Creatio Academy

Delivery in WorkspaceConsole

Deployment in WorkspaceConsole

Version 7.18

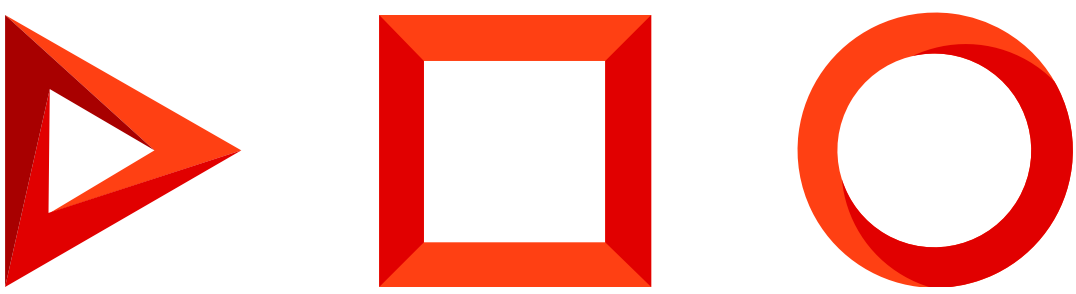


Table of Contents

Deployment in WorkspaceConsole

Medium

Creatio provides various functionality deployment tools.

The **deployment management tools** are as follows:

- Creatio IDE
- WorkspaceConsole utility

This article covers feature deployment using the WorkspaceConsole utility.

WorkspaceConsole is a utility that manages Creatio packages and configuration element schemas.

The WorkspaceConsole utility provides the following solution transfer options:

- Transfer packages and configuration element schemas between environments and configurations.
- Install new packages when updating or migrating from a development environment.
- Transfer the bound package data. For example, the lookup content, new system settings, demo section records, etc.
- Transfer the localizable resources.
- Create and transfer workspaces between environments.

Set up the WorkspaceConsole utility before usage.

Set up the WorkspaceConsole utility

1. Find out the relevant connection string value.

   To do this, open the `..\Terrasoft.WebApp\DesktopBin\WorkspaceConsole\Terrasoft.Tools.WorkspaceConsole.exe.config` file. The `connectionStringName` attribute of the `<db>` XML element contains the connection string.

   `Terrasoft.Tools.WorkspaceConsole.exe.config` file

   ```
   <terrasoft>
       ...
       <db>
           <general connectionStringName="db" securityEngineType="Terrasoft.DB.MSSql.MSSqlSecuri
       </db>
       ...
   </terrasoft>
   ```

2. Edit the connection string.

   To do this, open the `ConnectionStrings.config` file in the Creatio root directory. The `name` attribute of the

`<connectionStrings>` XML element contains the connection string. The value of the `name` attribute in the `ConnectionStrings.config` file must match the value of the `connectionStringName` attribute in the `Terrasoft.Tools.WorkspaceConsole.exe.config` file.

`ConnectionStrings.config` file

```
<connectionStrings>
    <add name="db" connectionString="Data Source=dbserver\MSSQL2016; Initial Catalog=YourDBNa
    <add name="dbOracle" connectionString="Data Source=(DESCRIPTION = (ADDRESS_LIST = (ADDRES
</connectionStrings>
```

Note. By default, the configuration file contains two connection strings. Creatio uses the `"db"` string to connect to the Microsoft SQL Server database and the `"dbOracle"` string to connect to the Oracle database

To perform a one-time operation with WorkspaceConsole, run the utility with the `-webApplicationPath` parameter. Specify the path to the Creatio directory in this parameter. In this case, the utility will fetch the necessary database connection parameters from the `ConnectionStrings.config` file. The database connection parameters in the `Terrasoft.Tools.WorkspaceConsole.exe.config` file will be ignored.

3. Set the `enabled` attribute of the `loadFromRemoteSources` element in the `Terrasoft.Tools.WorkspaceConsole.exe.config` file to `true`.

```
<loadFromRemoteSources enabled="true" />
```

4. Install the WorkspaceConsole utility.

   To do this, run a preconfigured batch command file in the `..\Terrasoft.WebApp\DesktopBin\WorkspaceConsole\` directory as an administrator. This action will install the executable file version and the libraries that the utility requires.

   The WorkspaceConsole utility batch command files:

   - For **32-bit operating systems**, run the `PrepareWorkspaceConsole.x86.bat` file.
   - For **64-bit operating systems**, run the `PrepareWorkspaceConsole.x64.bat` file.

5. Integrate the utility with the SVN repository (optional).

   To do this, copy the `SharpPlink-x64.svnExe`, `SharpSvn.dll`, and `SharpSvn-DB44-20-x64.svnDll` files from the relevant directory to the `...\Terrasoft.WebApp\DesktopBin\WorkspaceConsole` directory.

   - For **32-bit operating systems**, copy the files from the `...\Terrasoft.WebApp\DesktopBin\WorkspaceConsole\x86` directory.
   - For **64-bit operating systems**, copy the files from the `...\Terrasoft.WebApp\DesktopBin\WorkspaceConsole\x64` directory.

Use the WorkspaceConsole utility

The `Terrasoft.Tools.WorkspaceConsole.exe` executable file of the utility is located in the `..\Terrasoft.WebApp\DesktopBin\WorkspaceConsole` directory. The utility version matches the Creatio version.

Attention. The WorkspaceConsole utility version must match the Creatio version. For example, if the current Creatio version is 7.18.1.1794, and you need to update the packages to version 7.18.2.1658, use WorkspaceConsole version 7.18.2.1658. Contact support to obtain the relevant version of the utility.

WorkspaceConsole interacts directly with the Creatio database. Enter the database information in the `Terrasoft.Tools.WorkspaceConsole.exe.config` utility configuration file to ensure the utility works properly. If you deploy Creatio in the cloud, only the employees of the cloud service department will be able to work with the utility. In this case, contact support to transfer the changes.

We recommend typing the `WorkspaceConsole` commands in a text editor and saving them as a *.bat or *.cmd batch file.

To **transfer solutions using the WorkspaceConsole utility**:

1. Check the data bindings.
2. Back up the database.
3. Export the packages.
4. Import the packages.
5. Reboot the Creatio application in IIS.

