

# DevContainers Configuration

Git commit with doc: 3ea4da5

Author(s): Paulo Jerônimo

[HTML version](#)

1. Introduction .....	1
2. Configuration Details .....	1
2.1. Base Image .....	1
2.2. VS Code Extensions .....	2
2.3. Port Forwarding .....	2
2.4. Post-Creation Setup .....	2
3. Usage .....	3
3.1. Starting the DevContainer .....	3
3.2. Verifying the Setup .....	3
4. Customization .....	4

## 1. Introduction

This project uses Development Containers (devcontainers) to provide a consistent development environment across different machines and platforms. The configuration ensures that all developers have the same tools and dependencies installed, reducing "it works on my machine" issues.

## 2. Configuration Details

### 2.1. Base Image

The project uses Microsoft's Universal DevContainer image as its base:

```
"image": "mcr.microsoft.com/devcontainers/universal:2-linux",
```

This image comes with many development tools pre-installed, including:

- Git
- nvm (Node Version Manager)
- Basic build tools
- Common Unix utilities

## 2.2. VS Code Extensions

The following VS Code extensions are automatically installed:

```
"customizations": {  
  "vscode": {  
    "extensions": [  
      "vscjava.vscode-java-pack",  
      "redhat.java",  
      "vscjava.vscode-gradle",  
      "ms-azuretools.vscode-docker"  
    ]  
  }  
}
```

## 2.3. Port Forwarding

The container automatically forwards port 8080, which is used by the Redpanda Console:

```
}  
"forwardPorts": [  
  8080  
]
```

## 2.4. Post-Creation Setup

After the container is created, the `postCreate.sh` script runs automatically to set up the development environment. This script:

1. Updates the system packages and installs tmux:

```
echo Installing system packages...  
sudo apt-get update -y  
  
echo Installing/ configuring tmux...  
sudo apt-get install -y tmux  
cp .devcontainer/.tmux.conf ~/
```

2. Installs Java 21 (Temurin) using SDKMAN:

```
echo Installing Java 21 via SDKMAN...  
curl -s "https://get.sdkman.io" | bash  
source "/usr/local/sdkman/bin/sdkman-init.sh"  
yes | sdk install java 21.0.7-tem
```

3. Installs the LTS version of Node.js and sets it as default:

```
echo Installing Node.js LTS...
export NVM_DIR=/usr/local/share/nvm
source $NVM_DIR/nvm.sh
nvm install --lts
nvm alias default 'lts/*'
nvm use default
```

## 3. Usage

### 3.1. Starting the DevContainer

There are three ways to use this devcontainer:

1. In GitHub Codespaces:
  - Click the green "Code" button on the GitHub repository
  - Select "Create codespace on main"
2. In VS Code:
  - Install the "Remote Development" extension
  - Clone the repository
  - Press F1 and select "Dev Containers: Reopen in Container"
3. Using DevContainer CLI:

```
# Install the CLI
npm install -g @devcontainers/cli

# Start the container
devcontainer up --workspace-folder .

# Open a shell in the container
devcontainer exec --workspace-folder . bash
```

### 3.2. Verifying the Setup

After the container starts and the setup completes, you can verify the installation:

```
# Check Java version
java -version

# Check Node.js version
node --version

# Check if tmux is installed
```

## 4. Customization

If you need to modify the development environment:

- Edit `.devcontainer/devcontainer.json` for container configuration
- Modify `.devcontainer/postCreate.sh` for setup changes
- Update VS Code extensions in the `customizations.vscode.extensions` section of `devcontainer.json`