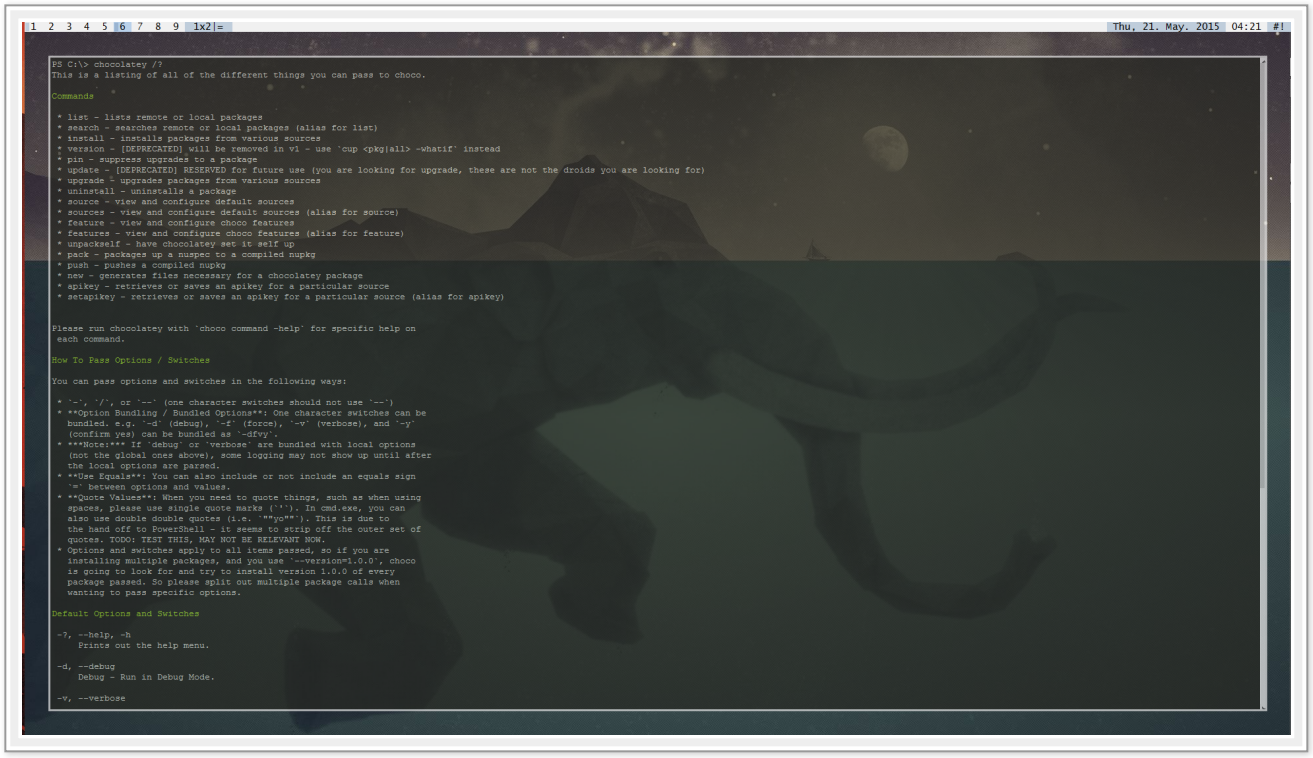
typicaltim

Creating A Local Chocolatey Repository

May 21, 2015 *by* typicaltim



Introduction

As much as I love Chocolatey, the documentation on the Github page is a bit lack-luster. I spent quite some time reading forum posts after I read the official page on local repositories – which was really vague – before I was able to set one up myself. To try to save other people some time, here is my quickstarter guide on the subject.

First off we are going to need a place to store all the packages we are going to create. I recommend that you put this on your local machine for now so that you can understand

how this works. You can always just move the entire hierarchy over to a network share if you want to put it somewhere else, it's that easy. I'm going to put this on my C:\ temporarily for simplicity's sake.

```
mkdir c:\chocolateyrepo\
```

Next up we are going to create the basic files for a chocolatey package. These are not super special in any way, and you can change them or download alternative templates from other users if you want. But I'm going to keep things as basic as possible for the purposes of this guide.

```
cd c:\chocolateyrepo\
chocolatey new -name mypackage
```

After these commands you should have a directory called "mypackage" in "chocolateyrepo" that contains four items.

The nuspec file is a simple XML document that will describe your package, and the two powershell scripts under the "tools" folder run the installation and uninstallation when the user calls them using Chocolatey. We will be editing the nuspec file and the install script to create our package. But first, we need something to package! I recommend something simple, like Google Chrome. The edits I will be making are specific to MSI files, so avoid EXE's and MSU's if you want to follow along.

Once you have your MSI, place it in your "mypackage" directory under "tools". Then open up the mypackage.nuspec file in your favorite text editor and replace the text between the tags to reflect the package you're making. The most important tag for our testing purposes is going to be . You're going to want to remember the name you put here, you'll be using it later. It is best practices to fill everything out in a production environment, but this will get it working for testing. Name the package with like so:

```
<id>googlechrome</id>
```

That's literally as difficult as this gets, so hang in there – we have two more files to edit. Open up the install script and edit the following section.

```
$ErrorActionPreference = 'Stop';
$packageName = 'googlechrome'
```

```
$installerType = 'MSI'
$url = 'C:\chocolateyrepo\mypackage\googlechromeenterprise.msi'
$silentArgs = '/quiet'
validExitCodes = @(0)
$toolsDir = "$(Split-Path -parent $MyInvocation.MyCommand.Definition)"
```

I have removed a lot of the comments as well as the 64-bit url in my case, if you are using a 64-bit installer you need to use that line instead of $url. Finally, edit the uninstall script:

```
$ErrorActionPreference = 'Stop';
$packageName = 'googlechrome'
$installerType = 'MSI'
$silentArgs = '/quiet'
$validExitCodes = @(0)
```

Okay, the hurdle is over. It is smooth sailing from here. Now that our package files are all set up, it's time to package it up!

```
cd c:\chocolateyrepo\mypackage\
cpack
```

You should now have a nupkg file created in your "mypackage" directory. This is what chocolatey will be looking for when you tell it to install a package. To make sure you are grabbing the package you made we are going to remove to global repository temporarily. You can add it back later by replacing "remove" with "add":

```
choco source remove -n chocolatey -s https://chocolatey.org/api/v2/
```

Now we can add the repository we made at the beginning of this guide.

```
choco source add -n myrepo -s c:\chocolateyrepo\
```

Now that your repo is added, you install the package you made!

```
choco install googlechrome
```

I hope this guide was a little more clear at introducing how the basic setup of a Chocolatey local repository is done. I know I got frustrated with it a bit when I first picked it up. I highly recommend that you take a look at the Chocolatey documentation on the Github page regardless of its cohesiveness. It goes into a tad more detail than I went into here. It is best practice to edit the install and uninstall script in full detail to ensure proper install and uninstall when using the Chocolatey command. Otherwise the user will have to manually remove the program – or worse.

Change the Windows 10 Active/Inactive Window Color →
