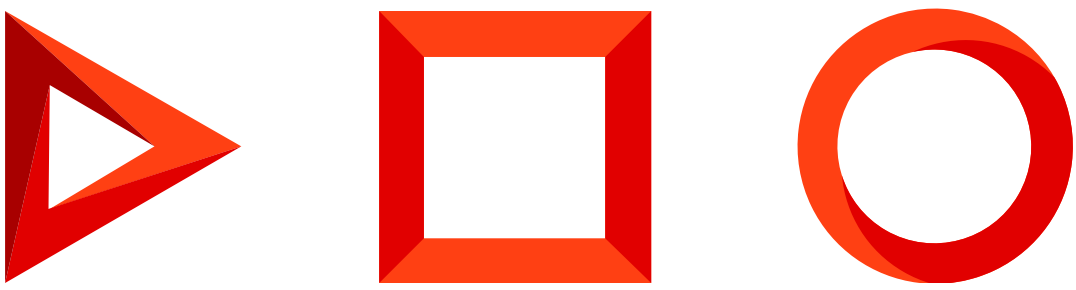
Creatio Academy

Freedom UI page customization

Hide functionality on a page

Version 8.0



This documentation is provided under restrictions on use and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this documentation, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

© 2023 Creatio. All rights reserved.

Table of Contents

© 2023 Creatio. All rights reserved.

Hide functionality on a page

Beginner

Creatio 8 Atlas lets you execute the following **actions** that hide functionality on a page:

- Hide functionality during development.
- Hide functionality due to insufficient access permissions.

Hide functionality during development

Creatio 8 Atlas uses the `sdk.FeatureService` service to check functionality status.

To **hide functionality during development on a page** :

1. Add a page component that contains the functionality during development at step 1 of the Freedom UI page customization procedure if needed.
2. Set up **how to hide the functionality on the page** at step 2 of the Freedom UI page customization procedure.

   a. Enable the `sdk.FeatureService` service that checks the functionality status. Enable the service similarly to the display procedure for the value of system variables.

   b. Add an attribute that stores data to the `viewModelConfig` schema section. Add the attribute similarly to the setup procedure for the field display condition.

   c. Bind the `visible` property to the corresponding model attribute in the `viewConfigDiff` schema section. Property binding is similar to that described in the setup procedure for the field display condition.

   d. Add a custom implementation of the `crt.HandlerViewModelInitRequest` system query handler to the `handlers` schema section. The handler is executed when the `View` model is initialized.

      a. Instantiate the service that checks the functionality status from `@creatio/sdk`.

      b. Get the status of the functionality by its code and write it to the corresponding attribute.

      View an example of a `crt.HandlerViewModelInitRequest` query handler that receives a feature status with the `SomeFeatureCode` code and writes it to the `SomeAttributeName` attribute below.

      `handlers` schema section

View a detailed example that hides functionality during development in a separate article: Hide a feature at the development stage on a page.

Hide functionality due to insufficient access permissions

Creatio 8 Atlas uses the `sdk.RightsService` service to check access permissions.

To **hide functionality due to insufficient access permissions**:

© 2023 Creatio. All rights reserved.

1. Add a page component with the functionality for which access permissions are configured at step 1 of the Freedom UI page customization procedure if needed.
2. Set up **how to hide functionality on the page due to insufficient access permissions** at step 2 of the Freedom UI page customization procedure.

   a. Enable the `sdk.RightsService` service that checks access permissions. Enable the service similarly to the display procedure for the value of system variables.

   b. Add an attribute that stores data to the `viewModelConfig` schema section. Add the attribute similarly to the setup procedure for the field display condition.

   c. Bind the `visible` property to the corresponding model attribute in the `viewConfigDiff` schema section. Property binding is similar to that described in the setup procedure for the field display condition.

   d. Add a custom implementation of the `crt.HandlerViewModelInitRequest` system query handler to the `handlers` schema section. The handler is executed when the `View` model is initialized.

      a. Instantiate the service that checks access permissions from `@creatio/sdk`.

      b. Get information about the user's permissions to perform the corresponding action.

      c. Write the result of the checkup to the corresponding attribute.

      View an example of the `crt.HandlerViewModelInitRequest` request handler that checks for permissions to perform a system operation with the `SomeOperationCode` code and writes the result to the `SomeAttributeName` attribute below.

      `handlers` schema section

View a detailed example that hides functionality due to insufficient access permissions in a separate article: Hide the feature on a page due to insufficient access permissions.

Hide a feature at the development stage on a page

Medium

Example. Hide a custom [*Feature*] button on a record page of a custom [*Feature Service*] section. The button contains the feature at the development stage.

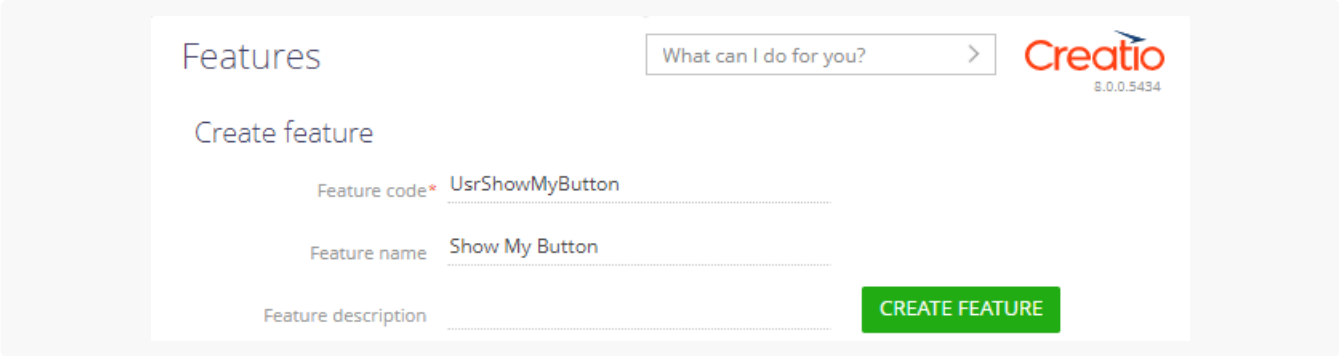
1. Set up the page UI

1. Add the **developed feature to hide**.

   a. Open the [*Feature*] page and fill out the **feature properties**:

   - Set [*Feature code*] to "`UsrShowMyButton`."
   - Set [*Feature name*] to "`Show My Button`."

© 2023 Creatio. All rights reserved.



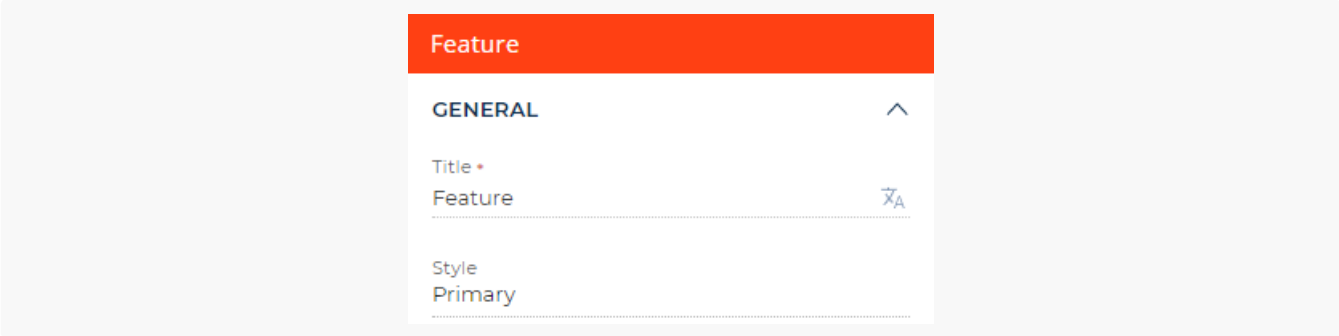
d. Click [*Create feature*].

2. Create a custom `Feature Service` app based on the [*Records & business processes*] template. To do this, follow the guide in the user documentation: Create a custom app.
3. Open the [*Feature service form page*] page in the working area of the `Feature Service` app page.
4. Delete the [*Name*] field the [*Feature Service form page*] page includes by default.
5. Add a **button that contains the feature at the development stage**.

   a. Add a [*Button*] type component to the toolbar of the Freedom UI Designer.

   b. Click in the action panel of the Freedom UI Designer and fill out the **button properties** in the setup area.

   - Set [*Title*] to "`Feature`."
   - Select "`Primary`" in the [*Style*] property.



6. Click in the action panel of the Freedom UI Designer. After you save the page settings, Creatio opens the source code of the Freedom UI page.

2. Hide the feature at the development stage

Configure the business logic in the Client Module Designer. For this example, hide the feature at the development stage.

1. Enable the `sdk.FeatureService` service that checks the feature status. To do this, add `@creatio-devkit/common` to the AMD module as a dependency.

   AMD module dependencies

© 2023 Creatio. All rights reserved.

```
/* Declare the AMD module. */
define("UsrAppFeatureService_FormPage", /**SCHEMA_DEPS*/["@creatio-devkit/common"] /**SCHEMA_
        ...
    };
});
```

2. Add the `ShowMyButton` attribute that stores feature status data to the `viewModelConfig` schema section.

`viewModelConfig` schema section

```
viewModelConfig: /**SCHEMA_VIEW_MODEL_CONFIG*/{
    "attributes": {
        ...,
        /* The attribute that stores the feature status. */
        "ShowMyButton": {}
    }
}/**SCHEMA_VIEW_MODEL_CONFIG*/,
```

3. Bind the `visible` property of the `FeatureButton` element to the `ShowMyButton` model attribute in the `viewConfigDiff` schema section.

`viewConfigDiff` schema section

```
viewConfigDiff: /**SCHEMA_VIEW_CONFIG_DIFF*/[
    {
        "operation": "insert",
        "name": "FeatureButton",
        "values": {
            ...,
            /* The property that flags the field as visible. Bound to the ShowMyButton attrib
            "visible": "$ShowMyButton"
        },
        ...
    }
]/**SCHEMA_VIEW_CONFIG_DIFF*/,
```

4. Add a custom implementation of the `crt.HandlerViewModelInitRequest` system query handler to the `handlers` schema section. Execute the handler when Creatio initializes the `View` model.

   a. Create an instance of the service that checks the feature status from `@creatio-devkit/common`.

   b. Retrieve the status of the feature that has the `UsrShowMyButton` code and specify the status in the `ShowMyButton` attribute.

`handlers` schema section

© 2023 Creatio. All rights reserved.

```
handlers: /**SCHEMA_HANDLERS*/[
    {
        request: "crt.HandleViewModelInitRequest",
        /* The custom implementation of the system query handler. */
        handler: async (request, next) => {
            /* Create an instance of the service that checks the feature status from @creatio
            const featureService = new sdk.FeatureService();
            /* Retrieve the UsrShowMyButton feature status and specify it in the ShowMyButton
            request.$context.ShowMyButton = await featureService.getFeatureState('UsrShowMyBu
            /* Call the next handler if it exists and return its result. */
            return next?.handle(request);
        }
    }
] /**SCHEMA_HANDLERS*/,
```

Complete source code of the page schema

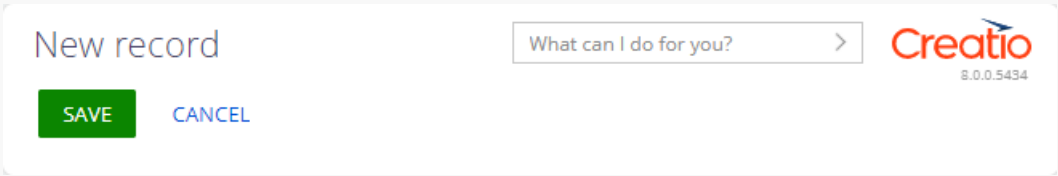
5. Click [*Save*] on the Client Module Designer's toolbar.

Outcome of the example

To **view the outcome of the example for the feature at the development stage**:

1. Open the `Feature Service` app page and click [*Run app*].
2. Click [*New*] on the `Feature Service` app toolbar.

As a result, Creatio will hide the [*Feature*] button that contains the feature at the development stage on the `Feature Service` app page.

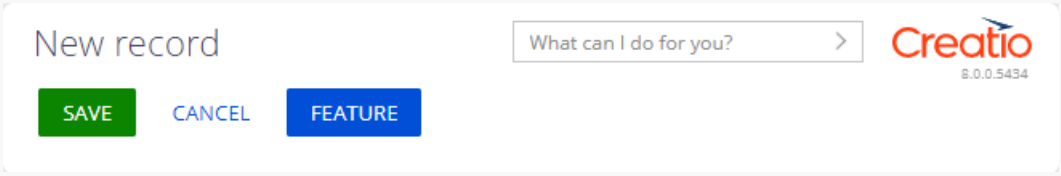


To **view the outcome of the example for the fully developed feature**:

1. Enable the feature that has `UsrShowMyButton` code.
   a. Open the [*Feature*] page.
   b. Enable the `Show My Button` feature in the page list.
   c. Click [*Save changes*] and refresh the page.
2. Refresh the `Feature Service` app page.
3. Click [*New*] on the `Feature Service` app toolbar.

© 2023 Creatio. All rights reserved.

As a result, Creatio will display the [*Feature*] button that contains the fully developed feature on the `Feature Service` app page.



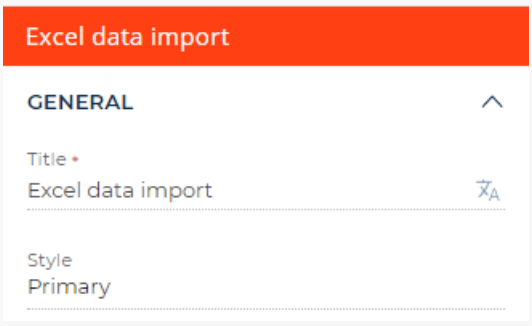
Hide the feature on a page due to insufficient access permissions

Medium

Example. Hide the [*Excel data import*] button on the record page of the custom [*Rights Service*] section if the user lacks permission to import Excel data.

1. Set up the page UI

1. Create a custom `Rights Service` app based on the [*Records & business processes*] template. To do this, follow the guide in the user documentation: Create a custom app.
2. Open the [*Rights Service form page*] page in the working area of the `Rights Service` app page.
3. Delete the [*Name*] field the [*Rights Service form page*] page includes by default.
4. Add a **button that starts Excel data import**.
   a. Add a [*Button*] type component to the toolbar of the Freedom UI Designer.
   b. Click in the action panel of the Freedom UI Designer and fill out the **button properties** in the setup area.
      - Set [*Title*] to "`Excel data import`."
      - Select "`Primary`" in the [*Style*] property.



© 2023 Creatio. All rights reserved.

5. Click </> in the action panel of the Freedom UI Designer. After you save the page settings, Creatio opens the source code of the Freedom UI page.

2. Hide the feature if the user lacks permission to access it

Configure the business logic in the Client Module Designer. For this example, hide the feature if the user lacks permission to access it

1. Enable the `sdk.RightsService` that checks access permissions. To do this, add `@creatio-devkit/common` to the AMD module as a dependency.

AMD module dependencies

```
/* Declare the AMD module. */
define("UsrAppRightsService_FormPage", /**SCHEMA_DEPS*/["@creatio-devkit/common"] /**SCHEMA_D
        ...
    };
});
```

2. Add the `CanImportFromExcel` attribute that stores the user's access permission data to the `viewModelConfig` schema section.

`viewModelConfig` schema section

```
viewModelConfig: /**SCHEMA_VIEW_MODEL_CONFIG*/{
    "attributes": {
        ...,
        /* The attribute that stores the user's access permission data. */
        "CanImportFromExcel": {}
    }
}/**SCHEMA_VIEW_MODEL_CONFIG*/,
```

3. Bind the `visible` property of the `ExcelDataImportButton` element to the `CanImportFromExcel` model attribute in the `viewConfigDiff` schema section. The `visible` property flags the button as visible.

`viewConfigDiff` schema section

```
viewConfigDiff: /**SCHEMA_VIEW_CONFIG_DIFF*/[
    {
        "operation": "insert",
        "name": "ExcelDataImportButton",
        "values": {
            ...,
            /* The property that flags the button as visible. Bound to the CanImportFromExcel
            "visible": "$CanImportFromExcel"
```

© 2023 Creatio. All rights reserved.

```
        },
        ...
    }
]/**SCHEMA_VIEW_CONFIG_DIFF*/,
```

4. Add a custom implementation of the `crt.HandlerViewModelInitRequest` system query handler to the `handlers` schema section. Execute the handler when Creatio initializes the `View` model.

   a. Create an instance of the service that checks access permissions from `@creatio-devkit/common`.

   b. Retrieve data about the user's access permission to the `CanImportFromExcel` system operation.

   c. Specify the data in the `CanImportFromExcel` attribute.

`handlers` schema section

```
handlers: /**SCHEMA_HANDLERS*/[
    {
        request: "crt.HandleViewModelInitRequest",
        /* The custom implementation of the system query handler. */
        handler: async (request, next) => {
            /* Create an instance of the service that checks access permissions from @creatio
            const rightService = new sdk.RightsService();
            /* Retrieve data about the user's access permission to the CanImportFromExcel sys
            const canImportFromExcel = await rightService.getCanExecuteOperation('CanImportFr
            /* Specify the data in the CanImportFromExcel attribute. */
            request.$context.CanImportFromExcel = canImportFromExcel;
            /* Call the next handler if it exists and return its result. */
            return next?.handle(request);
        },
    }
] /**SCHEMA_HANDLERS*/,
```

Complete source code of the page schema

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

Outcome of the example

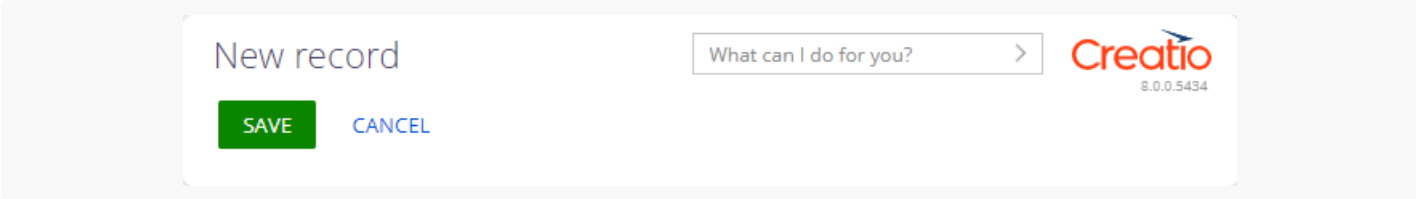
To **view the outcome of the example without the access permission**:

1. Log in to the app as a user who lacks the permission to import data from Excel. For example, create a new user or revoke the permission from an existing user. To add a user, follow the guide in the user documentation: Add users. To set up access permissions, follow the guide in the user documentation: System operation permissions. The [*Excel import*] (`CanImportFromExcel` code) system operation manages Excel data import.
2. Open the `Rights Service` app page and click [*Run app*].

© 2023 Creatio. All rights reserved.

3. Click [*New*] on the `Rights Service` app toolbar.

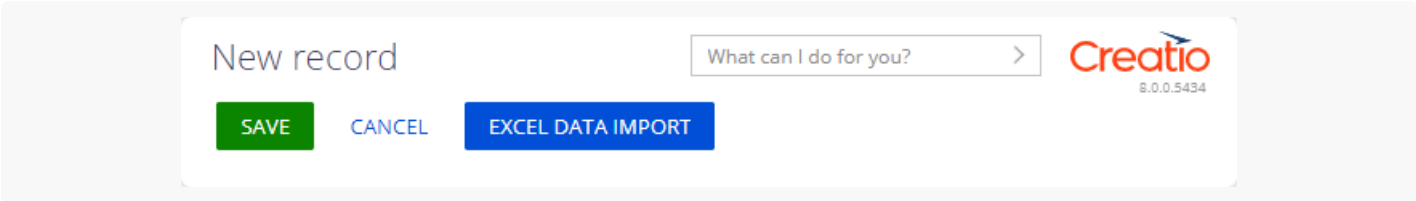
As a result, Creatio will hide the [*Excel data import*] button that starts Excel data import on the `Rights Service` app page.



To **view the outcome of the example with the access permission**:

1. Refresh the `Rights Service` app page.
2. Click [*New*] on the `Rights Service` app toolbar.

As a result, Creatio will display the [*Excel data import*] button that starts Excel data import on the `Rights Service` app page.



© 2023 Creatio. All rights reserved.