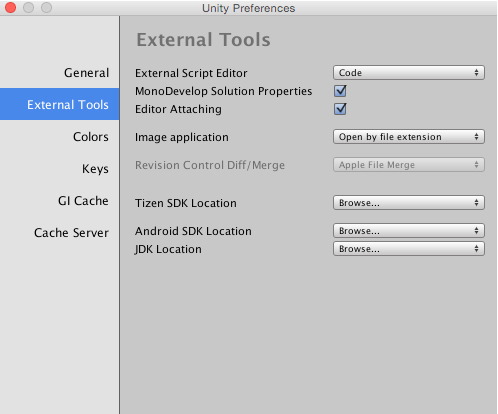
# Integration with Unity3D

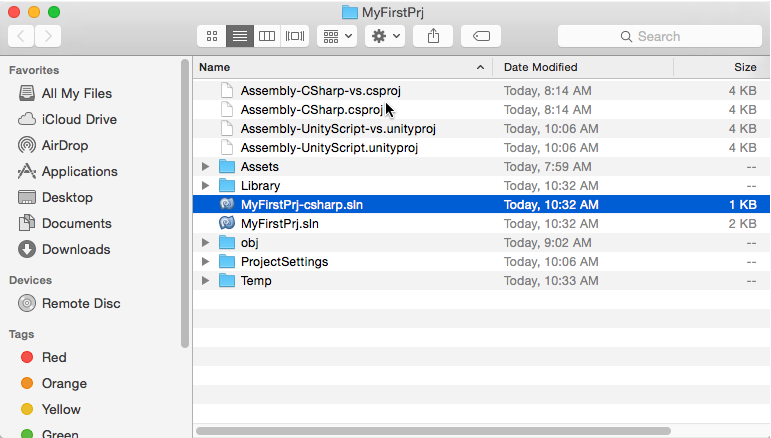
In this topic we will discuss how to integrate Unity3D and Visual Studio Code. Today, Unity3D is a very popular editor for game developers but it uses MonoDevelop to create code in C# or JavaScript. Of course, we are going to change it. Today I am going to talk about C# projects only. In order to start you need Unity3D editor installed on your Mac and you can open an existing project or create a new one. Just make sure that you have at least one C# file in the project.

On the first step you need to change MonoDevelop to Visual Studio Code. It’s easy to do if you open **Unity->Preferences** menu item and navigate to **External Tools** tab. You can find that **External Script Editor** box contains MonoDevelop but you can simply click **Browse** and navigate to **Applications** folder to select Visual Studio Code:



Right now you can click any C# file and Visual Studio Code will open it. But in case of single files Code works as a simple editor. Of course, it highlights syntaxes but IntelliSense system doesn’t work properly. So, we need to open the project folder and give Code information about the project.

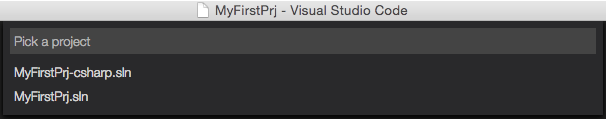
Before opening the folder you need to make sure that Unity project has already generated solutions and projects files. If you have created a new project you need to save the scene and use **Assets->Sync MonoDevelop Project** menu item in order to force the process. Once it’s done you can see at least two solution files in the project folder:



It’s time to open the folder in Visual Studio Code. But you still need to provide information about the project. Previously I mentioned that Code can work with **project.json** and **tsconfig.json** files but Code can understand **sln** and **csproj** files as well. Of course, it can not be initialized automatically because we have two solution files. So we need to select the right solution file, and you can do it by using the status bar:



After you click the ‘pick a project’ button, Visual Studio Code will provide a fly out that helps to select the right solution:



We need to select the solution with –csharp suffix there.

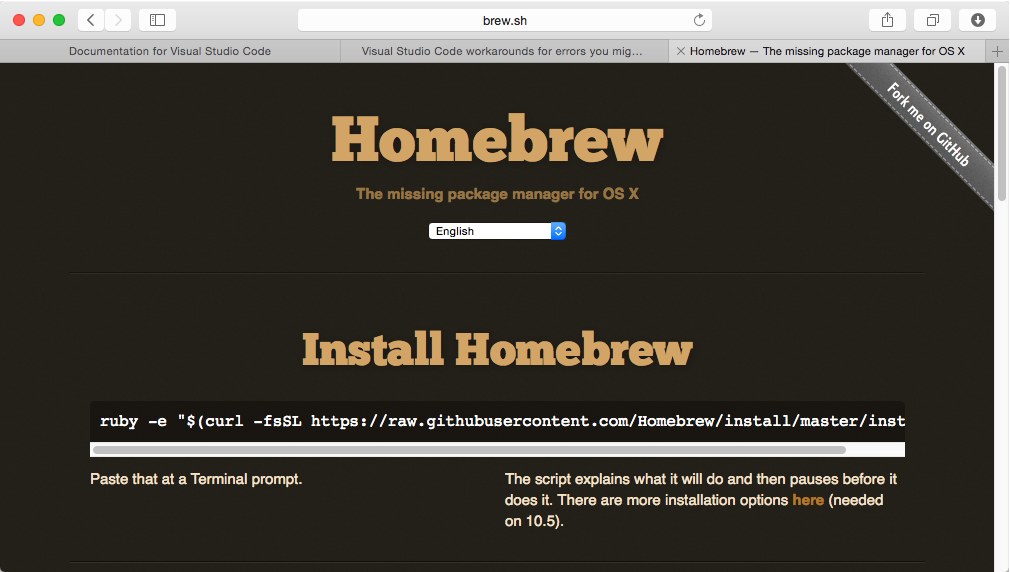
But it’s not enough to start coding. Let’s open any of C# files and you will see the following error:



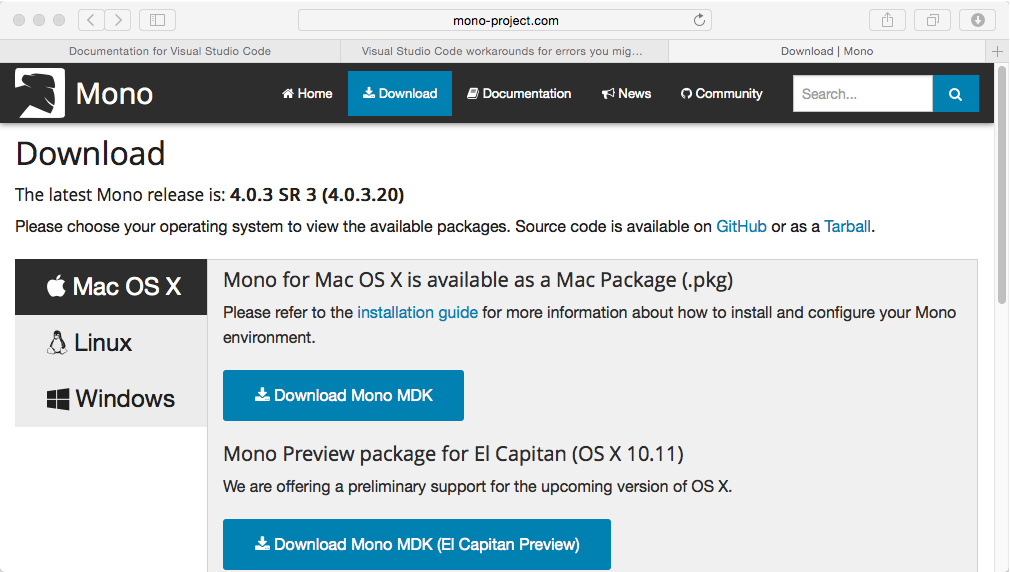
Visual Studio Code uses open source OmniSharp compiler but it requires the latest version of mono runtime. You can copy provided link and navigate to Microsoft page to see more details about the problem and check the solution:



Microsoft suggests using the brew tool to install missing mono runtime. If you don’t have the tool you can open brew.sh site and install it:



The second approach to install mono runtime is to visit mono project site directly. You can open mono-project.com and install mono runtime from there:



Once you install mono runtime you need to restart Visual Studio Code and open the C# file again. You will see that the compiler and IntelliSense system works fine:

