RTF to HTML .Net

(Multi-platform .Net library)

SautinSoft

Linux development manual

Table of Contents

1. Preparing environment 2

1.1. Check the installed Fonts availability 3

2. Creating “Convert RTF/DOCX to HTML” app 5

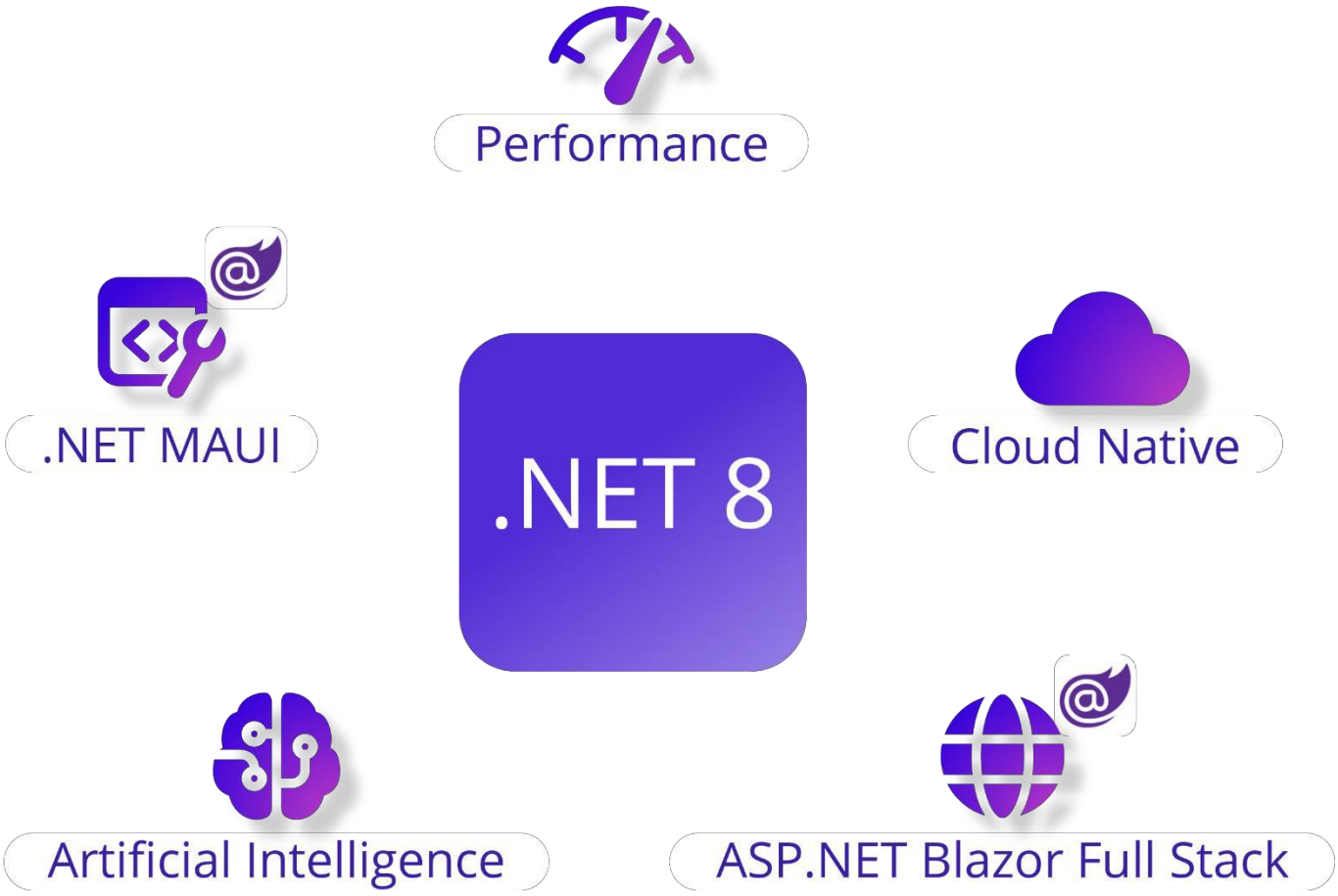
1. Preparing environment

In order to build multi-platform applications using .NET on Linux, the first steps are for installing in our Linux machine the required tools.

