

1 Contents

2	Enter setup	2
2.1	Create configuration list	2
2.2	Enter setup or add / change the license code	3
2.2.1	Enter setup	3
2.2.2	Production or development mode	3
2.2.3	Generate challenge code	4
3	Configuration of a form	5
3.1	Banner buttons	6
3.1.1	Save	6
3.1.2	Import /Export	6
3.1.3	Create restore point	6
3.1.4	Switch form	6
3.1.5	Delete	6
3.2	Formbuilder	7
3.2.1	Row	7
3.2.2	Column	7
3.2.3	Tabset	7
3.2.4	Rich text	8
3.2.5	HTML	8
3.2.6	Grid	8
3.2.7	Fields	8
3.3	Preview form	13
3.4	Rules	13
3.5	Custom JS	15
3.6	Custom CSS	16
3.7	Miscellaneous	17
3.7.1	Password	17
3.7.2	Form panel type	17
3.7.3	Comments	17
3.7.4	Tab colors	17
4	Modern DFFS WebPart	18

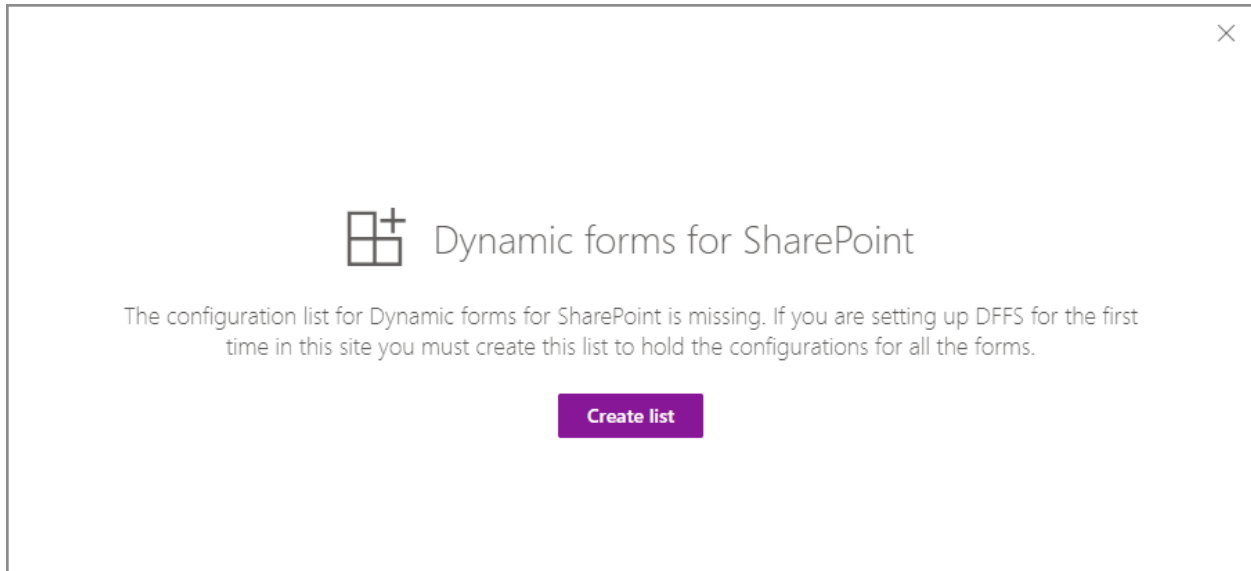
4.1	Setting a default redirect in the form URL	19
5	Troubleshooting	20
5.1	Blocked web resources.....	20
5.2	Other blocked resources	20

2 Enter setup

2.1 Create configuration list

When you have installed the Modern DFFS you will see a button “DFFS” in the banner of the list. This will only show for users with **Manage lists** permission. Click this button to enter setup.

