Welcome to SpecFlow+ Excel's documentation!

Oct 15, 2021

SpecFlow+ Excel

SpecFlow+ Excel is only available for SpecFlow 2.4 and earlier.

SpecFlow+ Excel is a SpecFlow plugin that allows requirements to be described in Excel files. These requirements can be used in a same way as normal plain text Gherkin feature files.

CHAPTER 1

Installation & Configuration

1.1 Installation

The SpecFlow+ Excel plugin is available from NuGet.

To add the package to your Visual Studio project:

1. Right-click on your specification project and select **Manage Nuget Packages for Solution**.
2. Search for "SpecFlow Excel" on nuget.org and install the SpecFlow+ Excel plugin

Alternatively, you can install the package from the NuGet console (**Tools | NuGet Package Manager | Package Manager Console**) as follows:

```
Install-Package SpecFlow.Plus.Excel -ProjectName myproject
```

1.2 Configuration

When installing the NuGet package, the necessary configuration steps are performed. You can verify that the SpecFlow+ Excel plugin is configured correctly for SpecFlow in the `App.config` file, which should contain the following section:

```
<specFlow>
  <plugins>
    <add name="SpecFlow.Plus.Excel" type="Generator" />
  </plugins>
</specFlow>
```

CHAPTER 2

Working with generated files

The SpecFlow+ Excel NuGet package configures the project to use SpecFlow's build time generation feature by adding an MsBuild target to the project file. This ensures that the tests in the Excel files are re-generated when necessary. This build time generation works with the normal generation process triggered by saving feature files in Visual Studio. A positive side-effect of this is that you no longer need to store the generated files in your source control! You can configure your source control system to ignore the following patterns (e.g. by adding these lines to `.gitignore`):

- `*.feature.cs`
- `*.feature.xlsx.*`

Note: The current version of SpecFlow+ Excel uses an updated version of the SpecFlow MsBuild integration. This updated version is included in the `SpecFlow.Plus.Excel` NuGet package and configures the project to use this updated version. If you previously enabled build time generation with SpecFlow, update the referenced target file to point to the one from this package.

CHAPTER 3

Excel Features

The SpecFlow+ Excel plugin allows you to describe your features in Excel files instead of plain text Gherkin feature files. Just like normal feature files, these Excel feature files can be added to your project, allowing you to execute all its scenarios as executable tests. Refer to SpecFlow+ Excel Getting Started for a quick introduction.

As the Excel feature file has to represent the same structure as a pain text feature file, you have to follow some rules when editing these Excel files. These rules are designed so as not to be too restrictive and conform to the normal way of working in Excel (e.g. you can worksheets to represent scenarios or scenario outlines). For a more detailed breakdown of these rules, see Excel feature file format. You may also want to take a look at this sample Excel feature file.

3.1 Converting Plain Text Feature Files to Excel Files

SpecFlow+ Excel's command line tool allows you to generate an initial Excel file from an existing feature file. You can also use this to experiment with how the different Gherkin elements are represented in Excel. For more details, see Command-line tool reference.

3.2 Converting Excel Cell Values

Cell values, in particular non-string values like dates or numbers, are converted using a specific culture when executing tests. Cell formatting options in Excel (e.g. number of decimal places, rounding, currency signs, date formats) are ignored. See Converting Excel Cells for more details.

3.3 Multi-language Support

This documentation uses the English Gherkin keywords (e.g. `Given`, `When`, `Then` and `Scenario Outline`), but SpecFlow+ Excel supports all 50+ languages supported by Gherkin itself. You can therefore create Excel feature files in the language of your own domain!

CHAPTER 4

Excel Feature File Format

Note: While English Gherkin keywords are used in this documentation, the SpecFlow+ Excel plugin supports all 50+ languages supported by Gherkin itself. You can also use the various variations as well (e.g. `Scenario Outline` and `Scenario Template`).

