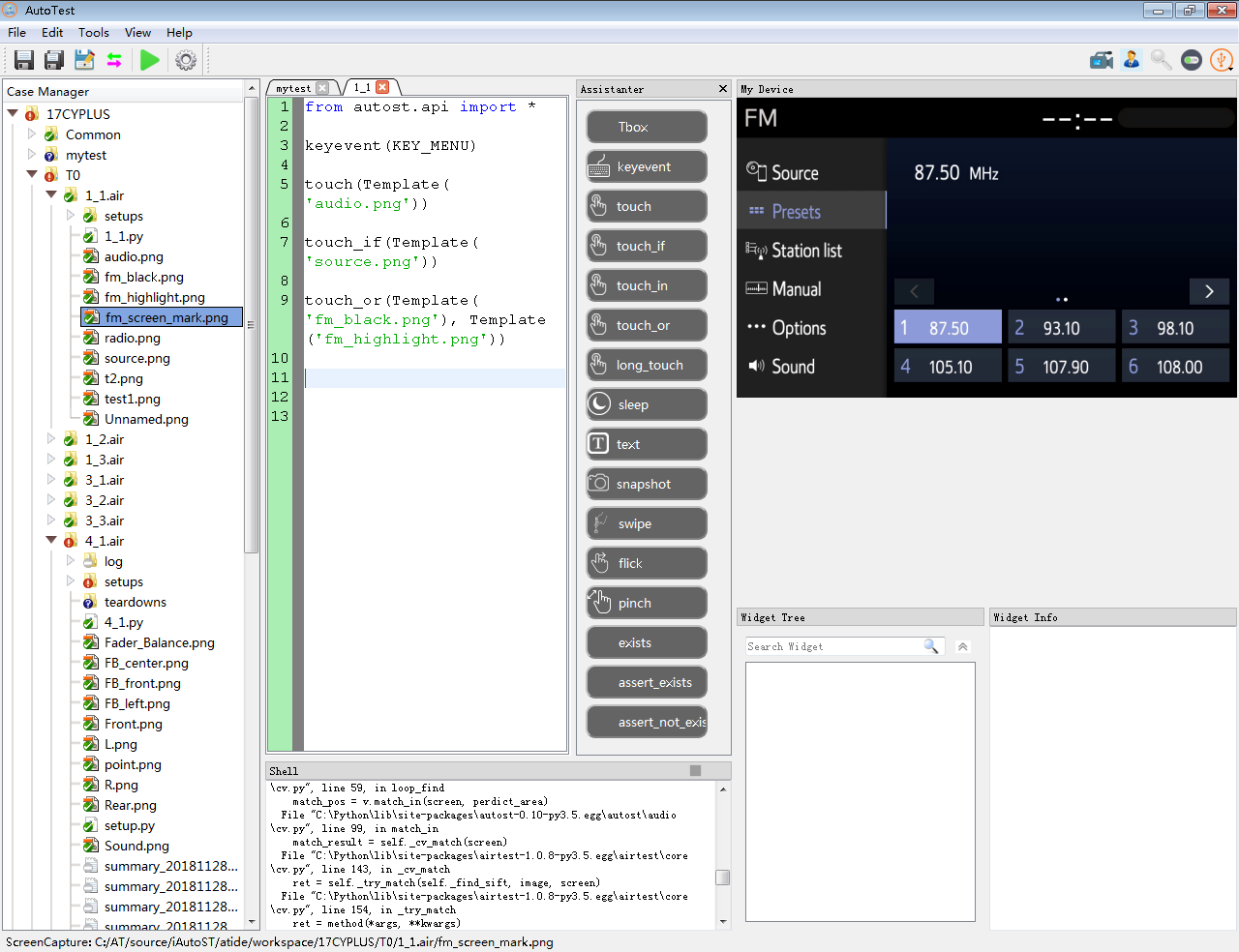
“touch(Template("AM.png"))”字样，touch am事件录入完毕。

