Project Layout

There are five non-test assemblies

* RoslynDomUtillities
  + *Should perhaps be renamed or combined with Common*
  + Contains general utilities
  + No dependencies on the .NET Compiler Platform or anything specific to RoslynDom
  + Could be reused in any project, but aren’t that interesting
* RoslynDomCommon
  + Contains interfaces, enums and common code
  + No dependencies on the .NET Compiler Platform
  + Could be reused to create a non-RoslynDom code implementation, but in that regard is probably most interesting as talking points
* RoslynDomRDomCommon
  + Contains Corporation and Provider implementations – expected to be the only ones needed
  + Contains interface for factories and workers
  + Contains utility methods that need this dependency set
  + Dependency on language agnostic parts of .NET Compiler Platform
  + This is the ONLY assembly with a reference to Unity (part of why it’s separate)
* RoslynDom
  + Contains the classes of RoslynDom and language independent
  + Contains a few utility methods for reuse within RoslynDom classes
  + Primary code for external access (except load/output)
  + Dependency on language agnostic parts of .NET Compiler Platform
* RoslynDomCSharpFactories
  + Contains worker classes to load and unload RoslynDom
  + Contains the RDomCSharp.Factory instance for loading and unloading RoslynDom
  + Dependency on everything else
  + This is the ONLY assembly with a .NET Compiler Platform CSharp dependency
* RoslynDomVisualBasicFactories
  + Will parallel the C# factories to allow loading in one language and output in another