Java Runtimes in Visual Studio Code

To use `java.configuration.runtimes` in Visual Studio Code (VS Code) settings, you need to configure it in your `settings.json` file. This setting allows you to specify multiple Java Development Kit (JDK) versions that VS Code can use for your Java projects. It's especially useful when working with different projects that require specific Java versions. Here's a step-by-step guide to set it up:

Steps to Configure `java.configuration.runtimes`

1. **Open VS Code Settings**:
   - On Windows/Linux: Go to `File > Preferences > Settings` or press `Ctrl + ,`.
   - On macOS: Go to `Code > Preferences > Settings` or press `Cmd + ,`.
2. **Access the JSON Settings File**:
   - In the Settings UI, search for `java.configuration.runtimes`.
   - You'll see an option like "Java: Configuration: Runtimes". Click on "Edit in settings.json"(usually a link below the setting description) to open the `settings.json` file.
3. **Edit `settings.json`**:
   - In the `settings.json` file, add or modify the `java.configuration.runtimes` array. This array contains objects, each representing a JDK version you want VS Code to recognize.
   - Each object typically includes:
     - `name`: The Java version identifier (e.g., `JavaSE-1.8`, `JavaSE-11`, `JavaSE-17`).
     - `path`: The absolute path to the JDK installation directory on your system.
     - `default` (optional): Set to `true` to make this the default JDK for unmanaged folders (projects without build tools like Maven or Gradle).

   Here's an example configuration:

```
{
    "java.configuration.runtimes": [
        {
            "name": "JavaSE-1.8",
            "path": "C:/Program Files/Java/jdk1.8.0_351",
            "default": true
        },
        {
            "name": "JavaSE-11",
            "path": "C:/Program Files/Java/jdk-11.0.15"
        },
            {
                "name": "JavaSE-17",
                "path": "C:/Program Files/Java/jdk-17.0.6"
            }
        ]
    }
```

4. **Verify JDK Paths**:
   - Ensure the `path` points to the root directory of your JDK installation (e.g., where the `bin` folder containing `java.exe` or `java` is located).
   - On Windows, use forward slashes (`/`) or escape backslashes (`\\`) in the path.
   - On macOS/Linux, use the appropriate file system path (e.g., `/usr/lib/jvm/java-17-openjdk`).
5. **Save and Reload**:
   - Save the `settings.json` file.
   - Restart VS Code or reload the window (`Ctrl + R` or `Cmd + R`) to apply the changes.
6. **Check Configuration**:
   - Open the Command Palette (`Ctrl + Shift + P` or `Cmd + Shift + P`) and run the command `Java: Configure Java Runtime`.
   - This opens a view showing the JDKs available for your projects. Verify that your configured runtimes appear under the "Project JDKs"tab.

How It Works

- **Unmanaged Folders**: For projects without build tools (e.g., plain Java files), VS Code uses the `default` JDK specified in `java.configuration.runtimes`.
- **Managed Projects (Maven/Gradle)**: For projects with build tools, the JDK version is determined by the build configuration (e.g., `pom.xml` or `build.gradle`), but VS Code will still recognize the runtimes listed here for compatibility.
- **Language Server**: The Java Language Server (part of the Java extensions) requires JDK 17+ to run. You can specify this separately with `java.jdt.ls.java.home` if needed (e.g., `"java.jdt.ls.java.home": "C:/Program Files/Java/jdk-17.0.6"`).

Example with Multiple JDKs

If you're on a Mac and have JDK 8 and 17 installed via Homebrew, your configuration might look like this:

```
{
    "java.configuration.runtimes": [
        {
            "name": "JavaSE-1.8",
            "path": "/Library/Java/JavaVirtualMachines/jdk1.8.0_351.jdk/Contents/Home",
            "default": true
        },
        {
            "name": "JavaSE-17",
            "path": "/Library/Java/JavaVirtualMachines/jdk-17.0.6.jdk/Contents/Home"
        }
    ]
}
```

Tips

- Use the correct `name` format (`JavaSE-<version>`), matching the Java version (e.g., `JavaSE-11` for Java 11).
- If you only have one JDK, you can still list it to ensure VS Code uses it explicitly.
- To find your JDK path, run `java -version` in a terminal and check the installation directory (e.g., via `where java` on Windows or `which java` on Unix-like systems, then trace back to the JDK root).

This setup gives you flexibility to work with multiple Java versions in VS Code seamlessly! Let me know if you need help finding your JDK paths or troubleshooting.