Neon Developer Setup

Introduction

This document details the steps required to configure a machine capable of building the Neon solution.

# Requirements

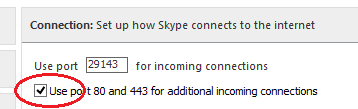
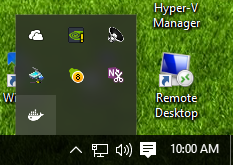
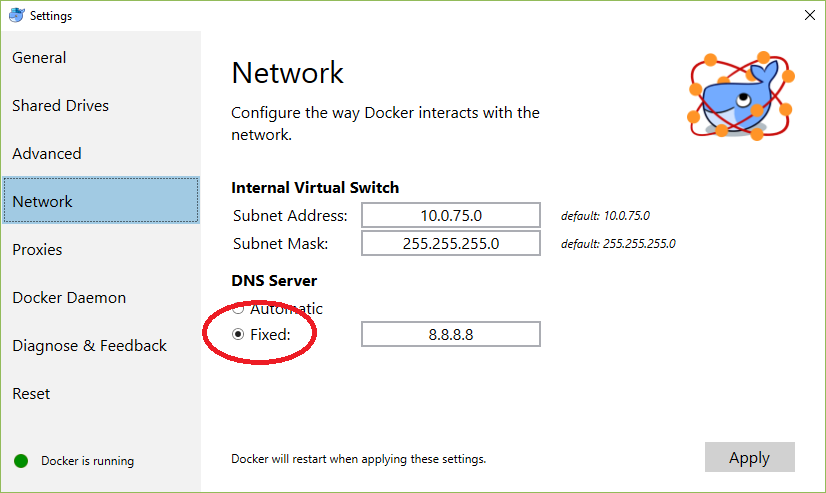
Neon projects requires a **64-bit** computer running **Windows 10** and **Visual Studio 2017 Community Edition** (full install).

Optional requirements for Xamarin iOS and Android development:

* Apple Developer Account
* An Apple Macintosh computer
* Android device(s)
* iOS device(s)

Windows Machine Configuration

Follow the steps below to configure a development or test machine:

1. **Important**: (this doesn’t apply to Win10 Skype preview?) Skype listens for inbound connections on ports 80 and 443 by default (which is really annoying). This will conflict with services running locally. To disable:  
   1. In Skype, select the **Tools/Options** menu.
   2. Select the **Advanced/Connection** tab on the left side.
   3. **Uncheck**: Use **port 80 and 443** for additional incoming connections:  
        
      
   4. **Restart Skype**.
2. Make sure that **Windows** is **fully updated**.
3. Install **Microsoft Visual Studio Community 2017 RC** from: [here](https://www.visualstudio.com/vs/visual-studio-2017-rc/)  
   1. Select **all workloads** on the first panel.
   2. Select **Individual components**
   3. Click to select **all components**
   4. Click **Install**
4. Create a **Visual Studio shortcut** and configure it to **run as administrator**. You’ll want to use this when you’re working with Neon solutions.  
     
   **NOTE: Neon builds will require that Visual Studio be started with administrator permissions.**
5. Install **Git for Windows** with defaults from: [here](https://git-scm.com/download/win)
6. Install **Docker for Windows** from [here](https://www.docker.com/products/docker#/windows).  
   1. Use the **Stable** channel unless you have a specific need for bleeding edge features.
   2. **Right-click** the Docker icon  in the system tray and select **Settings…**
   3. Select the **Shared Drives** tab and **share** the drive(s) with your project source code.
   4. You’ll need to enter your local machine **credentials**.
   5. Open a **DOS** command window.
   6. Run the following command: docker pull alpine
   7. If this **fails** with a **Network Timeout** you need to configure Docker via the following steps:
   8. Right-click the **Docker** icon in the Windows task bar and select **Settings…**  
        
      
   9. Click **Network** on the left, select the **Fixed DNS Server** and then **Apply**.  
        
      
7. **Clone** in the **Neon Research Git** repository:  
   1. Have one of the repository admins **grant access** to your Microsoft ID.

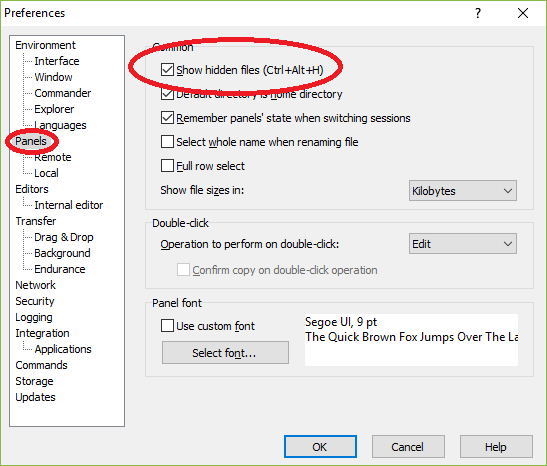
URL: **https://neonresearch.visualstudio.com/defaultcollection/\_git/NeonCluster**

PWD: use your **Microsoft ID** credentials

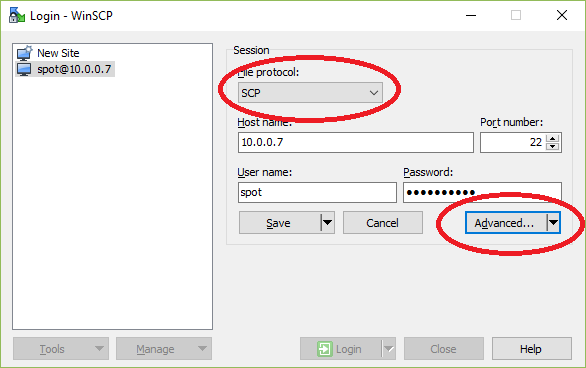
* 1. **Clone** the repository to a local folder. You should be able to use any folder on your machine but I (Lill) typically clone to **C:\src\NeonCluster.**
  2. Don’t try to build yet.
  3. **Close** all instances of **Visual Studio**.

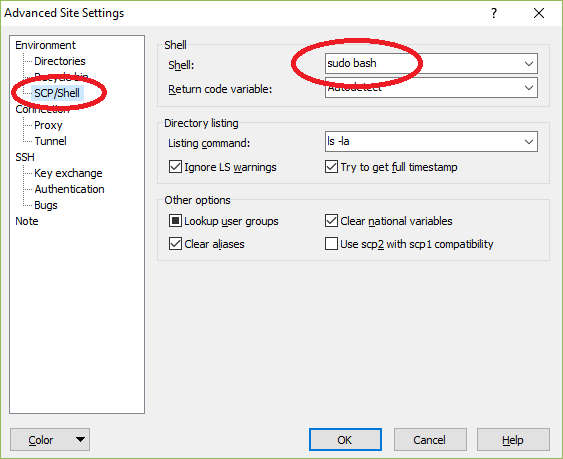
1. Configure your **environment variables**:

In your local root enlistment folder **$**, right-click on **buildenv.cmd** and **Run as administrator** to configure the build related environment.

1. Open the main solution **$/NeonCluster.sln** in Visual Studio. Note that VS2017-RC can take a very long time to load the solution the first time. Confirm that the solution **builds**.
2. Optional: Install **SQL Server Management Studio (SSMS)** from: [here](https://msdn.microsoft.com/en-us/library/mt238290.aspx)
3. Open the main solution **$\NeonCluster.sln** in Visual Studio. Confirm that the solution **builds**.
4. Some server components are deployed to Linux, so you’ll need terminal and file management programs. We’re currently standardizing on **PuTTY** for the terminal and **WinSCP** for file transfer. install both programs to their **default directories**.  
   1. Install both **WinSCP** and **PuTTY** from [here](http://winscp.net/eng/download.php). (PuTTY is near the bottom of the page)
   2. Optional: The default **PuTTY color scheme sucks** (dark blue on a black background doesn’t work for me). You can update the default scheme to **Zenburn Light** by **right-clicking** on the file below and selecting **Merge**:  
        
      $\External\zenburn-ligh-putty.reg
   3. WinSCP: Enable **hidden file display**. Start WinSCP and select **Tools/Preferences…** and then:  
        
      
   4. Optional: The **nc.exe scp** command launches WinSCP with SUDO access to cluster hosts. If for some reason you need to launch WinSCP directly or against a non-NeonCluster host, you’ll need to manually **Enable SUDO** access. By default, WinSCP will not be able to access protected directories and files, even though your login account as admin permissions. You can enable this manually when you create or edit connections:  
      1. NOTE: For the following to work, sudo on the Linux machine must be configured so that it doesn’t prompt for passwords. The standard NeonCluster VM templates and containers are configured this was by default. To accomplish this manually, you need to:  
           
         Edit the **/etc/sudoers** file to prevent sudo from requesting passwords (breaking scripts). Use the following command tom edit the file:  
           
         sudo visudo

and change the line starting with **%sudo** to:  
  
%sudo ALL=NOPASSWD: ALL

* + 1. Select **SCP** as the **file protocol** and then click the **Advanced…** button in the connection dialog:  
         
       
    2. Select **Environment/SCP** in the left panel and change the shell command to **sudo bash**.



1. Install **7-Zip (x64)** from [here](http://7-zip.org/download.html) to the default folder. This is used to encrypt/decrypt the SSL certificates required for local development and testing.
2. Install a local **Couchbase server** for development and unit testing:
   1. Go to this page: <http://www.couchbase.com/nosql-databases/downloads>
   2. Click **Community Edition** under the title.
   3. Download the **Current** **Windows (64-bit)** version.
   4. Run the install using the defaults.
   5. Open the admin UX at **http://localhost:8091** in a browser (if setup didn’t do this).
   6. Click **SETUP**: Change the **Data RAM Quota** to **256MB**.
   7. **NEXT**, **NEXT**, **NEXT** to the **NOTIFICATIONS** step. **Uncheck Enable Software Update…** and click **NEXT**.
   8. Enter **test000** in both the password boxes.
   9. Click **NEXT**. Your server will be ready in a few seconds.
   10. Click the **DATA BUCKETS** tab.
   11. Click the **blue triangle** to the right of the [default] bucket name.
   12. Click **EDIT** on the right side.
   13. Change per node **RAM** quota to **100MB** and **SAVE**.  
         
       **NOTE**: Admin credentials will be: **Administrator**/**test000**
3. Install a local **Couchbase Sync Gateway** for development and unit testing:
   1. Go to this page: <http://www.couchbase.com/nosql-databases/downloads>
   2. Click **COUCHBASE MOBILE** at the top.
   3. Click **Community Edition** under the title.
   4. Select **Current Release** from the drop-down
   5. Download the **Windows** version.
   6. Run the install using the defaults.
4. Manually start the **Couchbase Sync Gateway service**.
5. Optional: Install **Fiddler4** from [here](http://www.telerik.com/download/fiddler).
6. Optional: Install **Notepad++** from [here](https://notepad-plus-plus.org/download).
7. Optional: Install **S3 Browser** from [here](http://s3browser.com/download.aspx).