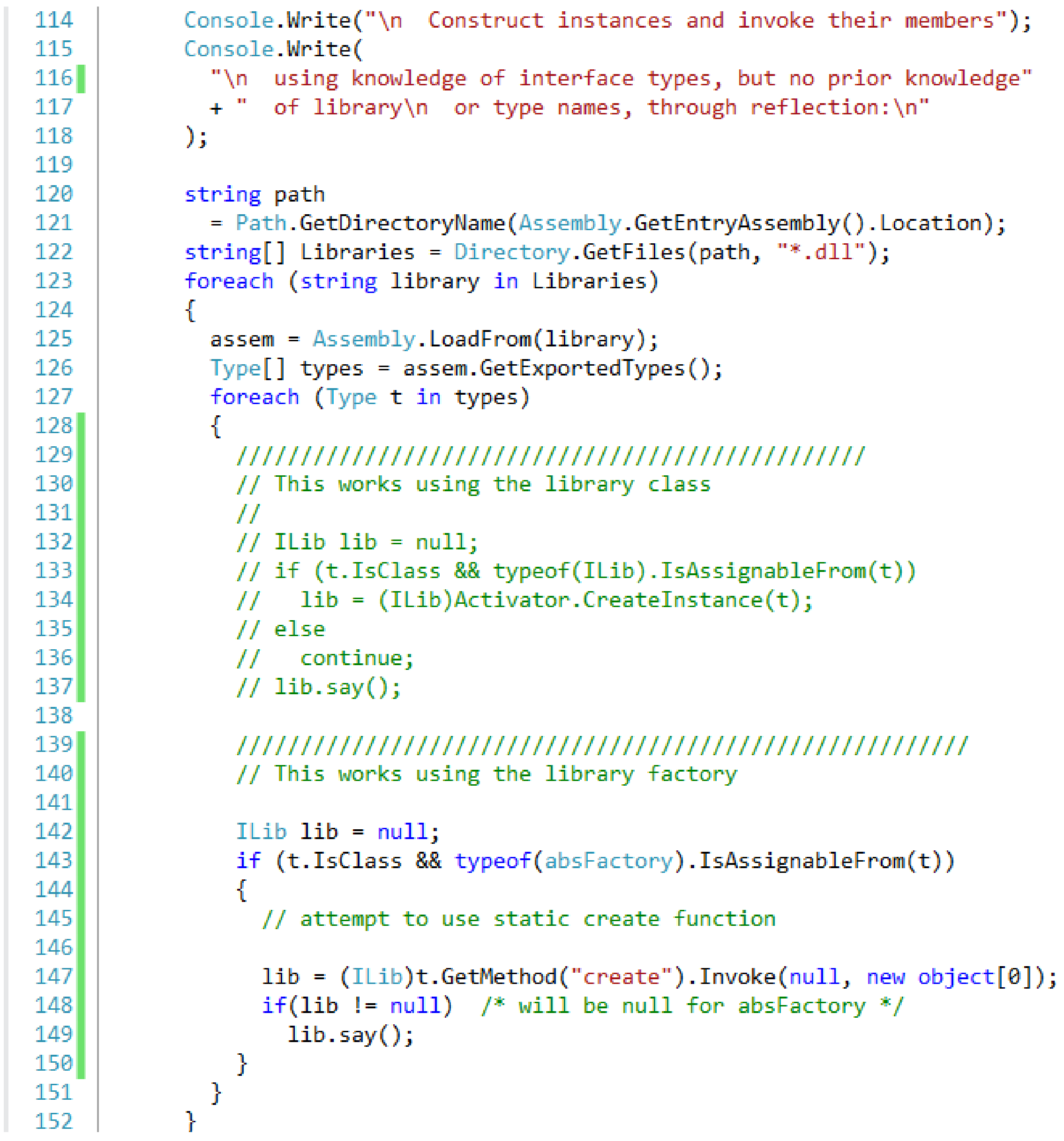
Handouts\CoreTechnologies\CSharp\code\DynamicLinkLibraries

Revision of the DemoLoadingDlls.cs handed out in class Monday 9/16/2013

Here, we show how to use the abstract factory provided by each library without knowing the concrete names of the factory classes. We do this with reflection.



```
Console.Write("\n  Construct instances and invoke their members");
Console.Write(
  "\n  using knowledge of interface types, but no prior knowledge"
  + "  of library\n  or type names, through reflection:\n"
);

string path
  = Path.GetDirectoryName(Assembly.GetEntryAssembly().Location);
string[] Libraries = Directory.GetFiles(path, "*.dll");
foreach (string library in Libraries)
{
  assem = Assembly.LoadFrom(library);
  Type[] types = assem.GetExportedTypes();
  foreach (Type t in types)
  {
    ////////////////////////////////////////////////
    // This works using the library class
    //
    // ILib lib = null;
    // if (t.IsClass && typeof(ILib).IsAssignableFrom(t))
    //   lib = (ILib)Activator.CreateInstance(t);
    // else
    //   continue;
    // lib.say();

    ////////////////////////////////////////////////////////
    // This works using the library factory

    ILib lib = null;
    if (t.IsClass && typeof(absFactory).IsAssignableFrom(t))
    {
      // attempt to use static create function

      lib = (ILib)t.GetMethod("create").Invoke(null, new object[0]);
      if(lib != null)  /* will be null for absFactory */
        lib.say();
    }
  }
}
```