Creatio Academy

Portal

Version 8.0

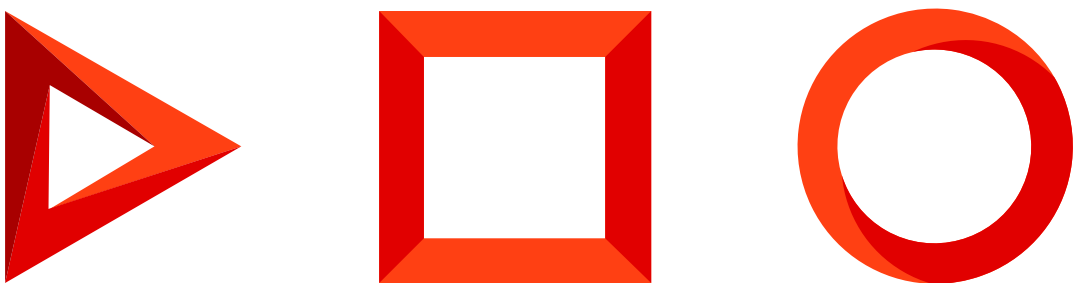


Table of Contents

Portal basics

Beginner

Creatio portal is a low-code component that provides secure and managed access to Creatio data and functionality for internal and external users, customers, and partners. The interface and tools of the portal are the same as those of the main Creatio application. The portal development process has the same principles as development for the main application. Learn more about setting up the Creatio portal in the "Getting started with Creatio portal" article.

Work with Portal

Creatio portal provides tools for solving a broad range of business tasks. Creatio portal is suitable for a variety of use cases, the most common being:

- `Customer self-service, such as in technical support.` Give a self-service option to your customers and focus the time and expertise of your support agents on tasks more important than case registration. Empower your customers to submit support cases and track the resolution progress directly on the portal. Provide your customer access to your knowledge base articles to help them find answers quickly. Service multiple customers at once while avoiding queues and loss in productivity.
- `Communications with internal and external customers, for instance, an HR portal.` Configure the ability to service external employees and contractors who do not actively use Creatio: create applications, submit them for approval, and track their progress. An HR portal can act as a central hub for all the essential company documents and policies that are in the public domain.
- `Interaction with external users (clients, dealers, and partners) at all sales stages.` Create partner programs, process leads, and close opportunities along with your partners by using lead management and corporate sales processes. Keep track of the partner tiers, training sessions, and certified experts.

Through the portal, Creatio users can access selected licensed sections and their associated data. You can use standard access permissions to choose which of your business data is available on the portal, and make sure that any sensitive and confidential information is safe out of external users' reach.
Portal users can:

- Create new records and modify the existing ones.
- Add notes.
- Attach files.
- Leave messages for the support service.
- Work using business processes.

In addition to the regular object permissions, the data available for portal users is limited by the [*List of objects available for portal users*] lookup. Only the objects included in the lookup are accessible via the portal UI.

The most common scenario is the collaborative work of portal users and system users in the same section but with different sets of page fields. Columns that can be configured in the section list and columns for filtering (in quick filters or dynamic groups) are limited for portal users. When you add fields to the edit page using the

Section Wizard, the columns are added to the [*List of schema fields for portal access*] lookup by default and become available. If a field is not available on the page, and you want it on the portal list, add the field to the lookup manually.

Portal users are grouped in the `“All portal users”` organizational role. You can create individual portal users or group them into an organization by connecting them to a specific account.

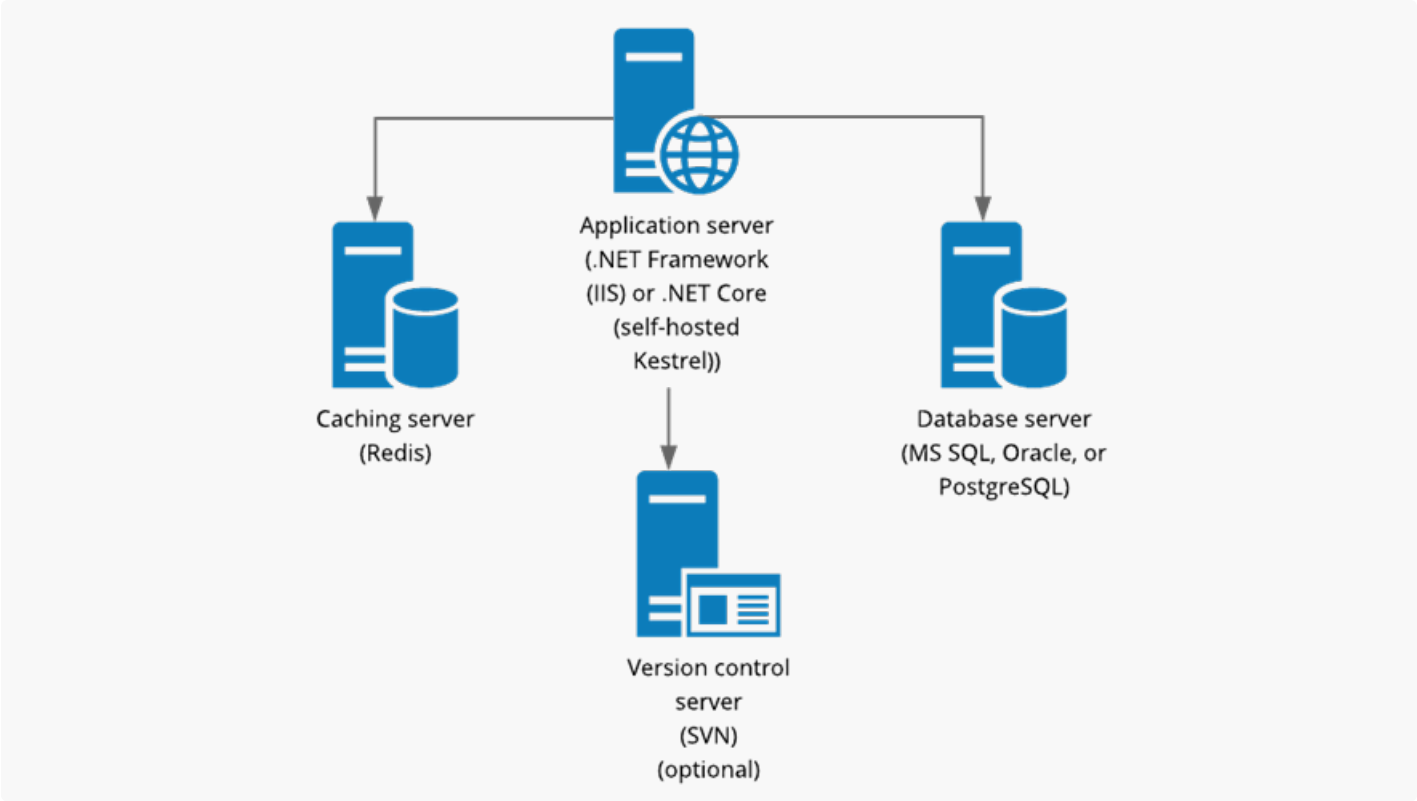
Creatio portal is available in the following configurations:

- `Self-service portal.` You can use the self-service portal both as the primary case registration channel and an auxiliary channel. A portal user can find information in the knowledge base or submit a support service request. This configuration limits the list of sections that portal users can access to the [Case] and [Knowledge base] sections. You can customize them using the system designer. You cannot add custom sections to the self-service portal.
- `Customer portal.` The customer portal is designed for process automation, e.g., providing services, confirming applications and service requests, etc. Portal users can initiate processes (e.g., create orders, cases, etc.) or participate in the (e.g., approve requests). This portal configuration type allows you to set up and use up to three custom portal sections. Create a custom section in the main application first, then make a portal version of the section. Portal users can also access the [Case], and [Knowledge base] sections. Additionally, users of Creatio Bank Customer Journey can access the [Applications] section. You can also add the [Documents] section if the main Creatio application supports it. A user-created section built on a licensed base product object is not considered a custom section.
- `Partner portal.` The partner portal is designed for companies that work with customers via partner networks. The partner portal is a joint communication platform for passing leads between partners and partner cross-sales. This configuration enables portal users to access the [Partner program], [Leads], [Opportunities], [Marketing activities], and [Orders] sections.

