

## Overview

EZX plugins are a powerful toolset to extend EZX functionality without modifying core application code.

EZX plugins use the following basic technologies:

1. **EZX Plugin API** - creates plugin configuration, attaches plugin to EZX site, provides PortalDI connection interface, etc.
2. **PortalDI API** - allows calls to webnative, using PortalDI ([more info can be found here](#))
3. **Smarty template engine API** - Smarty is a powerful template engine for PHP, that is used to enhance plugin templates ([more info can be found here](#))
4. **JavaScript (jQuery) 3.+** - can be used to add dynamic functionality to the plugin UI ([more info can be found here](#))
5. **Bootstrap 4.+** – the world’s most popular framework for building responsive, mobile-first sites ([more info can be found here](#))

You need some knowledge of PHP, JavaScript and CSS to start creating EZX Plugins. Download sample plugins [PreviewPlugin](#) (standard EZX plugin) and [xCsvMapPlugin](#) to use as a reference.

# Installation

1. Create or obtain a plugin package
2. Login to your EZX admin interface
3. Go to the **Plugins** page:

Name ▲	Sites	Info	Active	
<input type="checkbox"/> PreviewPlugin	default, demo121, local, tmsw	Title: Preview Version: 1.4 Description: Simple demo plugin that allows to open large image preview in popup or separate window	Yes	
<input type="checkbox"/> SupermailPlugin	default, demo121, tmsw	Title: SM2 Integration Version: 0.8 Description: Plugin allows to send basket contents or single files using NAPC Supermail 2.0	Yes	

4. Click **Actions** → **Install Plugin** and choose a plugin package file you want to install
5. Make sure the plugin is successfully added to the **Plugins** list:

Name ▲	Sites	Info	Active	
<input type="checkbox"/> CsvMapPlugin		Title: CSV Map Version: 1.1 Description: Allows to download a CSV file containing file and keyword names	Yes	
<input type="checkbox"/> PreviewPlugin	default, demo121, local, tmsw	Title: Preview Version: 1.4 Description: Simple demo plugin that allows to open large image preview in popup or separate window	Yes	
<input type="checkbox"/> SupermailPlugin	default, demo121, tmsw	Title: SM2 Integration Version: 0.8 Description: Plugin allows to send basket contents or single files using NAPC Supermail 2.0	Yes	

## Locating Plugin Installation Folder

The plugins will be installed into the following folder - **/<ezx install path>/api/lib/plugins.**

## Basic Plugin Structure

Let's continue with our sample plugin **PreviewPlugin**. If you look inside its root folder (`<ezx install path>/api/lib/plugins/PreviewPlugin`), you'll see the following structure:

- `config.json` → contains plugin configuration
- `form.json` → contains plugin form configuration (optional)
- `PreviewPlugin.php` → main plugin php file
- **CSS** → contains plugin `.css` files (optional)
- **JS** → contains plugin `.js` files (optional)
- **Templates** → contains plugin Smarty template files (optional)

*NOTE: plugin folder name has to end with "Plugin": PreviewPlugin, DemoPlugin, MyTestPlugin, etc*

*NOTE: the main plugin php file should ALWAYS be named the same as the plugin folder, excluding starting optional lowercase "x" like in **xCsvMapPlugin***