We need to install .NET SDK from Microsoft and to allow us to develop easier, we will install an advance editor with a lot of features, Visual Studio Code from Microsoft.

Both installations are very easy and the detailed description can be found by these two links:

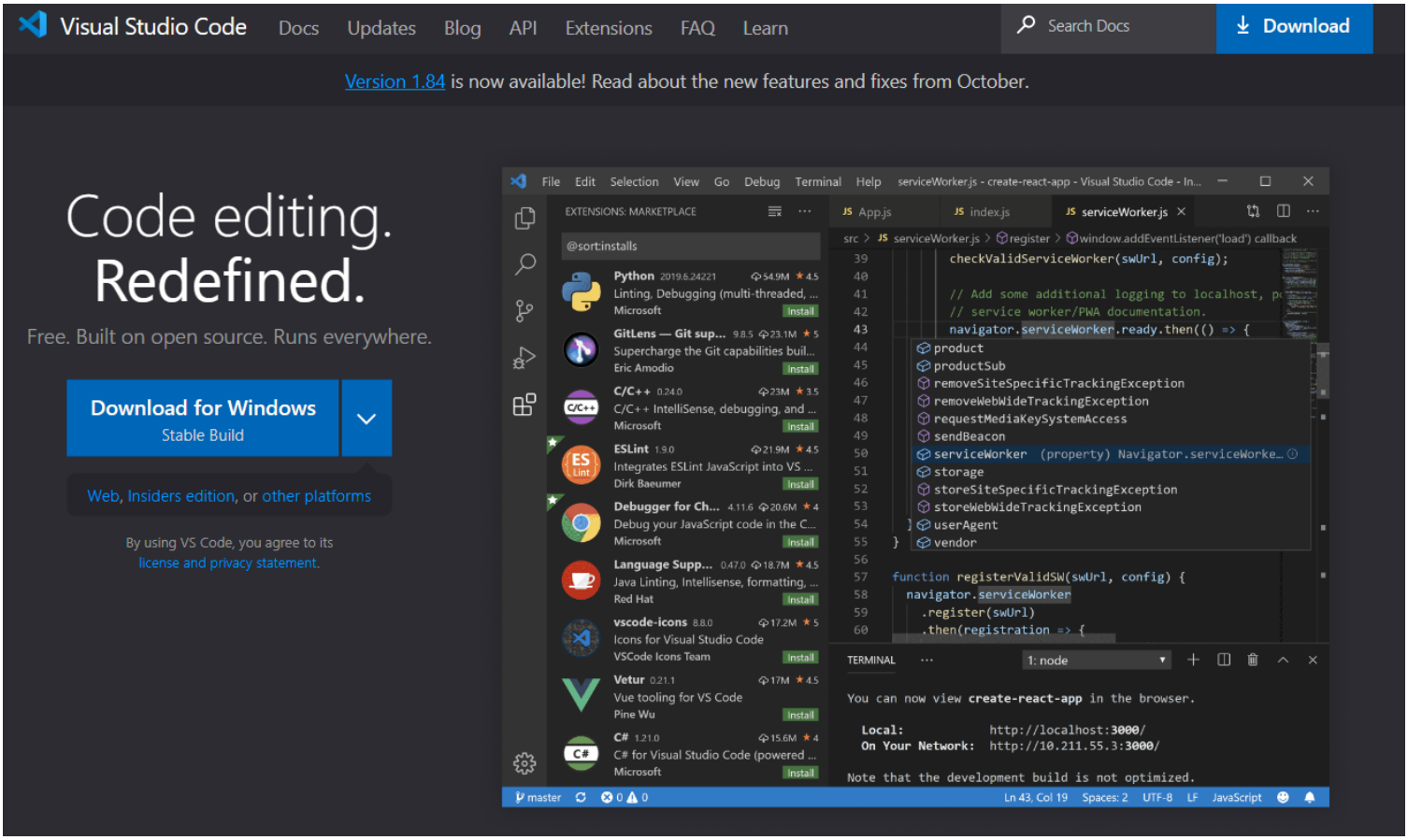
Install .NET SDK for Linux.



Install VS Code for Linux.

Once installed VS Code, you need to install a C# extension to facilitate us to code and debugging:

Install C# extension.



1.1. Check the installed Fonts availability

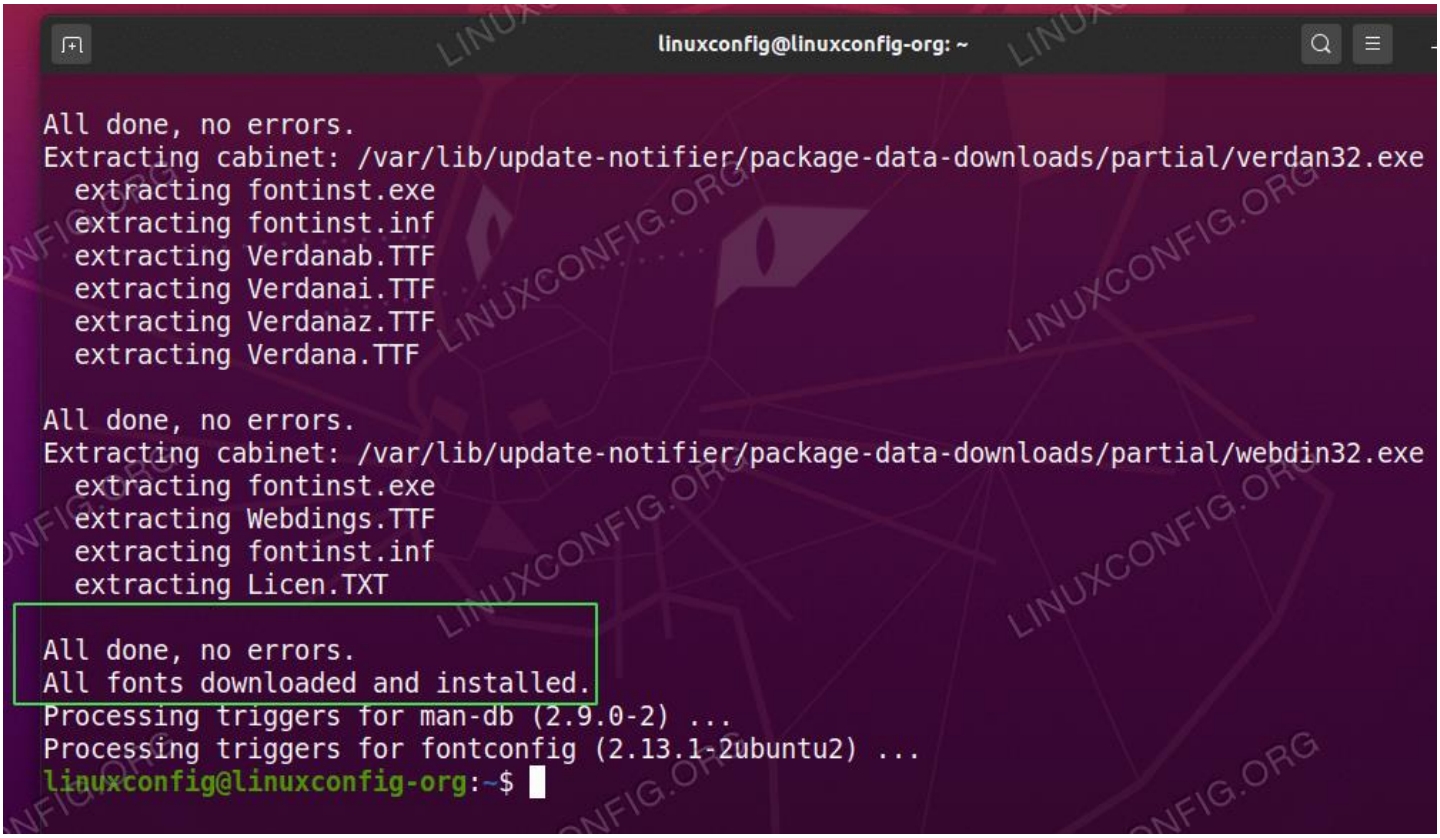
Check that the directory with fonts "/usr/share/fonts/truetype" is

exist. Also check that it contains *.ttf files.

