**General information**

Acumatica Customization Utility (ACU) is a utility designed to automate daily routine tasks related to the creation and maintenance of built-in applications (customization packages) of ERP Acumatica.

ACU is a CLI utility that implements a system of commands that are intuitively structured according to their functional purpose and allow you to perform conveniently the necessary actions when creating and maintaining customization packages. A complete description of all available commands and their corresponding options can be found in the ACU Reference.xlsx document.

ACU allows to perform the following actions (by groups of commands):

ERP: download of installation package of the required version, install of ERP Acumatica, remove of ERP Acumatica.

Site: Deploy, upgrade (instance or database mode), delete.

Code - The source code of the customization package and the extension library: downloading the source code of the customization package from the instance, compiling the extension library code, building the customization package (three modes are supported - normal, build for ISV, build for QA).

Package: upload and publish custom packages to the site, receive the package from the site.

**Installation**

The customization package can be obtained from:

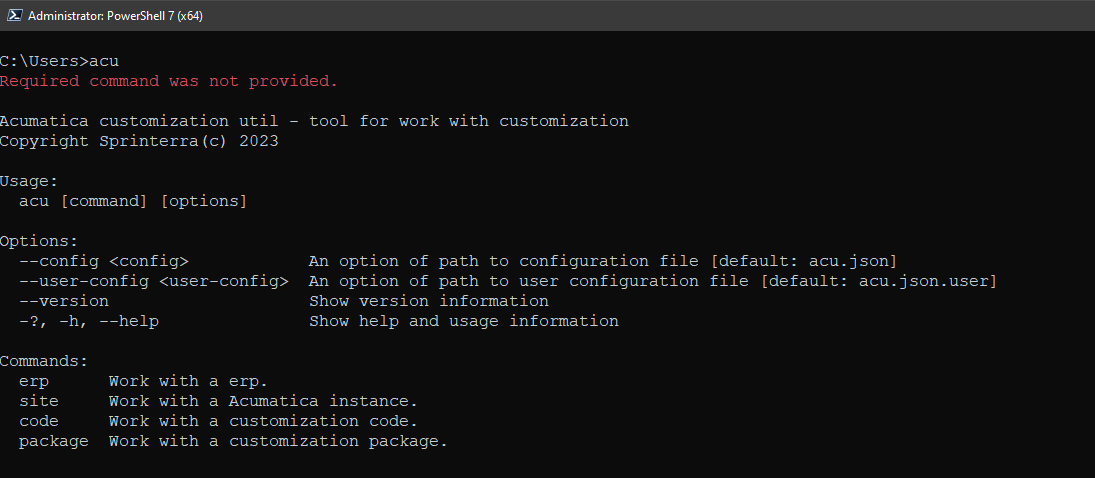
<https://drive.google.com/drive/folders/1K9JcIMjdnBsRxzI_VMFoBacPAH_vbU5t?usp=sharing>

The ACUCustomixzationUtil folder contains versions of the utility in folders corresponding to the version.

You should select the most recent version of the utility and copy the contents of the folder to your PC, for example, to the folder C:\Users\ %USERNAME%\ACU. After that, you need to add the folder with the utility to the variable Path of the environment;

Next, you need to check that the utility is working correctly: start PowerShell and run the **acu** /? command.

The output should be similar to the one shown in the image.



**Configuration**

The ACU configuration includes the following items:

The main configuration file, which is given by the global option -config, the default name is acu.json, JSON format;

User configuration file, given by --user-config option, default name is acu.json.user, JSON format;

Configuration options for the corresponding command.

The complete information on the configuration file parameters is presented in the ACU Configuration Reference.xlsx document, the information on command options is presented in the ACU Command Reference.xlsx document.

When using the utility, it is not necessary to set the acu.json acu.json.user files each time using the --config and --user-config options, these options only need to be used when using files with **not default** names .

To use the configuration it is enough to create and place the acu.json and (if necessary) acu.json.user files in the customization root directory.

The command options are used when using the utility without configuration files or for one-time modification of the configuration when executing a command.

