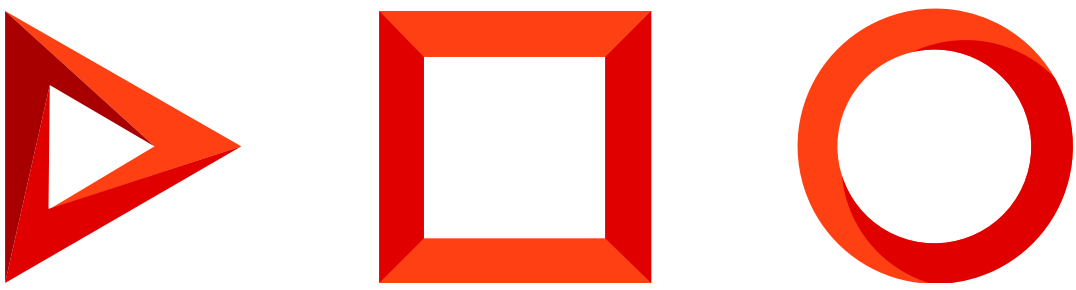
Creatio Academy

Marketplace app development

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

Prepare to develop a Marketplace app **5**
Steps of the Marketplace app life cycle 5
Steps to prepare for the Marketplace app development 5
Steps to develop the Marketplace app **8**
1. Prepare the app development environment 8
2. Develop the app functionality 9
Steps to develop the software solution **10**
1. Prepare the documentation of the software solution concept 10
2. Prepare for the software solution development 11
3. Develop the software solution 11
Requirements for Marketplace app **11**
Localization requirements 11
Functionality requirements 12
Front-end requirements 12
Security requirements 13
Compatibility requirements 13
Metadata requirements 14
Package requirements 14
Demo records requirements 15
Recommendations for front-end development of a Marketplace app **15**
Set up the Marketplace app navigation 16
Preconfigure the Marketplace app 17
Set up secure password storage 20
Develop Marketplace apps based on the base Creatio functionality 20
Add a custom icon of the Marketplace app section 21
Integrate the Marketplace app with Creatio 21
Set up the Marketplace app metadata **22**
Add metadata to a published Marketplace app 22
Add metadata to a new Marketplace app 25
Demo version of the Marketplace app **26**
Create a demo version of the Marketplace app 27
Test the Marketplace app **28**
1. Register a pre-production environment 28
2. Transfer the Marketplace app to the pre-production environment 29
Marketplace app licensing **29**
1. Generate the index of licensed elements 30

© 2023 Creatio. All rights reserved.

2. Define the license type 30
Plagiarism-proof the source code of a Marketplace app **32**
Plagiarism-proof the C# code 33
Plagiarism-proof the JavaScript code 40

© 2023 Creatio. All rights reserved.

Prepare to develop a Marketplace app

Easy

Creatio Marketplace is an online platform where users can easily find and order a ready-made solution for their business. Marketplace enables users to explore, select, and buy partner solutions.

Steps of the Marketplace app life cycle

1. Find an idea for your Marketplace app. Learn more on a separate page: Concept.
2. Develop the Marketplace app. Learn more on a separate page and in the developer documentation: Development, Marketplace app development.
3. Publish and certify the Marketplace application. Learn more on a separate page and in the developer documentation: Publishing, Marketplace app publication, Marketplace app certification.
4. Promote the Marketplace app. Learn more on a separate page: Promotion.
5. Support and grow the Marketplace app. Learn more on a separate page: Support and growth.

Steps to prepare for the Marketplace app development

1. Gain access to the Developer profile.
2. Set up the Developer profile.

1. Get access to the Developer profile

Developer profile is a marketplace section that lets you manage apps and Marketplace services.

To **get access to the Developer profile**:

1. Open the Developer profile registration page.
2. Log in to the Developer profile.

   - If you **have a single account**, use its login credentials to log in to the Developer profile.
   - If you **do not have a single account**, create it and use its login credentials to log in to the Developer profile.

As a result, you will get access to the Developer profile and be able to use tools to develop, publish, and update Creatio Marketplace apps.

2. Set up the Developer profile

Fill out the relevant information about your company in the Developer profile. This information is published as an individual page that is displayed for each of your applications in the Marketplace showcase. Enter this data before you start developing a Marketplace app.

© 2023 Creatio. All rights reserved.

1. Fill out the general information in the Developer profile

1. Open the Creatio Marketplace website.
2. Select [*Developer profile*] in the account menu of the toolbar.
3. Select [*Profile*] on the properties panel.
4. Go to the [*General info*] tab.
5. Fill out the **Developer profile properties**.

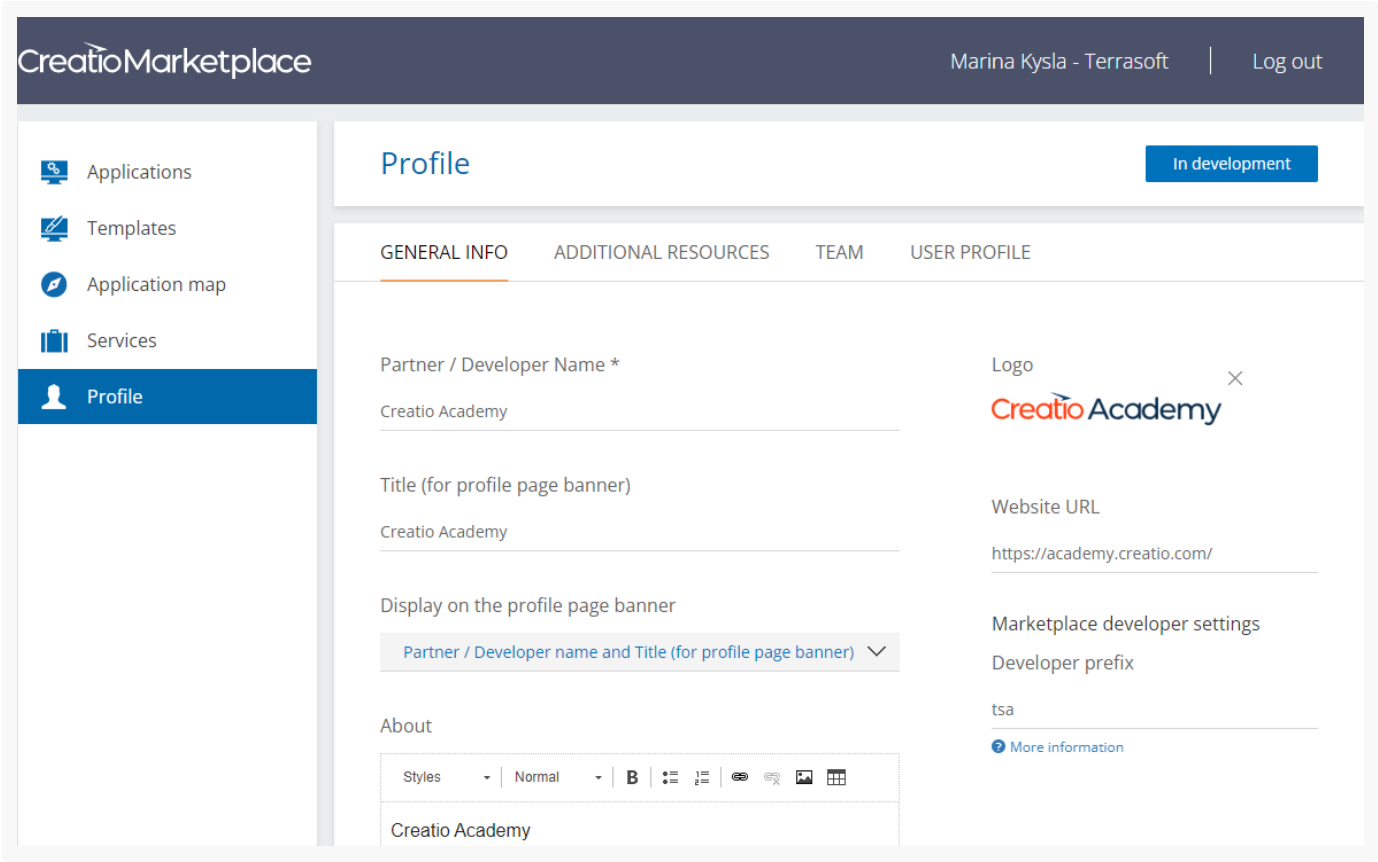
- Select the name of the company on whose behalf to publish the app on Creatio Marketplace in the [*Partner / Developer Name*] property. Required. If the developer is a private entrepreneur, specify their full name here.
- Enter the text to display on the banner of the app developer page on Creatio Marketplace in the [*Title (for profile page banner)*] property.
- Select the information to display on the banner of the app developer page on Creatio marketplace in the [*Display on the profile page banner*] property.

  Available **values**:

  - [*Display nothing*]: display no information.
  - [*Partner / Developer name*], [*Title (for profile page banner)*], [*Partner / Developer name and Title (for profile page banner)*]: display the information specified in the corresponding properties of the developer profile.
- Enter brief information about the activities and competencies of the Marketplace app developer in the [*About*] property.
- Select a continent, region, or country where the Marketplace app developer has the largest presence in the [*Region/Country*] property.
- Fill out the following properties in the [*Business contacts*] block:
  - Enter the person to contact regarding the developed Marketplace apps in the [*Contact person*] property.
  - Enter the phone number of the contact person in the [*Phone*] property.
  - Enter the email of the contact person in the [*Email*] property.
  - Enter the address of the Marketplace app developer in the [*Address*] property.
- Fill out the following properties in the [*Support contacts*] block:
  - Enter the support phone of the Marketplace app developer in the [*Phone*] property.
  - Enter the support email of the Marketplace app developer in the [*Email*] property.
- View the date when the profile of the Marketplace app developer was updated in the [*Update date*] property.
- View the employee who performed the last update of the Marketplace app developer's profile in the [*Update by*] property.
- Upload the corporate logo of the Marketplace app developer to the [*Logo*] property. We recommend using *.png, *.gif, *.jpg images that have white background and are 200px wide.

© 2023 Creatio. All rights reserved.

- Enter the link to the website of the Marketplace app developer in the [*Website URL*] property.
- Fill out the following properties in the [*Marketplace developer settings*] block:
  - Enter the unique ID of the marketplace app developer in the [*Developer prefix*] property. Creatio uses the prefix in the names of custom schemas, packages, objects, and columns in the objects that inherit from base objects. This enables users to identify the functionality created by the Marketplace app developer. Can contain Latin characters and digits. Must be from 3 to 15 characters long.



6. Click [*Save*] to save the changes without sending the Developer profile settings to the Creatio Marketplace support for verification.
7. Click [*Send for verification*] to send the Developer profile settings to the Creatio Marketplace support for verification.

