Class

XCUIDevice

A proxy that can simulate physical buttons, device orientation, and Siri interaction for an iOS, watchOS, or tvOS device.

iOS | iPadOS | Mac Catalyst | macOS | tvOS | visionOS | watchOS | Xcode 16.3+

```
@MainActor
class XCUIDevice
```

Overview

Use the `XCUIDevice` shared instance to perform the following interactions with a simulated iOS, watchOS, or tvOS device during a UI test:

- Press the volume, home, camera, and action buttons.
- Rotate the device.
- Turn the Digital Crown on a watchOS device.
- Determine whether the iOS device supports pointer interaction.
- Activate Siri.

This example shows a test that determines whether the action button is available on the shared device and, if it is, simulates pressing the button:

```
@MainActor
func testPressingActionButton() throws {
    let device = XCUIDevice.shared
    try XCTSkipUnless(device.hasHardwareButton(.action),
                      "The device doesn't have an action button.")
    let app = XCUIApplication()
    app.launch()
    device.press(.action)
    // Assert that your app responds correctly.
}
```

`XCUIDevice` is available in iOS, watchOS, and tvOS.

Topics

Accessing the current device

`class var shared: XCUIDevice`

The current device.

`var supportsPointerInteraction: Bool`

A Boolean value that indicates if the device supports pointer interaction.

`var supportsHandGestures: Bool`

A Boolean value that indicates if the device supports hand gestures.

Interacting with buttons and the Digital Crown

`func press(XCUIDevice.Button)`

Simulates the user pressing a physical button.

`func hasHardwareButton(XCUIDevice.Button) -> Bool`

Determines whether the device supports the button type you provide.

`enum Button`

A physical button on an iOS device.

`func rotateDigitalCrown(delta: CGFloat)`

Simulates the user rotating the Digital Crown on an Apple Watch by the delta amount.

`func rotateDigitalCrown(delta: CGFloat, velocity: XCUIGestureVelocity)`

Simulates the user rotating the Digital Crown on an Apple Watch by the delta amount and speed you provide.

Performing gestures

`func perform(handGesture: XCUIDeviceHandGesture)`

`enum XCUIDeviceHandGesture`

A hand gesture on a watchOS device.

Rotating and changing location

`var orientation: UIDeviceOrientation`

The orientation of the device.

`var location: XCUILocation?`

The proxy location a test uses to simulate longitude, latitude, and course information for the device.

`class XCUILocation`

A proxy that simulates a device’s location in terms of its longitude, latitude, and course information.

Interacting with the OS

`var system: XCUISystem`

An object that provides an interface to OS-specific properties and actions.

`var appearance: XCUIDevice.Appearance`

The interface style of the device.

`enum Appearance`

Constants that indicate an interface style.

Interacting with Siri

`var siriService: XCUISiriService`

An object that represents the Siri interface on the device.

Deprecated

~~`init()`~~

Creates an instance that represents the current device.

Deprecated

Relationships

Inherits From

`NSObject`

Conforms To

`CVarArg`
`CustomDebugStringConvertible`
`CustomStringConvertible`
`Equatable`
`Hashable`
`NSObjectProtocol`
`Sendable`

See Also

Device simulation

`class XCUISystem`

A proxy that provides an interface to OS-specific properties and actions.

`class XCUISiriService`

A proxy that simulates a device’s Siri interface.