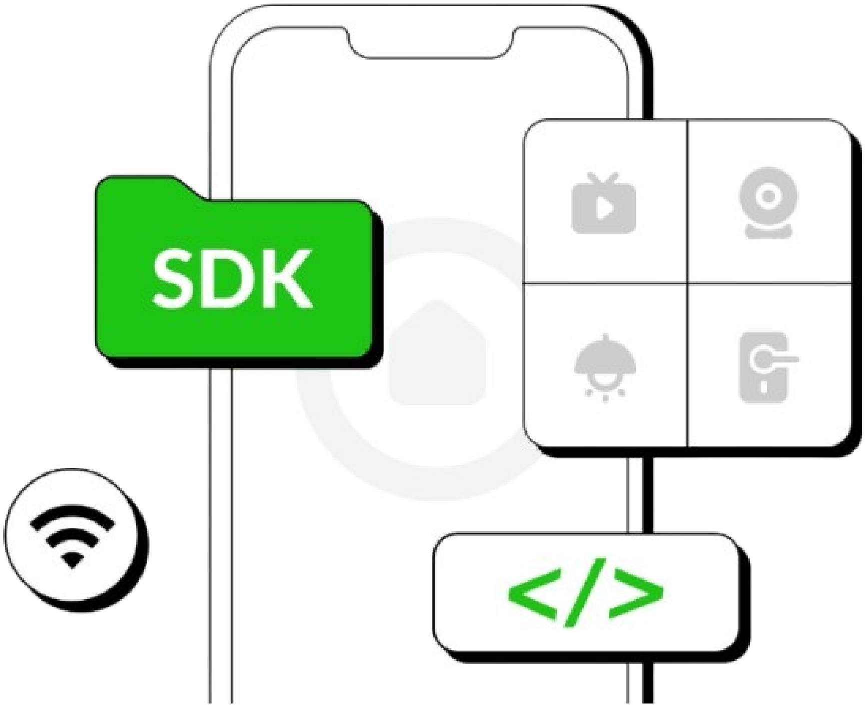
Contents [hide]

tuya

tuya 20240517 Smart App SDK



This topic describes the Wi-Fi Easy Connect (EZ) or SmartConfig mode to pair devices. After a user connects a mobile phone to a router, the router broadcasts is used to communicate and pair the mobile phone with the device. It is easy-to-use, but has compatibility requirements for mobile phones and routers. The success rate is lower than that of hotspot or access point (AP) mode.

Pairing process

1. Before pairing
   - Guide the user to reset the device to Wi-Fi EZ mode, typically identified by a fast-blinking Wi-Fi indicator.
   - Guide the user to connect their phone to a Wi-Fi network, usually a 2.4 GHz Wi-Fi network.
2. Get pairing token and Wi-Fi network credentials
   - The app gets the pairing token by calling the method in the SDK.
   - The app gets the Wi-Fi network credentials (SSID and password) input by the user. The SSID can also be obtained through an API call.
3. Start pairing
   The app calls the pairing method in the SDK to set the Wi-Fi network credentials (SSID and password) and the pairing token.
4. Finish pairing
   The app receives a callback from the SDK and finishes the pairing process.

info

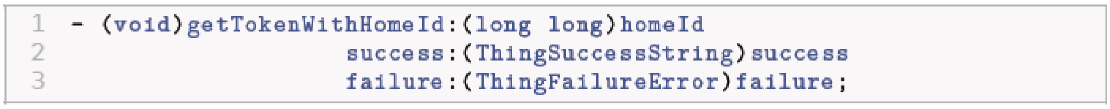
For iOS 14.5 and later, we recommend that you use the AP mode instead of the Wi-Fi EZ mode. The former trumps the latter when it comes to the following aspects:

- Compared with the Wi-Fi EZ mode, the AP mode results in a higher success rate, optimal reliability, and fewer compatibility requirements for mobile phones and routers.
- The app built with Xcode 12.5 cannot send the EZ pairing data packets from iPhone that runs iOS 14.5 or later. In this case, the permission com.apple.developer.networking.multicast must be enabled for the app. This permission must be approved by Apple before it can be enabled. As a temporary solution, you can use an earlier Xcode version, but the AP mode is still recommended.