When using two or three configuration items at the same time, the command option has the highest priority, the acu.json file has the lowest priority

For example, with this configuration, the final value will be C:\Acumatica\Sites\AcutestUtil

|  |  |  |
| --- | --- | --- |
| acu.json | acu.json.user | command option |
| site.instancePath | site.instancePath | --instancePath |
| C:\\ACU\\instances\\AcuTestUtil | D:\\ACU\\inst\\AcuTestUtil | C:\Acumatica\Sites\AcutestUtil |

In configuration files, values are written as “parameter\_name”:”value”

If the value of the parameter is not defined, you should use this entry

“parameter\_name”: null

The double backslash, \\, must be used as a path separator for files and directories;

A relative path (relative to the current directory) can be specified like this: folder\\subfolder.

**Built-in help system**

The utility contains a built-in help system. For each command, you can view a list of its subcommands and parameters. Options for calling help are -?, -h, --help

For example,

acu –help - will show the initial section of the help system;

acu erp –help - will show the help system section on the erp command;

acu erp download /? - will show the help section on the download command of the erp command

**Logging**

When running, the utility creates log files in the directory where the utility is launched from.

Log files are named acu-log[yyyyMMdd].txt, where [yyyyMMdd] is the variable part and indicates the date the file was created.

The log file has the following format

2023-06-29 18:41:09.668 +03:00 [ERR] Log Message, that is, the time of the event in UTC, the type of event, and the description of the event.

**Working with the utility**

The typical tasks when working with Acumatica ERP customizations are creating a customization project from scratch and deploying the finished project. If in the first case, the automation of routine tasks is not a critical factor determining the time spent on creating a project from scratch, then in the second case, deployment automation is a critical condition that directly affects the time for which the programmer will be ready to work with the project.

Consider the operation of the utility on these examples.

**Creating a customization project**

**Basic Steps**

When creating and configuring a customization project for further use, you need to perform the next steps

1. Determine the ERP version number, check for its availability, install if necessary

2. Install the Acumatica instance (Site) with which we will work

3. Create a C# library project to be used as an Extension library

4. Connect Extension library & Site to .NET Solution

5. Create an empty customization package, add the assembly file (dll) extension library to it

6. Set up and check the operation of the utility in the such scenarios:

a. Getting the source code;

b. Building the package using the utility;

c. Uploading the package to the Site and publishing the package;

d. Building a package for QA & ISV

7. Preparing the project for uploading to the repository

8. Uploading the project to the repository

**Directory structure**

Classic style:

The most of the implemented projects use the following directory structure:

ProjectName

ProjectName project C#, extension library

ProjectName.source customization project source code

ProjectName.package directory for customization packages

ProjectName.webapp directory to deploy the Acumatica instance

This was because the previous CI/CD automation tool was based on convention and the structure and naming of directories was the basic principle.

The same structure can continue to be used for existing customization projects, the ACU can be easily configured to work with any directory builder.

The modern style:

PackageName

code project C#, extension library

pkg directory for customization packages

src customization project source code

As you can see, there is no directory for deploying an instance here. And this is natural, because ACU allows not only to install the Akumatica instance in any directory convenient for the user, but also to configure customization projects for sharing instances of the required versions.

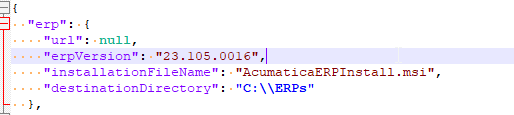
**Start creating a project**

The first step to create a customization project is to create a root folder where the project will be located. For convenience of presentation, let's agree that the project is called **TestOne** and is located in the **C:\ERPs\projects\TestOne folder.**

**Before starting work, it is optimal to clone the project repository into an empty directory. Later it will not be so easy to do!**

**ERP Installation**

To install the UKZ, you need to copy the **acu.json** file to the root folder of the project and open it for editing. The **erp** object is responsible for ERP configuration.



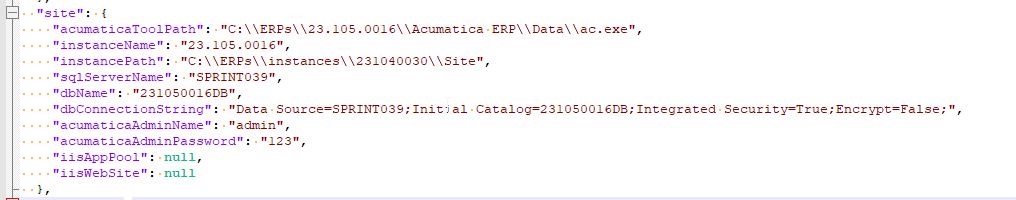
You need to change the erpVersion and destinationDirectory parameters according to your requirements. Then start PowerShell in administrator mode, go to the **C:\ERPs\projects\TestOne** directory and run the commands:

***PS > acu ero download***

***PS > acu erp install***

**Installing an instance**

The **site** object is responsible for configuring the instance in the configuration file



You have to enter your own parameter values (all parameters are required except iisAppPool and iisWebSite).

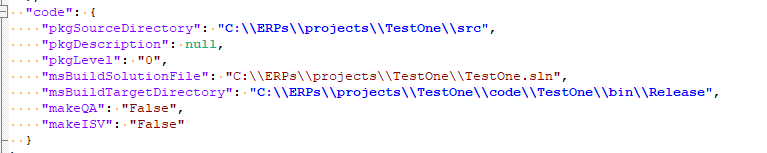
Then run the command

***PS > acu site install***

This command will install an instance with SalesDemo data and reset the admin user password as configured.

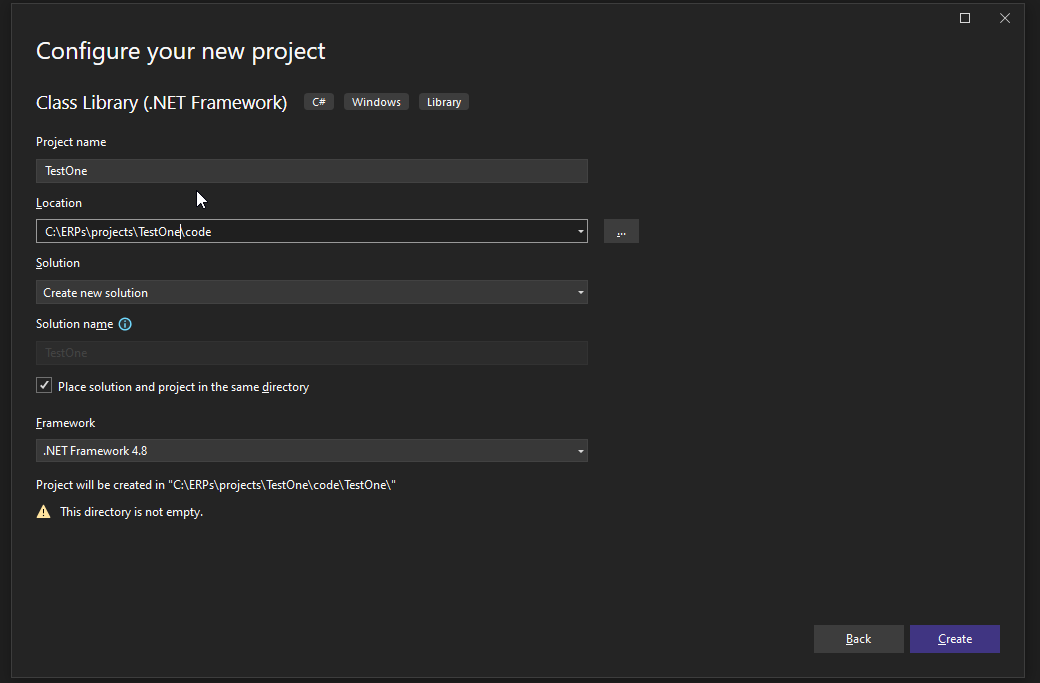
**Project setup C# (extension library):**

