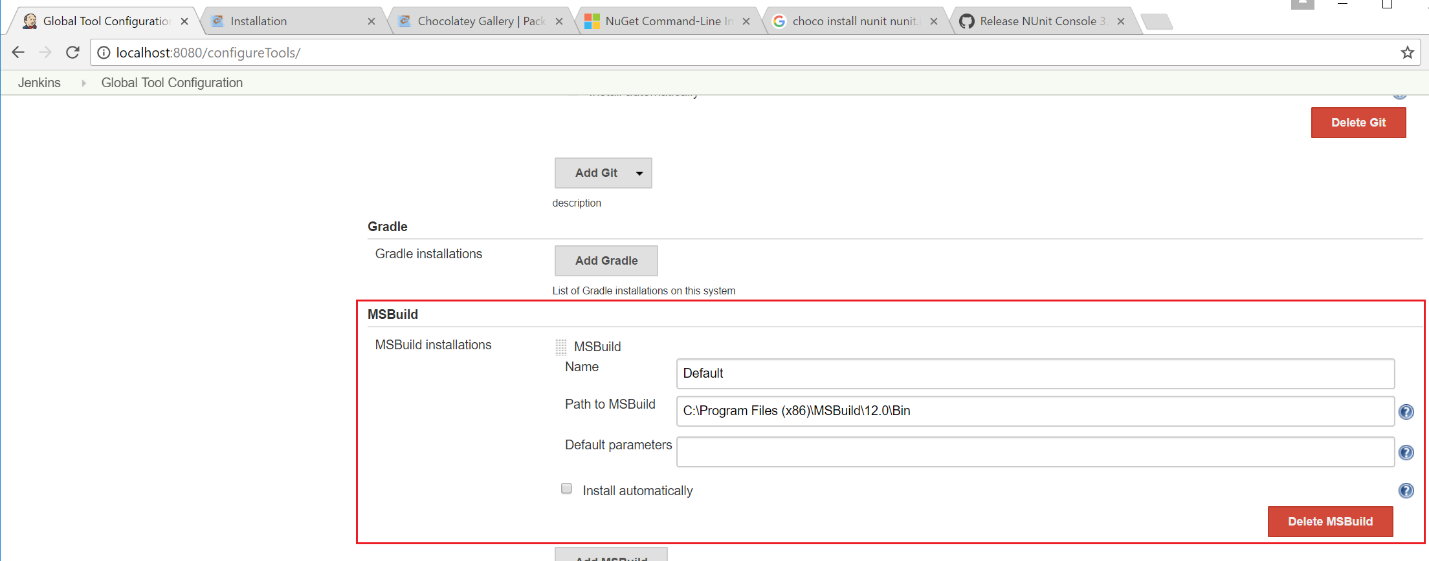
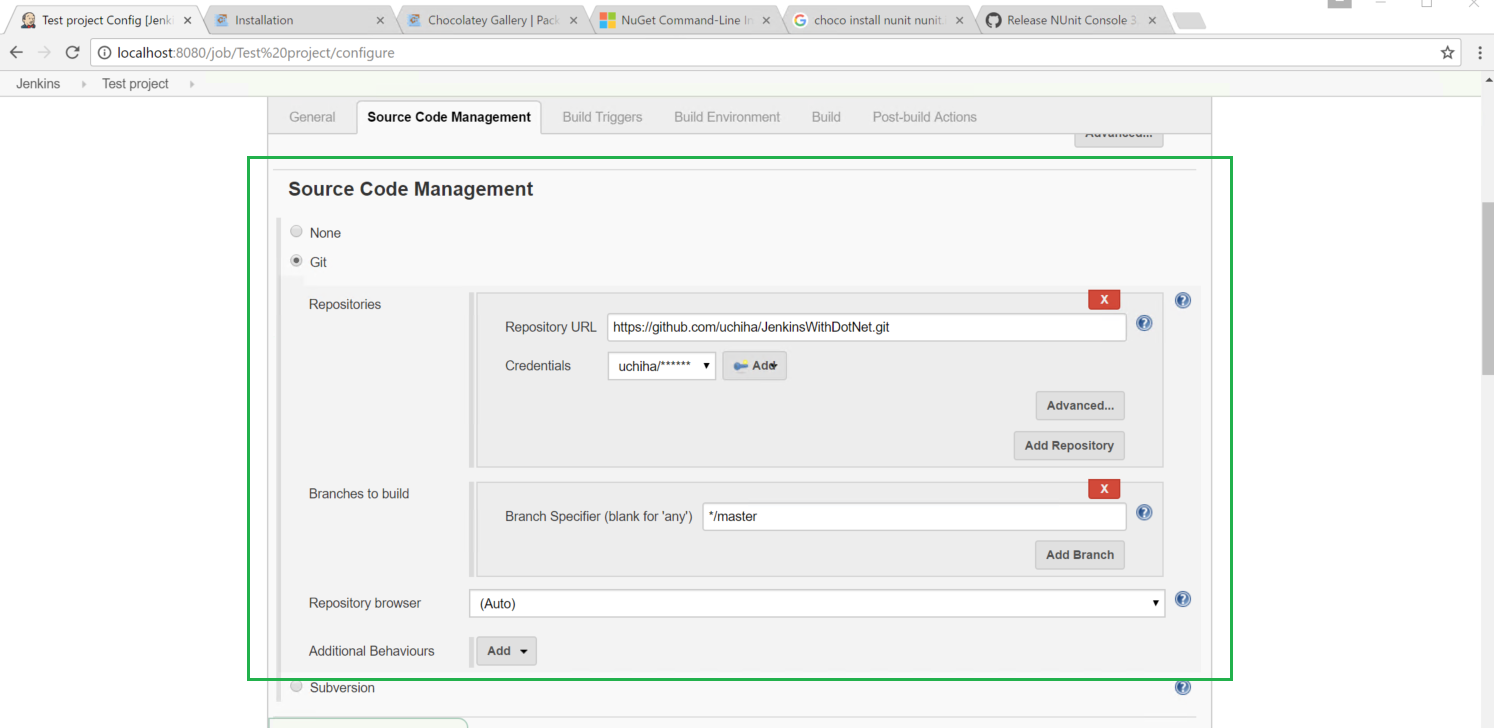
Using Jenkins for a .NET project:

1. In order to build the sourcecode, MSBuild must be installed. This can be done using chocolatey packages, but I