If you don't see this folder, you may install “Microsoft TrueType core fonts” using

terminal and command:

```
$ sudo apt install ttf-mscorefonts-installer
```



Read more about [TrueType Fonts and "How to install Microsoft fonts, How to update fonts cache files, How to confirm new fonts installation"] .

In next paragraphs we will explain in detail how to create simple console application. All of them are based on this VS Code guide:

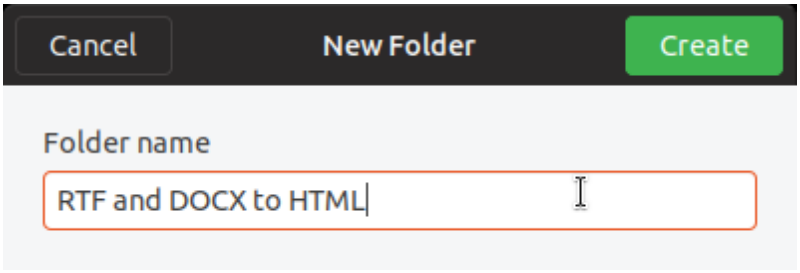
[Get Started with C# and Visual Studio Code]

Not only is possible to create .NET applications that will run on Linux using Linux as a developing platform. It is also possible to create it using a Windows machine and any modern Visual Studio version, as Microsoft Visual Studio Community 2022.

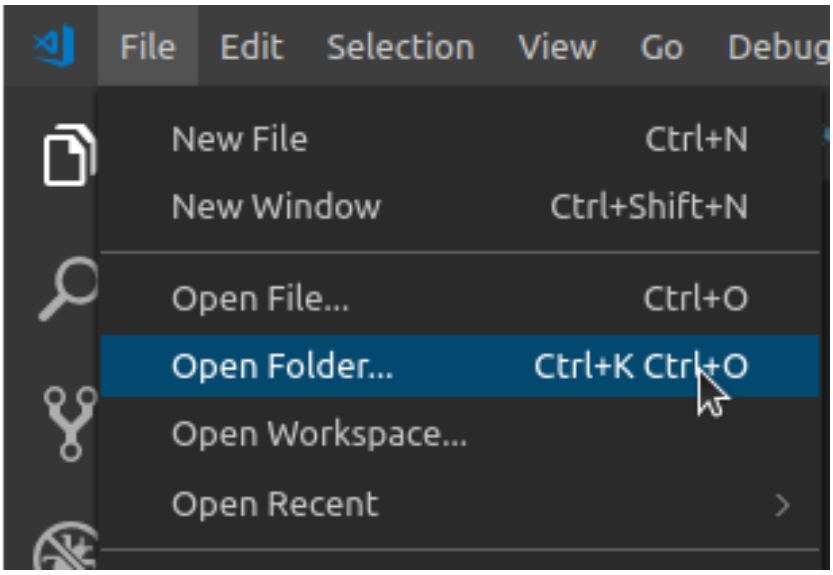
2. Creating “Convert RTF/DOCX to HTML” app

Create a new folder in your Linux machine with the name ***RTF and DOCX to HTML.***