Portal configurations can be used simultaneously. Each configuration enables portal users to access the basic portal functionality (main page, user profile settings, knowledge base, data segmentation). The list of available sections in the Creatio portal depends on the portal configuration.

Portal architecrute schema

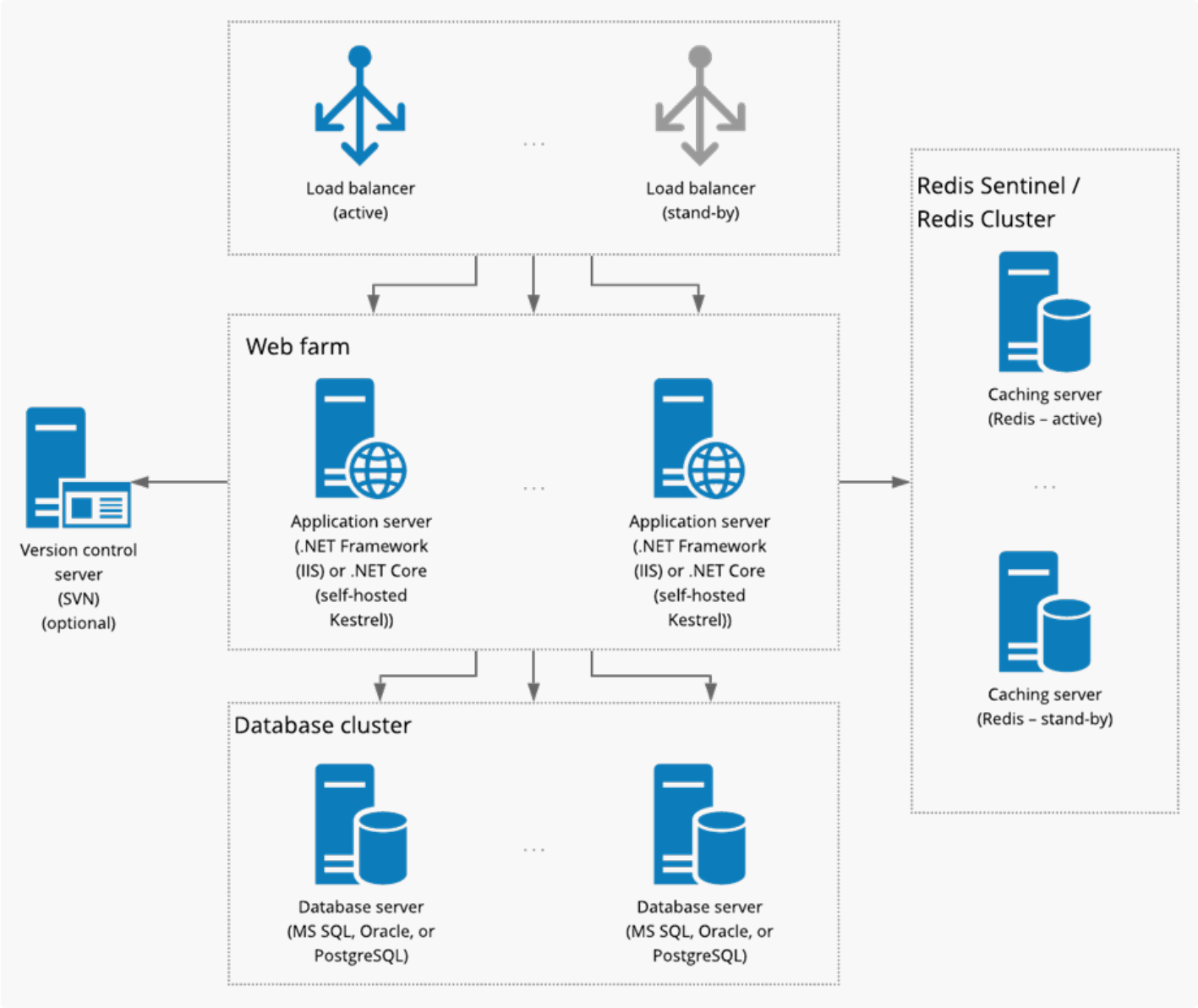
The architecture scheme of the Creatio portal is the same as the architecture scheme of the main application.