just downloaded an installer online. Afterwards, Manage Jenkins -> Global Tool Configuration and add the MSBuild installation:

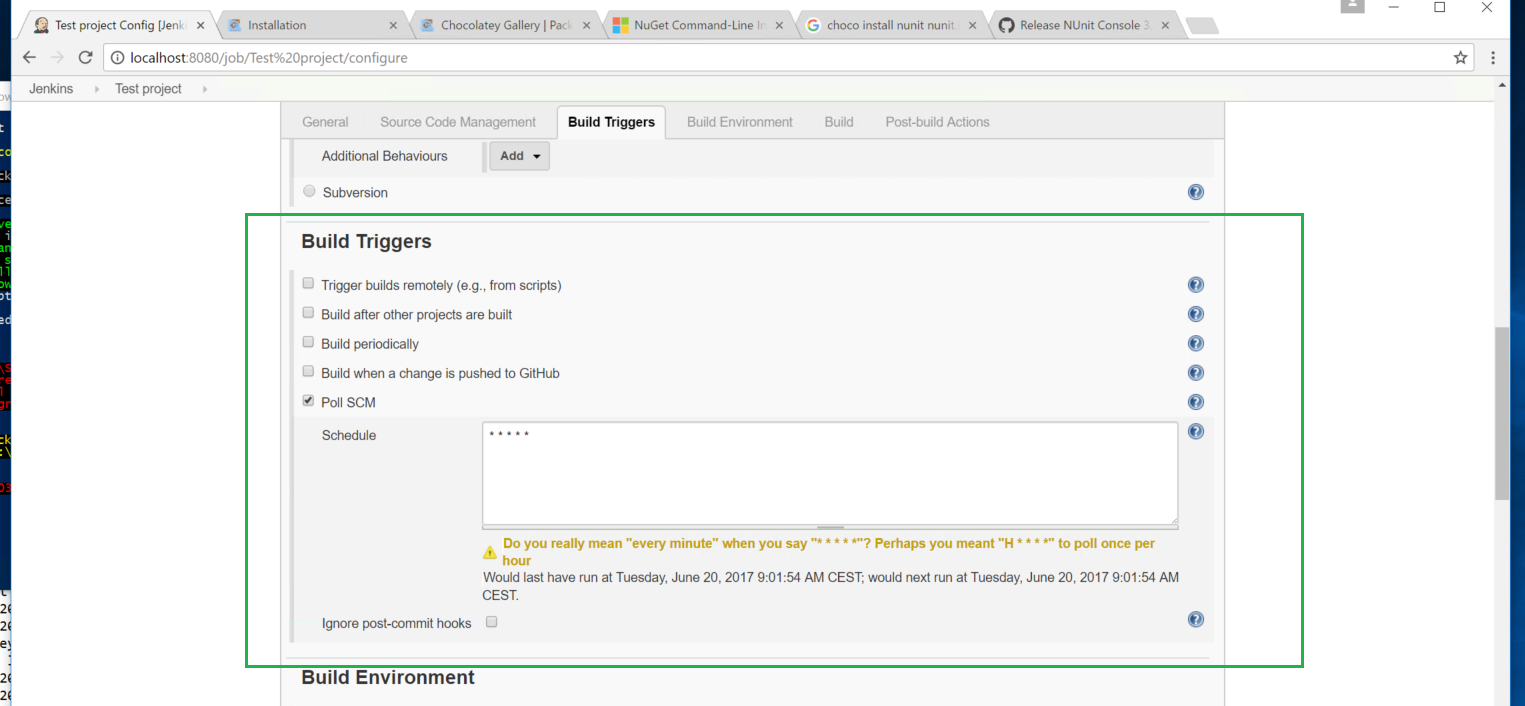


Hold your horses though, before Global Tools Configuration can