Contents

Manual:Extension/Forms

The **Forms** extension allows formatting a wiki page or a user dialog as a form. It is also used in conjunction with the Workflows extension where it provides input forms for workflow data.

Contents

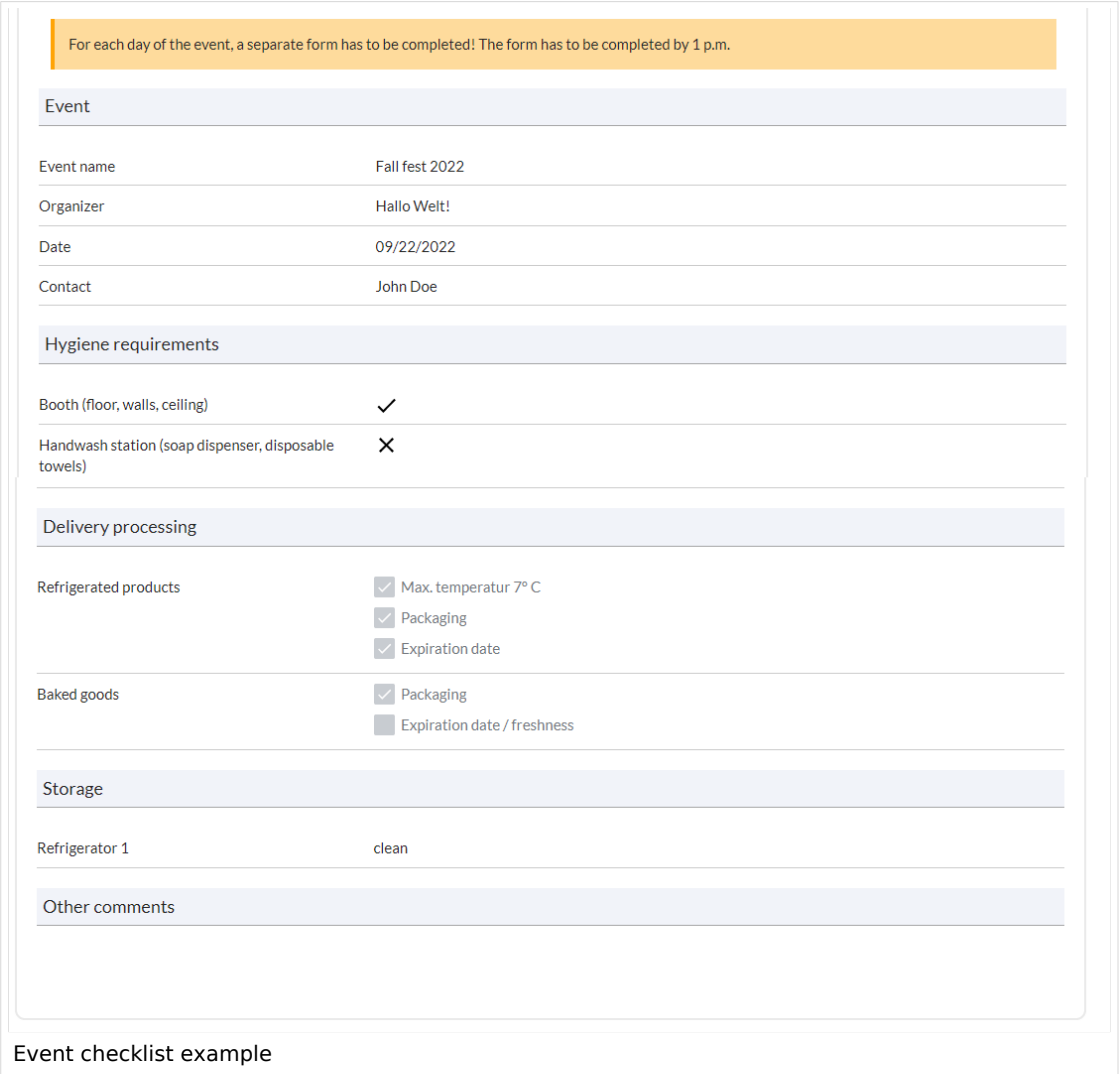
Introduction

To create a checklist as a form, for example, the following steps are necessary:

1. **Create and save the checklist form** via the *Special:Form_editor* special page. The form serves as a template for all checklists that are created with it.
2. **Create and save checklist instances** via the special page *Special:Create_form_instance*.

