GitHub Codespaces – Web 개발 환경 구축

목차

[GitHub Codespaces – Web 개발 환경 구축 1](#_Toc81578350)

[기본 개발환경에 내장된 도구 1](#_Toc81578351)

[Python 2](#_Toc81578352)

[Javascript (Frontend, Backend(Node.js)) 3](#_Toc81578353)

[Java (Spring Boot) 4](#_Toc81578354)

[.Net (ASP.NET, C#) 7](#_Toc81578355)

[Golang (Gin) 7](#_Toc81578356)

GitHub Codespace 의 기본 개발 환경에는 이미 Python, Go, Java, Node.js 등 다양한 개발 도구가 설정 되어 있습니다. 이러한 도구로 개발을 시작하는 방법에 관해 알아보도록 하겠습니다.

# 기본 개발환경에 내장된 도구

아래 두 링크를 방문하면, Codespaces 의 기본 환경의 구성이 기재되어 있습니다. 이를 참고하면 다음과 같은 도구가 미리 포함되어 있음을 알 수 있습니다. (본 문서 작성 시점 기준)

<https://github.com/microsoft/vscode-dev-containers/tree/main/containers/codespaces-linux>

<https://github.com/microsoft/vscode-dev-containers/blob/main/containers/codespaces-linux/history/dev.md>

* Ubuntu 20.04 LTS Focal Fossa x86-64
* 프로그래밍 언어 및 런타임
  + Node.js(+ nvm, nvs, yarn), Python(pylint, flake8 등 다수 패키지 포함), Java(Maven, Gradle 포함), .Net, Ruby(rbenv, ruby-build 포함), PHP(Xdebug, Composer 등 포함), PowerShell, GCC, Go(gopls, Delev 등 도구 포함), Jekyll
* 유틸리티
  + Git(Git LFS 포함), kubectl, Helm. Docker Compose, Sdkman, GitHub CLI, Docker CLI 및 Engine 등
* 그 외 사전 설치 패키지
  + Azure CLI, build-essential, gnupg2, jq 등

# Python

Python (3.x), pip, 등이 미리 설치되어 있습니다. 확장기능만 추가로 설치하여 바로 사용하시면 됩니다. 아래 명령을 순차적으로 실행하여 간단한 Django 앱 프로젝트를 초기화하여 테스트 할 수 있습니다.

Python 확장기능(확장ID: ms-python.python)

텍스트이(가) 표시된 사진

자동 생성된 설명

Django 확장기능(확장ID: batisteo.vscode-django)

텍스트이(가) 표시된 사진

자동 생성된 설명

(Python 설치 경로 조회)

|  |
| --- |
| which python |

[출력] /home/codespace/.python/current/bin/python

(Django 앱 생성 및 실행)

|  |
| --- |
| mkdir djangoapp && cd djangoapp  Python -m venv venv  . venv/bin/activate  pip install Django Django-admin  django-admin startproject djangoapp .  python manage.py runserver |

# Javascript (Frontend, Backend(Node.js))

Node.js 런타임, npm, yarn 또한 Codespace 컨테이너 환경에 이미 설치되어 있고, Javascript 지원도 GitHub Codespace 에 내장 되어 있으므로 바로 개발을 시작하면 됩니다. 아래 명령으로 간단한 Express.js 앱(백엔드) 와 React.js 앱(프론트엔드) 를 생성해서 테스트 할 수 있습니다.

(Node.js 설치 경로 조회)

|  |
| --- |
| which node |

[출력] /home/codespace/.python/current/bin/node

(Express.js 앱 생성)

|  |
| --- |
| npx express-generator expressapp  cd expressapp  npm install  npm start |

[React.js 앱 생성]

|  |
| --- |
| npx create-react-app reactapp  cd reactapp  yarn  yarn start |

# Java (Spring Boot)

Java 또한 JVM, JDK, Gradle, Maven 이 내장되어 있습니다. 여기에 프로젝트 생성과 테스트 및 빌드를 위해 몇 가지 확장 기능을 설치하여 사용하면 됩니다. 여기서는 Java 및 Spring Boot 백엔드 프로젝트 생성에 유용한 확장기능을 설치하고 프로젝트를 생성하여 실행해 보겠습니다.

다음과 같은 확장 기능 팩을 설치합니다.

