Dockerfile best practice

Incremental build time

Tip #1: Order matters for caching

```
FROM debian
COPY . /app
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh vim
COPY . /app
CMD ["java", "-jar", "/app/target/app.jar"]
```

Tip #2: More specific COPY to limit cache busts

```
FROM debian
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh vim
COPY . /app
COPY target/app.jar /app
CMD ["java", "-jar", "/app/target/app.jar"]
```

Tip #3: Identify cacheable units such as apt-get update & install

```
FROM debian
RUN apt-get update
RUN apt-get -y install openjdk-8-jdk ssh vim
RUN apt-get update \
 && apt-get -y install \
    openjdk-8-jdk ssh vim
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Reduce Image size

Tip #4: Remove unnecessary dependencies

```
FROM debian
RUN apt-get update \
 && apt-get -y install --no-install-recommends \
    openjdk-8-jdk ssh vim
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

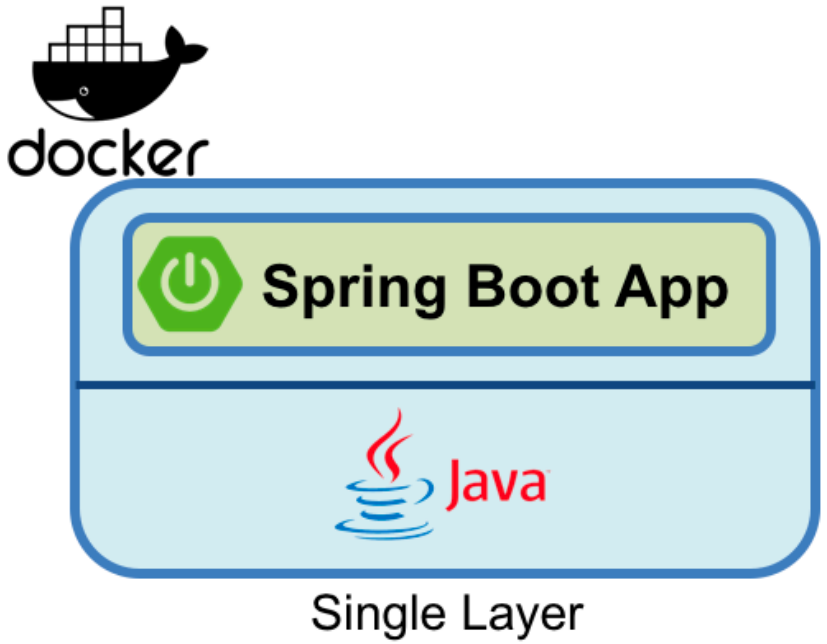
Tip #5: Remove package manager cache

```
FROM debian
RUN apt-get update \
 && apt-get -y install --no-install-recommends \
    openjdk-8-jdk \
 && rm -rf /var/lib/apt/lists/*
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Maintainability

Tip #6: Use official images when possible

```
FROM debian
RUN apt-get update \
 && apt-get -y install --no-install-recommends \
    openjdk-8-jdk\
 && rm -rf /var/lib/apt/lists/*
FROM openjdk
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```



Tip #7: Use more specific tags

```
FROM openjdk:latest
FROM openjdk:8
COPY target/app.jar /app
CMD ["java", "-jar", "/app/app.jar"]
```

Tip #8: Look for minimal flavors

REPOSITORY	TAG	SIZE
openjdk	8	624MB
openjdk	8-jre	443MB
openjdk	8-jre-slim	204MB
openjdk	8-jre-alpine	83MB

Reproducibility

Tip #9: Build from source in a consistent environment

The source code is the source of truth from which you want to build a Docker image. The Dockerfile is simply the blueprint.

```
~~FROM openjdk:8-jre-alpine~~
FROM maven:3.6-jdk-8-alpine
WORKDIR /app
~~COPY app.jar /app~~
COPY pom.xml .
COPY src ./src
RUN mvn -e -B package
CMD ["java", "-jar", "/app/app.jar"]
```