Optionally, the styling of the form can be adjusted.

As an example, we are creating a small checklist for an event where food-related hygiene information has to be tracked.

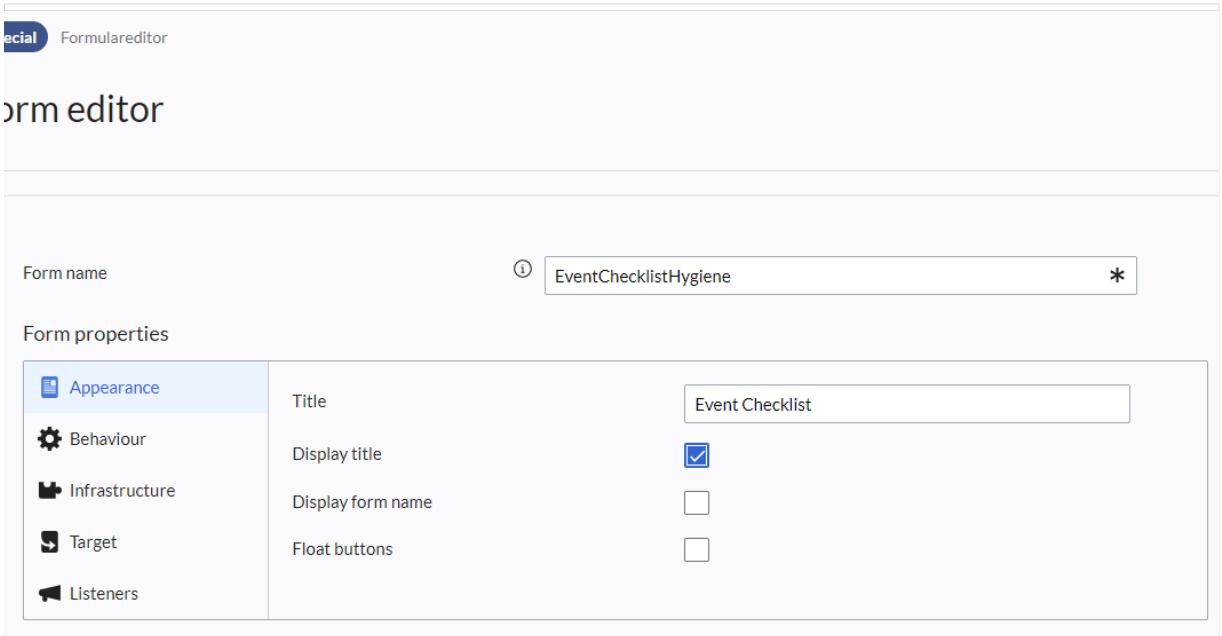


Event checklist example

Create a checklist form

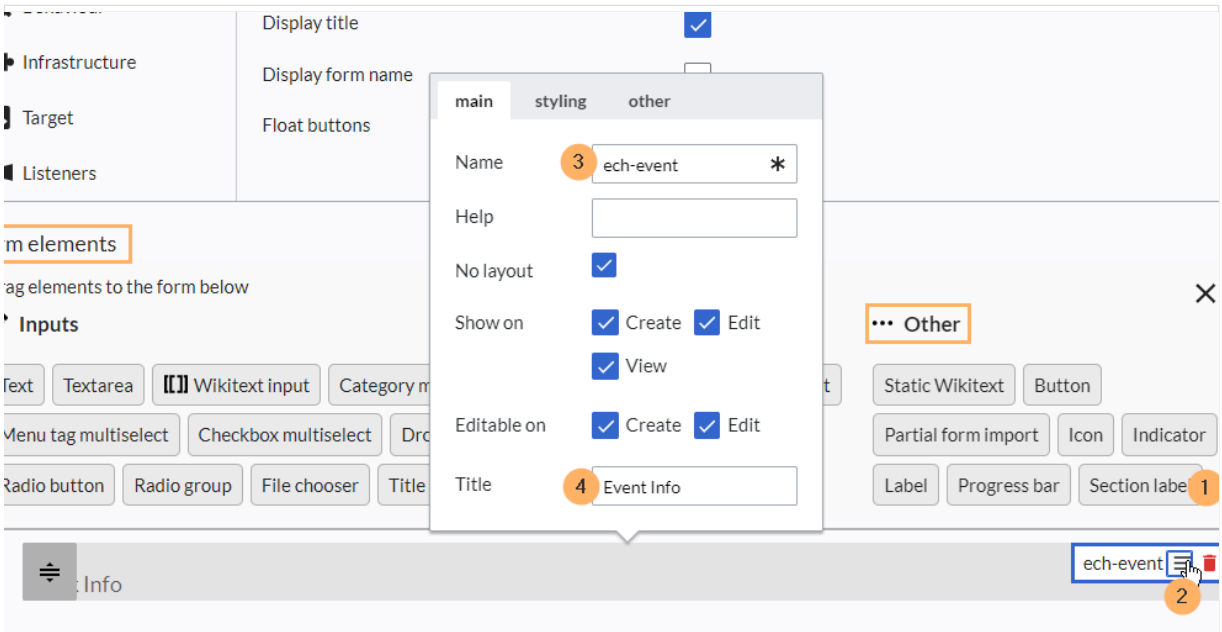
1. **Define** the main properties of the form:
   1. **Open** the *Special:Form editor* page.
   2. **Enter** a name for the new form, e.g. *EventChecklistHygiene*. This name is used as a unique identifier for the database. Therefore, do not use any special characters (apart from the hyphen "-").

3. **Enter** a title, e.g. Event Checklist. This is displayed as the display title instead of the form name if you activate the Display title checkbox.



