RYSKAMP LEARNING MACHINE

Developer’s Guide

Chapter 1: Download and Installation

1.1. Checking Requirements

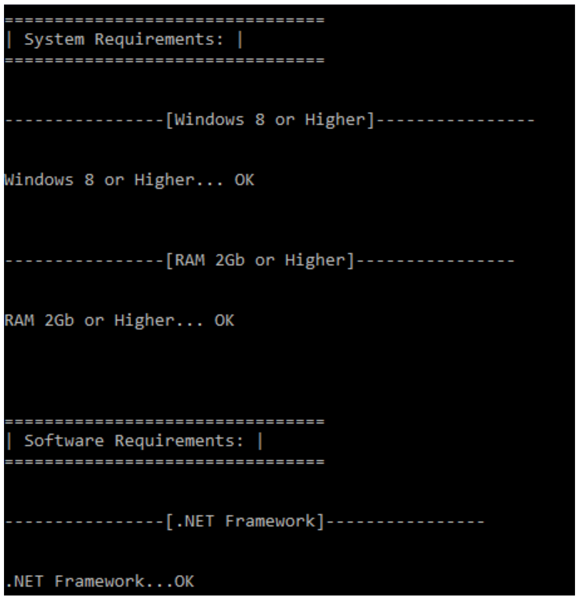
We have made an easy way for you to check the compatibility of your machine, which is located at the downloaded directory:

.\\ ExampleApps\UtilityTools\RequirementsTool\bin\debug\RequirementsTool.exe

It should check if you want to use the python code or not and immediate check if there are any compatibility issues and advise what to do.

For Windows

a. Visual Studio (Optional)
b. SQL Server 2016 and up
c. At Least .Net Framework 4.6.1
d. If are using Python Code (OPTIONAL)
    - Python 3.5
    - Python Tools 2.2.6 for Visual Studio 2015
    - Python .NET (explained in the guide)
e. Recommended Machine Requirements
    - CPU: Quad Core
    - RAM: 8 GB
    - OS: Windows 8 or 10
f. Minimum Machine Requirements
    - CPU: Single Core
    - RAM: 2 GB
    - OS: Windows 8 or 10



```
================================
| System Requirements: |
================================


---------------[Windows 8 or Higher]---------------


Windows 8 or Higher... OK



---------------[RAM 2Gb or Higher]---------------


RAM 2Gb or Higher... OK




================================
| Software Requirements: |
================================


---------------[.NET Framework]---------------


.NET Framework...OK
```

For Linux

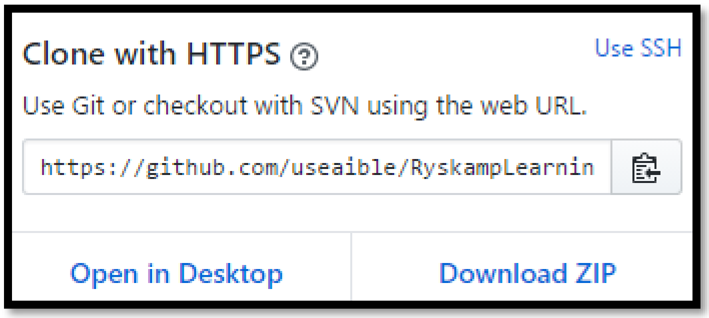
a. Ubuntu 16.04
b. At least .Net Core 2.0
c. IDE (Visual Studio Code Recommended)
d. Database Provider (SQL Server Recommended)
e. Internet Connection

1.2. Downloading the Source Code

To get the source code, please go to our **Github Repository** at https://github.com/useaible/ryskamplearningmachine

Once you see the Repository, click Clone or Download.

Then click **Download Zip**.


Clone with HTTPS

Use SSH

Use Git or checkout with SVN using the web URL.

```
https://github.com/useaible/RyskampLearnin
```

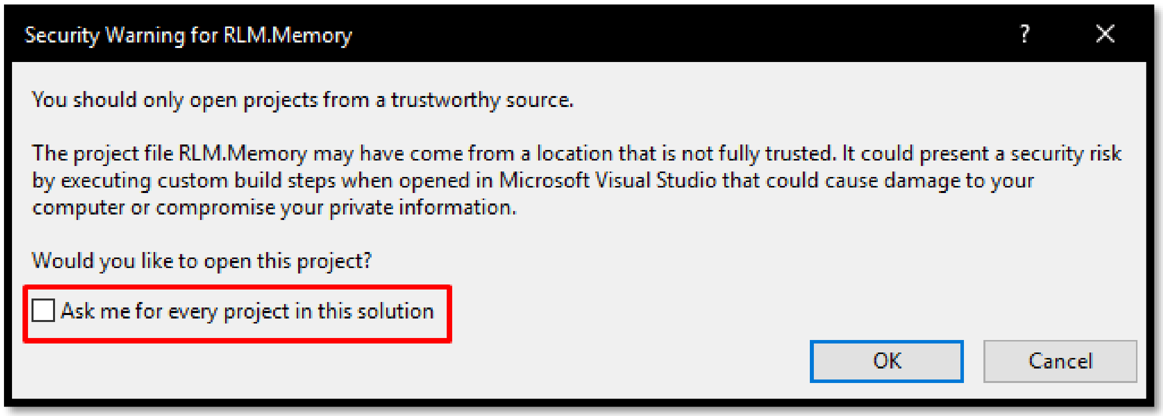
1.3. First Opening the Solution

Once the download is finished, Extract the zip file and you should now have a new folder named **RyskampLearningMachine-master**

RyskampLearningMachine-master

This should contain some files, one of which is **RLM.sln**, you will need to open it via Visual Studio.

You may run into this error below. You may just simply uncheck the **Ask me for every project in this solution** then click on OK to proceed.


Security Warning for RLM.Memory

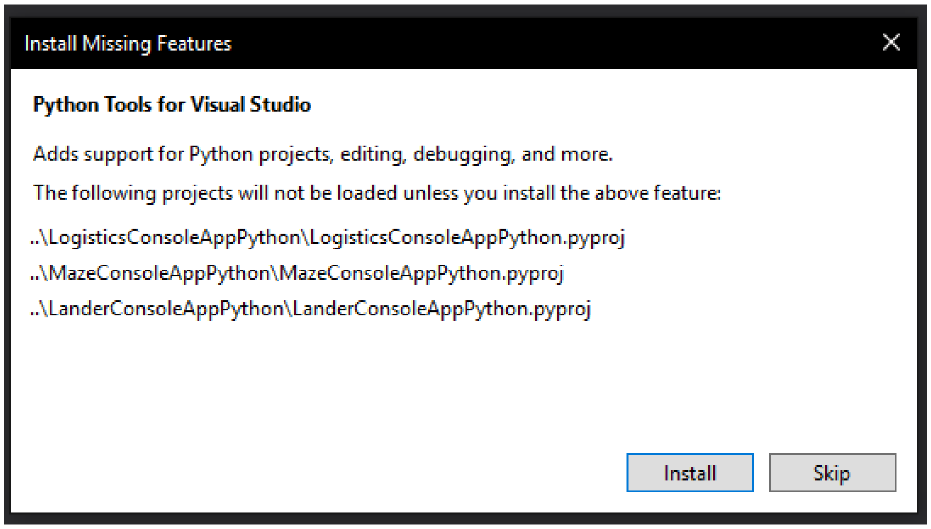
You should only open projects from a trustworthy source.

The project file RLM.Memory may have come from a location that is not fully trusted. It could present a security risk by executing custom build steps when opened in Microsoft Visual Studio that could cause damage to your computer or compromise your private information.

Would you like to open this project?

☐ Ask me for every project in this solution

Also, since the Solution includes Python code, in the event that you have not installed Python Tools 2.2.6 for Visual Studio 2015, you will be prompted to install Python Tools for Visual Studio. **THIS IS OPTIONAL BUT IF YOU WANT TO USE THE PYTHON GAMES WE RECOMMEND INSTALLING IT EXTERNALLY THROUGH DOWNLOADABLE INSTALLER** HERE.


Install Missing Features

Python Tools for Visual Studio

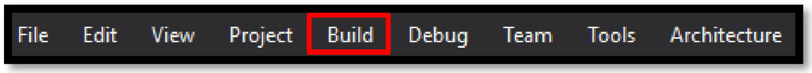
Adds support for Python projects, editing, debugging, and more.

The following projects will not be loaded unless you install the above feature:

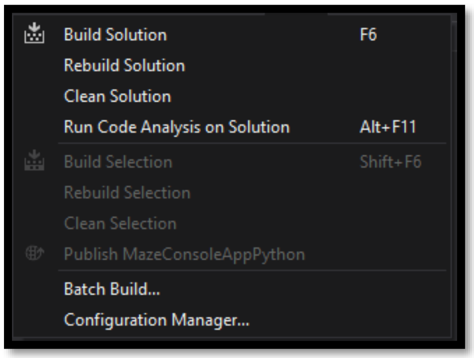
..\LogisticsConsoleAppPython\LogisticsConsoleAppPython.pyproj
..\MazeConsoleAppPython\MazeConsoleAppPython.pyproj
..\LanderConsoleAppPython\LanderConsoleAppPython.pyproj

1.4. Building the Solution

You will now want to build the code. This will download any missing packages and make the code run for the included applications. To build the code in Visual Studio, click on Build.



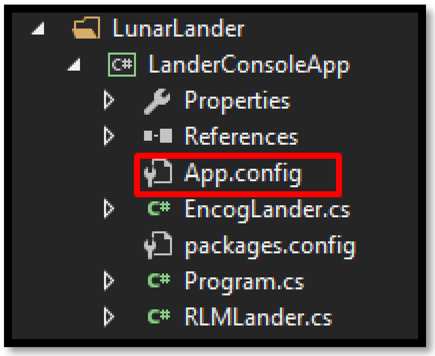
Then click on Build Solution or you could simply press F6. This might take a few seconds to a minute.



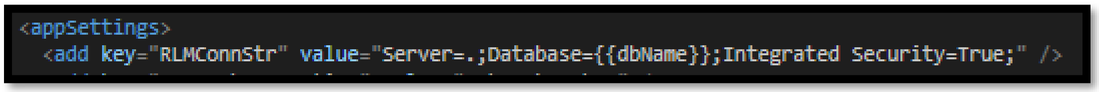
1.4.1. Connection Strings

Once the solution has been rebuilt, you may want to verify your connection strings. By default, it uses the SQL Server installed locally on your computer. You can configure this via the App.config for each of the Sample Apps.

Note that this has to be done for each of the app that you want to run.

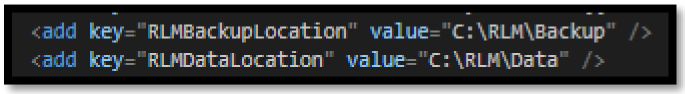


You will need to configure the line below to your needs.



```
<appSettings>
  <add key="RLMConnStr" value="Server=.;Database={{dbName}};Integrated Security=True;" />
```

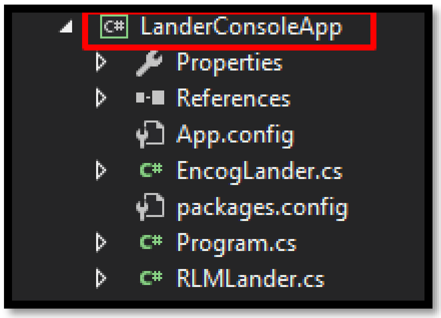
Another item for the App Settings are the Backup and Data location. By default, folders shown below are created. You may configure this as you see fit.

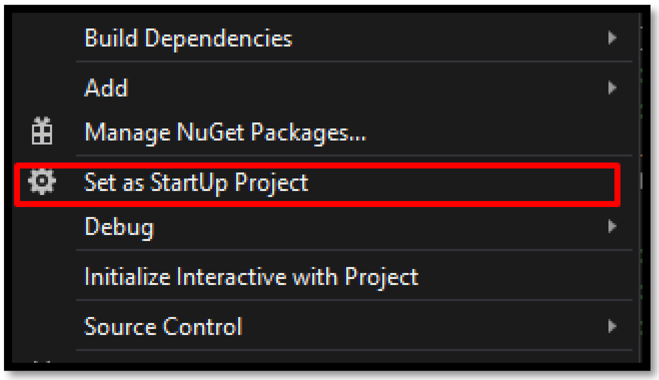


```
<add key="RLMBackupLocation" value="C:\RLM\Backup" />
<add key="RLMDataLocation" value="C:\RLM\Data" />
```

1.5. First Running a Sample App

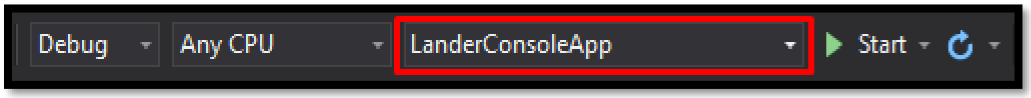
We should now be able to run the code. You can choose what your Startup Project is and Run that. On your Solution explorer, we can setup the LanderConsoleApp as the Startup Project.





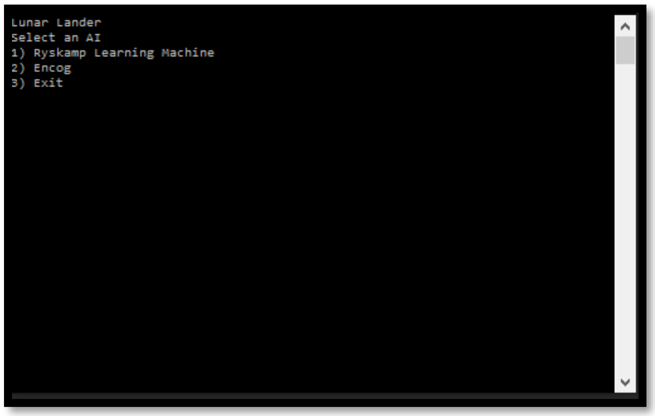
Right click on it and Select Set as Startup Project.

It should then show up as the Startup project located at the top of the page



Just click on Start and it should run.

Here we run the Lander Console App (Lunar Lander). For any errors, please refer to common errors section.



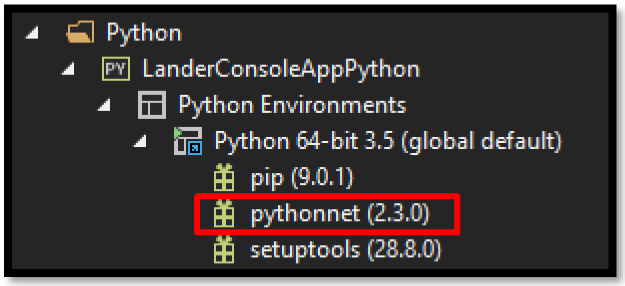
```
Lunar Lander
Select an AI
1) Ryskamp Learning Machine
2) Encog
3) Exit
```

1.6. Running the Python Sample Apps

1.6.1. Installing Python.Net

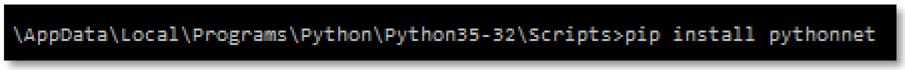
If you don't have it already, you might be missing python net.

But with the Python Environment already installed, it should be easy. Just get the Scripts folder directory of your Python Environment and run from Command Prompt with Administrative Privileges.



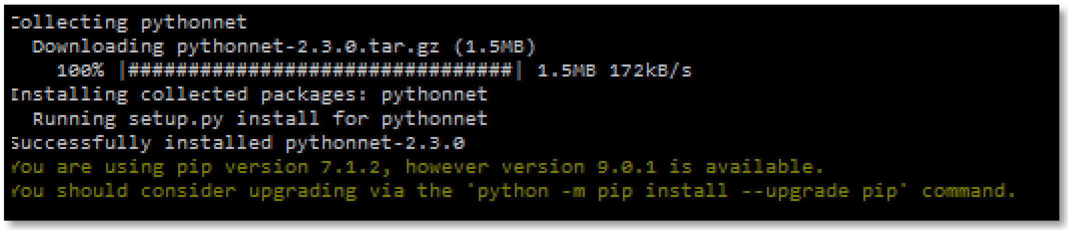
```
Python
  LanderConsoleAppPython
    Python Environments
      Python 64-bit 3.5 (global default)
        pip (9.0.1)
        pythonnet (2.3.0)
        setuptools (28.8.0)
```

The command is pip install python.net.



```
\AppData\Local\Programs\Python\Python35-32\Scripts>pip install pythonnet
```

It should go through the install process.