**控件事件代码录入的手顺（完结篇）**

**（以touch AM事件为例）**

## 3.3 断言的录入

step2：点击Assistanter按钮



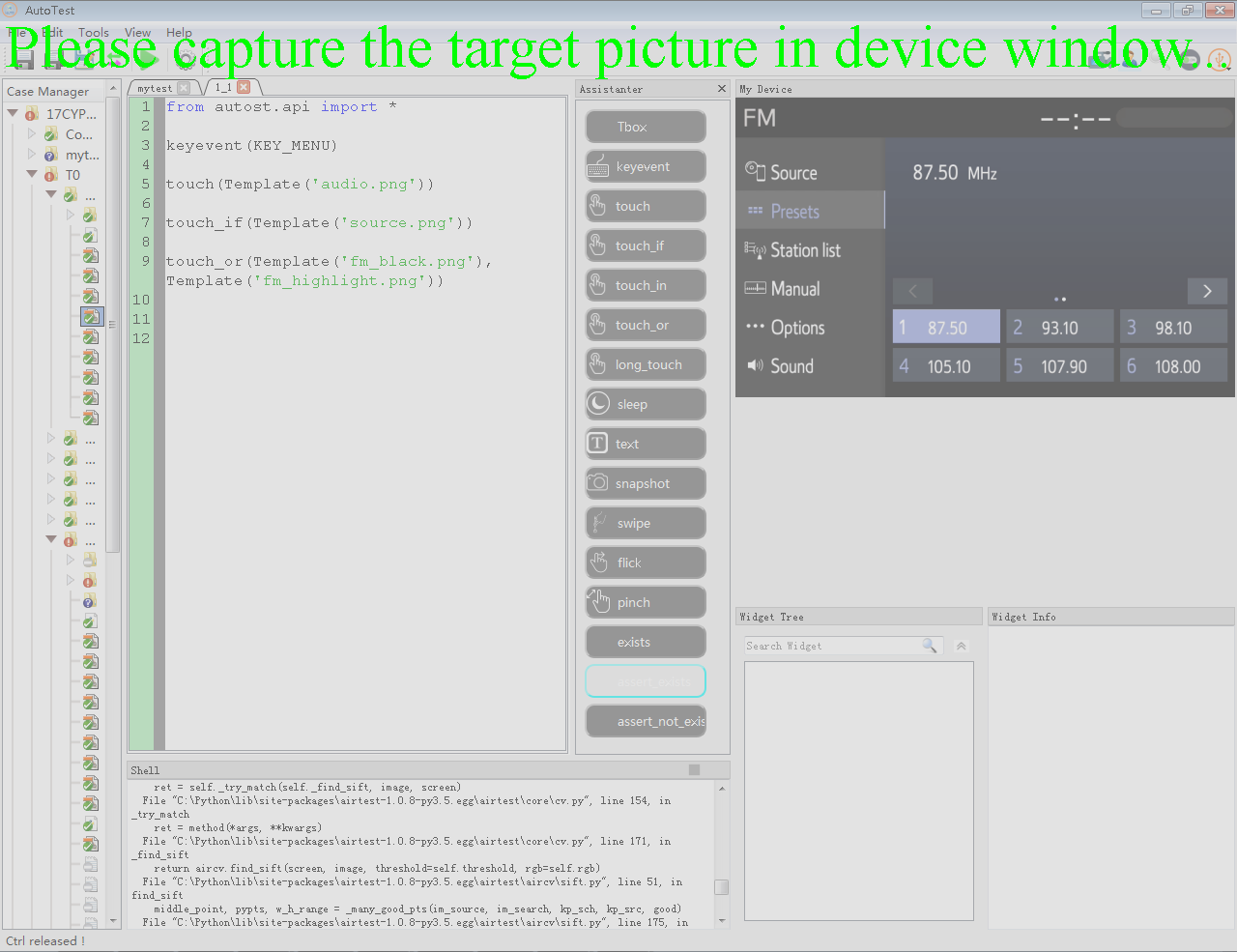
**断言的输入手顺(step1-3)**