1. Check the data bindings

Before exporting the package, make sure the data is bound to the package correctly. Bound data includes lookup values, new system settings, demo section records, etc.

If you create a section in the Section Wizard, the Wizard will bind the data required for correct operation automatically. Bind the corresponding value of the `SysModuleInWorkplace` object to ensure Creatio displays the section in the workplace after the import.

2. Back up the database

Before you implement the changes in Creatio, back up the database using the WorkspaceConsole utility. This will let you restore Creatio should the utility run with incorrect commands and parameters.

3. Export the packages

Use the WorkspaceConsole utility to export a package from the database or SVN repository.

Export the packages from the database

1. Configure the command to export packages.

WorkspaceConsole parameters for exporting packages from the database

Parameter	Value	Description
`-operation`	`SaveDBContent`	Saves the database content to the file system. The `-contentTypes` parameter value determines the content type. The `-destinationPath` parameter determines the content export path. Requires either the `-webApplicationPath` or `-configurationPath` parameter.
`-contentTypes`	`Repository`	The type of content exported to the drive from the database. If set to `Repository`, the utility will export the workspace specified in the `-workspaceName` parameter to the directory specified in the `-destinationPath` parameter.
`-workspaceName`	[*Workspace name*]	The name of the workspace (configuration) where the operation is performed. By default, all users work in the `Default` workspace.
`-destinationPath`	[*Path to a local directory*]	The path to a local directory. The utility will export the *.gz package archives to this directory.
`-webApplicationPath`	[*Path to a local directory*]	The path to the Creatio installation directory. The utility will follow this path to fetch the database connection information from the `ConnectionStrings.config` file. If you omit this parameter, the utility will connect to the database specified in the configuration string in the utility's configuration file.
`-configurationPath`	[*Path to a local directory*]	The path to the `..\Terrasoft.WebApp\Terrasoft.Configuration` directory. The utility will export the source code and resources of custom package schemas to this directory in the file system development mode.

Run the command to export packages from the database at the Windows command interpreter (console). Structure the command as follows:

```
[Path to WorkspaceConsole]\Terrasoft.Tools.WorkspaceConsole.exe -operation=SaveDBContent -con
```

2. Run the utility.

Note. The package export operation will export all packages of the workspace. The process may take several dozen minutes.

As a result, the utility will export the workspace package archives to the local directory.

Export packages from the SVN repository

1. Configure the command to export packages.

WorkspaceConsole parameters for exporting packages from SVN

Parameter	Value	Description
`-operation`	`SaveVersionSvnContent`	Downloads the package hierarchy as a set of *.zip archives. The `-destinationPath` parameter specifies the content export path in the file system. The `-sourcePath` parameter specifies the SVN repositories.
`-workspaceName`	[*Workspace name*]	The name of the workspace (configuration) where the operation is performed. By default, all users work in the `Default` workspace.
`-destinationPath`	[*Path to a local directory*]	The path to a local directory. The utility will export the *.gz package archives to this directory.
`-workingCopyPath`	[*Path to a local directory*]	The local directory for the working copies of packages stored in SVN.
`-sourcePath`	[*Path to the SVN repository*]	The address of the SVN repository that stores the package structure and metadata. Can accept several comma separated values.
`-packageName`	[*Package name*]	The name of the package from the source SVN repository. The utility will download the packages on which the current package depends as well.
`-packageVersion`	[*Package version*]	The version of the package from the source SVN repository.
`-sourceControlLogin`	[*SVN username*]	The login of the SVN repository user
`-sourceControlPassword`	[*SVN user password*]	The password of the SVN repository user.

`-cultureName`	[*Language culture*]	The language culture code. For example, `en-US`.
`-excludeDependentPackages`	`true` or `false`	The flag that specifies whether to export packages on which the package listed in the `-packageName` parameter depends.
`-logPath`	[*Path to a local directory*]	The utility will save the operation log file to this directory. The file name consists of the operation execution date and time. Optional.

Run the command to export packages from the SVN repository at the Windows command interpreter (console). Structure the command as follows:

```
[Path to WorkspaceConsole]\Terrasoft.Tools.WorkspaceConsole.exe -operation=SaveVersionSvnCont
```

2. Run the utility.

As a result, the utility will export the workspace package archives to the local directory.

4. Import packages

1. Configure the package import command.

WorkspaceConsole parameters for importing packages into the database

Parameter	Value	Description
`-operation`	`InstallFromRepository`	Imports the package content and metadata from *.zip archives into the configuration. If needed, the utility will run the bound SQL scripts, re-generate the source code, install the bound data. Works only with updated or new packages and their elements. Requires either the `-webApplicationPath` or `-configurationPath` parameter. Also requires the `-confRuntimeParentDirectory` parameter.
`-packageName`	[*Package name*]	The package name in the configuration specified in the `-workspaceName` parameter. The utility will download the packages on which the current package

		depends as well. If you omit the parameter, the utility will download all packages of the configuration.
-workspaceName	[*Workspace name*]	The name of the workspace (configuration) where the operation is performed. By default, all users work in the `Default` workspace.
-sourcePath	[*Path to a local directory*]	The path to a local directory. The directory must contain the *.gz package archives to install.
-destinationPath	[*Path to a local directory*]	The path to a local directory. The utility will export the *.gz package archives specified in the `-sourcePath` parameter to this directory.
-skipConstraints	`false`	Skips the creation of foreign keys in database tables. Can be `true` or `false`.
-skipValidateActions	`true`	Skips checking if it is possible to create table indexes when updating the database structure. Can be `true` or `false`.
-regenerateSchemaSources	`true`	Specifies whether to re-generate the source code after saving the packages to the database. Can be `true` or `false`. The default value is `true`.
-updateDBStructure	`true`	Specifies whether to update the database structure after saving the packages. Can be `true` or `false`. The default value is `true`.
-updateSystemDBStructure	`true`	Specifies whether to modify the structure of the system schema database before installing the packages. Also creates the missing indexes in the system tables. Can be `true` or `false`.
-installPackageSqlScript	`true`	Specifies whether to run the SQL scripts before and after saving the packages. Can be `true` or `false`. The default value is `true`.

