멀티 스테이지 빌드 (Multi-stage builds)

멀티 스테이지 빌드 사용

멀티 스테이지 빌드에서는 Dockerfile에서 여러 FROM 문을 사용합니다.
각 FROM 명령어(instruction)는 서로 다른 기반을 사용할 수 있으며 각 명령어는 빌드의 새로운 단계(stage)를 시작합니다.
최종 이미지에서 원하지 않는 모든 것을 남겨 두고 한 단계(stage)에서 다른 단계로 artifacts(유물, 결과물?)를 선택적으로 복사할 수 있습니다.

다음 Dockerfile에는 두 개의 개별 단계가 있습니다. 하나는 바이너리를 빌드하는 단계이고 다른 하나는 바이너리를 복사하는 단계입니다.

```
# syntax=docker/dockerfile:1
FROM golang:1.21
WORKDIR /src
COPY <<EOF ./main.go
package main

import "fmt"

func main() {
    fmt.Println("hello, world")
}
EOF
RUN go build -o /bin/hello ./main.go

FROM scratch
COPY --from=0 /bin/hello /bin/hello
CMD ["/bin/hello"]
```

최종 결과는 내부에 바이너리만 포함된 작은 프로덕션 이미지입니다.
결과 이미지에 애플리케이션을 빌드하는 데 필요한 빌드 도구는 포함되지 않습니다.

두 번째 FROM 명령어는 scratch 이미지를 기본으로 사용하여 새로운 빌드 단계(stage)를 시작합니다.
`COPY --from=0` 라인은 이전 스테이지에서 빌드된 결과만 새 스테이지에 복사합니다.
Go SDK 및 모든 중간 결과는 남겨지며 최종 이미지에 저장되지 않습니다.

빌드 스테이지 이름 지정 (Name your build stages)

기본적으로 스테이지는 이름이 지정되지 않으며 첫 번째 FROM 명령에 대해 0부터 시작하여 정수 번호로 스테이지를 참조합니다.
그러나 FROM 명령어에 AS <NAME>을 추가하여 스테이지 이름을 지정할 수 있습니다.
이 예제는 스테이지 이름을 지정하고 COPY 명령에서 해당 이름을 사용하여 이전 스테이지를 개선합니다.
이는 Dockerfile의 명령(instructions)이 나중에 다시 정렬(re-ordered)되어도 COPY가 중단되지 않음을 의미합니다.

```
# syntax=docker/dockerfile:1
FROM golang:1.21 as build
WORKDIR /src
COPY <<EOF /src/main.go
package main

import "fmt"

func main() {
    fmt.Println("hello, world")
}
EOF
RUN go build -o /bin/hello ./main.go

FROM scratch
COPY --from=build /bin/hello /bin/hello
CMD ["/bin/hello"]
```

특정 빌드 스테이지에서 중지 (Stop at a specific build stage)

이미지를 빌드할 때 반드시 모든 스테이지를 포함하여 전체 Dockerfile을 빌드할 필요는 없습니다. 대상 빌드 스테이지를 지정할 수 있습니다.
빌드 하고싶은 스테이지를 지정할 수 있습니다. 다음 명령은 이전 Dockerfile을 사용하고 있지만 'build'라는 스테이지에서 중지한다고 가정합니다.

```
$ docker build --target build -t hello .
```

`--target` 옵션으로 중간 빌드 스테이지를 결과 이미지의 최종 스테이지로 지정할 수 있습니다. 데몬(daemon)은 지정한 스케이지 이후의 명령을 건너뜁니다.

외부 이미지를 스테이지로 사용 (Use an external image as a stage)

멀티 스테이지 빌드를 사용하는 경우 이전에 생성한 스테이지에서만 복사하는 것으로 제한되지 않습니다.
`COPY --from` 명령을 사용하면 로컬 이미지 이름, 로컬 또는 Docker 레지스트리에서 사용 가능한 태그(tag), 또는 태그 ID를 사용하여 별도의 이미지에서 복사할 수 있습니다.
Docker 클라이언트는 필요한 경우 이미지를 가져오고(pull) 거기에서 artifact를 복사합니다.
구문은 다음과 같습니다.

```
$ COPY --from=nginx:latest /etc/nginx/nginx.conf /nginx.conf
```

이전 스테이지를 새 스테이지로 사용 (Use a previous stage as a new stage)

FROM 을 사용할 때 이전 스테이지를 참조하여 중단된 위치를 선택할 수 있습니다.

```
# syntax=docker/dockerfile:1

FROM alpine:latest AS builder
RUN apk --no-cache add build-base

FROM builder AS build1
COPY source1.cpp source.cpp
RUN g++ -o /binary source.cpp

FROM builder AS build2
COPY source2.cpp source.cpp
RUN g++ -o /binary source.cpp
```

Differences between legacy builder and BuildKit