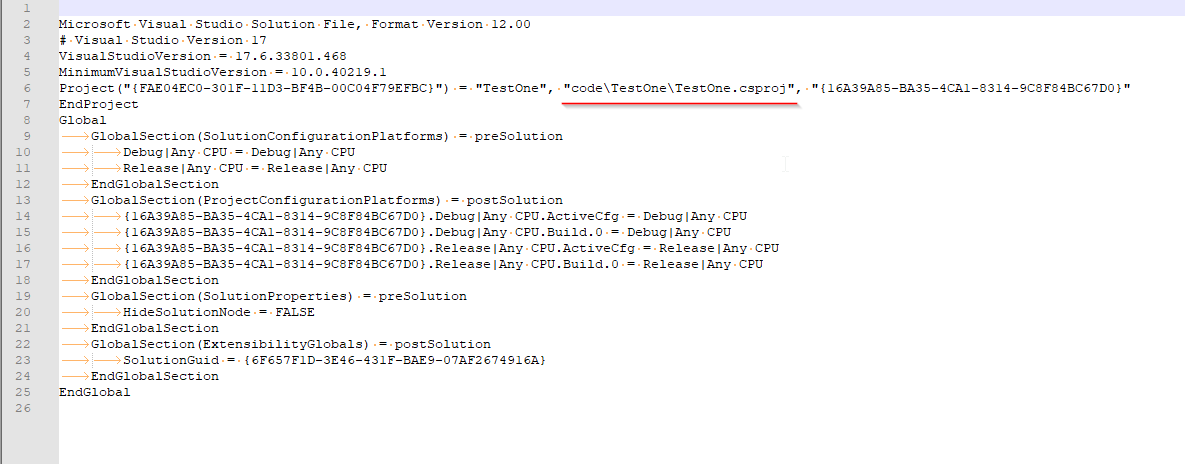
The values of the source code configuration parameters are stored in the **code** object of the configuration file.

****

To configure the extension library source code, you must enter values for the msBuildSolutionFile and msBuildTargetDirectory parameters.

To create a project, follow these steps:

1. Open the MS Visual Studio
2. File->New-Project, select the темплейт Class Library (.NET Framework) template and and clich the Next button
3. Enter the Project Name, Project Location  
   
4. Click the Create button and creat a project and solution.
5. Open the project folder of the external library and move the \*.sln file to the root folder of the custom project.
6. Open the \*.sln file and make changes to the path to the \*.csproj project file



1. Open the project file and add the task to it:

<Target Name="BeforeBuild">

<ItemGroup>

<AssemblyAttributes Include="AssemblyVersion">

<\_Parameter1>$(Version)</\_Parameter1>

</AssemblyAttributes>

</ItemGroup>

<MakeDir Directories="$(IntermediateOutputPath)" />

<WriteCodeFragment Language="C#"

OutputFile="$(IntermediateOutputPath)Version.cs"

AssemblyAttributes="@(AssemblyAttributes)" />

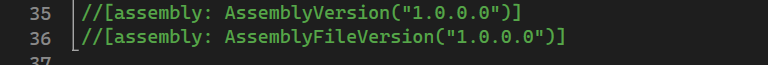
<ItemGroup>

<Compile Include="$(IntermediateOutputPath)Version.cs" />

</ItemGroup>

</Target>

1. Open the AssemblyInfo.cs file and comment out the 2 last line in it:



1. Open the project properties editing window, go to the BuildEvents tab, add the following lines to the Post-build event command-line

**xcopy /F /Y $(TargetPath) C:\ERPs\instances\231040030\Site\Bin\  
xcopy /F /Y $(TargetDir)$(TargetName).pdb C:\ERPs\instances\231040030\Site\Bin\**where C:\ERPs\instances\231040030\Site\Bin\ is your path to the Bin directory of the instance against which the customization project is being developed

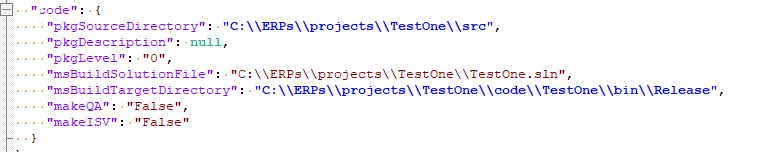
10. Build the project, make sure that the assembly files [ProjectName].dll & [ProjectName].pdb are successfully copied to the [Path to Instance]/Site/Bin directory.