`-installPackageData`	`true`	Specifies whether to install the bound package data after saving the packages. Can be `true` or `false`. The default value is `true`.
`-continueIfError`	`true`	Specifies whether to abort the installation after receiving the first error. If set to `true`, the installation process will continue until the end. You will be able to review all errors that occurred. Can be `true` or `false`. The default value is `false`.
`-webApplicationPath`	[*Path to a local directory*]	The path to the Creatio installation directory. The utility will follow this path to fetch the database connection information from the `ConnectionStrings.config` file. If you omit this parameter, the utility will connect to the database specified in the configuration string in the utility's configuration file.
`-confRuntimeParentDirectory`	[*Path to a local directory*]	The path to the parent directory of the `..\Terrasoft.WebApp\conf` directory.
`-logPath`	[*Path to a local directory*]	The utility will save the operation log file to this directory. The file name consists of the operation execution date and time. Optional.
`-configurationPath`	[*Path to a local directory*]	The path to the `..\Terrasoft.WebApp\Terrasoft.Configuration` directory. The utility will export the source code and resources of custom package schemas to this directory in the file system development mode.

Run the command to import packages into the database at the Windows command interpreter (console). Structure the command as follows:

```
[Path to WorkspaceConsole]\Terrasoft.Tools.WorkspaceConsole.exe -operation=InstallFromReposit
```

2. Run the utility.
3. Configure the command to generate the static content in the file system.

WorkspaceConsole parameters for compiling the configuration

Parameter	Value	Description
`-operation`	`BuildConfiguration`	Generates the static content in the file system. Requires either the `-webApplicationPath` or `-configurationPath` parameter.
`-workspaceName`	[*Workspace name*]	The name of the workspace (configuration) where the exported packages are specified. By default, all users work in the `Default` workspace.
`-destinationPath`	[*Path to a local directory*]	The path to a local directory. The utility will export the *.gz package archives specified in the `-sourcePath` parameter to this directory.
`-webApplicationPath`	[*Path to a local directory*]	The path to the Creatio installation directory. The utility will use this path to fetch the database connection information from the `ConnectionStrings.config` file. If you omit this parameter, the utility will connect to the database specified in the configuration string in the utility's configuration file.
`-confRuntimeParentDirectory`	[*Path to a local directory*]	The path to the parent directory of the `..\Terrasoft.WebApp\conf` directory.
`-logPath`	[*Path to a local directory*]	The utility will save the operation log file to this directory. The file name consists of the operation execution date and time. Optional.
`-force`	`true` or `false`	Sets the conditions of the file content generation If set to `true`, the utility will generate the file content for all schemas. If set to `false`, the utility will generate the file content for updated schemas. The default value is `false`. Requires either the `-webApplicationPath` or `-configurationPath` parameter.
`-configurationPath`	[*Path to a local directory*]	The path to the `..\Terrasoft.WebApp\`

		`Terrasoft.Configuration` directory. The utility will export the source code and resources of custom package schemas to this directory in the file system development mode.

Run the command to generate static content in the file system at the Windows command interpreter (console). Structure the command as follows:

```
[Path to WorkspaceConsole]\Terrasoft.Tools.WorkspaceConsole.exe -operation=BuildConfiguration
```

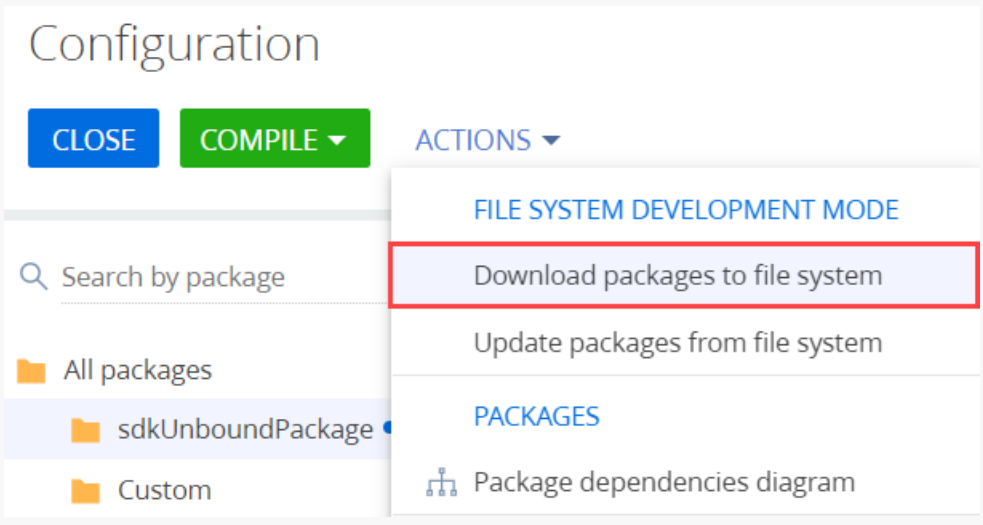
4. Run the utility.

Note. Creatio considers packages imported using the WorkspaceConsole as preinstalled. You cannot modify them.

We do not recommend using the WorkspaceConsole utility for importing packages into the database if the **file system development mode is enabled**. If you use the utility, it will modify the schema source code in the database but not in the file system. As such, if you open the configuration element schema in Creatio IDE, the schema will contain unmodified code from the file system. However, the modification date of the configuration element schema will be updated. This will make the schema falsely appear to have transferred correctly.

To import packages into the database when the **file system development mode is enabled**:

1. Follow this instruction and use the WorkspaceConsole utility to import packages into the database.
2. Select [*Download packages to the file system*] in the [*File system development mode*] action group on the toolbar.



As a result, Creatio will download all packages to the `..\Terrasoft.WebApp\Terrasoft.Configuration\Pkg` path as a set of directories that match package names.

5. Reboot Creatio in IIS

Reboot Creatio in IIS to apply the changes. This is required since the WorkspaceConsole utility makes changes directly in the database.

Export packages from the database

 Medium

Example. Export the packages of the `Default` workspace to a directory.

- `C:\creatio` is the Creatio installation directory.
- `C:\SavedPackages` is the package export directory.
- `C:\Logs` is the export directory of the operation log file.