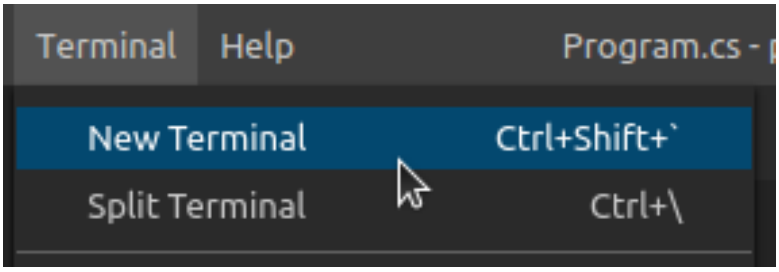
For example, let’s create the folder “***RTF and DOCX to HTML***” on Desktop (Right click-> New Folder):



Open VS Code and click in the menu ***File->Open Folder***. From the dialog, open the folder you’ve created previously:

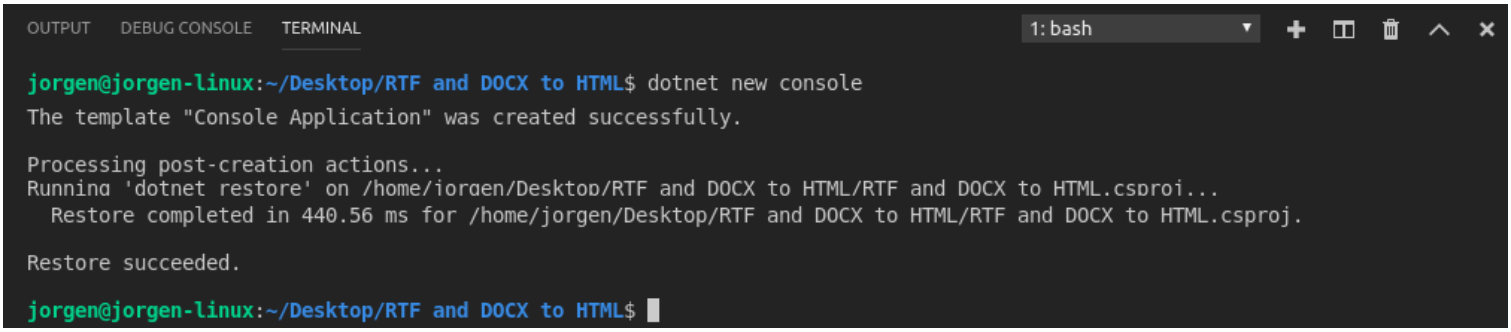


Now, open the integrated console – the Terminal: follow to the menu ***Terminal -> New Terminal*** (or press Ctrl+Shift+’):



Create a new console application, using ***dotnet*** command.

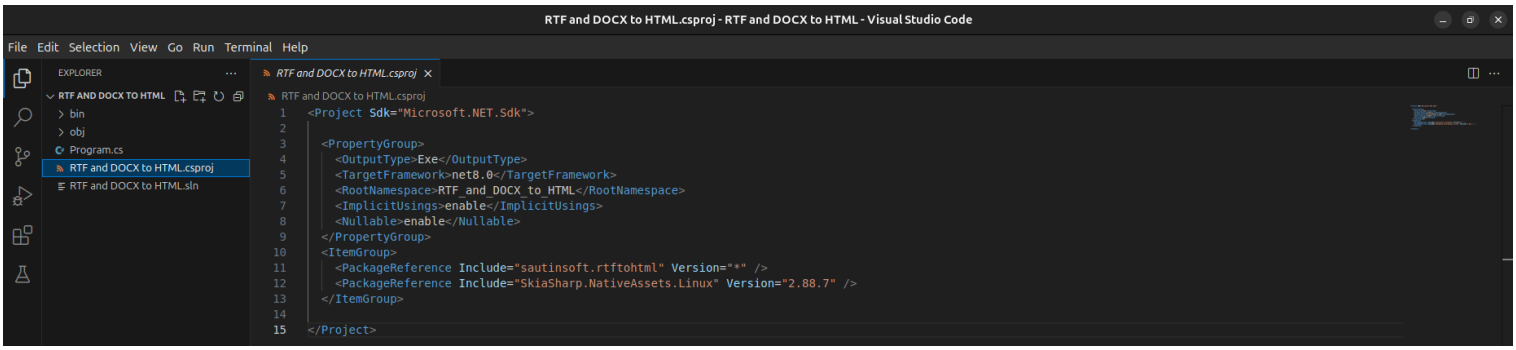
Type this command in the Terminal console: ***dotnet new console***



Now we are going to modify this simple application into an application that will convert rtf and docx files into HTML format.

