**Exercise**

**Reference the package from the application**

In this part, you'll get the new Tailspin.SpaceGame.Web code that has the model classes removed. Instead of referencing the models directly, the code will reference them from the package you created in the previous unit.

Here is a list of the steps:

* Get the new code from a branch of the original **Tailspin.SpaceGame.Web** repository.
* Reference the new Models package, version 1.0.0.
* Change the build pipeline to look for this package in your Azure Artifacts feed.
* Watch the pipeline successfully build the application.

**Fetch the branch from GitHub**

Fetch the models-package branch from GitHub and check out, or switch to, that branch.

This branch contains the *Space Game* project you worked with in the previous modules, but the **Models** directory has been removed.

1. From a new instance of Visual Studio Code, open the integrated terminal.
2. Go to the **mslearn-tailspin-spacegame-web** directory. Here's an example that uses the home directory, ~.

Bash

cd ~/mslearn-tailspin-spacegame-web

1. From the **File** menu, select **Open**.
2. Go to the root directory of the *Space Game* web project.

(If you need a refresher, run the pwd command in the terminal window to see the full path.)

You see the directory and file tree in the file explorer.

1. From the terminal, run the following git commands to fetch a branch named models-package from the Microsoft repository. Then switch to that branch.

BashCopy

git fetch upstream models-package

git checkout -b models-package upstream/models-package

The format of these commands enables you to get starter code from the Microsoft GitHub repository, known as upstream. Shortly, you'll push this branch up to your GitHub repository, known as origin.

1. As an optional step, verify that the **Models** directory no longer exists in the file explorer.

Instead, you see **Controllers**, **Views**, and other directories.

**Reference the Models package**

1. Open the **Tailspin.SpaceGame.Web.csproj** file, add the following PackageReference, and save the file.

XMLCopy

<ItemGroup>

<PackageReference Include="Newtonsoft.Json" Version="12.0.1" />

<PackageReference Include="Tailspin.SpaceGame.Web.Models" Version="1.0.0" />

</ItemGroup>

1. Modify the version number to include the pre-release prefix that was generated during the build process. Here's an example.

XMLCopy

<PackageReference Include="Tailspin.SpaceGame.Web.Models" Version="1.0.0-CI-20200610-165738" />

This references the **Tailspin.SpaceGame.Web.Models** package that you created in Azure Artifacts. Notice the version number, 1.0.0, plus the pre-release suffix. This matches the initial version that you published to Azure Artifacts in the previous part.

**Note**

When you save the file, Visual Studio Code might ask you to restore dependencies. We're not going to run this locally, so there is no need to restore the dependencies.

**Add the pipeline configuration**

The models-package branch doesn't contain an initial *azure-pipelines.yml* file. Here's your chance to create it.

1. From Visual Studio Code, select **File** > **New File**. Then select **File** > **Save** to save the blank file as *azure-pipelines.yml* in your project's root directory, such as *~/mslearn-tailspin-spacegame-web*.

**Important**

On Windows, ensure that you select **YAML** from the **Save as type** field.

1. Copy the following YAML code into *azure-pipelines.yml*:

ymlCopy

trigger:

- '\*'

pool:

vmImage: 'ubuntu-18.04'

demands:

- npm

variables:

buildConfiguration: 'Release'

wwwrootDir: 'Tailspin.SpaceGame.Web/wwwroot'

dotnetSdkVersion: '3.1.300'

steps:

- task: UseDotNet@2

displayName: 'Use .NET Core SDK $(dotnetSdkVersion)'

inputs:

version: '$(dotnetSdkVersion)'

- task: NuGetToolInstaller@0

inputs:

versionSpec: '5.6.0'

- task: Npm@1

displayName: 'Run npm install'

inputs:

verbose: false

- script: './node\_modules/.bin/node-sass $(wwwrootDir) --output $(wwwrootDir)'

displayName: 'Compile Sass assets'

- task: gulp@1

displayName: 'Run gulp tasks'

- script: 'echo "$(Build.DefinitionName), $(Build.BuildId), $(Build.BuildNumber)" > buildinfo.txt'

displayName: 'Write build info'

workingDirectory: $(wwwrootDir)

- task: NuGetCommand@2

displayName: 'Restore project dependencies'

inputs:

command: 'restore'

restoreSolution: '\*\*/\*.sln'

feedsToUse: 'select'

vstsFeed: 'Space Game - web - Dependencies/Tailspin.SpaceGame.Web.Models'

- task: DotNetCoreCLI@2

displayName: 'Build the project - $(buildConfiguration)'

inputs:

command: 'build'

arguments: '--no-restore --configuration $(buildConfiguration)'

projects: '\*\*/\*.csproj'

- task: DotNetCoreCLI@2

displayName: 'Publish the project - $(buildConfiguration)'

inputs:

command: 'publish'

projects: '\*\*/\*.csproj'

publishWebProjects: false

arguments: '--no-build --configuration $(buildConfiguration) --output $(Build.ArtifactStagingDirectory)/$(buildConfiguration)'

zipAfterPublish: true

- task: PublishBuildArtifacts@1

displayName: 'Publish Artifact: drop'

condition: succeeded()

The highlighted code shows where the pipeline restores dependencies, and looks in your Azure Artifacts feed for the dependencies that might be there.

1. Stage, commit, and push your changes to GitHub.

BashCopy

git add .

git commit -m "Add reference to Models package"

git push origin models-package

1. Go to Microsoft Azure Pipelines and watch the build run. The build picks up your Models package from Azure Artifacts, and builds the project successfully.