1. Configure the command for the package export from the database

1. Create a *.bat or *.cmd Windows batch file in a text editor.
2. Add the command that runs the utility to the file.

Command that runs the utility

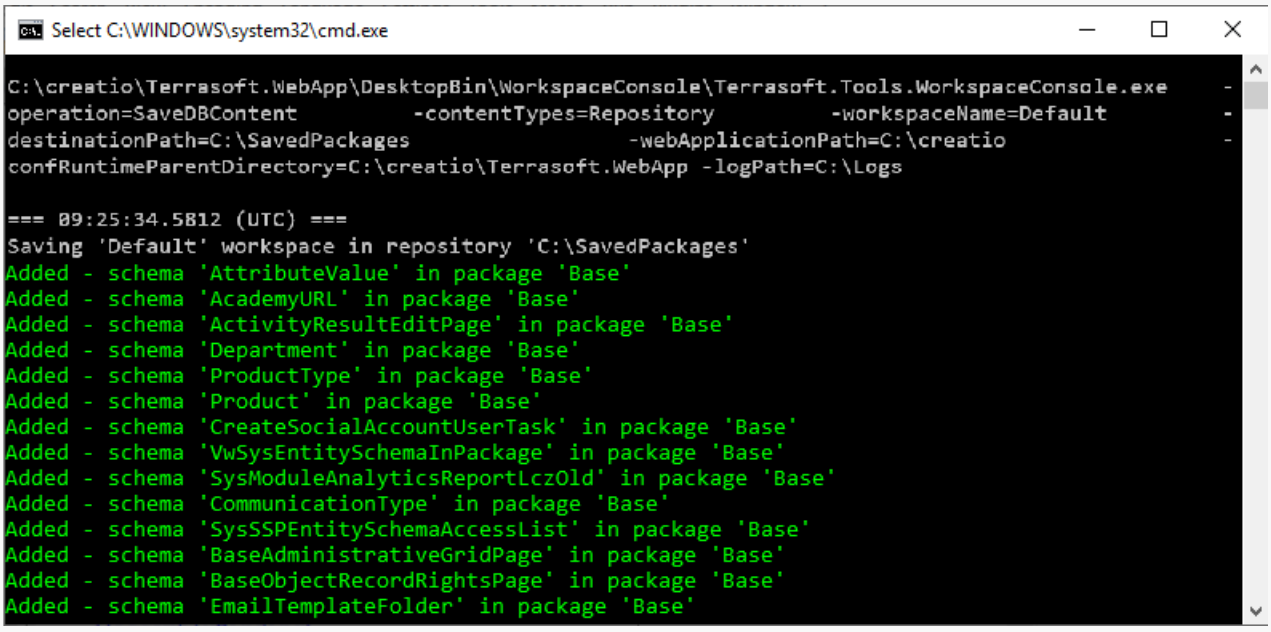
```
C:\creatio\Terrasoft.WebApp\DesktopBin\WorkspaceConsole\Terrasoft.Tools.WorkspaceConsole.exe
pause
```

Save the batch file.

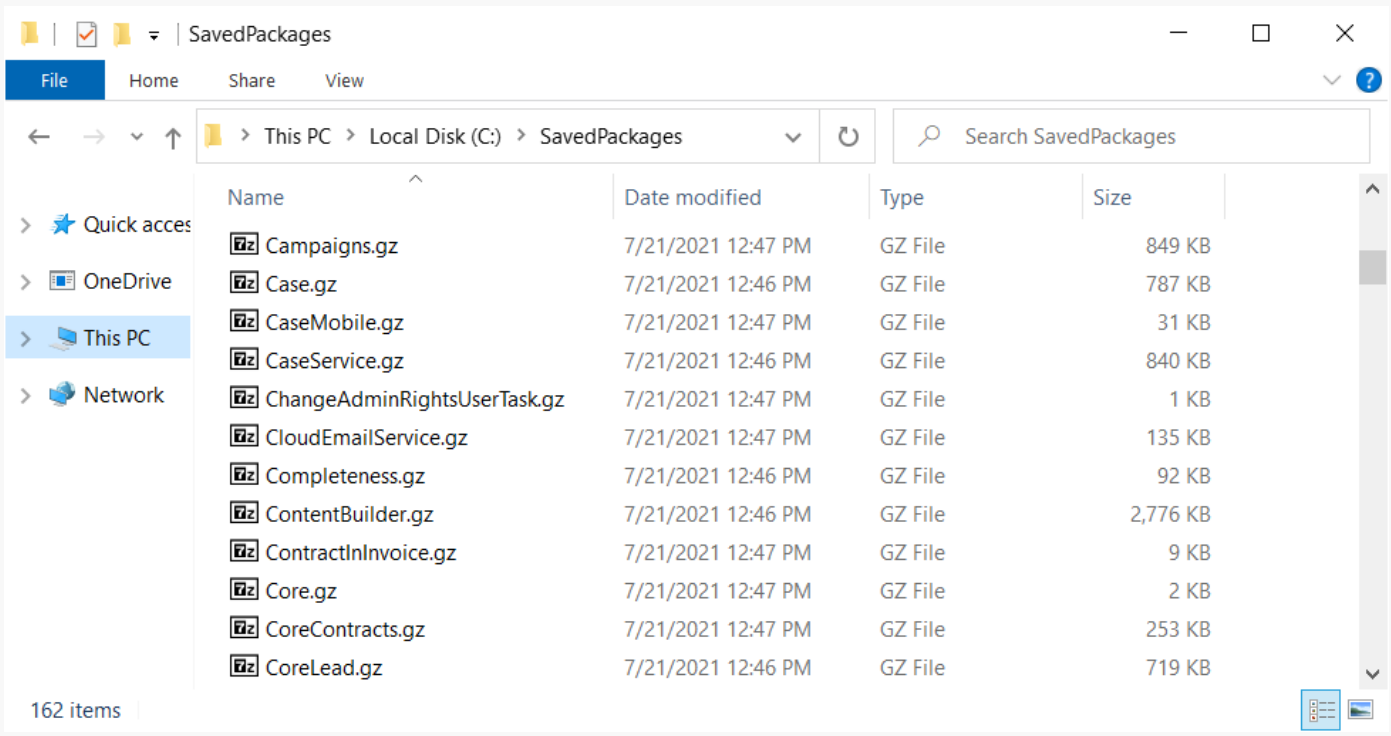
2. Export the package from the database

To **export the package from the database**, double-click the batch file name.