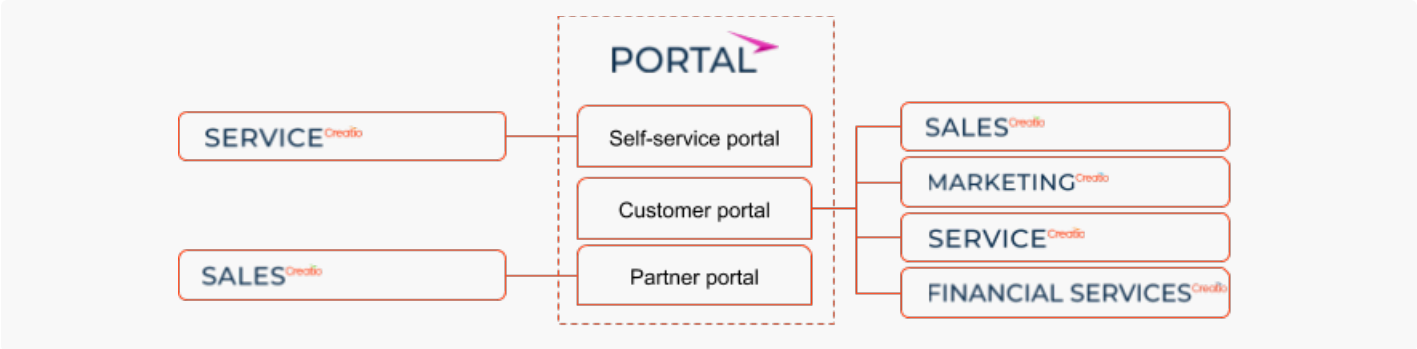
Portal scalability

You can enhance the performance of large-scale Creatio projects through horizontal scaling. When using horizontal scaling, the architecture of the Creatio portal includes the following components.

- Load balancers.
  The load balancer may be either hardware or software. To work in fault-tolerant mode, use the HTTP/HTTPS traffic balancer that supports the WebSocket protocol. Learn more about setting up the HAProxy load balancer in the article.
- Web farm (multiple application servers).
- A Redis or Redis Sentinel caching server / a Redis cluster (multiple Redis caching servers).
- A database server or a database cluster (multiple database servers).
- Version control system server `optional`.



Portal compatibility with Creatio products



Portal deployment options

Creatio portal is available for applications deployed both on-site and in the cloud.

To ensure the safety of data, the on-site application must be deployed using a web farm when installing the portal. Learn more about setting up a Creatio application with external network portal access in the article.

Self-service portal

Advanced

The Self-service Portal (SSP) is an integral part of the Service Creatio, enterprise edition and Creatio customer center products, as well as all bundles that these products are part of.

The SSP is a workplace where portal users are automatically redirected after login.

Portal users have access to the [*Cases*] and [*Knowledge base*] sections and to the [*Self-service portal main page*], which contains general summary information and works as a single workplace for a portal user.

The [*Cases*] section is used for registration of cases by the customers, viewing the status of their cases, entering additional case information and for obtaining information about case resolution process. By default, a portal user has access to those cases where this user is specified as a contact. The user can enter additional information, publish messages and interact with the service staff on the case page. The case history is displayed on the case page at the [*Processing*] tab.

