

```
mcs -out:HelloWorld.exe HelloWorld.cs
```

The resulting `HelloWorld.exe` can then be executed with:

```
mono HelloWorld.exe
```

which will produce the output:

```
Hello, world!  
Press any key to exit..
```

Creating a new program using .NET Core

First install the [.NET Core SDK](#) by going through the installation instructions for the platform of your choice:

- [Windows](#)
- [OSX](#)
- [Linux](#)
- [Docker](#)

After the installation has completed, open a command prompt, or terminal window.

1. Create a new directory with `mkdir hello_world` and change into the newly created directory with `cd hello_world`.
2. Create a new console application with `dotnet new console`.
This will produce two files:

- **hello_world.csproj**

```
<Project Sdk="Microsoft.NET.Sdk">  
  
  <PropertyGroup>  
    <OutputType>Exe</OutputType>  
    <TargetFramework>netcoreapp1.1</TargetFramework>  
  </PropertyGroup>  
  
</Project>
```

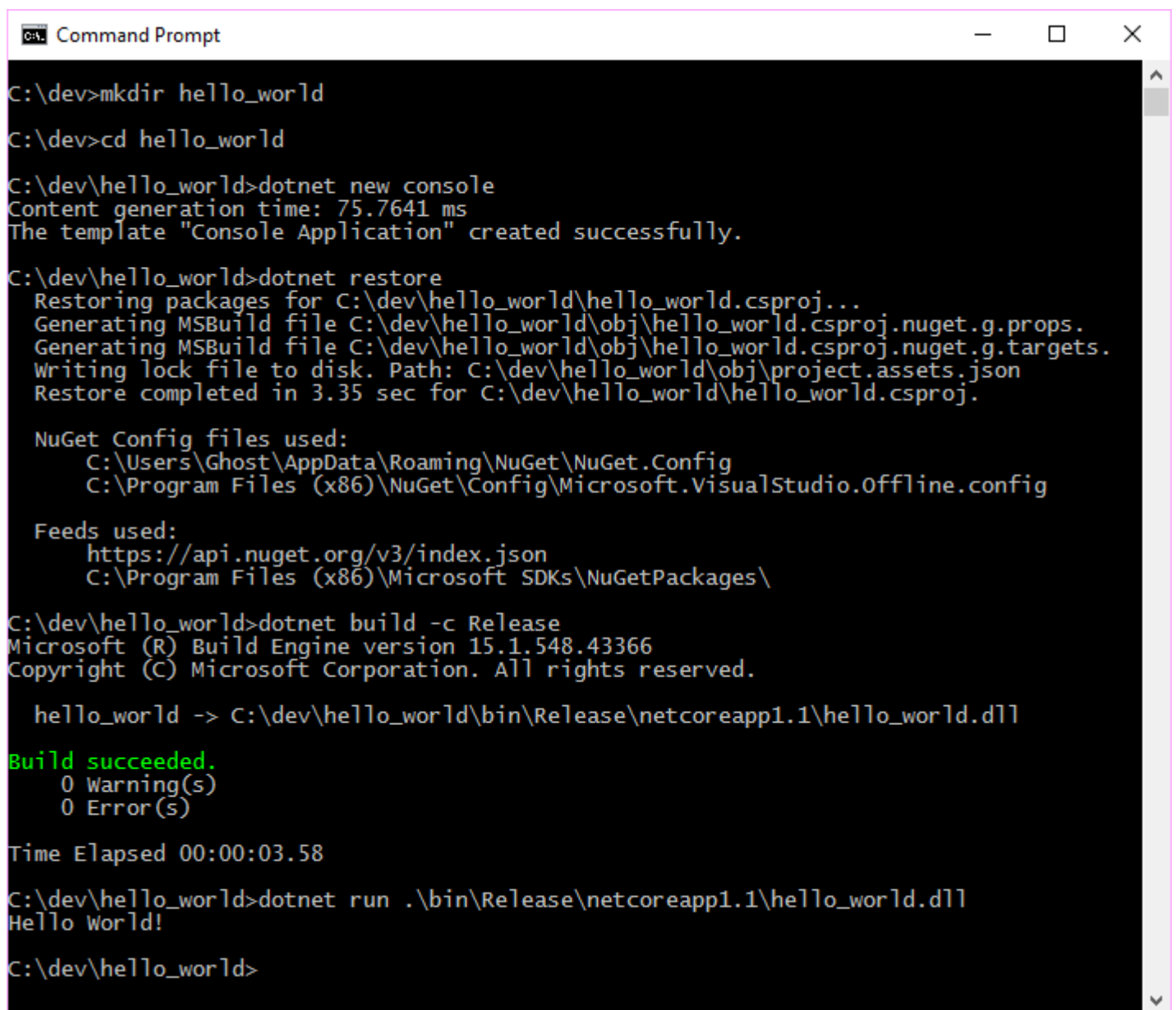
- **Program.cs**

```
using System;  
  
namespace hello_world  
{  
    class Program  
    {  
        static void Main(string[] args)  
        {  
            Console.WriteLine("Hello World!");  
        }  
    }  
}
```

```
}  
}  
}
```

3. Restore the needed packages with `dotnet restore`.
4. *Optional* Build the application with `dotnet build` for Debug or `dotnet build -c Release` for Release. `dotnet run` will also run the compiler and throw build errors, if any are found.
5. Run the application with `dotnet run` for Debug or `dotnet run .\bin\Release\netcoreapp1.1\hello_world.dll` for Release.

Command Prompt output

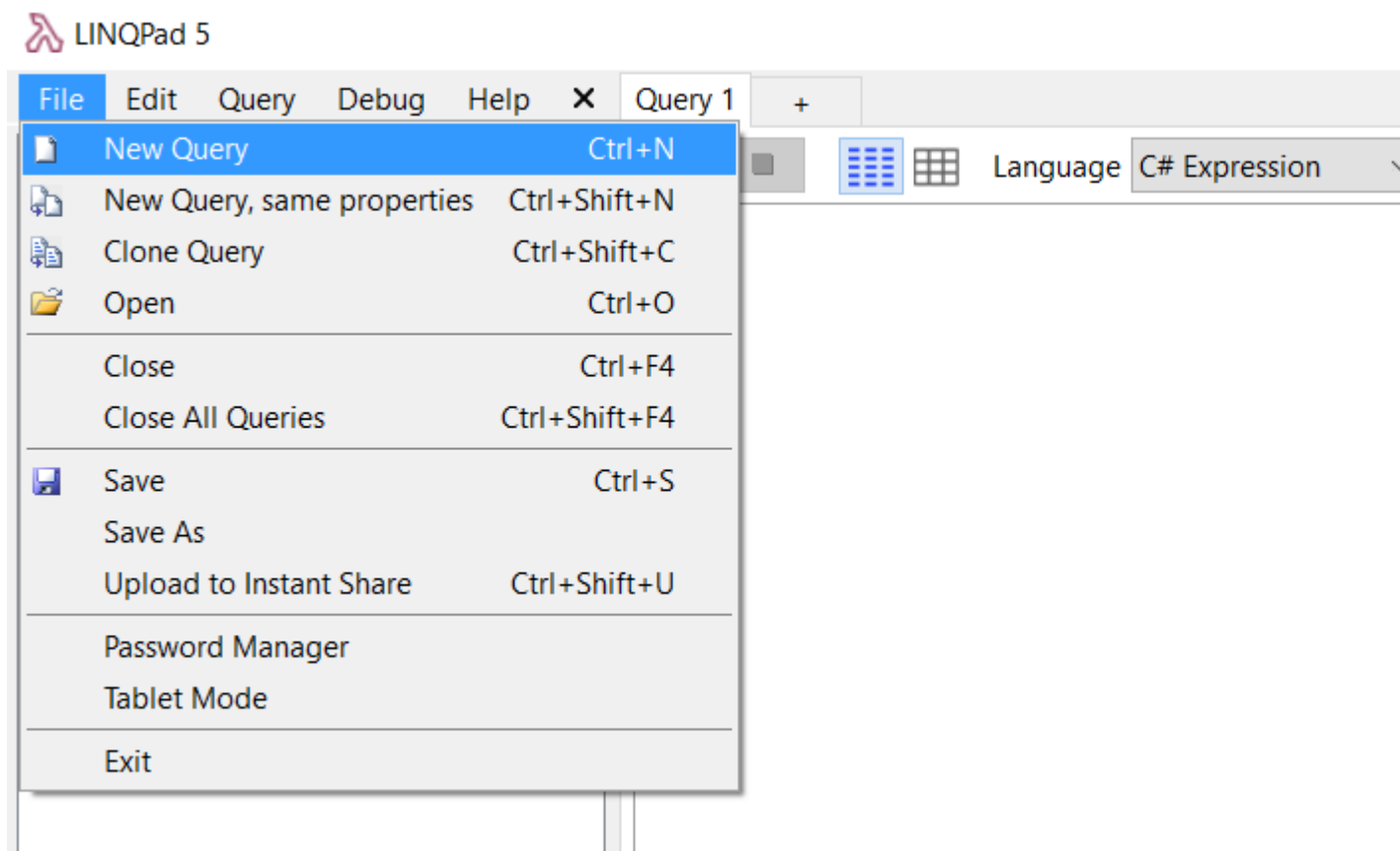


```
C:\dev>mkdir hello_world  
C:\dev>cd hello_world  
C:\dev\hello_world>dotnet new console  
Content generation time: 75.7641 ms  
The template "Console Application" created successfully.  
C:\dev\hello_world>dotnet restore  
Restoring packages for C:\dev\hello_world\hello_world.csproj...  
Generating MSBuild file C:\dev\hello_world\obj\hello_world.csproj.nuget.g.props.  
Generating MSBuild file C:\dev\hello_world\obj\hello_world.csproj.nuget.g.targets.  
Writing lock file to disk. Path: C:\dev\hello_world\obj\project.assets.json  
Restore completed in 3.35 sec for C:\dev\hello_world\hello_world.csproj.  
  
NuGet Config files used:  
  C:\Users\Ghost\AppData\Roaming\NuGet\NuGet.Config  
  C:\Program Files (x86)\NuGet\Config\Microsoft.VisualStudio.Offline.config  
  
Feeds used:  
  https://api.nuget.org/v3/index.json  
  C:\Program Files (x86)\Microsoft SDKs\NuGetPackages\  
  
C:\dev\hello_world>dotnet build -c Release  
Microsoft (R) Build Engine version 15.1.548.43366  
Copyright (C) Microsoft Corporation. All rights reserved.  
  
hello_world -> C:\dev\hello_world\bin\Release\netcoreapp1.1\hello_world.dll  
Build succeeded.  
    0 Warning(s)  
    0 Error(s)  
  
Time Elapsed 00:00:03.58  
C:\dev\hello_world>dotnet run .\bin\Release\netcoreapp1.1\hello_world.dll  
Hello world!  
C:\dev\hello_world>
```

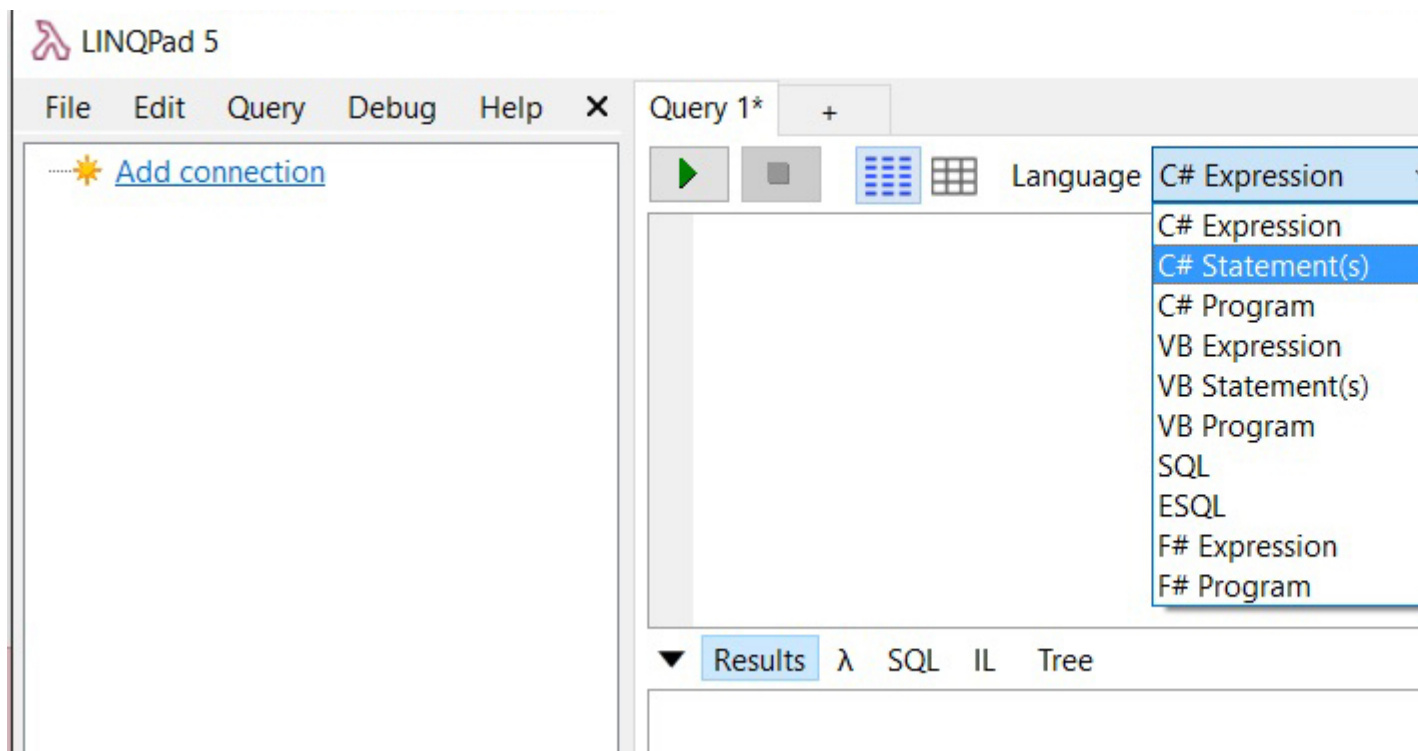
Creating a new query using LinqPad

LinqPad is a great tool that allows you to learn and test features of .Net languages (C#, F# and VB.Net.)

