Payroll Engine Console Application

👉 This application is part of the Payroll Engine.

The Payroll Console application provides API-like commands. See the Payroll Engine samples and tests for examples of how to use this tool. For a better understanding of the working concepts, it is recommended to read the Payroll Engine Whitepaper.

The application is controlled by two types of command line arguments:

- Command: a single command
- Command file: a file containing several commands.

Commands

The Payroll Console provides the following commands:

Command	Group	Description
`ActionReport`	Action	Report custom actions from an assembly
`CaseTest`	Payroll	Test case availability, build data and user input validation
`ChangePassword`	System	Change a user password
`DataReport`	Report	Report data to JSON file
`Help`	System	Show the command reference

`HttpGet`
`HttpPost`
`HttpPut`
`HttpDelete`
	System	Execute HTTP GET, POST, PUT or DELETE request

`LogTrail`	System	Trace the tenant log [1)]
`PayrollExport`	Payroll	Export any payroll data to JSON file
`PayrollImport`	Payroll	Import any payroll data from JSON/zip file
`PayrollImportExcel`	Payroll	Import payroll data from Excel file
`PayrollResults`	Payroll	Report payroll data to screen and/or file
`PayrunEmployeeTest`	Test	Execute employee payrun and test the results
`PayrunJobDelete`	Data management	Delete payrun job with payroll results
`PayrunRebuild`	Script	Rebuild payrun
`PayrunStatistics`	Statistics	Display payrun statistics
`PayrunTest`	Test	Execute payrun and test the results

Command	Group	Description
`RegulationRebuild`	Script	Rebuild the regulation objects
`RegulationShare`	Regulation share	Manage regulation shares
`Report`	Report	Report to file [2]
`ReportTest`	Test	Test report output data
`ScriptExport`	Script	Export regulation scripts to folder
`ScriptPublish`	Script	Publish scripts from C# file
`Stopwatch`	System	Stopwatch based on environment user variable
`TenantDelete`	Data management	Delete tenant
`UserVariable`	System	View and change the environment user variable
`Write`	System	Write to the console and/or log file

[1] Tenant logs are generated by the regulations and should not be confused with the application log.

[2] Based on FastReports.

The required command parameters and options/toggles are passed after the command name. An example of how to bulk import a payroll from a JSON file:

```
PayrollConsole PayrollImport MyPayroll.json /bulk
```

The `Help` command describes the commands and their parameters:

```
PayrollConsole Help PayrollImport
```

Command Files

The command file is a file with the extension `.pecmd` that contains a series of instructions to the Payroll Engine Console. The lines of the file are executed sequentially and contain the following line types

- Comment starting with `#`
- Command instruction
- Starts another command file

Example command file `Delete.pecmd` with a comment and a command statement:

```
# delete my tenant
TenantDelete MyTenant /trydelete
```

The following example uses the `Delete.pecmd` command file and then runs the command to test the payrun:

```
# clranup and execute test
Delete.pecmd
PayrunTest *.pt.json
```

Command files access path

If the called command file is in a different folder, it will be activated before execution:

```
Test1/Test.pecmd
Test2/Test.pecmd
```

This is necessary if the called command file is to be defined with local references. The `keepPath` option prevents directories from being changed:

```
Tools/Import1.pecmd /keepPath
Tools/Import2.pecmd /keepPath
```

In the example above, both import command files are run in the startup folder.

Command file parameters

Command files can also be controlled by parameters. Such parameters are used with the placeholder `$Name$` in the commands. In the following example, the command file `OwnerTest.pecmd` has the parameter `Owner`:

```
# payrun employee test
# argument owner: job owner
PayrunEmployeeTest fileMask:*.et.json owner:$owner$ /wait
```

Calling this command file with the `Owner` parameter looks like this:

```
OwnerTest.pecmd owner:Test01
```

Command files in the operating system

Command file association in Windows:

1. Open File Explorer (right click Start -> File Explorer)
2. Find the file you want to associate
3. Right click the file and select `Properties`
4. In this window click `Opens With: Change...`

5. Select the payroll console program

Command file association in MacOS:

1. On your Mac, click the Finder icon in the Dock to open a Finder window.
2. Select a file, then choose `File` > `Get Info`.
3. You can also Control-click the file, then choose `Get Info`.
4. In the Info window, click the arrow next to `Open with`.
5. Click the pop-up menu, choose an app, then click `Change All`.

> Note: Don't click `Change All` if you only want the specific file you selected to open with that app.

Command file association in Linux:

1. TODO

Global Toggles

The following toggles apply to all Payroll Console commands:

Name	Description	Values	Command file
Display level	Command information level	`full` [1], `compact`, `silent`	Preset for executing commands
Error mode	Show failed tests and errors	`errors`[1], `noerrors`	Preset for executing commands
Wait mode	Wait at the program end	`waiterror`[1], `wait`, `nowait`	Final application wait mode
Path mode	Path change mode	`switchpath`[1], `keeppath`	Only for command files

[1] Default value is the first entry.

Application Extensions

The PAyroll Console provides a plug-in mechanism for integrating custom commands:

- Command parameters with names and options
- Access to the Payroll REST API via the Payroll HTTP client
- Control output to console and logger
- Program exit code definition

To develop a Payroll Console extension, do the following

1. Develop the extension library in C# (example PayrollEngine.Client.Tutorials).
2. Copy the output DLL of the library into the `extensions` subfolder of the application.
3. Display the command help `PayrollEngine Help MyCommandName`.
4. Run the command with `PayrollEngine MyCommandName`.

Application Configuration

The Payroll Console configuration `appsetings.json` contains the following settings:

Payroll Console Configuration

Setting	Description	Default
`StartupCulture`	The payroll console process culture (string)	System culture

Payroll Http Configuration

Setting	Description	Default
`BaseUrl`	The backend base URL (string)	
`Port`	The backend url port (string)	
`Timeout`	The backend request timeout (TimeSpan)	100 seconds

Serilog

File and console logging with Serilog.

> It is recommended that you save the application settings within your local User Secrets.

Application Logs

Under Windows, the payroll console stores its logs in the system folder `%ProgramData%/PayrollConsole/logs`.

Third party components

- Excel conversion with NPOI - licence `Apache 2.0`
- Logging with Serilog - licence `Apache 2.0`