This will open the console that will output the execution process of the operation specified in the corresponding WorkspaceConsole command.



As a result, the utility will export the *.gz package archives of the `Default` configuration to the `C:\SavedPackages` directory.



Export a package from SVN

Medium

Example. Export the package from the SVN repository to the `Default` workspace directory.

- `C:\creatio` is the Creatio installation directory.
- `C:\SavedPackages` is the package export directory.
- `C:\WorkingCopy` is the export directory of the package structure in the SVN repository.
- `http://server-svn:8050/Packages` is the SVN repository URL.
- `sdkTestPackage` is the package to export from the SVN repository.
- `7.18.1` is the version of the package to export from the SVN repository.
- "`User`" is the login of the SVN repository user.
- "`Password`" is the password of the SVN repository user.
- `en-EN` is the language culture.
- `C:\Logs` is the export directory of the operation log file.

1. Configure the command for the package export from the SVN repository

1. Create a *.bat or *.cmd Windows batch file in a text editor.
2. Add the command that runs the utility to the file.

Command that runs the utility

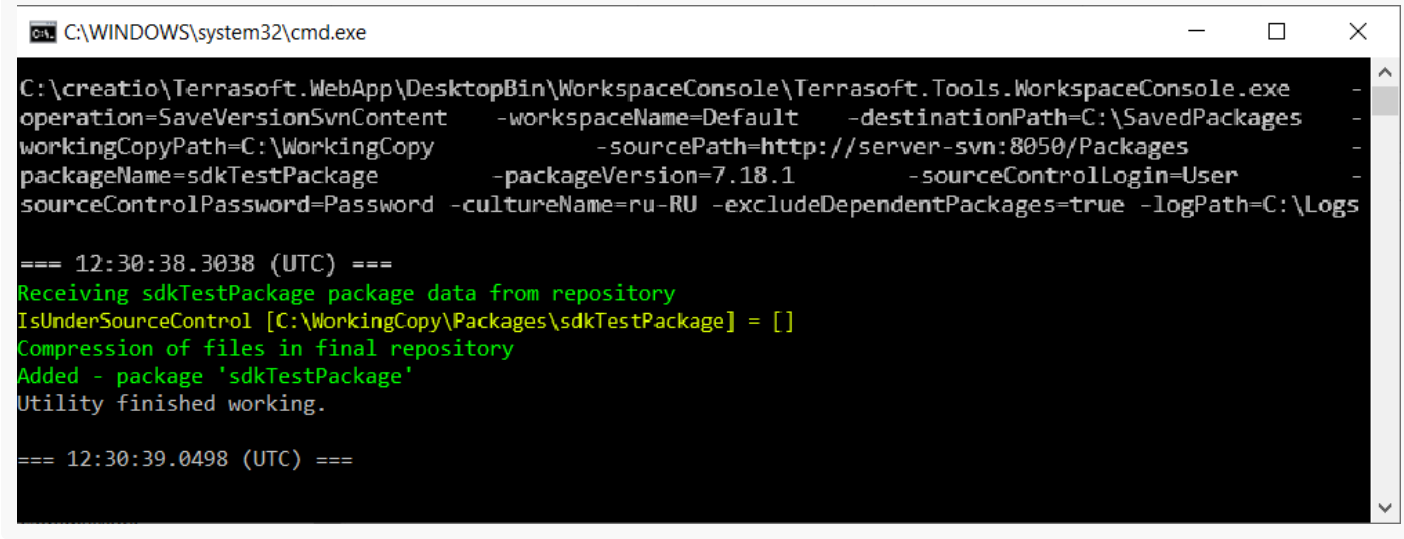
```
C:\creatio\Terrasoft.WebApp\DesktopBin\WorkspaceConsole\Terrasoft.Tools.WorkspaceConsole.exe
pause
```

Save the batch file.

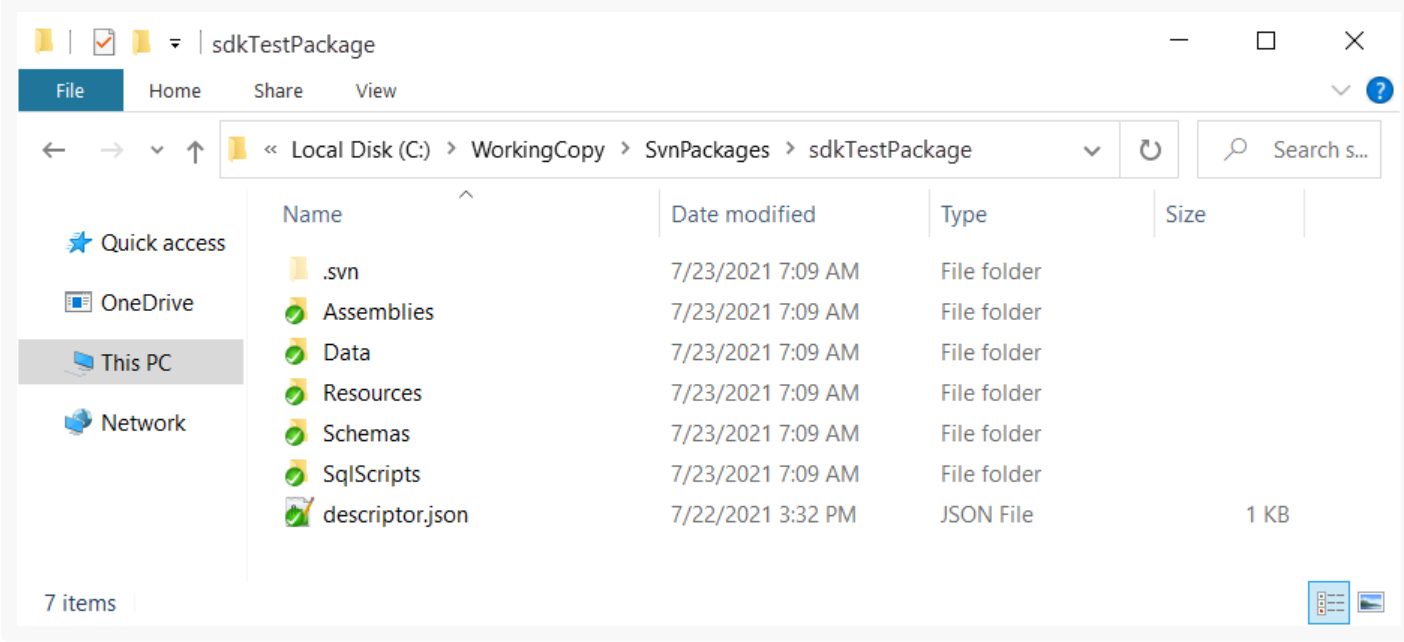
2. Export the package from SVN

To **export the package from SVN**, double-click the batch file name.