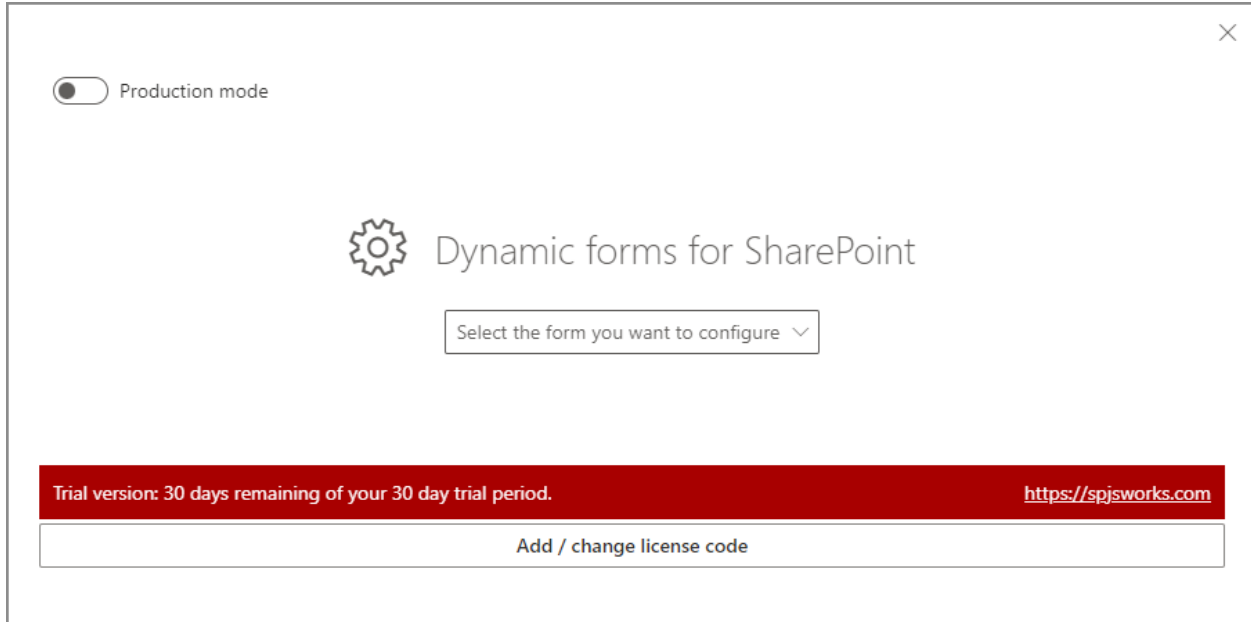
The first time you enter setup you must create the configuration list:



This configuration list is a standard *custom list*, but it is by default hidden from all site contents. You are not supposed to manually edit this list, but it can be accessed by typing in the list name in the URL like this: `.../Lists/DFFSConfigurationList`

2.2 Enter setup or add / change the license code

When you have created the configuration list, you will see this screen:



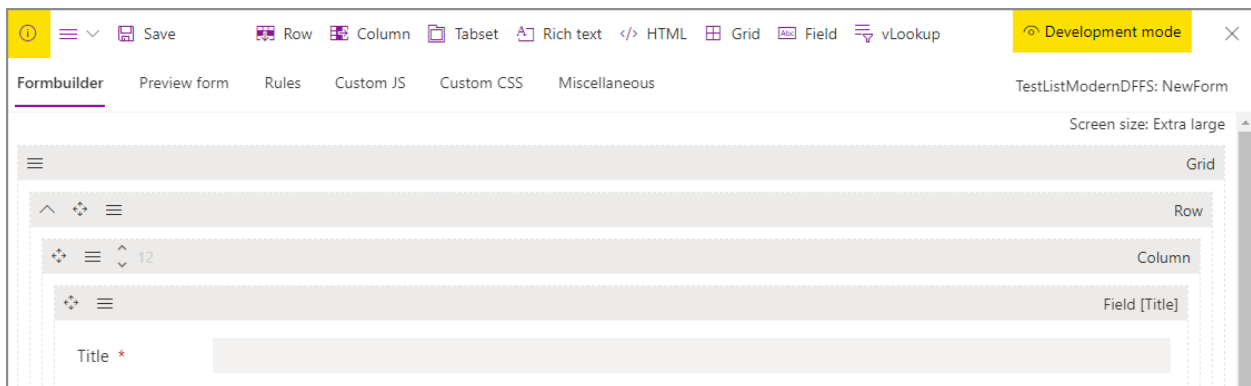
2.2.1 Enter setup

Here you can select the form you want to configure from the drop-down menu. You can choose from the following forms: NewForm, DispForm and EditForm.

