Creatio Academy

Data operations (back-end)

Execute operations in the background

Version 8.0

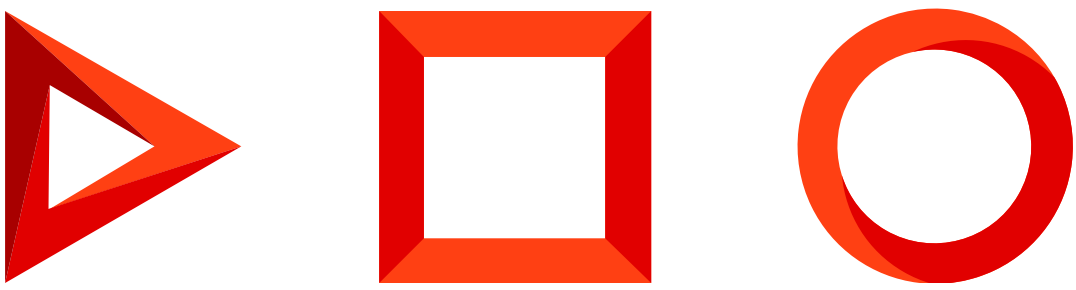


Table of Contents

Execute operations in the background

Medium

Execute operations in the background to run time-consuming operations without holding up the UI.

The `Terrasoft.Core.Tasks.Task` class implements the `StartNew()` and `StartNewWithUserConnection()` methods that start background operations. You can use base .NET data types (e. g., `string`, `int`, `Guid`, etc) or custom types as parameters. Unlike `StartNew()`, the `StartNewWithUserConnection()` method runs background operations that require the `UserConnection` user connection.

The `MessagePack-CSharp` module converts parameters accepted by the background operation to a byte array. View the implementation on the `MessagePack-CSharp` module on GitHub. If the module cannot serialize or deserialize a parameter value, the module will throw an exception.

Attention. We recommend against using infinite loops in background operations as they block other Creatio tasks.

Describe the execution of an asynchronous operation in an individual class that must implement the `IBackgroundTask<in TParameters>` interface.

`IBackgroundTask<in TParameters>` interface

```
namespace Terrasoft.Core.Tasks
{
    public interface IBackgroundTask<in TParameters>
    {
        void Run(TParameters parameters);
    }
}
```

If the action requires a user connection, implement the `IUserConnectionRequired` interface in the class as well.

`IUserConnectionRequired` interface

```
namespace Terrasoft.Core
{
    public interface IUserConnectionRequired
    {
        void SetUserConnection(UserConnection userConnection);
    }
}
```

Implement the methods of the `Run` and `SetUserConnection` interfaces in the class that implements the `IBackgroundTask<in TParameters>` and `IUserConnectionRequired` interfaces.

Good to know when **implementing the methods**:

- Do not pass `UserConnection` to the `Run` method.
- Do not call the `SetUserConnection` method from the `Run` method. The application core calls this method and initializes a `UserConnection` instance when the background operation starts.
- Pass structures that comprise only simple data types to the `Run` method. Complex class instances are highly likely to cause parameter serialization errors.

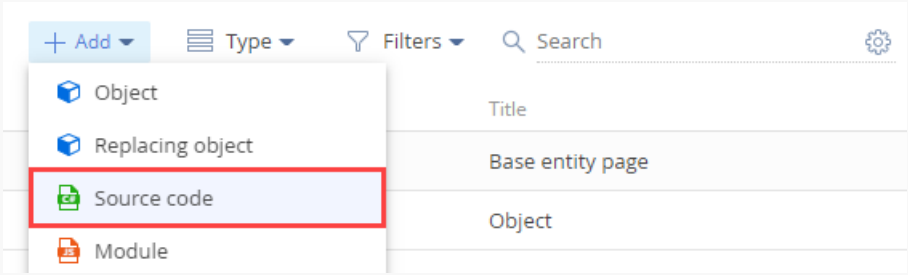
Register a background operation

Medium

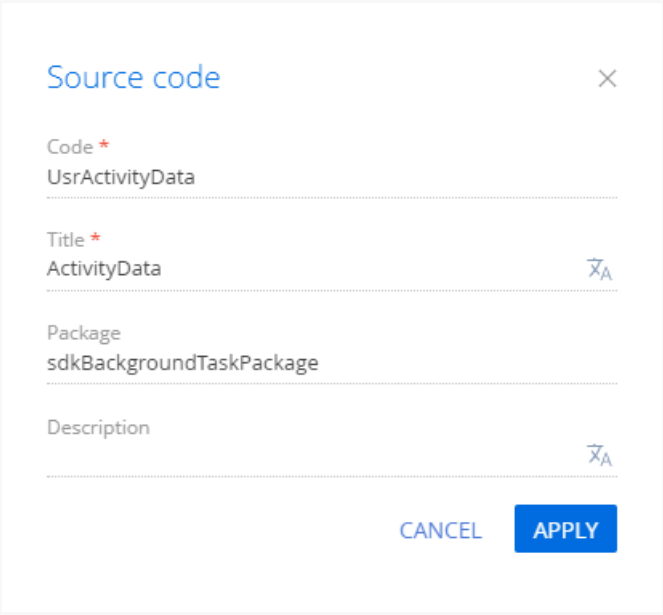
Example. Create a business process that registers a background operation. The background operation takes approximately 30 seconds. After this time passes, the process must add an [*Activity created by background task*] record to the list view of the [*Activities*] section list.

1. Create a class for the activity object

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



3. Go to the Schema Designer and fill out the schema properties:
   - Set [*Code*] to "`UsrActivityData`."
   - Set [*Title*] to "`ActivityData`."



Click [*Apply*] to apply the properties.

4. Add the source code in the Schema Designer.

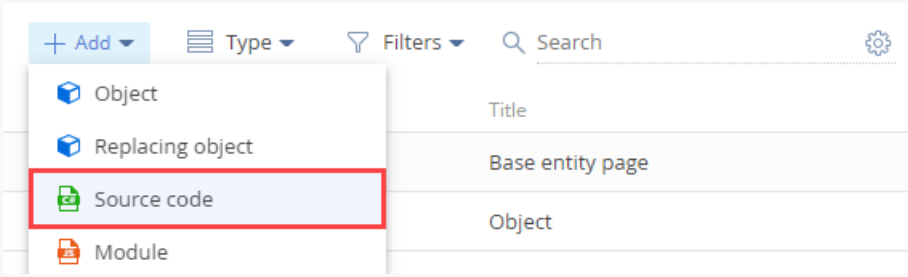
UsrActivityData

```
namespace Terrasoft.Configuration
{
    using System;
    using Terrasoft.Common;
    using Terrasoft.Core;
    using Terrasoft.Core.DB;
    public class UsrActivityData
    {
        public string Title { get; set; }
        public Guid TypeId { get; set; }
    }
}
```

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

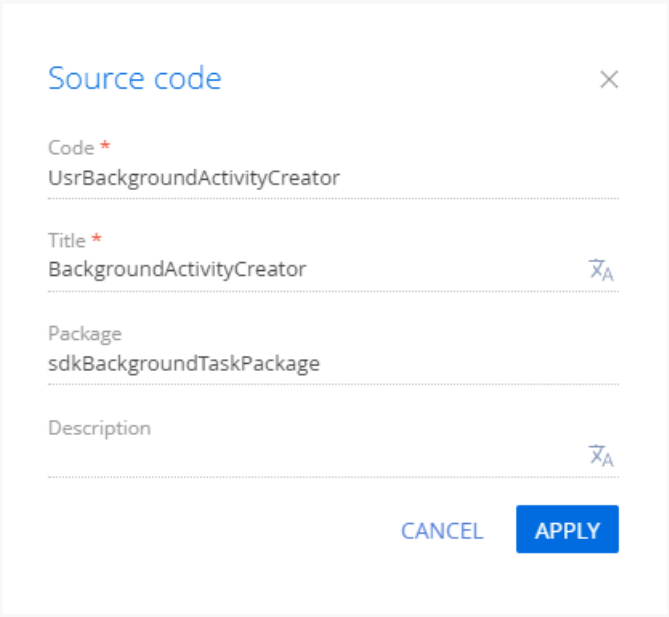
2. Create a class that adds an activity

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



3. Go to the Schema Designer and fill out the schema properties:

   - Set [*Code*] to "`UsrBackgroundActivityCreator`."
   - Set [*Title*] to "`BackgroundActivityCreator`."