have this, you need to install the MSBuild plugin from Jenkins plugin…

2. Create a new Job, Jenkins -> New Item and make this a freestyle project. That’s is, oh yes, this is my Source Code Management:



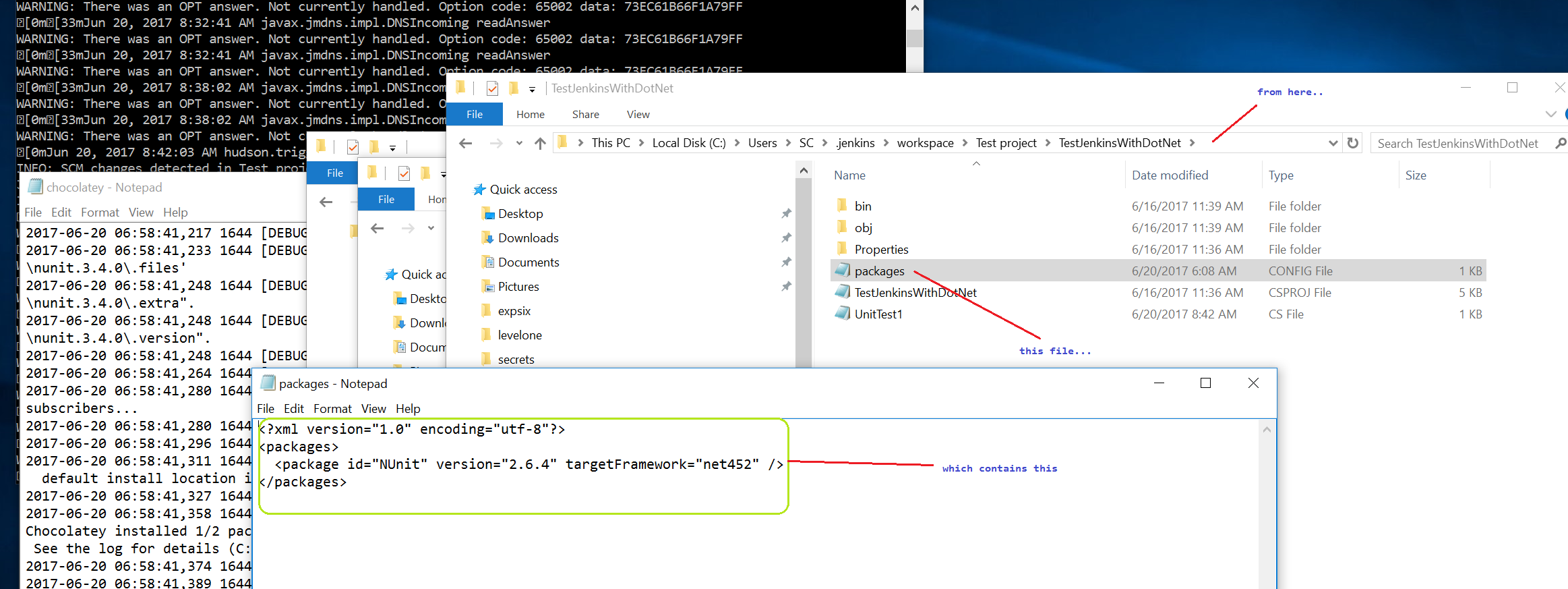
