

# LIKE A BOSS

## FLATRIS PIPELINE BUILD

QUICK REFERENCE GUIDE



I realize it can be hard to follow along with a video sometimes. If you get stuck, you can always refer to the steps documented here. This should have you up and running with your Flatris game in minutes. Enjoy.

### 1. In Azure, create a Linux WebApp Service

Home > Linux5601RG > New >

### Create Web App

any platform. Meet rigorous performance, scalability, security and compliance requirements while using a fully managed platform to perform infrastructure maintenance. [Learn more](#)

**Project Details**  
Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \*    
Resource Group \*    
[Create new](#)

**Instance Details**

Name \*   .azurewebsites.net

Publish \*  Code  Docker Container

Runtime stack \*

Operating System \*  Linux  Windows

Region \*    
ⓘ Not finding your App Service Plan? Try a different region.

**App Service Plan**  
App Service plan pricing tier determines the location, features, cost and compute resources associated with your app. [Learn more](#)

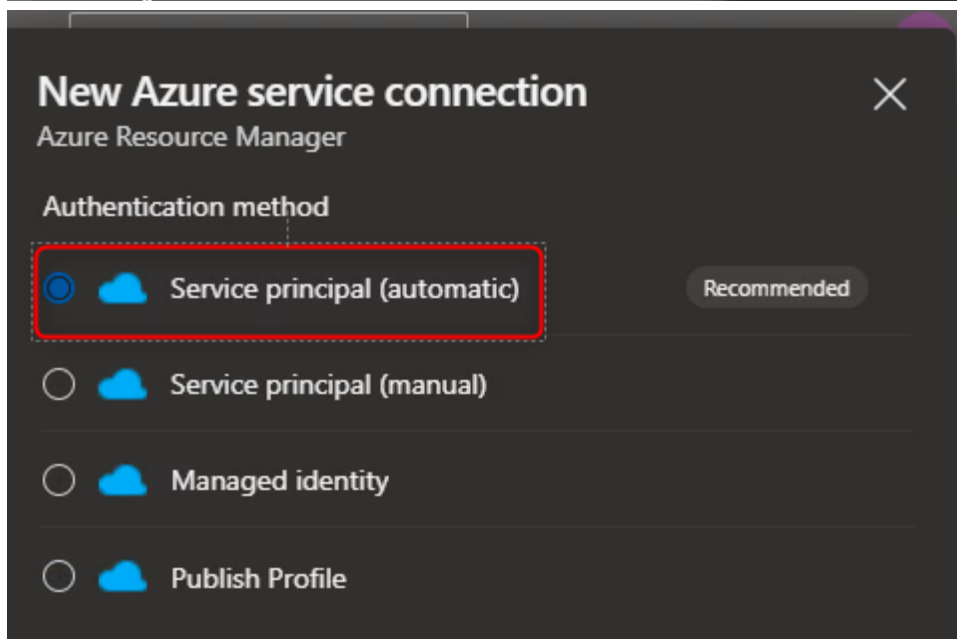
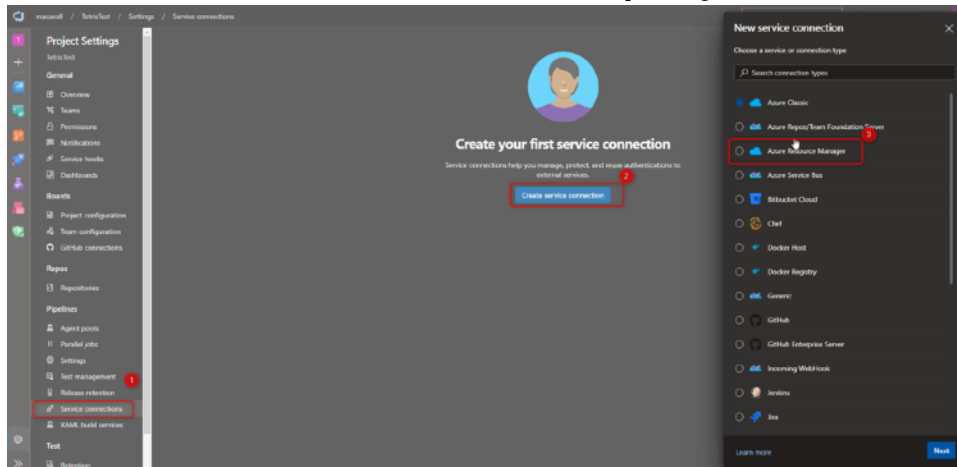
Linux Plan (East US 2) \*    
[Create new](#)