The portal's [*Knowledge base*] section is used for searching for reference information or a solution. This section can be filled only by the helpdesk staff from the main interface of the system.

Portal interface

From the development point of view, the portal is a preconfigured separate workplace. By default, this workplace is not available for the ordinary portal users. A system user with the "portal user" type automatically enters this workplace (the portal main page) after authorization.

Portal sections and pages are the same as sections and system edit pages from the main system interface and they work with the same `Entities`. The case edit page on the portal is simpler compared to the regular case page and does not contain the majority of fields. These edit pages are different objects in configuration (`CasePage` and `PortalCasePage`).

The system of portal access permissions slightly differs. To grant access to the specific entities (EntitySchema) for the portal users, you need to specify these entities in the [*List of objects available for portal users*] lookup. Self-service portal licenses limit the number of records that can be added to this lookup. By default, the number is limited to 70 records.

Working with the page wizard on the portal

The portal user cannot access the functions of the page-, detail- and section wizards These functions can be accessed from the main system interface with administrator permissions in the following way:

1. Enter the [*Workplace setup*] section in the system designer.
2. Select the [*Portal*] workplace and click the [*Open*] button.
3. Select the required section and click the [*Section wizard*] button.

The standard section wizard will open.

Configuring the portal and portal users

To start using the portal:

1. Ensure that the `/configuration/terrasoft/auth` option in the `web.config` file of the application loader (the "external" `web.config`) has the following in it:

   `/configuration/terrasoft/auth` option

   ```
   <terrasoft>
       auth providerNames="InternalUserPassword,SSPUserPassword" …>
       ...
   </terrasoft>
   ```

   This setting is responsible for the authorization in the portal users in the system.
2. Create a contact for the user.
3. Create a user with the "Portal user" type. Fill out the required fields.
4. Provide all necessary licenses for the user.

A portal user is required to have a valid time zone specified in the profile. The time zone is not specified for new users by default. Portal users must edit their profiles and select the proper time zone The system will display all dates and times in the portal user's local time.

Restricting access to web services for portal users

The portal is a Creatio platform component whose main purpose is to implement self-service processes for your customers and partners. You can also use the portal to organize the work of internal users if you need to restrict access permissions to functionality of the primary application.

Using portal enables many external users (who are not employees of your organization) to access some of the Creatio data. For this, you need to manage which users (portal or company employees) have access to the application web services.

Attention. This article is valid for applicaation version 7.13.2 and up.

Route prefixes for web services configuration

Route prefixes in Creatio enable managing access to the application web services. You can set the needed route prefix using specific `ServiceRoute` attributes of the service class.

Route prefixes for configuration web services

Access level	Attribute	Route prefix	Example of code

Access level	Attribute	Route prefix	Example of code

For portal users only	`SspServiceRoute`	`/ssp`	`[ServiceContract]`
			`[SspServiceRoute]`
			`public class SspOnlyService : BaseServic`
			`{`
			`}`
			
			Example of call
			`~/ssp/rest/SspOnlyService`
			`~/ssp/soap/SspOnlyService`

For internal users only	`DefaultServiceRoute`
	or
	no `ServiceRoute` attribute is specified
		none	`[ServiceContract]`
			`public class InternalService : BaseServi`
			`{`
			`}`
			
			or
			
			`[ServiceContract]`
			`[DefaultServiceRoute]`
			`public class InternalService : BaseServi`
			`{`
			`}`
			
			Example of call
			`~/rest/InternalService`
			`~/soap/InternalService`

For both: the internal and portal users	Both the `DefaultServiceRoute` and `SspServiceRoute`	`/ssp` or none	`[ServiceContract]`
			`[DefaultServiceRoute]`
			`[SspServiceRoute]`
			`public class CommonService : BaseService`
			`{`
			`}`
			