First of all, we need to add the package reference to the ***sautinsoft.rtftohtml*** assembly using Nuget.

In order to do it, follow to the ***Explorer*** and open project file "***RTF and DOCX to HTML.csproj***" :



Add these lines into the file "***RTF and DOCX to HTML.csproj***":

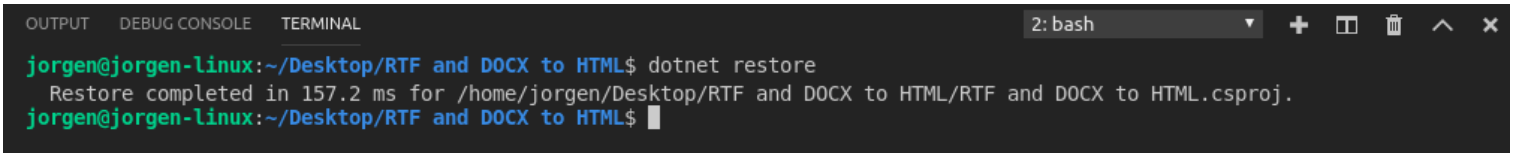
```
<ItemGroup>

      <PackageReference Include="sautinsoft.rtftohtml" Version="*" />
      <PackageReference Include="SkiaSharp.NativeAssets.Linux" Version="2.88.7" />

</ItemGroup>
```

The first reference installs the latest version ***sautinsoft.rtftohtml*** package from Nuget.

The second reference installs the ***SkiaSharp.NativeAssets.Linux*** package, which adds 2D graphics to .Net applications for Linux.

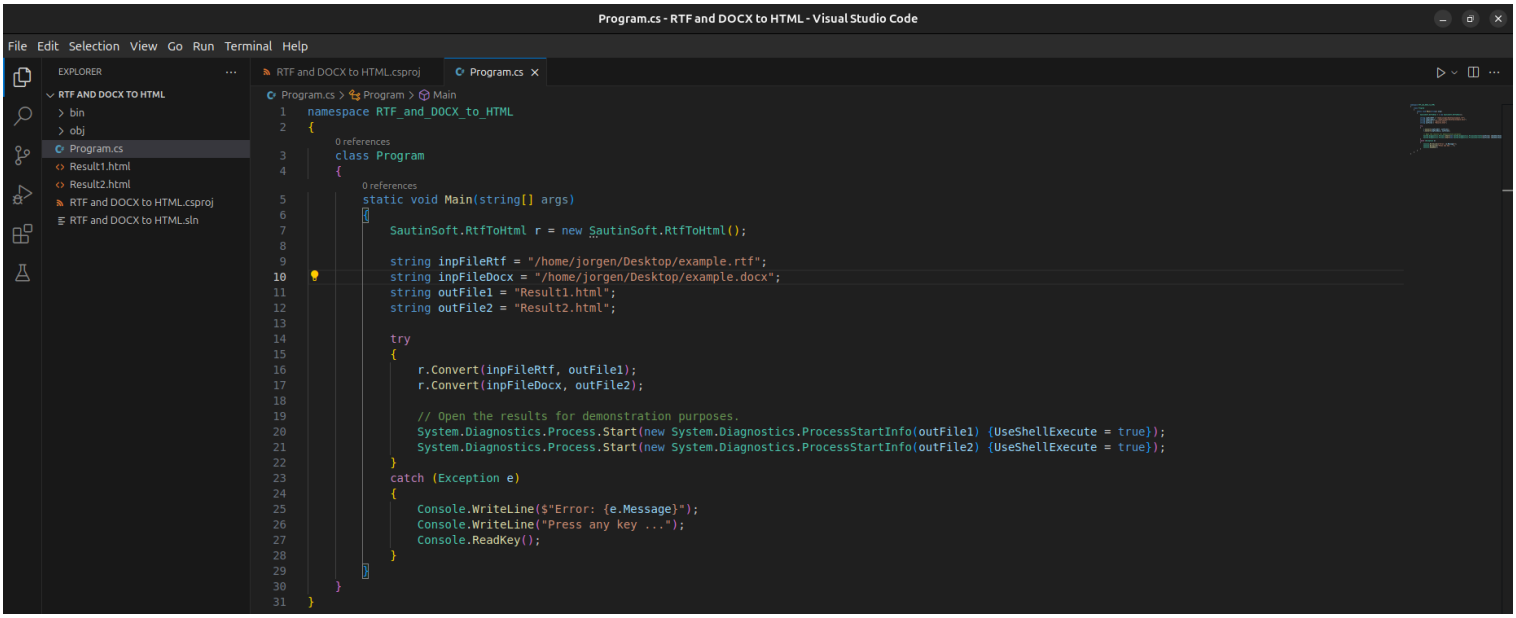
At once as we've added the package references, we have to save the "***RTF and DOCX to HTML.csproj***" and restore the added packages.