**(以判断界面上是否出现FM图标为例)**

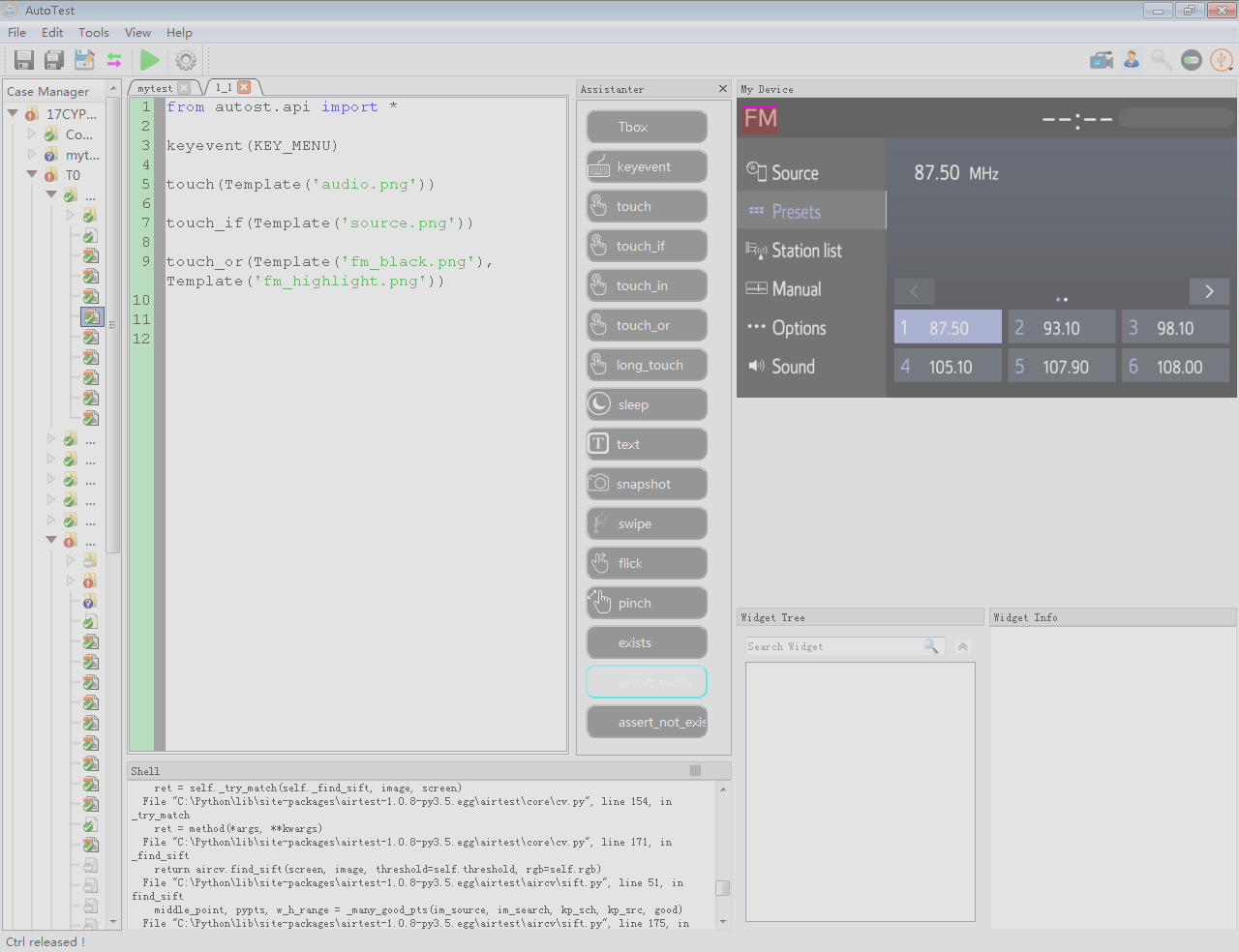
step1:

在代码编辑窗口，光标放置在现有代码下另起一行的位置。

step3:点击assert\_exists按钮



step1-3完成后,界面如图所示.