Form properties

2. **Add** a form section:
   1. **Select** the *Section label* element under *Form Elements > Other* and, holding down the left mouse button, drag it into the interactive form editor field below.
   2. **Click** the section heading radio button. The editing dialog opens.
   3. **Enter** a field name. This is saved as a field ID in the database and should therefore not contain any special characters.
   4. **Enter** the title for the heading. This is used as the display title in the form.

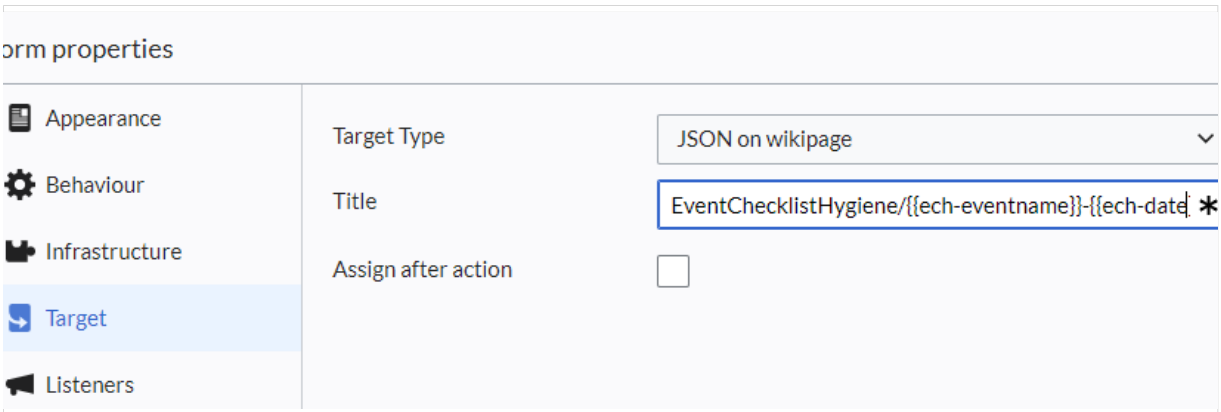


Create section label

3. **Create** the form fields for the Event section. Here we add a text box as an example.
   1. **Drag** the *Text* element under the section heading while holding down the left mouse button.

2. **Click** the text box radio button.
3. **Enter** a name. This is used as the database ID for this field and should therefore not contain any special characters.
4. **Enter** a label. This is displayed as the label for the text field.
5. **Add** more form elements.

4. Define which form fields should be used to generate the page name of the form instances created with this form. As an example, we define the page name for the checklists as subpages to the form name with the values from the event name and date fields (both fields must exist in the form: `EventChecklistHygiene/{{ech-eventname}}-{{ech-date}}`
   1. **Mark** *event name* and *day of event* as required so that these values always exist.
   2. **Switch** to the *Target* view in the form properties.
   3. **Select** "Json in a wiki page" as the target type.
   4. **Enter** your title syntax: `EventChecklistHygiene/{{ech-eventname}}-{{ech-date}}`



Define the page name syntax for form instances

5. **Click** Submit at the bottom of the page to save the form. The form is now stored in the main namespace as *EventChecklistHygiene.form*.

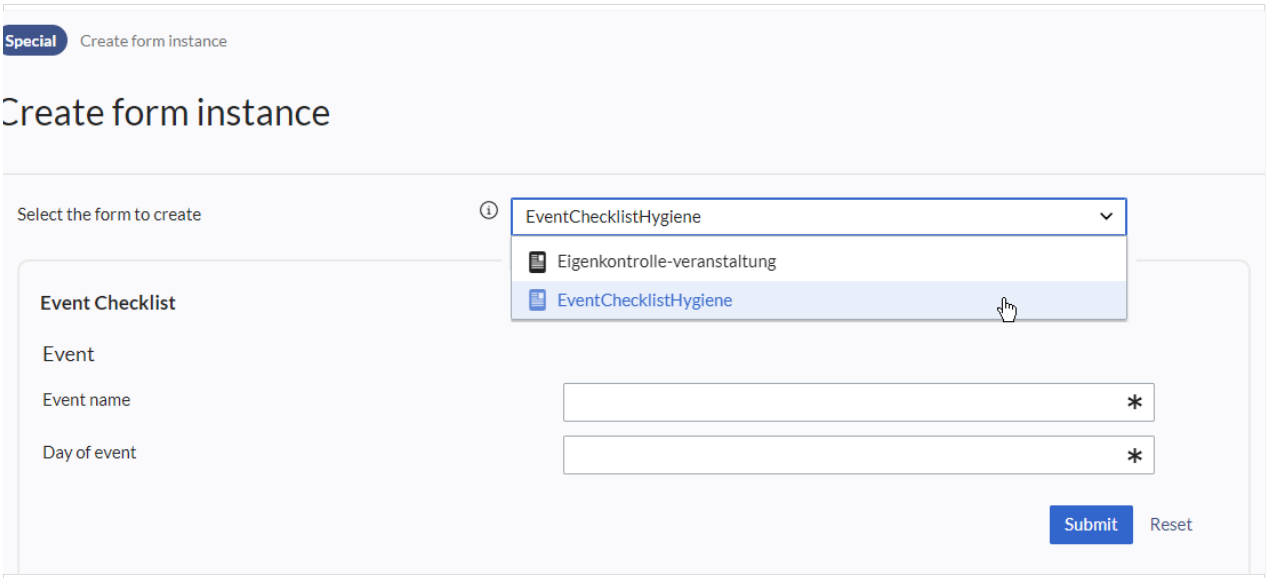
Saved form

To continue editing the form, switch to the editing mode of the page (1).

Create a checklist instance

To use the form, create your first checklist.

1. **Open** the page *Special:Create_form_instance*.
2. **Choose** the form *EventChecklistHygiene*.
3. **Fill out** the form fields.
4. **Click** *Submit*.

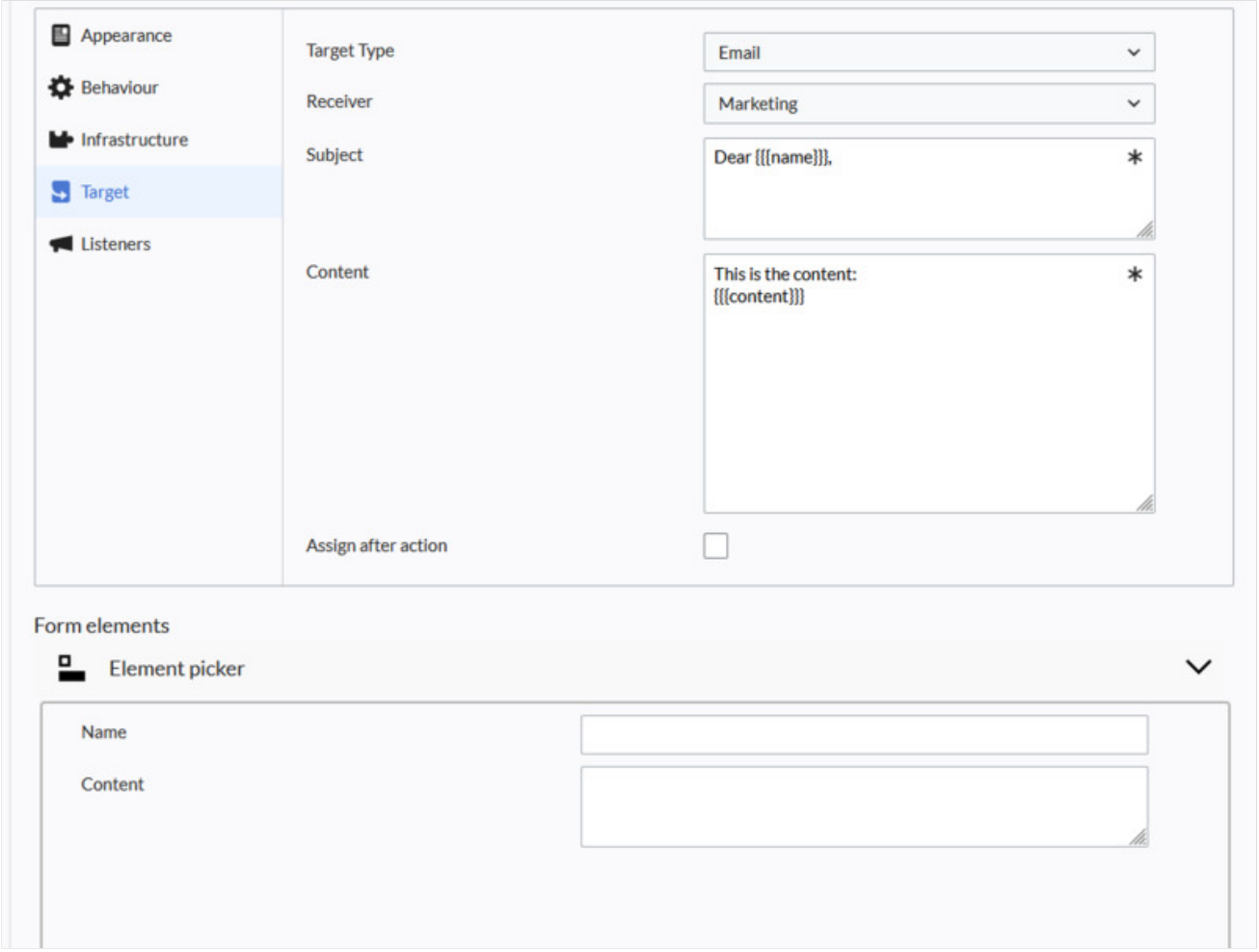


Create a checklist instance

The checklist is now saved in the wiki and can be edited further.

Send email

The form can also send an Email after being saved:



Email target

- **Target type:** Email
- **Recipient:** Possible recipients must have been configured on the server for security reasons. Example:

```
 $wgFormsTargetEMailRecipients = [
'Marketing' => "marketing@example.com",
'Administration' => 'WikiSysop'
];
```

- **Subject:** Existing form fields can be used as variables here. The value is taken from the form.
- **Content:** As in the subject, existing form fields can also be used as variables here. The value is taken from the form.

Localization messages in combination with form fields can also be used as subject or email body:

```
{{int:Mailsubject| {{{name}}}|{{{surname}}} }}
```

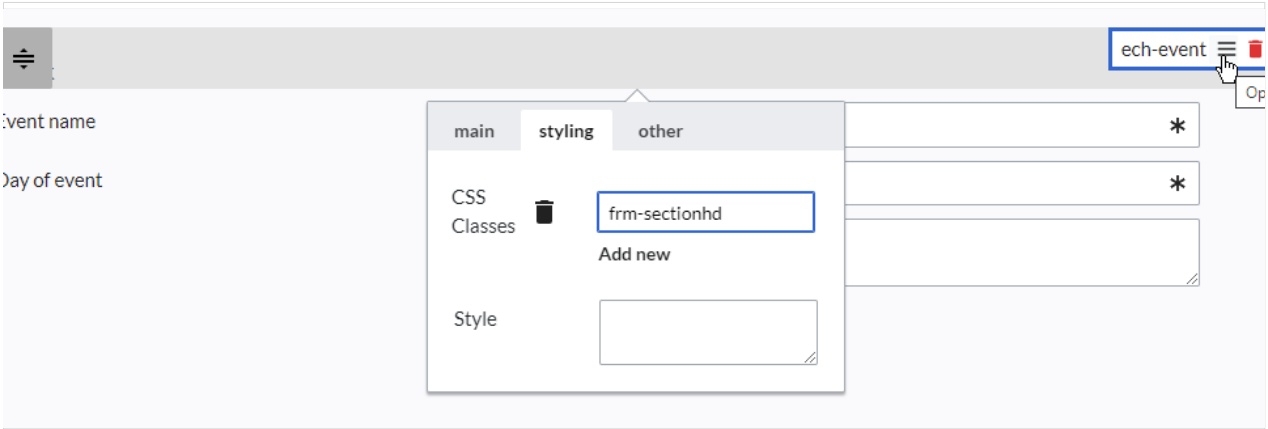
Adjust the styling

Common.css

To style individual elements in the form, go to the styling tab in the options dialog. There, you can enter a css class name. The styling is then defined on the *MediaWiki:Common.css* page (admin rights are required).

To style the section heading:

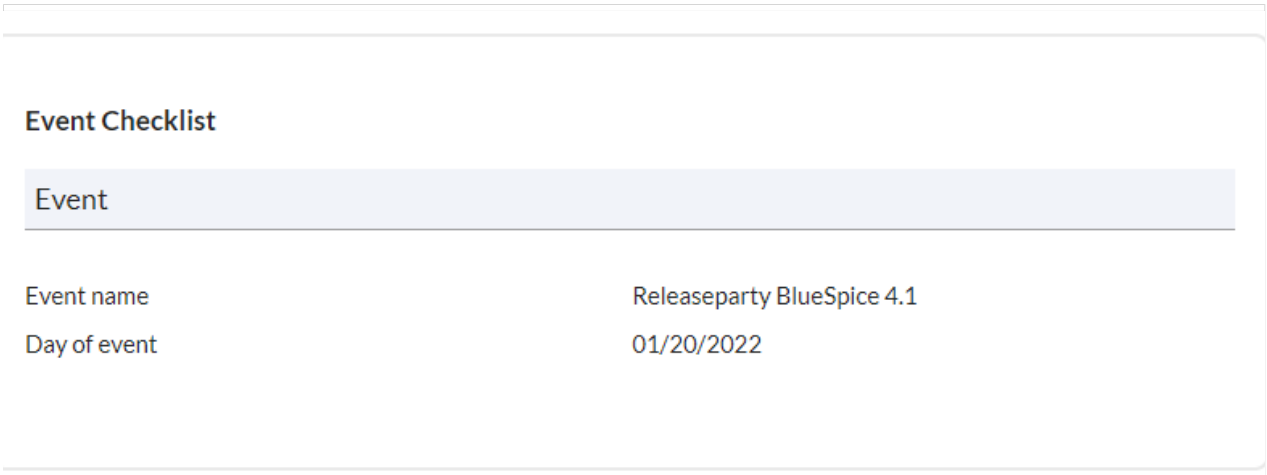
1. **Open** the styling tab in the options menu of the Event heading field.
2. **Add** a new CSS class and enter a selector name (*CSS Classes* field), e.g. *frm-sectionhd*.



Add a css class

3. **Paste** the following style declarations into the *MediaWiki:Common.css* page, e.g.:

```
border-bottom: 1px solid #a6a6a7;  background:#f1f3f9; padding:6px;margin:30px 0;
```



Section label with customized styling

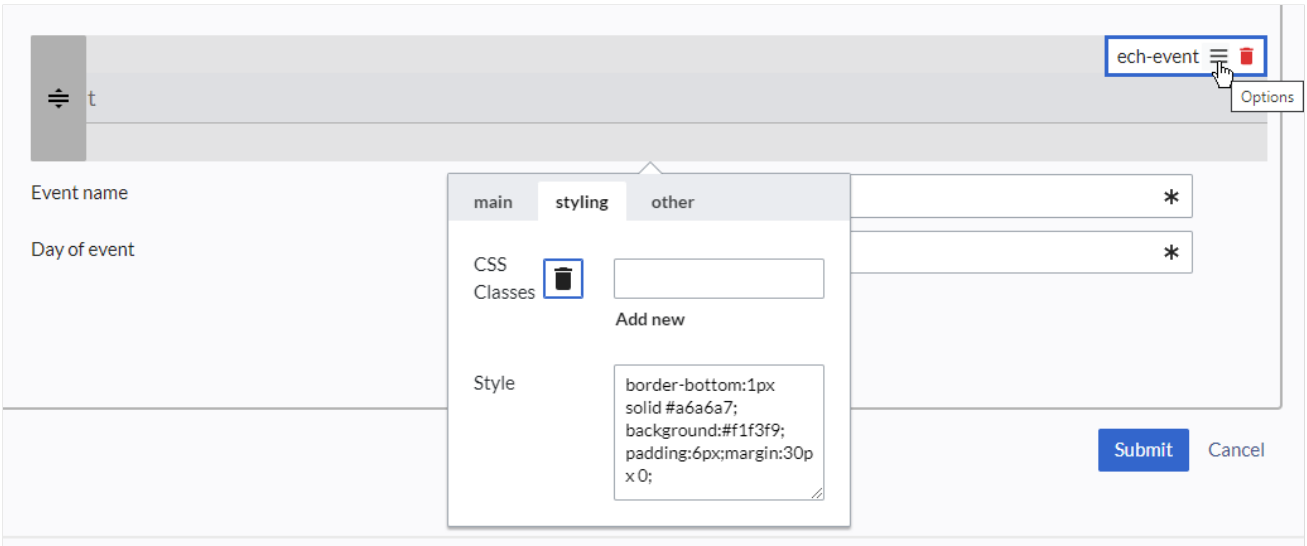
If you do not have permission to view this page, you can enter style information directly for each element in the form via inline styling.

Inline styling

Without access to *MediaWiki:Common.css*, the style information can be entered directly in the form. However, with many recurring style declarations this is not efficient in contrast to the definition of CSS classes in *MediaWiki:Common.css*.

As an example, let's apply the previous section heading example directly as inline styling. To do this, open the options menu of the *ech-event* form element again. Enter the following statement in the *Styling* field and save the change:

```
border-bottom: 1px solid #a6a6a7;  background:#f1f3f9; padding:6px;margin:30px 0;
```



Inline styling of a form element

Related info

- [https://www.mediawiki.org/wiki/HTMLForm Full documentation on mediawiki.org]

Technical Reference: Forms

Reference:Forms

Extension: Forms

all extensions

Overview			
Description:	Allround form framework for MediaWiki		
State:	stable	**Dependency:**	MediaWiki
Developer:	Hallo Welt!	**License:**	GPL-3.0-only
Type:	MediaWiki	**Category:**	Rich Articles
Edition:	BlueSpice pro, BlueSpice Farm, BlueSpice Cloud	**Version:**	4.1+
View user help page			

Features

This extension allows the formatting of a wiki page or a dialog window as a form. Besides being useful for creating form instances as wiki pages, this extension is also used in conjunction with the Workflows extension.

Full documentation on mediawiki.org

Technical Information

This information applies to BlueSpice 4. Technical details for BlueSpice Cloud can differ in some cases.

Requirements

MediaWiki: 1.39.0

Integrates into

- Forms

Special pages

- CreateFormInstance
- FormEditor

Permissions

Name	Description	Role
forms-create-form	Create forms	editor
forms-edit-form-definition	Edit form definitions	admin

Configuration

Name	Value
FormsTargetEMailRecipients	array ( )

API Modules

- forms-form-submit
- forms-get-definitions

Hooks

- BlueSpiceDiscoveryTemplateDataProviderAfterInit
- ChameleonSkinTemplateOutputPageBeforeExec
- CodeEditorGetPageLanguage
- ContentHandlerDefaultModelFor
- LoadExtensionSchemaUpdates
- ParserFirstCallInit
- SkinTemplateNavigation::Universal

Accessibility

Test status:	2-testing complete
Checked for:	Web, Authoring tool
Last test date:	2022-08-08
WCAG level:	AA
WCAG support:	does not support (workaround: no)

Comments:	Form editor: • drag-and-drop; no keyboard support Form instance edit: • missing field label associations Read instance: • missing field label associations, checkboxes are read multiple times.
Extension type:	extended
Extension focus:	editor

Reference:Workflows

Extension: Workflows

all extensions

Overview			
Description:	Allows to define and execute various workflows on the wiki		
State:	stable	**Dependency:**	MediaWiki
Developer:	Hallo Welt!	**License:**	GPL-3.0-only
Type:	MediaWiki	**Category:**	Quality Assurance
Edition:	BlueSpice pro, BlueSpice Farm, BlueSpice Cloud	**Version:**	4.1+
View user help page			

Features

Workflow functionality based on BPMN. BlueSpice included some standard workflows. Custom workflows can be created additionally.

Technical Information

This information applies to BlueSpice 4. Technical details for BlueSpice Cloud can differ in some cases.

Requirements

MediaWiki: 1.39.0
Forms: *

- OOJSPlus: *

Integrates into

- BlueSpiceDiscovery
- UnifiedTaskOverview
- Workflows

Special pages

- WorkflowsOverview

Permissions

Name	Description	Role
workflows-admin	Manage workflows	admin
workflows-execute	Initiate and advance workflows	admin, reviewer, editor
workflows-view	View workflow items	reader

User options

Name	Value
echo-subscriptions-email-workflow-cat	1

Hooks

- ArticleDeleteComplete
- BeforePageDisplay
- BlueSpiceDiscoveryTemplateDataProviderAfterInit

- CodeEditorGetPageLanguage
- ContentHandlerDefaultModelFor
- LoadExtensionSchemaUpdates
- MWStakeCommonUIRegisterSkinSlotComponents
- MWStakeRunJobsTriggerRegisterHandlers
- PageSaveComplete
- SetupAfterCache
- SkinTemplateNavigation::Universal
- UnifiedTaskOverviewGetTaskDescriptors