**Summary**

**Summary**

* 3 minutes

Great work! You covered a lot of ground in this module. You and the team have come a long way in creating an automated pipeline. You learned how to map script commands on a build server to automated pipeline tasks that run when you push code to GitHub. The result of the pipeline is a *.zip* file that contains the built *Space Game* web application.

Along the way, you learned how to use variables to simplify your code.

You also learned how to use templates to encapsulate sets of tasks that you can repeat throughout your build process. You used a template to build the application's Release and Debug configurations.

Finally, you practiced your Git skills by pushing commits to a branch and building from that branch. Working from a branch lets you work in isolation from the main code base. That helps you experiment and try new things without affecting the main development branch, master.

Keep in mind that this build configuration focuses on building a .NET Core application. The tasks you choose will depend on the kind of application you're building, the tools you use to build it, and the programming language it's written in.

When you approach your own builds, it's a good idea to start by making sure you can build the application from scratch from the terminal or from a Bash or PowerShell script. From there, you can map each command to a pipeline task that accomplishes the same thing.

When you create your own pipeline, you have two choices: the visual designer or YAML files. If you choose the visual designer to get started, you can switch to the YAML version of your configuration to learn how each task is structured.

**Note**

At this point, you have a build artifact that you could deploy to a QA or production environment. For the rest of this learning path, you'll focus on using Azure Pipelines to build and test your software. You'll learn how to use Azure Pipelines to deploy your applications in a future learning path.

**Additional resources**

Here are some additional resources you can use to further your understanding.

If you're looking for a more academic approach, [*Continuous Delivery*](https://www.oreilly.com/library/view/continuous-delivery-reliable/9780321670250) by Jez Humble and David Farley is a great starting point.

**Learn YAML**

If you're interested in learning YAML, check out [Learn YAML in Y minutes](https://learnxinyminutes.com/docs/yaml). You can then review the Azure Pipelines [YAML schema reference](https://docs.microsoft.com/en-us/azure/devops/pipelines/yaml-schema?view=azure-devops&tabs=schema) to get a better sense of how pipelines are structured.

**Explore the documentation**

We provide complete [reference documentation](https://docs.microsoft.com/en-us/azure/devops/pipelines/?view=azure-devops) that goes deeper into the concepts and tasks we discussed here. The documentation also provides starter guides for many application types, like Java, C++, and Node.js.

The [Build and release tasks](https://docs.microsoft.com/en-us/azure/devops/pipelines/tasks/?view=azure-devops) section can help you map your existing build commands to built-in tasks.

**Create your own build pipeline**

In this module, you created the pipeline from Azure DevOps. You can repeat a similar process to create your own pipeline. You can also create a pipeline from the Azure portal or from the GitHub Marketplace application.

* [Use the Azure portal](https://docs.microsoft.com/en-us/azure/devops/pipelines/get-started-azure-devops-project?view=azure-devops)
* [Azure Pipelines](https://github.com/marketplace/azure-pipelines) application on GitHub Marketplace