(Extension Pack for Java, 확장ID: vscjava.vscode-java-pack)

텍스트이(가) 표시된 사진

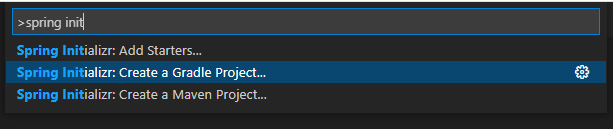
자동 생성된 설명

(Spring Boot Extension Pack, 확장ID: pivotal.vscode-boot-dev-pack)

텍스트이(가) 표시된 사진

자동 생성된 설명

Command Palete (Crtl + Shift + P) 에서 “Spring Initializer: Create a Gradle Project…” 를 선택하여 Spring Boot 프로젝트 생성을 시작합니다.



Spring Boot 버전을 선택합니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Java 를 선택합니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

패키지 네임을 지정합니다.

텍스트이(가) 표시된 사진

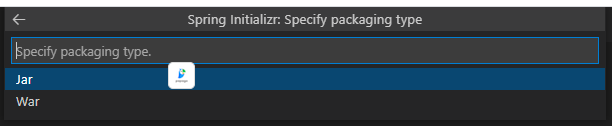
자동 생성된 설명

아티팩트 이름을 지정합니다.

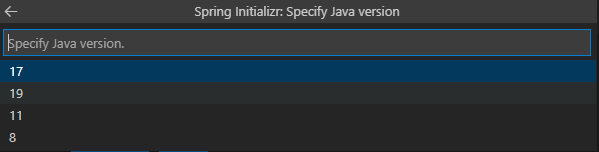
텍스트이(가) 표시된 사진

자동 생성된 설명

패키지 유형을 지정합니다.



Java 버전을 지정합니다.



필요한 의존성 라이브러리를 선택합니다. 보통 Spring Boot DevTools 는 기본적으로 사용하고, 그 외에 Web(MVC), Rest Repositories, Session. Security, Data JPA 등 댜양한 의존성을 조합해서 사용합니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

프로젝트 파일을 생성할 경로를 지정합니다. (참고: 프로젝트 생성 확장 기능이 폴더를 자동으로 생성하지 않아, 필요한 경우 수동으로 폴더를 만들어야 합니다.)

텍스트이(가) 표시된 사진

자동 생성된 설명

터미널에서 생성된 프로젝트 경로에 접근하여 실행해 봅니다.

|  |
| --- |
| cd springbootapp/demo  gradle bootRun |

# .Net (ASP.NET, C#)

.Net 런타임과 SDK또한 미리 설치되어 있어, 확장 기능만 추가로 설치하여 바로 개발을 시작할 수 있습니다.

C# 확장기능(확장ID: ms-dotnettools.csharp)

텍스트이(가) 표시된 사진

자동 생성된 설명

아래 명령을 순차적으로 실행하여, ASP.NET Core 앱 프로젝트를 생성하고 실행할 수 있습니다.

|  |
| --- |
| dotnet new webapp -o aspnetcoreapp --no-https  dotnet restore  dotnet run |

# Golang (Gin)

Golang 툴체인과 또한 이미 설치되어 있어, 확장 기능을 추가로 설치하여 바로 개발을 시작할 수 있습니다.

Go 확장기능(확장ID: golang.go)

텍스트이(가) 표시된 사진

자동 생성된 설명

아래 절차를 따라 Gin 웹앱을 만들어 테스트 할 수 있습니다.

프로젝트 디렉터리를 만들고, Go Module 을 초기화 합니다.

|  |
| --- |
| mkdir ginapp  cd ginapp  go mod init ginapp |

아래 명령을 실행하여 main.go 파일을 생성하고 편집기로 엽니다.

|  |
| --- |
| code main.go  편집기에 아래 Gin 앱 예제 코드를 넣고 저장 합니다.  package main  import "github.com/gin-gonic/gin"  func main() {  r := gin.Default()  r.GET("/ping", func(c \*gin.Context) {  c.JSON(200, gin.H{  "message": "pong",  })  })  r.Run() // listen and serve on 0.0.0.0:8080 (for windows "localhost:8080")  } |

소스코드에서 의존성을 찾아 의존성 정보가 담긴 go.mod, go.sum 파일을 갱신하고, 바로 Gin 웹앱을 실행 합니다.

|  |
| --- |
| go mod tidy  go run . |