1. Install [LinqPad](#)
2. Create a new Query (**Ctrl + N**)

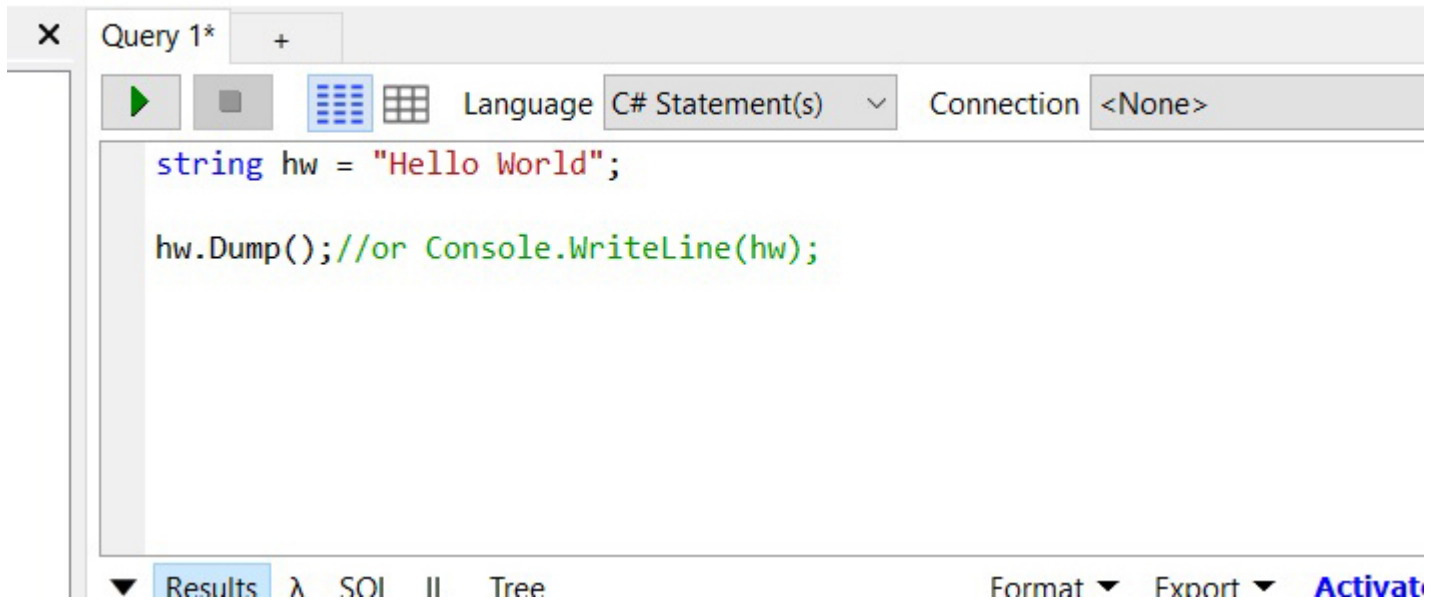


3. Under language, select "C# statements"

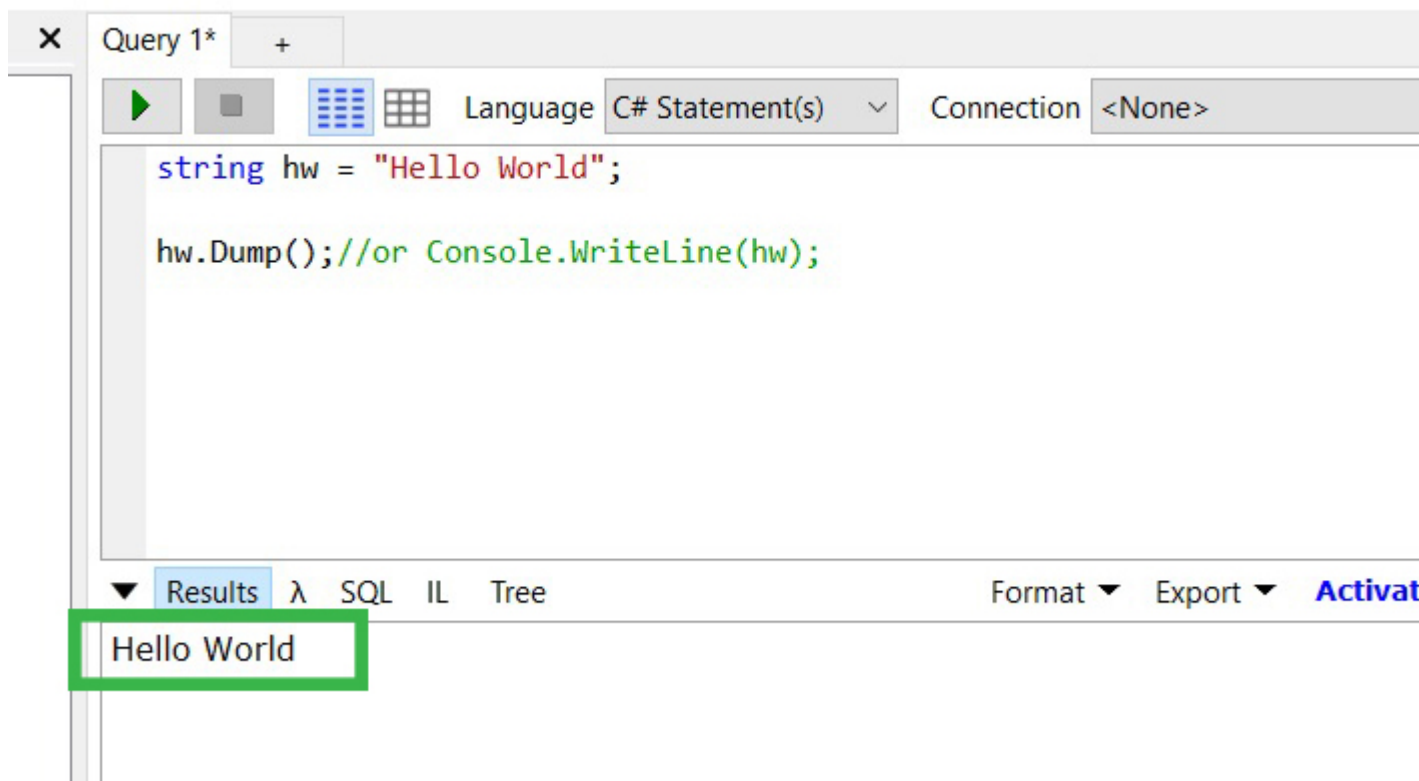


4. Type the following code and hit run (F5)

```
string hw = "Hello World";  
  
hw.Dump(); //or Console.WriteLine(hw);
```



5. You should see "Hello World" printed out in the results screen.



6. Now that you have created your first .Net program, go and check out the samples included in LinqPad via the "Samples" browser. There are many great examples that will show you many different features of the .Net languages.

The screenshot shows the LINQPad 5 application window. The title bar reads "LINQPad 5". The menu bar includes "File", "Edit", "Query", "Debug", and "Help". The main editor area on the right contains a C# query labeled "Query 1*" with the following code:

```
string hw = "Hello World";  
hw.Dump(); //or Console.WriteLine(hw);
```

Below the code editor, there are tabs for "Results", "λ", "SQL", "IL", and "Tree". The "Results" tab is selected, displaying the output "Hello World". At the bottom of the window, a status bar indicates "Query successful (00:00.000)".

On the left side of the window, there is a sidebar with a tabbed interface. The "My Queries" tab is active, showing a list of saved queries:

- [-] LINQPad 5 minute induction
- [-] C# 6.0 in a Nutshell
- [-] F# Tutorial

Below the list is a link "Download/import more samples". The sidebar area is highlighted with a green rectangle.

Notes:

1. If you click on "IL", you can inspect the IL code that your .net code generates. This is a great learning tool.