You use NewForm to create new list items (a document library does not have an NewForm), DispForm to display an existing list item and EditForm to edit an existing list item.

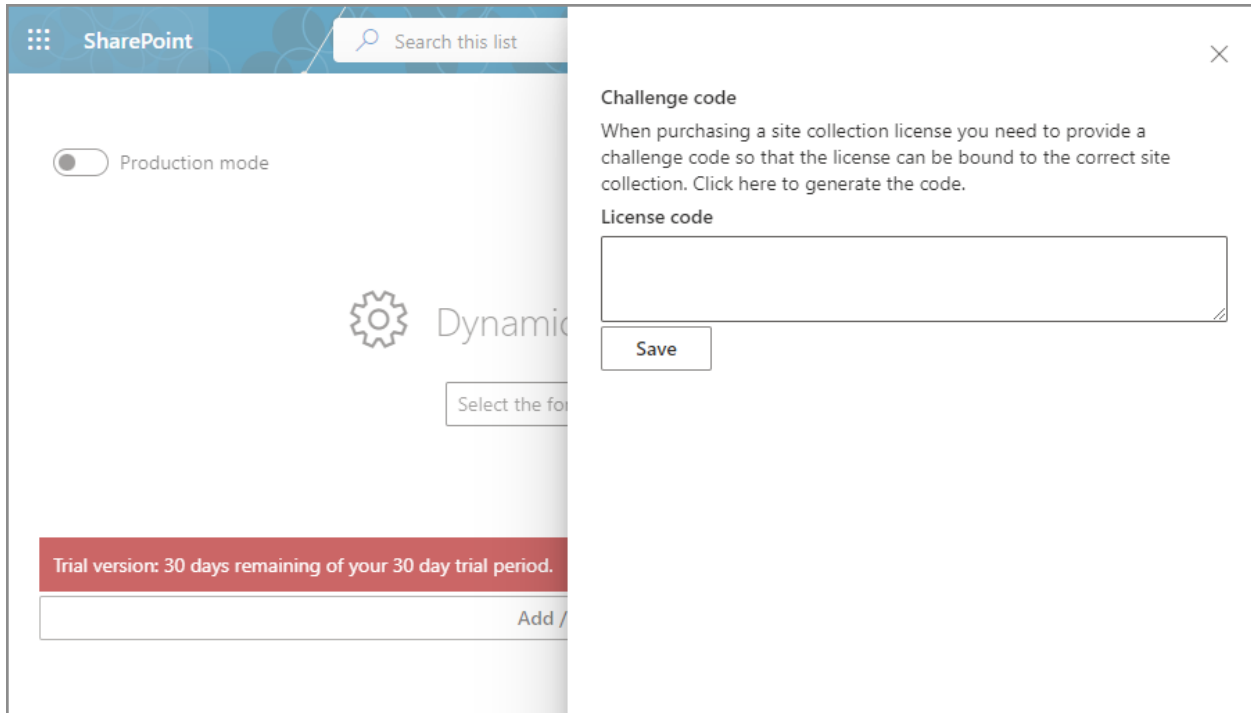
2.2.2 Production or development mode

You can toggle between production and development mode using the toggle in the top left corner. When toggling this to Development mode and entering a form configuration you will see a yellow button in the top right corner. If you click this button, you can assess your development mode configuration in a new window.



