Creatio Academy

Implement a custom component

Custom UI component based on a classic Creatio page element

Version 8.0

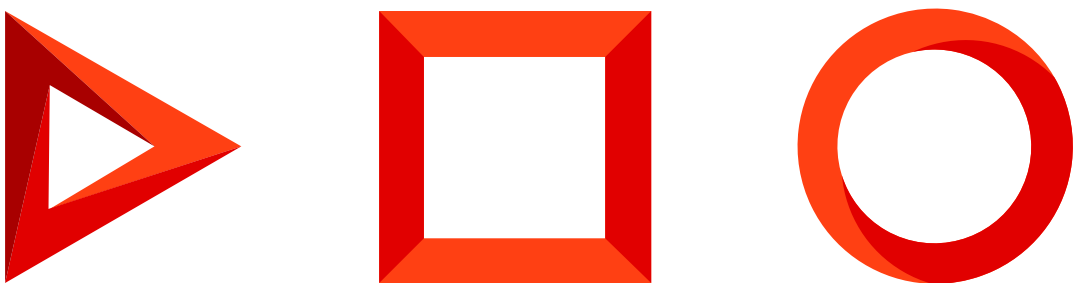


Table of Contents

Custom UI component based on a classic Creatio page element

Beginner

In Creatio, you can expand the out-of-the-box set of Freedom UI page components with custom components.

Creatio lets you implement the following custom component **types**:

- Custom component based on a classic Creatio page element. Supported in Creatio 8.0.2 Atlas and later.
- Remote module. Supported in Creatio 8.0.3 Atlas and later.

You can implement a custom component based on a classic Creatio page element in Creatio version 8.0.2 Atlas and later.

To **implement a custom component based on a classic Creatio page element**:

1. Create a custom component.
2. Add the custom component to the Freedom UI page.

1. Create a custom component

1. Create a module schema. To do this, follow the instruction in a separate article: Client module.
2. Implement a **custom component**.

   a. Add a dependency on the `@creatio-devkit/common` library to the AMD module.

   Example that adds a dependency to the `UsrAppClientSchemaName` AMD module

   ```
   /* AMD module declaration. */
   define("UsrAppClientSchemaName", ["@creatio-devkit/common"], function (sdk) {
       ...
   });
   ```

   b. Declare the component class.

   Example that declares the `UsrAppClientSchemaName` class

   ```
   /* AMD module declaration. */
   define("UsrAppClientSchemaName", ["@creatio-devkit/common"], (sdk) {
       /* Declare the class. */
       class UsrAppClientSchemaName extends HTMLElement {
   ```

```
        constructor() {
            super();
            const shadowDom = this.attachShadow({mode: 'open'});
            shadowDom.innerHTML = '<h1>UsrAppClientSchemaName works!</h1>'
        }
    }
    ...
});
```

c. Register the component.

Example that registers the `UsrAppClientSchemaName` component

```
/* AMD module declaration. */
define("UsrAppClientSchemaName", ["@creatio-devkit/common"], (sdk) {
    ...
    /* Register the component. */
    customElements.define('usr-custom-view-element', UsrAppClientSchemaName);
    ...
});
```

d. Register the web component as a view element.

Example that registers the `usr-custom-view-element` component

```
/* AMD module declaration. */
define("UsrAppClientSchemaName", ["@creatio-devkit/common"], (sdk) {
    ...
    /* Register the web component as a view element. */
    sdk.registerViewElement({
        type: 'usr.CustomViewElement',
        selector: 'usr-custom-view-element'
    });
});
```

2. Add the custom component to the Freedom UI page

1. Add the custom component module to the AMD module declaration as a dependency.

Example that adds the `UsrAppClientSchemaName` dependency

```
/* AMD module declaration. */
define("UsrAppClientSchemaName1", ["UsrAppClientSchemaName"], () {
    ...
```

```
});
```

2. Add the configuration object of the module that contains the custom component to the `viewConfigDiff` schema section.

   View the configuration object of the module that contains the `UsrCustomViewElement` custom component below.

`viewConfigDiff` schema section

```
viewConfigDiff: /**SCHEMA_VIEW_CONFIG_DIFF*/[
    {
        "operation": "insert",
        "name": "UsrAppClientSchemaName",
        "values": {
            "type": "usr.CustomViewElement",
            "layoutConfig": {
                "column": 1,
                "row": 1,
                "colSpan": 3,
                "rowSpan": 1
            }
        },
        "parentName": "Main",
        "propertyName": "items",
        "index": 0
    }
]/**SCHEMA_VIEW_CONFIG_DIFF*/,
```

Note. It is not possible to add a custom component based on a classic Creatio page element to the component library of the Freedom UI Designer. The Freedom UI Designer displays a placeholder instead of the custom component on the canvas. This is because classic Creatio page elements are implemented using the ExtJs framework. To display a custom component in the element library of the Freedom UI Designer, implement a remote module. To do this, follow the instruction in a separate article: Implement a remote module.

View a detailed example that implements the custom component in a separate article: Implement a custom component based on a classic Creatio page.

Implement a custom component based on a classic Creatio page

Medium

The example is relevant to version 8.0.2 and later.

Example. Display the history of the selected account on the [*Account timeline*] tab of the record page of the [*Requests*] custom section. The tab is a web component. Implement the web component based on the 7.X [*Timeline*] tab displayed on the contact page.

1. Create an app

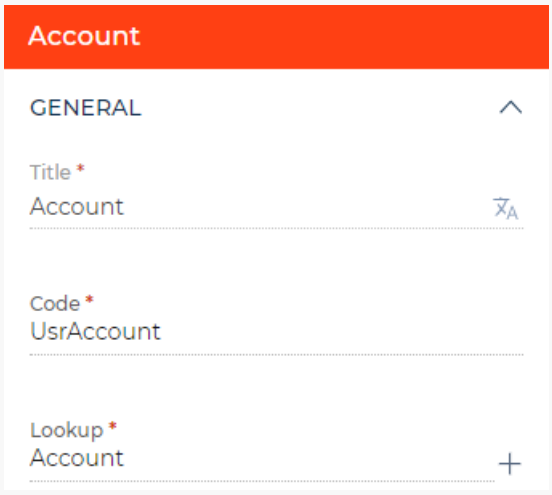
1. Create a custom `Requests` app based on the [*Records & business processes*] template. To do this, follow the procedure in the user documentation: Create a custom app.
2. Open the [*Requests form page*] page in the workspace of the `Requests` app page.

   The [*Requests form page*] page includes the [*Name*] field by default.
3. Add an **account field**:

   a. Add a new field of the [*Dropdown*] type to the Freedom UI Designer's workspace.

   b. Click the button on the Freedom UI Designer's action panel and fill out the **field properties** in the setup area:

   - Set [*Title*] to "`Account`."
   - Set [*Code*] to "`UsrAccount`."
   - Select "`Account`" in the [*Lookup*] property.
   - Clear the [*Enable adding new values*] checkbox.



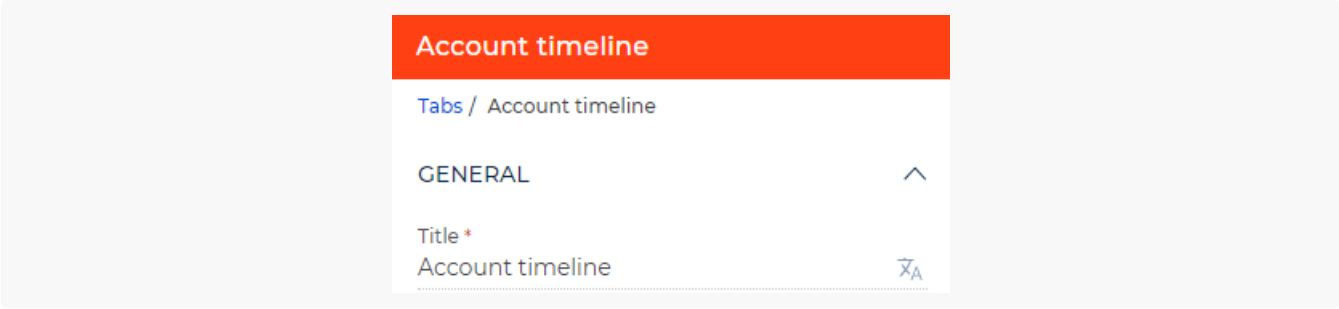
4. Add a **tab that contains the history of the selected account**.

   a. Add a new [*Tabs*] layout element to the Freedom UI Designer's workspace.

   b. Delete the [*Tab 2*] tab the [*Requests form page*] page includes by default.

   c. Click the button on the Freedom UI Designer's action panel and specify "`Account timeline`" in the

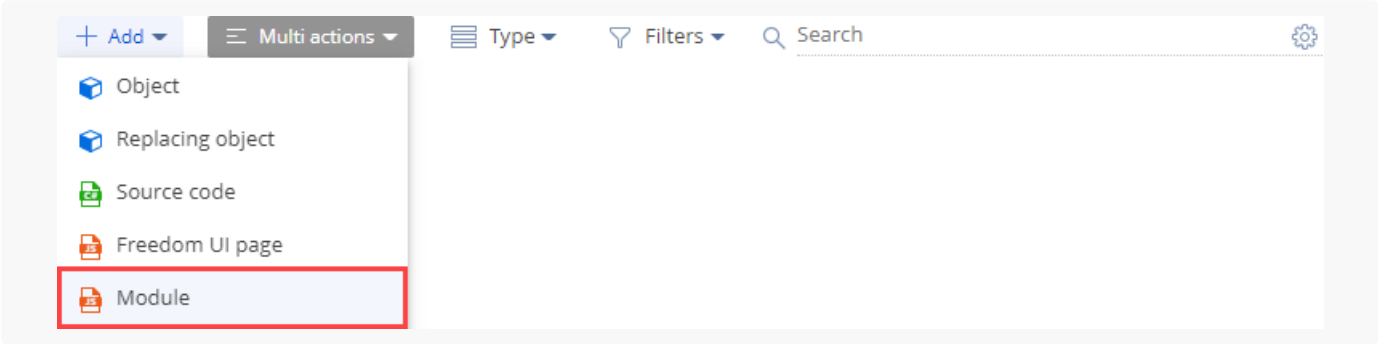
[*Title*] **tab property** in the setup area.



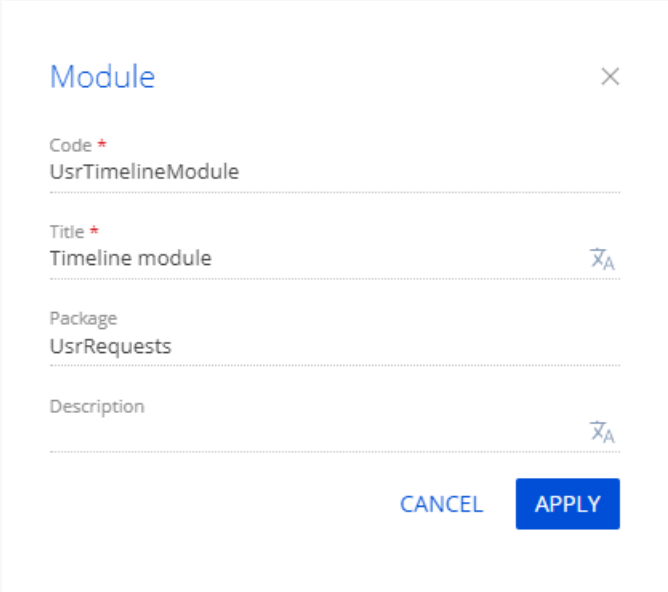
5. Click [*Save*] on the Freedom UI Designer's action panel.

2. Create a custom web component

1. Go to the [*Configuration*] section and select a custom package to add the schema.
2. Click [*Add*] → [*Module*] on the section list toolbar.



3. Fill out the **schema properties** in the Module Designer.
   - Set [*Code*] to "`UsrTimelineModule`."
   - Set [*Title*] to "`Timeline module`."



4. Implement a **custom web component**.

   a. Add the `@creatio-devkit/common`, `Base7xViewElement`, and `ckeditor-base` modules as dependencies to the AMD module declaration.

   Dependencies of the `UsrTimelineModule` AMD module

   ```
   /* AMD module declaration. */
   define("UsrTimelineModule", ["@creatio-devkit/common", "Base7xViewElement", "ckeditor-base"
       ...
   });
   ```

   b. Declare a custom `UsrTimelineModule` web component class.

   `UsrTimelineModule` class declaration

   ```
   /* AMD module declaration. */
   define("UsrTimelineModule", ["@creatio-devkit/common", "Base7xViewElement", "ckeditor-base"
       /* Class declaration. */
       class UsrTimelineModule extends Base7xViewElement {
           set primaryColumnValue(value) {
               this._primaryColumnValue = value;
               this._init();
           }

           get primaryColumnValue() {
               return this._primaryColumnValue;
           }

           set entitySchemaName(value) {
               this._entitySchemaName = value;
   ```

```
            this._init();
        }

        get entitySchemaName() {
            return this._entitySchemaName;
        }

        set cardSchemaName(value) {
            this._cardSchemaName = value;
            this._init();
        }

        get cardSchemaName() {
            return this._entitySchemaName;
        }

        constructor() {
            super("Timeline");
        }

        _init() {
            if (this._primaryColumnValue && this._cardSchemaName && this._entitySchemaName)
                this.initContext(() => {
                    this._moduleId = this.sandbox.id + "_UsrTimelineModule";
                    this.sandbox.subscribe("GetColumnsValues", (attributeNames) => this._ge
                    this.sandbox.subscribe("GetEntityInfo", () => this._getEntityInfo(), nu
                    this._loadTimelineSchemaModule();
                });
            }
        }

        _loadTimelineSchemaModule() {
            this._moduleId = this.sandbox.loadModule("BaseSchemaModuleV2", {
                id: this._moduleId,
                renderTo: this._renderTo,
                instanceConfig: {
                    schemaName: "TimelineSchema",
                    isSchemaConfigInitialized: true,
                    useHistoryState: false,
                    showMask: true,
                    parameters: {
                        viewModelConfig: {
                            "CardSchemaName": this._cardSchemaName,
                            "ReferenceSchemaName": this._entitySchemaName,
                            "InitialConfig": {
                                "entities": []
                            }
                        },
```

```
                    },
                }
            });
        }

        _getColumnValues(attributeNames) {
            const values = {};
            attributeNames?.forEach((attributeName) => {
                switch (attributeName) {
                    case "Id":
                        values[attributeName] = this._primaryColumnValue?.value;
                        break;
                    case "Name":
                        values[attributeName] = this._primaryColumnValue?.displayValue;
                        break;
                    default: break;
                }
            });
            return values;
        }

        _getEntityInfo() {
            return {
                entitySchemaName: this._entitySchemaName,
                primaryColumnValue: this._primaryColumnValue?.value,
                primaryDisplayColumnValue: this._primaryColumnValue?.displayValue
            };
        }

        getMessages() {
            const messages = super.getMessages();
            return Object.assign(messages, {
                "GetColumnsValues": {
                    mode: Terrasoft.MessageMode.PTP,
                    direction: Terrasoft.MessageDirectionType.SUBSCRIBE
                },
                "GetEntityInfo": {
                    mode: Terrasoft.MessageMode.PTP,
                    direction: Terrasoft.MessageDirectionType.SUBSCRIBE
                }
            });
        }

        disconnectedCallback() {
            this.sandbox.unloadModule(this._moduleId, this._renderTo);
        }
    }
    ...
});
```

c. Register the `UsrTimelineModule` web component on the page.

Register the `UsrTimelineModule` web component

```
/* AMD module declaration. */
define("UsrTimelineModule", ["@creatio-devkit/common", "Base7xViewElement", "ckeditor-base'
    ...
    /* Web component registration. */
    customElements.define('usr-timeline', UsrTimelineModule);
    ...
});
```

d. Register the `usr-timeline` web component as a visual element.

`usr-timeline` web component registration

```
/* AMD module declaration. */
define("UsrTimelineModule", ["@creatio-devkit/common", "Base7xViewElement", "ckeditor-base'
    ...
    /* Register a web component as a visual element. */
    sdk.registerViewElement({
        type: 'usr.Timeline',
        selector: 'usr-timeline',
        inputs: {
            primaryColumnValue: {},
            cardSchemaName: {},
            entitySchemaName: {}
        }
    });
});
```

Complete source code of the page schema

5. Click [*Save*] on the Module Designer's toolbar.

3. Add the custom web component to the Freedom UI page

1. Open the `UsrRequests_FormPage` schema of the Freedom UI [*Requests form page*] page in the [*Configuration*] section.
2. Click the </> button on the Freedom UI Designer's action panel. This opens the source code of the Freedom UI page.
3. Add the **custom web component**.

a. Add the `UsrTimelineModule` module of the custom web component as a dependency to the AMD module declaration.

Dependencies of the `UsrRequests_FormPage` AMD module

```
/* AMD module declaration. */
define("UsrRequests_FormPage", /**SCHEMA_DEPS*/["UsrTimelineModule"]/**SCHEMA_DEPS*/, funct
    ...
});
```

b. Add the configuration object of the `UsrTimelineModule` module with the custom web component to the `viewConfigDiff` schema section.

`viewConfigDiff` schema section

```
viewConfigDiff: /**SCHEMA_VIEW_CONFIG_DIFF*/[
    ...,
    {
        "operation": "insert",
        "name": "Timeline_qwerty",
        "values": {
            "type": "usr.Timeline",
            "layoutConfig": {
                "column": 1,
                "row": 1,
                "colSpan": 12,
                "rowSpan": 8
            },
            "primaryColumnValue": "$UsrAccount",
            "cardSchemaName": "AccountPageV2",
            "entitySchemaName": "Account"
        },
        "parentName": "GridContainer_qaocexw",
        "propertyName": "items",
        "index": 0
    },
    ...
]/**SCHEMA_VIEW_CONFIG_DIFF*/,
```

Complete source code of the page schema

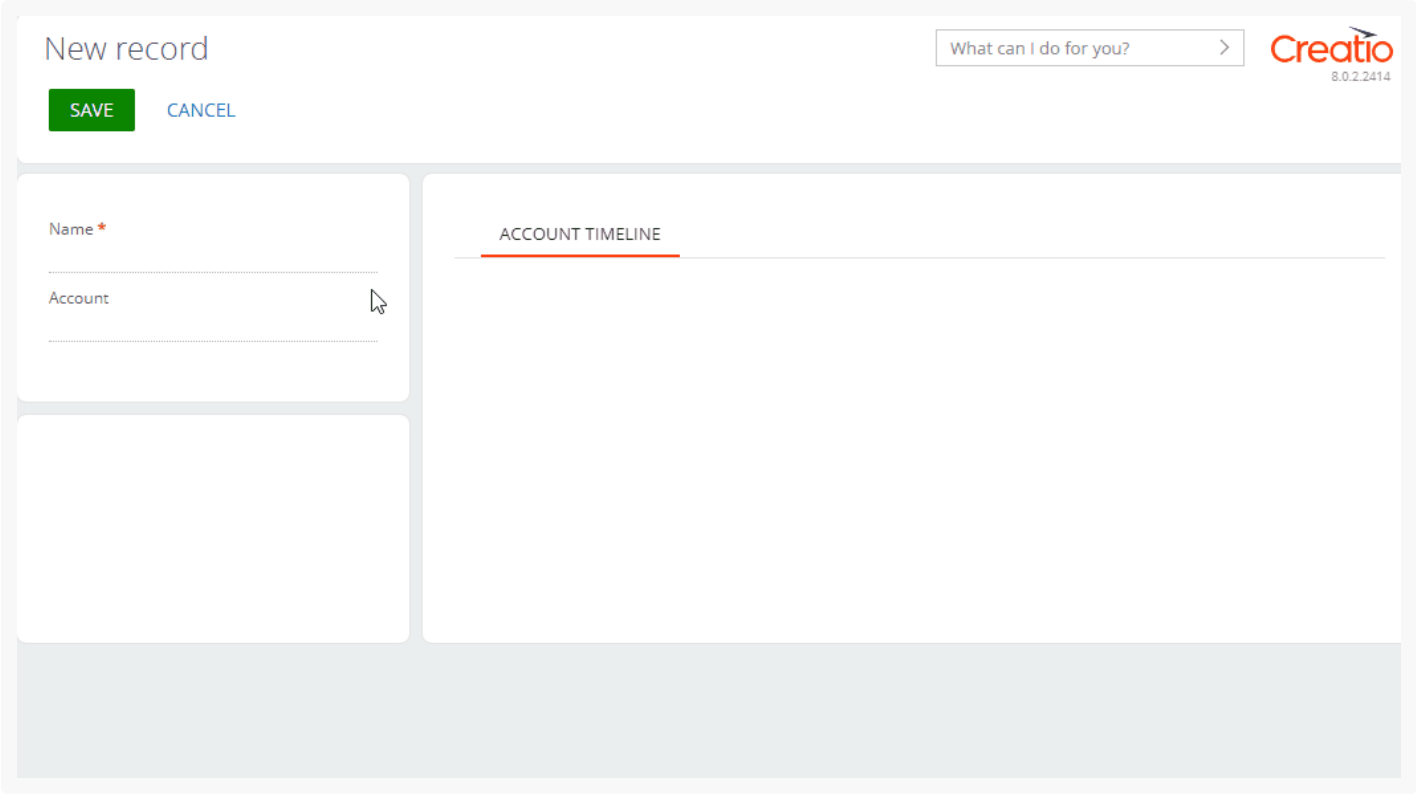
4. Click [*Save*] on the Client Module Designer's toolbar.

Outcome of the example

To **view the outcome of the example**:

1. Open the `Requests` app page and click [*Run app*]
2. Click [*New*] on the `Requests` app toolbar.
3. Enter "`Request's name`" in the [*Name*] field.
4. Select an account in the [*Account*] field, for example, "`Accom`."

As a result, Creatio will display the [*Account timeline*] tab of the selected "`Accom`" account on the request page.



Freedom UI Designer will display a stub in place of the custom web component.