			Example of call
			`~/rest/CommonService`

Access level	Attribute	Route prefix	Example of code

			`~/soap/CommonService`
			`~/ssp/rest/CommonService`
			`~/ssp/soap/CommonService`

	The `ServiceRoute` attribute with the specified prefix (e.g., "`custom`")
	[*ServiceRoute("custom")*]
		Arbitrary route prefix of the service	`[ServiceContract]`
			`[ServiceRoute("custom")]`
			`public class CustomPrefixService : BaseS`
			`{`
			`}`
			
			Example of call
			`~/custom/rest/CustomPrefixService`
			`~/custom/soap/CustomPrefixService`

Note. You can use several attributes at the same time: `ServiceRoute`, `SspServiceRoute` and `DefaultServiceRoute`. As a result, the routes for services with all prefix variants will be created.

Restricting access to the internal API for portal users

If a portal user (`SspUserConnection`) addresses a service with a route that does not contain the "`/ssp`" prefix, the service will return error 403 (`Forbidden`).

Restricting access to the portal API for internal users

If an internal user (`UserConnection`) addresses the service with the route that contains the "`/ssp`" prefix, the service will return a page with code 403 (`Forbidden`).

The ServiceStack core services

The `ServiceStack` core services (`DataService`, `ManagerService`, etc.) are available by default to the internal users only.

To make any of these methods available on the portal, tag such method with the `SspServiceAccess` attribute – in this case, the method will have an additional route with the `~/DataService/ssp/…` prefix.

If you need to have a different logic for the portal, create a new service by specifying the `SspServiceAccess` attribute for it and pass the name of the original method to its constructor. Example:

SspServiceAccess attribute

```
[SspServiceAccess(nameof(SelectQuery))]
```

```
public class SspSelectQuery : SelectQuery
{
}
```

This service creates a contract whose method will be registered at the following path: `~/DataService/ssp/SelectQuery`

Access to the `ServiceStack` methods with the “`ssp`” prefix is denied to the internal users. Access to the `ServiceStack` methods without the “`ssp`” prefix is denied to the portal users.

Example of using the PortalMessagePublisherExtensions mixin

Advanced

Example. Using the PortalMessagePublisherExtensions mixin.

Example implementation

CaseSectionActionsDashboard

```
define("CaseSectionActionsDashboard", ["PortalMessagePublisherExtensions"], function() {
    return {
        mixins: {
            /**
             * @class PortalMessagePublisherExtensions extends tabs and publishers configs.
             */
            PortalMessagePublisherExtensions: "Terrasoft.PortalMessagePublisherExtensions"
        },
        methods: {
            /**
             * @inheritdoc Terrasoft.SectionActionsDashboard#getExtendedConfig
             * @overridden
             */
            getExtendedConfig: function() {
                // Getting the tab configuration object from the parent method.
                var config = this.callParent(arguments);
                // Calling the mixin method, adding a portal tab configuration.
                this.mixins.PortalMessagePublisherExtensions.extendTabsConfig.call(this, config)
                // Returns the extended configuration object.
                return config;
            },
```

```
            /**
             * @inheritdoc Terrasoft.SectionActionsDashboard#getSectionPublishers
             * @overridden
             */
            getSectionPublishers: function() {
                // Getting a collection of message publishers from the parent method.
                var publishers = this.callParent(arguments);
                // Calling the mixin method, adding a portal channel.
                this.mixins.PortalMessagePublisherExtensions.extendSectionPublishers.call(this,
                // Returns the extended collection of message publishers.
                return publishers;
            }
    },
        diff: /**SCHEMA_DIFF*/[
            {
                "operation": "insert",
                "name": "PortalMessageTab",
                "parentName": "Tabs",
                "propertyName": "tabs",
                "values": {
                    "items": []
                }
            },
            {
                "operation": "insert",
                "name": "PortalMessageTabContainer",
                "parentName": "PortalMessageTab",
                "propertyName": "items",
                "values": {
                    "itemType": this.Terrasoft.ViewItemType.CONTAINER,
                    "classes": {
                        "wrapClassName": ["portal-message-content"]
                    },
                    "items": []
                }
            },
            {
                "operation": "insert",
                "name": "PortalMessageModule",
                "parentName": "PortalMessageTab",
                "propertyName": "items",
                "values": {
                    "classes": {
                        "wrapClassName": ["portal-message-module", "message-module"]
                    },
                    "itemType": this.Terrasoft.ViewItemType.MODULE,
                    "moduleName": "PortalMessagePublisherModule",
                    "afterrender": {
                        "bindTo": "onMessageModuleRendered"
```

```
                    },
                    "afterrerender": {
                        "bindTo": "onMessageModuleRendered"
                    }
                }
            }
        ]/**SCHEMA_DIFF*/
    };
});
```

Change access to web service for portal users

Advanced

The application has a set of base services and only internal users have access to them.

To change access to the base service:

1. In the custom package, create a service with the access settings for portal users.
2. In the service, add the base service methods that must be available for portal users.
3. Change the custom or extend the base client schemas by changing the call of base service to the call of the created service (see step 1).
4. Compile the configuration.

Example. Change access to web service for portal users.

Example implementation

Example of the service source code that extends access to the `ActivityUtilService` base service:

PartnerActivityUtilService

```
namespace Terrasoft.Configuration
{
    using System;
    using System.IO;
    using System.Runtime.Serialization;
    using System.ServiceModel;
    using System.ServiceModel.Activation;
    using System.ServiceModel.Web;
    using Terrasoft.Configuration.FileUpload;
    using Terrasoft.Core.Factories;
```

```
    using Terrasoft.Web.Common;
    using Terrasoft.Web.Common.ServiceRouting;

    [ServiceContract]
    // The service is available for both: the internal and portal users
    [DefaultServiceRoute]
    [SspServiceRoute]
    [AspNetCompatibilityRequirements(RequirementsMode = AspNetCompatibilityRequirementsMode.Requ
    public class PartnerActivityUtilService: BaseService {
        // Base service, access needs to be extended
        private static readonly ActivityUtilService _baseService = new ActivityUtilService();

        [OperationContract]
        [WebInvoke(Method = "POST", BodyStyle = WebMessageBodyStyle.Wrapped, RequestFormat = Web
        public Guid CreateActivityFileEntity(JsonActivityFile jsonActivityFile) {
            return _baseService.CreateActivityFileEntity(jsonActivityFile);
        }
        [OperationContract]
        [WebInvoke(Method = "POST", BodyStyle = WebMessageBodyStyle.Wrapped, RequestFormat = Web
        public Guid CreateFileEntity(JsonEntityFile jsonEntityFile) {
            return _baseService.CreateFileEntity(jsonEntityFile);
        }
    }
}
```

PortalMessagePublisherExtensions mixin JS

Advanced

A mixin is a class designed to extend the functions of other classes. Mixins are separately created classes with additional functionality. Learn more about mixins in the "Mixins. The "mixins" property" article.

The `PortalMessagePublisherExtensions` mixin is used for the extension of the `SectionActionsDashboard` schema (and its derived schemas). It allows you to extend the configuration of the `SectionActionsDashboard` tabs with the portal message tab `PortalMessageTab` and add the corresponding Portal message `portal`. The mixin is implemented in the [*PortalMessagePublisher*] package and is available in the Service Enterprise product (or in the bundles that include this product).

Methods

`extendTabsConfig(config) : Object`

Extends the configuration of the `SectionActionsDashboard` tabs with the portal messages tab `PortalMessageTab`.

Returns the augmented object (`Object`) of the `SectionActionsDashboard` tab configuration.

The `config` parameter (`Object`) – `SectionActionsDashboard` tab configuration object.

`extendSectionPublishers(publishers) : Array`

Adds a portal channel (`Portal`) to the message publisher collection.

Returns the augmented collection of message publishers (`Array`).

The `publishers` parameter `(Array)` is the collection of message publishers.