2.2.3 Generate challenge code

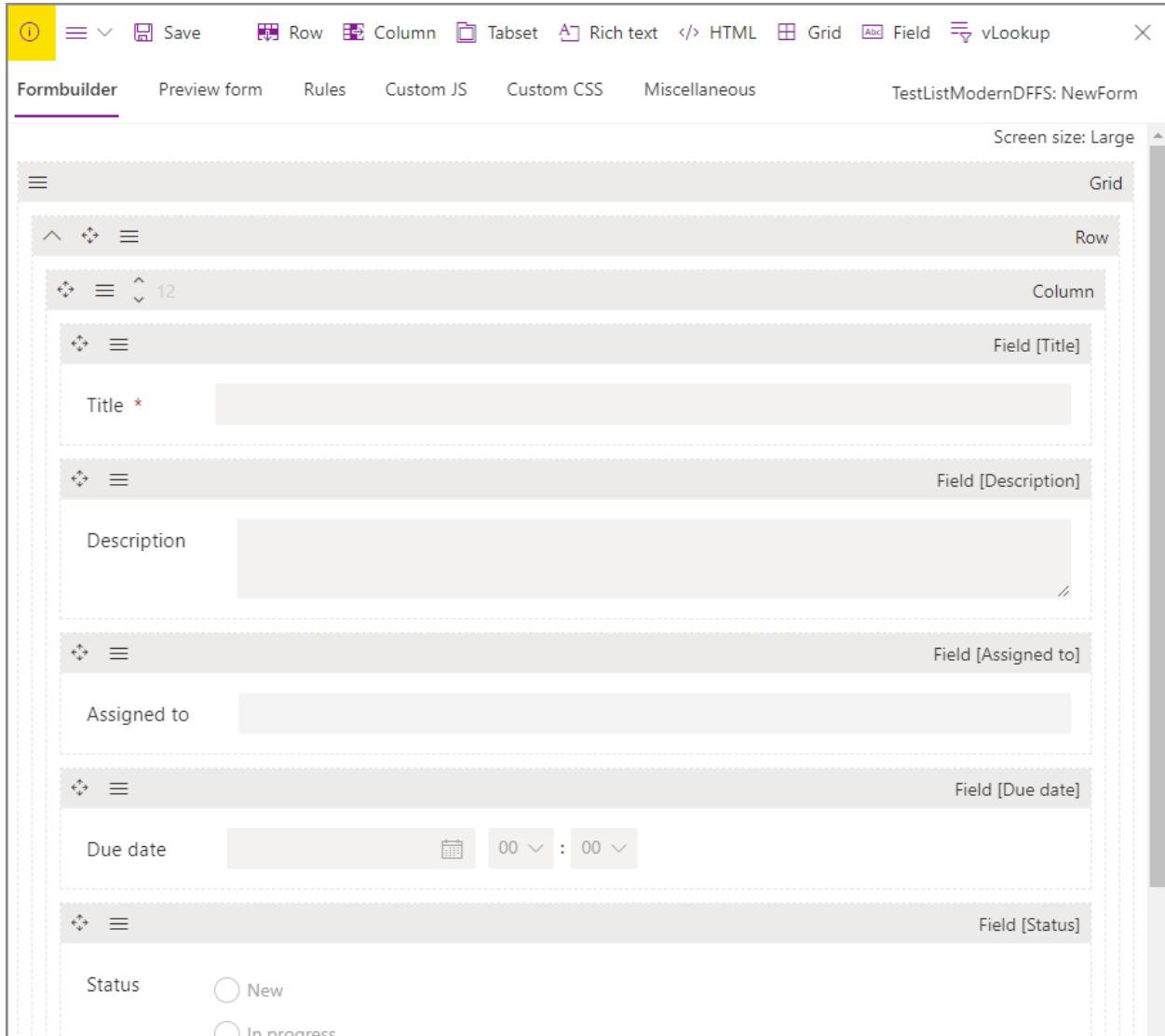
When you purchase a site collection license you must generate a challenge code to bind the license code to the current site collection. Click the **Add / change license code** to access the challenge code.



You get a 30-day trial when you first install the solution. You can purchase a license or ask for a quote from the here: <https://spjsworks.com/purchase>

3 Configuration of a form

When you open the configuration for the first time, all fields are automatically added to the form. You can delete the ones you do not want to have visible in the form. Deleting a field from the configuration for one form does not delete it from the list settings.



To add new fields to your list you must use the default list settings. You find a link to open list settings from the Miscellaneous tab.

You can rearrange the fields by drag-and-drop, and you can access the properties pane for all elements by using the burger-button or on the right click menu. The different form elements have different properties – you find a bit more information in the next section.

3.1 Banner buttons

3.1.1 Save

Saves the configuration.

3.1.2 Import /Export

This opens a panel where you can browse through other configurations and restore points created for this list, import configurations from the Classic DFFS or import / export as JSON.

3.1.3 Create restore point

Create a restore point that lets you change back to this configuration if needed. It is recommended that you create a restore point before doing major changes.

3.1.4 Switch form

Switch between NewForm, DispForm and EditForm of the list (NewForm is not available in document libraries).

3.1.5 Delete

Use this to permanently delete the configuration and change back to the out of the box SharePoint form.

3.2 FormBuilder

Use the banner buttons to drag-and-drop form elements onto the form.

You can choose from the following elements:

- Row
- Column
- Tabset
- Rich text
- HTML
- Grid
- Field
- Lookup

You can configure all form elements by opening the properties pane using the burger-button or the right click menu.

3.2.1 Row

Rows are used as placeholders for columns. You can add an unlimited number of rows to your form.

3.2.2 Column

A row has 12 available sections. You can have one column that takes up all 12 section or add any number of columns (up to 12) and distribute the sections between them.

You can set the distribution of space depending on the screen size to make a more responsible for different screen sizes.

3.2.3 Tabset

You can create tabsets and add any number of tabs. If you do not want to use the default tab color (if is based on the color theme of your site) you can set the color of the individual tab, or add a default color for all tabs in the Miscellaneous tab. You can make a tabset “sticky” to ensure it stays on the top of the screen when scrolling a long form.

Each tab will have a placeholder where you can add rows and columns.

3.2.3.1 *Select same tab when navigating from DispForm to EditForm*

If you use the same tabs in DispForm and EditForm and want to select the same tab in EditForm when navigating from DispForm you can set up a rule in EditForm triggering on *Form is loaded*, add the *Select tab* action and select the option *When navigating from DispForm, select the same tab*.


3.2.3.2 *Set selected tab in a tabset*

When configuring a tabset you can set the default selected tab for each tabset.

If you want to create a link to a form and want to select a specific tab - that is not the default selected tab specified in the tabset - you can create a link with an URL query string key like this:

sTab=[id_of_the_tab]

To find the ID of your specific tab you must right click the tab you want to select and select *Inspect* to use the developer tools in your browser. This will open the developer tools and show something like this:



```
ms-selected="597" name="Case" data-content="Case" data-is-focusable="true" tabindex="0" >@</button>  
▼ <button type="button" data-highlight="0" data-id="9502543918899333" role="tab" aria-selected="false"  
id="Pivot240-Tab1" class="ms-Button ms-Button--action ms-button--command ms-Pivot-link link-398"  
name="Comments" data-content="Comments" data-is-focusable="true" tabindex="0">  
  ::before  
  ▼ <span class="ms-Button-flexContainer flexContainer-164" data-automationid="splitbuttonprimary">  
    flex  
    ▼ <span class="ms-Pivot-linkContent linkContent-394">  
      ▶ <span class="ms-Pivot-text text-395">... </span> == $0  
    </span>  
  </span>  
  ::after  
</button>  
▶ <button type="button" data-highlight="0" data-id="0578741705076600" role="tab" aria-selected="0" >
```

The number is used in the URL key like this: sTab= 9502543918899333

3.2.4 Rich text

You can use this control to add your own rich text to the form. This is not text that will be saved to the list, but instructions to the user that fill in the form.

3.2.5 HTML

You can add your own custom HTML. Accompany this with Custom JS and Custom CSS to create your own form elements. The values in any HTML-elements you create will not be saved to the list when saving your form element unless you use your own Custom JS to do so.

3.2.6 Grid

Use this to make a grid layout where you can add rows and columns.

3.2.7 Fields

You can select from all the built in SharePoint field types. To be able to add a field to the form, you must first create it in the default list settings for that list. You find a button to open the list settings in the Miscellaneous tab.

3.2.7.1 *Render a single line of text field as a drop-down*

You can create a drop-down menu on a single line of text field where the options are pulled in from another custom list in the site collection.

Render field as dropdown

Yes

Display name or GUID (include curly braces) of source list

The GUID of the source web (leave empty if the list is in the current site)

Internal name of the field you want to get the options from

Placeholder text when the select is empty

REST filter to get a subset of items for the source list

If you use the value from another field in the filter, this dropdown will be redrawn if that field is changed.

Autofill if the dropdown only has one option

No

Open the field properties and toggle **Render as dropdown** to Yes.

Now specify the display name or the GUID of the source list, the internal name of the field you want to use as the selectable options, the placeholder text, and any REST filter you want to use to filter the data source.

You can also check the **Autofill if the dropdown only has one option** to have it automatically fill in the value when it is the only option.