Sku and size \* **Premium V2 P1v2**  
210 total ACU, 3.5 GB memory  
[Change size](#)

Make sure to use a clone of the Flatris-LAB GitHub repository:  
<https://github.com/likeabosslearning/Flatris-LAB>

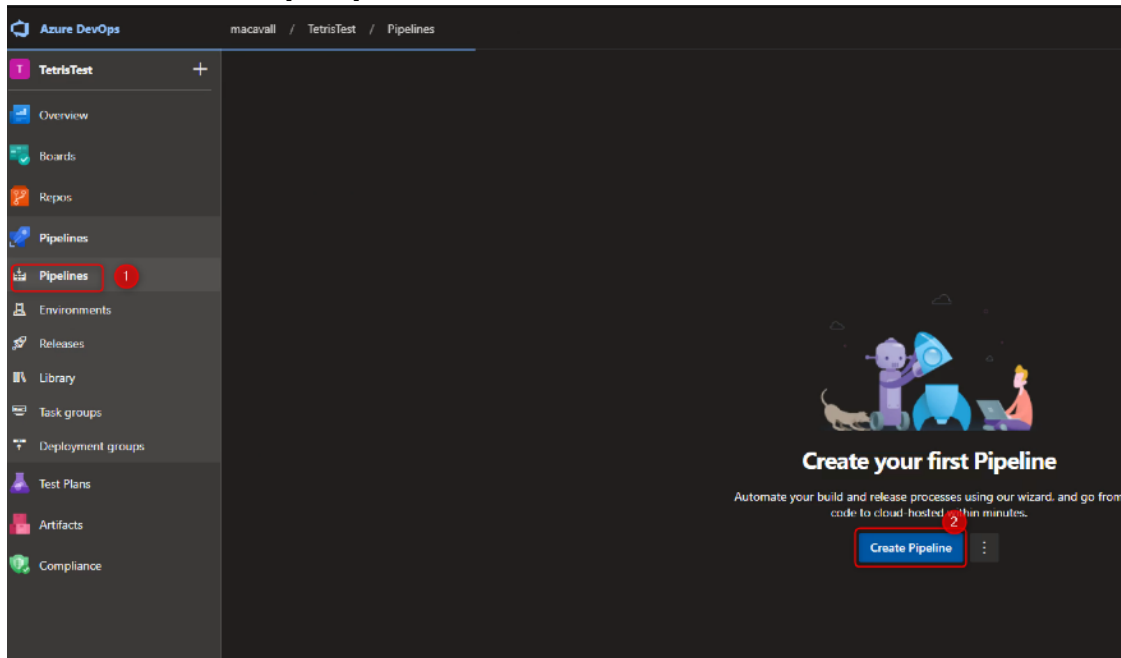
<https://t.me/learningnets>

## 2. Create a Service Connection to link DevOps to your Azure subscription.

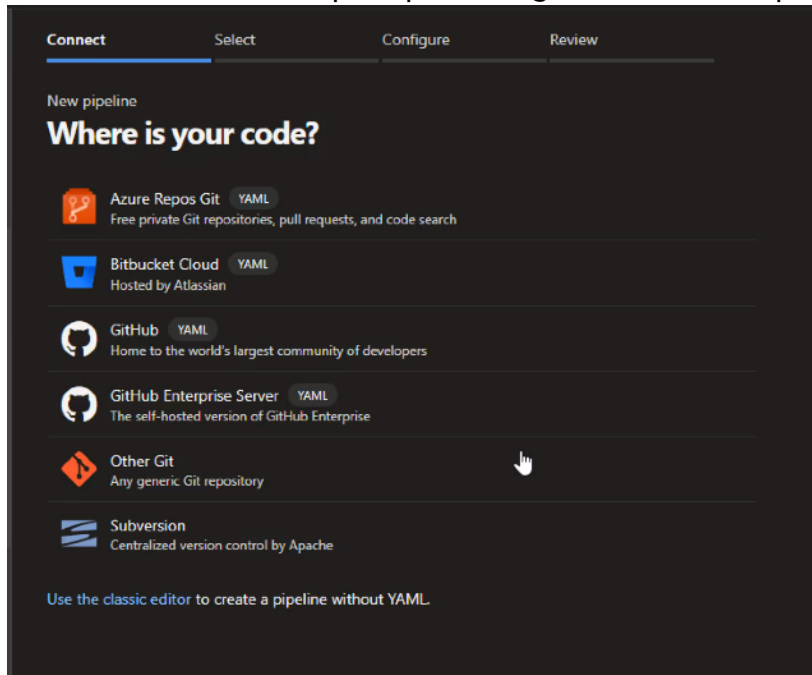


You'll use this Service connection name in the YAML file.

### 3. Create the DevOps Pipeline.



Point to the Azure DevOps Repo hosting the Flatris DevOps app.



Use the first option, 'Node.js to get started with a general Node.js project.

#### 4. Use the following custom YAML file, replacing the first 4 variables with the Service Connection, App Service, and Resource Group Names that match your environment. (You should be able to copy and paste.)

```
# Node.js Express Web App to Linux on Azure
# Build a Node.js Express app and deploy it to Azure as a Linux web app.
# Add steps that analyze code, save build artifacts, deploy, and more:
# https://docs.microsoft.com/azure/devops/pipelines/languages/javascript

trigger:
- master

variables:

  # Azure Resource Manager connection created during pipeline creation
  azureSubscription: 'ServiceConnectionName'

  # Web app name
  webAppName: 'LinuxApp5601'

  # Resource group
  resourceGroupName: 'Linux5601RG'

  # Environment name
  environmentName: 'LinuxApp5601'

  # Agent VM image name
  vmImageName: 'ubuntu-latest'

stages:
- stage: Archive
  displayName: Archive stage
  jobs:
  - job: Archive
    displayName: Archive
    pool:
      vmImage: $(vmImageName)
    steps:
    - task: AzureAppServiceSettings@1
      inputs:
        azureSubscription: $(azureSubscription)
        appName: $(webAppName)
        resourceGroupName: $(resourceGroupName)
        appSettings: |
          [
            {
              "name": "SCM_DO_BUILD_DURING_DEPLOYMENT",
              "value": "true"
            }
          ]
    - task: ArchiveFiles@2
      displayName: 'Archive files'
      inputs:
        rootFolderOrFile: '$(System.DefaultWorkingDirectory)'
        includeRootFolder: false
        archiveType: zip
        archiveFile: $(Build.ArtifactStagingDirectory)/$(Build.BuildId).zip
```

```
    replaceExistingArchive: true

  - upload: $(Build.ArtifactStagingDirectory)/$(Build.BuildId).zip
    artifact: drop

- stage: Deploy
  displayName: Deploy stage
  dependsOn: Archive
  condition: succeeded()
  jobs:
  - deployment: Deploy
    displayName: Deploy
    environment: $(environmentName)
    pool:
      vmImage: $(vmImageName)
    strategy:
      runOnce:
        deploy:
          steps:
            - task: AzureWebApp@1
              displayName: 'Azure Web App Deploy: Matt-Test-NodeJS-Deploy'
              inputs:
                azureSubscription: $(azureSubscription)
                appType: webAppLinux
                appName: $(webAppName)
                runtimeStack: 'NODE|10.14'
                package: $(Pipeline.Workspace)/drop/$(Build.BuildId).zip
```

**5. Save and run the pipeline, then open the link associated with your WebApp.**