A build trigger: can research more on this, but \* \* \* \* \* basically means poll every minute:



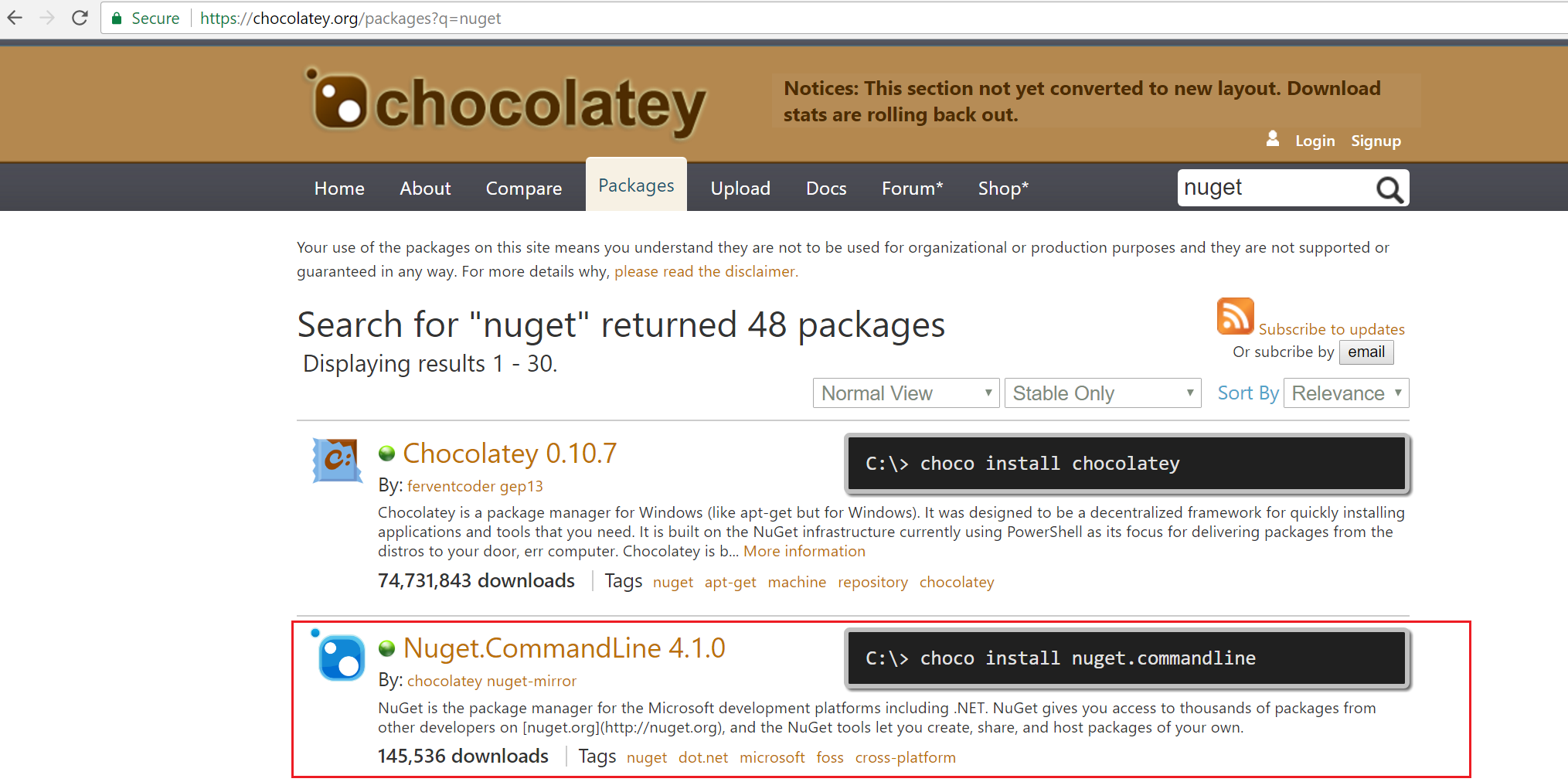
3. And now the build Section. In order to make this work, first, we restore