You can make your custom dropdowns cascading by using the **REST filter to get a subset of items for the source list** option. Use the **Add dynamic content** menu in the bottom right corner (where you focus the filter field) to pick the field you want to use as filter. The filter is updated when that field is changed.

3.2.7.2 *Render a lookup field as a treeview*

A lookup column can be rendered as a treeview. Open the field properties and toggle **Render as treeview** to Yes.

Render as a treeview

Yes

Internal name of field that holds the key to the parent item

This field must be empty on all the root level items. On all child items it must have a value matching the value from the **Country_x002d_Region** field in a parent item so they can be ordered in the parent-child hierarchy used to draw the treeview.

Selectable

Internal name of the boolean field that determines whether or not an item is selectable. Leave empty if not in use.

You must have a field that links the child items to the parent to maintain the parent-child relationship. For example, a field named Parent that for the child items holds the Title of the parent item.

You can also have a Boolean field in your list that determines whether you can select each item or not.

Here is an example where the Selectable option is set to false for Alabama.

The screenshot shows a treeview with the following structure:

- > England
- ∨ USA
 - ∨ Alabama
 - Autauga County
 - Baldwin County
 - Arizona

3.2.7.3 *Create cascading lookups*

You can make your lookup fields cascading by using the Filter option. Use the **Add dynamic content** menu in the bottom right corner (where you focus the filter field) to pick the field you want to use as filter. The filter is updated when that field is changed.

3.2.7.4 *vLookup*

Use this control to show a table view of items from a child list. You configure the control by selecting the web, list and filter to retrieve the desired items.

You can select which fields to populate the table and configure **Add new** to be able to create a new child item that is automatically linked to the current item by prefilling values from the current item (the same values as you use in the filter).

3.2.7.5 Autocomplete

You can render a single or multiline plain text field as an autocomplete lookup field that lists the options from another list in your SharePoint site collection. Configuration is done in the field properties like this:

Render field as autocomplete search box

Yes

Display name or GUID (include curly braces) of source list

The GUID of the source web (leave empty if the list is in the current site)

Internal name of the field you want to get the options from

Placeholder for the search field

REST filter to get a subset of items for the source list

If you use the value from another field in the filter, this autocomplete will be redrawn if that field is changed.

Open the field properties and toggle **Render field as autocomplete search box** to Yes.

Now specify the display name or the GUID of the source list, the internal name of the field you want to use as the selectable options, the placeholder text, and any REST filter you want to use to filter the data source.

You can also check the **Autofill if the dropdown only has one option** to have it automatically fill in the value when it is the only option.

You can make your custom dropdowns cascading by using the **REST filter to get a subset of items for the source list** option. Use the **Add dynamic content** menu in the bottom right corner (where you focus the filter field) to pick the field you want to use as filter. The filter is updated when that field is changed.

3.2.7.6 *Render a single line of text field using a custom render function [advanced]*

A single line of text field can be rendered using a custom function if you want to add your own customized rendering.

Start by adding the field to your form and then open the field properties. Check the **Use a custom render function** and add the name of the function – for example *myTxtFieldRenderFn*.

Open the Custom JS tab and add a function with the name you specified. This example will render a dropdown menu with two options. You can render any control you like as long as the value can be stored in a single line of text field.

```
function myTxtFieldRenderFn(f) {
  let options = [{ "val": "Option 1", "text": "Option one" }, { "val": "Option 2", "text": "Option two" }];
  let currFieldValue = getFieldValue(f.InternalName);
  let b = [];
  b.push("<select onchange='changeCustomDropdown(\"" + f.InternalName + "\", this)' style='padding: 6px 10px'>");
  b.push("<option value='>----</option>");
  options.forEach(opt => {
    b.push("<option value='" + opt.val + "'" + (currFieldValue === opt.val ? " selected=true" : "") + ">" + opt.text + "</option>");
  });
  b.push("</select>");
  return b.join("");
}

function changeCustomDropdown(fin, elm) {
  var value = elm.value;
  // Set the value to ensure it is saved with the form
  setFieldValue(fin, value);
}
```

3.3 Preview form

You can preview the form configuration from within the configurator. Some fields are rendered as read-only, and lookup columns rendered as treeview will have dummy content.

3.4 Rules

You can create rules to make your form dynamic. You can combine different triggers to create complex triggers by using a combination of and / or.

You can use the form fields as trigger using these operators:

- Is equal to
- Is not equal to
- Contains
- Does not contain
- Is greater than
- Is greater than or equal to
- Is less than
- Is less than or equal to
- Starts with
- Does not start with
- Ends with
- Does not end with

In addition to these triggers:

- Form is loaded
- Before save of the form
- After save of the form
- Using a mobile device
- SharePoint group
- Selected tab
- URL query string value
- Result from another rule

When you have set up the trigger you must select the actions. You can set up different actions based on the result of the rule – **If yes** and **If No**.

You can select among these actions:

- Required fields
- Optional fields
- Visible fields
- Hidden fields
- Editable fields
- Readonly fields
- Show elements

- Hide elements
- Call a function
- Set field value
- Send email
- Stop email
- Show / hide tabs
- Select tab
- Show a message box
- Show a field message
- Remove a field message

3.5 Custom JS

In this tab you can load external *.js files or add custom js directly to an editor.

You can find code examples in the forum here: <https://spjsblog.com/forums/topic/custom-js-examples/>

3.6 Custom CSS

You can add your own custom css style your custom HTML sections or to override any styling on your form.

You can load external CSS files by writing this in the Custom CSS field:

```
@import "/path_to_file/filename.css";
```

If you want to override any of the default classnames you must replace the auto-generated ID with the text SUFFIX. Example: change formLabel_a04d0572 to formLabel_SUFFIX. The reason for this is that the suffix number is automatically generated between each load of the page.

3.7 Miscellaneous

This tab contains various settings.

3.7.1 Password

You should set a password to protect your configuration from others.

3.7.2 Form panel type

Choose from the following list of form types:

- Full width
- Large
- Medium
- Small

3.7.3 Comments

Check the **Show comments button in the top right corner of display form and edit form** to let the users comment on the list items. This uses the out-of-the-box SharePoint comment functionality.

3.7.4 Tab colors

You can set the default colors for the tabsets you add to the form.

4 Modern DFFS WebPart

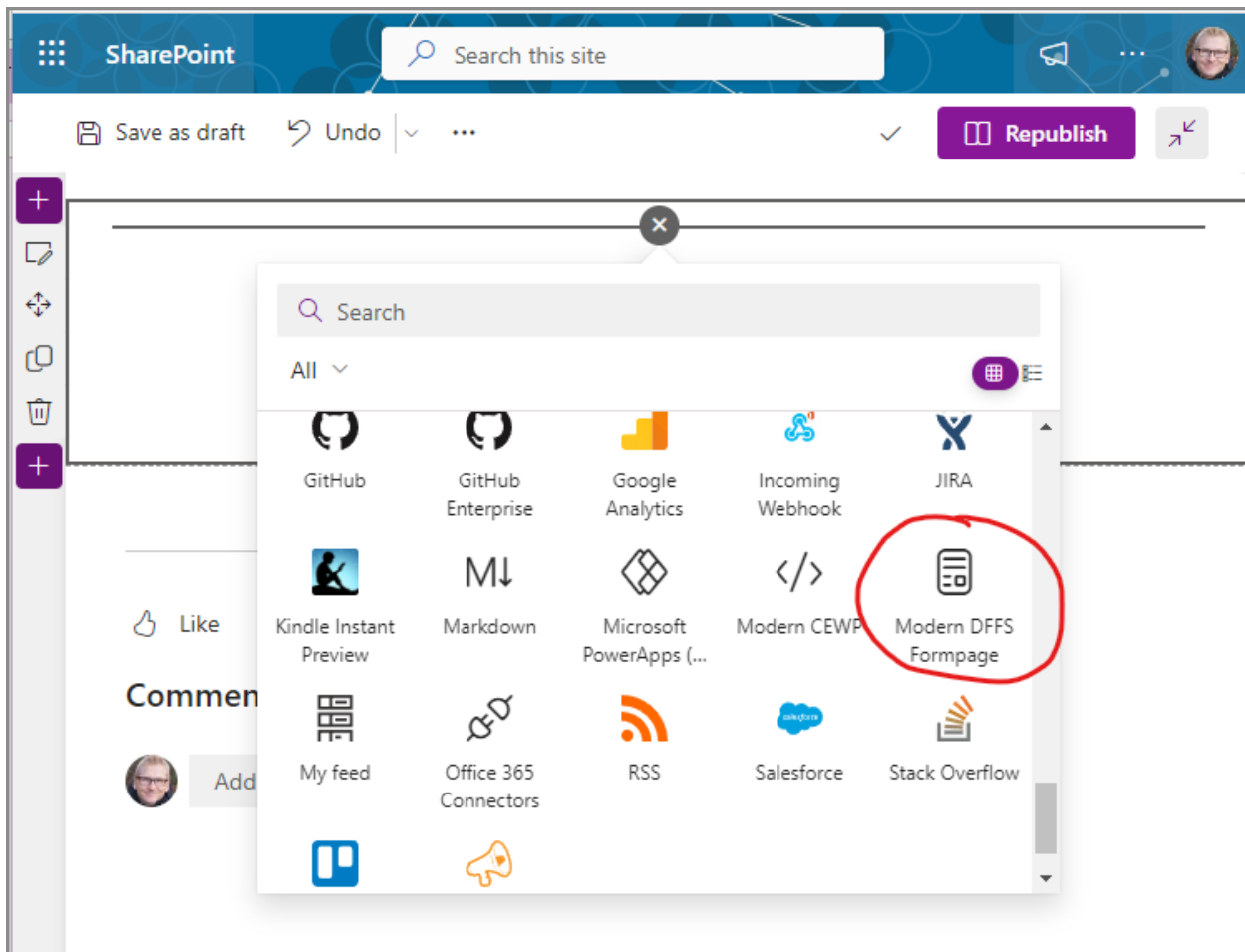
Requires Modern DFFS solution v1.0.14.0 or later.

