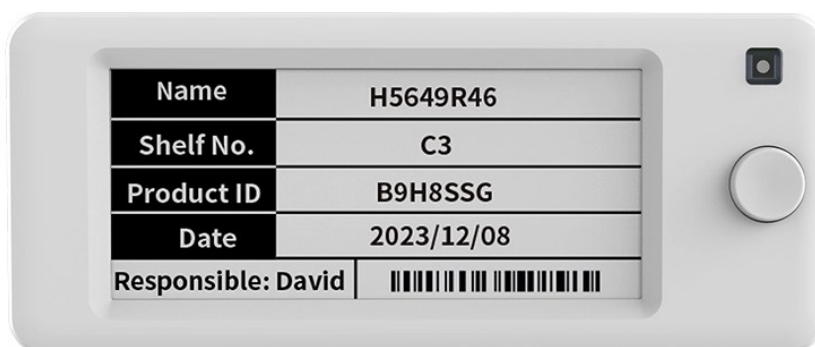


SE-290A button event handling



White paper

The information in this document is subject to change without notice.

Document History

Model Number:	SE-290A button event handling
Edition:	1.0.0.2
Date:	November 4 th , 2025

© 2025 Opticon. All rights reserved.

This manual may not, in whole or in part, be copied, photocopied, reproduced, translated, or converted to any electronic or machine-readable form without prior written consent from Opticon.

Limited Warranty and Disclaimers

Please read this manual carefully before installing or using the product.

Serial Number

A serial number appears on all Opticon products. This official registration number is directly related to the device purchased. Do not remove the serial number from your Opticon device. Removing the serial number voids the warranty.

Warranty

Unless otherwise agreed in a written contract, all Opticon products are warranted against defects in materials and workmanship for two years after purchase excluding batteries. Opticon will repair or, at its option, replace products that are defective in materials or workmanship with proper use during the warranty period. Opticon is not liable for damages caused by modifications made by a customer. In such cases, standard repair charges will apply. If a product is returned under warranty and no defect is found, standard repair charges will apply. Opticon assumes no liability for any direct, indirect, consequential or incidental damages arising out of use or inability to use both the hardware and software, even if Opticon has been informed about the possibility of such damages.

Packaging

The packing materials are recyclable. Damage caused by improper packaging during shipment is not covered by the warranty.

Trademarks

Trademarks used are the property of their respective owners.

Opticon Inc. and Opticon Sensors Europe B.V. are wholly owned subsidiaries of OPTOELECTRONICS Co., Ltd., 12-17, Tsukagoshi 4-chome, Warabi-shi, Saitama, Japan 335-0002. TEL +81-(0) 48-446-1183; FAX +81-(0) 48-446-1184

Support

USA		Europe	
Phone:	800-636-0090	Phone:	+31235692728
Email:	support@opticonusa.com	Email:	support@opticon.com
Web:	www.opticonusa.com	Web:	www.opticon.com

Contents

1	Introduction.....	4
2	Requirement.....	4
3	SE-290A/SE-290ARY Specifications	4
4	Button Event Handling using Templates.....	4
4.1	Adding a button object to a template.....	4
4.1.1	Add the Button Object	5
4.1.2	Button object settings.....	5
5	Button Event handling using external CMS	7
5.1	Added multiple pages (images)	7
5.2	Adding LED behavior to an image.....	8
5.3	Added button event actions to an image.....	9
6	Event handler	10
6.1	Default behavior	11
6.2	Enabling of the event handler	11
6.3	Adding a button event handler	11
6.4	Input response API-endpoint.....	12
6.4.1	Retrieving the page and image IDs from button events.....	13
6.4.2	Input response API-endpoint with custom action and LED response.....	14
6.5	Performance	15
6.5.1	Direct API-response	15
6.5.2	Keep-alive	16

1 Introduction

This manual documents the technical implementation and configuration of handling button events for interactive use of the **Opticon SE-290A** and **SE-290ARY** Electronic Shelf Labels (ESLs). It covers button event processing, automated event actions, and event forwarding.

2 Requirement

The minimal software requirements for using the latest button features of the SE-290A are:

*EBS-50 or ESL Web server application: **IBGV0121a (3.0.9.xxxx) or higher***

*SE-290A / SE-290ARY firmware: **IxVA0200 or higher***

*EBS-40: **IBUV0072 or higher***

3 SE-290A/SE-290ARY Specifications

The SE-290A series is designed for interactive applications like warehouse picking and user feedback, featuring a dedicated button and an RGB LED.

Feature	Specification
Display Size	2.9 inches
Resolution	296 x 128 pixels (W x H)
Colors (SE-290A)	Black, White
Colors (SE-290ARY)	Black, White, Red, Yellow
Operating Indicator	1x RGB LED , with adjustable brightness and duty level
User Input	1x Button
Protocol	IEEE 802.15.4 based, 2.4 GHz
Power	2x CR2450 lithium primary batteries

4 Button Event Handling using Templates

The button can behave differently depending on how it's been configured using the assigned image templates that can be created using the Template Designer.

See [chapter 5 Button Event handling using external CMS](#) if you wish to handle button events without using templates and pro

4.1 Adding a button object to a template

The SE-290A supports **button** events that can trigger actions on the ESL or in the back-end — for example, to change pages, to activate an LED or to trigger an image update. Configuration is done through the **Button Object** in the template designer.

4.1.1 Add the Button Object

To add a button object to a template:

1. Open your template in the **Template Designer**.
2. From the vertical **toolbar**, select the **Button Object** (the icon with a small button, as indicated in the image below).



3. Draw a small rectangle on your template to **place the button** where desired

4.1.2 Button object settings

After adding the button object, the button settings are displayed at the top.

Button Settings

Event

Short Press

No action

Long Press

Disabled

Very Long Press

Disabled

Double Click

Disabled

Timeout

Disabled

☒ Execute on success

☐ Execute on failure

Action

No action

Disabled

Disabled

Disabled

Disabled

Options

Timeout (sec)

120

Long press (ms)

2000

Indicate OK

None

Indicate Fail

None

Condition:

4.1.2.1 Event

The button can behave differently depending on how it is pressed:

Event Type	Description
Short Press	Short button press
Long Press	Press and hold beyond configured long press time
Very Long Press	Press and hold beyond the very long threshold (currently not supported)
Double Click	Two quick taps
Timeout	Event that occurs {Timeout} seconds after a button press (for example to revert to the default page after x seconds)

4.1.2.2 Action

Under **Action**, choose what happens when the button event occurs.

Event	Description
No action	The event is sent to the backend, but no action is triggered
Show next page	Displays the next page in the template sequence.
Show previous page	Displays the previous page in the template sequence.
Show page X	Displays a specific page in the template sequence.
Refresh image	Refreshes currently displayed image
Invert image	Inverts the currently displayed image
Turn ON LED	Turn the LED on (as specified in the current template page).
Turn OFF LED	Turn the LED off
Toggle LED	Toggle the LED state (requires firmware IxVA0200 or higher)
Disabled	The event is not sent to the backend *

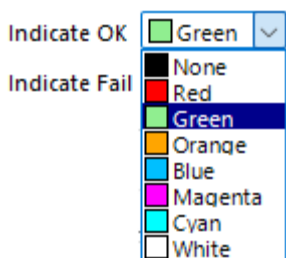
* Note: Disabling the **double press** and **long press** event will cause the **short press** event to be sent upon **pressing** of the button instead of upon **releasing** of the button

4.1.2.3 Conditions and options

Using the ‘Execute on success’ and ‘Execute on failure’ checkboxes to configure whether the specified action should always be executed upon successful transmission of the button-event to the ESL web server and/or if the transmission fails.

- ☒ Execute on success
☐ Execute on failure

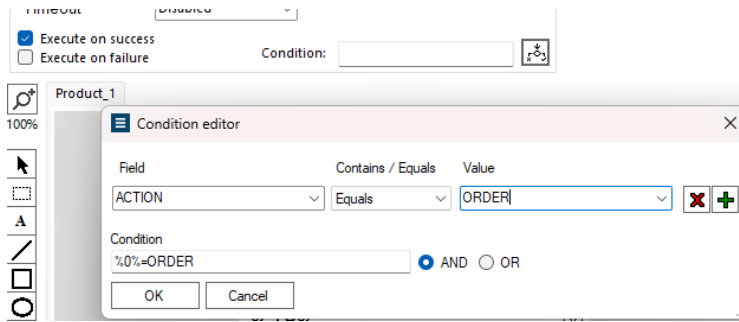
Besides the configured action, it is also possible to configure the default LED response, upon success or failure to send the button event to the ESL web server, by selecting a LED color on the drop-down menus of ‘Indicate OK’ and ‘Indicate Fail’



4.1.2.3.1 Advanced conditional behavior

It is possible to add multiple button objects to a single template and add different conditions to specify which button action should be executed.

For example: when ‘Picking list = 1’ -> show blue LED. ‘Picking list = 2’ -> show green LED



Example of a condition

5 Button Event handling using external CMS

The alternative to specifying button actions in a template is by providing images and button actions to the ESL Web Server using the external Content Management System (CMS) configuration.

Using an External Content Management System (CMS) means a 3rd party system will supply images and XML-files to the ESL Web Server Application by placing them into the Input-folder or uploading them to the API. See chapters 17.6 and 16.9 of the ESL Web Server manual for more information.

To support loading multiple images, add LED settings and configure button event actions, the XML-files have been extended with the following options.

5.1 Added multiple pages (images)

Note: Requires firmware IxVA0200 or higher

The standard XML-format to load a single image when using CMS is as follows:

```
<EslImageInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <ImageFile>B4AE0076.png</ImageFile>
  <ID>12345</ID>
  <Note>Paracetamol 500mg (50pcs)</Note>
  <Label>B4AE0076</Label>
</EslImageInfo>
```

To be able to load additional pages (images), add a list of Images using the following format. Page '0' is the image page that is displayed by default, so the next page is 1, etc.

```
<EslImageInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <ImageFile>B4AE0076.01.png</ImageFile>
  <ID>12345</ID>
  <Note>Paracetamol 500mg (50pcs)</Note>
  <Label>B4AE0076</Label>
  <Images>
    <Image>
      <Page>1</Page>
      <File>B4AE0076.02.png</File>
    </Image>
    <Image>
      <Page>2</Page>
      <File>B4AE0076.03.png</File>
    </Image>
  </Images>
</EslImageInfo>
```

```

</Image>
</Images>
</EslImageInfo>

```

5.2 Adding LED behavior to an image

The LED behavior of an ESL can be defined in XML-files by adding a Led-object to the default page as well as to the other pages using the following format:

```

<EslImageInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <ImageFile>B4AE0076.01.png</ImageFile>
  <ID>B4AE0076</ID>
  <Note>Paracetamol 500mg (50pcs)</Note>
  <Label>B4AE0076</Label>
  <Images>
    <Image>
      <Page>1</Page>
      <File>B4AE0076.02.png</File>
      <Led>
        <Mode>FLASH</Mode>
        <Color>RED</Color>
        <Period>1000</Period>
        <DutyCycle>2</DutyCycle>
        <FlashCount>100</FlashCount>
        <Brightness>0</Brightness>
      </Led>
    </Image>
  </Images>
  <Led>
    <Mode>FLASH</Mode>
    <Color>RED</Color>
    <Period>1000</Period>
    <DutyCycle>2</DutyCycle>
    <FlashCount>100</FlashCount>
    <Brightness>0</Brightness>
  </Led>
</EslImageInfo>

```

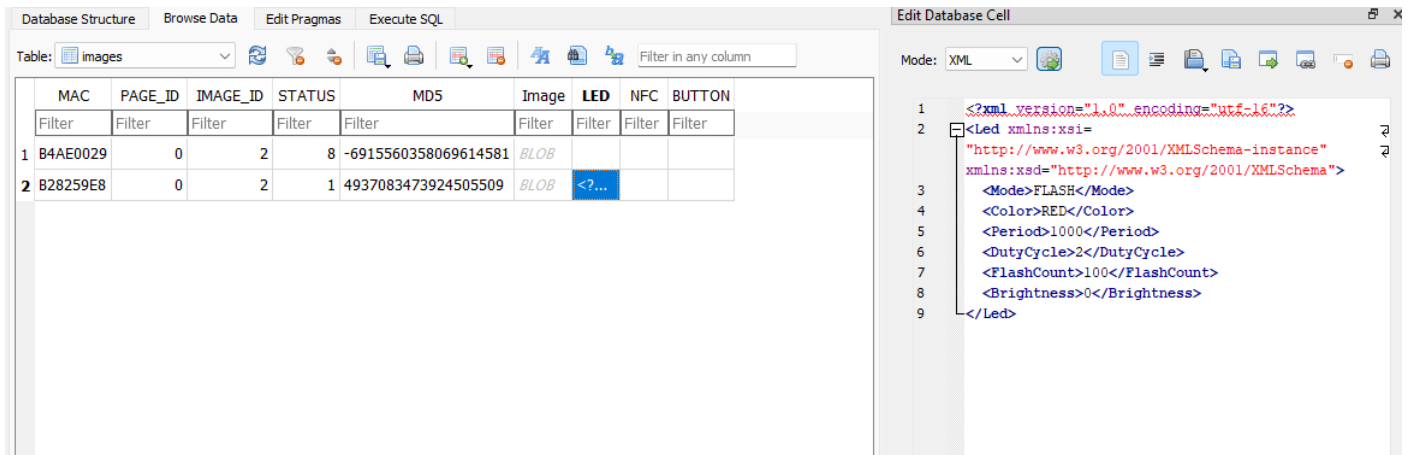
Tip:

The easiest way to generate and test the XML for LED-settings is to create a template with an LED object and then link an ESL to the given template. The resulting image (incl the LED XML-code) will be stored in the “Images” table of the (logging) database. By default, it is a SQLITE database located in the Output-folder of your installation.

LED settings

Mode:
Flash
Color:
Red
Brightness (%):
50
Period (ms):
1000
Count:
100
Led ON (%):
2

Template with LED object



Resulting image with LED object in the Images table

5.3 Added button event actions to an image

Button event actions can be added to the XML-files by adding a button object to the default page as well as to the other pages using the following format:

```
<EslImageInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <ImageFile>B4AE0076.01.png</ImageFile>
  <ID>B4AE0076</ID>
  <Note>Paracetamol 500mg (50pcs)</Note>
  <Label>B4AE0076</Label>
  <Images>
    <Image>
      <Page>1</Page>
      <File>B4AE0076.02.png</File>
      <Button>
        <ShortPressEvent>2</ShortPressEvent>
        <TimeoutEvent>3</TimeoutEvent>
        <Timeout>900</Timeout>
        <LongPressTime>20</LongPressTime>
        <IndicateOkOptions>2</IndicateOkOptions>
        <IndicateFailOptions>1</IndicateFailOptions>
        <ExecuteOnSuccess>1</ExecuteOnSuccess>
        <ExecuteOnFailure>1</ExecuteOnFailure>
      </Button>
    </Image>
  </Images>
  <Button>
    <ShortPressEvent>1</ShortPressEvent>
    <Timeout>0</Timeout>
    <LongPressTime>20</LongPressTime>
    <IndicateOkOptions>2</IndicateOkOptions>
    <IndicateFailOptions>1</IndicateFailOptions>
    <ExecuteOnSuccess>1</ExecuteOnSuccess>
    <ExecuteOnFailure>1</ExecuteOnFailure>
  </Button>
</EslImageInfo>
```

Tip:

The easiest way to generate and test the XML for a button event action is to create a template with a Button object and then link an ESL to the given template. The resulting image (incl the Button XML-code) will be stored in the “Images” table of the (logging) database. By default, it is a SQLITE database located in the Output-folder of your installation.

Button Settings

Event	Action
Short Press	Show next page
Long Press	Disabled
Very Long Press	Disabled
Double Click	Disabled
Timeout	Disabled

☒ Execute on success
☐ Execute on failure

Options

Timeout (sec) 30
Long press (ms) 2000
Indicate OK ■ Green
Indicate Fail ■ Red

Condition:

Template with button object

Table: images

	MAC	PAGE_ID	IMAGE_ID	STATUS	MD5	Image	LED	NFC	BUTTON
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	B4AE0029	0	4	1	4007686005270274547	BLOB			<?xml...
2	B28259E8	0	2	8	4937083473924505509	BLOB	<?...		

Mode: XML

```

1 <?xml version="1.0" encoding="utf-16"?>
2 <Button xmlns:xsi=
3   "http://www.w3.org/2001/XMLSchema-instance"
4   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
5   <ShortPressEvent>1</ShortPressEvent>
6   <LongPressEvent>9</LongPressEvent>
7   <VeryLongPressEvent>9</VeryLongPressEvent>
8   <DoubleClickEvent>9</DoubleClickEvent>
9   <TimeoutEvent>9</TimeoutEvent>
10  <Timeout>30</Timeout>
11  <ShortPage>1</ShortPage>
12  <LongPressTime>20</LongPressTime>
13  <IndicateOkOptions>2</IndicateOkOptions>
14  <IndicateFailOptions>1</IndicateFailOptions>
15 </Button>

```

Resulting image with button object in the Images table

6 Event handler

The event handler is a tool that can be used to make the ESL Web Server pass information to other systems or to its own API. One example is the handling of button events of the SE-290A.

Since the reaction expected from a button press can vary, it's possible to forward the button event to a Back Office system and send a response to the ESL based on the response of the back office.

The event handler page can be reached through the menu bar: Advanced > Event handler



Event handler page

6.1 Default behavior

If no event handler is enabled for handling button events, then the ESL Web server sends a default (Acknowledge) response to each button event it receives. The ESL will then perform the configured button event action (as specified in the template (or show a short white LED when no button object has been added to a template)

6.2 Enabling of the event handler

At the top of the page is a switch to turn all event handlers on or off simultaneously, which is useful for maintenance, or for preparing a system to go live with a single switch.

To create a new event handler, click on the large card decorated with the plus sign. This will open the event handler creation window.

Figure 1: Event handler creation window

The very first option is a switch that designates whether this event is active.

NOTE: this switch is ignored if the global switch, located on the main event handler page, is switched off.

The second option is a set of dropdowns, the left one regarding the source of the event, and the right one regarding the specific trigger within that event source you want to respond to.

6.3 Adding a button event handler

To handle a button event, select 'button' and then select the event you want to capture.

The third option consists of two textboxes: the action to perform and the parameters to send to the action. Valid actions are an API endpoint (make sure the URL is prepended with “http or https”), a path to an XML file (present on the host device) to execute, a path to a SQL file (present on the host device) to execute or a path to any other type of executable (present on the host device) to be performed.

Parameters are variables that you want to supply to the performed action. These can be static or read from the event (and therefore dependent on the type of event) like “%MAC%” for the 8-digit long MAC address of the ESL raising the event, or %ID% for the UID of the product the ESL raising the event is linked to.

Parameters on an API endpoint action are added to the URL query string, for the other actions they are supplied as parameters appended to the command line tool calling the action.

Besides the **trigger source** and **event** selected in the dropdown, you can specify another condition before the action is performed. This condition can be used to narrow down the specific event you want to react to. For instance, only when ESL is linked to a specific product range.

6.4 Input response API-endpoint

When a button event is received, the ESL expects a response within 2 seconds.

