Deployment Manual

The source code for the game can be found in the following repositories:
`https://uclix.visualstudio.com/Fizzyo/_git/MinigolfMadness`
`https://github.com/MJPCrepin/Fizzyo-MinigolfMadness`
Both repositories are maintained and the same updates are pushed to both; the former is owned by the project client while the latter is publicly accessible. Pull requests are more likely to be considered on the latter.

To deploy the game it will first need to be built in Unity (5.6.1 or higher) and then packaged using Visual Studio (2017 or higher).

Once opened in Unity, ensure $ProBuilder$ has been deleted (found in $\_Imports$) and all objects in any Scene are not marked as $Static$ before attempting a build. The Build Settings need to include all finalised Scenes in the correct order ($HomePage$ as Scene 0), see System Manual for further details. The Windows Store module needs to be downloaded and installed (UWP SDK 10.0.15063.0 or higher). The SDK used was $Universal10$ with Target Device being PC. UWP Build type used was $D3D$ to run on the Local machine.

When the build is completed, the game should be opened using the generated .sln file in Visual Studio. From here, the game can be directly run on the Local Machine (x86 or x64) which will install the game. To export to another Windows machine without Visual Studio, it will need to be packaged as a Windows Store application first.

For integration with the Fizzyo Hub, $App.cs$ and $Package.appmanifest$ need to be edited as described below.
The $Declarations$ tab in $Package.appmanifest$ needs to have a $Protocol$ added which defines a name for the UWP application, in this case the Fizzyo Hub calls for $minigolfmadness$. Other settings can be left as default.
Then, the following $App.cs$ method needs to be edited as follows in order to allow the defined protocol name to be called and launched by another UWP application successfully.

```
private void ApplicationView_Activated
(CoreApplicationView sender, IActivatedEventArgs args) {
    CoreWindow.GetForCurrentThread().Activate();
        if (args.Kind == ActivationKind.Protocol)
            CoreWindow.GetForCurrentThread().Activate(); }
```

To package the opened project as a Windows Store application, navigate to and select the following menu item:

`Project > Store > Create App Packages...`

After refusing to upload to the Windows Store, the app can then be packaged locally, and the resulting build can be freely distributed to other machines. To install, the generated .appxbundle file can be run and the game will be installed automatically with the correct certificates on the machine. Alternatively, the powershell (.ps1) file can be run as an Administrator at the same effect but with debug information.