

Cross Platform Tasks

There are several options to alleviate cross-platform compatibility issues.

- Running tasks in a container
- Using the tasks-system in VS Code which provides options to allow commands to be executed specific to an operating system.

Docker or Container Based

Using containers as development machines allows developers to get started with minimal setup and abstracts the development environment from the host OS by having it run in a container. DevContainers can also help in standardizing the local developer experience across the team.

The following are some good resources to get started with running tasks in DevContainers

- [Developing inside a container.](#)
- [Tutorial on Development in Containers](#)
- For samples projects and dev container templates see [VS Code Dev Containers Recipe](#)
- [Dev Containers Library](#)

Tasks in VSCode

Running Node.js

The example below offers insight into running Node.js executable as a command with tasks.json and how it can be treated differently on Windows and Linux.

```
{
  "label": "Run Node",
  "type": "process",
  "windows": {
    "command": "C:\\\\Program Files\\nodejs\\node.exe"
  },
  "linux": {
    "command": "/usr/bin/node"
  }
}
```

In this example, to run Node.js, there is a specific windows command, and a specific linux command. This allows for platform specific properties. When these are defined, they will be used instead of the default properties when the command is executed on the Windows operating system or on Linux.

Custom Tasks

Not all scripts or tasks can be auto-detected in the workspace. It may be necessary at times to defined your own custom tasks. In this example, we have a script to run in order to set up some environment correctly. The script is stored in a folder inside your workspace and named test.sh for Linux & macOS and test.cmd for Windows. With the tasks.json file, the execution of this script can be made possible with a custom task that defines what to do on different operating systems.

```
{
  "version": "2.0.0",
```

```
"tasks": [  
  {  
    "label": "Run tests",  
    "type": "shell",  
    "command": "./scripts/test.sh",  
    "windows": {  
      "command": ".\\scripts\\test.cmd"  
    },  
    "group": "test",  
    "presentation": {  
      "reveal": "always",  
      "panel": "new"  
    }  
  }  
]  
}
```

The command here is a shell command and tells the system to run either the test.sh or test.cmd. By default, it will run test.sh with that given path. This example here also defines Windows specific properties and tells it execute test.cmd instead of the default.

Resources

VS Code Docs - [operating system specific properties](#)

Last update: August 26, 2024