To send a button event response to an ESL the following endpoint is available

HTTP VERB	ACTION	MAC	Param(s)	Role
POST	Input response	MAC of ESL	See below	
Endpoint	https://{address}/api/ESL/{mac}/INPUT_RESPONSE/{event}/{response}/{seqnr}			

Parameter	Value
{mac}	8-digit MAC-address of the ESL (use %MAC% to automatically fill this in)
{event}	Button event type (use %EVENT% to automatically fill this in). <ul style="list-style-type: none"> PRESSED LONG_PRESSED VLONG_PRESSED DOUBLE_PRESSED
{event}	Response. Valid responses are: <p>ACK -> Acknowledged (keep fast polling, because a new update might be sent)</p> <p>NAK -> Not acknowledged (but keep polling fast a bit longer)</p> <p>ACK NO_RESPONSE -> Acknowledged (do not execute button event action)</p> <p>ACK DONE -> Acknowledged (stop fast polling, no image or LED update is coming)</p> <p>NAK DONE -> Not Acknowledged (stop fast polling, no image or LED update is coming)</p>
{seqnr}	4-byte Sequence number (8 hexadecimal digits). Use %SEQNR% to automatically fill this in.

The default event-handler to respond to a button event would be:

<https://localhost:{port}/api> POST ESL/%MAC%/INPUT_RESPONSE/%EVENT%/ACK/%SEQNR%

Event

Active: ON

If

Button

Pressed

Do

https://localhost:5001/api
POST

ESL/%MAC%/

INPUT_RESPONSE/

%EVENT%/ACK/%SEQNR%

Add condition

Delete

Cancel

Save

Default event handler

Example:

Using the default event handler, the response to a button press event could result in:

```
api/ESL/B4AE0029/INPUT_RESPONSE/PRESSED/ACK/004c0035
```

Where:

B4AE0029	MAC-address of the ESL
PRESSED	Event (PRESSED, LONG_PRESSED, VLONG_PRESSED and DOUBLE_PRESSED)
ACK	Acknowledged (keep fast polling, because a new image might be sent)
004c0035	4-byte Sequence number

For the SE-290A the parameters **%EVENT%** and **%SEQNR% *** are particularly important, because they specify which button event (short press, long press, or double press) was received and the unique 8-digit sequence number (**SEQNR**) of the event (so the ESL can identify the response as being valid and ignore duplicate responses)

** Note: Version 3.0.8.x' and lower of the ESL Web server, uses %MESSAGE% instead of %SEQNR%*

To respond to SE-290A button events these parameters are mandatory to acknowledge an input event.

6.4.1 Retrieving the page and image IDs from button events

From firmware version IxVA0200 and higher, the parameter **%VALUE%** can be used to verify from which **page** the button is fired and whether it is from an old image (for example when a new image has not yet been received) or from a current image.

%VALUE% contains 4 digits; 2 for the **Image ID** (00 = reported image is not in the image table anymore) and 2 for the **page number**. When **%VALUE%** is empty, then the default image is currently shown, or the ESL is running old firmware.

6.4.2 Input response API-endpoint with custom action and LED response

Since version 3.0.9.xxx and higher, the following endpoint has been added to overrule any button settings that were defined by templates (or by XML when using CMS).

HTTP VERB	ACTION	MAC	Param(s)	Role
POST	Input response	MAC of ESL	See below	
Endpoint	https://{address}/api/ESL/{mac}/INPUT_RESPONSE/{event}/{response}/{seqnr}/{action}/{led color}			

Parameter	Value	
{mac}	8-digit MAC-address of the ESL (use %MAC% to automatically fill this in)	
{event}	Button event type (use %EVENT% to automatically fill this in). <ul style="list-style-type: none"> PRESSED LONG_PRESSED VLONG_PRESSED DOUBLE_PRESSED 	
{event}	Response. Valid responses are: ACK -> Acknowledged (keep fast polling, because a new update might be sent) NAK -> Not acknowledged (but keep polling fast a bit longer) ACK NO_RESPONSE -> Acknowledged (do not execute button event action) ACK DONE -> Acknowledged (stop fast polling, no image or LED update is coming) NAK DONE -> Not Acknowledged (stop fast polling, no image or LED update expected)	
{seqnr}	4-byte Seq. number (8 hexadecimal digits). Use %SEQNR% to automatically fill this in	
{action}	Valid actions are: NoAction ShowNextPage ShowPrevPage ShowPageX ShowDefaultImage RefreshImage InvertImage TurnOffLed TurnOnLed ToggleLed	Performs no operation. Displays the next page (if present). Displays the previous page (if present). Displays a specific page number (X = 1 -> page 1, etc.) Shows the device's default image. Refreshes the current image. Inverts the colors of the current image Turns the LED off (as defined by the current image). Turns the LED on (as defined by the current image). Toggles the LED state between on and off.
{led_color}	Valid colors are: None Red Green Blue Orange Magenta Cyan White	

6.5 Performance

6.5.1 Direct API-response

The fastest way to make an SE-290A respond to a button event is by using an **event handler** that replies with an `INPUT_RESPONSE`.

The response can either specify the action dynamically or use a predefined template action.

A typical response to an input event is an `INPUT_RESPONSE` encoded in an `EslServerApiObject`:

```
<EslServerApiObject
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Action>INPUT_RESPONSE</Action>
  <MAC>B4AE0029</MAC>
  <Param1>PRESSED</Param1>
  <Param2>ACK</Param2>
  <Param3>01020303</Param3>
  <Result>OK</Result>
</EslServerApiObject>
```

Field	Description
INPUT_RESPONSE	Response action to a button event
B4AE0029	MAC address of the ESL
PRESSED	Event type (PRESSED, LONG_PRESSED, VLONG_PRESSED, DOUBLE_PRESSED)
ACK	Acknowledgment flag — keeps fast polling active in case a new image is sent
01020303	4-byte sequence number (use %SEQNR% in the event handler)
OK	API result

However, in some situations it is desired to trigger an update of the image by modifying the product database or to execute an API-call directly (i.e. an LED command), even when there's no incoming API-access through a firewall.

To achieve this, it's possible to respond to an incoming button event directly with an API-call embedded in the response.

The response format would be an `EslServerApiObject` with an API-endpoint in the 'Action' field and an API-key in a field called 'XApiKey'

```
<EslServerApiObject xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Action>api/products</Action>
  <MAC>B4AE0029</MAC>
  <Param1>POST</Param1>
  <Param2>001</Param2>
  <Param3>Description</Param3>
  <Param4>It works :)</Param4>
  <Result>OK</Result>
  <XApiKey>444596F9-CC6B-4FA2-9188-2BBAD678916B</XApiKey>
</EslServerApiObject>
```

Field	Description
api/products	API endpoint (see the <i>API chapter</i> of the ESL Web Server Manual)
B4AE0029	MAC address of the ESL
POST	HTTP method (GET, POST, PUT, or DELETE)
001	Parameter 1 (product ID)
Description	Parameter 2 (product table column)
It works :)	Parameter 3 (new value for the specified column)
OK	API result
XApiKey	API key with sufficient rights to alter products

The API key can also be added in the **HTTP header** of the response, by adding the header: **x-api-key**

HEADERS	BODY	AUTHORIZATION	VARIABLES
Name	Value		
x-api-key	E85E22A5-8256-4802-8D8F-B539EA065CB0		

6.5.2 Keep-alive

The largest delay in executing API calls to a secure endpoint comes from establishing the **SSL connection**, which can easily take up to a second.

To minimize this delay, you can configure an event handler to perform a **keep-alive call** at regular intervals (e.g., every 1 minute). This keeps the connection warm and reduces latency for subsequent API calls.

Example: an event handler that calls a keep-alive endpoint every minute.

Event

Active: ☒

If Scheduled

Time

00:00

☒ Check to perform task every minute

And condition is met,

Do

https://localhost:44376 GET

api/KEEP-ALIVE

Delete

Cancel

Save