*NOTE: you may add other folders/files into the plugin folder as required*

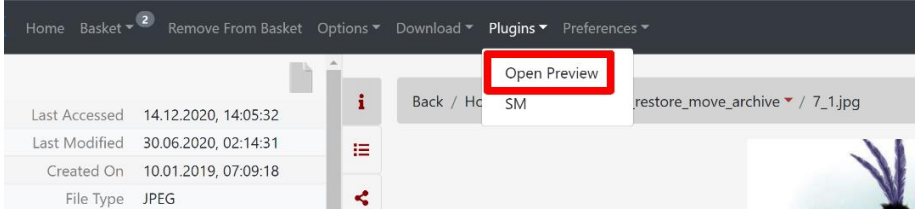
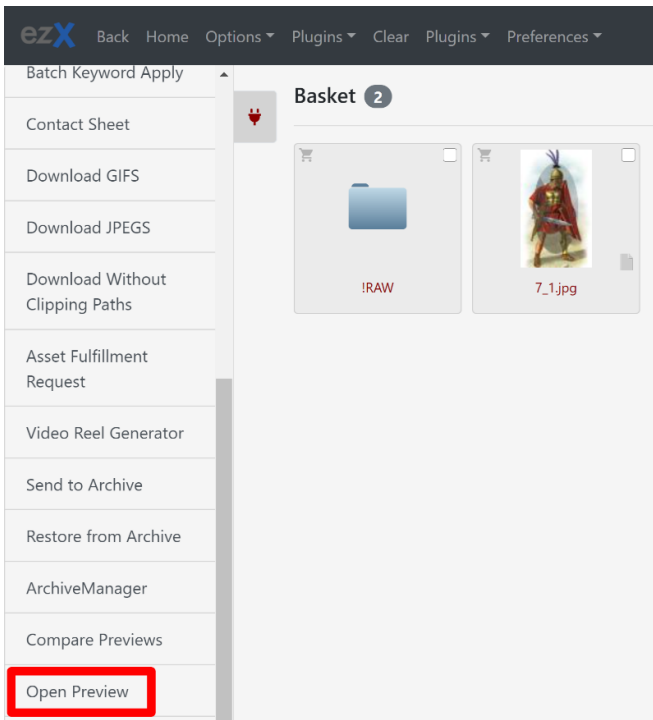
### Creating Plugin Package

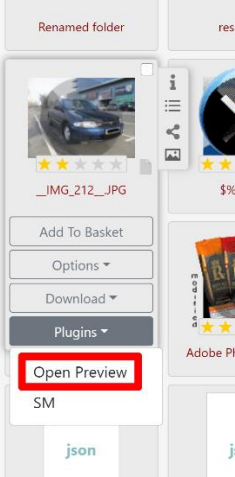
Simply zip the plugin folder and if all the requirements (see above) are met – you'll get an installable EZX plugin

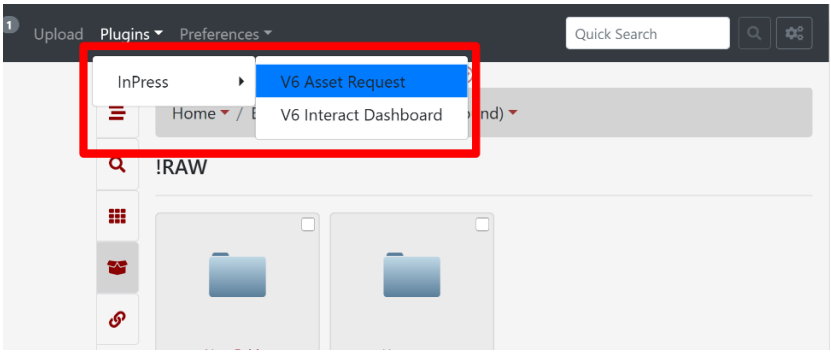
# config.json

This file determines how the plugin will be attached to EZX interface.

## Available config.json options:

title	Plugin title. Can be any value, name it something intuitive
version	Plugin version. Helps users to understand which plugin version they currently have
type	Currently the only available value is <b>"inline"</b> . You can skip this parameter. In this case plugin will stay <b>"invisible"</b> . You with <b>"inline_js"</b> to allow custom JS code execution inside EZX UI
description	Custom plugin description. May include any info you find necessary
actions	<p>An array of EZX pages the plugin will be attached to. Possible values: <b>"generic", "home", "browse", "basket", "basket-plugin" "view"</b></p> <p><i>NOTE: set options <b>applyToFiles</b> and/or <b>applyToFolders</b> to <b>true</b> if you want to use action <b>"view"</b></i></p> <p><i>NOTE: the plugin will be added to the main menu of the selected pages (excluding <b>"login"</b> page):</i></p>  <p><i>NOTE: use <b>"basket-plugin"</b> action to attach your plugin to the list of <b>Webnative</b> basket plugins:</i></p> 
supportsEmptyBasket	Set to <b>true</b> to ensure that the plugin button remains active even if the basket is empty