   Click [*Apply*] to apply the properties.

4. Add the source code in the Schema Designer.

`UsrBackgroundActivityCreator`

```
namespace Terrasoft.Configuration
{
    using System;
    using Terrasoft.Common;
    using Terrasoft.Core;
    using Terrasoft.Core.DB;
    using Terrasoft.Core.Tasks;
    using System.Threading.Tasks;

    public class UsrBackgroundActivityCreator : IBackgroundTask<UsrActivityData>, IUserConnec
    {
```

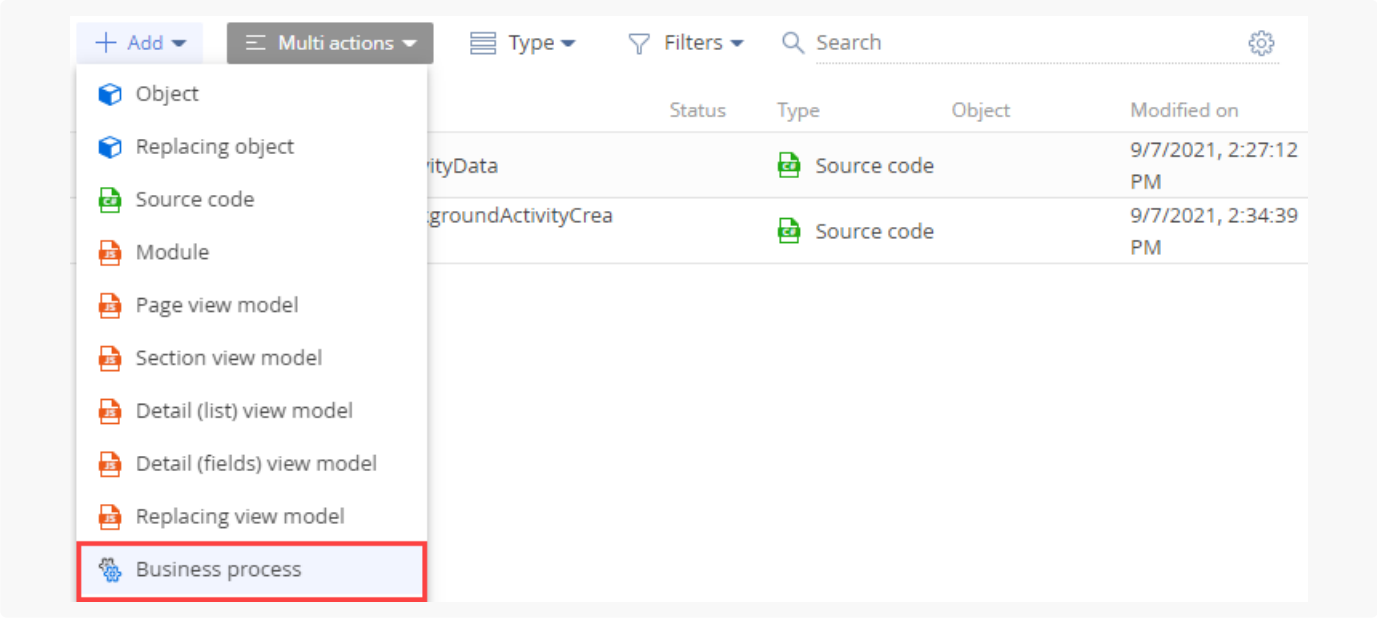
```
        private UserConnection _userConnection;

        /* Implement the Run method of the IBackgroundTask interface. */
        public void Run(UsrActivityData data) {
            /* Forced 30 second delay. */
            System.Threading.Tasks.Task.Delay(TimeSpan.FromSeconds(30));
            /* Create an activity. */
            var activity = new Activity(_userConnection){
                UseAdminRights = false,
                Id = Guid.NewGuid(),
                TypeId = data.TypeId,
                Title = data.Title,

                /* Set the activity category to "To do." */
                ActivityCategoryId = new Guid("F51C4643-58E6-DF11-971B-001D60E938C6")
            };
            activity.SetDefColumnValues();
            activity.Save(false);
        }

        /* Implement the SetUserConnection method of the IUserConnectionRequired interface. *
        public void SetUserConnection(UserConnection userConnection) {
            _userConnection = userConnection;
        }
    }
}
```

The `UsrBackgroundActivityCreator` class implements the `IBackgroundTask<UsrActivityData>` and `IUserConnectionRequired` interfaces. After a forced 30 seconds delay, the `Run()` method creates an instance of the [*Activities*] section object based on the given `UsrActivityData` instance.

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

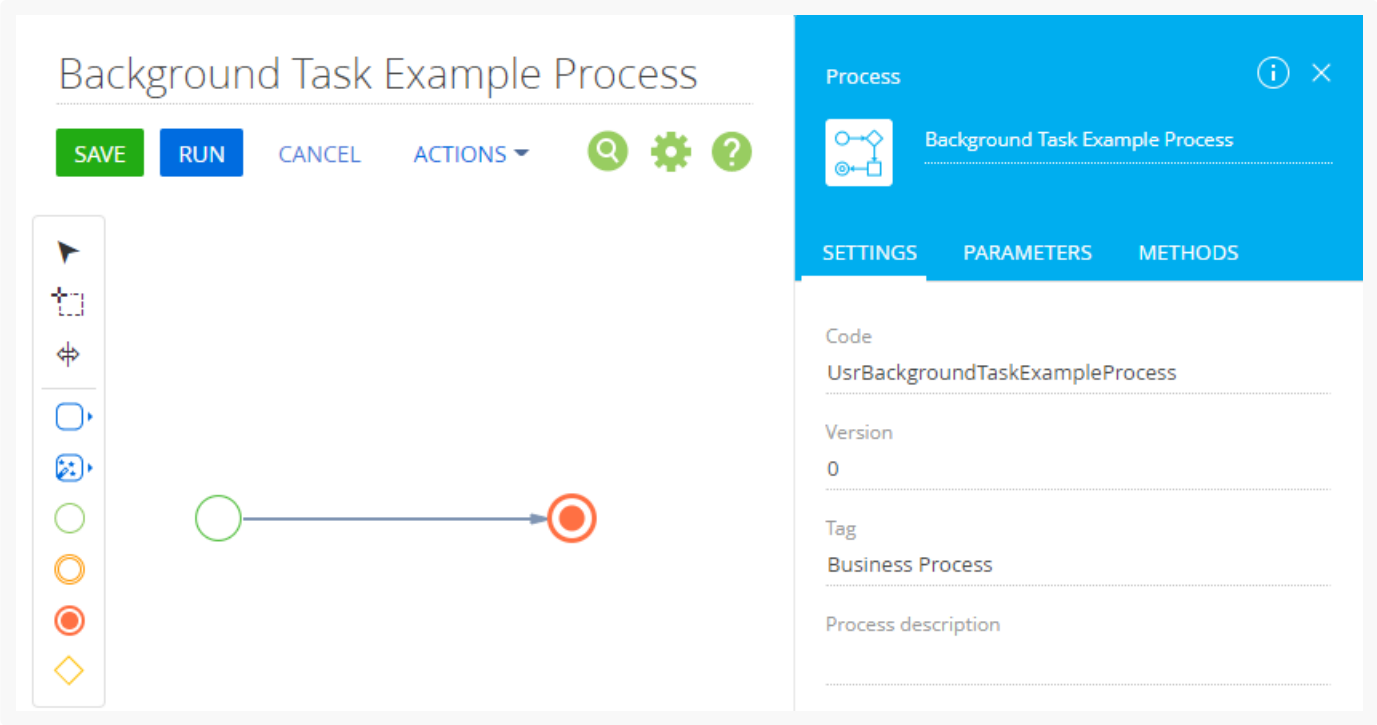
3. Create a business process that starts the background operation

