

Develop your first application

Step 5. Create a custom web service

Version 7.17



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.

Table of Contents

Step 5. Create a custom web service	4
Create a custom web service	4
Modify the page source code	6

Step 5. Create a custom web service

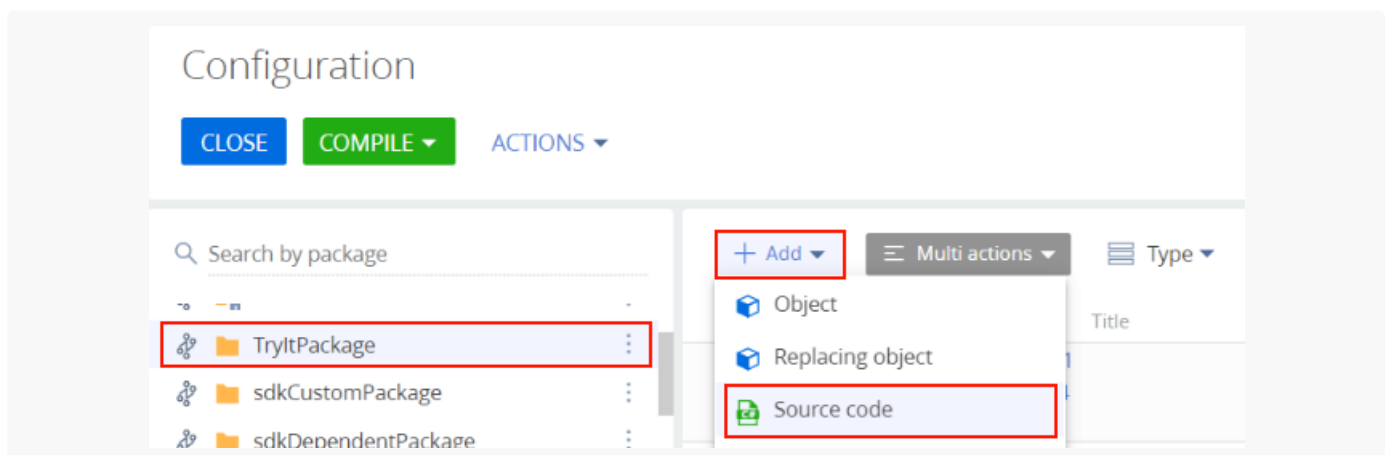
 Beginner

On the [previous step](#), we implemented population of the group class timetable.

Now, implement a web service that provides information about the number of classes in the timetable. To call a web service, add the [*Web service*] button to the record page.

Create a custom web service

1. [Go to the \[Configuration \] section](#).
2. Select the "TryItPackage" [package](#) from the package list.
3. Click [*Add*] on the workspace toolbar and select the [*Source code*] configuration element type.



4. **Fill out the property fields** as follows:

- Set [*Code*] to "UsrClassService".
- Set [*Title*] to "Class service".

Source code ×

Code *
 UsrClassService

Title *
 Class service ↕A

Package
 TryItPackage

Description ↕A

CANCEL
APPLY

5. Add the source code in the Schema Designer.

```

UsrClassService.cs

namespace Terrasoft.Configuration.UsrClassService
{
    using System;
    using System.ServiceModel;
    using System.ServiceModel.Web;
    using System.ServiceModel.Activation;
    using Terrasoft.Core;
    using Terrasoft.Core.DB;
    using Terrasoft.Common;
    using Terrasoft.Web.Common;
    using Terrasoft.Core.Entities;

    [ServiceContract]
    [AspNetCompatibilityRequirements(RequirementsMode = AspNetCompatibilityRequirementsMode.Required)]
    public class UsrClassService: BaseService
    {

        [OperationContract]
        [WebInvoke(Method = "POST", RequestFormat = WebMessageFormat.Json, BodyStyle = WebMessageFormat.Json)]
        [WebResponseFormat = WebMessageFormat.Json]
        public int GetTrainingsQuantity(string code) {
            var classQuery = new Select(UserConnection)
                .Column("Id")
                .From("UsrClass")
    }
    }
    
```

```

        .Where("UserCode")
            .IsEqual(Column.Parameter(code))
        as Select;
Guid id = classQuery.ExecuteScalar<Guid>();
if (id==Guid.Empty) {
    return -1;
}
var countQuery = new Select(UserConnection)
    .Column(Func.Count("Id")).As("Count")
    .From("UsrGroupTraining")
    .Where("UsrClassId")
        .IsEqual(Column.Parameter(id))
    as Select;
int result = countQuery.ExecuteScalar<int>();

return result;
    }
}
}

```

6. Click [*Publish*] to save the schema.

Modify the page source code

Add a class page button that calls the web service and displays the dialog box that contains the number of group classes in the timetable.

1. [Go to the \[Configuration \] section.](#)
2. Select the "TryItPackage" [package](#) from the package list.
3. The Wizards added schemas of various types to the package. Filter schemas by the [*Client module*] type.

The screenshot shows the 'Configuration' window in Creatio. On the left, a tree view shows the package structure with 'TryItPackage' selected. The main area displays a table of schemas filtered by 'Client module'.

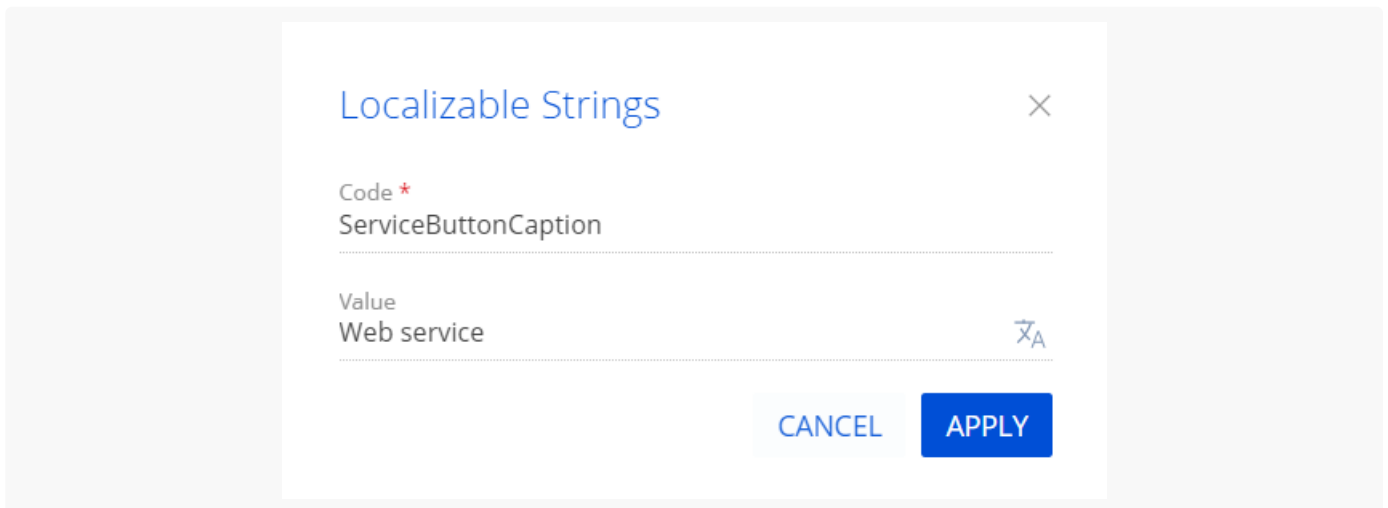
Name	Title	Status	Type	Object	Modified on	Package
UsrClass1Page *	Edit page: "Classes"		Client module		6/8/2022, 1:46:58 PM	TryItPackage
UsrClass5383a01Section *	Section schema: "Classes"		Client module		6/8/2022, 1:46:34 PM	TryItPackage
UsrSchemaaa01c268Page *	Group training		Client module		6/8/2022, 2:07:17 PM	TryItPackage
UsrSchemae5be1ee3Detail *	Detail schema: "Group trainings"		Client module		6/8/2022, 2:07:24 PM	TryItPackage

4. Double-click the `UsrClass1Page` schema to open it.
5. Add a new localizable string for the button name to the schema.

Click the **+** button in the [*Localizable strings*] block of the properties panel and fill out the **localizable string**

properties:

- Set [*Code*] to "ServiceButtonCaption".
- Set [*Value*] to "Web service".



6. Modify the source code.

UsrClass1Page.js

```

define("UsrClass1Page", ["ServiceHelper", "ProcessModuleUtilities"], function(ServiceHelper,
    return {
        entitySchemaName: "UsrClass",
        /* No changes. */
        messages: {
            // ...
        },

        /* No changes. */
        attributes: {
            // ...
        },

        modules: /**SCHEMA_MODULES*/{}/**SCHEMA_MODULES*/,

        /* No changes. */
        details: /**SCHEMA_DETAILS*/{
            // ...
        }/**SCHEMA_DETAILS*/,

        businessRules: /**SCHEMA_BUSINESS_RULES*/{}/**SCHEMA_BUSINESS_RULES*/,

        /* Add the new method to the existing methods. */
        methods: {

```

```

// ...

/* The method that handles the button clicks. */
onGetServiceInfoClick: function() {
    /* Retrieve the section code to pass as the incoming parameter of the service
    var code = this.get("UsrCode");
    var serviceData = {
        code: code
    };
    /* Call the service method. */
    ServiceHelper.callService("UsrClassService", "GetTrainingsQuantity",
        function(response) {
            var result = response.GetTrainingsQuantityResult;
            /* Display the service method output in the dialog box. */
            this.showInformationDialog(result);
        }, serviceData, this);
    }
},
dataModels: /**SCHEMA_DATA_MODELS*/{/**SCHEMA_DATA_MODELS*/,

/* Display the button on the record page. */
diff: /**SCHEMA_DIFF*/[

    // ...
    /* Add the button element to the record page. */
    {
        "operation": "insert",
        /* The parent element name. */
        "parentName": "ActionButtonsContainer",
        "propertyName": "items",
        /* The element name. */
        "name": "GetServiceInfoButton",
        "values": {
            /* Set the element type to button. */
            itemType: Terrasoft.ViewItemType.BUTTON,
            /* Retrieve the element caption from the localizable string. */
            caption: {bindTo: "Resources.Strings.ServiceButtonCaption"},
            /* The method that handles the button clicks. */
            click: {bindTo: "onGetServiceInfoClick"},
            enabled: true,
            /* The button position on the page. */
            "layout": {"column": 1, "row": 6, "colSpan": 2, "rowSpan": 1}
        }
    },

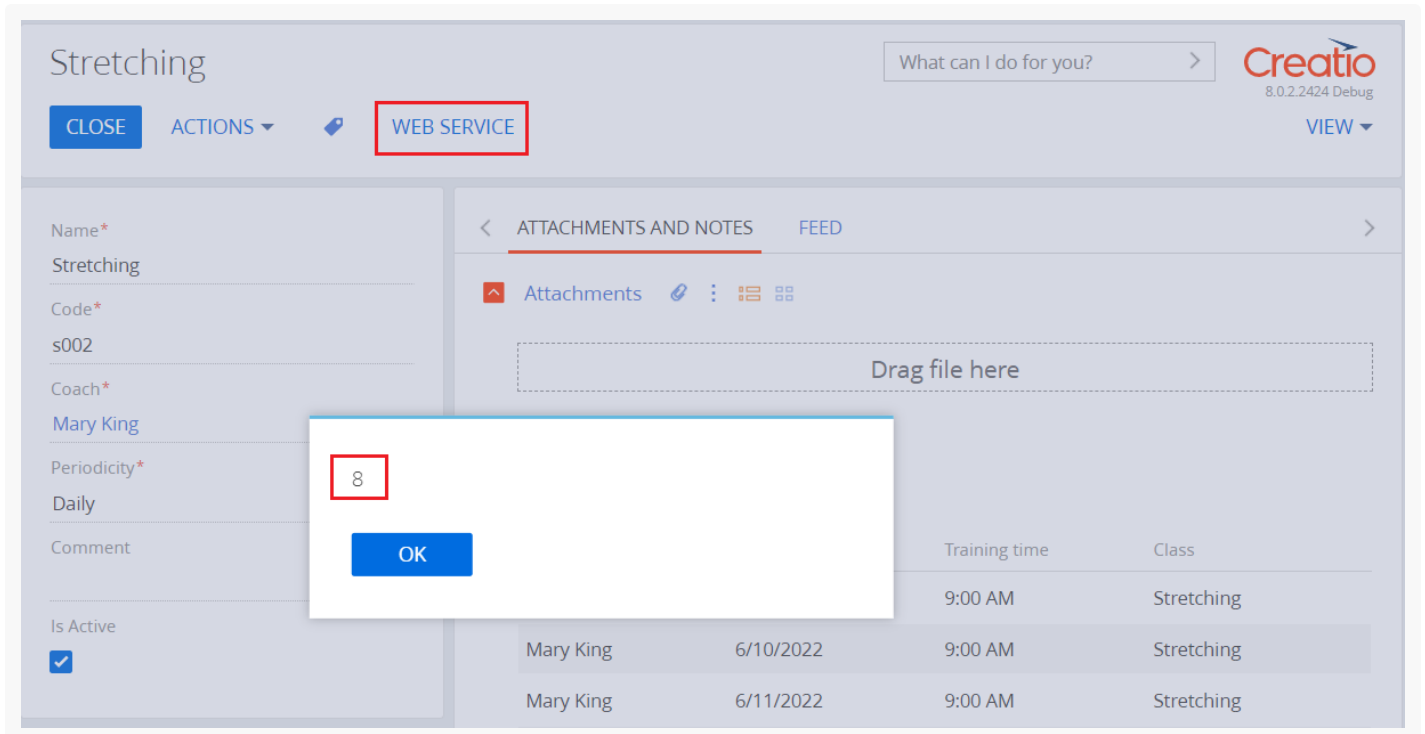
/**SCHEMA_DIFF*/

];
});

```


7. Click [Save] to save the schema.

As a result, we implemented the web service that returns the number of group classes.



The screenshot displays the Creatio user interface for the 'Stretching' entity. The top navigation bar includes a search box, the 'Creatio' logo (version 8.0.2.2424 Debug), and a 'VIEW' dropdown. The main header shows 'Stretching' with 'CLOSE', 'ACTIONS', and 'WEB SERVICE' buttons. The 'WEB SERVICE' button is highlighted with a red box. The left sidebar contains fields for 'Name*' (Stretching), 'Code*' (s002), 'Coach*' (Mary King), 'Periodicity*' (Daily), 'Comment', and 'Is Active' (checked). The right sidebar shows 'ATTACHMENTS AND NOTES' and 'FEED' tabs. Below these is an 'Attachments' section with a 'Drag file here' area. A modal dialog box is open in the center, showing the number '8' in a red box and an 'OK' button. In the background, a table lists training classes:

	Training time	Class
	9:00 AM	Stretching
Mary King	6/10/2022	9:00 AM
Mary King	6/11/2022	9:00 AM
	9:00 AM	Stretching