Get a token

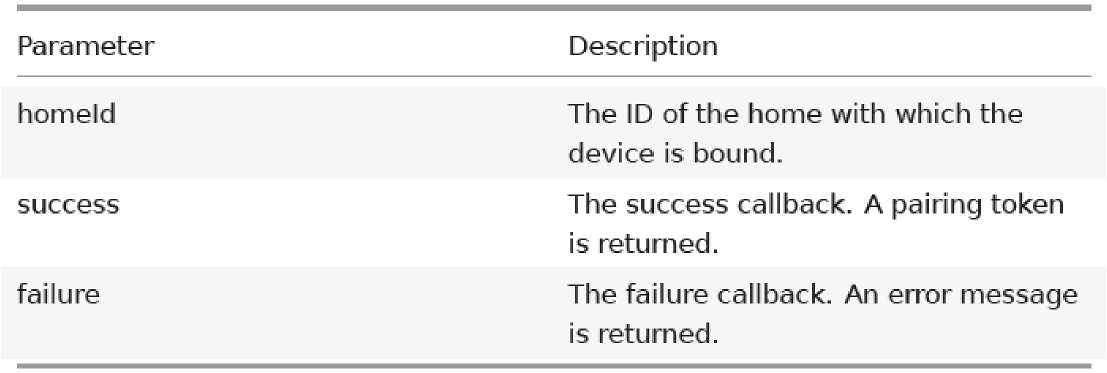
Before the Wi-Fi EZ pairing process, the SDK must get a pairing token from the cloud in the networked state. The token is valid for 10 minutes and expires immediately after the device is paired. A new token must be generated if the device needs to be paired again.

API description



```
- (void)getTokenWithHomeId:(long long)homeId
                   success:(ThingSuccessString)success
                   failure:(ThingFailureError)failure;
```

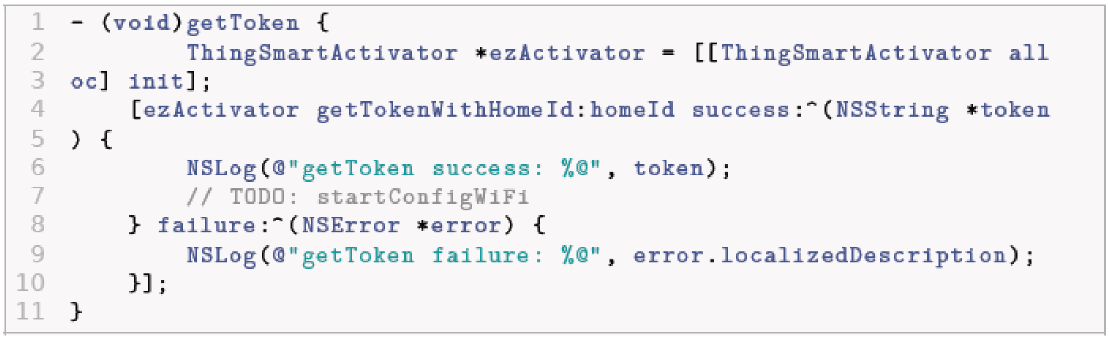
Parameters



Parameter	Description
homeId	The ID of the home with which the device is bound.
success	The success callback. A pairing token is returned.
failure	The failure callback. An error message is returned.

