Creatio Academy

App development in Creatio

Back-end (C#)

Version 8.0

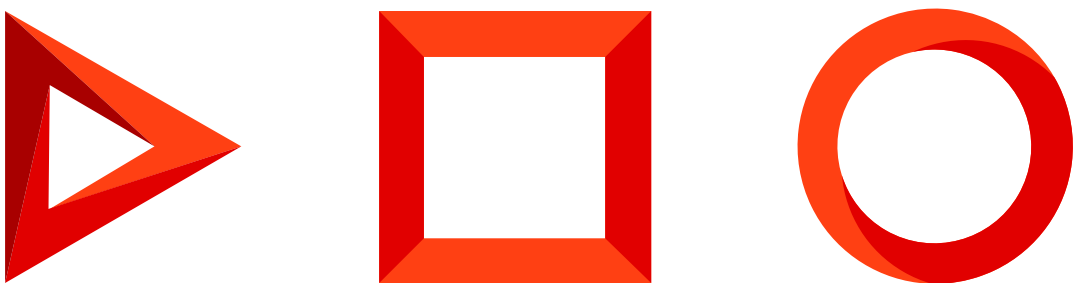

Table of Contents

Back-end (C#)

Beginner

Creatio solution development involves several customization levels depending on your business needs. That said, the core level is an unmodifiable system component, and Creatio development is done on the configuration level.

Back-end level Creatio customization workloads involve working with data, web services, setting up the objects' business logic, etc.

Creatio offers a set of various tools that fulfill these business needs — from developing the [*Source Code*] or [*User Task*] schemas in built-in IDE to creating completely custom projects connected to Creatio as external libraries.

Back-end development workloads

The back-end level involves the following workloads:

- The ORM data model and the methods of working with it.
- Implementing direct access to the database.
- Creating and using Creatio web services.
- Setting up integrations with external services.
- Working with Creatio components and microservices
- Extended setup of business processes and built-in Creatio objects' processes.
- Developing the objects' business logic.

The ORM data model and the direct access to the database

The **ORM data model** in Creatio involves `Terrasoft.Core.Entities` namespace classes:

- `Terrasoft.Core.Entities.EntitySchemaQuery` — builds queries that select database table records, affected by the current user's permissions.
- `Terrasoft.Core.Entities.Entity` — accesses an object that represents a record in the database table.

The back-end core components allow for direct access to the database. However, in most cases we recommend using the object model to access data.

Creatio back-end core's class group from the `Terrasoft.Core.DB` namespace grants **direct access to the database**:

- `Terrasoft.Core.DB.Select` — builds queries that select database table records.
- `Terrasoft.Core.DB.Insert` — builds queries that insert database table records.
- `Terrasoft.Core.DB.InsertSelect` — builds queries that insert database table records.
- `Terrasoft.Core.DB.Update` — builds queries that update database table records.

- `Terrasoft.Core.DB.Delete` — builds queries that delete database table records.
- `Terrasoft.Core.DB.DBExecutor` — allows to build and execute complex database queries, for example, queries with several nested filters, various combinations of JOIN commands, etc.

Web services

Creatio service model includes a base set of web services that allow you to integrate Creatio with third-party applications and systems.

Creatio **system service** examples:

- odata — exchanges data with Creatio via the OData 4 protocol.
- ProcessEngineService.svc — runs Creatio business processes from third-party applications.

Besides the system services, there are **configuration web services** designed to be called from the client part of Creatio.

In addition to the base services, developers can create a **custom web service** to fulfill a particular project's specific business needs.

Third-party services

Creatio core has classes that let you integrate third-party services. For instance, `Terrasoft.Social` namespace lets you integrate various social networks.

Creatio components and microservices

Creatio core has classes that let you work with Creatio components and microservices. For instance, Creatio core's `Terrasoft.Sync` namespace provides classes for the built-in remote repository synchronization mechanism (Sync Engine). Sync Engine lets you create, update and delete `Entity` objects in Creatio based on the data from remote repositories as well as export data to remote repositories. The synchronization process relies on the `SyncAgent` class.

Business processes and built-in objects' processes

You may require back-end development to **set up complex business processes** ([*Script task*] process element) or to create custom **recurrent business process operations** ([*User Task*] configuration schema).

You can set up **built-in objects' processes** not only with no-code tools but also with back-end development. Custom coding allows for a more flexible object behavior configuration that accounts for specific events.

The objects' business logic

Creatio lets you develop the objects' business logic without using event sub-processes. To do this, simply create a class that inherits from the base `Terrasoft.Core.Entities.Events.BaseEntityEventListener` core class and decorate it with the `EntityEventListener` attribute. The attribute has to specify the name of the entity for which to execute the event subscription. Then, you can redefine the relevant events' handler methods.

Back-end development tools and utility capabilities

Source code schema

The back-end development's main capability is creating a [*Source code*] schema in the custom package.

All the schema changes made in the object designer are kept in RAM. To save the changes on the schema metadata level, you need to save the schema. To apply the changes to the database level, you need to publish the schema.

You can develop the [*Source code*] schema in the file system using an IDE of your choice.

Business process configuration

You can add specific back-end logic to a business process using custom code the [*Script task*] element executes.

Similar operations are a common task when working with business processes in Creatio. The [*User task*] element that lets you specify the most fitting action for a particular situation serves this purpose the best.

By default, Creatio includes a wide variety of user tasks. In some situations, a new user task to execute a particular business process is required.

You can create a new user task with the [*User task*] configuration schema type. In its simple implementation the process task partially follows the [*Script task*] process element's logic. However, you can use the existing task multiple times in separate processes. If you make changes to the process task, they will be immediately applied to all processes that use it.

External libraries

You can add custom-made external libraries to the custom package's structure. This allows you to implement complex logic, inheritance mechanisms, and encapsulations when developing a specific project solution.

Project package

The project package is one of the Creatio tools that accelerate Creatio back-end development. This is a package that lets you develop functionality similar to a regular C# project. The new functionality is added to the file content package as a compiled library and a collection of *.cs files. When starting or restarting Creatio, Creatio will collect information on pre-set libraries in the packages and immediately load them.