This will open the console that will output the execution process of the operation specified in the corresponding WorkspaceConsole command.



As a result, the utility will export the `sdkTestPackage` package of the `Default` configuration to the `C:\SavedPackages` directory. Learn more about the structure of the directory with the package name: Packages basics.



Import a package into the database

Medium

Example. Import the package from the directory into the `Default` workspace.

- `C:\creatio` is the Creatio installation directory.
- `sdkTestPackage` is the package to import into Creatio. The path to the package is `C:\SavedPackages`.
- `C:\SavedPackages` is the package directory.

- `C:\TempPackages` is the package import directory.
- `C:\Logs` is the export directory of the operation log file.

1. Configure the command to import the package into the database

1. Create a *.bat or *.cmd Windows batch file in a text editor.
2. Add the command that runs the utility to the file.

Command that runs the utility

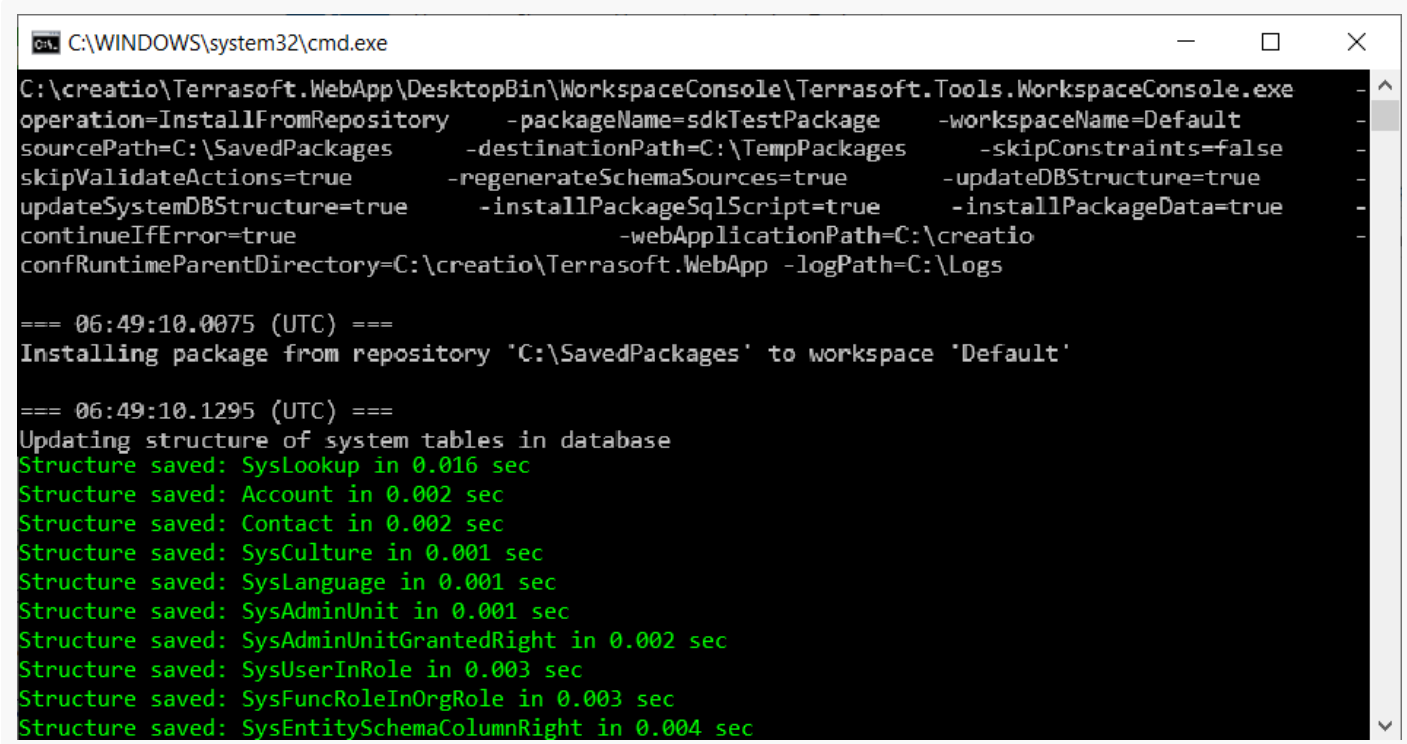
```
C:\creatio\Terrasoft.WebApp\DesktopBin\WorkspaceConsole\Terrasoft.Tools.WorkspaceConsole.exe
pause
```

Save the batch file.

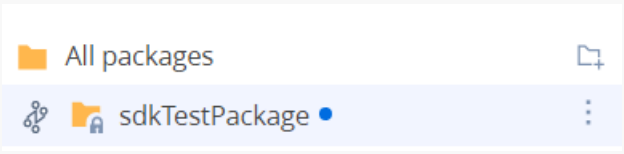
2. Import the package into the database

To **import the package into the database**, double-click the batch file name.

This will open the console that will output the execution process of the operation specified in the corresponding WorkspaceConsole command.



As a result, the utility will import the `sdkTestPackage` package into the `Default` configuration.



3. Configure the command to generate the static content in the file system

1. Create a *.bat or *.cmd Windows batch file in a text editor.
2. Add the command that runs the utility to the file.

Command that runs the utility

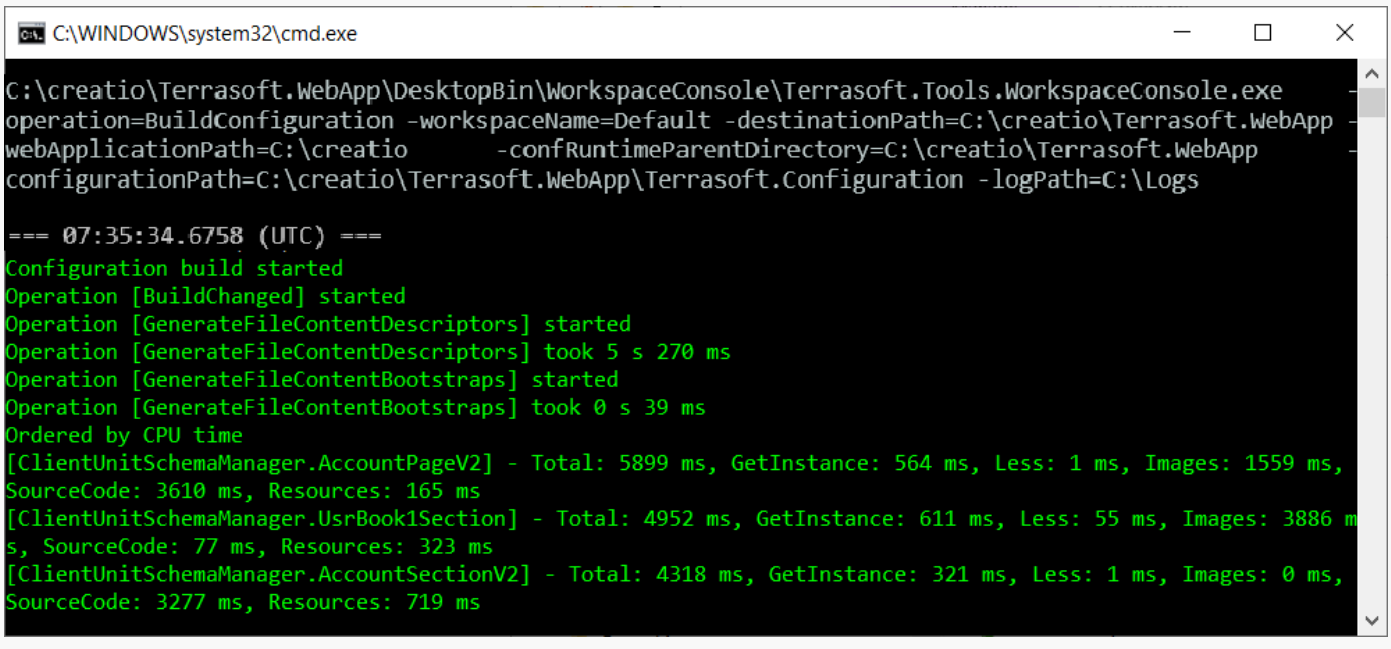
```
C:\creatio\Terrasoft.WebApp\DesktopBin\WorkspaceConsole\Terrasoft.Tools.WorkspaceConsole.exe
pause
```

Save the batch file.

4. Generate the static content in the file system

To **generate the static content in the file system**, double-click the batch file name.

This will open the console that will output the execution process of the operation specified in the corresponding WorkspaceConsole command.



As a result, the utility will generate the static content of the updated schemas in the file system.

WorkspaceConsole utility parameters CLI

Medium

`-help`

Displays the comprehensive list of utility parameters and their brief descriptions. If you specify other parameters, the utility will ignore them.

`-operation`

Name of the operation to execute. Required. The default value is `LoadLicResponse`.

Possible values

`LoadLicResponse`	Uploads licenses to the database specified in the connection string. The only operation that does not require the `-workspaceName` parameter.
`SaveRepositoryContent`	Saves the *.zip package archives specified in the `-contentTypes` parameter from the directory specified in the `-sourcePath` parameter to the directory specified in the `-destinationPath` parameter.
`SaveDBContent`	Saves the database content to the file system. The `-contentTypes` parameter determines the content type. The `-destinationPath` parameter determines the content export path in the file system. Requires either the `-webApplicationPath` or `-configurationPath` parameter.
`SaveVersionSvnContent`	Dumps the package hierarchy as a set of *.zip archives. The `-destinationPath` parameter determines the content export path in the file system. The `-sourcePath` parameter specifies the SVN repositories.
`RegenerateSchemaSources`	Re-generates the source code and compiles it. Requires the `-confRuntimeParentDirectory` parameter.
`InstallFromRepository`	Imports the package content and metadata from *.zip archives into the configuration. If needed, the utility will run the bound SQL scripts, re-generate the source code, install the bound data. Works exclusively with updated or new

	packages and their elements. Requires either the `-webApplicationPath` or `-configurationPath` parameter. Also requires the `-confRuntimeParentDirectory` parameter.
`InstallBundlePackages`	Installs the bundle of packages listed in the `-packageName` comma-separated list to the workspace specified in the `-workspaceName` parameter. Requires either the `-webApplicationPath` or `-configurationPath` parameter. Also requires the `-confRuntimeParentDirectory` parameter.
`PrevalidateInstallFromRepository`	Checks if installing packages from *.zip archives is possible.
`ConcatRepositories`	Merges several repositories.
`ConcatSVNRepositories`	Merges several SVN repositories.
`ExecuteProcess`	Runs the business process in the configuration if the utility detects such a process. Requires the `-confRuntimeParentDirectory` parameter.
`UpdatePackages`	Updates the packages (`-packageName` parameter) in the database. Will only update the packages included in the product package hierarchy (`-productPackageName` parameter). Requires either the `-webApplicationPath` or `-configurationPath` parameter. Also requires the `-confRuntimeParentDirectory` parameter.
`BuildWorkspace`	Compiles the workspace (configuration). Use when developing schemas in Visual Studio. Requires the `-confRuntimeParentDirectory` parameter.
`ReBuildWorkspace`	Recompiles the workspace (configuration). Use when developing schemas in Visual Studio. Requires the `-confRuntimeParentDirectory` parameter.
`UpdateWorkspaceSolution`	Updates the solution and files of the Visual Studio project.
`BuildConfiguration`	Generates the static content in the file system. Uses the following parameters: • `-workspaceName` • `-destinationPath`

	- `-webApplicationPath` - `-logPath` - `-force` - `-configurationPath` Requires either the `-webApplicationPath` or `-configurationPath` parameter.

Attention. Execute the `BuildConfiguration` operation after executing the `InstallFromRepository`, `InstallBundlePackages`, `UpdatePackages` operations to ensure Creatio works correctly.

`-user`

Username for authorization. Specify the value if there is no username in the utility's configuration file or you need to execute the operation on behalf of a different user.

`-password`

User password for authorization. Specify the value if there is no user password in the utility's configuration file or you need to execute the operation on behalf of a different user.

`-workspaceName`

Name of the workspace (configuration) where the exported packages are specified. By default, all users work in the `Default` workspace.

`-autoExit`

Specifies whether to end the utility process automatically after the operation finishes. Can be `true` or `false`. The default value is `false`.

`-processName`

Name of the process to run.

`-repositoryUri`

Address of the SVN repository that stores the package structure and metadata. Optional. If you specify the parameter, the utility will use its value instead of the value of the `-workspaceName` parameter.

`-sourceControlLogin`

Login of the SVN repository user.

`-sourceControlPassword`

Password of the SVN repository user.

`-workingCopyPath`

Local directory of the working copies of packages stored in SVN.

`-contentTypes`

Type of content exported to the drive from the database.

Possible values

`SystemData`	JSON data of system schemas. Dumps all system schemas and their columns except for data specified in the `-excludedSchemas` parameter.
`ConfigurationData`	JSON data of configuration element schemas. Dumps all schemas except for data specified in the `-excludedSchemas` parameter.
`Resources`	Localizable XML resources of configuration element schemas.
`LocalizableData`	Localizable XML content of configuration element schemas. Dumps only the text columns. Specify additional restrictions in the `-excludedSchemas` and `-excludedSchemaColumns` parameters.
`Repository`	ZIP workspace data.
`SqlScripts`	SQL scripts bound to packages.
`Data`	JSON data of system schemas and configuration element schemas. The combination of `SystemData` and `ConfigurationData` values.
`LocalizableSchemaData`	Data of localizable objects.
`All`	All content types.

`-sourcePath`

Path to a local directory from which to fetch the data. For example, packages, schemas, or resources. Can

accept several values separated by commas for the `ConcatRepositories` and `SaveVersionSvnContent` operations.

`-destinationPath`

Path to a local directory to save the data. For example, packages, schemas, or resources.

`-webApplicationPath`

Path to the Creatio installation directory. The utility will follow this path to fetch the database connection information from the `ConnectionStrings.config` file. If you omit this parameter, the utility will connect to the database specified in the configuration string in the utility's configuration file.

The `-webApplicationPath` parameter specifies the path to the `Terrasoft.WebApp` directory for the `BuildWorkspace`, `ReBuildWorkspace`, and `UpdateWorkspaceSolution` operations.

`-configurationPath`

Path to the `..\Terrasoft.WebApp\Terrasoft.Configuration` directory. The utility will export the source code and resources of custom package schemas to this directory in the file system development mode.

`-filename`

File name. Required for the `LoadLicResponse` operation.

`-excludedSchemas`

Names of configuration element schemas to exclude from the operation.

`-excludedSchemaColumns`

Names of the columns of configuration element schemas to exclude from the operation.

`-excludedWorkspaceNames`

Names of the workspaces (configurations) to exclude from the operation.

`-includedSchemas`

Names of configuration element schemas to forcibly include in the operation.

`-includedSchemaColumns`

Names of the columns of configuration element schemas to forcibly include in the operation.

`-cultureName`

Language culture code. Required if you specify the `Resources` and/or `LocalizableData` values in the `-contentTypes` parameter.

`-schemaManagerNames`

Names of configuration element schema managers in the operations. The default value is `EntitySchemaManager`.

`-packageName`

Package name in the configuration specified in the `-workspaceName` parameter. The utility will download the packages on which the current package depends as well. If you omit the parameter, the utility will download all packages of the configuration.

`-clearWorkspace`

Specifies whether to clear the workspace before the update. Can be `true` or `false`. The default value is `false`.

`-installPackageSqlScript`

Specifies whether to run the SQL scripts before and after saving the packages. Can be `true` or `false`. The default value is `true`.

`-installPackageData`

Specifies whether to install the bound package data after saving the packages. Can be `true` or `false`. The default value is `true`.

`-updateDBStructure`

Specifies whether to update the database structure after saving the packages. Can be `true` or `false`. The default value is `true`.

`-regenerateSchemaSources`

Specifies whether to re-generate the source code after saving the packages to the database. Can be `true` or `false`. The default value is `true`.

`-continueIfError`

Specifies whether to abort the installation after receiving the first error. If set to `true`, the installation process will

continue until the end. You will be able to review all errors that occurred. Can be `true` or `false`. The default value is `false`.

Always set the `-continueIfError` parameter to `false` for the `InstallFromSvn` and `InstallFromRepository` operations. These operations manage updated or new packages and their elements. The utility detects the updated elements by comparing the new and existing package structures. As such, if you run the command without specifying the `-continueIfError=true` key and an error occurs, then rerun the command for the same configuration, the second command will finish without errors yet will not update the database. In this case, the first operation synchronizes the package structures of the configuration and repository, meaning the second operation will be unable to detect the updated package elements.

`-skipCompile`

Specifies whether to run the compilation. Will work if you set the `-updateDBStructure` parameter to `false`. Can be `true` or `false`. The default value is `false`.

`-autoUpdateConfigurationVersion`

Updates the configuration version in the database to the Creatio version. Can be `true` or `false`. The default value is `false`.

`-warningsOnly`

Specifies whether to display an error notification if the utility detects an operation execution error. Can be `true` or `false`. The default value is `false`.

`-confRuntimeParentDirectory`

Path to the parent directory of the `..\Terrasoft.WebApp\conf` directory. You must use the parameter in commands that compile Creatio or generate the static content:

- `RegenerateSchemaSources`
- `InstallFromRepository`
- `InstallBundlePackages`
- `UpdatePackages`
- `BuildWorkspace`
- `RebuildWorkspace`
- `ExecuteProcess`