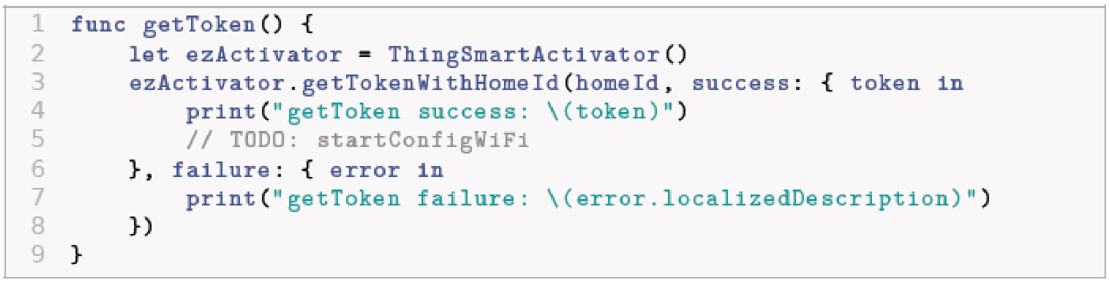
Example

Objective-C:



```
- (void)getToken {
        ThingSmartActivator *ezActivator = [[ThingSmartActivator all
oc] init];
    [ezActivator getTokenWithHomeId:homeId success:^(NSString *token
) {
        NSLog(@"getToken success: %@", token);
        // TODO: startConfigWiFi
    } failure:^(NSError *error) {
        NSLog(@"getToken failure: %@", error.localizedDescription);
    }];
}
```

Swift

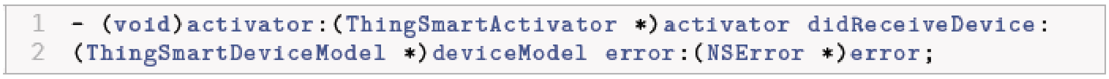


```
func getToken() {
    let ezActivator = ThingSmartActivator()
    ezActivator.getTokenWithHomeId(homeId, success: { token in
        print("getToken success: \(token)")
        // TODO: startConfigWiFi
    }, failure: { error in
        print("getToken failure: \(error.localizedDescription)")
    })
}
```

Callback of pairing delegate

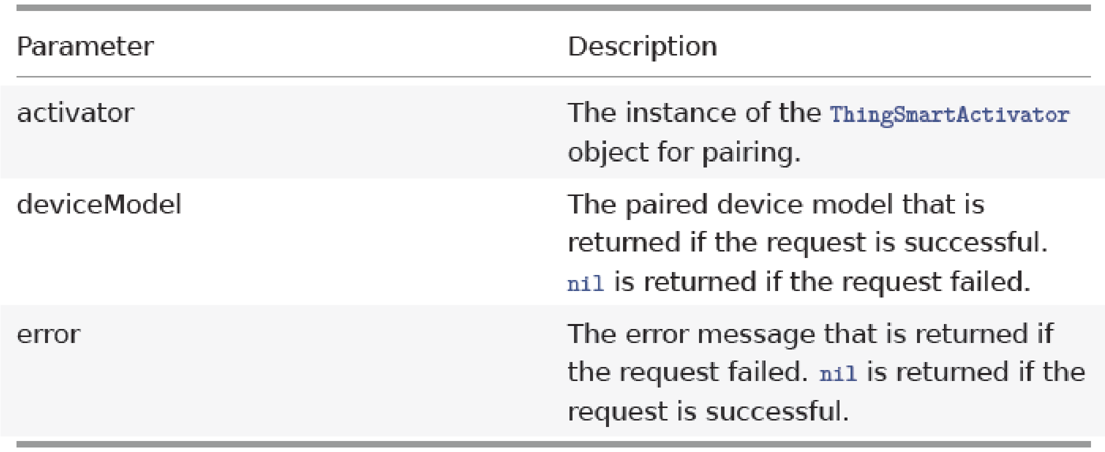
For pairing in the Wi-Fi EZ mode, you only need to implement this delegate call-back method and handle the devices and errors in this method. When pairing is successful or fails (in case of pairing timeout or error), callbacks are made on this delegate.

API description



```
- (void)activator:(ThingSmartActivator *)activator didReceiveDevice:
(ThingSmartDeviceModel *)deviceModel error:(NSError *)error;
```

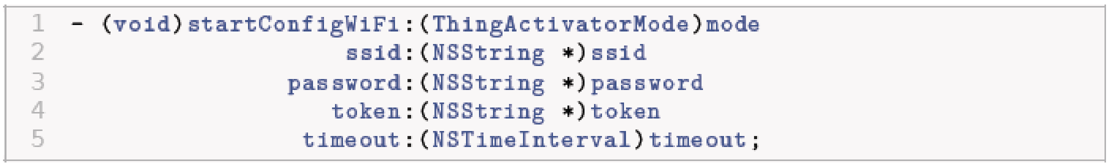
Parameters



Parameter	Description
activator	The instance of the `ThingSmartActivator` object for pairing.
deviceModel	The paired device model that is returned if the request is successful. `nil` is returned if the request failed.
error	The error message that is returned if the request failed. `nil` is returned if the request is successful.

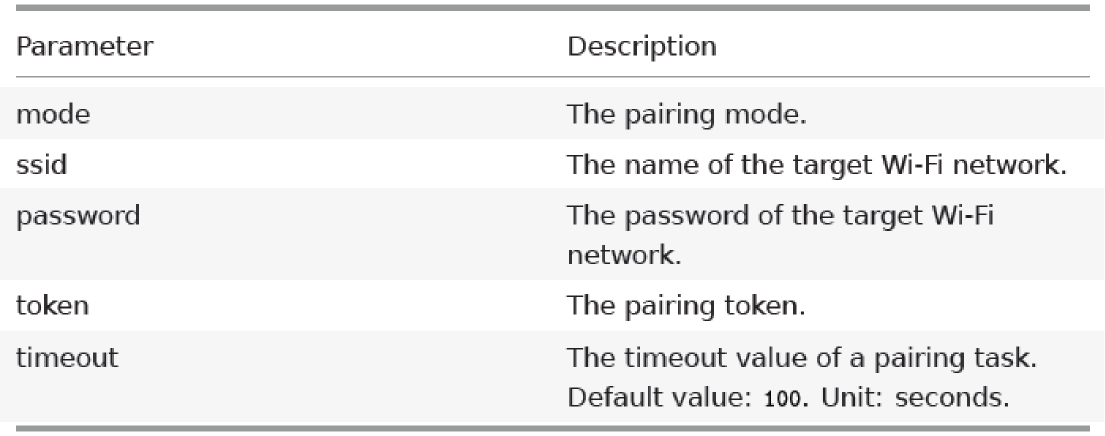
Start pairing

API description



```
- (void)startConfigWiFi:(ThingActivatorMode)mode
                   ssid:(NSString *)ssid
               password:(NSString *)password
                  token:(NSString *)token
                timeout:(NSTimeInterval)timeout;
```

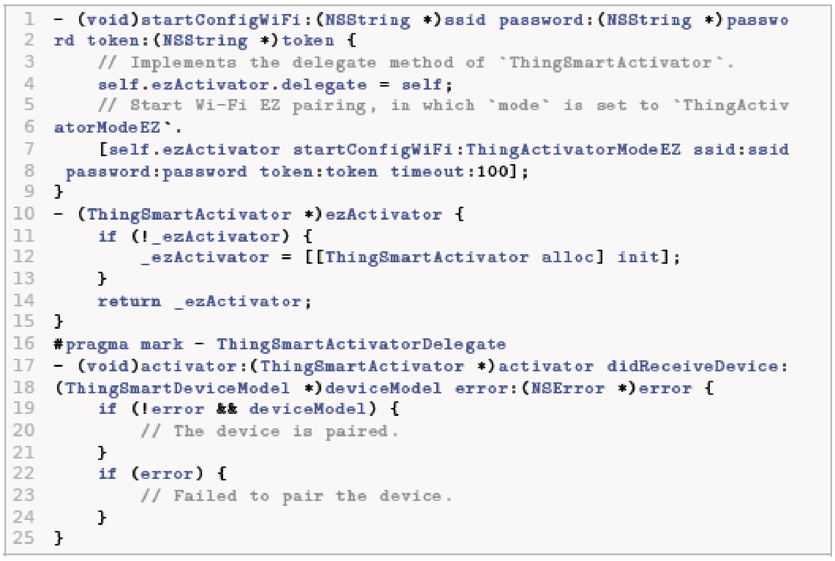
Parameters



Parameter	Description
mode	The pairing mode.
ssid	The name of the target Wi-Fi network.
password	The password of the target Wi-Fi network.
token	The pairing token.
timeout	The timeout value of a pairing task. Default value: `100`. Unit: seconds.

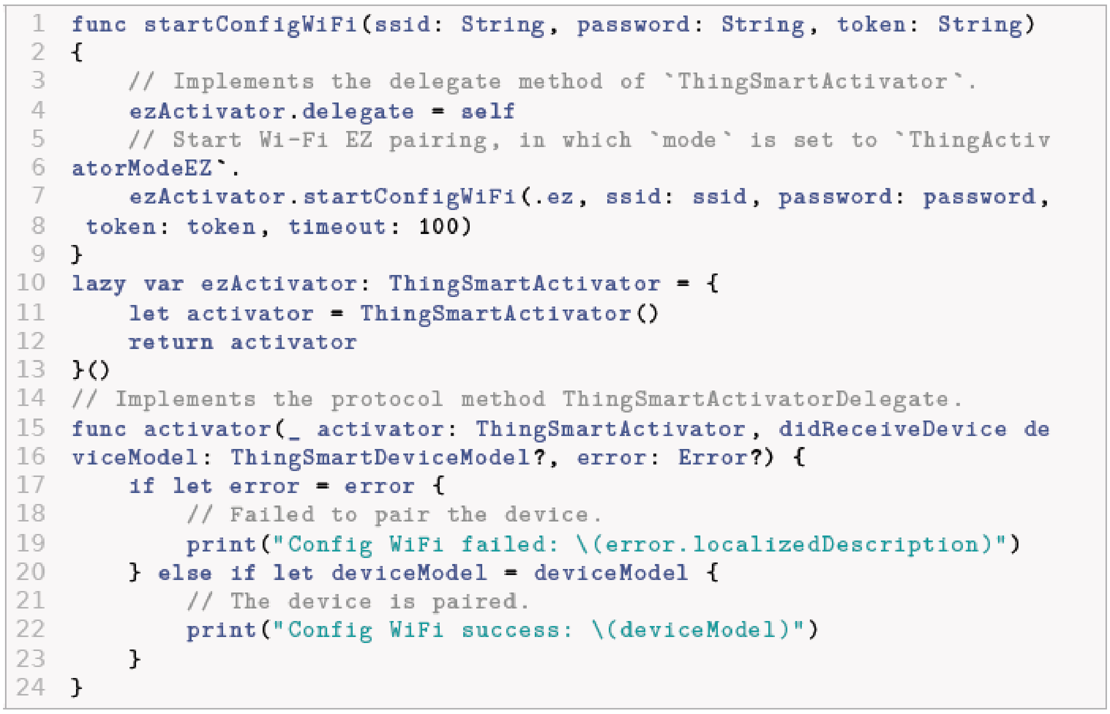
Example

Objective-C:



```
- (void)startConfigWiFi:(NSString *)ssid password:(NSString *)passwo
rd token:(NSString *)token {
    // Implements the delegate method of `ThingSmartActivator`.
    self.ezActivator.delegate = self;
    // Start Wi-Fi EZ pairing, in which `mode` is set to `ThingActiv
atorModeEZ`.
    [self.ezActivator startConfigWiFi:ThingActivatorModeEZ ssid:ssid
 password:password token:token timeout:100];
}
- (ThingSmartActivator *)ezActivator {
    if (!_ezActivator) {
        _ezActivator = [[ThingSmartActivator alloc] init];
    }
    return _ezActivator;
}
#pragma mark - ThingSmartActivatorDelegate
- (void)activator:(ThingSmartActivator *)activator didReceiveDevice:
(ThingSmartDeviceModel *)deviceModel error:(NSError *)error {
    if (!error && deviceModel) {
        // The device is paired.
    }
    if (error) {
        // Failed to pair the device.
    }
}
```

Swift:



```
func startConfigWiFi(ssid: String, password: String, token: String)
{
    // Implements the delegate method of `ThingSmartActivator`.
    ezActivator.delegate = self
    // Start Wi-Fi EZ pairing, in which `mode` is set to `ThingActiv
atorModeEZ`.
    ezActivator.startConfigWiFi(.ez, ssid: ssid, password: password,
 token: token, timeout: 100)
}
lazy var ezActivator: ThingSmartActivator = {
    let activator = ThingSmartActivator()
    return activator
}()
// Implements the protocol method ThingSmartActivatorDelegate.
func activator(_ activator: ThingSmartActivator, didReceiveDevice de
viceModel: ThingSmartDeviceModel?, error: Error?) {
    if let error = error {
        // Failed to pair the device.
        print("Config WiFi failed: \(error.localizedDescription)")
    } else if let deviceModel = deviceModel {
        // The device is paired.
        print("Config WiFi success: \(deviceModel)")
    }
}
```

Stop pairing

After the pairing process is started, the app continuously broadcasts the pairing data until a device is paired or the process times out. To allow users to cancel or com-plete pairing during the process, you must call the API method [ThingSmartActivator stopConfigWiFi] to stop pairing.

API description

```
- (void)stopConfigWiFi;
```

Example

Objective-C:

```
- (void)stopConfigWifi {
    self.ezActivator.delegate = nil;
    [self.ezActivator stopConfigWiFi];
}
```

Swift

```
func stopConfigWifi() {
    ezActivator.delegate = nil
    ezActivator.stopConfigWiFi()
}
```

FAQ

- **What is Wi-Fi EZ Mode?**

  Wi-Fi EZ Mode is a method to pair devices using Wi-Fi Easy Connect or SmartConfig. It involves using a router to broadcast and pair a mobile phone with a device.

- **What is the validity period of a pairing token?**

  The pairing token is valid for 10 minutes and expires immediately after the device is paired.

- **Can I use Wi-Fi EZ Mode on iOS 14.5 and later**

  It is recommended to use AP mode instead of Wi-Fi EZ mode for iOS 14.5 and later due to better compatibility and success rate.

Documents / Resources

 tuya Wi-Fi EZ Mode Version: 20240517 Online Version	tuya 20240517 Smart App SDK [pdf] User Guide 20240517 Smart App SDK, 20240517, Smart App SDK, App SDK

References

- User Manual

20240517, 20240517 Smart App SDK, App SDK, Smart App SDK,

Tuya tuya

—Previous Post

tuya Smart App SDK User Guide

Next Post—

tuya 905ML Keyless Deadbolt Lock Instruction Manual

Leave a comment

Your email address will not be published. Required fields are marked *

Comment *

Name

Email

Website

□ Save my name, email, and website in this browser for the next time I comment.

Post Comment

Search:

e.g. whirlpool wrf535swhz

Search

This website is an independent publication and is neither affiliated with nor endorsed by any of the trademark owners. The "Bluetooth®" word mark and logos are registered trademarks owned by Bluetooth SIG, Inc. The "Wi-Fi®" word mark and logos are registered trademarks owned by the Wi-Fi Alliance. Any use of these marks on this website does not imply any affiliation with or endorsement.