Show how to traditionally add DLL reference to .NET project

* Right-click project and Add References
* Select DLL
* Needs to be done for every project that needs to reference the DLL
* The DLL needs to be checked into source control somewhere
  + Builds need to know where to check the file out from so that it can be referenced
  + Devs’ workspaces need to be updated to check the file out
  + Both need to be updated when the NuGet package is updated which means every dev workpace and build workspace needs to be updated
* Hard to remember to update periodically

Show NuGet Package Manager

* Right-click solution and "Manage NuGet Packages"
* This is where you can:
  + search for NuGet packages to install
  + look at currently installed packages and add them to additional projects or uninstall them
  + easily find and install updates
* Search for YouTube APIs
  + Show dependencies
* You point to different package sources
  + We usually only need to worry about nuget.org and nuget.techsmith.com
  + You can point to a local folder if you’re developing NuGet packages locally
  + Each dev can point their VS to different package sources, but the NuGet.config file can define all package sources that a solution needs.
    - This is good for the build server since you don’t have to set the VS instance’s package sources on every build server

Show how to add .NET NuGet package

* Right-click solution and “Manage NuGet Packages”
* Find the package (ALanguageDetection) and install it to the project
* The solution is now in control of what packages it references
  + No build workspaces or dev workspaces need to be updated
  + If someone does update a NuGet package, it doesn’t require other devs to do anything and builds should not need to be updated
* Show what the inside of the package looks like

Show overview of how to traditionally add reference to native project

* Right-click, then properties
  + C/C++ > General > Additional Include Directories
  + Linker > General > Additional Library Directories
  + Linker > Input > Additional Dependencies
* Like mentioned earlier for .NET projects, this needs to be done for each project and each configuration (Release / Debug) individually
* The DLLs, LIBs also need to be in source control somewhere
  + Like mentioned earlier for .NET projects, every devs’ machine and every build definition needs to be updated to check out these files
* If someone updates the library, this typically means that each project and configuration needs to be updated again as well as all devs’ workspaces and all builds

Show how to add NuGet package to native projects

* The nice thing here is that the steps are the same as they were for .NET NuGet packages
* Right-click solution and “Manage NuGet Packages”
* Find the package (TSCUtl) and install it to the project
* Show that .h file is coming from packages folder
* The solution is now in control of what packages it references
  + No build workspaces or dev workspaces need to be updated
  + If someone does update a NuGet package, it doesn’t require other devs to do anything and builds should not need to be updated

Create .NET NuGet Package

* Start with your code
* Run nuget.exe pack \*.csproj
* Use NuGet package
* Update NuGet package and show that it works
* You can use a file to define metadata for the package
  + Show nuget.exe spec \*.csproj command
    - The default nuspec file is the same as not providing one
    - Describe properties, they come from AssemblyInfo.cs
      * You can hard-code values in or let the project fill them in automatically
* Show what the inside of the package looks like

Create native NuGet Package

* Start with your code
* Show .autopkg file
  + This file is required unlike .nuspec
  + Nuspec section
  + Include section
  + Pivots
  + Additional files
* Use CoApp Write-NuGetPackage to create package
* Use NuGet package
* Update NuGet package and show that it works
* Show what the inside of the package looks like

Other

* Always manage packages from the solution level instead of project level
  + This helps prevent you from using different versions of the same NuGet package if it is not required