Attention. You can publish the Marketplace app only from a verified profile.

2. Fill out the information about additional Developer profile resources

1. Open the Creatio Marketplace website.
2. Select [*Developer profile*] in the account menu of the toolbar.
3. Select [*Profile*] on the properties panel.
4. Go to the [*Additional resources*] tab.
5. Fill out the **properties of additional developer resources**.

© 2023 Creatio. All rights reserved.

- Upload a background image to display on the Creatio Marketplace page of the Marketplace app developer to the [*Page Banner*] property. We recommend using *.png, *.gif, *.jpg images.
- Upload a presentation of the Marketplace app developer to the [*Upload Presentation*] property. We recommend using *.txt, *.pdf, *.doc, *.docx files.
- Fill out the following properties in the [*Photo*] block:
  - Upload images to display on the Creatio Marketplace page of the Marketplace app developer to the [*Add a new file*] property. We recommend using *.png, *.gif, *.jpg images.
- Enter a link to an external (for example, YouTube) video about the Marketplace app developer to the [*Video*] property.

3. Invite employees to collaborate on Marketplace app development

1. Open the Creatio Marketplace website.
2. Select [*Developer profile*] in the account menu of the toolbar.
3. Select [*Profile*] on the properties panel.
4. Go to the [*Team*] tab. The Developer profile moderator is a person specified on the [*Team*] tab.
5. Invite other employees to collaborate on Marketplace applications.
   a. Enter the email of the developer company employee in the [*Invite a new team member*] field. An invitation to collaborate on Marketplace app development will be sent to this email. The employee must click the link to confirm their participation in joint Marketplace app development.
   b. Click [*Send*] to send the invitation.

Steps to develop the Marketplace app

Easy

Steps to develop the Marketplace app are as follows:

1. Prepare the app development environment.
2. Develop the app functionality.

1. Prepare the app development environment

Use the development environment, a separate website, to develop the Marketplace app. Learn more about the development environment in a separate article: Environments.

You can deploy the app development website in the following **ways**:

- On-site: deploy the Marketplace app on the local servers of the customer.
- Cloud: deploy the Marketplace app on the cloud. The cloud development environment is a bundle of Creatio base products (Sales Creatio, Marketing Creatio, and Service Creatio). This lets you use any base Creatio functionality when developing Marketplace apps.

Learn more about the advantages and limitations of the Marketplace app development website in a separate

© 2023 Creatio. All rights reserved.

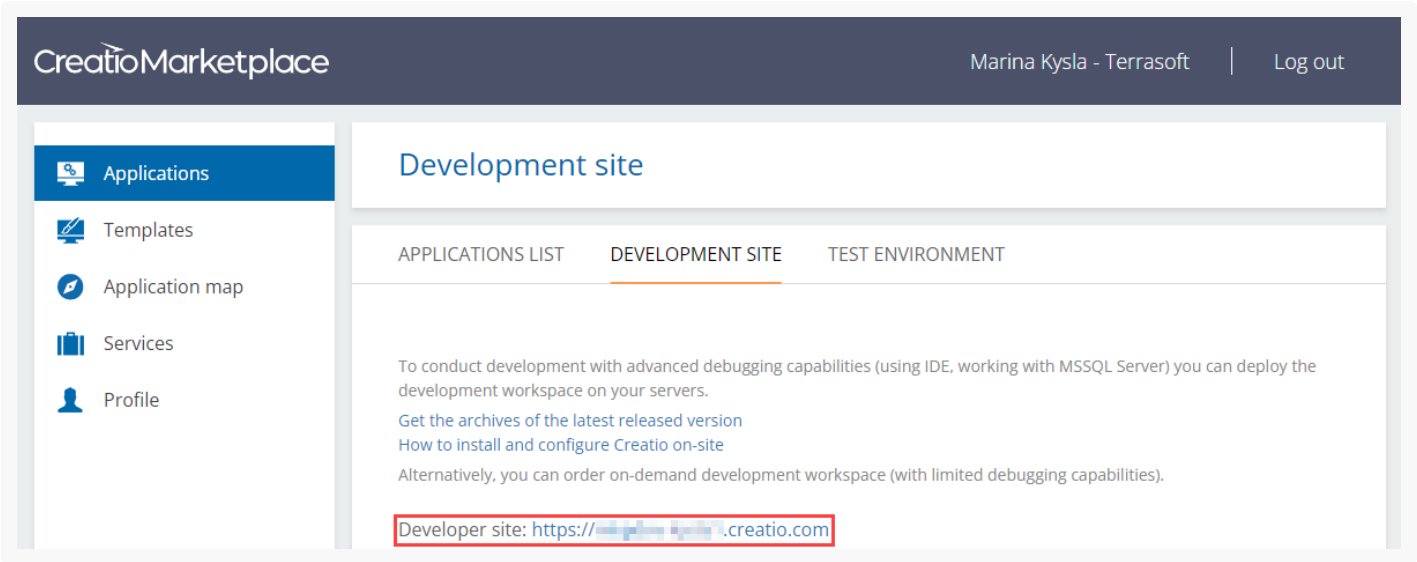
article: Creatio main application.

To **prepare a local development environment**, contact Creatio Technical support (support@creatio.com) to retrieve the binary files of the Creatio app.

To **prepare a cloud development environment**:

1. Open the Creatio Marketplace website.
2. Select [*Developer profile*] in the account menu of the toolbar.
3. Click [*Applications*] → [*Development site*] on the properties panel.
4. Enter the email address to send the link to the development environment in the [*Order cloud development site*] field.
5. Click [*Send*] to request a cloud development environment to the Creatio Technical support.

Creatio Technical support provides a free cloud development environment after processing the request. The user receives an email with a link to the development environment app to the email address from the [*Order cloud development site*] field. The link to the development environment app is displayed on the [*Development site*] tab of the [*Applications*] section in the Developer profile.



2. Develop the app functionality

When developing a Marketplace app, the developer can create any additional elements of the system, configuration, business processes, mobile app, or integration with software, service, or external app. The Marketplace app can be an improvement of any base Creatio product.

A Marketplace app is a set of packages. It lets you extend the functionality of the base Creatio products. Learn more about packages in a separate article: Packages basics.

The number and composition of Marketplace app packages depend on the complexity of the implemented functionality. For example, to invoke a third-party service from an existing record page, replace one or more configuration element schemas, and group them in a single custom package. To add a Creatio section, implement multiple dozen configuration element schemas.

The Marketplace app development, as well as the Creatio instance, is performed using development tools and based on the basic principles of software design, in particular, the **Don’t repeat yourself (DRY)** principle.

© 2023 Creatio. All rights reserved.

Creatio architecture implements this principle via **package dependencies**. Learn more about the DRY principle on the Wikipedia. Learn more about package dependencies in a separate article: Packages basics.

Learn more about tools for Marketplace app development in separate guides: Development tools, Marketplace app development.

After that, you can move on to publishing the Marketplace app. Learn more in a separate article: Steps to publish the Marketplace app.

Steps to develop the software solution

Beginner

Software solution is an app developed based on an out-of-the-box Creatio product. The app fulfills a need in a specific industry and has its own business value. Learn more in the Creatio Marketplace partner program. Learn more about Marketplace app types in a separate article: Marketplace app types.

The **steps** to develop the software solution are as follows:

1. Prepare the documentation of the software solution concept.
2. Prepare for the software solution development.
3. Develop the software solution.

1. Prepare the documentation of the software solution concept

Before you begin the development, analyze market and apps on Creatio Marketplace to understand the user needs. Prepare the documentation of the software solution concept based on this analysis.

Prepare the following **documents** based on the results of the market analysis:

- market research
- roadmap

The market research document must contain the following **data**:

- profiles of the software solution's target audience
- software solution value for the target audience
- functional comparison of the software solution with competitive Marketplace apps
- price comparison of the software solution with competitive Marketplace apps

The roadmap document must include the following **data**:

- describe the functionality you are going to provide to users as part of the future software solution releases
- specify the preliminary release dates (quarter or month) of the software solution functionality

Prepare the roadmap for the next 12 months.

After you prepare the documentation, contact Creatio Marketplace support to discuss the software solution concept.

© 2023 Creatio. All rights reserved.

2. Prepare for the software solution development

To prepare for the software solution development, take the steps in a separate article: Prepare to develop a Marketplace app.

3. Develop the software solution

> **Note.** You can start developing the software solution only after Creatio Marketplace support approves the concept.

To develop the software solution, take the steps in a separate article: Steps to develop the Marketplace app.

The software solution development has the following **special features**:

- Only organizations that have the Creatio Partner status can develop and publish software solutions.
- Software solution must fulfill a need in a specific industry and have its own business value.
- Software solution must be based on a single base product.
- Software solution can use only personal licenses. Learn more in a separate article: Marketplace app licensing.
- Use objects as licensed elements of a software solution. You can use 3-5 licensed objects. Learn more in a separate article: Marketplace app licensing.

Requirements for Marketplace app

 Medium

Make sure the Marketplace app meets the following **groups of requirements**:

- localization requirements
- functionality requirements
- front-end requirements
- security requirements
- compatibility requirements
- metadata requirements
- package requirements
- demo records requirements

Localization requirements

- English version is mandatory.
- Translations to other languages are optional.
- Translate the following **elements** if users can use the Marketplace app in other languages:

© 2023 Creatio. All rights reserved.

  - app UI, for example, sections, details, fields, dashboards, notifications, etc.
  - localizable strings
  - bound data, for example, lookup values, demo data, system settings, etc.
- Translation of system data is optional. For example, headings and elements of business processes, headings of objects and schemas, and log data.
- The Marketplace app package must contain the localizations (Developer profile → [*General information*] tab → [*Localization*] property). Learn more in a separate article: Manage UI languages.

Functionality requirements

- Use the Marketplace app to solve a business task that cannot solve with the basic development tools, or this solution is time-consuming or optimizes the business task solution.
- Do not duplicate base Creatio functionality in the Marketplace app. If the Marketplace app extends base functionality, use existing Creatio functionality.

  Examples that comply with the requirement:

  - Extend the [*CTI panel*] tab functionality of the base communication panel that use for the call management during the new telephony connector development.
  - Extend the [*Chats*] tab functionality of the base communication panel used for the chat management during the new chat connector development. You need not develop a UI that displays chats.
  - Develop a custom contact action and add it to the [*Actions*] button menu of the contact page toolbar. Replacing the CSS styles and developing the new button in another place on the contact page are unnecessary.