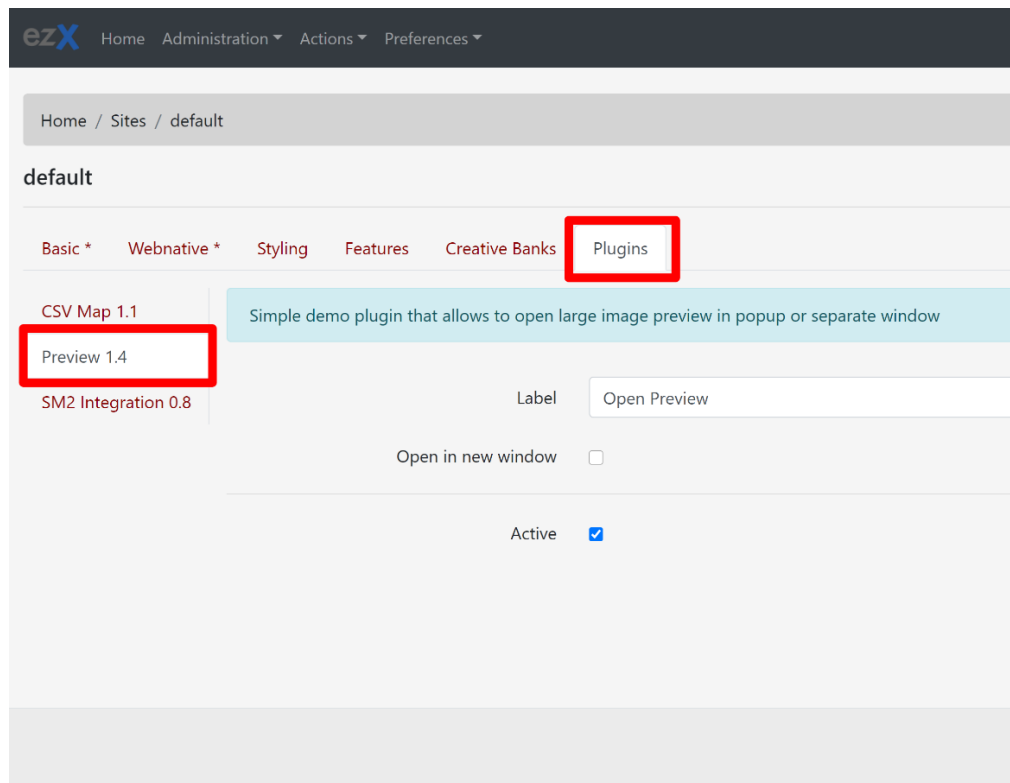
	<i>NOTE: action <b>"basket-plugin"</b> is required for this option to work</i>
applyToFolders	<p>If set to <b>true</b>, the plugin button will be automatically added to every folder menu</p> <p><i>NOTE: action <b>"view"</b> is required for this option to work</i></p>
applyToFiles	<p>If set to <b>true</b>, the plugin button will be automatically added to every file menu:</p>  <p><i>NOTE: action <b>"view"</b> is required for this option to work</i></p>
enabledAssetsOnly	<p>If set to <b>true</b>, the plugin button will be added to enabled (unexpired) assets only</p> <p><i>NOTE: action <b>"view"</b> is required for this option to work</i></p>
onlineAssetsOnly	<p>If set to <b>true</b>, the plugin button will be added to online (not archived) assets only</p> <p><i>NOTE: action <b>"view"</b> is required for this option to work</i></p>
isNewWindow	Set to <b>true</b> , to open the plugin in a new browser tab, rather than popup window
width	<p>"&lt;num&gt;px" or "&lt;num&gt;%" – can be used to set plugin popup window width</p> <p><i>NOTE: set <b>isNewWindow</b> to false to utilize this option</i></p>
height	<p>"&lt;num&gt;px" or "&lt;num&gt;%" – can be used to set plugin popup window height</p> <p><i>NOTE: set <b>isNewWindow</b> to false to utilize this option</i></p>
maximizable	<p>Set to <b>true</b> to allow maximizing of the plugin popup window</p> <p><i>NOTE: set <b>isNewWindow</b> to false to utilize this option</i></p>
maximized	<p>Set to <b>true</b> to make plugin popup window maximized by default</p> <p><i>NOTE: set <b>isNewWindow</b> to false to utilize this option</i></p>
css	An array of relative paths to .css files that will be attached to the plugin page
js	An array of relative paths to .js files that will be attached to the plugin page