We recommend checking out our sample Excel feature file first to get an idea of how the files are structured.

4.1 Naming Feature Files

Excel feature file must have the `.feature.xlsx` extension, e.g. `Calculator.feature.xlsx`. SpecFlow's build-time generation can only convert Excel files in your project with this extension.

4.2 Cell Content

The content in your Excel files is converted by SpecFlow+ Excel using the following rules:

- If a cell contains a formula, the result of the formula is used
- Formatting (e.g. currency or date format) is ignored
- The binding culture is used to convert cell values to strings for your tests
- The binding culture is the default Gherkin language of your project, but can overridden. See SpecFlow documentation for more details.

Note: Gherkin tables do not support merged cells. If your Excel table contains merged cells, the first of the merged cells will contain the value; the other merged cells will be empty.

4.2.1 Ignored Content

Certain content in your Excel files is ignored by SpecFlow+ Excel. You can use this content for comments, helper calculations etc.

The following is ignored:

- Data on hidden sheets or sheets whose name begins with an underscore ('_')
- Cell values where there are at least 2 empty cells to the immediate left of the cell (i.e. use two empty columns to separate comments from data). This includes rows where the first two cells are empty.
- Empty rows

4.3 Defining Steps

Each step must be defined on a separate row. The step can be split up over multiple cells in the same row, in which case it is treated as though there is a space between the content of neighbouring cells.

4.4 Using Formulas

You can use Excel formulas anywhere in the document. The plugin uses the result of the formula, so you can use formulas to calculate test data values.

4.5 Using Tables

Cell ranges can be used to specify Gherkin tables. These cell ranges must to be "indented" by one column, ie. the first cell has to be left empty.

4.6 Gherkin Equivalents in Excel

The contents of an Excel file are converted to the Gherkin syntax automatically. The contents of the Excel file are mapped to the Gherkin syntax as follows:

CHAPTER 5

Excel Examples

The SpecFlow+ Excel plugin allows you to extend scenario outlines in plain text with Excel tables. Add the Excel files containing these test examples to your project so that the examples can be used by your executable tests. See SpecFlow+ Excel Getting Started for a quick introduction.

The sample feature file and the related Excel file include additional examples.

5.1 Using the Same Excel File to Describe Multiple Example Sets

One Excel examples file can be used to describe multiple example sets or even multiple scenario outlines. In this case, each worksheet should contain a single example set. In general, you should create an Excel examples file for each feature file. You can specify the name of the Excel sheet explicitly in the feature file, but this is not necessary if you name your sheets the same as your scenario outline titles. For more details, see Prepare Feature Files for External Examples.

5.2 Converting Excel Cell Values

Cell values, in particular non-string values like dates or numbers, are converted using a specific culture when executing tests. Cell formatting options in Excel (e.g. number of decimal places, rounding, currency signs, date formats) are ignored. See Converting Excel Cells for more details.

5.3 Prepare Feature Files for external Examples

In order to use SpecFlow+ Excel to extend your scenario outlines with additional examples, you need to extend the scenario outline examples blocks with a special `@source` tag as follows:

```
@source:excel-file-path[:sheet-name]
```

The `excel-file-path` is the path to the Excel file, relative to the feature file itself.

The `sheet-name` is optional. If no worksheet name is specified, the plugin uses the Excel sheet with the same name as the scenario outline, or the first worksheet in the file if there is no worksheet with a matching name.

The `Examples` block must contain a header row. The header row needs to match in both the feature file and the Excel file (same columns, same order).

You can also define concrete examples in the feature file. In this case, the examples from the Excel file are merged with the examples defined in the feature file. This sample feature file is extended with additional examples by this Excel file.

```
Feature: Calculator

Scenario Outline: Add two numbers
        Given I have entered <a> into the calculator
        And I have entered <b> into the calculator
        When I press add
        Then the result should be <result> on the screen

@source:CalculatorExamples.xlsx
Examples:
        | case | a | b | result |
```

CHAPTER 6

Excel Examples File Format

6.1 General Rules

We recommend that you take a look at this sample feature file and its related Excel file, which includes additional examples for the feature file, before studying rules in detail.

- Values (in particular non-string values) entered in cells are converted according to the general cell conversion rules.
- You can use Excel formulas in the Excel file. The plugin uses the result of the formula result to execute your tests.
- Empty Excel rows are ignored.
- Merged cells: Gherkin tables do not support merged cells, so merged cells are converted so that the first cell contains the value and the other cells are empty.
- The first row in the sheet is the examples header row, and must match to the header row specified in the feature file (same columns, same order).

6.2 Excel examples matching to the Gherkin scenario outline examples

CHAPTER 7

Converting Excel Cells

The SpecFlow+ Excel uses the following rules to convert cell values:

- If a cell contains a formula, the result of the formula is used
- Formatting (e.g. currency or date format) is ignored
- The binding culture is used to convert cell values to strings for your tests
- The binding culture is the default Gherkin language of your project, but can overridden. See SpecFlow documentation for details.

CHAPTER 8

Command Line Tool Reference

You can invoke SpecFlow+ Excel's features using the command line tool located in the `tools` folder of the NuGet package.

If you start the tool without any parameters, the available options are summarised:

```
SpecFlow.Plus.Excel.Converter.exe
```

For help about a specific command, use the `help` command:

```
SpecFlow.Plus.Excel.Converter.exe help convert
```

The command line tool also works on non-Windows machines using Mono. In this case you need to invoke the tool through Mono:

```
mono SpecFlow.Plus.Excel.Converter.exe
```

8.1 Available commands

- `convert`: Converts an Excel feature file (.feature.xlsx) to a plain text Gherkin file
- `initialize`: Initializes an Excel feature file (.feature.xlsx) based on a plain-text Gherkin file (one-time conversion)
- `about`: Displays information about the tool and your license, including the licensee, expiry date and the date until which upgrades are included
- `register`: Registers a license
- `unregister`: Unregisters the license

8.2 Convert

Use the `convert` command to transform an Excel feature file to plain text Gherkin file.

Usage:

```
SpecFlow.Plus.Excel.Converter.exe convert excelFilePath outputFeatureFilePath [/
↪lang:value] [/culture:value]
```

- `excelFilePath`: The Excel file's path
- `outputFeatureFilePath`: The generated Gherkin feature file's target path
- `[/lang:value]`: The Gherkin language used to convert the Excel file; **default:** en-US
- `[/culture:value]`: The culture used to format cell values (e.g. dates); **default: same** as `lang`. For more details on how values are converted, see Converting Excel Cells.

8.3 Initialize

Use the `initialize` command to create an Excel feature file from an existing plain text feature file. You can also use this option to experiment with how different Gherkin elements are represented in Excel.

Usage:

```
SpecFlow.Plus.Excel.Converter.exe initialize featureFilePath outputExcelFilePath [/
↪lang:value]
```

- `featureFilePath`: The feature file's path
- `outputExcelFilePath`: The generated Excel file's target path
- `[/lang:value]`: The default Gherkin language used to convert the feature file; **default:** en-US

8.4 Register

Use the `register` command to register your license key. You only need to register your license once for all SpecFlow+ components. Use the `about` command to display your current license status.

Usage:

```
SpecFlow.Plus.Excel.Converter.exe register licenseKey issuedTo
```

- `licenseKey`: The license key
- `issuedTo`: The exact 'issued to' value of the license. If you ordered SpecFlow+ via SWREG, this is the mail address used to purchase SpecFlow+. If you purchased SpecFlow+ directly from TechTalk, this value is specified in the email you received with your license details. **Note:** If your 'issued to' value contains spaces, enclose it with quotation marks

Example:

```
SpecFlow.Plus.Excel.Converter.exe register KGg45...A== "John Doe"
```