This web part is used to show a Modern DFFS form in a page. This can be handy if you want to show a new item form, or you want to let users edit a list item by for example passing them a link to an item in an email.

You can use the Moder DFFS WebPart to add, view or edit items in a list that has Modern DFFS Forms configured. To use it you must pass the **DFFSList** (GUID of the target list), the **DFFSForm** (new, disp or edit) and the **DFFSID** (if you want to display or edit an existing item). You can also add a **Source** attribute to have the user redirected to that address after they cancel or save the form.

See instructions when adding the web part.

Add the web part like this:



When you have added it, you can read the instructions on how to use it in the placeholder:

Dynamic forms for SharePoint

This webpart is used to show a Modern DFFS form. You must create a link to this page with the following URL query string parameters

- DFFSList=[The GUID of the list]
- DFFSForm=[The form type - you can use new, disp or edit]
- DFFSID=[The ID of the list item to display]
- Source=[The URL to redirect to after save or cancel]

Example URL to create a new item in a list

```
https://spjsblog.sharepoint.com/sites/ModernDFFS/SitePages/DFFSForm.aspx?DFFSList={insert_list_guid_here}&DFFSForm=new&Source=/sites/ModernDFFS
```

<https://spjsworks.com>

It is currently not possible to use this formpage when adding, viewing or editing a list item from a list view – only by using a custom link created on the format described above.

4.1 Setting a default redirect in the form URL

When using the Modern DFFS Formpage webpart you can use an URL parameter to se a default redirect. This can still be overridden in Custom JS if you use the function `dffs_PostSaveAction` or `dffs_PostCancelAction`.

Create the URL like this

https://spjsblog.sharepoint.com/sites/ModernDFFS/SitePages/DFFSForm.aspx?DFFSList={insert_list_guid_here}&DFFSForm=new&DFFSSource=/The url to redirect to

5 Troubleshooting

5.1 Blocked web resources

If you are behind a company firewall you might have trouble loading some of the resources used in the Modern DFFS. For example, the Monaco-editor (the code editor used in Custom JS and Custom CSS) is loaded from cdn.jsdelivr.net. To use the code editor (and not have to edit the code as plain text) you must ensure that this CDN source is not blocked.

The license is loaded from an Azure CDN on this address: <https://spjs.blob.core.windows.net>. To be able to use DFFS you must also unblock this resource.

5.2 Other blocked resources

Because firewall rules are individually set up for different companies there might be other resources that must be unblocked in your company. If you have trouble loading the Modern DFFS you can use the developer tools to look for error messages in the Console or the Network tab.

If you have any questions please contact support: <https://spjsworks.com/support>