rt

2023

EasySave

Prosoft

23/02/2023

Technical documentation

**Une image contenant outil

Description générée automatiquement**

Table of contents

[1. Introduction 2](#_Toc128052261)

[2. Available operating systems 2](#_Toc128052262)

[3. Langages configuration 3](#_Toc128052263)

[4. Technologies used 4](#_Toc128052264)

[5. Program structure 4](#_Toc128052265)

[6. Log files 5](#_Toc128052266)

[7. Window Client 6](#_Toc128052267)

# Introduction

This documentation is intended to list all the technical details of EasySave V3.

This document will cover details such as the operating systems on which EasySave runs, the structure of the application and the configuration of the languages.

# Available operating systems

As for the operating systems on which EasySave can run, it is quite general, by "general" we mean that we can generate a version of the application for each operating system.

The application has been generated for a 64-bit Windows system, but we could have generated a version for Linux or Mac OS systems. To access another version (other than Windows) you will have to ask our sales department to generate a version for another operating system.

Une image contenant texte

Description générée automatiquementHere is the list of available operating systems:

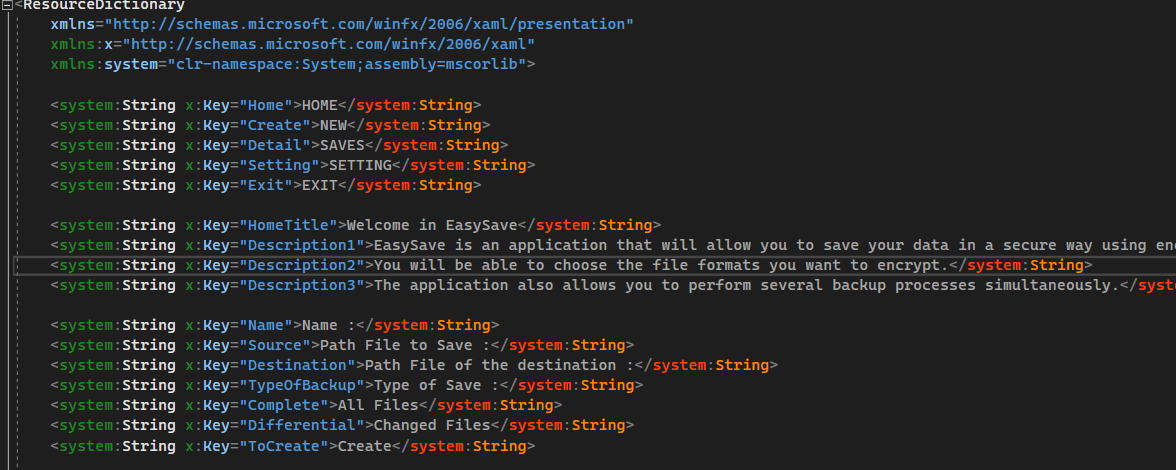
# Langages configuration

As for the translation of the application, it is managed by .xaml files organised as follows:

Une image contenant texte, orange

Description générée automatiquement

Each "Language" file modifies all the languages ​​in the application.

Here is an example of the "LanguageFr" file:

To add new translations you just need to add a new .xaml, for example for German you would have: LanguageDe.xaml.

# Technologies used

For the development of this application the technologies used are :

* The .NET 6.0 framework
* The C# language
* The JSON format
* The XML format
* WPF interface
* Github for versioning and collaborative work
* Visual Studio 2022 IDE

# Program structure

The structure of the program is based on the MVVM model.

Here is the architecture of the solution:

You can find the View and the ViewModels in the easy\_save.Desktop project which contains the whole WPF interface of the application.

Finally we have easy\_save.Desktop which contains the entire WPF interface of the application.

Une image contenant texte

Description générée automatiquement

For the details of the code structure please refer to the UML diagrams.

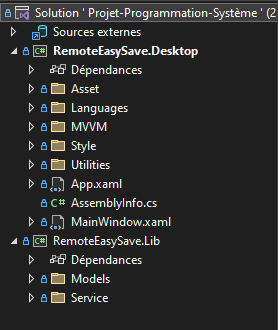
# Log files

Here are all the different log files.

|  |  |
| --- | --- |
| State log file  json | Une image contenant texte  Description générée automatiquement |
| State log file  xml | Une image contenant texte  Description générée automatiquement |
| Daily log file  json |  |
| Daily log file  xml | Une image contenant texte  Description générée automatiquement |

# Window Client

The EasySave application also includes a client project that allows to deport a window to access only the process control part.



The structure of the solution is similar to that of the initial project.

With RomoteEasySave.Desktop which contains all the interface in WPF, then the party RemoteEasySave.Lib containing the templates and served for the client.

Note that we use sockets to realize this remote window.

You can also change the IP and Port of connection between the server and the client from the app.config files of the Client and the Server:

* \* App.Config Server:

Une image contenant texte

Description générée automatiquement

* Une image contenant texte

  Description générée automatiquementApp.Config Client :