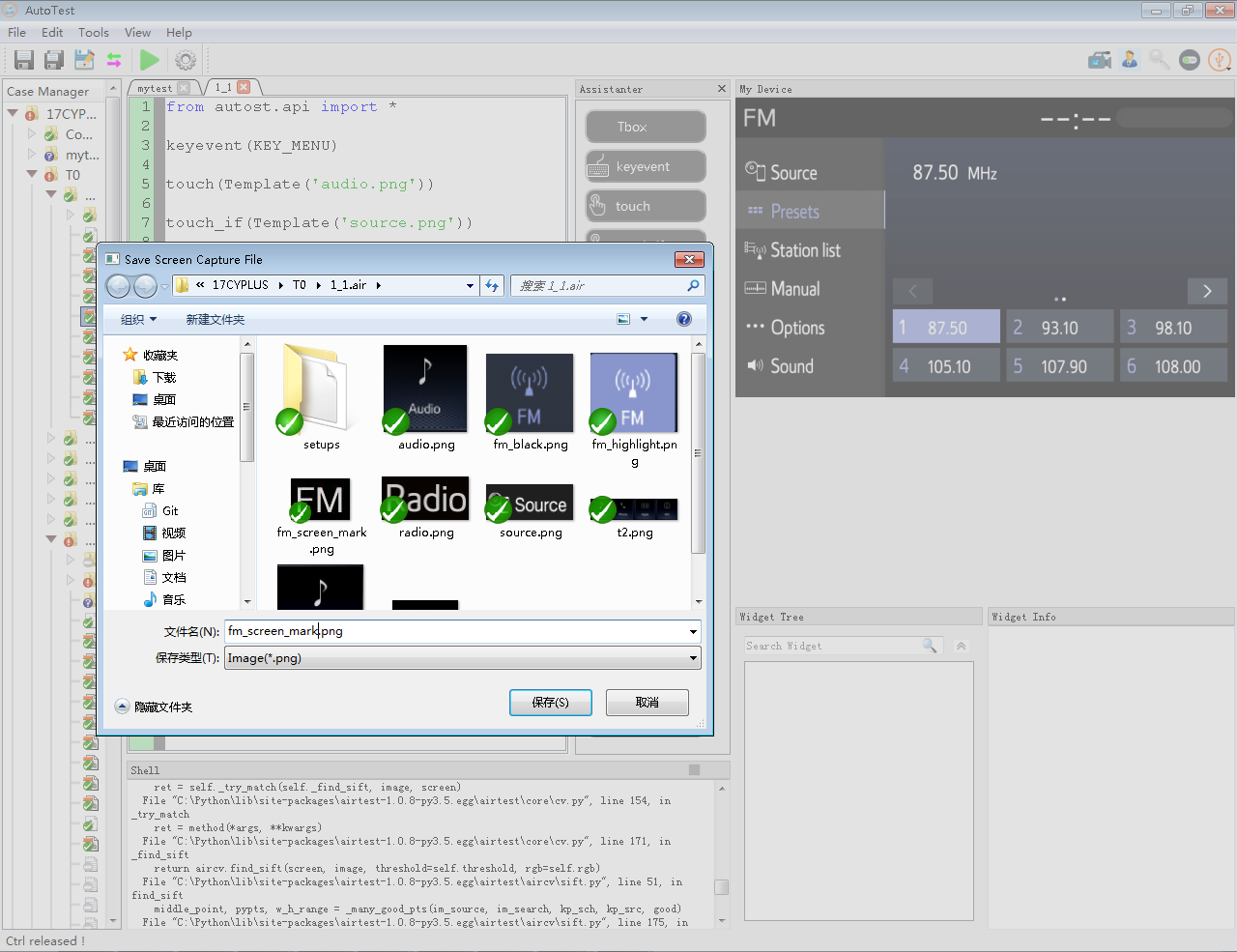


**断言的输入手顺(step4)**

**(以判断界面上是否出现FM图标为例)**

step4:

鼠标从FM图标的左上角向右下角拖动，选中的矩形区域变成粉红色。

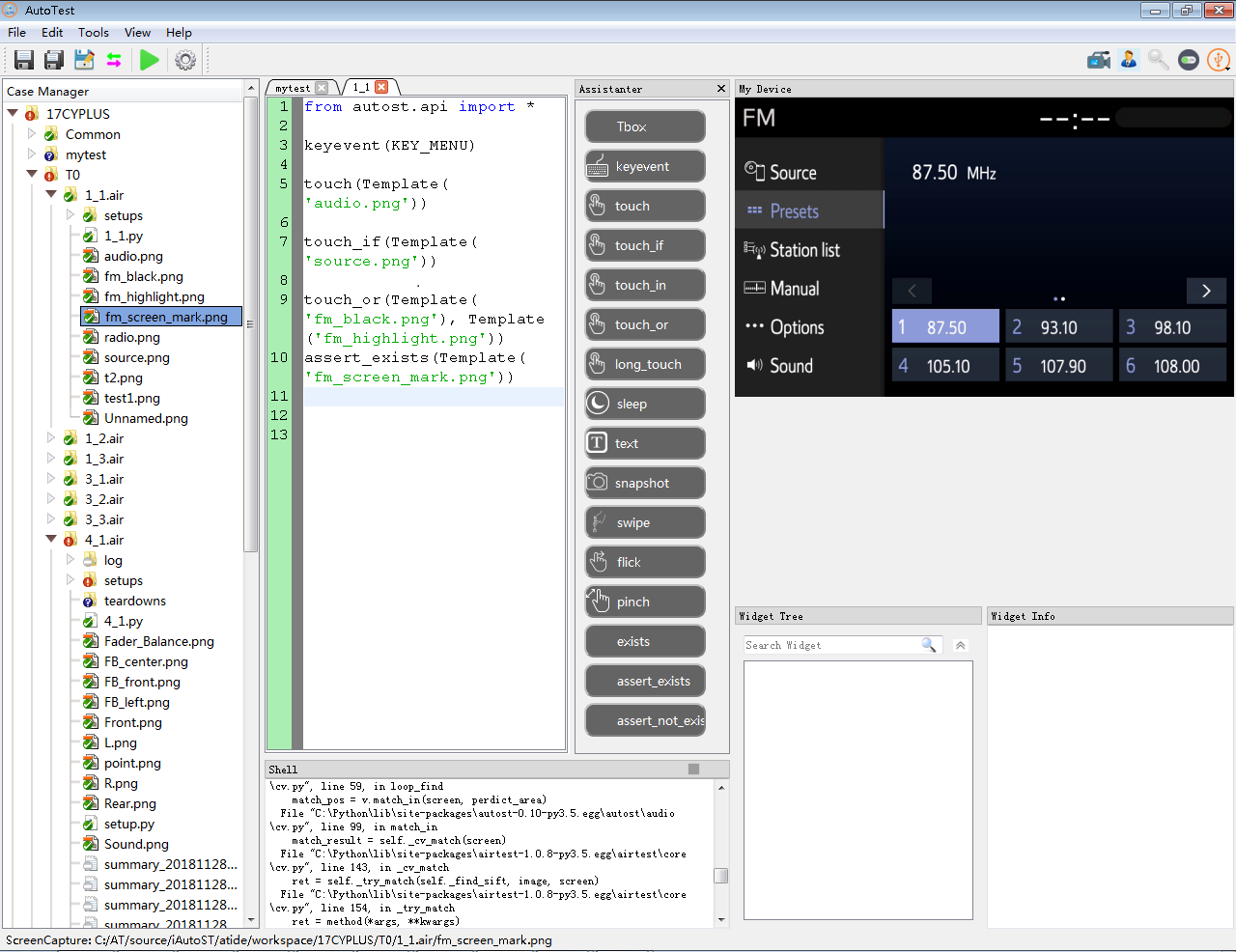


**断言的输入手顺(step5)**

**(以判断界面上是否出现FM图标为例)**

step5:

松开鼠标后,出现"Save Screen Capture File"对话框,输入文件名,保存。



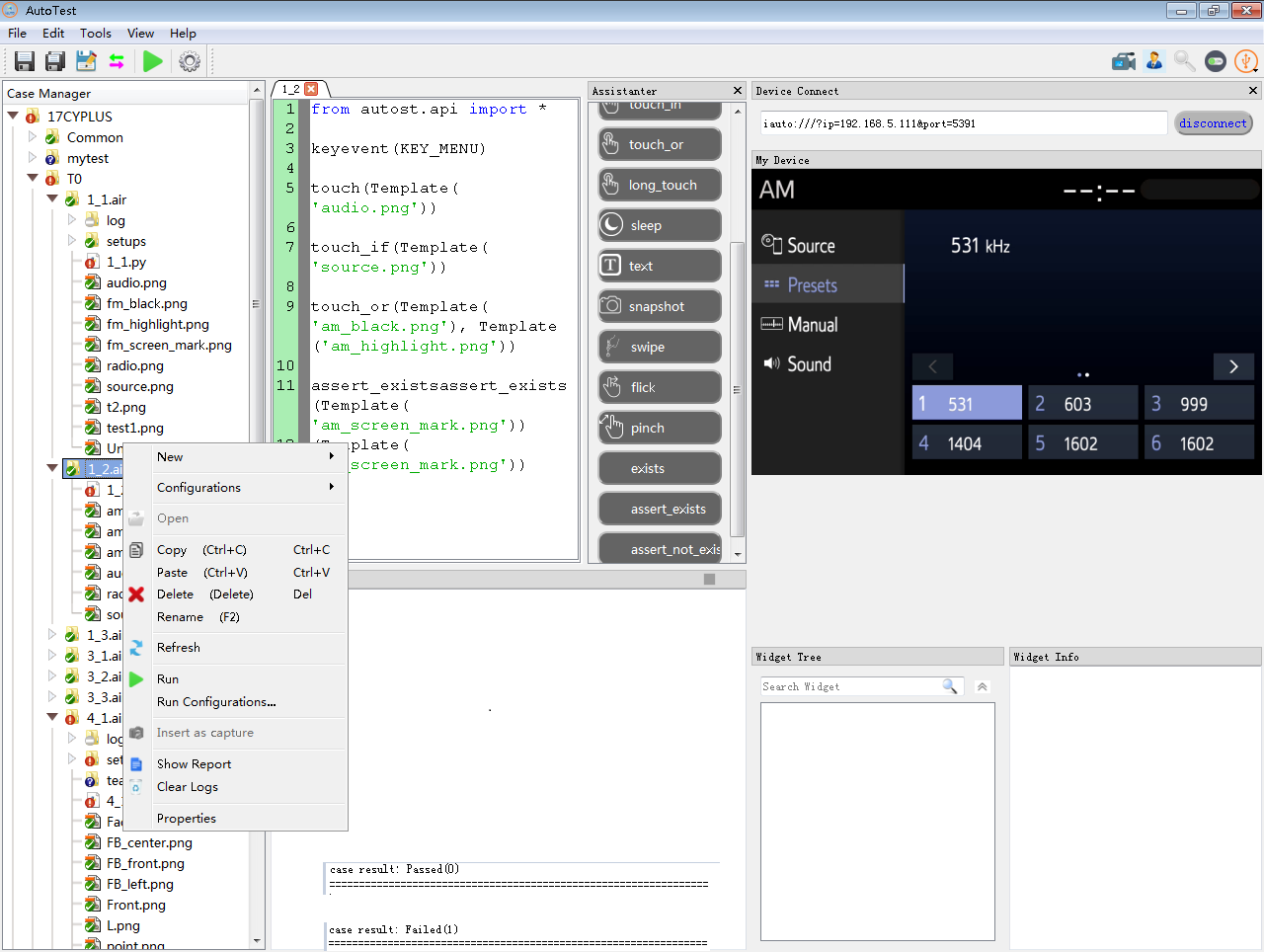
**断言的输入手顺（完结篇）**

**(以判断界面上是否出现FM图标为例)**

以上操作完成后，代码编辑窗口将自动出现“assert\_exists(Template("fm\_screen\_mark.png"))”字样，断言录入完毕。

# 4 AutoTest脚本的运行及报告显示

## 4.1 运行一个测试case，及查看报告



**运行**

case测试ok的log显示

**()**

**（）**

case测试fail的log显示

**()**

**（）**

**日志显示窗口**

**(case运行时log的输出窗口)**

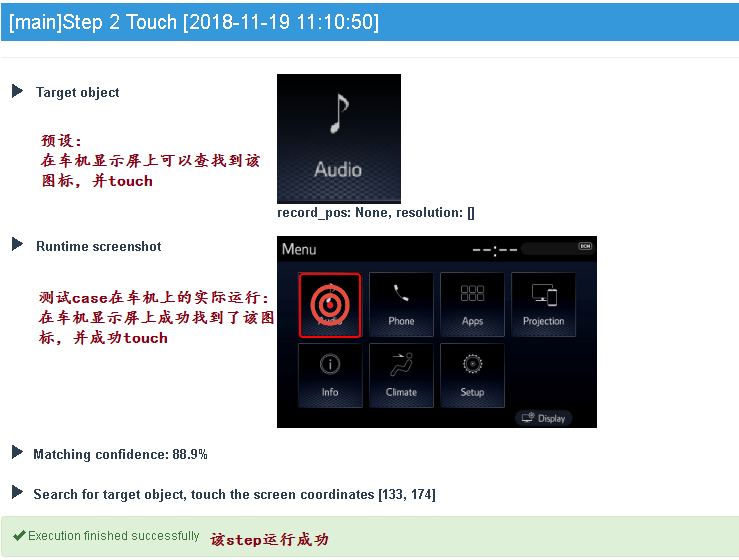
**()**

**（）**

**查看报告**

**运行**

## 4.2 报告分析







## 4.3 使用命令行运行脚本

命令行语法：

python -m autost casedir [--device=DEVICE] [--logdir=LOGDIR] [--rounds=ROUNDS]

参数说明：

casedir: TestCase的路径，或者TestCase集合的路径

--device: 设备URL，默认为"iauto:///?ip=127.0.0.1&port=5391"

--logdir: TestCase的运行日志目录，默认为casedir/log

--rounds: TestCase的执行次数，默认为1（执行1次），-1表示无限循环执行

命令行示例：

1.python -m autost D:\workspace\MyAir\src\AT\workspace\T0\1\_1.air --device="iauto:///?ip=192.168.5.111&port=5391"

2.python -m autost C:\liuxinxing\workspace\AT\source\workspace\T0 --device="iauto:///?ip=192.168.5.111&port=5391" --round=2 --logdir=C:\liuxinxing\workspace\AT\source\workspace\T0\_Report\20180719