the packages needed by the unit test project (which is nunit). So in order to do that, I’ve needed to install the package manager tool for .NET which is chocolatey:



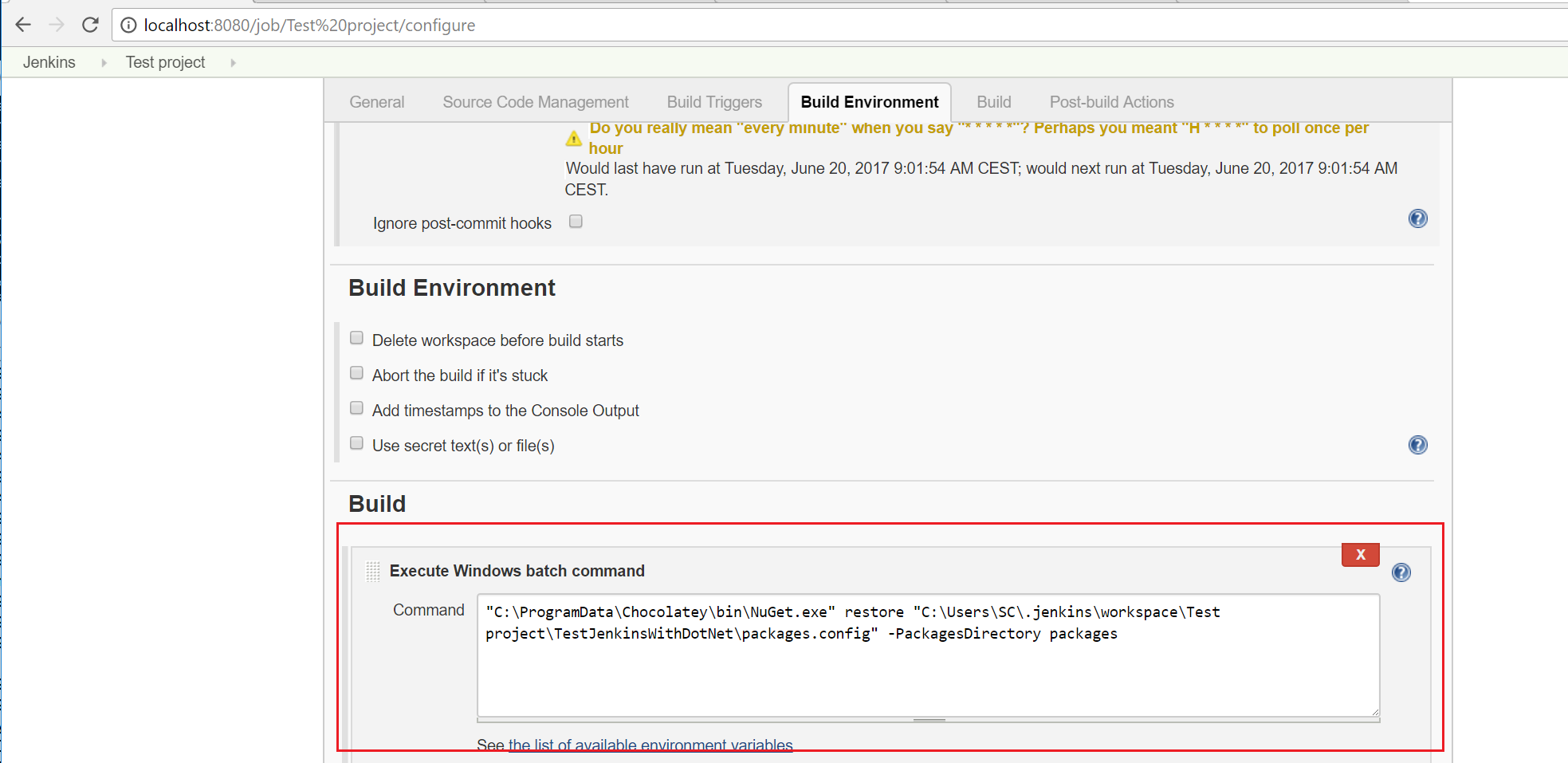
Afterwards, in order to restore nuget dependencies:



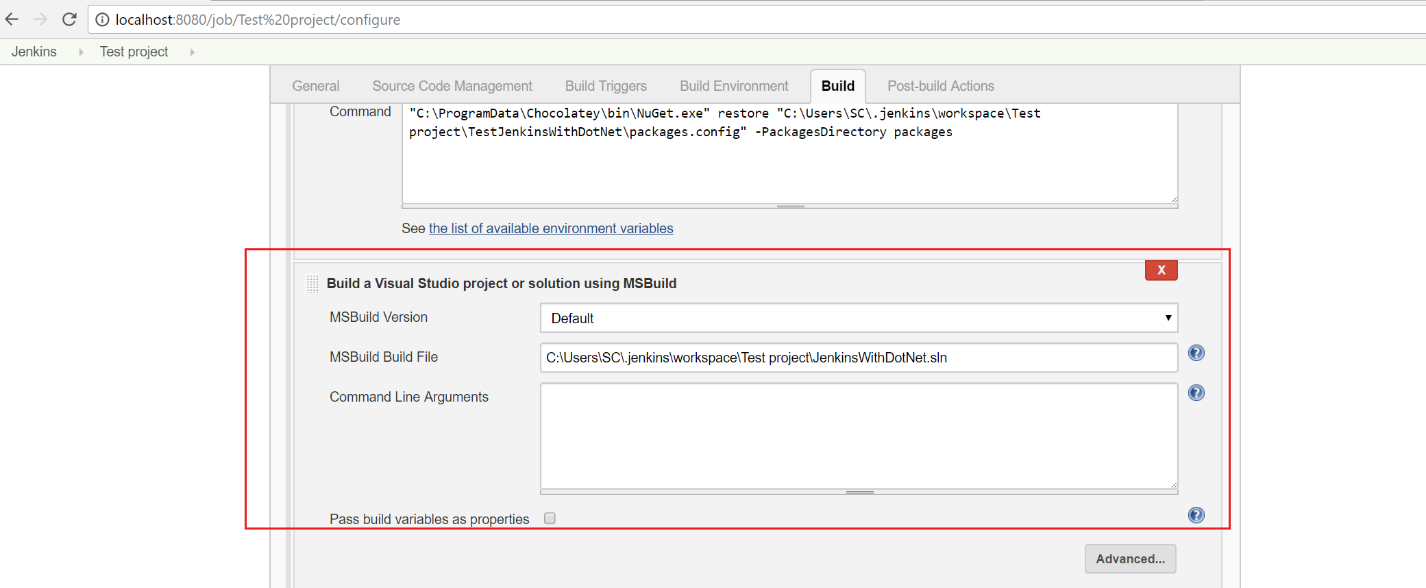
We need to install nuget via chocolatey:



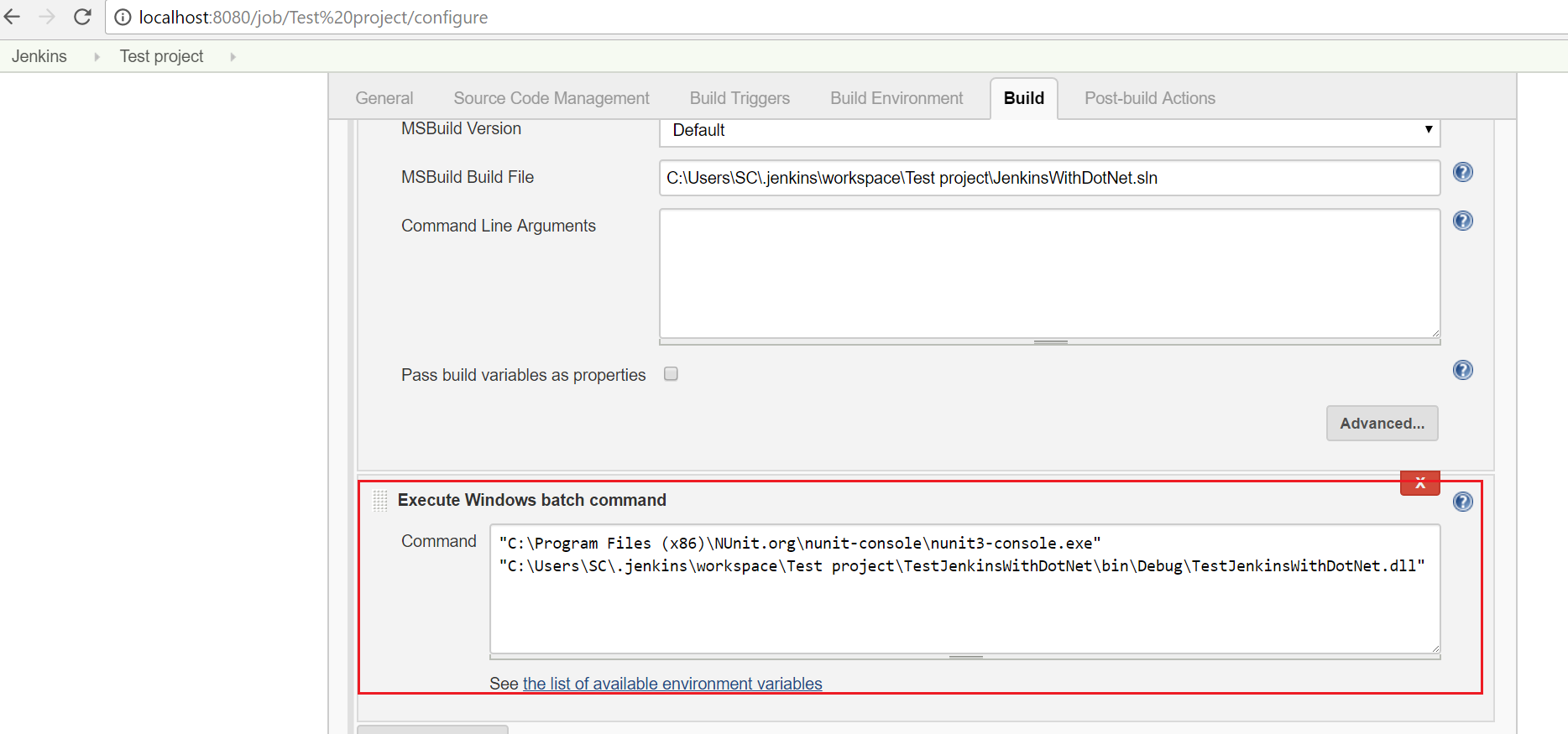
And this will enable the first Execute windows batch command in this build:



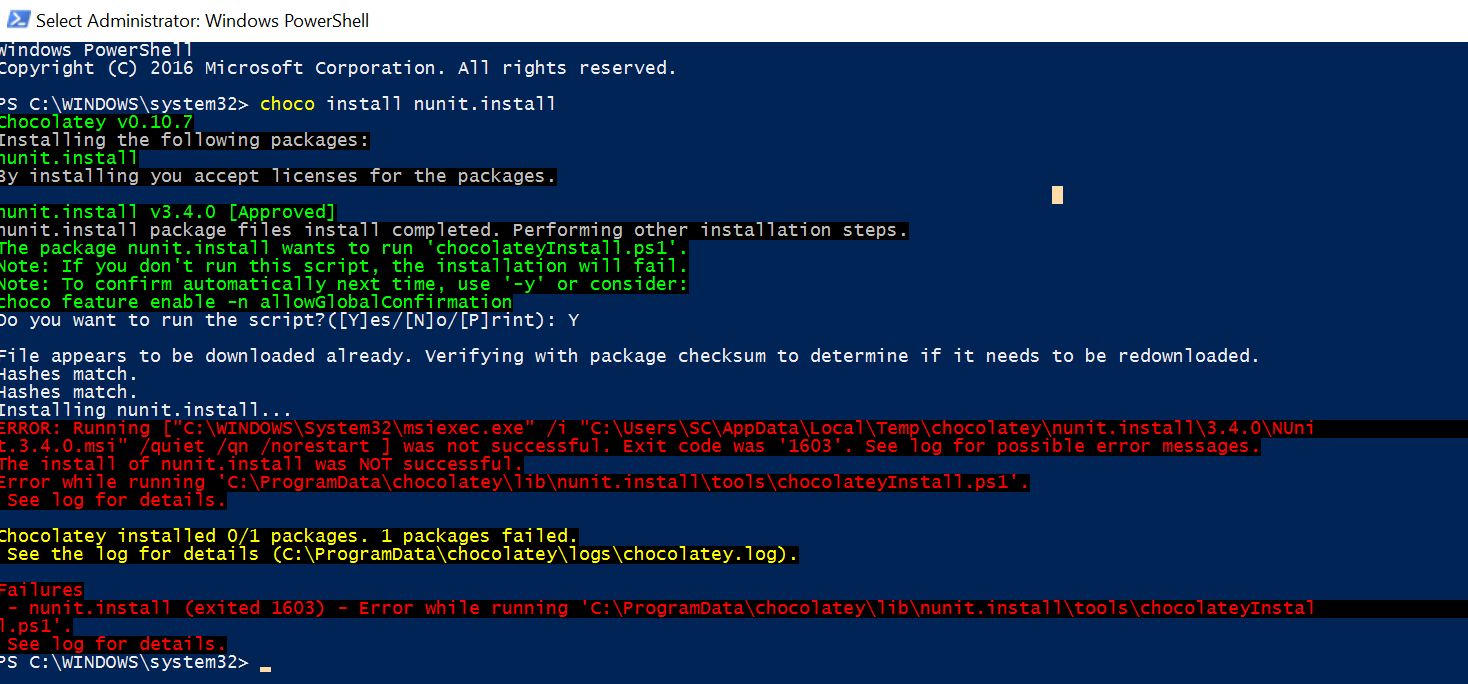
The second is only pointing to the solution’s .sln file and using our MSBuild installation to build it:



Lastly, when the test project is built, which should be because of the first windows batch command, we need to run the test dll against the nunit3 console:



The nunit-console I just installed using downloaded installer, because using nuget with it, it fails for me:



This is when I finally had a good build:

