

A Combination of JS Object & PLC Web Browser

Sending push messages via LINE Notify

Demo Project

Contents

- 1. Overview & Operation 1
- 2. Setting up the Screen 7
- 3. Addresses 14

1. Overview & Operation

Overview

This demo project demonstrates how to use JS Object and PLC Web Browser to obtain the Access Token, which represents the authorization from a LINE user to send push messages via LINE Notify.

LINE Notify is a push messaging service that utilizes the OAuth 2.0 authentication process. Before we proceed, let's take a look at the following roles:

1. Service Provider (Message Sender)

A service provider must have a "callback" URL for OAuth 2.0 to invoke after the authentication process to obtain the Access Token. In this demo project, PLC Web Browser and JS Object are used to obtain this token.

2. Authorization Grantor (LINE User)

A LINE user can authorize LINE Notify to send messages. To do so, the LINE user must have a LINE account and log in to grant authorization.

3. Message Receiver

When authorizing LINE Notify, a LINE user can specify whether the message should be sent to a 1-on-1 chat or a LINE group.

Operation

Step 1. Visit LINE Notify website (<https://notify-bot.line.me/en/>) and log in to your account. Add a service provider and specify a callback URL. In this demo project the callback URL <https://127.0.0.1:8000> is used. Once you have provided all necessary information, you will receive a Client ID and Client Secret.

To service providers

If you connect your service with LINE Notify, LINE users can easily receive notifications from it.


[Add service >](#)

[LINE Notify API Document >](#)

Add service

Fill in the fields below and confirm your email address to connect your service with LINE Notify.

Service logo



Service name

Service description

Service URL

Company/Enterprise

Country/region

Representative

Email address

Callback URL

Note: You can register up to five callback URLs by placing each URL on a new line.

Required

[Terms and Conditions of Use](#) [Privacy Policy](#)

Agree to the above to continue and add your service.

[Agree and continue](#)

Weintek

Client ID

Client Secret [Display](#) [Reset](#)

Users 1

- Step 2.** In window 10 of the project, the Client ID and Client Secret are needed in two places:
- Layer 2, CB_1 Combo Button's Execute JS action in Action Group 1.
 - Layer 3, CB_2 Combo Button's Execute JS action in Action Group 1.

Execute JS		
Config		
Name	Type	Value
config	Object (6)	
browser_current_url	Address	Browser Current URL : ...
client_id	String	Your_Client_ID
client_secret	String	Your_Client_Secret
redirect_uri	String	https://127.0.0.1:8000
token	Address	New-Token : PLW-23...
get_token_success	Address	get_token_success_bit...

Execute JS		
Config		
Name	Type	Value
config	Object (3)	
client_id	String	Your Client ID
destination	Address	Browser Destination URL : PL...
redirect_uri	String	https://127.0.0.1:8000

Step 3. Once finished, download the project to HMI.

Step 4. Make sure that the HMI is connected to the internet and press the [Connect With The Service] button to initiate the authorization process.

JS Object x Web Browser

LINE users can easily receive notifications from LINE Notify's official account by connecting to this service.
Please press the "Connect With The Service" button to begin the registration process.

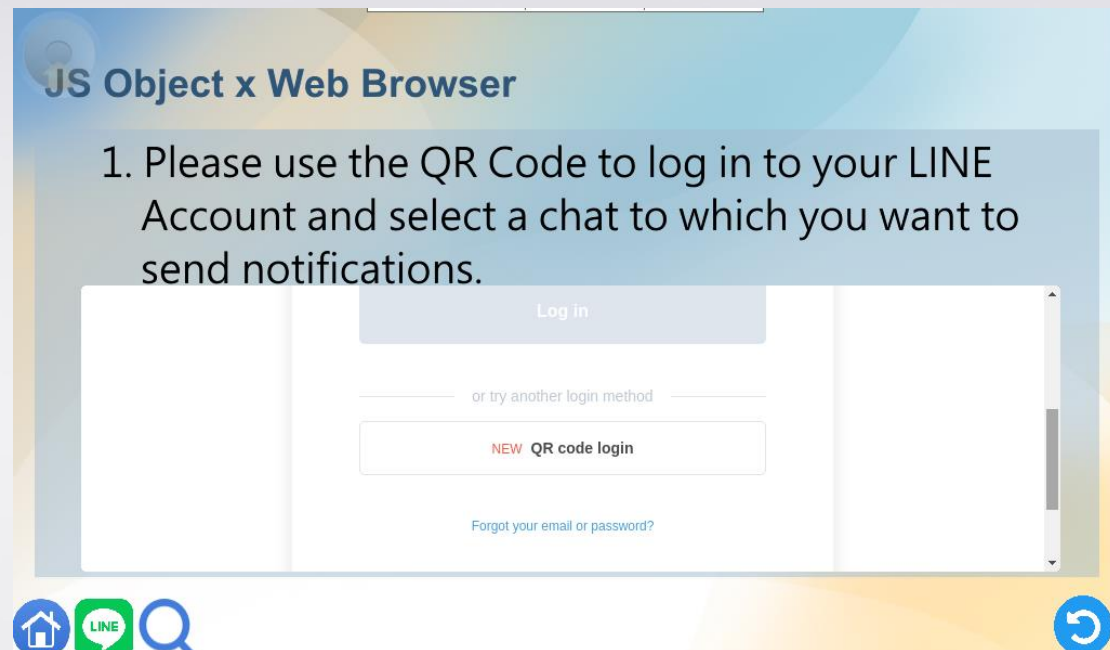
[Connect With The Service](#)

Once registered, the service owner can send notifications to LINE users who are registered in groups or 1-on-1 chats.

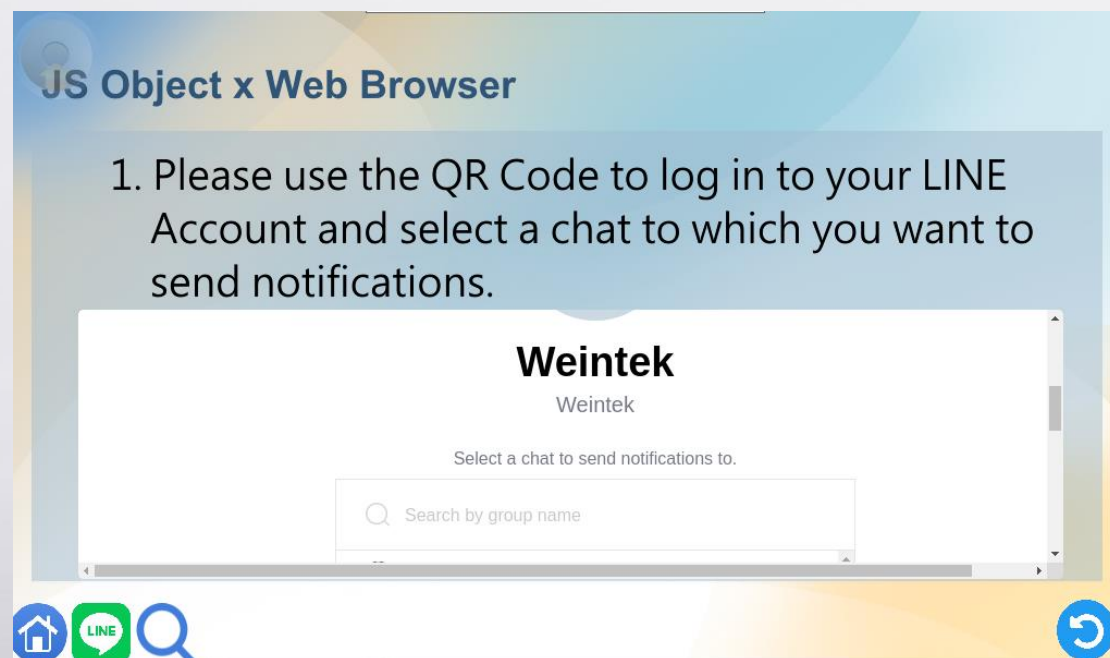
Please press the LINE icon located in the bottom left corner to send the messages.



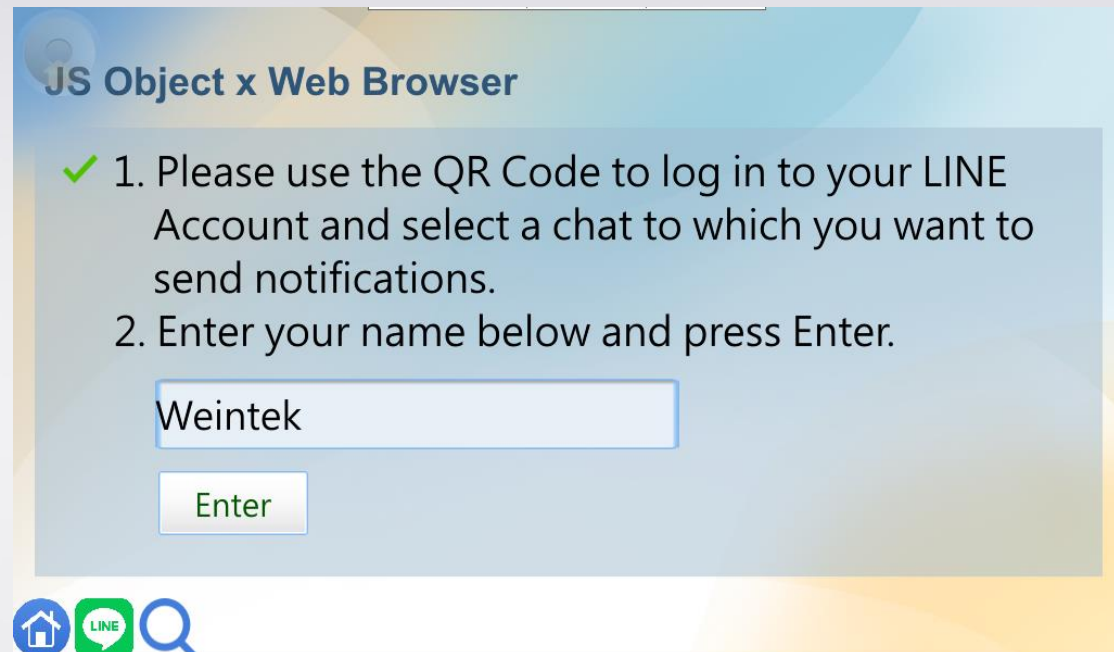
Step 5. Log in to your LINE account.



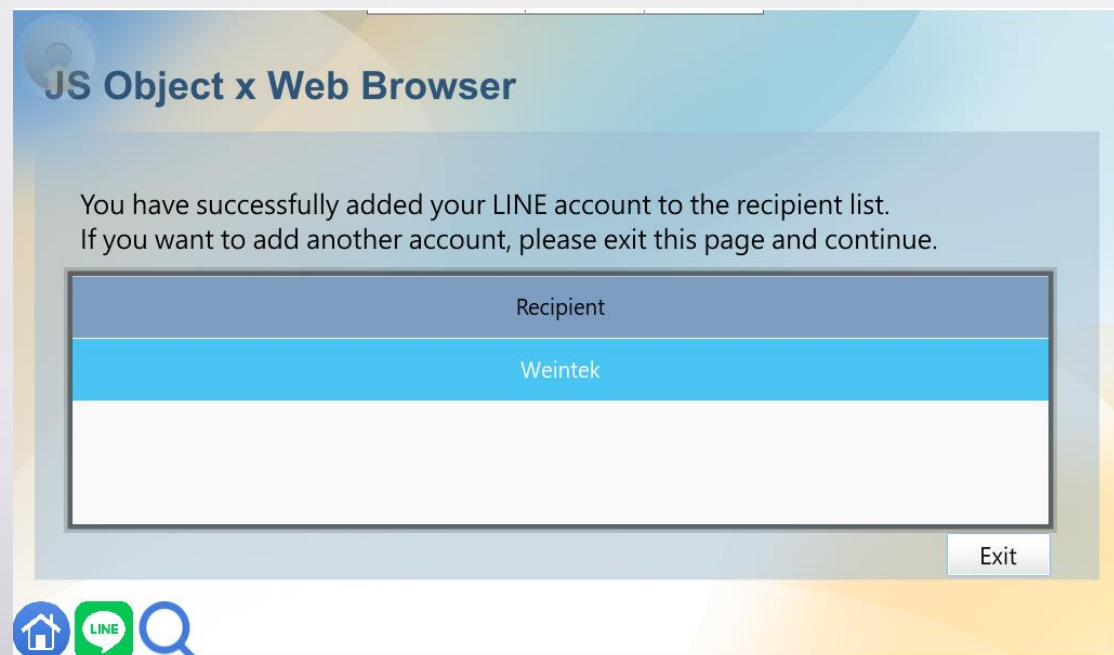
Step 6. Specify the chat window where LINE Notify should send the messages.



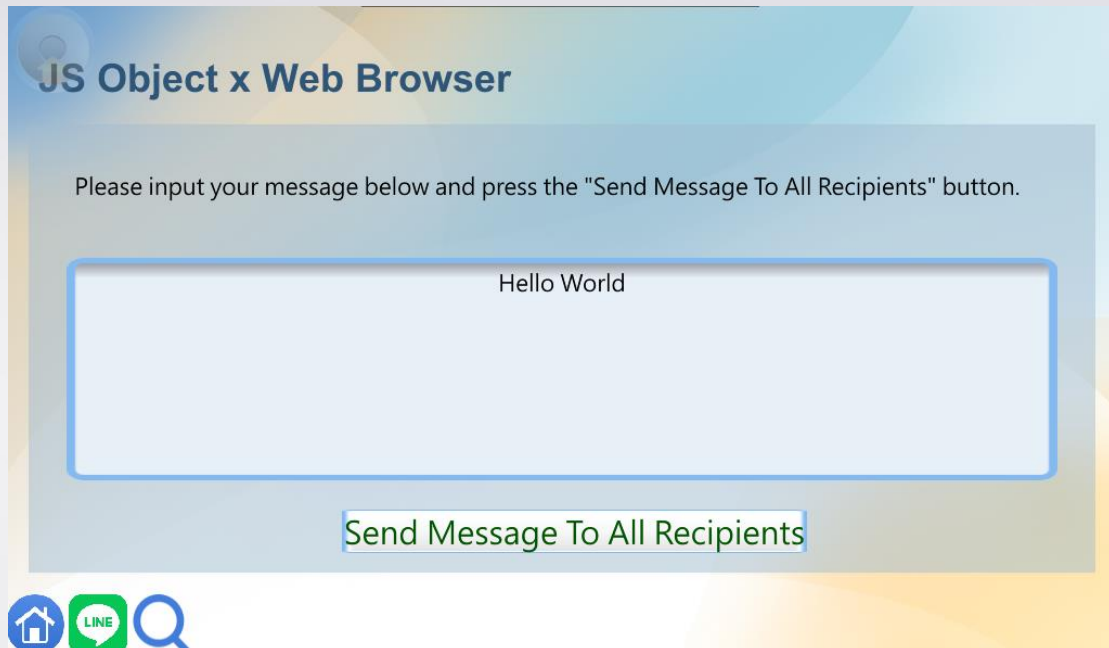
Step 7. Enter the service name used in the project and then press Enter.



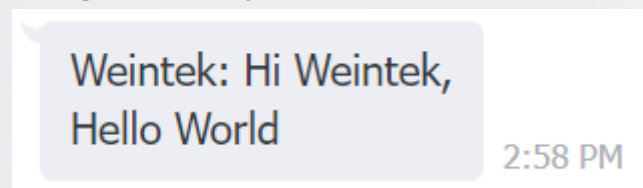
Step 8. Once the authorization is successful, you will be able to see the LINE users added in the Recipient list.



Step 9. Press the LINE icon in the lower-left corner, enter your message in the provided field, and then press the [Send Message To All Recipients] button to send your message to all recipients on the list.

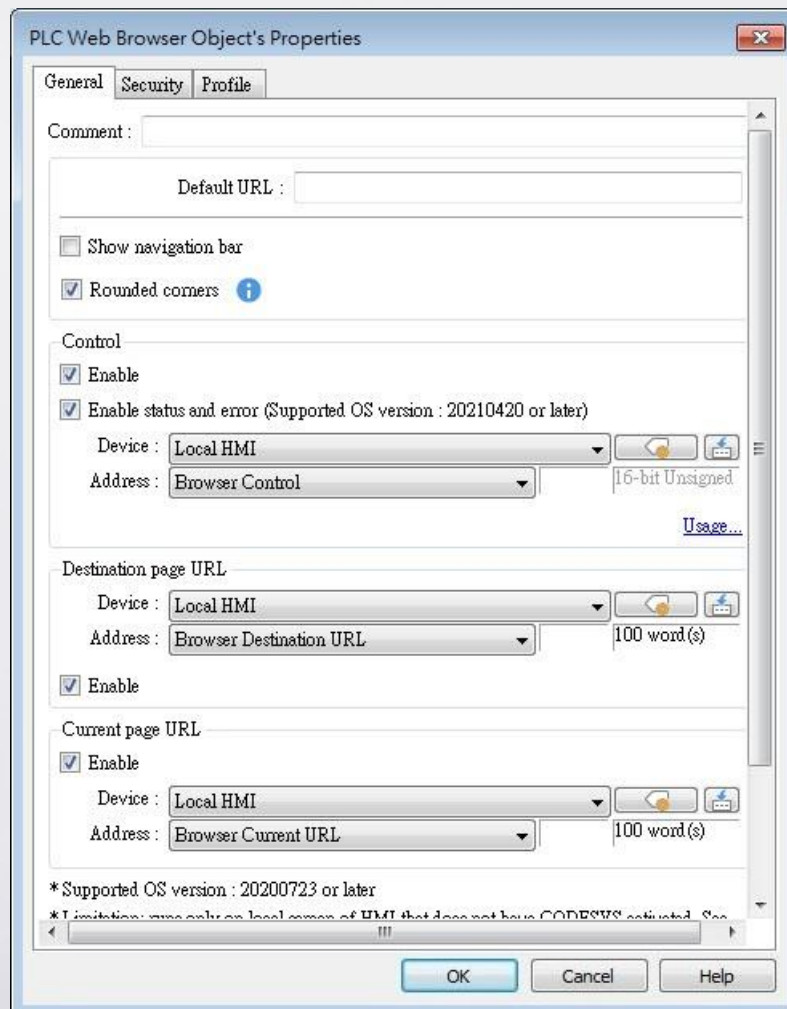


Step 10. All the LINE users on the recipient list in the project will receive the push message that was just sent.



2. Setting up the Screen

Step 1. Create a PLC Web Browser object, set the control address, and complete Security settings.



Step 2. Create a Combo Button and add Execute JS and Set Word actions into the Action Group. The current page URL generated by the JS object will be used as the input for the PLC Web Browser created in step 1. (The following only shows a part of the commands; for more details, please refer to the settings in Window 10, Layer 3, CB_2 object.)

Combo Button Object's Properties

General Security Shape Font Profile

Comment :

Lamp

Mode :

Down actions

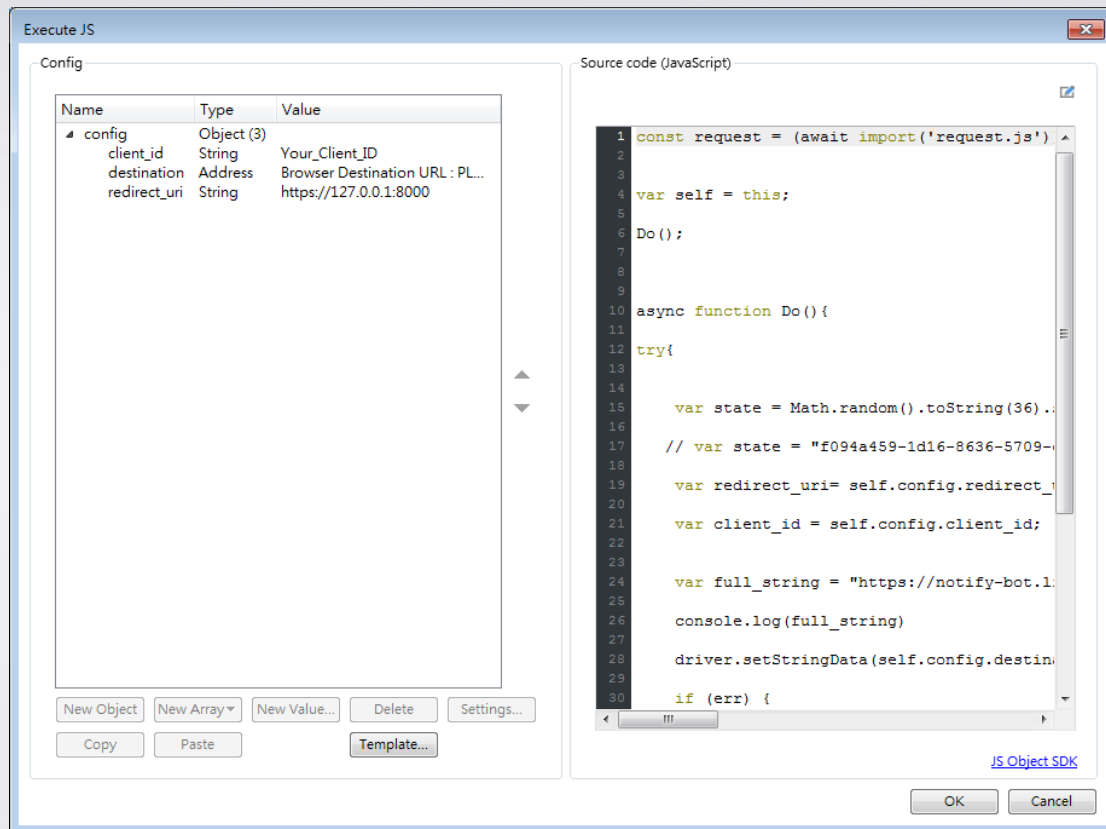
Action Group 1	Action Group 2
Execute JS	
Set Word (Write constant value : 6...	Set Word (Write constant
<input type="text"/>	<input type="text"/>
<input data-bbox="662 806 694 840" type="button" value="+"/>	<input data-bbox="1061 795 1093 828" type="button" value="+"/>

Up actions

Action Group 1
<input data-bbox="534 1030 566 1064" type="button" value="+"/>

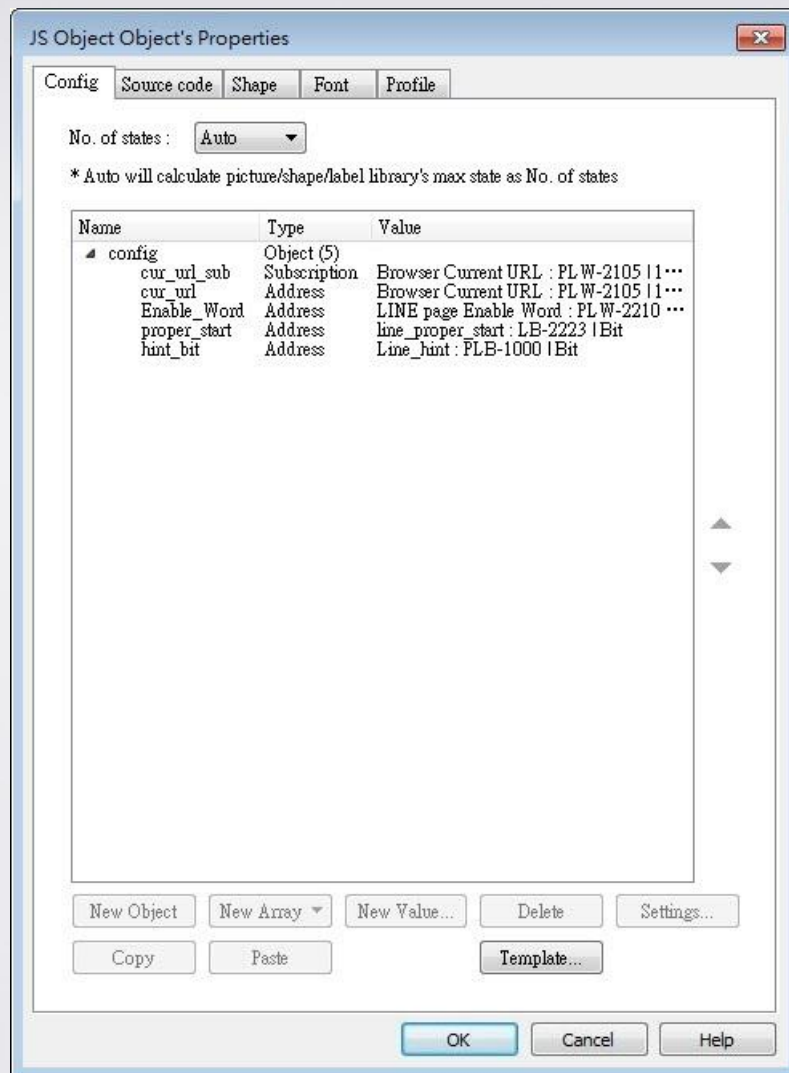
* Up actions may run before down actions end.

OK Cancel Help



Here is only a part of the commands, please refer to the project for more details.

Step 3. Create a JS Object that monitors the current page URL of the PLC Web Browser. When a LINE user logs in and authorizes this project, and the authorization server redirects the browser back to the redirect URL, this JS object will change the security settings of the relevant object to display the corresponding step instructions. (The following only shows a part of the commands; for more details, please refer to the settings in Window 10, Layer 1, CO_0 object.)



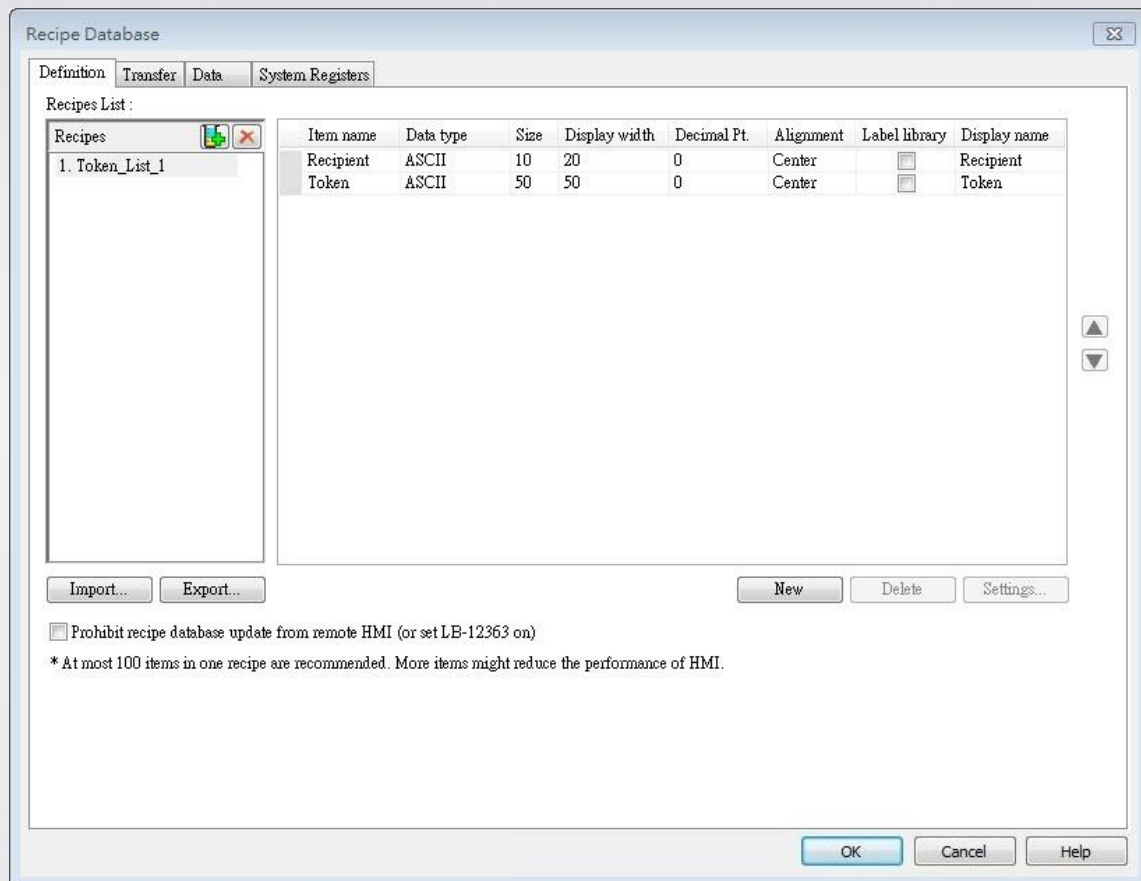
```

1 var self = this;
2
3
4 setTimeout(() => {
5
6 self.config.cur_url_sub.onResponse( async(err, data) => {
7   if (err) {
8     console.log('Error:', err.message);
9   } else {
10
11     var url = await binary2str();
12     console.log('url:', url);
13
14     if (url.indexOf("127.0.0.1:8000/?code=") != -1 ){
15
16
17
18     var state_2 = 10;
19     driver.setData(self.config.Enable_Word, state_2, (err) => {
20       if (err) {
21         console.log('Error:', err.message);
22       } else {
23         console.log('state2 write success');
24       }
25     });
26   });
27
28
29 }
30
31

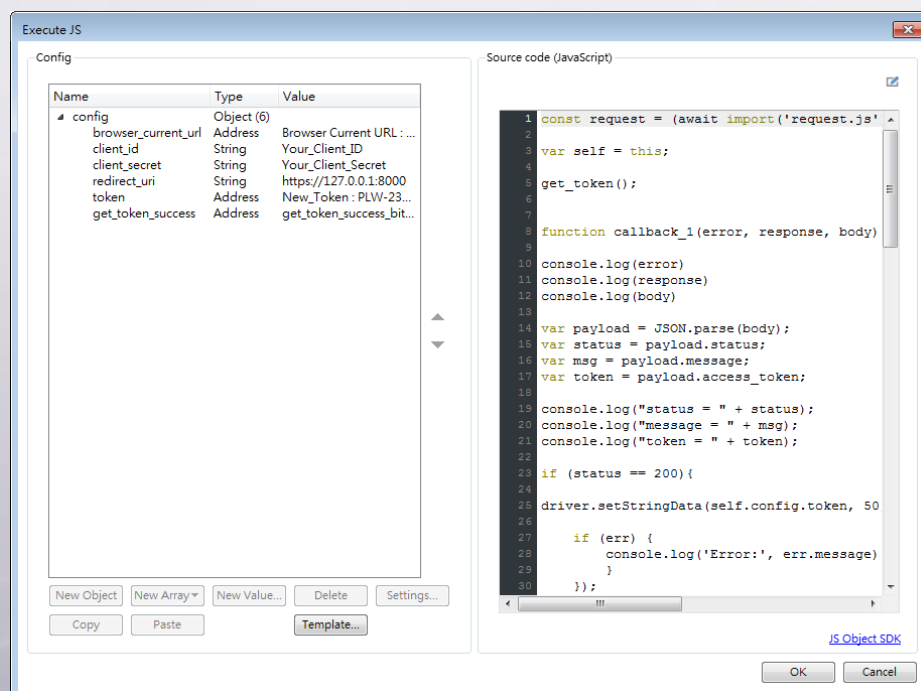
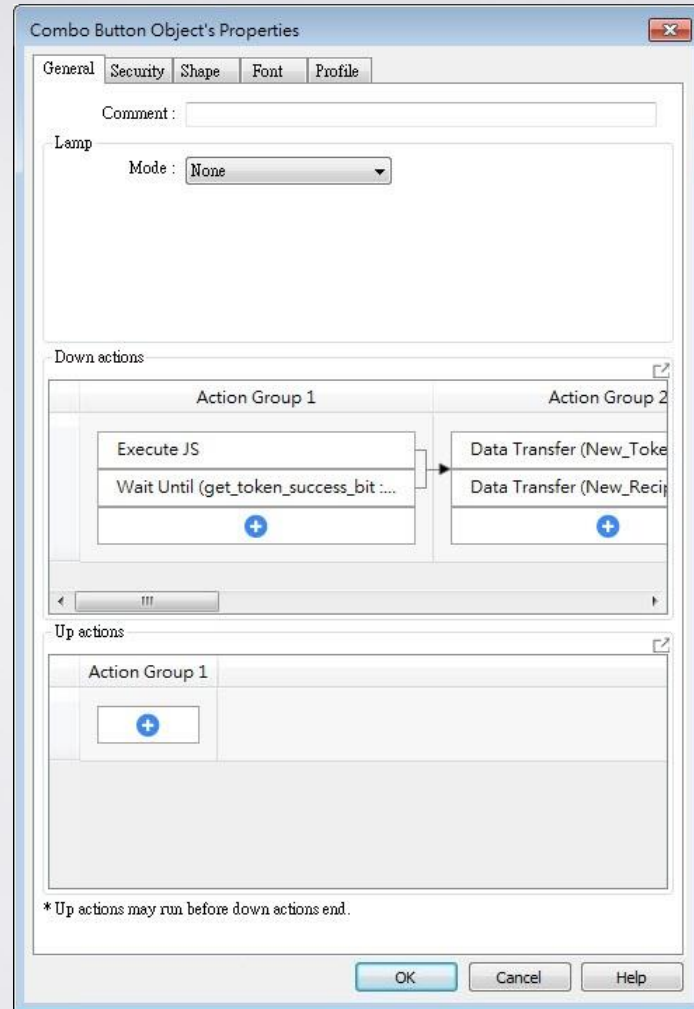
```

Here is only a part of the commands, please refer to the project for more details.

Step 4. Create a Recipe Database named Token_List_1 and add Recipient and Token into the list as shown below.

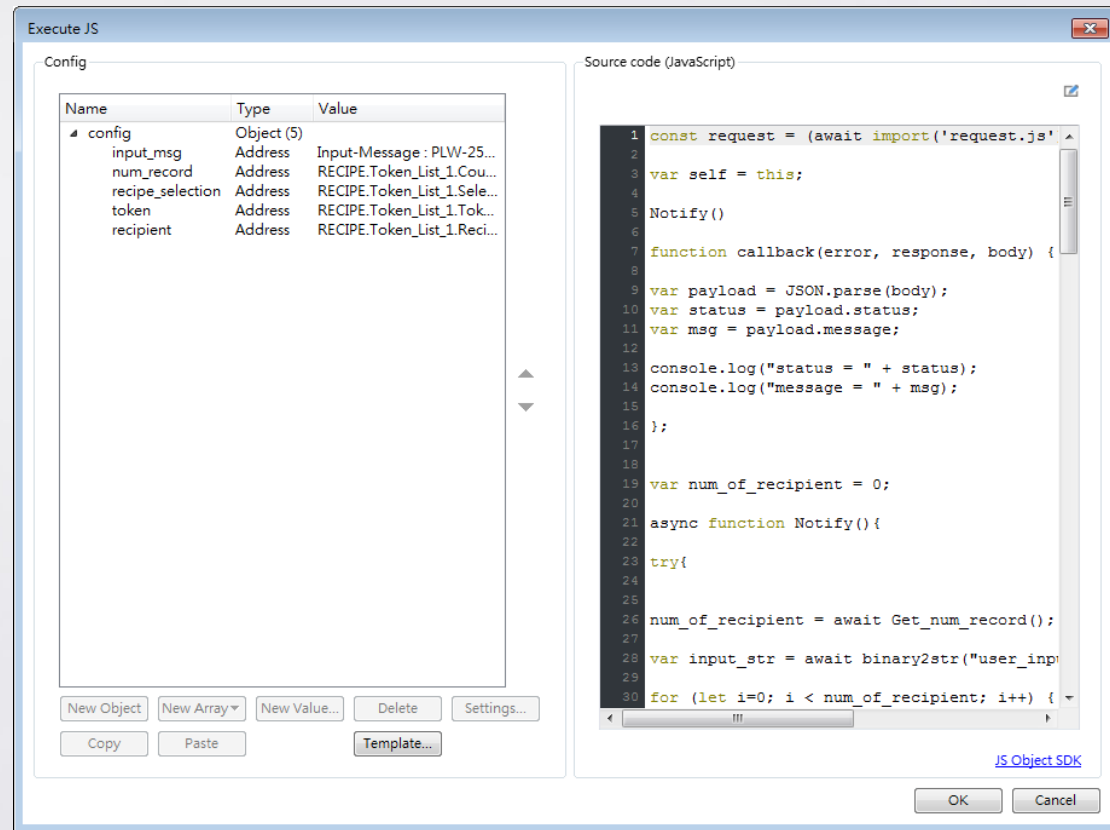


Step 5. Create a Combo Button. After a LINE user has authorized the project, and the authentication server has returned the Authentication Grant information to the Redirect URL in step 3, it is necessary to exchange this Authentication Grant for an Access Token from the authentication server, and then store the token in Recipe Database. (The following only shows a part of the commands; for more details, please refer to the settings in Window 10, Layer 2, CB_1 object.)



Here is only a part of the commands, please refer to the project for more details.

Step 6. Create a Combo Button that executes JS and uses the Token saved in Recipe Database to send push messages to LINE users. (The following only shows a part of the commands; for more details, please refer to the settings in Window 16, CB_0 object.)



3. Addresses

The addresses of key objects used in this demonstration are listed below, please adjust as necessary.

Object	Address	Object ID	Description
Window 10			
PLC Web Browser	PLW-2000 PLW-2005 PLW-2105	WB_0	Used for browsing web pages.
JS Object	PLW-2105 PLW-2210 LB-2223 PLB1000	CO_0	Monitors the current page URL.
Combo Button	PLW-2105 PLW-2310 PLB-2221	CB_1	Gets Access Token.
Combo Button	PLW-2005	CB_2	Generates URL.
Window 16			
ASCII	PLW-2500	AE_0	Used for entering push messages.
Combo Button	PLW-2500	CB_0	Sends push messages to LINE users.