Front-end requirements

- Requirements for the **icon** of the Marketplace app section are as follows:
  - Style: a flat white image without small elements or lines, background must be transparent.
  - Format: *.png or *.svg.
  - Size: 38x38 pixels or smaller. Note that the required icon size without border margins is 32x32 pixels.
  - We recommend selecting an icon of the Marketplace app section from the section icon library.
- Capitalize the first word of the UI element name. Using the capital letter for each word of the interface element name is unnecessary.

  View an example of a custom field name below.

  Example of a custom field name

✅	❌
[*Serial number*]	[*Serial Number*]

© 2023 Creatio. All rights reserved.

- Use the basic page and navigation elements of the Creatio instance for the Marketplace app interface. For example, fields, buttons, tabs, etc.
- Enable the user to use the base Creatio functionality in its original form, whether implementing the new functionality or extending the basic Creatio functionality in the Marketplace app.
- Follow these **principles** when you develop the Marketplace app functionality:
  - **Clarity**. Develop a clear and logical interface to help users orient in the Marketplace app. Name UI elements as calls to action, for example, add, upload, etc.
  - **Consistency**. Design the app functionality so that it solves a specific business problem in its entirety as opposed to improving the experience of an individual functional or organizational role.
  - **Specificity**. Let the user use the additional technology implemented in the Marketplace app, for example, keyboard and mouse input, touch input, etc.

Security requirements

- The Marketplace app must not decrease Creatio performance.
- The Marketplace app must follow information security best practices. Use the recommended API Creatio calls in the Marketplace app. Ensure that Open Web Application Security Project (OWASP) Top 10 vulnerabilities are absent in the Marketplace app. Do not use components with known vulnerabilities. Learn more about vulnerabilities on the official OWASP website.
- Use the law way to get data access, and do not send data from the Creatio instance in the Marketplace app. Describe the data sending on the Marketplace app page and restrict the data sending before the user authorization when you integrate with external apps.
- Do not use the anonymous web services without an alternative authorization type in the Marketplace app. Learn more in a separate article: Custom web services.
- Do not use external libraries from unknown sources in the Marketplace app.
- Use package install scripts to execute CRUD operations with data as part of the package installation process instead of SQL scripts. Install scripts execute operations with data using the `Entity` class. Learn more in a separate article: Customize delivery process.

Compatibility requirements

- Implement compatibility with the MS SQL and PostgreSQL databases for new Marketplace apps (the Developer profile → the [*Installation and setup*] tab → the [*Compatibility*] block → the [*DBMS compatibility*] property). Implementing compatibility with the Oracle database for Marketplace apps is optional.
- The Marketplace app must support .NET Framework (Developer profile → [*Installation and setup*] tab → [*Compatibility*] block → [*Platform*] property). .NET Core support is optional.
- Implement compatibility with the latest Creatio version of the Marketplace app.
- Implement compatibility with the existing Marketplace app updates with the previous app version.
- Describe all implemented corrections and modifications when you release the new version of the existing Marketplace app (the Developer profile → the [*Packages and updates*] tab → the [*Whats new?*] property).
- The Marketplace app must be compatible with base Creatio products (Developer profile → [*Installation and setup*] tab → [*Compatibility*] block → [*Product compatibility*] property) and versions (Developer profile →

© 2023 Creatio. All rights reserved.

[*Installation and setup*] tab → [*Compatibility*] block → [*Version compatibility*] property). If you use the Studio Creatio platform as a base product, the Marketplace app is compatible with all base Creatio products.

- Support all versions of the Marketplace that the users use with actual Marketplace app licenses or the paid support package.
- Use the file management API in your Marketplace app instead of direct SQL queries. The API lets you access files stored in the database and external storage. Learn more about the API in a separate article: API for file management.

Metadata requirements

- Do not use the base metadata template generated automatically.
- Make sure the [*Please name your application*] field value entered when creating the app matches the app name on Creatio Marketplace.
- Select any icon in the [*Select icon and color*] block or use custom icon. Make sure the icon matches Marketplace app functionality.
- Select any icon color in the [*Select icon and color*] block.
- If you use custom icon for the Marketplace app, make sure the icon meets the following requirements:
  - Style: a flat white image, background must be transparent.
  - Size: 32x32 pixels.
  - Format: *.svg.

Package requirements

- Release the Marketplace app as a single Creatio app and/or setting in an external app.
- Fill out the Marketplace app metadata before delivery. Learn more in separate articles: Set up the app metadata, Add metadata to a published Marketplace app.
- Convert simple packages and project packages that contain Marketplace app functionality to assembly packages. To do this, follow the instructions in a separate article: Package conversion.
- Deliver the implemented customization as a package. Do not customize Creatio beyond the configuration, for example, create or change files in the Creatio file system.
- The Marketplace app functionality must include the unique developer ID (prefix) (Developer profile → [*General info*] tab → [*Marketplace developer settings*] block → [*Developer prefix*] property). Do not develop the Marketplace app without a prefix or with the `Usr` prefix.
- Upload the Marketplace app as a *.zip archive (Developer profile → [*Packages and updates*] tab → [*Add file*] property) that contains one or more *.gz packages.
- Add all .*gz package archives in the root folder of the *.zip archive. Do not use subfolders.
- Name the *.zip archive using the following template: `Marketplace-app-name-Package.Version.Number`.

  View an example that names the *.zip archive of the custom Marketplace app below.

  Example of a custom *.zip archive

© 2023 Creatio. All rights reserved.

✅	❌
`Salesloft-connector-1.15.zip`	`Salesloft_connector_01.06.2022_10_16.zip`

Demo records requirements

- The number of section records must provide enough data for nice-looking analytics (20 on average).
- The first three section records must be showcase records.
- You can add up to 1000 demo records to the corresponding database table.
- Record data must be logically consistent.
- Demo data must be relevant for the target audience and region where you are going to distribute the app (addresses, names, phone numbers).
- Each section record must be unique. The record variety must be reflected in folders and analytics.
- Use records that are logically related to records of other sections. We recommend creating record chains instead of going from section to section.
- Bind the demo records to base demo data for data integrity.
- Add demo records not only to sections, but also feed records, reminders, notifications, emails, comments, likes, etc.
- Demo data must have a positive message.
- Demo data must be in English.
- You can fill out lookups using either softkey or demo data.
- Lookup records must be unique.
- If Creatio must add the corresponding activities when you add demo records, verify that the records are present in the activity schedule.
- If demo data must include dashboard analytics, verify the charts in each section and take the update date into account.
- Some sections use preconfigured filters by owner and period, for example [*Activities*], [*Invoices*]. Ensure the section list includes demo records that match the filter conditions.
- Implement a demo mode for connectors. You can do it using pre-configured test users or a simulated integration that showcase how the connector operates.

Learn more about the demo records in a separate article: Demo version of the Marketplace app.

Recommendations for front-end development of a Marketplace app

Beginner

The negative user experience in installing, configuring, and starting work with the app is one of the main factors that significantly reduce the probability of purchasing an app after a trial. Potential customers often perceive apps

© 2023 Creatio. All rights reserved.

negatively because they require complex and time-consuming setups.

Follow the recommendations for the front-end development of a Marketplace app and requirements from a separate article: Requirements for Marketplace app, to **avoid a negative user experience**.

Recommendations for the front-end development of a Marketplace app:

- set up the Marketplace app navigation
- preconfigure the Marketplace app
- develop the Marketplace app based on the base Creatio functionality
- add a custom icon of the Marketplace app section

Set up the Marketplace app navigation

The Marketplace app navigation streamlines the following **actions**:

- searching for the required Marketplace app
- opening the required Marketplace app functionality
- using the functionality

To set up the marketplace app navigation, take the following **steps**:

1. Add the app section to a workplace.
2. Add the link to the app setup page to the System Designer.
3. Configure the transition to the settings from the section actions menu.

1. Add the app section to a workplace

It is the required step for the Marketplace app with the new Creatio section. To **add the app section to a workplace**, follow the instruction in a separate article: Set up workplaces.

View an example that adds the custom [*Chats*] section to a workplace below.

Example that adds the custom [*Chats*] section

	
Add the custom [*Chats*] section to the [*Sales*] and/or [*Marketing*] workplace.	Create a single workplace and add the [*Chats*] custom section.
	Do not add the custom section to any workplaces.

2. Add the link to the app setup page to the System Designer

Creatio lets you add the link to the app setup page to the settings blocks of the System Designer specified in the table below.

© 2023 Creatio. All rights reserved.

Settings blocks of the System Designer

Settings block	Purpose	How to use for a Marketplace app
[*Import and integration*]	Use the block to transition to any integration settings page and import the records.	Block can use for connectors.
[*System setup*]	Use the block to transition to the page and configure the system behavior.	Block can use for transition to the wizard that manages the add-on behavior and vertical solution logic.

3. Add the link to the app setup page to the section actions menu

Users understand they need to perform additional setup after they review the Marketplace app functionality on the corresponding Creatio page. To enable users to **perform the required setup** add the corresponding item to the ,[*Actions*] button menu on the section page and/or record section page.

View an example that adds the custom setup page with authentication settings for a third-party app in the table below. In this example, custom [*Integrations*] Creatio section interacts with a third-party app.

Example that adds the custom setup page with authentication settings for a third-party app

	
Add the [*Set up connection to the third-party app*] item to the [*Actions*] button menu of the custom [*Integrations*] section.	Implement a setup page accessible only via a URL.

Preconfigure the Marketplace app

Use the following Creatio **no-code tools** to preconfigure the Marketplace app:

- [*System settings*] section
- [*Lookups*] section
- user profile
- dedicated setup page

Preconfigure the app in the [*System settings*] section

Learn more about the [*System settings*] section in user documentation: Manage system settings.

Follow these **recommendations** when you add a new system setting:

- Use short system setting names. Ensure the system setting name represents its behavior and specifies the relevant functionality.

© 2023 Creatio. All rights reserved.

- Group system settings into folders. Grouping lets a user find the necessary system setting without knowing its name. Learn more about folders in user documentation: Folders.
- Add system setting descriptions. Description lets a user understand the purpose and usage of the system setting.

Examples that use short and clear names for system settings

✅	❌
`"Base currency"`	`"The currency that is used in the app by default"` Not recommended because it is too long.
`"Order status by default"`	`"Order status"` Not recommended because it does not represent the behavior.
`"Automatic start of the incident management process"`	`"Automatic launch of the process"` Not recommended because it does not specify the relevant functionality.

Preconfigure the app in the [*Lookups*] section

Learn more about the [*Lookups*] section in user documentation: Manage lookup values. To **create a lookup**, follow the instruction in user documentation: Create new lookups.

Lookup use cases

© 2023 Creatio. All rights reserved.

Business goal	Lookup	Lookup description	Lookup values

Specify the index of data to select	[*Document statuses*]	Contains an index of document statuses during the workflow.	`"Active"`
			`"Inactive"`
			`"Draft"`

Populate the field automatically during integration	[*Lead channels*]	The lookup contains a list of resource types to receive the lead.	`"Social accounts"`
			`"Search-based advertising"`
			`"Email"`

Customize business logic	[*Case notification rule*]	Contains an index of rules for sending notifications to the contact about the progress of their case.	`"Send immediately"`
			`"Send after a delay"`
			`"Disabled"`

Specify a system index	[*Webitel users*]	Contains an index of Webitel users.	
Display the functionality in different Creatio sections	[*Gantt chart configuration*]	Contains an index of Gantt chart settings.	

Preconfigure the app in the user profile

Individual settings of a specific user are listed in the user profile. For example, individual login/password for authorization in a third-party app or preferences for using certain product functionality.

Examples that use the settings in a user profile:

- **Integration connection settings**.
  - The ID of the telephony operator or the phone number that enables the user to use the telephony connector.
  - Mail service connection settings for the user.
- **System behavior settings for the user**.
  - Notification settings.
  - Default language the user uses in the business card scanner.

© 2023 Creatio. All rights reserved.

Preconfigure the app in the dedicated setup page

The single settings page is a user-friendly but difficult-to-implement way to manage the Marketplace app settings.

The users can open the dedicated setup page in the following **ways**:

- corresponding section
- System Designer

We recommend using a dedicated page for Marketplace app settingsin the following **cases**:

- The setup has a specific sequence.
- The setup process requires the population of settings and lookups.
- The setup process requires additional actions besides filling in the settings and lookups.
- The setup process includes both the preconfiguration and additional setup as part of working with the Marketplace app.

Set up secure password storage

Creatio implements methods for secure password storage. These methods let you store user passwords and secret connection keys for external systems, for example, Secret key, domain token, etc.

To **store passwords securely**, use the following data types:

- `Encrypted string` type for the [*System settings*] section
- `Encrypted string` type for string fields of Creatio objects

> **Note.** The values of `Encrypted string` fields are not available in the front-end.

Develop Marketplace apps based on the base Creatio functionality

Development based on the base Creatio functionality implies the correct population of fields when creating leads via Marketplace apps. Creating leads when the Creatio is integrated with messengers, chats, landing pages, or social networks is one of the typical integration tasks. When a lead is created automatically, it is necessary to keep the correct logic of filling the fields on the lead page, including data about lead generation channels. The developer of the Marketplace application lets users provide the ability to use end-to-end analytics on leads. Match the name of the field to its purpose.

Examples of lead field names

© 2023 Creatio. All rights reserved.

Name	How to use
[*Registration method*]	Lead registration **methods**: • created automatically • added manually • an incoming call or email • landing page • case
[*Channel*]	Lead origin **types**: • web • social networks • offline advertisement • event • recommendation or personal contact
[*Source*]	The name of the specific lead origin resource, for example, Twitter, Google, etc.
[*Transition website*]	The website from which the user clicked on the landing page resulted in a registered lead in the app. The field is non-editable and populated automatically when a lead is received from the landing page.

Add a custom icon of the Marketplace app section

Make sure your icon for the custom section of the Marketplace app meets the requirements from a separate article Requirements for Marketplace app.

You can obtain the appropriate custom section icon in the following **ways**:

- Use the section icon library available via the link
- Use a free service that searches for and converts flat icons.

Integrate the Marketplace app with Creatio

Creatio enables a range of methods for integration with the Marketplace app. Learn more about the available Creatio integration options in a separate article: Integration options.

Authentication is required to access Creatio, as with third-party apps. For app integration, we recommend using the **Forms authentication** method implemented using the `AuthService.svc` web service. Learn more about the authentication types Creatio supports in a separate article: Authentication.

© 2023 Creatio. All rights reserved.

Attention. Do not use the anonymous web services without implementing an alternative authorization type in the Marketplace app. Learn more in a separate article: Custom web services.

Interact with Creatio via the **DataService** data service or **OData** protocol. Learn more in a separate guide: Data services. Creatio also lets you use the **iframe** HTML element for Marketplace apps. Learn more in a separate article: HTML-element <iframe>. You can use the **Web-To-Object** mechanism for simple integrations. Learn more in a separate article: Web-To-Object.

Set up the Marketplace app metadata

Medium

Creatio version 8.0.2 and later can store Marketplace app metadata.

In this case, **metadata** is app details stored within the app itself regardless of an environment.

The Marketplace app metadata stores the following **parameters**:

- icon ([*Select icon*] field)
- icon color ([*Color*] field)
- name ([*Give it a name*] field)
- description ([*Describe what it's for (optional)*] field)
- code ([*Application code*] field)

Set up the parameter values within the Marketplace app metadata in the Application Hub. Learn more about the Application Hub in the user documentation: Create a custom app. Creatio lets you add metadata to a published or new Marketplace app.

To set up or change app metadata, follow the instructions in the user documentation for generic Creatio apps: Set up the app metadata. You can fill out the parameters only when you create an app, but you can change them at any moment. The code is generated on app creation and cannot be changed later.

Note. Add metadata to Marketplace apps developed for version 8.0.1 and earlier, i. e., apps without metadata.

Make sure the app metadata meets the requirements listed in a separate article: Requirements for Marketplace app.

Add metadata to a published Marketplace app

The procedure to add metadata to a published Marketplace app depends on the number of packages that contain the app functionality.

© 2023 Creatio. All rights reserved.

Add metadata to a published Marketplace app that has a single package

1. Install the Marketplace app into Creatio version 8.0.2 or later. To do this, follow the instruction in the user documentation: Install apps from the Marketplace.
2. Unlock the package that contains the Marketplace app functionality using the Clio utility. To do this, follow the instructions in the utility documentation on GitHub.
3. Unlock the Marketplace app in the Application Hub. To do this, execute the SQL script provided below in the database.

SQL script template

Microsoft SQL

```
UPDATE SysInstalledApp
SET Maintainer = (SELECT TextValue FROM SysSettings
    JOIN SysSettingsValue ON SysSettingsValue.SysSettingsId = SysSettings.Id
    WHERE Code = 'Maintainer'
    AND SysSettingsValue.SysAdminUnitId = 'A29A3BA5-4B0D-DE11-9A51-005056C00008')
WHERE name='SomeMarketplaceApplicationName'
```

Oracle

```
UPDATE "SysInstalledApp"
SET "Maintainer" = (SELECT "TextValue" FROM "SysSettings"
    JOIN "SysSettingsValue" ON "SysSettingsValue"."SysSettingsId" = "SysSettings"."Id"
    WHERE "Code" = 'Maintainer'
    AND "SysSettingsValue"."SysAdminUnitId" = '{A29A3BA5-4B0D-DE11-9A51-005056C00008}')
WHERE "Name" = 'SomeMarketplaceApplicationName'
```

PostgreSQL

```
UPDATE "SysInstalledApp"
SET "Maintainer" = (SELECT "TextValue" FROM "SysSettings"
    JOIN "SysSettingsValue" ON "SysSettingsValue"."SysSettingsId" = "SysSettings"."Id"
    WHERE "Code" = 'Maintainer'
    AND "SysSettingsValue"."SysAdminUnitId" = 'A29A3BA5-4B0D-DE11-9A51-005056C00008')
WHERE "Name" = 'SomeMarketplaceApplicationName'
```

`SomeMarketplaceApplicationName` is the name of the installed package that contains the Marketplace app functionality.

4. Change the Marketplace app metadata. To do this, follow the instruction in the user documentation: Set up the

© 2023 Creatio. All rights reserved.

app metadata.

5. Lock the package that contains the Marketplace app functionality. To do this, execute the SQL script provided below.

SQL script template

Microsoft SQL

```
UPDATE SysInstalledApp
SET Maintainer='SomeMarketplaceApplicationMaintainer'
WHERE Name='SomeMarketplaceApplicationName'
```

Oracle

```
UPDATE "SysInstalledApp"
SET "Maintainer" = 'SomeMarketplaceApplicationMaintainer'
WHERE "Name" = 'SomeMarketplaceApplicationName'
```

PostgreSQL

```
UPDATE "SysInstalledApp"
SET "Maintainer" = 'SomeMarketplaceApplicationMaintainer'
WHERE "Name" = 'SomeMarketplaceApplicationName'
```

`SomeMarketplaceApplicationMaintainer` is the publisher of the package that contains the Marketplace app functionality.

`SomeMarketplaceApplicationName` is the name of the installed package that contains the Marketplace app functionality.

6. Export the package that contains the Marketplace app functionality. To do this, follow the instruction in a separate article: Application Hub.
7. Upload the *.zip archive that contains the Marketplace app to the developer workspace.
8. Send the Marketplace app for verification. After the Creatio Marketplace support verifies the Marketplace app, it is automatically published on Creatio Marketplace. After the Marketplace app is installed into Creatio, the app will display the specified parameter values.

Add metadata to a published Marketplace app that has multiple packages

If you install a Marketplace app that comprises multiple packages and has no metadata into Creatio version 8.0.2 or later, Creatio generates metadata automatically and saves it to an arbitrary package. If Marketplace apps that have different metadata uses the same base packages, the package installation ends with an error.

© 2023 Creatio. All rights reserved.

Attention. Use the target package that contains metadata only in one Marketplace app.

To **add metadata to a published Marketplace app that has multiple packages**:

1. Choose a unique package to store the Marketplace app metadata. Do not use this package in other Marketplace apps.
2. Prepare the unique package before installing it into Creatio version 8.0.2 or later.
   a. Install the Clio utility (performed once). To do this, follow the instructions in the utility documentation on GitHub.
   b. Unpack the *. zip archive that contains the Marketplace app packages.
   c. Run the `Extract package` Clio command to unpack the unique package. Learn more in the utility documentation on GitHub.
   d. Download the metadata file and add it to the `...\[Path to package folder]\Files\` directory.
   e. Add the Marketplace app packages to the `Packages` file property. You can modify other property values in the Application Hub.
   f. Run the `Compress package` Clio command to compress the changed unique package into a *.gz package archive. Learn more in the utility documentation on GitHub.
   g. Compress the changed unique package and other Marketplace app packages into a *.zip archive.
3. Install the Marketplace app that contains metadata into Creatio version 8.0.2 or later. To do this, follow the instruction in the user documentation: Install apps from the Marketplace.
4. Unlock the packages that contain the Marketplace app functionality using the Clio utility. To do this, follow the instructions in the utility documentation on GitHub.

   Attention. If at least one package is locked, an additional package is created automatically when the metadata is changed.
5. Repeat steps 3–8 of the instruction to add metadata to a published Marketplace app that has a single package.

Add metadata to a new Marketplace app

The procedure to add metadata to a new Marketplace app depends on the number of packages that contain the app functionality.

Add metadata to a new Marketplace app that has a single package

To **add metadata to a new Marketplace app that has a single package**, follow the instruction in the user documentation: Set up the app metadata.

Add metadata to a new Marketplace app that has multiple packages

Creatio version 8.0.2 and later lets you set a target package and create metadata in it before you start developing a new Marketplace app that contains multiple packages.

© 2023 Creatio. All rights reserved.

Attention. Before you add metadata to a new Marketplace app that contains multiple packages, make sure that the Marketplace app packages are unlocked and do not contain any Marketplace app metadata in Creatio.

To **add metadata to a new Marketplace app that has multiple packages**:

1. Choose a unique package to store the Marketplace app metadata. Do not use this package in other Marketplace apps.
2. Add a lookup based on the `SysInstalledApp` object. To do this, follow the instruction in a separate article: Create new lookups.
3. Add a lookup record and fill out the **record properties**:
   - Enter the app name to display in the Application Hub in the [*Name*] property.
   - Enter the app code in the [*Code*] property.
   - Specify the Marketplace app developer in the [*Maintainer*] property. The value of the property must match the value of the [*Publisher*] (`Maintainer` code) system setting.
   - Enter brief information about the Marketplace app in the [*Description*] property.
   - The Marketplace app icon in the [*App icon*] property is populated automatically.
   - The background of the Marketplace app icon in the [*Icon background color*] property is populated automatically.
4. Add a lookup based on the `SysPackageInInstalledApp` object. To do this, follow the instruction in a separate article: Create new lookups.
5. Add the Marketplace app packages to the lookup and fill out the **record properties**:
   - Select the Marketplace app package in the [*Package*] property.
   - Select the record from the previous step in the [*Installed application*] property.
   - Set the [*Primary*] property to `true` for the unique package. Set the property to `false` for other Marketplace app packages.
6. Repeat steps 4, 6–8 of the instruction to add metadata to a published Marketplace app that has a single package.

Demo version of the Marketplace app

Medium

A **demo version** of a Marketplace app is a set of packages that contain sample data, and pre-configured functionality, such as test integration for the connector. A demo version enables the user to use the Marketplace app immediately after installation without additional setup. The demo version availability minimizes the time for preparing the presentation to the users and ensures a sufficient user satisfaction level. The **purpose** of the demo version is to showcase how the Marketplace app operates in Creatio.

The Marketplace app must contain the following **types of sample data**:

© 2023 Creatio. All rights reserved.

- **Softkey data** are samples of the base Creatio objects or sections that enable users to get to know the product quicker. Add softkey data to the package that contains the Marketplace app functionality. Softkey records must be in English and have the `(sample)` suffix at the end of the name.
- **Demo data** (demo records) are samples of Creatio object records stored in an individual package. This package must depend on the package that contains the Marketplace app functionality.
- **Showcase records** are section records that contain the maximum volume of demo data. The first three records in a section must be showcase records.

Before you create a demo version, ensure that you follow the requirements listed in a separate article: Requirements for a Marketplace app.

Create a demo version of the Marketplace app

1. Deploy a build of the latest Creatio version that contains demo data.

   You can order a build in the following **ways**:

   - on the trial order page via the link
   - by contacting the Creatio support (support@creatio.com)

2. Install a package that contains the Marketplace app functionality. To do this, follow the instruction in a separate article: Install apps from the Marketplace.
3. Create a dedicated app for demo data. To do this, follow the instruction in a separate article: Create a custom app. Use the following package name template: `PackageName_Demo`. For example, `LabReports_Demo`. Store this app in an individual package.
4. Add a package that contains the Marketplace app functionality to the package dependencies.
5. Add demo data to sections and lookups.

   - If you use Creatio **on-site** or in the **cloud**, add demo data on behalf of the `John Best` user (`76929f8c-7e15-4c64-bdb0-adc62d383727` contact ID).
   - If you use Creatio **trial**, add demo data on behalf of the Application Hub user that ordered the trial.

   To **add demo data to Creatio on-site**:

   a. Add a filter `id = 76929f8c-7e15-4c64-bdb0-adc62d383727` to the [*Contacts*] section and fix the contact name.

   b. Add a Creatio user and connect them to the contact from the previous step. To do this, follow the instruction in a separate article: Add users.

   c. Log in to Creatio as the created user and add demo data on their behalf.

6. Bind Marketplace app demo data to Creatio demo data. Bindings is recommended for data integrity. Showcase records must be bound to the `Alexander Wilson` contact (`98dae6f4-70ae-4f4b-9db5-e4fcb659ef19` ID) and `Alpha Business` account (`98dae6f4-70ae-4f4b-9db5-e4fcb659ef19` ID) where appropriate.
7. Configure the sorting of demo records so that the showcase records are displayed at the top of the section list. If you have not selected or filled out the showcase records yet, configure the sorting before populating the showcase records.

© 2023 Creatio. All rights reserved.

8. Bind the list settings, sorting, app demo data, and needed base demo data to the package. To do this, follow the instruction in a separate article: Packages basics.
9. Download a demo version of the Marketplace app. To do this, follow the instruction in a separate article: Application Hub.
10. Verify the completeness and correctness of bound data.
    a. Order a Creatio trial via the link.
    b. Install packages that contain the Marketplace app functionality. To do this, follow the instruction in a separate article: Transfer packages.
    c. Install the package that contains the Marketplace app demo version. To do this, follow the instruction in a separate article: Transfer packages.
11. Add the package that contains the Marketplace app demo data (the [*Package with demo data*] property of the [*Packages and updates*] tab) to the Marketplace app page in Developer profile.
12. Click [*Send for verification*] to send the package that contains the demo version to Creatio Marketplace support for verification.

Test the Marketplace app

Medium

Test the software solution before you publish it to verify its operability. Test the solution in the pre-production environment.

Developing and implementing a Marketplace app is the same as the Creatio app. Learn more about developing and implementing a Creatio instance in a separate article: Delivery management process.

Perform the testing as part of the Marketplace app delivery. The **steps** to deliver the app are as follows:

1. Register a pre-production environment.
2. Transfer the Marketplace app to the pre-production environment.

1. Register a pre-production environment

Use the pre-production environment to test the Marketplace app functionality.

To **register a testing environment app**, order a trial of the product app that is selected as a development environment (i. e. the app specified on the [*Development site*] tab).

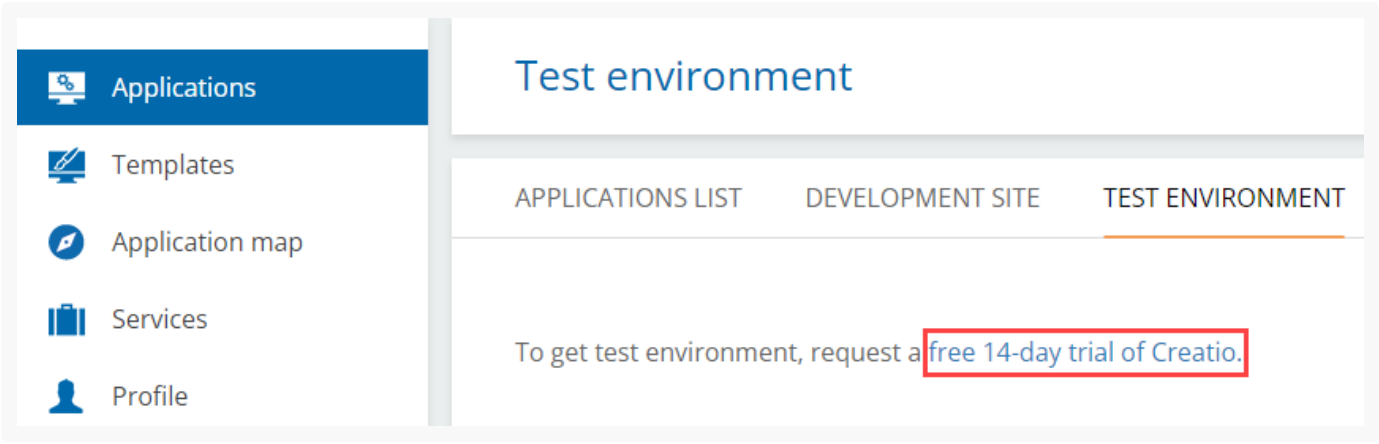
You can order a trial in the following **ways**:

- on the trial order page via the link
- using the Developer profile

To **order a trial using the Developer profile**:

1. Open the Developer profile.
2. Click [*Applications*] → [*Test environment*] on the properties panel.
3. Click the link to order a trial.

© 2023 Creatio. All rights reserved.



4. Select the relevant product on the opened page and click [*Try it free*].
5. Log in to an existing Creatio account or create a new account.

As a result, the pre-production environment is ready to use. The user receives an email with the activation/deactivation dates of the testing environment app and a link to the testing environment app to the email address specified when registering.

2. Transfer the Marketplace app to the pre-production environment

Transfer the Marketplace app to the pre-production environment to test its operability.

To **transfer the Marketplace app to the pre-production environment**:

1. Export packages that contain the Marketplace app as a *.zip archive. To do this, follow the instruction in a separate article: Delivery management process. If the Marketplace app functionality is implemented in multiple packages, combine the exported *.gz archives of packages into a single *.zip archive.
2. Import a *.zip archive of Marketplace app packages or a single *.gz package archive into the testing environment app. To do this, follow the guide in a separate article: Delivery management process.
3. Verify the operability of the developed functionality. If you find an error, improve the functionality by fixing it. After that, repeat steps 1–2.

Marketplace app licensing

Medium

The **purpose** of licensing is to control the use of paid Marketplace apps. Learn about licensing in a separate guide: Licensing.

If Creatio has no licenses uploaded for a product, **Creatio demo mode** is turned on. The entirety of app functionality is available in this mode. However, users can add only up to 1000 records to each database table.

The licensing process consists of the following **steps**:

1. Generate the index of **licensed elements**.

© 2023 Creatio. All rights reserved.

2. Define the **license type**.
3. Send data obtained on previous steps to the Creatio Marketplace support (marketplace@creatio.com). The developer must send data via the Developer profile when publishing the Marketplace app. Learn more in a separate article: Steps to publish the Marketplace app.
4. Creatio Marketplace support generates licenses for the Marketplace app based on the received data.

The developer must take steps 1–2 when developing the Marketplace app.

1. Generate the index of licensed elements

The licensed **elements** are as follows:

- **Licensed objects**. Names of custom objects added to the Marketplace app. Licensed objects are any custom objects, for example, section, detail, or lookup objects.
- **Licensed operations**. Names of operations added to the Marketplace app logic to verify the availability of a license for specific functionality. For example, an additional action was added to a base Creatio section. The action must be connected to the licensed operation. When the action is called, the app program logic verifies the license availability. Based on the result, Creatio executes or interrupts the functionality.

The license can include licensed objects and/or licensed operations. Make sure the licensed elements meet the requirements specified in a separate article: Requirements for published Marketplace app resources. The licensed elements do not depend on the license type.

2. Define the license type

Creatio provides the following **license types**:

- **Personal licenses**. Grant access to the app for specific users. Bound to the user account and cannot be used by other users. The system administrator can redistribute licenses among users at any time.
- **Server licenses**. Grant access to the app without binding access to specific users. All Creatio users that have the corresponding permissions have access to the licensed functionality.

The Marketplace app developer can contact Creatio Marketplace support (marketplace@creatio.com) to **find out the number of paid and granted licenses**.

The **licensed options** of the Marketplace app depend on the license type and licensed element.

Licensed options of the Marketplace app

© 2023 Creatio. All rights reserved.

Licensed element	Marketplace app type	Subscription type
Personal license		
Objects	New Creatio section.	Paid per user that works with the section.
Operations	Connector to an external service, for example, a telephony connector.	Paid per user that can access to the service. For example, access to a telephony connector is paid per individual user.
Objects and/or operations	Section with records that use connectors to various external services.	Paid per user that can access the section, and paid per service connector. For example, the [*Requests*] section was improved. The section is integrated with an external system that registers requests. The customer pays for access to the section functionality per user.
Server license		
Objects	New Creatio section.	Fixed price regardless of the number of users that work with the section.
Operations	Connector to an external service, for example, web chat that registers leads or requests in Creatio.	Fixed price regardless of the number of users that have access to the service.

Attention. Do not use personal and server licenses that control the same objects and operations at the same time.

The license lasts for 365 days regardless of the type. The license expiration date is specified in the license and controlled by Creatio.

You can verify the license expiration date in the following **ways**:

- If the **licensed objects** are included in the license, you do not need to implement the license availability verification. This verification is performed using out-of-the-box Creatio tools. The name of the licensed object starts with the prefix specified in the [*Developer prefix*] field of the Developer profile. Learn more about the Developer profile in a separate article: Prepare to develop a Marketplace app.
- If the **licensed operations** are included in the license, implement the availability verification for the operation license in the Marketplace app source code.

© 2023 Creatio. All rights reserved.

Attention. The schema source code can be read and replaced. Therefore, we do not recommend implementing the availability verification for the operation license in configuration element schemas as this option is insecure. Use the licensed objects to determine the license expiration date instead. We recommend implementing the availability verification for the operation license in an external assembly. To do this, follow the instruction in a separate article: Plagiarism-proof the source code of a Marketplace app.

You can implement the license availability verification in the following **ways**:

- in the back-end
- in the front-end

To verify the availability of the operation license **in the back-end**, use the following **methods** of the `Terrasoft.Core.LicHelper` class:

- `GetHasOperationLicense(string sysPackageOperationCode)`. If the license is found, returns `true`. Otherwise returns `false`.
- `GetHasOperationLicense(string sysPackageOperationCode)`. If the license is found, returns `true`. Otherwise generates the `LicException` exception.

`sysPackageOperationCode`. A string parameter that contains the name of the licensed operation. Specify this name in the Developer profile when publishing the Marketplace app. Learn more in a separate article: Steps to publish the Marketplace app.

Example that use the GetHasOperationLicense() method

```
UserConnection.LicHelper.GetHasOperationLicense("MyMarketplaceApplication.Use");
```

Example that use the CheckHasOperationLicense() method

```
UserConnection.LicHelper.CheckHasOperationLicense("MyMarketplaceApplication.Use");
```

To implement the license availability verification **in the front-end**:

- Implement the custom configuration web service that verifies the license availability.
- Add the program logic of the request to the implemented web service.

Plagiarism-proof the source code of a Marketplace app

Advanced

Creatio stores the open-source code of a Marketplace app in a read-only package. The code is not plagiarism-

© 2023 Creatio. All rights reserved.

proof. Users that have the corresponding access permissions can view the code. Plagiarism-proof your front-end and back-end code separately.

Plagiarism-proof the C# code

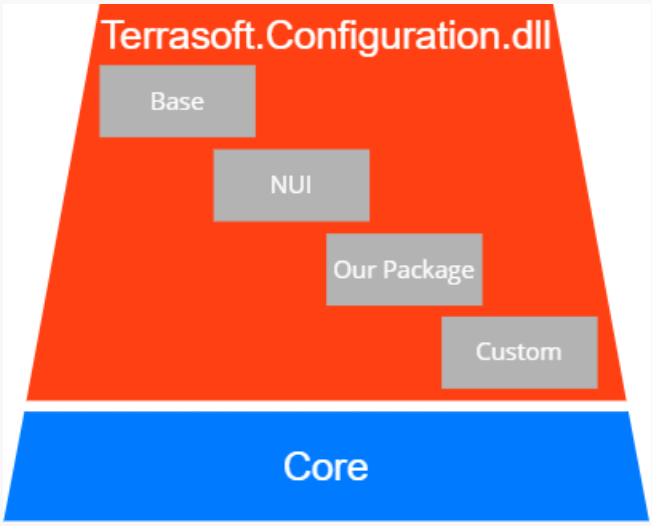
Use the project package to protect the C# source code of the Marketplace app from plagiarism.

Attention. You are permitted to plagiarism-proof only C# source code developed by you.

The **ways** to protect the C# code of a Marketplace app from plagiarism are as follows:

- Develop a **new Marketplace app** as a project package.
- Convert an **existing Marketplace app** to a project package.

Develop your Marketplace app on the configuration level that contains preinstalled Creatio packages, similar to the development of other apps. Compile the C# source code of the packages into the `Terrasoft.Configuration.dll` as part of the publishing. The code can interact with the core. Learn more about Creatio customization levels in a separate article: Creating apps on Creatio platform. View the elaborate diagram of Creatio customization levels on the figure below.



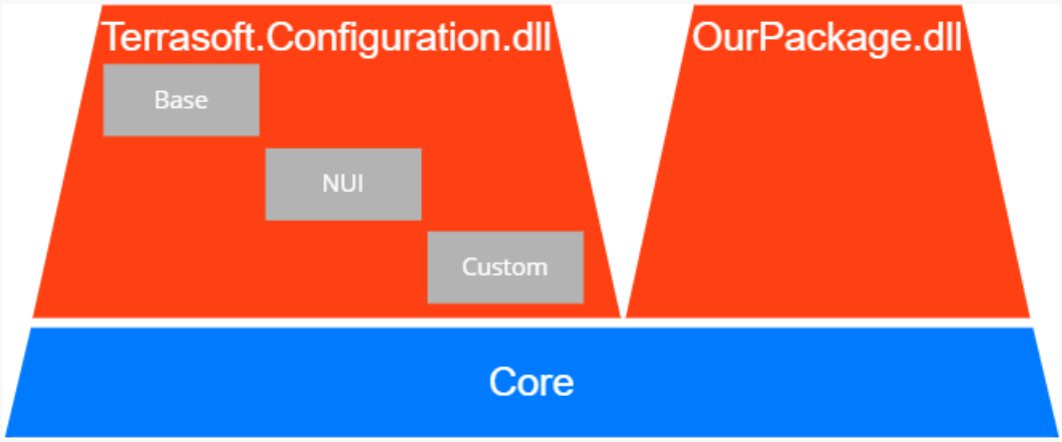
- `Base`, `NUI` are base Creatio packages.
- `OurPackage` is the project package that contains the Marketplace app.
- `Custom` is a special Creatio package.

Project packages offer the following **advantages** for Marketplace apps:

- Exclude the C# code of the custom Marketplace app from `Terrasoft.Configuration.dll`.
- Install the Marketplace app as a separate *.dll.

View the diagram of Creatio customization levels that contain the Marketplace app's project package in the figure below.

© 2023 Creatio. All rights reserved.



Develop the Marketplace app as a project package

We recommend developing new Marketplace apps as project packages.

To **develop the Marketplace app as a project package**:

1. Set up Creatio for file system development.
2. Create a custom package.
3. Code the custom functionality.
4. Build the project package.

1. Set up Creatio for file system development

To set up Creatio for file system development, follow the instructions in a separate article: IDE settings for development.

2. Create a custom package

Use one of the following **tools** to create a custom package:

- Creatio IDE. Learn more about creating a custom package using Creatio IDE in a separate article: Create a custom package.
- Creatio command-line interface utility (clio) utility.

To **create a custom package using the** clio utility:

1. Install clio (if needed).

   Command that installs clio

   ```
   dotnet tool install clio -g
   ```

   Learn more about installing clio in the official utility documentation on GitHub.
2. Go to the Creatio `Pkg` directory.

© 2023 Creatio. All rights reserved.

Command that opens the `Pkg` directory

```
cd C:\inetpub\wwwroot\creatio\Terrasoft.WebApp\Terrasoft.Configuration\Pkg;
```

3. Create a new package.

Command that creates a new package

```
clio init OurPackage;
```

4. Set up the package dependencies. To do this, modify the `descriptor.json` file.

   See the example that sets up the dependencies (the `DependsOn` property) of the `OurPackage` package on the `ProductCore` package and adds the package description (the `Descriptor` property) below.

Example that sets up the package dependencies and adds the package description

```
{
    "Descriptor": {
        "UId": "45cc06b2-6448-4d9e-9f51-bee31a6dbc25",
        "PackageVersion": "7.8.0",
        "Name": "OurPackage",
        "ModifiedOnUtc": "/Date(1633420586000)/",
        "Maintainer": "Customer",
        "Description": "Payment calculator",
        "DependsOn": [
            {
                "UId": "2fabaf6c-0f92-4530-aef8-40345c021da2",
                "PackageVersion": "7.8.0",
                "Name": "ProductCore"
            }
        ]
    }
}
```

This will create the `OurPackage` custom package that depends on the `ProductCore` package.

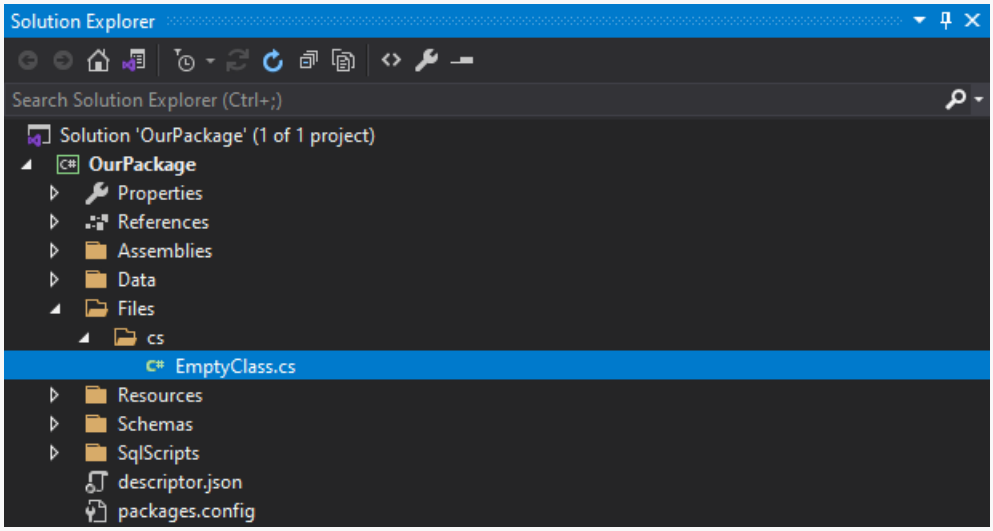
3. Code the custom functionality

You can code the custom functionality in any external IDE. This example uses Microsoft Visual Studio Code.

To **code the custom functionality**:

1. Open the `OurPackage.sln` project.

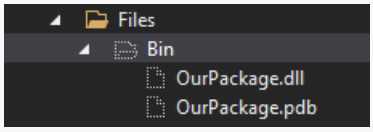
© 2023 Creatio. All rights reserved.



2. Install the CreatioSDK NuGet package from the repository available on the official nuget site. Select the relevant CreatioSDK version.
3. Implement the custom functionality in the `Files\cs` app directory.

   You can create the app in the IDE while developing the C# code. To do this, press `Ctrl+Shift+B` in Visual Studio.
4. Build the app.

   If the IDE builds the app successfully, the *.dll, *.pdb, and other auxiliary files will be placed in the `Files\Bin` app directory.

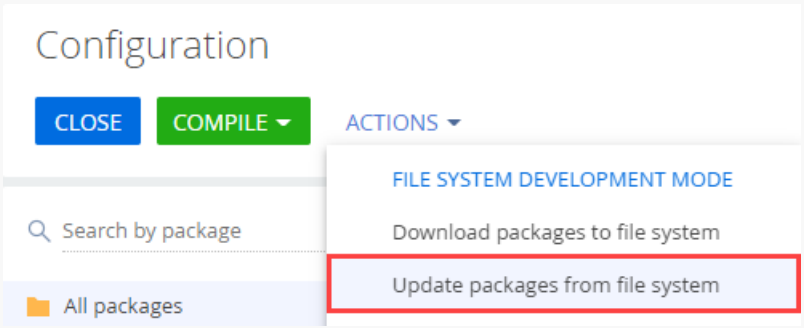


5. Upload the `OurPackage` package from the `[Path to app]\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` directory to the database.

   a. Click to open the System Designer.

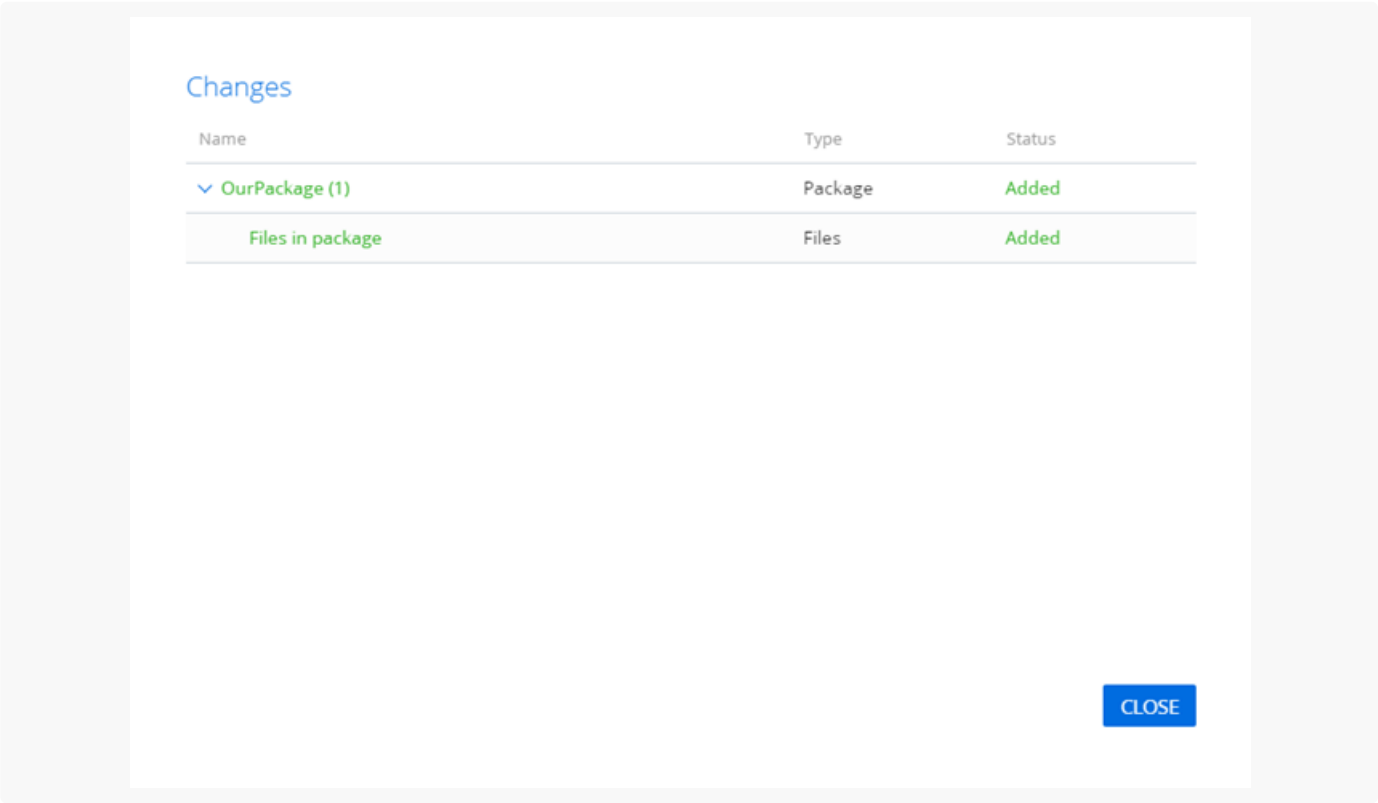
   b. Click [*Advanced settings*] in the [*Admin area*] block.

   c. Click [*Update packages from file system*] in the [*File system development mode*] group of the [*Actions*] dropdown list on the toolbar.



© 2023 Creatio. All rights reserved.

This will upload the `OurPackage` package to Creatio IDE.



6. Restart Creatio.

Command that restarts Creatio

```
clio restart
```

4. Build the project package

Build the project package to **prepare** the Marketplace app for publishing on the Creatio Marketplace online platform.

Attention. If you want to exclude the C# source code that belongs to you from the project package, make sure to delete the code before exporting the package.

To **build the assembly package**:

1. Delete the C# source code that belongs to you from the project package (if necessary). Do this if you want to exclude the C# source code from the project package.
2. Create the `PackagePublish.target` file to automate the project package building.
3. Add the following code to the `PackagePublish.target` file.

© 2023 Creatio. All rights reserved.

`PackagePublish.target` **file**

```
<?xml version="1.0" encoding="utf-8" ?>
<Project xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
    <PropertyGroup>
        <DestinationFolder>C:\PkgRelease\$(AssemblyName)</DestinationFolder>
        </PropertyGroup>

    <ItemGroup>
        <PkgAssemblies Include="Assemblies\**"/>
    </ItemGroup>
    <ItemGroup>
        <PkgData Include="Data\**"/>
    </ItemGroup>
    <ItemGroup>
        <PkgFiles Include="Files\Bin\**"/>
    </ItemGroup>
    <ItemGroup>
        <PkgProperties Include="Properties\**"/>
    </ItemGroup>
    <ItemGroup>
        <PkgResources Include="Resources\**"/>
    </ItemGroup>
    <ItemGroup>
        <PkgSchemas Include="Schemas\**"/>
    </ItemGroup>
    <ItemGroup>
        <PkgSqlScripts Include="SqlScripts\**"/>
    </ItemGroup>
    <ItemGroup>
        <PkgDescriptor Include="descriptor.json"/>
    </ItemGroup>

    <Target Name="CopyFiles">
        <Copy
            SourceFiles="@(PkgAssemblies)"
            DestinationFiles="@(PkgAssemblies->'$(DestinationFolder)\Assemblies\%(RecursiveDi
        />
        <Copy
            SourceFiles="@(PkgData)"
            DestinationFiles="@(PkgData->'$(DestinationFolder)\Data\%(RecursiveDir)%(Filename
        />
        <Copy
            SourceFiles="@(PkgFiles)"
            DestinationFiles="@(PkgFiles->'$(DestinationFolder)\Files\Bin\%(RecursiveDir)%(Fi
        />
        <Copy
            SourceFiles="@(PkgProperties)"
```

© 2023 Creatio. All rights reserved.

```
            DestinationFiles="@(PkgProperties->'$(DestinationFolder)\Properties\%(RecursiveDi
        />
        <Copy
            SourceFiles="@(PkgResources)"
            DestinationFiles="@(PkgResources->'$(DestinationFolder)\Resources\%(RecursiveDir)
        />
        <Copy
            SourceFiles="@(PkgSchemas)"
            DestinationFiles="@(PkgSchemas->'$(DestinationFolder)\Schemas\%(RecursiveDir)%(Fi
        />
        <Copy
            SourceFiles="@(PkgSqlScripts)"
            DestinationFiles="@(PkgSqlScripts->'$(DestinationFolder)\SqlScripts\%(RecursiveDi
        />
        <Copy
            SourceFiles="@(PkgDescriptor)"
            DestinationFiles="@(PkgDescriptor->'$(DestinationFolder)\%(RecursiveDir)%(Filenam
        />
    </Target>

    <Target Name="CreateRelease" AfterTargets="CopyFiles">
        <Exec Command="clio generate-pkg-zip  $(DestinationFolder) -d C:\PkgRelease\$(Assembl
    </Target>
</Project>
```

4. Add the following string to the `OurPackage.csproj` file.

`OurPackage.csproj` file

```
<Import Project="PackagePublish.target" />
```

5. Open the command line and run the following command.

```
msbuild /t:CreateRelease
```

This will download the project package to the `C:\PkgRelease` directory that contains the `OurPackage` subdirectory and the `OurPackage.gz` *.gz archive. The archive contains the Marketplace app ready to be published on the Creatio Marketplace online platform.

Convert the Marketplace app package to a project package

You might need to modify your Marketplace app significantly to prepare it for the conversion to a project package.

Use the `clio convert` command to convert the existing Marketplace app to a project package. Learn more about the command in the official utility documentation on GitHub.

© 2023 Creatio. All rights reserved.

Some files and schemas are not suitable for conversion. For example, the [*User task*] business process element. The [*User task*] process element remains a partial class regardless of the [*Partial*] flag status. Place the element to `Terrasoft.Configuration.dll` library. By default, the utility saves the code of the [*User task*] element to the `AutoGenerated` directory of the project package, not `Terrasoft.Configuration.dll`, as part of the conversion.

To **convert the existing Marketplace app to a project package**, run one of the following commands:

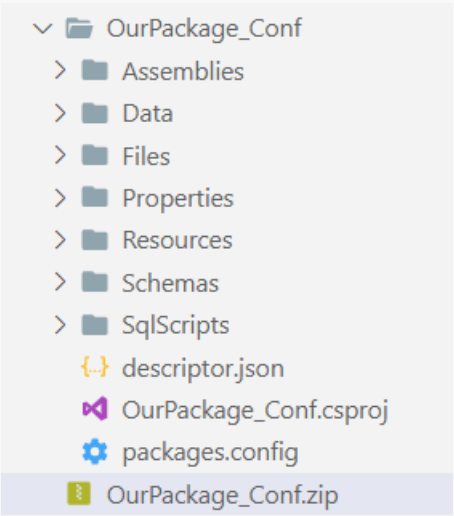
- `clio convert .\OurPackage_Conf\ -c false`.
- `clio convert .\OurPackage_Conf\`.

The outcome of each command is identical since the utility sets the `-c` (ConvertSourceCode) key to `false` by default.

After the conversion, the C# package will contain **the following**:

- The project package, which contains the converted Marketplace app.
- The *.zip archive that contains the original Marketplace app. Learn more about the structure of the project package in a separate article: Project package.

View the structure of the C# project after the conversion on the figure below.



This will convert the existing Marketplace app to a project package ready to be installed into Creatio.

Note. Generate the project package as part of the CI/CD pipeline. We recommend storing the unprotected source code of your Marketplace app in the repository.

Plagiarism-proof the JavaScript code

The **ways** to plagiarism-proof the JavaScript code of a Marketplace app are as follows:

- minification
- obfuscation

© 2023 Creatio. All rights reserved.

Attention. Do not modify the structure of the client module schema. Creatio Designers expect a particular schema structure.

The best way to plagiarism-proof the JavaScript code is to implement the protected logic using mixins. We do not recommend obfuscating client module schemas that Creatio Wizards (Section Wizard, Page Wizard, Detail Wizard) utilize.

You can obfuscate the JavaScript code using a large number of open source solutions. This example uses JavaScript Obfuscator. View the official documentation of the obfuscator on GitHub.

To **plagiarism-proof the JavaScript code using JavaScript Obfuscator**:

1. Install JavaScript Obfuscator.

   Command that installs JavaScript Obfuscator

   ```
   npm install javascript-obfuscator -g
   ```

   Learn more about installing JavaScript Obfuscator in the official obfuscator documentation on GitHub.

2. Prepare the JavaScript code for obfuscation.

   a. Create a mixin. In this example, this is `MRKT_DemoMixin`. Learn more about mixins in a separate article: Mixins. The "mixins" property.

   b. Implement the JavaScript code to obfuscate in the mixin.

   `MRKT_DemoMixin`

   ```
   define("MRKT_DemoMixin", [], function() {
       Ext.define("Terrasoft.configuration.mixins.MRKT_DemoMixin", {
           alternateClassName: "Terrasoft.MRKT_DemoMixin",

           secretMethod: function(){
               console.log("MRKT_DemoMixin");
           },
       });
   });
   ```

3. Specify the `MRKT_DemoMixin` mixin in the `mixins` property of the client schema. For example, `ContactPageV2`.

   `ContactPageV2`

   ```
   define("ContactPageV2", ["MRKT_DemoMixin"],
       function() {
           return {
               entitySchemaName: "Contact",
   ```

© 2023 Creatio. All rights reserved.

```
            mixins: {
                "MRKT_DemoMixin": "Terrasoft.MRKT_DemoMixin"
            },
            attributes: {},
            modules: /**SCHEMA_MODULES*/{}/**SCHEMA_MODULES*/,
            details: /**SCHEMA_DETAILS*/{}/**SCHEMA_DETAILS*/,
            businessRules: /**SCHEMA_BUSINESS_RULES*/{}/**SCHEMA_BUSINESS_RULES*/,
            methods: {
                onEntityInitialized: function() {
                    this.callParent(arguments);

                    /* Consume MRKT_DemoMixin. */
                    this.secretMethod();
                },
            },
            dataModels: /**SCHEMA_DATA_MODELS*/{}/**SCHEMA_DATA_MODELS*/,
            diff: /**SCHEMA_DIFF*/[]/**SCHEMA_DIFF*/
        };
    });
```

4. Back up the unprotected JavaScript code. You might need to edit it later.
5. Obfuscate the JavaScript code.

Command that obfuscates the JavaScript code

```
javascript-obfuscator MRKT_DemoMixin.js --output MRKT_DemoMixin.obfuscated.js
```

Learn more about obfuscation in JavaScript Obfuscator in the official obfuscator documentation on GitHub.

As a result, JavaScript Obfuscator will generate an obfuscated file. View the example of the obfuscated file below.

Example of an obfuscated file

```
function a0_0x2c69() {
    var _0x272852 = ['Terrasoft.MRKT_DemoMixin', 'Ok\x20-\x20MRKT_DemoMixin', '636527SjITzq', '4
    a0_0x2c69 = function() {
        return _0x272852;
    };
    return a0_0x2c69();
}
var a0_0x31e3ab = a0_0x3f9a;

function a0_0x3f9a(_0x104961, _0x4f9883) {
    var _0x2c694d = a0_0x2c69();
    return a0_0x3f9a = function(_0x3f9a09, _0x1d247d) {
        _0x3f9a09 = _0x3f9a09 - 0xa7;
```

© 2023 Creatio. All rights reserved.

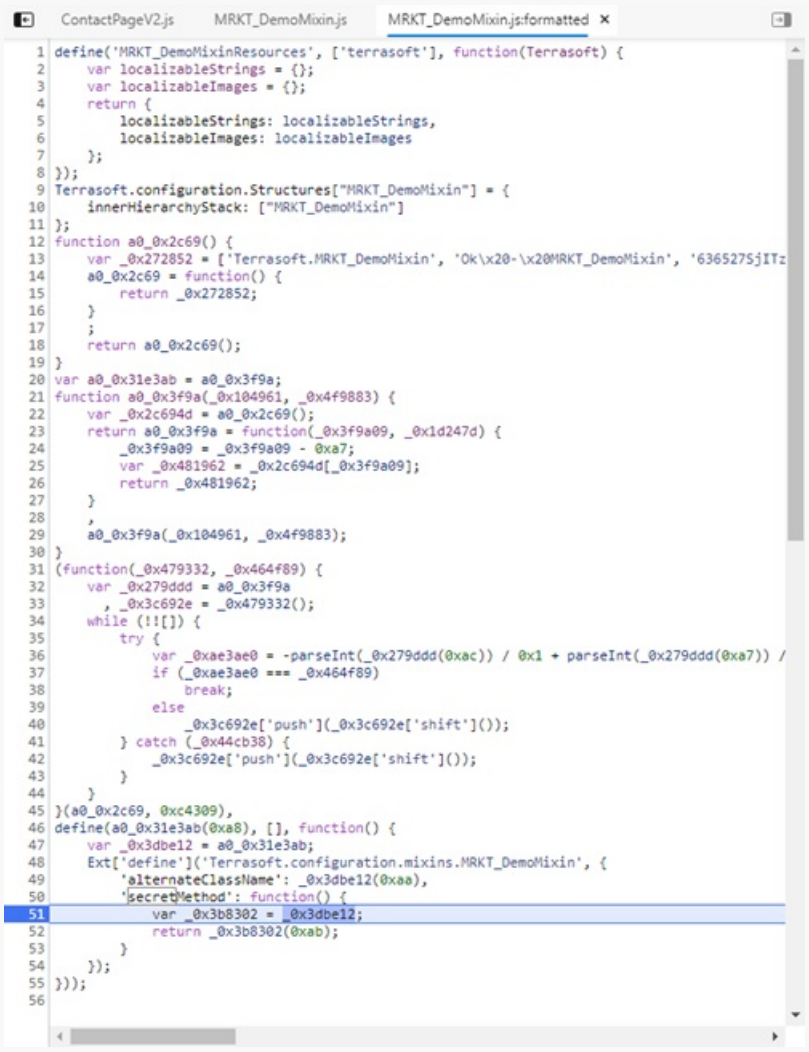
```
        var _0x481962 = _0x2c694d[_0x3f9a09];
        return _0x481962;
    }, a0_0x3f9a(_0x104961, _0x4f9883);
}(function(_0x479332, _0x464f89) {
    var _0x279ddd = a0_0x3f9a,
        _0x3c692e = _0x479332();
    while (!![]) {
        try {
            var _0xae3ae0 = -parseInt(_0x279ddd(0xac)) / 0x1 + parseInt(_0x279ddd(0xa7)) / 0x2 +
            if (_0xae3ae0 === _0x464f89) break;
            else _0x3c692e['push'](_0x3c692e['shift']());
        } catch (_0x44cb38) {
            _0x3c692e['push'](_0x3c692e['shift']());
        }
    }
}(a0_0x2c69, 0xc4309), define(a0_0x31e3ab(0xa8), [], function() {
    var _0x3dbe12 = a0_0x31e3ab;
    Ext['define']('Terrasoft.configuration.mixins.MRKT_DemoMixin', {
        'alternateClassName': _0x3dbe12(0xaa),
        'secretMethod': function() {
            var _0x3b8302 = _0x3dbe12;
            return _0x3b8302(0xab);
        }
    });
}));
```

To **view the JavaScript code in the browser page**:

1. Clear the browser cache.
2. Refresh the page.
3. Open the developer tools.

View the example of the JavaScript code in the browser page in the figure below.

© 2023 Creatio. All rights reserved.



Note. Obfuscate the JavaScript code as part of the CI/CD pipeline. We recommend storing the unprotected JavaScript source code of your Marketplace app in the repository.

© 2023 Creatio. All rights reserved.