Follow to the ***Terminal*** and type the command: ***dotnet restore***



Good, now our application has all the references and we can write the code to convert DOCX and RTF documents into HTML format.

Follow to the ***Explorer***, open the ***Program.cs***, remove all the code and type the new:



The code:

```
namespace RTF_and_DOCX_to_HTML
{
    class Program
    {
        static void Main(string[] args)
        {
            SautinSoft.RtfToHtml r = new SautinSoft.RtfToHtml();

            string inpFileRtf = "/home/jorgen/Desktop/example.rtf";
            string inpFileDocx = "/home/jorgen/Desktop/example.docx";
            string outFile1 = "Result1.html";
            string outFile2 = "Result2.html";

            try
            {
                r.Convert(inpFileRtf, outFile1);
                r.Convert(inpFileDocx, outFile2);
```

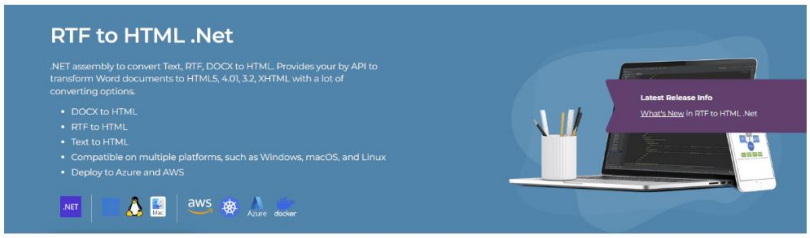
```
                // Open the results for demonstration purposes.
                System.Diagnostics.Process.Start(new
System.Diagnostics.ProcessStartInfo(outFile1) {UseShellExecute = true});
                System.Diagnostics.Process.Start(new
System.Diagnostics.ProcessStartInfo(outFile2) {UseShellExecute = true});
            }
            catch (Exception e)
            {
                Console.WriteLine($"Error: {e.Message}");
                Console.WriteLine("Press any key ...");
                Console.ReadKey();
            }
        }
    }
}
```

To make tests, we need the input RTF and DOCX documents. For our tests, let's place the files "example.rtf" and "example.docx" at the Desktop.

Launch our application to create a new DOCX document, type the command: ***dotnet run***

If you see the opening browser with the output HTML documents, everything is fine and we can check the results produced by the RTF to HTML .Net library.

Welcome to SautinSoft!
This is «HTML to RTF .Net» sample



Subtitle

This rich text document content serves as basis for trying out various rich text formatting.

Header 1

Bold *italic* underlined ~~strikethrough~~ $N_{subscript}$ $N^{superscript}$

Sub header 1.1

Fonts:

Times new roman

Arial

Well done! You have created the “RTF/DOCX to HTML” application under Linux!

If you have any troubles or need extra code, or help, don’t hesitate to ask our SautinSoft Team at support@sautinsoft.com!