inline_js	Relative path to .js file that will be attached to EZX pages to which the plugin is linked through <b>actions</b> option. Can be used to modify EZX page UI.
template	Relative path to the main plugin Smarty template file
dynamicConfig	Set to <b>true</b> to override <b>config.json</b> inside a main plugin php file (see <b>Main Plugin PHP File</b> below for more info)
map	<p>Allows to create a set of sub-plugins with different configurations based on a single plugin:</p>  <p>Map example:</p> <pre> {   "home": [     {       "title": "Map Sample",       "url": "http://google.com"     }   ],   "view": [     {       "title": "New Window",       "isNewWindow": true     },     {       "title": "Popup",       "isNewWindow": false,       "maximized": true     }   ] } </pre> <p><b>NOTE:</b> map property names should match chosen action names</p> <p><b>NOTE:</b> map supports the following actions: <b>“home”</b>, <b>“browse”</b>, <b>“basket”</b>, <b>“view”</b></p> <p><b>NOTE:</b> each property value is an array of plugin configurations</p>
url	<p>Set this option to override internal EZX plugin url.</p> <p><b>NOTE:</b> this is useful if you want the plugin to redirect to an external web page</p> <p><b>NOTE:</b> depending on the current page (<b>“home”</b>, <b>“browse”</b>, <b>“basket”</b>, <b>“view”</b>) EZX can pass additional parameters to provided url. To add parameters to the url, format it as follows:</p> <p><a href="http://external.page.com?param={&lt;ezx_param1&gt;}&amp;param2={&lt;ezx_param2&gt;}.">http://external.page.com?param={&lt;ezx_param1&gt;}&amp;param2={&lt;ezx_param2&gt;}</a>.</p>

*You can execute the plugin without url property to see which parameters are available*

# form.json

This file contains a json array of form-control objects. Each form control object will be shown as an input element in the plugin configuration form (in the site settings):



**NOTE:** each EZX site will have its own plugin configuration

## Available form control parameters:

type	“text”, “textarea”, “password”, “checkbox”, “select”, “radio”, “datetime”
label	Arbitrary label that will be associated with the given form control
value	Default value
separator	Set <b>true</b> to add “<hr>” separator above the form control
hint	Arbitrary hint that will be shown as a tooltip for “?” icon: 



maxlength	Maximum number of characters for a text input field (default is 255)
rows	Number of rows for a textarea (default is 5)
readonly	Set to <b>true</b> to show control as readonly
options	An object that contains options for “ <b>select</b> ” control element. Example: <code>{"opt1": "option1", "opt2": "option2"}</code>
hasEmptyValue	Set to <b>true</b> if you want “ <b>select</b> ” control element to have initial empty value
validators	An object that contains form control value validators. Below you can find the list of all validators with available parameters: <pre>{   "required":true,   "emails":true,   "email":true,   "length": {     "length":&lt;number&gt;,     "min":&lt;boolean&gt;(default - false),     "isMultiline":&lt;boolean&gt;(default - false)   },   "max": {     "maxValue":&lt;number&gt;,     "compareToControl":&lt;string&gt;(another control element name to compare the value against)   },   "min": {     "minValue": &lt;number&gt;,     "compareToControl":&lt;string&gt;(another control element name to compare the value against)   } }</pre>

## Special control elements

You can add the following optional control elements to **form.json**:

is_active	<code>{"value": true}</code>  Allows to enable/disable the plugin
-----------	---

	<i>NOTE: <b>is_active</b> element may be omitted. In this case it will be added automatically and set to <b>false</b></i>
is_new_window	<pre>{   "type": "checkbox",   "value": false,   "label": "Open in new window" }</pre> <p>Allows to control how plugin is displayed – in a separate browser tab or popup.</p> <i>NOTE: overrides config.json <b>isNewWindow</b> option</i>

## Main Plugin PHP File

- Main plugin file has to have the same name as plugin folder,
- Plugin class, described inside this file has to have the same name as plugin file
- Plugin class, described inside this file has to extend **AbstractTemplatePlugin** (for a plugin with UI) or **AbstractPlugin** (for a plugin without UI) EZX API classes

*NOTE: EZX API was developed to support PHP version 5.2 or higher*

### Methods, available for AbstractPlugin descendants:

name	getCredentials
description	Can be used to get basic information about current session
parameters	-
returns	<b>Credentials</b> object that contains the following data: <ul style="list-style-type: none"><li>• username,</li><li>• password,</li><li>• siteId,</li><li>• type,</li><li>• wnHost,</li><li>• wnUserId,</li><li>• wnUserGroups</li><li>• cbUsername</li></ul>

name	getParameters
description	Can be used to obtain \$_GET and \$_POST parameters, passed to the plugin
parameters	-
returns	<b>Parameters</b> object that contains the following data: <ul style="list-style-type: none"><li>• get,</li><li>• post,</li><li>• config</li></ul>

name	getUrl
description	Can be used to get plugin url with additional parameters attached
parameters	<ul style="list-style-type: none"><li>• <b>parameters</b> (array) – array of parameters that will be attached to the url</li></ul>
returns	string – generated plugin url

name	getSetting
description	Can be used to obtain a plugin setting for the current site

parameters	<ul style="list-style-type: none"> <li>• <b>setting</b> (string) – setting name as configured in <b>form.json</b> file</li> </ul>
returns	mixed   null – setting value or null if setting was not found

<b>name</b>	<b>getSiteSetting</b>
description	Can be used to obtain a setting for the current site
parameters	<ul style="list-style-type: none"> <li>• <b>setting</b> (string) – setting name</li> </ul>
returns	mixed   null – setting value or null if setting was not found

<b>name</b>	<b>getSiteSettings</b>
description	Can be used to obtain all settings for the current site
parameters	-
returns	array containing all current site settings

<b>name</b>	<b>getDataFromServer</b>
description	Can be used to perform a <b>portalDI</b> call and obtain data from <b>Webnative</b> server as an array
parameters	<ul style="list-style-type: none"> <li>• <b>actionName</b> (string) – a list of all available portalDI actions can be found <a href="#">here</a></li> <li>• <b>getParams</b> (array) – a list of params that will be passed into PortalDI as \$_GET</li> <li>• <b>postParams</b> (array) – a list of params that will be passed to portalDI as \$_POST</li> </ul>
returns	array   false – array containing response data or false on error

<b>name</b>	<b>getRawDataFromServer</b>
description	Can be used to perform a <b>portalDI</b> call and obtain data from <b>Webnative</b> server in a “raw” state. Useful to get thumbnails, previews or download files
parameters	<ul style="list-style-type: none"> <li>• <b>actionName</b> (string) – a list of all available portalDI actions can be found <a href="#">here</a></li> <li>• <b>getParams</b> (array) – a list of params that will be passed into PortalDI as \$_GET</li> <li>• <b>postParams</b> (array) – a list of params that will be passed to portalDI as \$_POST</li> </ul>
returns	string containing response data

## Static functions, that can be implemented in AbstractPlugin descendants

These static functions can be used to dynamically override plugin configuration or execute additional business logic:

<b>name</b>	<b>getDynamicConfig</b>
description	Implement it to override some/all settings in <b>config.json</b> file and/or add more arbitrary parameters to it.  This config object can be reached from outside (for example from inline_js file) with the following path: <b>document.exposed.sitePlugins['&lt;PLUGIN_NAME&gt;']</b>
parameters	<ul style="list-style-type: none"> <li>• <b>credentials</b> (Credentials) – credentials object</li> <li>• <b>settings</b> (array) – plugin configuration for the current site</li> </ul>
returns	array – array that will be later merged with the settings from <b>config.json</b> file

<b>name</b>	<b>onInstall</b>
description	Function may contain additional business logic, executed on plugin install. For example, it can be used to install some third-party libraries, make configurations based on server environment, etc.
parameters	-
returns	void

<b>name</b>	<b>beforeSave</b>
description	Function may contain additional business logic (for example validation), executed before site save
parameters	<ul style="list-style-type: none"> <li>• <b>siteId</b> (integer) – current site id</li> <li>• <b>settings</b> (array) – plugin configuration for the current site</li> </ul>
returns	void

<b>name</b>	<b>onSave</b>
description	Function may contain additional business logic, executed on site save
parameters	<ul style="list-style-type: none"> <li>• <b>siteId</b> (integer) – current site id</li> <li>• <b>settings</b> (array) – plugin configuration for the current site</li> </ul>
returns	void
<b>name</b>	<b>overrideForm</b>
description	Function can override plugin form before it is displayed in the site configuration UI
parameters	<ul style="list-style-type: none"> <li>• <b>form</b> (array) – plugin form configuration</li> </ul>
returns	array – array that can be used as <b>form.json</b> substitution

### Constants that can be set in AbstractPlugin descendants:

<b>name</b>	<b>ACCESS_VERIFICATION_NEEDED</b>
type	boolean (default - true)
description	Set to <b>false</b> to allow plugin execution by anonymous (not logged in) users

### Methods, available for AbstractTemplatePlugin descendants:

<b>name</b>	<b>getTemplateVariables</b>
description	This method returns an array of data which then can be used in the template file(s)
parameters	-
returns	array

<b>name</b>	<b>getJavascriptVariables</b>
description	This method returns an array of data that is turned into javascript variables by the template engine. You can then use these variables in the plugin javascript code
parameters	-
returns	array

<b>name</b>	<b>onDisplay</b>
description	All plugin business logic can reside in this method, which is called before plugin display
parameters	-
returns	array

## Templates

The folder may contain .html files with Smarty templates. While only one template can be the default one (set up in **config.json**) there may be more .html files linked to each other using [Smarty API](#).

## Exceptions Handling

If you need to throw an exception, you can utilize EZX API exception call –

```
throw new ApiException($message, $code);
```

This exception will be registered in the EZX error log (`/<ezx install path>/logs/error.log`) and properly displayed in the plugin UI:

### Error 400

---

This is an exception test



# Logging

If you need to log some data, you can use EZX built-in global logging function

<b>name</b>	<b>execLogHandler</b>
description	Logs arbitrary data into a file in “/<ezx install path>/logs” directory
parameters	<ul style="list-style-type: none"><li>• data (array) – arbitrary data you want to log</li><li>• logfile (string) – arbitrary log file name</li><li>• enabled (boolean) – set to <b>true</b> to log data. Omit this parameter to log data only if the application is in <b>debug</b> mode (application url contains debug=1 parameter)</li></ul>
returns	void

# Sample Plugins

## [PreviewPlugin](#)

Simple demo plugin that allows to open large image previews in popup or separate windows. The plugin is a standard part of EZX. The plugin extends **AbstractTemplatePlugin** class and has its own template and UI.

## [CsvMapPlugin](#)

Allows to download a CSV file containing file and keyword names. The plugin extends **AbstractPlugin** and does not have a UI. Instead, it's JS is integrated into the **browse** EZX page. The plugin uses **document.exposed** JS object that contains data populated by EZX to be used in third party JS scripts