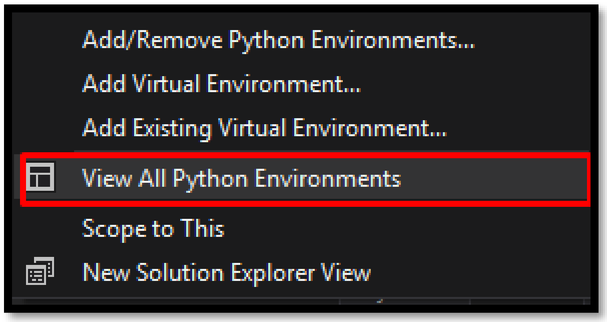
```
Collecting pythonnet
  Downloading pythonnet-2.3.0.tar.gz (1.5MB)
    100% |################################| 1.5MB 172kB/s
Installing collected packages: pythonnet
  Running setup.py install for pythonnet
Successfully installed pythonnet-2.3.0
You are using pip version 7.1.2, however version 9.0.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

For errors, check section 1.7. Common Errors.

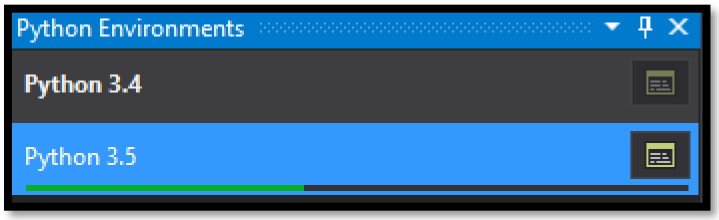
1.6.2. Python Environments

If there are no available options under Python Environments and it's already installed, you might need to refresh it.

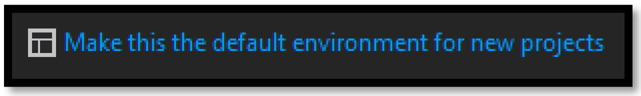
Right click the Python Environments under your selected Sample Python App. It should show up an option to view all python environments.


Add/Remove Python Environments...
Add Virtual Environment...
Add Existing Virtual Environment...
View All Python Environments
Scope to This
New Solution Explorer View

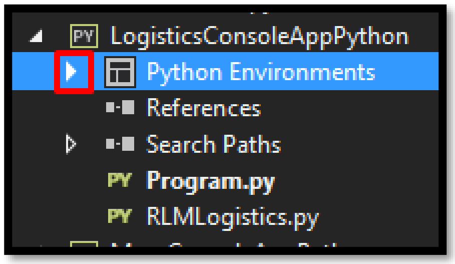
At the moment, we highly recommend Python 3.5. Just refresh it and it should load. If it's the only Python Environment you have, it should automatically populate.



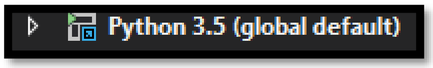
Once that is done loading, make it your default by clicking the link below it as shown below



To confirm that everything has been done correctly, go back to the solution explorer, and you should now see a drilldown for the Python Environments



Once clicked, it should show your selected environment.

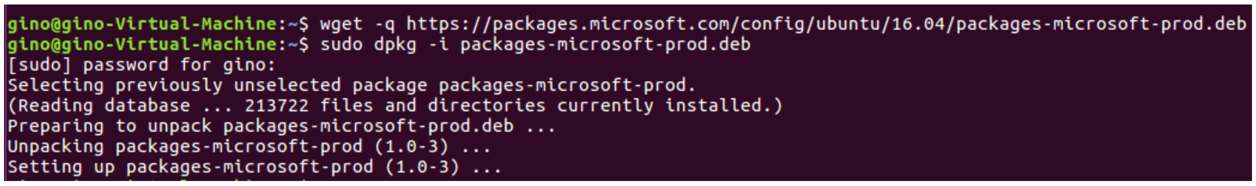


1.7. Using Sample Apps on Ubuntu

1.7.1. Installing .Net Core SDK on Linux Ubuntu 16.04

Before installing .Net, you'll need to register the Microsoft Key, register the product repository, and install required dependencies. This only needs to be done once per machine. Open the terminal and run the following commands.

```
wget -q https://packages.microsoft.com/config/ubuntu/16.04/packages-microsoft-prod.deb

sudo dpkg -i packages-microsoft-prod.deb
```

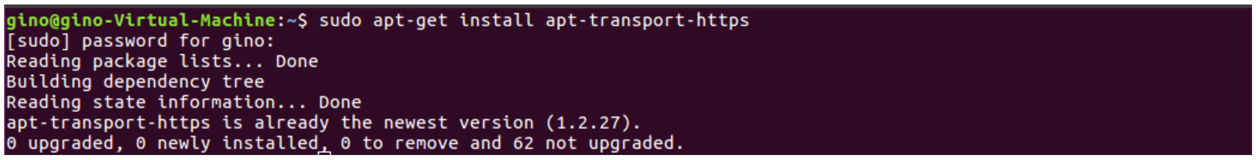


```
gino@gino-Virtual-Machine:~$ wget -q https://packages.microsoft.com/config/ubuntu/16.04/packages-microsoft-prod.deb
gino@gino-Virtual-Machine:~$ sudo dpkg -i packages-microsoft-prod.deb
[sudo] password for gino:
Selecting previously unselected package packages-microsoft-prod.
(Reading database ... 213722 files and directories currently installed.)
Preparing to unpack packages-microsoft-prod.deb ...
Unpacking packages-microsoft-prod (1.0-3) ...
Setting up packages-microsoft-prod (1.0-3) ...
```

Update the products available for installation, then install the .Net SDK.

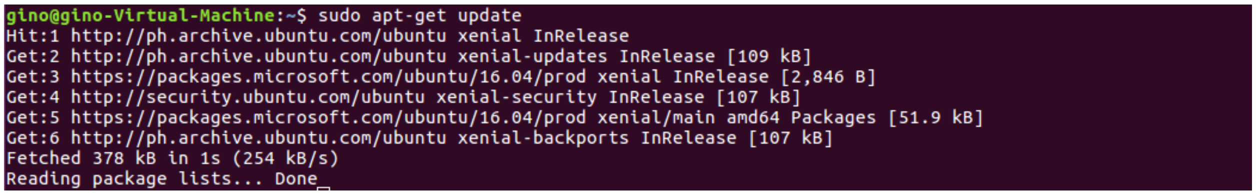
In the terminal, run the following commands.

```
sudo apt-get install apt-transport-https
```



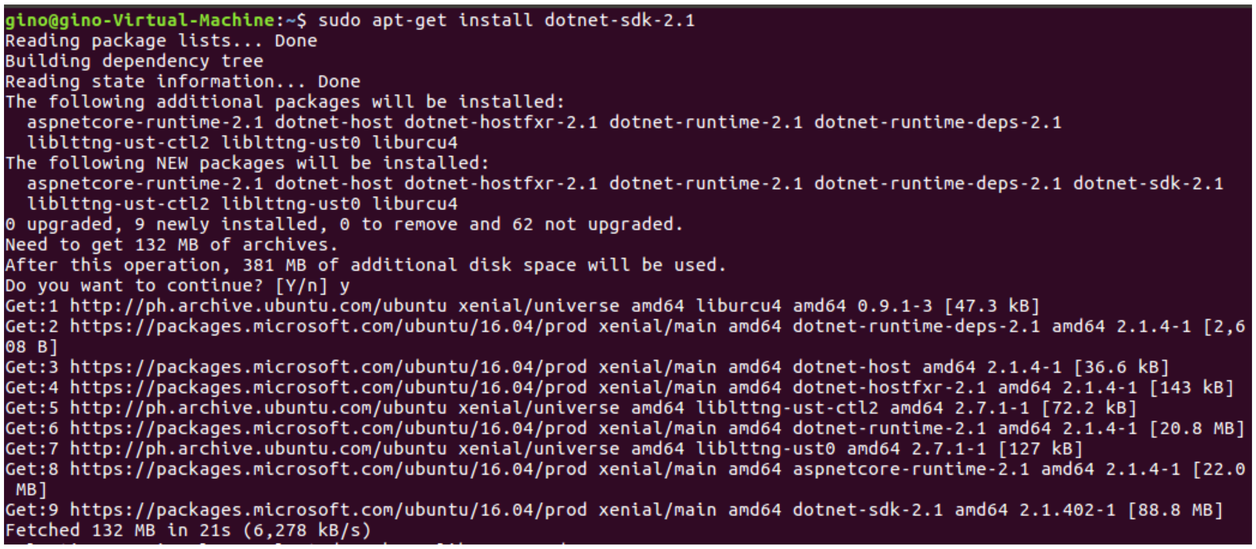
```
gino@gino-Virtual-Machine:~$ sudo apt-get install apt-transport-https
[sudo] password for gino:
Reading package lists... Done
Building dependency tree
Reading state information... Done
apt-transport-https is already the newest version (1.2.27).
0 upgraded, 0 newly installed, 0 to remove and 62 not upgraded.
```

```
sudo apt-get update
```



```
gino@gino-Virtual-Machine:~$ sudo apt-get update
Hit:1 http://ph.archive.ubuntu.com/ubuntu xenial InRelease
Get:2 http://ph.archive.ubuntu.com/ubuntu xenial-updates InRelease [109 kB]
Get:3 https://packages.microsoft.com/ubuntu/16.04/prod xenial InRelease [2,846 B]
Get:4 http://security.ubuntu.com/ubuntu xenial-security InRelease [107 kB]
Get:5 https://packages.microsoft.com/ubuntu/16.04/prod xenial/main amd64 Packages [51.9 kB]
Get:6 http://ph.archive.ubuntu.com/ubuntu xenial-backports InRelease [107 kB]
Fetched 378 kB in 1s (254 kB/s)
Reading package lists... Done
```

```
sudo apt-get install dotnet-sdk-2.1
```

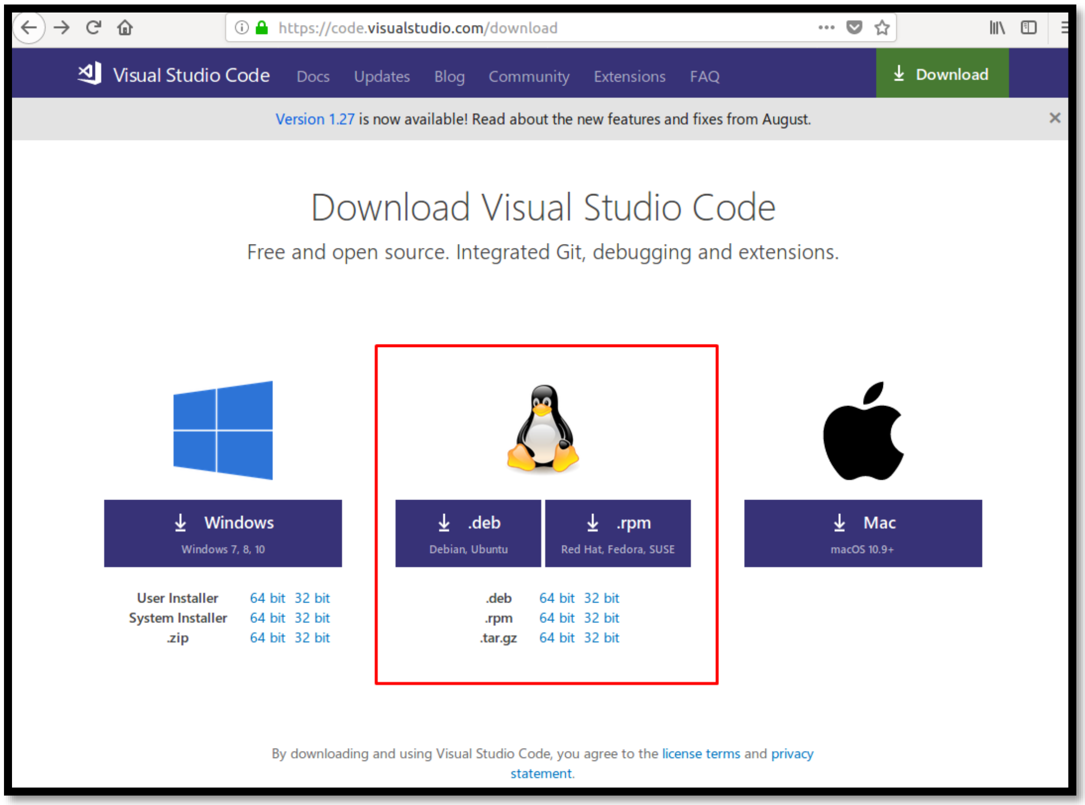


```
gino@gino-Virtual-Machine:~$ sudo apt-get install dotnet-sdk-2.1
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  aspnetcore-runtime-2.1 dotnet-host dotnet-hostfxr-2.1 dotnet-runtime-2.1 dotnet-runtime-deps-2.1
  liblttng-ust-ctl2 liblttng-ust0 liburcu4
The following NEW packages will be installed:
  aspnetcore-runtime-2.1 dotnet-host dotnet-hostfxr-2.1 dotnet-runtime-2.1 dotnet-runtime-deps-2.1 dotnet-sdk-2.1
  liblttng-ust-ctl2 liblttng-ust0 liburcu4
0 upgraded, 9 newly installed, 0 to remove and 62 not upgraded.
Need to get 132 MB of archives.
After this operation, 381 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu xenial/universe amd64 liburcu4 amd64 0.9.1-3 [47.3 kB]
Get:2 https://packages.microsoft.com/ubuntu/16.04/prod xenial/main amd64 dotnet-runtime-deps-2.1 amd64 2.1.4-1 [2,6
08 B]
Get:3 https://packages.microsoft.com/ubuntu/16.04/prod xenial/main amd64 dotnet-host amd64 2.1.4-1 [36.6 kB]
Get:4 https://packages.microsoft.com/ubuntu/16.04/prod xenial/main amd64 dotnet-hostfxr-2.1 amd64 2.1.4-1 [143 kB]
Get:5 http://ph.archive.ubuntu.com/ubuntu xenial/universe amd64 liblttng-ust-ctl2 amd64 2.7.1-1 [72.2 kB]
Get:6 https://packages.microsoft.com/ubuntu/16.04/prod xenial/main amd64 dotnet-runtime-2.1 amd64 2.1.4-1 [20.8 MB]
Get:7 http://ph.archive.ubuntu.com/ubuntu xenial/universe amd64 liblttng-ust0 amd64 2.7.1-1 [127 kB]
Get:8 https://packages.microsoft.com/ubuntu/16.04/prod xenial/main amd64 aspnetcore-runtime-2.1 amd64 2.1.4-1 [22.0
 MB]
Get:9 https://packages.microsoft.com/ubuntu/16.04/prod xenial/main amd64 dotnet-sdk-2.1 amd64 2.1.402-1 [88.8 MB]
Fetched 132 MB in 21s (6,278 kB/s)
```

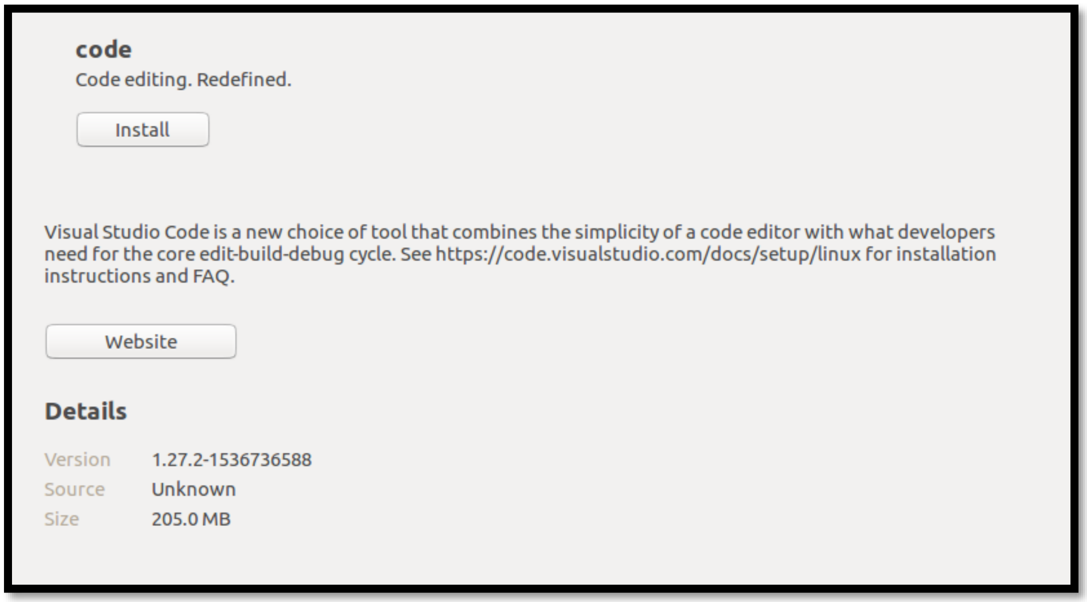
1.7.2. Using IDE (Visual Studio Code)

If you want to create a sample project in Ubuntu, you need to have an IDE. In this example, we recommend Visual Studio Code although you can choose other IDEs.

The best way to install Visual Studio Code for Ubuntu is to download it from their website. You'll need to go to the Visual Studio Code website code.visualstudio.com/download.


https://code.visualstudio.com/download
Visual Studio Code
Docs
Updates
Blog
Community
Extensions
FAQ
Download
Version 1.27 is now available! Read about the new features and fixes from August.
Download Visual Studio Code
Free and open source. Integrated Git, debugging and extensions.
Windows
Windows 7, 8, 10
.deb
Debian, Ubuntu
.rpm
Red Hat, Fedora, SUSE
Mac
macOS 10.9+
User Installer 64 bit 32 bit
System Installer 64 bit 32 bit
.zip 64 bit 32 bit
.deb 64 bit 32 bit
.rpm 64 bit 32 bit
.tar.gz 64 bit 32 bit
By downloading and using Visual Studio Code, you agree to the license terms and privacy statement.

Download the .deb file of your chosen system type. Once downloaded, run the file to install.

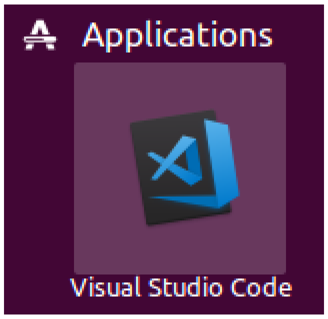


Visual Studio Code is a new choice of tool that combines the simplicity of a code editor with what developers need for the core edit-build-debug cycle. See https://code.visualstudio.com/docs/setup/linux for installation instructions and FAQ.

Details

Version	1.27.2-1536736588
Source	Unknown
Size	205.0 MB

Once installed, it should appear in your Applications.

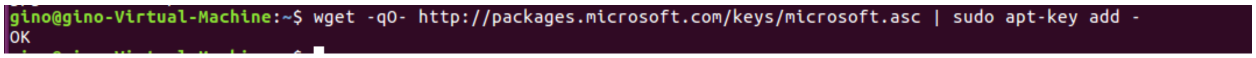


1.7.3. Installing DB Provider (SQL Server) - Optional

To install SQL Server on Ubuntu 16.04, we will need to run a few scripts to get the required files.

We will need to import the public repository GPG keys. Run the command below. Once successful, it should return an OK.

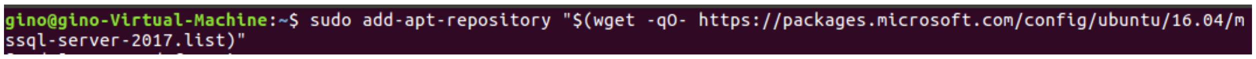
```
wget -qO- https://packages.microsoft.com/keys/microsoft.asc | sudo apt-key add -
```