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



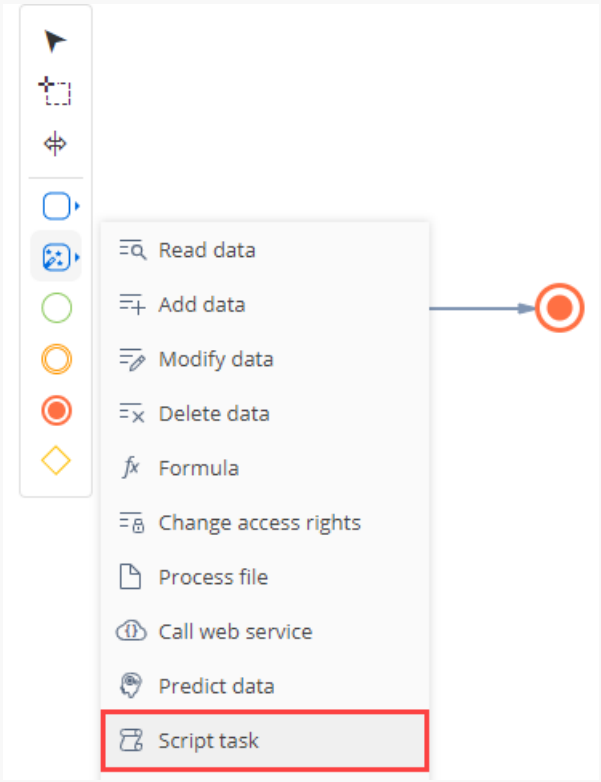
3. Go to the Process Designer and fill out the process properties:

   - Set the [*Title*] property in the element setup area to "`Background Task Example Process`."
   - Set the [*Code*] property on the [*Settings*] tab of the element setup area to "`UsrBackgroundTaskExampleProcess`."



4. Implement the business process.

   a. Click [*System actions*] in the Designer's element area and place a [*Script task*] element between the [*Simple*] start event and [*Terminate*] end event on the Process Designer's working area.



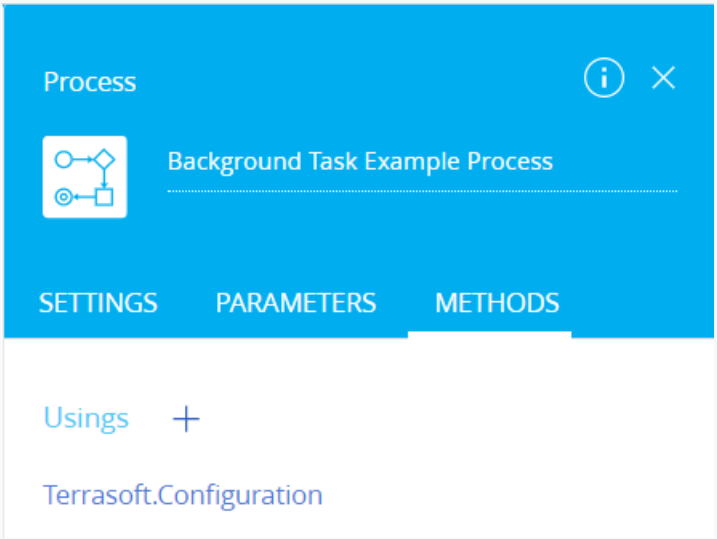
b. Set the name of the [*Script task*] element to "`Create background task`."

c. Add the code of the [*Script task*] element.

The code of the [*Script task*] element

```
var data = new UsrActivityData {
    Title = "Activity created by background task",
    TypeId = ActivityConsts.TaskTypeUId
};
Terrasoft.Core.Tasks.Task.StartNewWithUserConnection<UsrBackgroundActivityCreator, UsrActi

return true;
```

d. Click the + button in the [*Usings*] block on the [*Methods*] tab of the Process Designer and add the `Terrasoft.Configuration` namespace. This is required for the business process to support the implementations of the class for the activity object and class that adds an activity.



5. Click [*Save*] on the Designer's toolbar. A pop-up box will appear after Creatio saves the process.
6. Click [*Publish*] in the box to compile the code of the [*Script task*] element.

Outcome of the example

To **run** the `Background Task Example Process` business process, click [*Run*] on the Process Designer's toolbar.

As a result, the `Background Task Example Process` business process will add the [*Activity created by background task*] record to the list view of the [*Activities*] section list.