11. Add the necessary References from the Site\Bin directory

12. Enter the project parameters into the code object of the configuration file

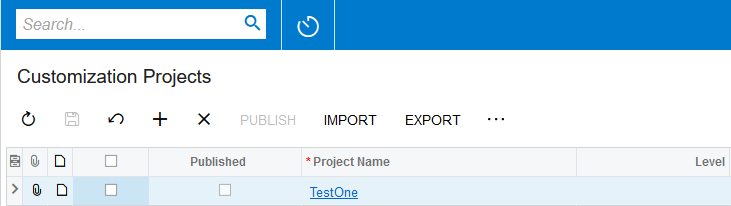
**Creating and configuring a custom package**

The code object of the configuration file is responsible for the configuration of the source code of the customization package.



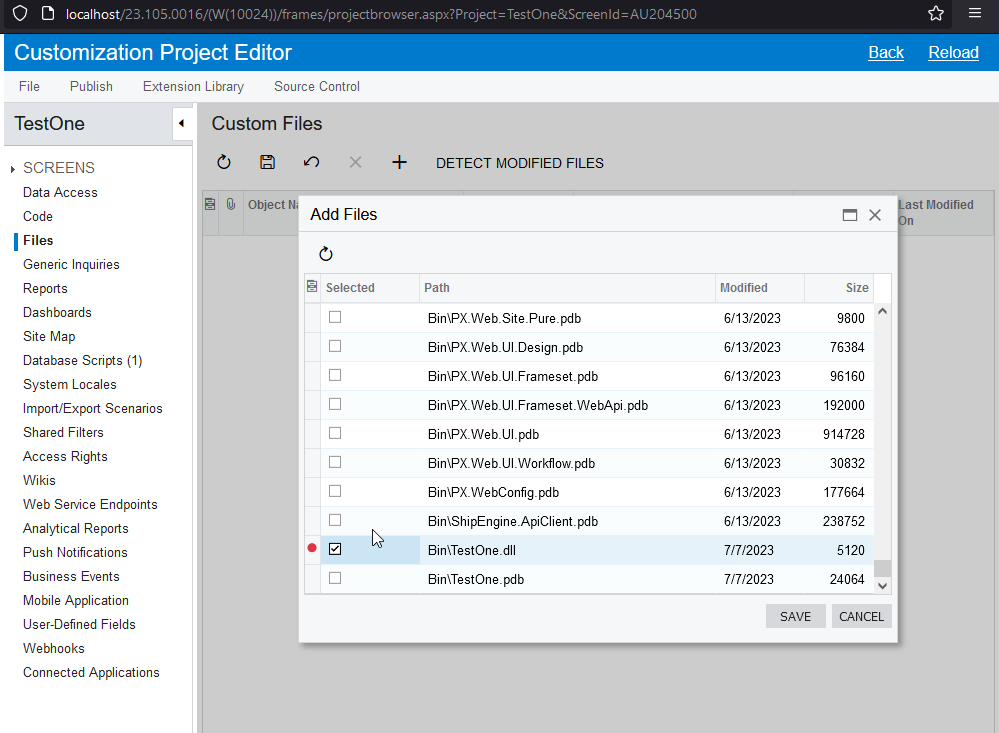
You can set values for pkgSourceDirectory, pkgDescription, pkgLevel.

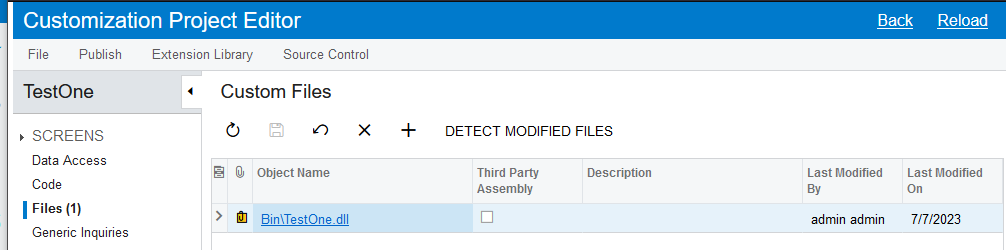
To create a customization package, you need to open the instance in the browser, go to the Configuration->Customization section and open the CustomizationProjects screen. After that, on the screen, add a line to the grid and enter the name of the customization.



Next, after clicking on the customization name, go to the customization project editor.

In the left panel, select Files, in the right panel, click +, in the window that opens, add the external assembly file to customization:





After that, you can close the customization editor.

**Getting the customization source code**

To get the customization source code, run the command

***PS > acu code src***

If the command was executed without errors, you need to check that the customization source code was pulled into the configured directory (pkgSourceDirectory parameter)

**Building a package**

To build a package from source, run the following commands:

***PS > acu code compile***

***PS > acu code make***

If the commands were executed without errors, you need to check that the customization package appeared in the configured directory (packageDirectory parameter).

**Building a package for QA and ISV**

To build packages from source for QA and ISV, you need to run the following commands:

***PS > acu code make --mode QA***

***PS > acu code make --ISV***

Если команды выполнилась без ошибок, необходимо проверить что пакеты кастомизации появились в сконфигурированном каталоге (параметр packageDirectory)

If the commands were executed without errors, you need to check that the customization package appeared in the configured directory (packageDirectory parameter).

**Uploading and publishing a package to an instance**

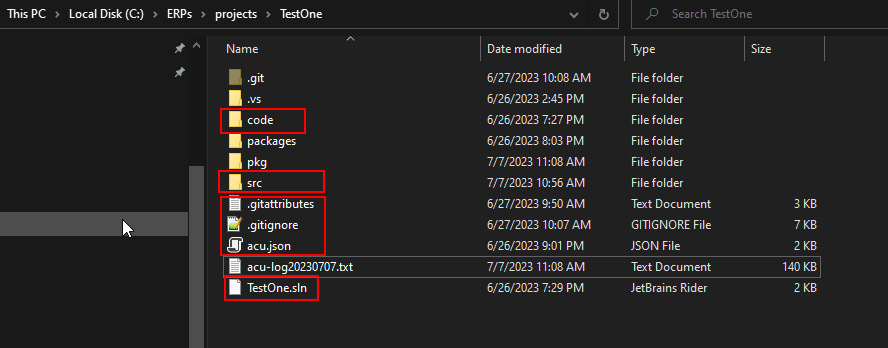
To download and publish a package, run the following commands:

***PS > acu package upload***

***PS > acu package publish***

**Uploading a customization project to Git**

After all the manipulations with the creation of the outer library project, changing the settings and checking the operation of the commands, the folder from the customization package should look like this



The folders and files circled in red should get into the repository.

The following rules should be added to the .gitignore file

*acu-log\*.txt  
/pkg/\*.zip  
bin  
obj  
Debug  
Release  
.vs*  
  
It is also necessary to add all files from the **src** directory to the repository

***PS > git add src\Bin\\*.dll -f***

***PS > git add src\\_project\\****

***PS > git add src\\****

After completing these steps, you can upload the created project to the repository.

**Deploying a customization project**

For convenience, we will show the deployment on a finished project. All paths and names can be changed according to the user's requirements.

In our example, the projects will be located in the C:\ERPs\projects folder, TestOne project folder

1. Clone the repository

You need to run PowerShell in administrator mode and run the commands

***PS > cd C:\ERPs\projects  
PS > git clone --progress -v*** "https://aleksejslusar@bitbucket.org/sprinterraacumatica/testone.git" "C:\ERPs\projects\TestOne"

1. ERP installation  
   Run command in thePowerShell window:  
   ***PS > cd C:\ERPs\projects/TestOne  
   PS > acu erp download  
   PS > acu erp install***
2. Acumatica instance installation  
   Run command in thePowerShell window:  
   ***PS > acu site install***
3. Package building  
   ***PS > acu code compile  
   PS > acu code make***
4. Package publishing   
   ***PS > acu package upload  
   PS > acu package publish***

**Development**

When developing the required functionality, two things are critical: getting the code from the repository and pushing the code to the repository. Consider. What commands should be executed in the first and second case.

**Updating the code from the repository.**

When updating the source code from the repository, it should be understood that both the extension library code and the customization code can be updated. Therefore, to update the package installed on the instance, you need to rebuild the package, update it on the instance, and publish it. Commands for this:

***PS > acu code compile  
PS > acu code make  
PS > acu package upload  
PS > acu package publish***

**Submitting code to the repository**

Before submitting the code to the repository, you need to update the customization source code.

To do this, open the customization project editor and sequentially select the values from the list on the left. For positions where the “Reload From Database” button will appear on the right, you need to update the content from the database (click this button).

After that, execute the commands

***PS > acu code src  
PS > acu code compile***