```
gino@gino-Virtual-Machine:~$ wget -qO- http://packages.microsoft.com/keys/microsoft.asc | sudo apt-key add -
OK
```

Then, we will need to register the Microsoft SQL Server Ubuntu repository through the following code in the terminal

```
sudo add-apt-repository "$(wget -qO- https://packages.microsoft.com/config/ubuntu/16.04/mssql-server-2017.list)"
```

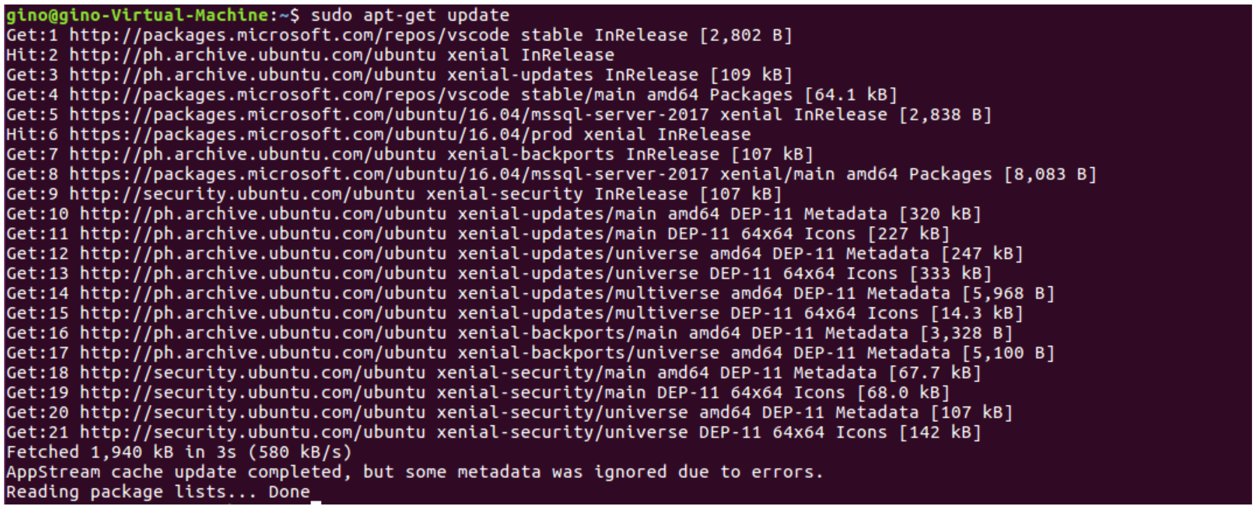


```
gino@gino-Virtual-Machine:~$ sudo add-apt-repository "$(wget -qO- https://packages.microsoft.com/config/ubuntu/16.04/m
ssql-server-2017.list)"
```

This should have no confirmation.

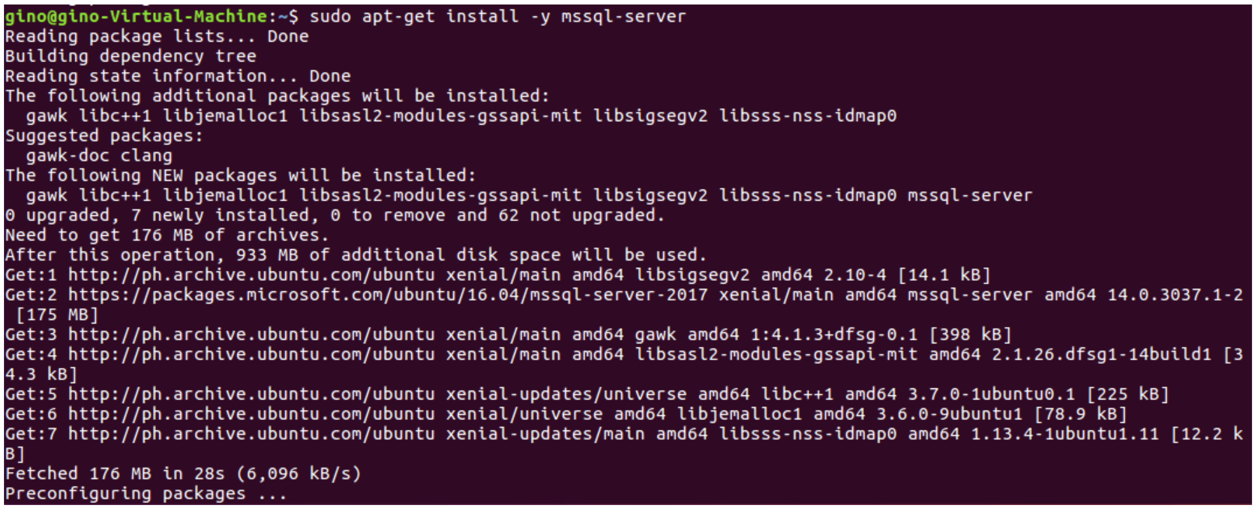
Next, we will be installing SQL Server by typing in this code:

```
sudo apt-get update
```



```
gino@gino-Virtual-Machine:~$ sudo apt-get update
Get:1 http://packages.microsoft.com/repos/vscode stable InRelease [2,802 B]
Hit:2 http://ph.archive.ubuntu.com/ubuntu xenial InRelease
Get:3 http://ph.archive.ubuntu.com/ubuntu xenial-updates InRelease [109 kB]
Get:4 http://packages.microsoft.com/repos/vscode stable/main amd64 Packages [64.1 kB]
Get:5 https://packages.microsoft.com/ubuntu/16.04/mssql-server-2017 xenial InRelease [2,838 B]
Hit:6 https://packages.microsoft.com/ubuntu/16.04/prod xenial InRelease
Get:7 http://ph.archive.ubuntu.com/ubuntu xenial-backports InRelease [107 kB]
Get:8 https://packages.microsoft.com/ubuntu/16.04/mssql-server-2017 xenial/main amd64 Packages [8,083 B]
Get:9 http://security.ubuntu.com/ubuntu xenial-security InRelease [107 kB]
Get:10 http://ph.archive.ubuntu.com/ubuntu xenial-updates/main amd64 DEP-11 Metadata [320 kB]
Get:11 http://ph.archive.ubuntu.com/ubuntu xenial-updates/main DEP-11 64x64 Icons [227 kB]
Get:12 http://ph.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 DEP-11 Metadata [247 kB]
Get:13 http://ph.archive.ubuntu.com/ubuntu xenial-updates/universe DEP-11 64x64 Icons [333 kB]
Get:14 http://ph.archive.ubuntu.com/ubuntu xenial-updates/multiverse amd64 DEP-11 Metadata [5,968 B]
Get:15 http://ph.archive.ubuntu.com/ubuntu xenial-updates/multiverse DEP-11 64x64 Icons [14.3 kB]
Get:16 http://ph.archive.ubuntu.com/ubuntu xenial-backports/main amd64 DEP-11 Metadata [3,328 B]
Get:17 http://ph.archive.ubuntu.com/ubuntu xenial-backports/universe amd64 DEP-11 Metadata [5,100 B]
Get:18 http://security.ubuntu.com/ubuntu xenial-security/main amd64 DEP-11 Metadata [67.7 kB]
Get:19 http://security.ubuntu.com/ubuntu xenial-security/main DEP-11 64x64 Icons [68.0 kB]
Get:20 http://security.ubuntu.com/ubuntu xenial-security/universe amd64 DEP-11 Metadata [107 kB]
Get:21 http://security.ubuntu.com/ubuntu xenial-security/universe DEP-11 64x64 Icons [142 kB]
Fetched 1,940 kB in 3s (580 kB/s)
AppStream cache update completed, but some metadata was ignored due to errors.
Reading package lists... Done
```

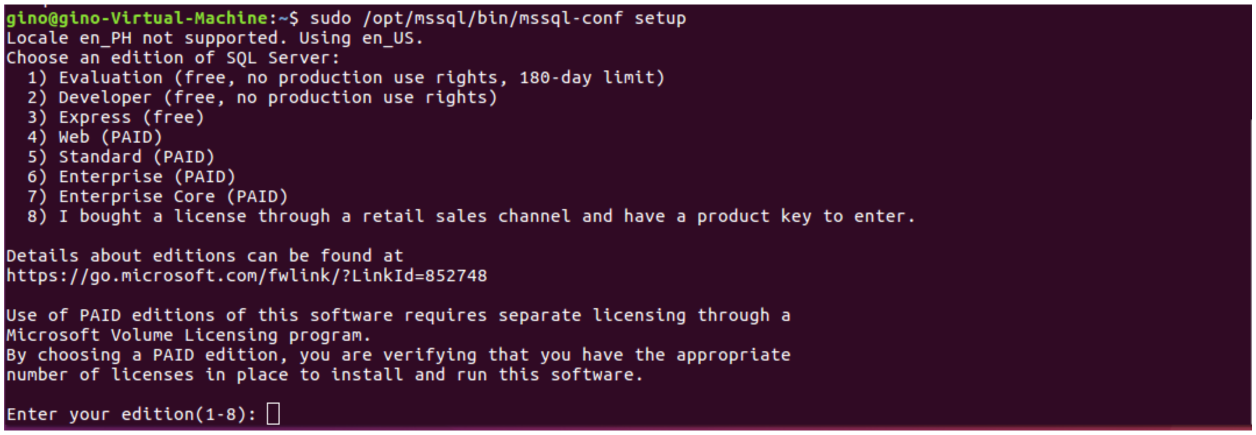
```
sudo apt-get install -y mssql-server
```



```
gino@gino-Virtual-Machine:~$ sudo apt-get install -y mssql-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  gawk libc++1 libjemalloc1 libsasl2-modules-gssapi-mit libsigsegv2 libsss-nss-idmap0
Suggested packages:
  gawk-doc clang
The following NEW packages will be installed:
  gawk libc++1 libjemalloc1 libsasl2-modules-gssapi-mit libsigsegv2 libsss-nss-idmap0 mssql-server
0 upgraded, 7 newly installed, 0 to remove and 62 not upgraded.
Need to get 176 MB of archives.
After this operation, 933 MB of additional disk space will be used.
Get:1 http://ph.archive.ubuntu.com/ubuntu xenial/main amd64 libsigsegv2 amd64 2.10-4 [14.1 kB]
Get:2 https://packages.microsoft.com/ubuntu/16.04/mssql-server-2017 xenial/main amd64 mssql-server amd64 14.0.3037.1-2
 [175 MB]
Get:3 http://ph.archive.ubuntu.com/ubuntu xenial/main amd64 gawk amd64 1:4.1.3+dfsg-0.1 [398 kB]
Get:4 http://ph.archive.ubuntu.com/ubuntu xenial/main amd64 libsasl2-modules-gssapi-mit amd64 2.1.26.dfsg1-14build1 [3
4.3 kB]
Get:5 http://ph.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 libc++1 amd64 3.7.0-1ubuntu0.1 [225 kB]
Get:6 http://ph.archive.ubuntu.com/ubuntu xenial/universe amd64 libjemalloc1 amd64 3.6.0-9ubuntu1 [78.9 kB]
Get:7 http://ph.archive.ubuntu.com/ubuntu xenial-updates/main amd64 libsss-nss-idmap0 amd64 1.13.4-1ubuntu1.11 [12.2 k
B]
Fetched 176 MB in 28s (6,096 kB/s)
Preconfiguring packages ...
```

After the installation finishes, run the following code

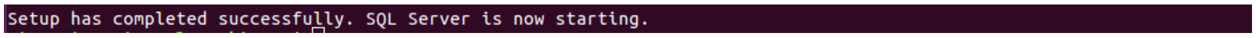
```
sudo /opt/mssql/bin/mssql-conf setup
```



```
gino@gino-Virtual-Machine:~$ sudo /opt/mssql/bin/mssql-conf setup
Locale en_PH not supported. Using en_US.
Choose an edition of SQL Server:
  1) Evaluation (free, no production use rights, 180-day limit)
  2) Developer (free, no production use rights)
  3) Express (free)
  4) Web (PAID)
  5) Standard (PAID)
  6) Enterprise (PAID)
  7) Enterprise Core (PAID)
  8) I bought a license through a retail sales channel and have a product key to enter.

Details about editions can be found at
https://go.microsoft.com/fwlink/?LinkId=852748

Use of PAID editions of this software requires separate licensing through a
Microsoft Volume Licensing program.
By choosing a PAID edition, you are verifying that you have the appropriate
number of licenses in place to install and run this software.

Enter your edition(1-8):
```

You'll be asked to choose your edition. We recommend using option number 3, SQL Server Express. Then just follow the prompts.

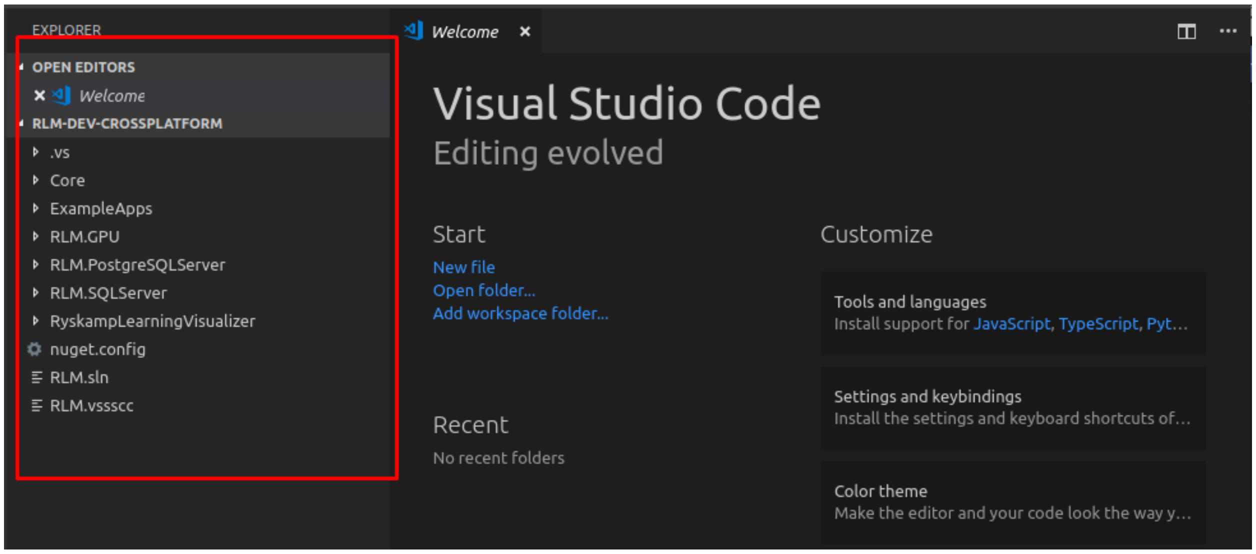
Once successful, it should prompt you.



```
Setup has completed successfully. SQL Server is now starting.
```

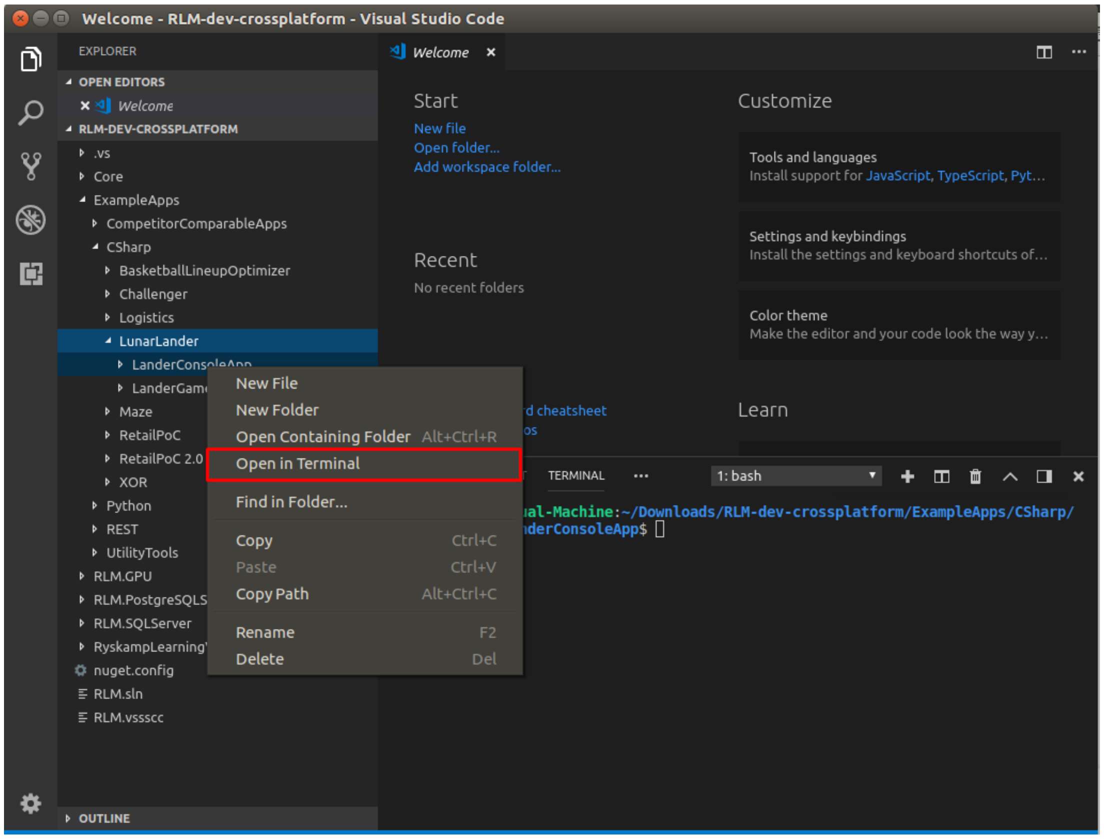
1.7.4. Opening the RLM Project in Ubuntu

Download the source code from Github and save it to folder. Using Visual Studio Code, open the folder where you save the source code. The files should appear in the left hand side of the IDE.



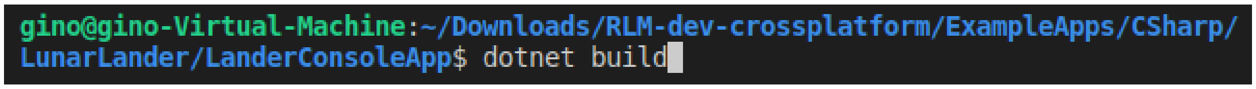
To try opening an application, like the Lunar Lander, navigate through Example Apps > CSharp > LunarLander > LanderConsoleApp.

Right clickthe LanderConsoleApp Folder and select Open in Terminal.



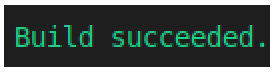
```
ual-Machine:~/Downloads/RLM-dev-crossplatform/ExampleApps/CSharp/
nderConsoleApp$
```

Once the terminal appears, type in dotnet build. This will restore all the needed files to run.

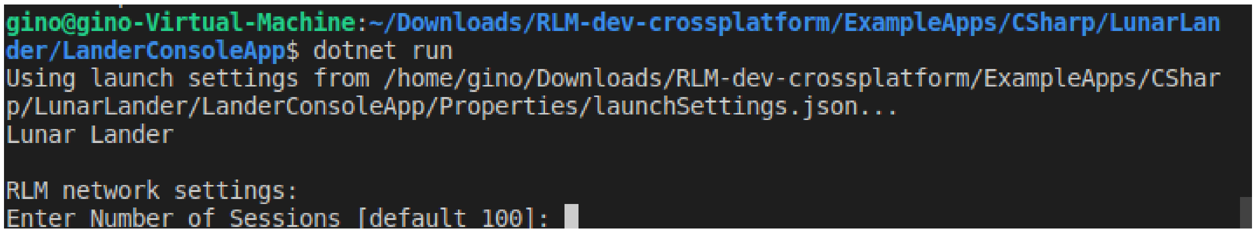


```
gino@gino-Virtual-Machine:~/Downloads/RLM-dev-crossplatform/ExampleApps/CSharp/
LunarLander/LanderConsoleApp$ dotnet build
```

It should prompt you with the following on success.

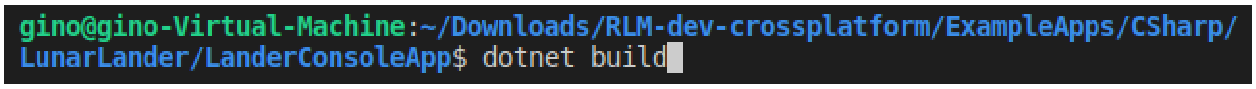

```
Build succeeded.
```

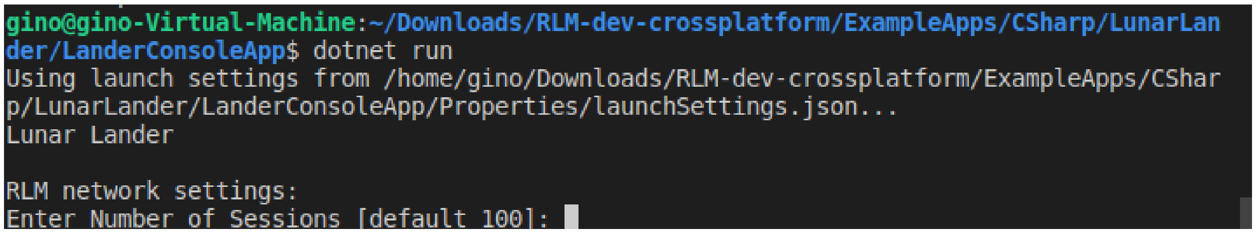
Finally, to run the code, just type in dotnet run.


```
gino@gino-Virtual-Machine:~/Downloads/RLM-dev-crossplatform/ExampleApps/CSharp/LunarLan
der/LanderConsoleApp$ dotnet run
Using launch settings from /home/gino/Downloads/RLM-dev-crossplatform/ExampleApps/CShar
p/LunarLander/LanderConsoleApp/Properties/launchSettings.json...
Lunar Lander

RLM network settings:
Enter Number of Sessions [default 100]:
```

1.7.5. Running without IDE (Terminal)

While the above example uses IDE (Visual Studio Code) You can also run the sample apps directly without installing IDE. This is by using the terminal. Just type in the following:

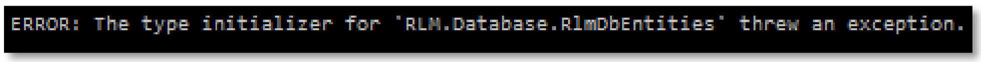

```
gino@gino-Virtual-Machine:~/Downloads/RLM-dev-crossplatform/ExampleApps/CSharp/
LunarLander/LanderConsoleApp$ dotnet build
```


```
gino@gino-Virtual-Machine:~/Downloads/RLM-dev-crossplatform/ExampleApps/CSharp/LunarLan
der/LanderConsoleApp$ dotnet run
Using launch settings from /home/gino/Downloads/RLM-dev-crossplatform/ExampleApps/CShar
p/LunarLander/LanderConsoleApp/Properties/launchSettings.json...
Lunar Lander

RLM network settings:
Enter Number of Sessions [default 100]:
```

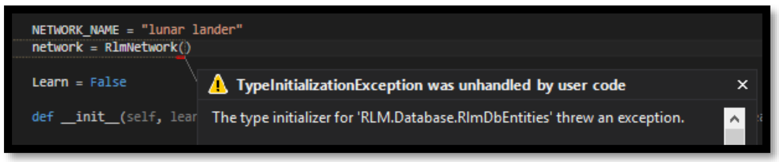
1.8. Common ERRORS

1.8.1. Missing Database

C#


```
ERROR: The type initializer for 'RLM.Database.RlmDbEntities' threw an exception.
```

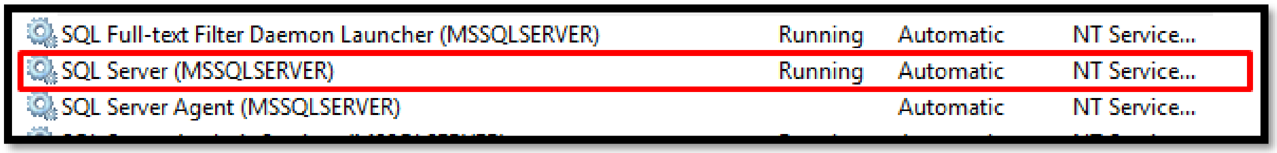
Python


```
NETWORK_NAME = "lunar lander"
network = RlmNetwork()

Learn = False

def __init__(self, lear
```
TypeInitializationException was unhandled by user code

The type initializer for 'RLM.Database.RlmDbEntities' threw an exception.

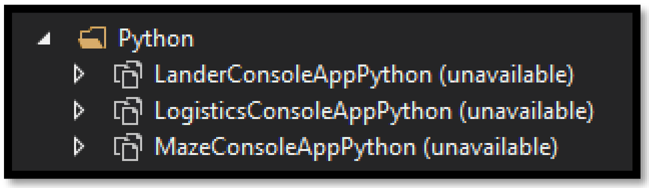
Solution:

You're SQL might not be turned on or existing. You can check this through services.msc



```
SQL Full-text Filter Daemon Launcher (MSSQLSERVER)    Running    Automatic    NT Service...
SQL Server (MSSQLSERVER)                              Running    Automatic    NT Service...
SQL Server Agent (MSSQLSERVER)                                   Automatic    NT Service...
```

1.8.2. Python Tools Not Installed

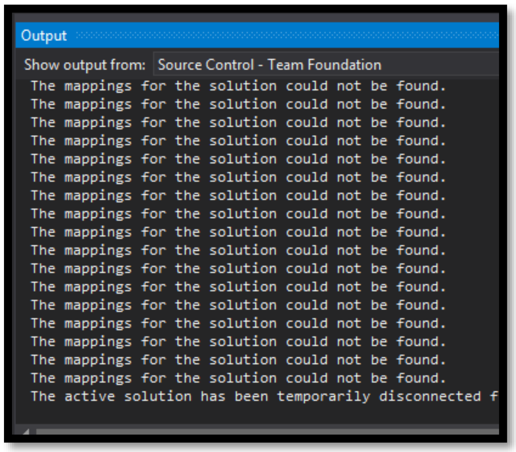


```
Python
    LanderConsoleAppPython (unavailable)
    LogisticsConsoleAppPython (unavailable)
    MazeConsoleAppPython (unavailable)
```

Solution:

You can install Python Tools which can be downloaded HERE. Once installed, just relaunch Microsoft Visual Studio.

1.8.3. The Mappings for the Solution could not be found

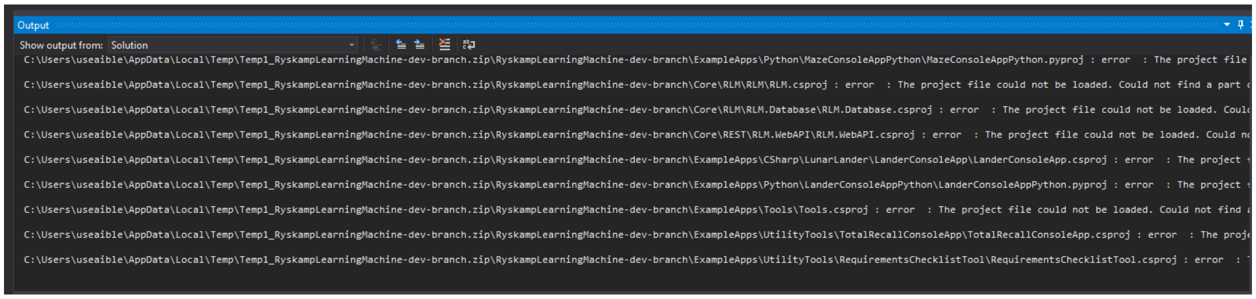


```
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The mappings for the solution could not be found.
The active solution has been temporarily disconnected f
```

Solution:

If you're getting his type of error on your output after first opening it, just disregard it for now.

1.8.4. Project File cannot be loaded



Output

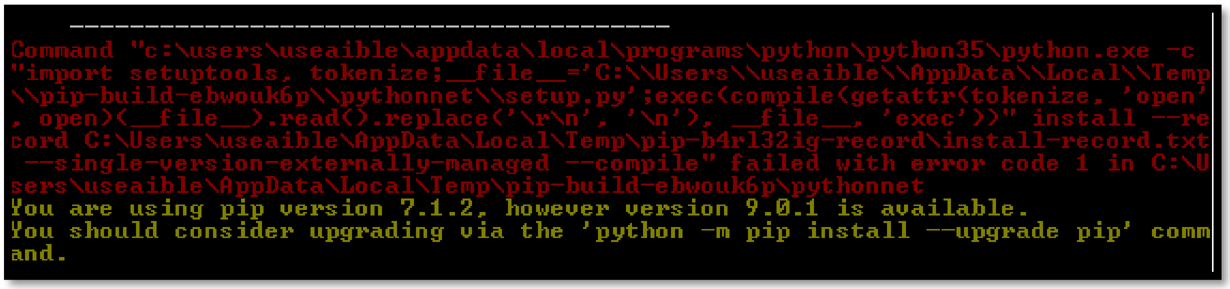
Show output from: Solution

```
C:\Users\useaible\AppData\Local\Temp\Temp1_RyskampLearningMachine-dev-branch.zip\RyskampLearningMachine-dev-branch\ExampleApps\Python\MazeConsoleAppPython\MazeConsoleAppPython.pyproj : error  : The project file

C:\Users\useaible\AppData\Local\Temp\Temp1_RyskampLearningMachine-dev-branch.zip\RyskampLearningMachine-dev-branch\Core\RLM\RLM\RLM.csproj : error  : The project file could not be loaded. Could not find a part

C:\Users\useaible\AppData\Local\Temp\Temp1_RyskampLearningMachine-dev-branch.zip\RyskampLearningMachine-dev-branch\Core\RLM\RLM.Database\RLM.Database.csproj : error  : The project file could not be loaded. Coul

C:\Users\useaible\AppData\Local\Temp\Temp1_RyskampLearningMachine-dev-branch.zip\RyskampLearningMachine-dev-branch\Core\REST\RLM.WebAPI\RLM.WebAPI.csproj : error  : The project file could not be loaded. Could n

C:\Users\useaible\AppData\Local\Temp\Temp1_RyskampLearningMachine-dev-branch.zip\RyskampLearningMachine-dev-branch\ExampleApps\CSharp\LunarLander\LanderConsoleApp\LanderConsoleApp.csproj : error  : The project

C:\Users\useaible\AppData\Local\Temp\Temp1_RyskampLearningMachine-dev-branch.zip\RyskampLearningMachine-dev-branch\ExampleApps\Python\LanderConsoleAppPython\LanderConsoleAppPython.pyproj : error  : The project

C:\Users\useaible\AppData\Local\Temp\Temp1_RyskampLearningMachine-dev-branch.zip\RyskampLearningMachine-dev-branch\ExampleApps\Tools\Tools.csproj : error  : The project file could not be loaded. Could not find

C:\Users\useaible\AppData\Local\Temp\Temp1_RyskampLearningMachine-dev-branch.zip\RyskampLearningMachine-dev-branch\ExampleApps\UtilityTools\TotalRecallConsoleApp\TotalRecallConsoleApp.csproj : error  : The proj

C:\Users\useaible\AppData\Local\Temp\Temp1_RyskampLearningMachine-dev-branch.zip\RyskampLearningMachine-dev-branch\ExampleApps\UtilityTools\RequirementsChecklistTool\RequirementsChecklistTool.csproj : error  :
```

Solution:

You will need to extract the solution first from the zip file to be able to open the project without problems.

1.8.5. Error while installing Python.Net through Command Prompt



```
    ----------------------------------------
Command "c:\users\useaible\appdata\local\programs\python\python35\python.exe -c
"import setuptools, tokenize;__file__='C:\\Users\\useaible\\AppData\\Local\\Temp
\\pip-build-ebwouk6p\\pythonnet\\setup.py';exec(compile(getattr(tokenize, 'open'
, open)(__file__).read().replace('\r\n', '\n'), __file__, 'exec'))" install --re
cord C:\Users\useaible\AppData\Local\Temp\pip-b4rl32ig-record\install-record.txt
 --single-version-externally-managed --compile" failed with error code 1 in C:\U
sers\useaible\AppData\Local\Temp\pip-build-ebwouk6p\pythonnet
You are using pip version 7.1.2, however version 9.0.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' comm
and.
```

Solution:

This is an error easily remedied by updating your Python PIP. To do so, type the command below on the Scripts directory

```
pip install –upgrade pip
```

Note that there are two dashes (not -, but --)

It should update, then after which you can proceed to installing Python.Net with the command

```
pip install pythonnet
```

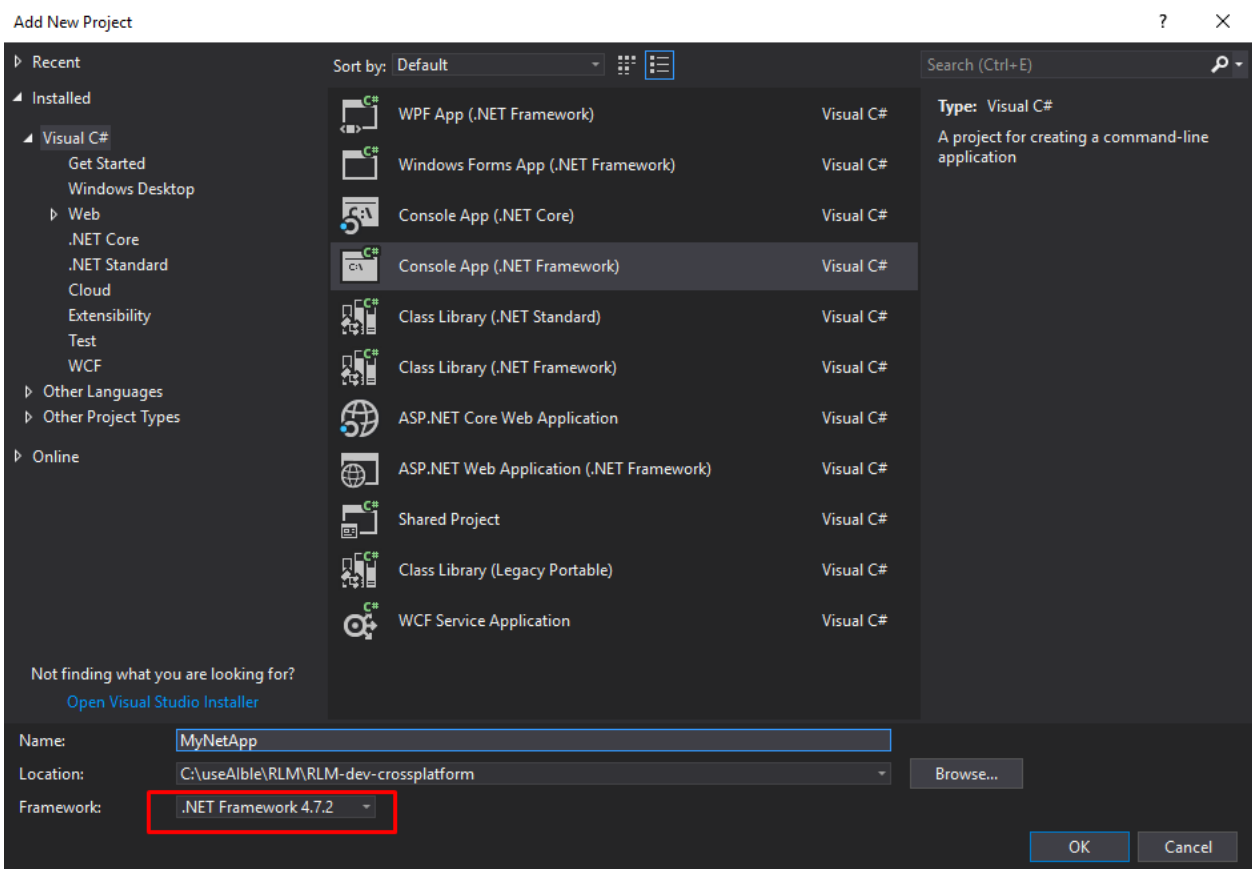
Chapter 2: The RLM Setup

2.1. Setting up References

.Net Framework Set up

We want you to also set up your own game to try with our RLM. For us to do that, we're going to walk you through setting up your network and training the RLM to fit your needs. You'll be glad to know that it is very easy.

If you want to create a sample project, don't make it under the core folder. You need to set it to .Net Framework 4.7.2


Add New Project
Recent
Installed
Visual C#
Get Started
Windows Desktop
Web
.NET Core
.NET Standard
Cloud
Extensibility
Test
WCF
Other Languages
Other Project Types
Online
Not finding what you are looking for?
Open Visual Studio Installer
Sort by: Default
WPF App (.NET Framework) Visual C#
Windows Forms App (.NET Framework) Visual C#
Console App (.NET Core) Visual C#
Console App (.NET Framework) Visual C#
Class Library (.NET Standard) Visual C#
Class Library (.NET Framework) Visual C#
ASP.NET Core Web Application Visual C#
ASP.NET Web Application (.NET Framework) Visual C#
Shared Project Visual C#
Class Library (Legacy Portable) Visual C#
WCF Service Application Visual C#
Search (Ctrl+E)
Browse...
OK
Cancel

Type: Visual C#

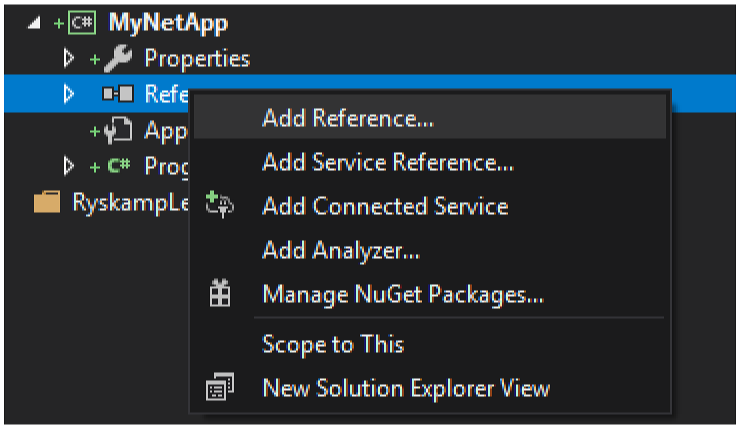
A project for creating a command-line application

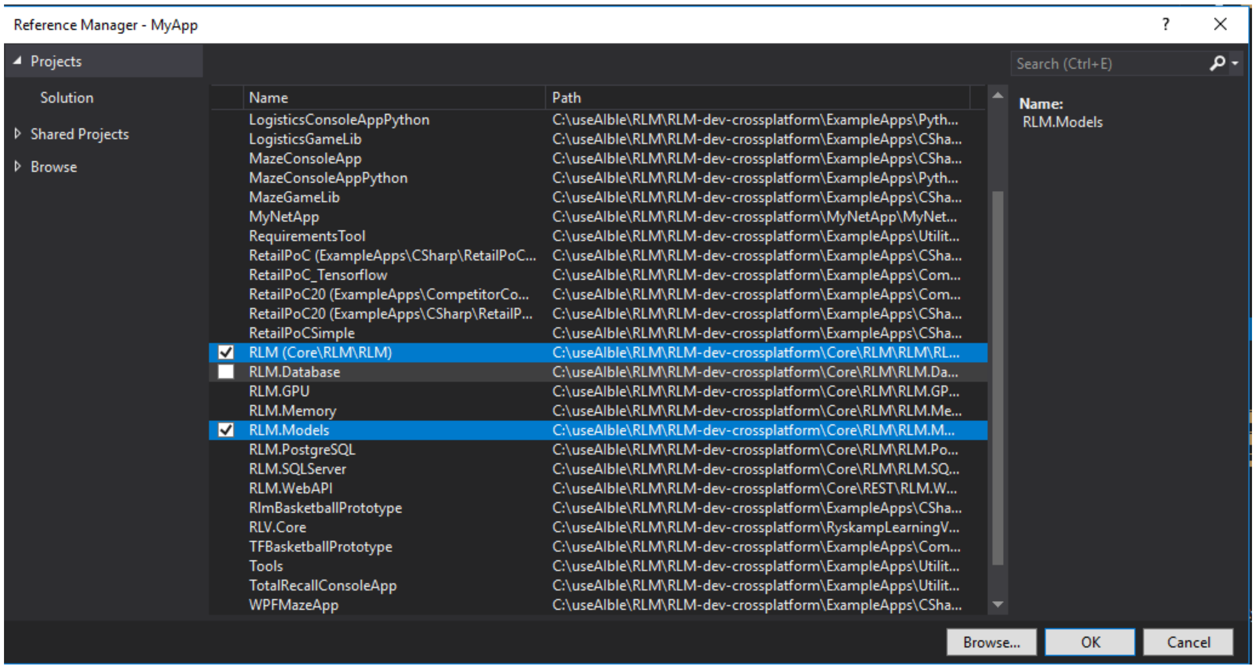
Name: MyNetApp

Location: C:\useAIble\RLM\RLM-dev-crossplatform

Framework: .NET Framework 4.7.2

Once you've created your project, add the references. The most important ones are the RLM Core and RLM Models.




Reference Manager - MyApp

Projects
Solution
Shared Projects
Browse

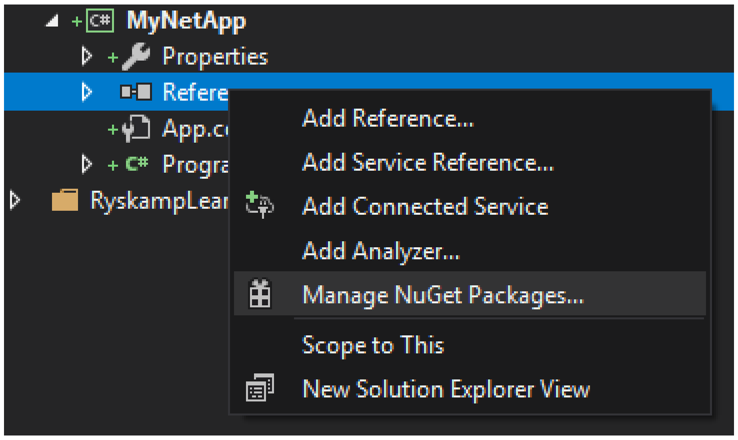
Search (Ctrl+E)

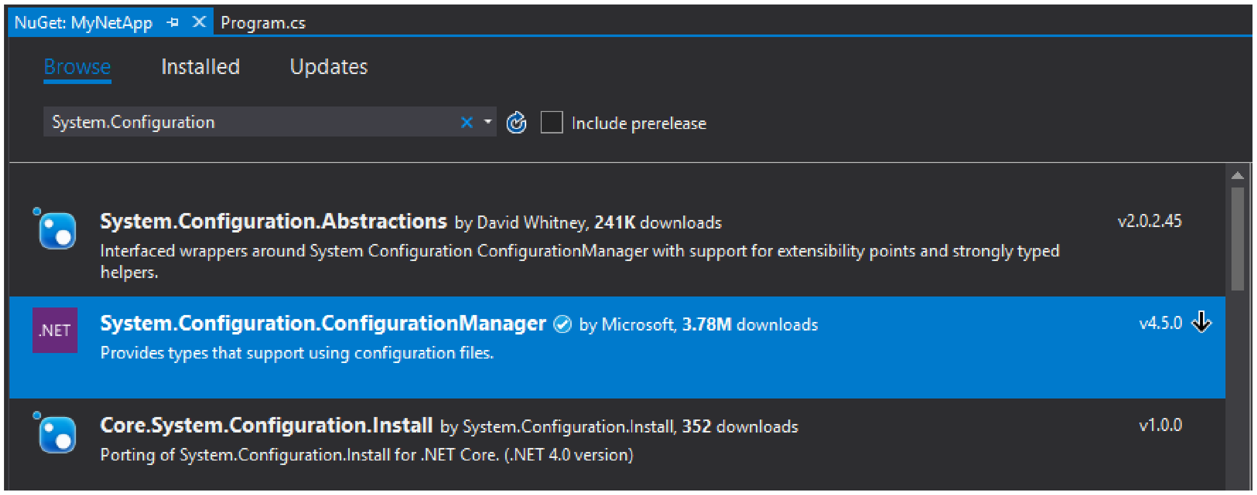
	Name	Path
	LogisticsConsoleAppPython	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Pyth...
	LogisticsGameLib	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	MazeConsoleApp	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	MazeConsoleAppPython	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Pyth...
	MazeGameLib	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	MyNetApp	C:\useAIble\RLM\RLM-dev-crossplatform\MyNetApp\MyNet...
	RequirementsTool	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Utilit...
	RetailPoC (ExampleApps\CSharp\RetailPoC...	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	RetailPoC_Tensorflow	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Com...
	RetailPoC20 (ExampleApps\CompetitorCo...	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Com...
	RetailPoC20 (ExampleApps\CSharp\RetailP...	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	RetailPoCSimple	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
☑	RLM (Core\RLM\RLM)	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM\RL...
☐	RLM.Database	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.Da...
	RLM.GPU	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.GP...
	RLM.Memory	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.Me...
☑	RLM.Models	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.M...
	RLM.PostgreSQL	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.Po...
	RLM.SQLServer	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.SQ...
	RLM.WebAPI	C:\useAIble\RLM\RLM-dev-crossplatform\Core\REST\RLM.W...
	RlmBasketballPrototype	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	RLV.Core	C:\useAIble\RLM\RLM-dev-crossplatform\RyskampLearningV...
	TFBasketballPrototype	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Com...
	Tools	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Utilit...
	TotalRecallConsoleApp	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Utilit...
	WPFMazeApp	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...

Name:
RLM.Models

Browse... OK Cancel

You also need to include the following packages:




NuGet: MyNetApp Program.cs

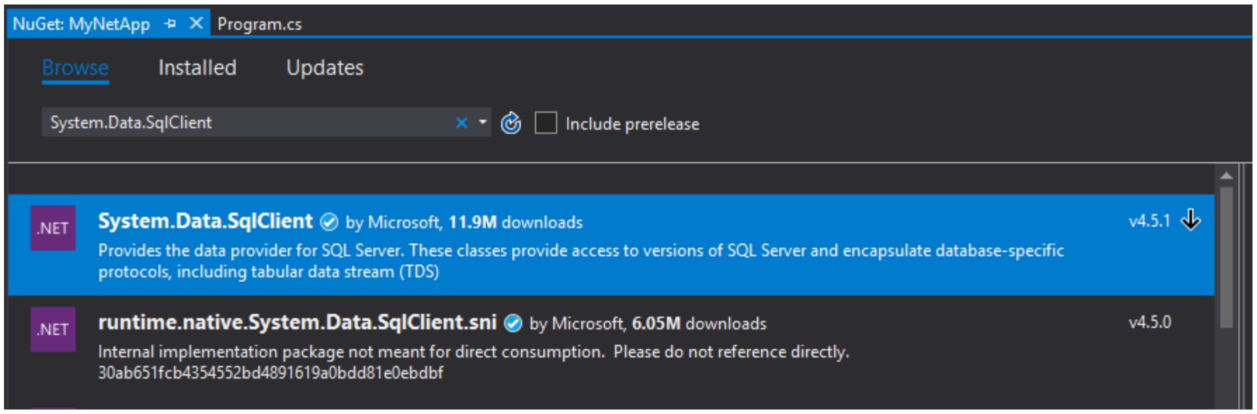
Browse Installed Updates

System.Configuration Include prerelease

System.Configuration.Abstractions by David Whitney, **241K** downloads v2.0.2.45
Interfaced wrappers around System Configuration ConfigurationManager with support for extensibility points and strongly typed helpers.

System.Configuration.ConfigurationManager by Microsoft, **3.78M** downloads v4.5.0
Provides types that support using configuration files.

Core.System.Configuration.Install by System.Configuration.Install, **352** downloads v1.0.0
Porting of System.Configuration.Install for .NET Core. (.NET 4.0 version)

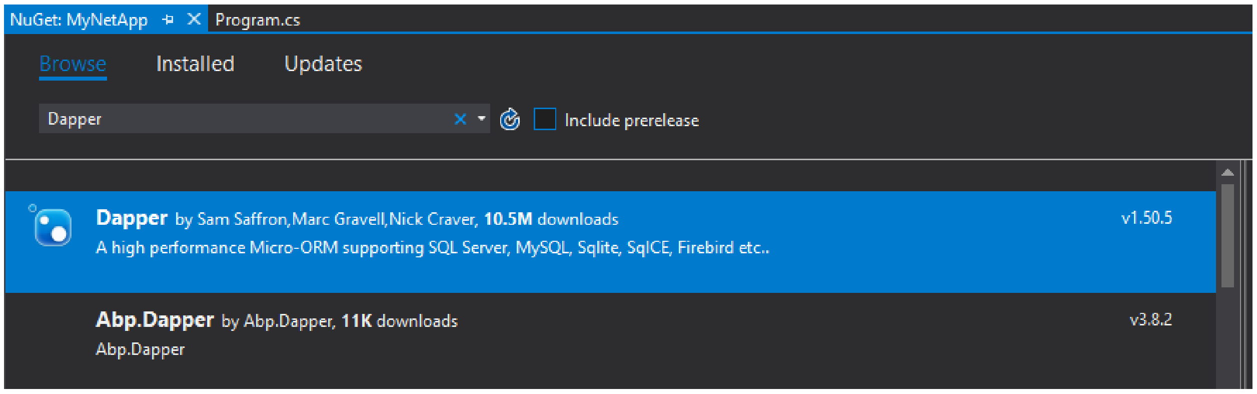

NuGet: MyNetApp Program.cs

Browse Installed Updates

System.Data.SqlClient Include prerelease

System.Data.SqlClient by Microsoft, **11.9M** downloads v4.5.1
Provides the data provider for SQL Server. These classes provide access to versions of SQL Server and encapsulate database-specific protocols, including tabular data stream (TDS)

runtime.native.System.Data.SqlClient.sni by Microsoft, **6.05M** downloads v4.5.0
Internal implementation package not meant for direct consumption. Please do not reference directly.
30ab651fcb4354552bd4891619a0bdd81e0ebdbf

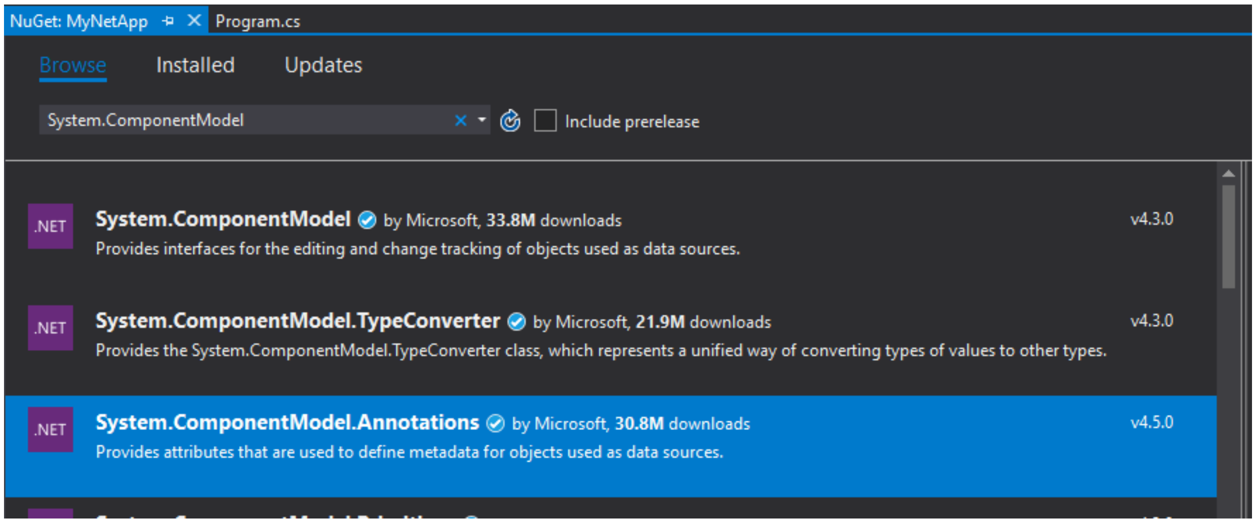

NuGet: MyNetApp Program.cs

Browse Installed Updates

Dapper Include prerelease

Dapper by Sam Saffron,Marc Gravell,Nick Craver, **10.5M** downloads v1.50.5
A high performance Micro-ORM supporting SQL Server, MySQL, Sqlite, SqlCE, Firebird etc..

Abp.Dapper by Abp.Dapper, **11K** downloads v3.8.2
Abp.Dapper

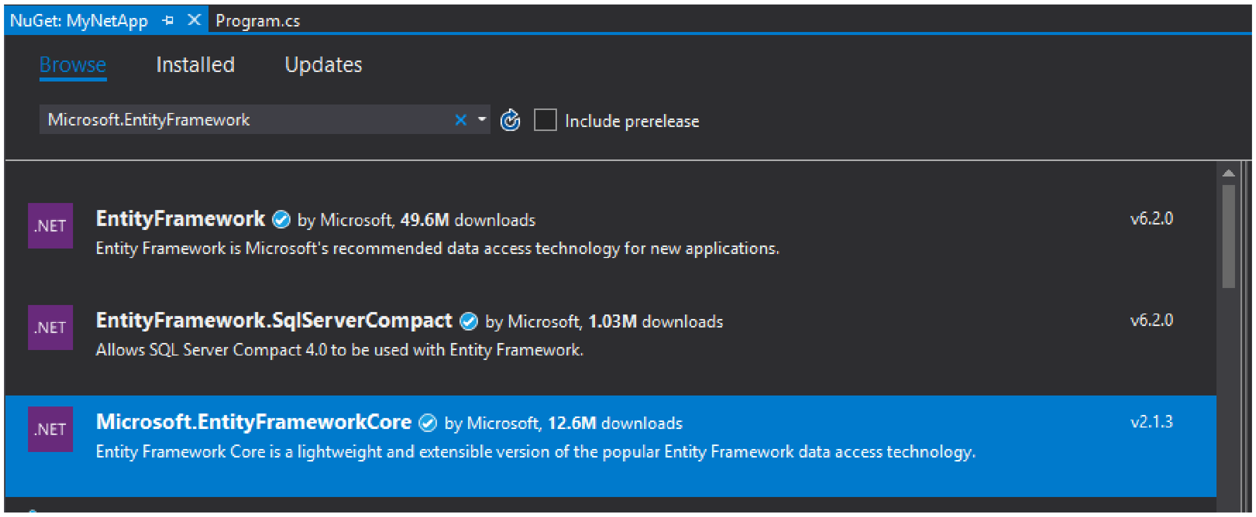


System.ComponentModel

System.ComponentModel by Microsoft, 33.8M downloads v4.3.0
Provides interfaces for the editing and change tracking of objects used as data sources.

System.ComponentModel.TypeConverter by Microsoft, 21.9M downloads v4.3.0
Provides the System.ComponentModel.TypeConverter class, which represents a unified way of converting types of values to other types.

System.ComponentModel.Annotations by Microsoft, 30.8M downloads v4.5.0
Provides attributes that are used to define metadata for objects used as data sources.

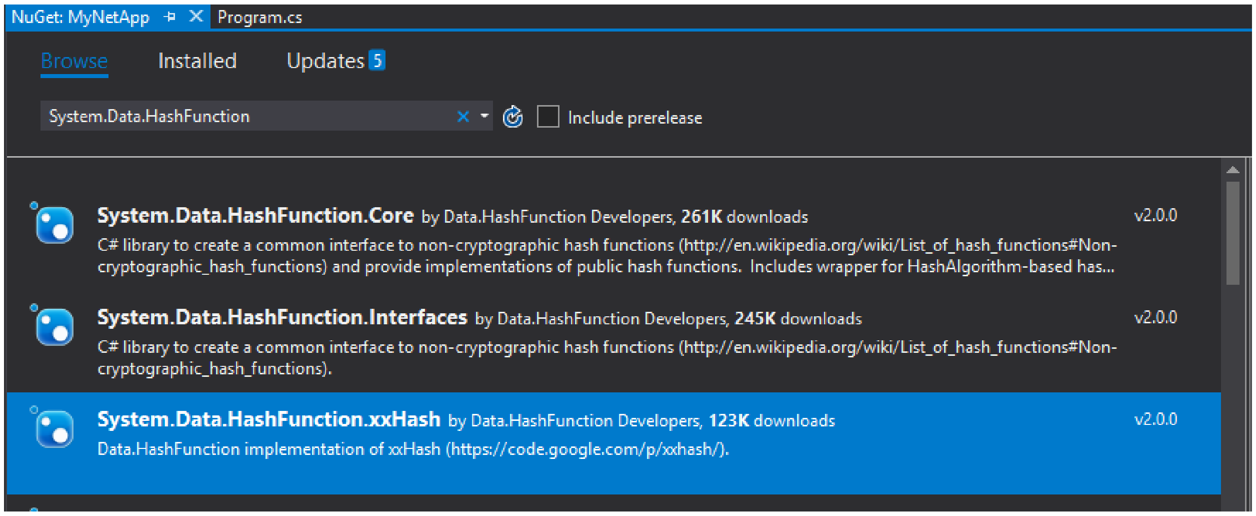


Microsoft.EntityFramework

EntityFramework by Microsoft, 49.6M downloads v6.2.0
Entity Framework is Microsoft's recommended data access technology for new applications.

EntityFramework.SqlServerCompact by Microsoft, 1.03M downloads v6.2.0
Allows SQL Server Compact 4.0 to be used with Entity Framework.

Microsoft.EntityFrameworkCore by Microsoft, 12.6M downloads v2.1.3
Entity Framework Core is a lightweight and extensible version of the popular Entity Framework data access technology.



System.Data.HashFunction

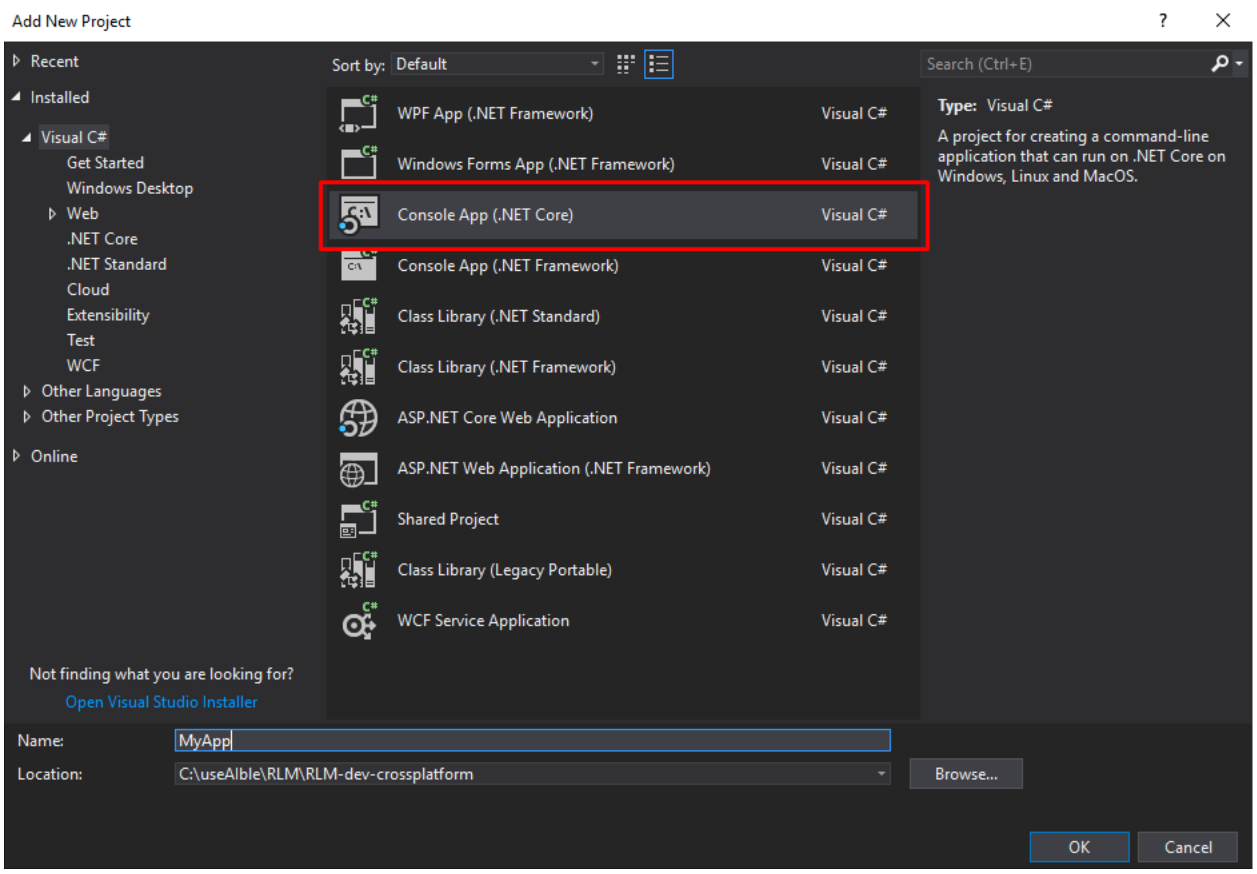
System.Data.HashFunction.Core by Data.HashFunction Developers, 261K downloads v2.0.0
C# library to create a common interface to non-cryptographic hash functions (http://en.wikipedia.org/wiki/List_of_hash_functions#Non-cryptographic_hash_functions) and provide implementations of public hash functions. Includes wrapper for HashAlgorithm-based has...

System.Data.HashFunction.Interfaces by Data.HashFunction Developers, 245K downloads v2.0.0
C# library to create a common interface to non-cryptographic hash functions (http://en.wikipedia.org/wiki/List_of_hash_functions#Non-cryptographic_hash_functions).

System.Data.HashFunction.xxHash by Data.HashFunction Developers, 123K downloads v2.0.0
Data.HashFunction implementation of xxHash (https://code.google.com/p/xxhash/).

.Net Core Set up

You can also create your sample project in .Net Core. You need to follow the same steps as mentioned above with the exception of installing the packages.


Add New Project

Type: Visual C#

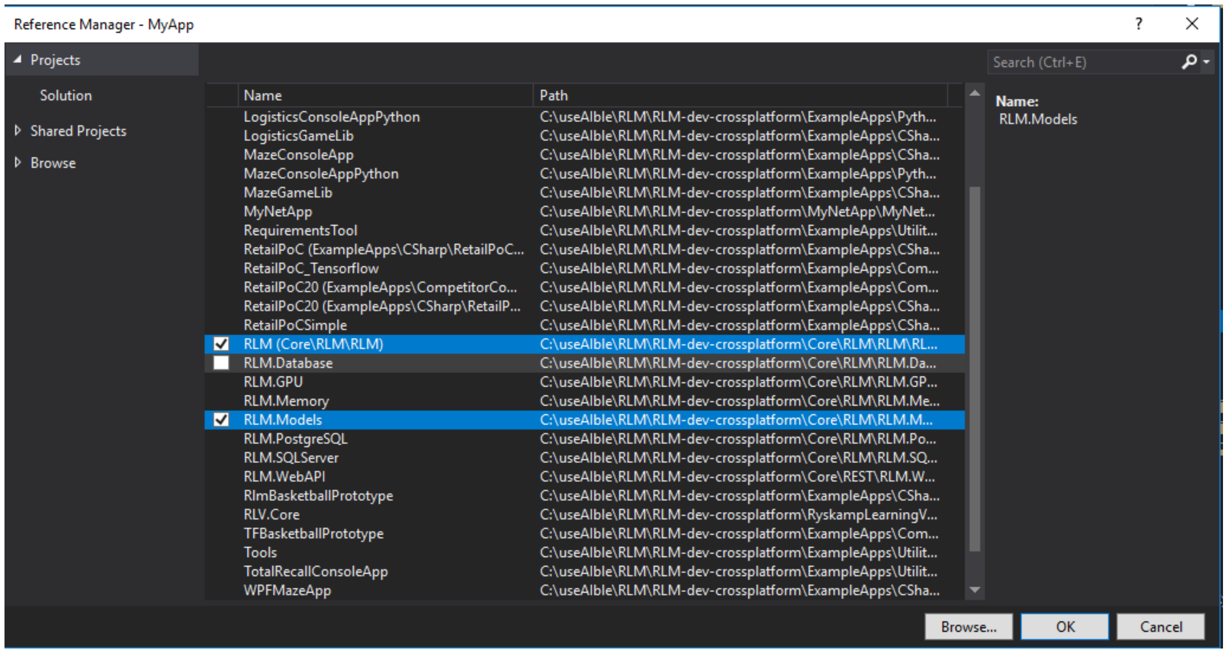
A project for creating a command-line application that can run on .NET Core on Windows, Linux and MacOS.

WPF App (.NET Framework)	Visual C#
Windows Forms App (.NET Framework)	Visual C#
Console App (.NET Core)	Visual C#
Console App (.NET Framework)	Visual C#
Class Library (.NET Standard)	Visual C#
Class Library (.NET Framework)	Visual C#
ASP.NET Core Web Application	Visual C#
ASP.NET Web Application (.NET Framework)	Visual C#
Shared Project	Visual C#
Class Library (Legacy Portable)	Visual C#
WCF Service Application	Visual C#

Name: MyApp

Location: C:\useAIble\RLM\RLM-dev-crossplatform

Same as with using .Net Framework, the same references need to be added.


Reference Manager - MyApp

	Name	Path
	LogisticsConsoleAppPython	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Pyth...
	LogisticsGameLib	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	MazeConsoleApp	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	MazeConsoleAppPython	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Pyth...
	MazeGameLib	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	MyNetApp	C:\useAIble\RLM\RLM-dev-crossplatform\MyNetApp\MyNet...
	RequirementsTool	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Utilit...
	RetailPoC (ExampleApps\CSharp\RetailPoC...	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	RetailPoC_Tensorflow	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Com...
	RetailPoC20 (ExampleApps\CompetitorCo...	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Com...
	RetailPoC20 (ExampleApps\CSharp\RetailP...	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	RetailPoCSimple	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
✓	RLM (Core\RLM\RLM)	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM\RL...
☐	RLM.Database	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.Da...
	RLM.GPU	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.GP...
	RLM.Memory	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.Me...
✓	RLM.Models	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.M...
	RLM.PostgreSQL	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.Po...
	RLM.SQLServer	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.SQ...
	RLM.WebAPI	C:\useAIble\RLM\RLM-dev-crossplatform\Core\REST\RLM.W...
	RlmBasketballPrototype	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	RLV.Core	C:\useAIble\RLM\RLM-dev-crossplatform\RyskampLearningV...
	TFBasketballPrototype	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Com...
	Tools	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Utilit...
	TotalRecallConsoleApp	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Utilit...
	WPFMazeApp	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...

Name:
RLM.Models

2.2. RLM Network

The RLM is quite easy to set up and this is by design to rid the notion that working with A.I. or Machine Learning is 'rocket science'.

First and foremost, we need to instantiate an instance of the RlmNetwork class. This represents our Neural Network where the Rneurons (neurons), Inputs, Outputs, and other settings are kept and utilized during training and prediction.

There are two ways you can instantiate the RlmNetwork class, first is by calling the default constructor:

```
// constructor with no parameters
var network = new RlmNetwork();
```

This will instantiate the RlmNetwork with Data Persistence being turned off which means RLM metadata and training data will not be saved and reused at a later time. This works best when you only need to do a series of quick tests like figuring out certain settings for the RLM or for use cases where you just need it to do a one-time train and predict to get the results immediately without the need to load the network later on.

Another way to instantiate the RlmNetwork would be to call the overloaded constructor that accepts the IRlmDbData implementation

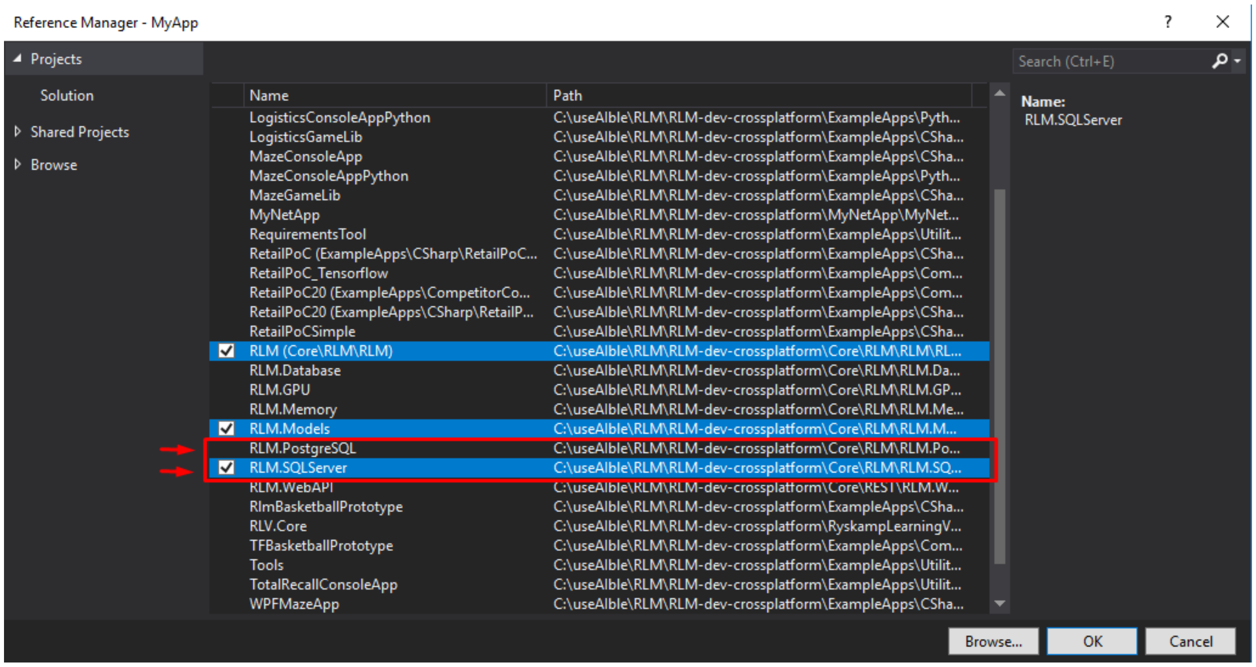
The IRlmDbData interface will be utilized by different database provider implementations.

```
public interface IRlmDbData
{
    string DatabaseName { get; set; }

    void CreateTable<T>();
    void Create<T>(T entity);
    void Create<T>(IEnumerable<T> entity);
    void Update<T>(T entity);
    IEnumerable<T> FindAll<T>();
    IEnumerable<TResult> FindAll<TBase, TResult>();
    T FindByID<T>(long id);
    TResult FindByID<TBase, TResult>(long id);
    T FindByName<T>(string name);
    TResult FindByName<TBase, TResult>(string name);
    void DropDB(string dbName);
    int Count<T>();
    TResult Max<TSource, TResult>(Expression<Func<TSource, TResult>> propertyExpr);

    // Load network specific functions
    int CountBestSolutions();
    IEnumerable<BestSolution> LoadBestSolutions();
    IEnumerable<Session> LoadVisibleSessions();
}
```

We have 2 pre-built implementations:

- SQLServer
- PostGreSQL

Note that these need to be referenced. For the database, you have an option to choose either of the two.



Reference Manager - MyApp

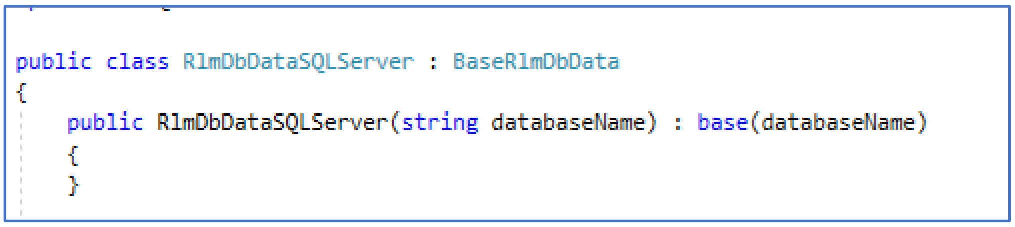
	Name	Path
	LogisticsConsoleAppPython	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Pyth...
	LogisticsGameLib	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	MazeConsoleApp	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	MazeConsoleAppPython	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Pyth...
	MazeGameLib	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	MyNetApp	C:\useAIble\RLM\RLM-dev-crossplatform\MyNetApp\MyNet...
	RequirementsTool	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Utilit...
	RetailPoC (ExampleApps\CSharp\RetailPoC...	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	RetailPoC_Tensorflow	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Com...
	RetailPoC20 (ExampleApps\CompetitorCo...	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Com...
	RetailPoC20 (ExampleApps\CSharp\RetailP...	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	RetailPoCSimple	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
✓	RLM (Core\RLM\RLM)	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM\RL...
	RLM.Database	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.Da...
	RLM.GPU	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.GP...
	RLM.Memory	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.Me...
✓	RLM.Models	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.M...
	RLM.PostgreSQL	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.Po...
✓	RLM.SQLServer	C:\useAIble\RLM\RLM-dev-crossplatform\Core\RLM\RLM.SQ...
	RLM.WebAPI	C:\useAIble\RLM\RLM-dev-crossplatform\Core\REST\RLM.W...
	RlmBasketballPrototype	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...
	RLV.Core	C:\useAIble\RLM\RLM-dev-crossplatform\RyskampLearningV...
	TFBasketballPrototype	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Com...
	Tools	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Utilit...
	TotalRecallConsoleApp	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\Utilit...
	WPFMazeApp	C:\useAIble\RLM\RLM-dev-crossplatform\ExampleApps\CSha...

Name:
RLM.SQLServer

2.1.1. Setting up Own Implementation of IRlmDbData

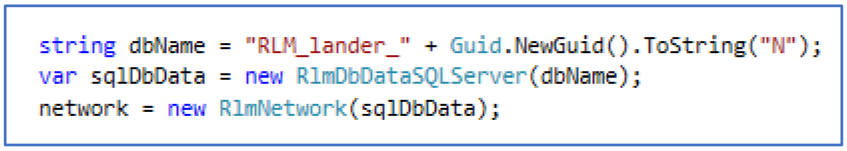
Now that you're done setting up the references we will need to start coding.

To start, the IRlmDbData should not be implemented directly. Instead, you should inherit the base class which is BaseRlmDbData. The reason for this is that there is code implemented on the BaseRlmDbData that is needed by the program flow of the RLM.



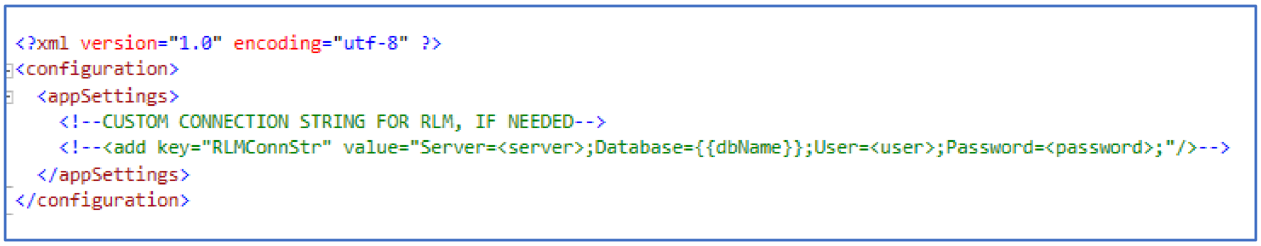
```
public class RlmDbDataSQLServer : BaseRlmDbData
{
    public RlmDbDataSQLServer(string databaseName) : base(databaseName)
    {
    }
```

Second is to implement all the abstract methods to create, read, update and delete your database. We will now be using our new or pre-built database provider implementation.



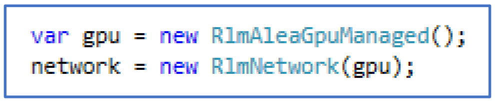
```
string dbName = "RLM_lander_" + Guid.NewGuid().ToString("N");
var sqlDbData = new RlmDbDataSQLServer(dbName);
network = new RlmNetwork(sqlDbData);
```

If you need to customize your connection string, go to app config and add your connection string.



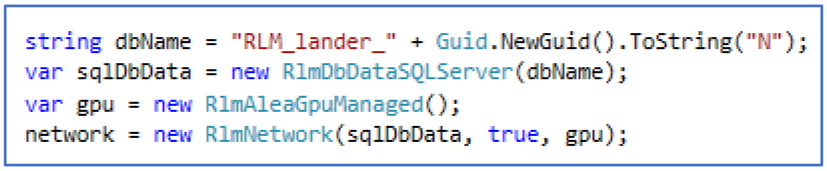
```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>
    <!--CUSTOM CONNECTION STRING FOR RLM, IF NEEDED-->
    <!--<add key="RLMConnStr" value="Server=<server>;Database={{dbName}};User=<user>;Password=<password>;"/>-->
  </appSettings>
</configuration>
```

Another option is to utilize the GPU for processing (use RlmAleaGpuManaged). You need to reference RLM.GPU project.



```
var gpu = new RlmAleaGpuManaged();
network = new RlmNetwork(gpu);
```

Finally, if you want to use the gpu and a database, you can follow the constructor below.



```
string dbName = "RLM_lander_" + Guid.NewGuid().ToString("N");
var sqlDbData = new RlmDbDataSQLServer(dbName);
var gpu = new RlmAleaGpuManaged();
network = new RlmNetwork(sqlDbData, true, gpu);
```

In addition, we have added 2 events to handle saving in the Database. This sends you data on the progress and the status of the data persistence process.

```
rlmNet.DataPersistenceComplete += RlmNet_DataPersistenceComplete;
rlmNet.DataPersistenceProgress += RlmNet_DataPersistenceProgress;
```

2.3. Setting up the Inputs and Outputs

The second step is to identify the Inputs and Outputs and set them up in a way that the RLM is able to identify and use them for training.

```
public RlmIO(String name, String dotnettype, double min, double max, Enums.RlmInputType type, long id = 0)
    : this(name, dotnettype, min, max, id)
{
    this.Type = type;
}
```

You will have to make a list of the RlmIO object as the method only accepts a list.

```
var ins = new List<RlmIO>();
ins.Add(new RlmIO("RlmInputOne", typeof(bool).ToString(), 0, 1, RlmInputType.Distinct));
ins.Add(new RlmIO("RlmInputTwo", typeof(bool).ToString(), 0, 1, RlmInputType.Distinct));
```

Above you can see that we have made a new list of inputs. First we create the list and then we fill it up with RlmIO objects, which we defined above as RlmInputOne and RlmInputTwo, both are set to boolean with .Net type and are Distinct. We have not set IDs, these are optional.

Outputs are made in the same way yet a little different. For it, we will be using a different constructor for the RlmIO object.

```
public RlmIO(String name, String dotnettype, double min, double max, long id = 0)
{
    this.Name = name;
    this.DotNetType = dotnettype;
    this.Min = min;
    this.Max = max;
    this.ID = id;
}
```

This will not have an RLM input type being that it's not an input.

```
var outs = new List<RlmIO>();
outs.Add(new RlmIO("RlmOutput", typeof(bool).ToString(), 0, 1));
```

Similar to how we made the inputs list, we made the outputs list and added a new output RlmOutput. Note that there is no RLM Input type parameter.

2.4. Loading the Network

Loading a Network means to save in memory all the saved settings for that network. To load the network, we need to check first if a network is already existing and that if it is, we should load that instead.

```
rlmNet.LoadNetwork();
```

If there's no network of the specified name, then we should create a new Network. However, the LoadNetwork method does not make a new network but only returns a boolean type, so we will need to use the NewNetwork method.

```
rlmNet.NewNetwork("RlmNetwork", ins, outs);
```

The NewNetwork method sets up a network together with its inputs and outputs. As we've discussed in the previous section, the lists of inputs and outputs are going to be used as parameters. We will then need to combine both in an IF statement.

```
if (!rlmNet.LoadNetwork())
{
    var ins = new List<RlmIO>();
    ins.Add(new RlmIO("RlmInputOne", typeof(bool).ToString(), 0, 1, RlmInputType.Distinct));
    ins.Add(new RlmIO("RlmInputTwo", typeof(bool).ToString(), 0, 1, RlmInputType.Distinct));

    var outs = new List<RlmIO>();
    outs.Add(new RlmIO("RlmOutput", typeof(bool).ToString(), 0, 1));

    rlmNet.NewNetwork("RlmNetwork", ins, outs);
}
```

Here we check if a database is existing and if it does, we load it, however, if it is not existing, then we create a new one.

2.5. Additional Settings

Part of the network setup are settings that help how much efficiently the engine trains. This would be the percentage of randomness, the number of sessions, and for linear problems, the linear bracket.

The NumSessions is an int type that determines how many times the RLM trains. The more sessions, the more opportunities for the engine to learn and apply its learning.

```
rlmNet.NumSessions = 50;
```

Shown above is how to set up the NumSessions to 50 sessions.

The Randomness Property, defined with its min and max, dictates how much randomness is applied on the training of the engine. As it moves forward to more sessions, the randomness depreciates which allows the AI to apply its learning more as opposed to learning something new until it reaches its set EndRandomness.

```
rlmNet.StartRandomness = 100;
rlmNet.EndRandomness = 0;
```

Here we see the engine is set to start at 100% randomness, allowing it to have no limitations in trying different things, of which will depreciate as it moves further through its sessions until it reaches the last session which then its Randomness will be 0, making it only apply its learning.

Another setting is the Linear Bracket used for Linear Inputs. A good indication on when to use this is when an input's value is large and granular enough that you may consider close neighboring values to be the same depending on the linear bracket.

```
rlmNet.MaxLinearBracket = 15;
rlmNet.MinLinearBracket = 3;
```

Here we set the Min and Max to 3 and 15, respectively. In our testing, this has worked with the best results.

Another example would be, the values 7 and 8 will be processed by the RLM similarly if those two numbers are within the linear bracket at that time. But if this input was set as a Distinct Input then 7 and 8 would be two different values and processed differently by the RLM. The linear bracket (Max - Min) decreases after each session and is applied at each cycle when getting the best solution.

There is no perfect number setting for the Linear Bracket, it’s more of a trial and error and check what setting allows the engine to learn more efficiently.

Chapter 3: Identifying a Problem

3.1. The Lunar Lander Problem

The Lunar Lander is a linear problem wherein a player attempts to land a spacecraft with limited fuel from midair by utilizing its thrusters.

3.1.1. The Goal

The player must safely land the spacecraft by ensuring that the falling velocity is near 0 m/s. The player does this by firing the thrusters, ensuring that there is less waste of fuel by thrusting only when needed.

```
public LanderSimulator()
{
    Fuel = 200;
    Seconds = 0;
    Altitude = 10000;
    Velocity = 0;
}
```

3.1.2. Sample Simulation

We will be creating a Lunar Lander Simulator that will handle calculation and session scoring.

This lunar lander simulator class will be the one to process outputs from the RLM and generate feedback in the form of a score of which in turn the RLM makes into a best solution. We will be discussing how we made the simulator class and ultimately show the full code at the end.

To start, we need to discuss the variables to be used, categorized into three: Inputs, Outputs, and others.

Inputs

The inputs are the ones passed to the RLM for it to generate a decision and return an output.

1. Fuel (Integer)
   - Fuel is the resource used when the spacecraft fires its thrusters.
   - Fuel is a static value assigned at initialization, with default at 200
   - Every time a thrust is made, 1 point of fuel is deducted.
   - We use fuel as a requirement for thrusting. The player will not be able to fire thrusters without fuel.

```
if (thrust && Fuel > 0)
{
    Fuel--;
    Velocity += Thrust;
}
```

2. Velocity (Integer)
   - This is the main factor in resolving if the spacecraft crashes as it reaches the landing point.
   - Downward movement is represented in negative velocity.
   - The spacecraft must be near zero velocity to safely land.
   - Velocity is added value every cycle with the added amount equal to the set Gravity

```
Velocity -= Gravity;
```

   - The max and min is based on the value of the Terminal Velocity

```
Velocity = Math.Max(-TerminalVelocity, Velocity);
Velocity = Math.Min(TerminalVelocity, Velocity);
```

3. Altitude (Double)
   - Determines the current distance from the landing area.
   - It is recommended that as long as far from the landing point, the player can be lenient in using the thrust.
   - Altitude has the default value of 10000, which would be the spacecraft's starting point.
   - Each cycle, the altitude is added with the Velocity for that cycle.
   - Minimum altitude is set to 0, with an added check in case it becomes lower than 0

```
if (Altitude < 0)
    Altitude = 0;
```

4. Seconds (Integer)
   - This is simply the duration of the game until the spacecraft has landed.
   - This is used in the scoring.
   - This is incremented every turn and doesn't mean the actually seconds, but the cycle.

```
Seconds++;
```

Outputs

Outputs are the actions that the simulator expects from the player. So in this scenario, we would like the RLM to return if we want to thrust or not.

1. Thrust (Boolean)
   - Action received from user/RLM that commands the spacecraft to fire its thrusters and lower down fall speed.
   - Continuous thrusting will result in positive velocity and will result in the spacecraft to move up.
   - This will only thrust given that there is still remaining fuel

```
if (thrust && Fuel > 0)
{
    Fuel--;
    Velocity += Thrust;
}
```

Other Variables

There are also other variables that we will be using that will affect how the game works. But these are not things that directly influence the decision of the player, like the inputs.

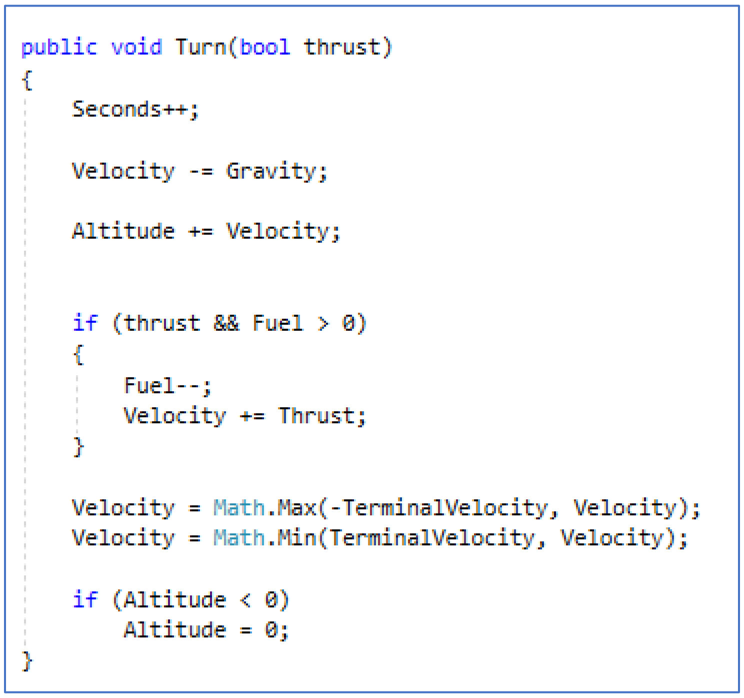
```
public const double Gravity = 1.62;
public const double Thrust = 10;
public const double TerminalVelocity = 40;
```

1. Gravity
   - Is the value increase to velocity each cycle as it goes down

2. ThrustDistance
   - The value added to altitude whenever the player fires the thrusters
3. TerminalVelocity
   - The maximum velocity set for the spacecraft

3.1.3. Code

All these inputs, outputs and other variables are used in a function called Turn, which calculates the results for each cycle



```
public void Turn(bool thrust)
{
    Seconds++;

    Velocity -= Gravity;

    Altitude += Velocity;


    if (thrust && Fuel > 0)
    {
        Fuel--;
        Velocity += Thrust;
    }

    Velocity = Math.Max(-TerminalVelocity, Velocity);
    Velocity = Math.Min(TerminalVelocity, Velocity);

    if (Altitude < 0)
        Altitude = 0;
}
```

3.1.4. Scoring

Cycle Score (RLM Specific)

Although the Cycle Score is not included within the simulator, this is the best opportunity to discuss it as we discuss scoring. This is the feedback that the RLM receives for each decision it makes. In contrast, the simulator only gives feedback at the end of each game.

Cycle score helps make the RLM learning easier and will produce accurate results in a shorter time.

```
if (sim.Altitude <= 1000)
{
    if (sim.Fuel > 0 && sim.Velocity >= -5 && !thrust)
    {
        retVal = sim.Fuel;
    }
    else if (sim.Fuel > 0 && sim.Velocity < -5 && thrust)
    {
        retVal = sim.Fuel;
    }
}
else if (sim.Altitude > 1000 && !thrust)
{
    retVal = 200;
}
```

The code above is implemented in a way that rewards the RLM if he boosts at the right time and maintains the proper velocity.

- Reward is given per cycle if while velocity is greater than -5 and the RLM did not thrust for that cycle.
- Reward is given per cycle if while velocity is less than -5 and the RLM thrusts for that cycle.

This is not coded on the Simulator itself, but on the RLM Engine.

Session Score

As we mentioned above, the session score is how we feedback the player if he did good on the game or not. For the Lunar Lander Simulator, we will be using a weighted average to emphasize what metrics are more important.

Definitely, the Velocity takes precedence as if it's far off from the perfect velocity, the spacecraft can either crash or not land at all until it runs out of fuel.

Fuel is also important, as with more fuel left means that the player was efficient in using its resources, only firing the thrusters when needed.

And Finally, time, the faster the player can land the spacecraft, the better.

```
public int Score
{
    get { return (int)((Fuel * 10) + Seconds + (Velocity * 1000)); }
}
```

3.1.5. Source Code

```
public class LanderSimulator
{
    public const double Gravity = 1.62;
    public const double Thrust = 10;
    public const double TerminalVelocity = 40;

    public LanderSimulator()
    {
        Fuel = 200;
        Seconds = 0;
        Altitude = 10000;
        Velocity = 0;
    }

    public int Fuel { get; set; }
    public int Seconds { get; set; }
    public double Altitude { get; set; }
    public double Velocity { get; set; }

    public int Score
    {
        get { return (int)((Fuel * 10) + Seconds + (Velocity * 1000)); }
    }

    public void Turn(bool thrust)
    {
        Seconds++;

        Velocity -= Gravity;

        Altitude += Velocity;


        if (thrust && Fuel > 0)
        {
            Fuel--;
            Velocity += Thrust;
        }

        Velocity = Math.Max(-TerminalVelocity, Velocity);
        Velocity = Math.Min(TerminalVelocity, Velocity);

        if (Altitude < 0)
            Altitude = 0;
    }

    public bool Flying
    {
        get { return Altitude > 0; }
    }
}
```

Chapter 4: Solving a Problem

4.1. Solving the Lunar Lander

Now that we have identified our problem and made an environment to test on, we can now configure the RLM to start training. Since we already discussed the creation of the Rlm Network, we will apply that here as well as use our newly made Lander Simulator.

4.1.1. Start a Session

To start a session, all we need to do is call a method from the RlmNetwork instance.

```
var sessionId = network.SessionStart();
```

This code also returns the current session id, and we will be using this later to run a cycle.

4.1.2. Passing Inputs to the RLM

Before processing our inputs, we need to determine when our simulation would end so that we will know when to stop the process. For this case, we will know that our simulation has ended when the Lander is already on the ground, that's when the simulator value for Altitude is already 0.

Below is the code in our simulator.

```
public bool Flying
{
    get { return Altitude > 0; }
}
```

Now that we already know when our simulation ends, we can now wrap our code in a conditional loop that only stops when the Lunar Lander is no longer flying.

```
while (sim.Flying)
{

}
```

And within this code block, we will need to setup the current inputs from the simulator to be processed by the RLM.

```
var inputs = new List<RlmIOWithValue>();
var fuel = network.Inputs.First(a => a.Name == "fuel");
var altitude = network.Inputs.First(a => a.Name == "altitude");
var velocity = network.Inputs.First(a => a.Name == "velocity");

inputs.Add(new RlmIOWithValue(fuel, sim.Fuel.ToString()));
inputs.Add(new RlmIOWithValue(altitude, Math.Round(sim.Altitude, 2).ToString()));
inputs.Add(new RlmIOWithValue(velocity, Math.Round(sim.Velocity, 2).ToString()));
```

After setting up our current inputs, let us create an instance of RlmCycle to be able to run a cycle and feed those Inputs from the simulator to the RLM like below.

```
bool learn = true;
RlmCycle cycle = new RlmCycle();
RlmCyclecompleteArgs cycleOutcome = cycle.RunCycle(network, network.CurrentSessionID, inputs, learn);
IEnumerable<RlmIOWithValue> cycleOutputs = cycleOutcome.CycleOutput.Outputs;
string currentOutput = cycleOutputs.First(a => a.Name == "thrust").Value;

bool thrust = Convert.ToBoolean(currentOutput);
```

The code above shows how to instantiate the RlmCycle and how to call the method RunCycle, which processes the parameters needed to process the simulator inputs.

The first parameter, network, is the current RlmNetwork we are using.

The second parameter is the current sessions ID, which is returned by our SessionStart method. T

The third parameter are the list of inputs we have created.

Finally, the last parameter is a Boolean value to determine whether the current process will be to either Train or Predict.

4.1.3. Scoring the RLM Cycle Output

To score a cycle, we need to call the method ScoreCycle from the RlmNetwork instance.

```
double score = scoreTurn(sim, thrust);
network.ScoreCycle(cycleOutcome.CycleOutput.CycleID, score);
```

This code demonstrates how we score a cycle. We use the method ScoreCycle from the RlmNetwork instance which takes two parameters, the current Cycle ID and the Score coming from the simulator.

```
public double ScoreTurn(LanderSimulator sim, bool thrust)
{
    double retVal = 0;

    if (sim.Altitude <= 1000)
    {
        if (sim.Fuel > 0 && sim.Velocity >= -5 && !thrust)
        {
            retVal = sim.Fuel;
        }
        else if (sim.Fuel > 0 && sim.Velocity < -5 && thrust)
        {
            retVal = sim.Fuel;
        }
    }
    else if (sim.Altitude > 1000 && !thrust)
    {
        retVal = 200;
    }

    return retVal;
}
```

This is the same code we used to discuss the Cycle Score on the Lunar Lander Problem section. In this case, it accepts the LanderSimulator as a parameter so it can take the Input values from it and if the Lunar Lander did a thrust.

NOTE: Depending on the problem, there might be cases where you will not need to score a cycle. In this case, you will still need to call the ScoreCycle method as it is required. But instead of passing just random values, it is best to just pass a parameter value of 0.

4.1.4. Ending a Session

To end a session, you need to call a method SessionEnd from the RlmNetwork instance. This accepts the session score as a parameter and will process this if it is the best solution.

```
network.SessionEnd(sim.Score);
```

The code above shows how we end the session and pass in the Simulation Score.

4.1.5. Source Code

Here's the complete code in running the training, our LanderSimulator Class.

```
var sim = new LanderSimulator();
var sessionId = network.SessionStart();

while (sim.Flying)
{
    var inputs = new List<RlmIOWithValue>();
    var fuel = network.Inputs.First(a => a.Name == "fuel");
    var altitude = network.Inputs.First(a => a.Name == "altitude");
    var velocity = network.Inputs.First(a => a.Name == "velocity");

    inputs.Add(new RlmIOWithValue(fuel, sim.Fuel.ToString()));
    inputs.Add(new RlmIOWithValue(altitude, Math.Round(sim.Altitude, 2).ToString()));
    inputs.Add(new RlmIOWithValue(velocity, Math.Round(sim.Velocity, 2).ToString()));

    bool learn = true;
    RlmCycle cycle = new RlmCycle();
    RlmCyclecompleteArgs cycleOutcome = cycle.RunCycle(network, network.CurrentSessionID, inputs, learn);
    IEnumerable<RlmIOWithValue> cycleOutputs = cycleOutcome.CycleOutput.Outputs;
    string currentOutput = cycleOutputs.First(a => a.Name == "thrust").Value;

    bool thrust = Convert.ToBoolean(currentOutput);

    var score = scoreTurn(sim, thrust);
    network.ScoreCycle(cycleOutcome.CycleOutput.CycleID, score);

    sim.Turn(thrust);

}

network.SessionEnd(sim.Score);
```