Tip #10: Fetch dependencies in a separate step

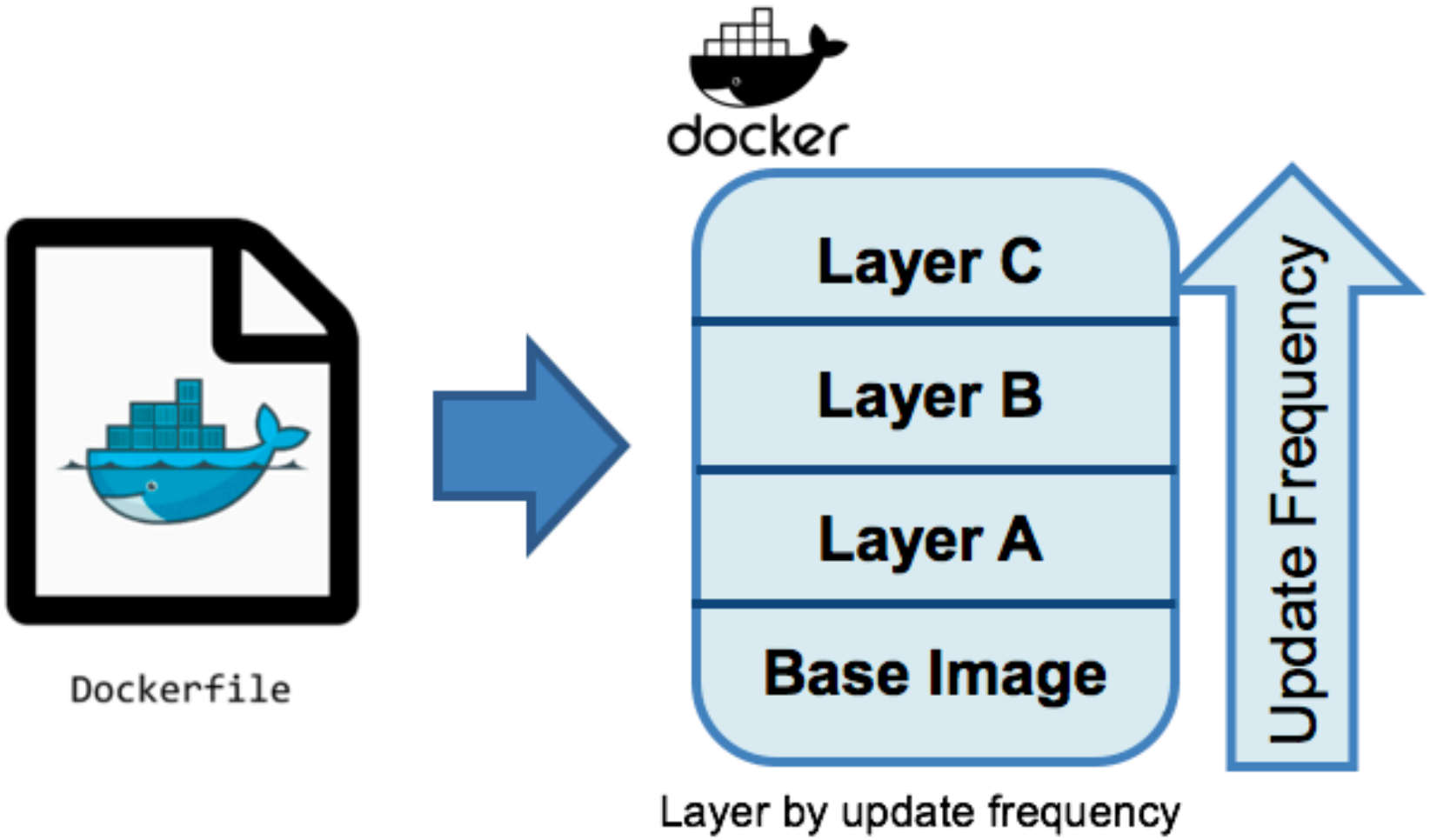
```
FROM maven:3.6-jdk-8-alpine
WORKDIR /app
COPY pom.xml .
RUN mvn -e -B dependency:resolve
COPY src ./src
RUN mvn -e -B package
CMD ["java", "-jar", "/app/app.jar"]
```

Reproducibility

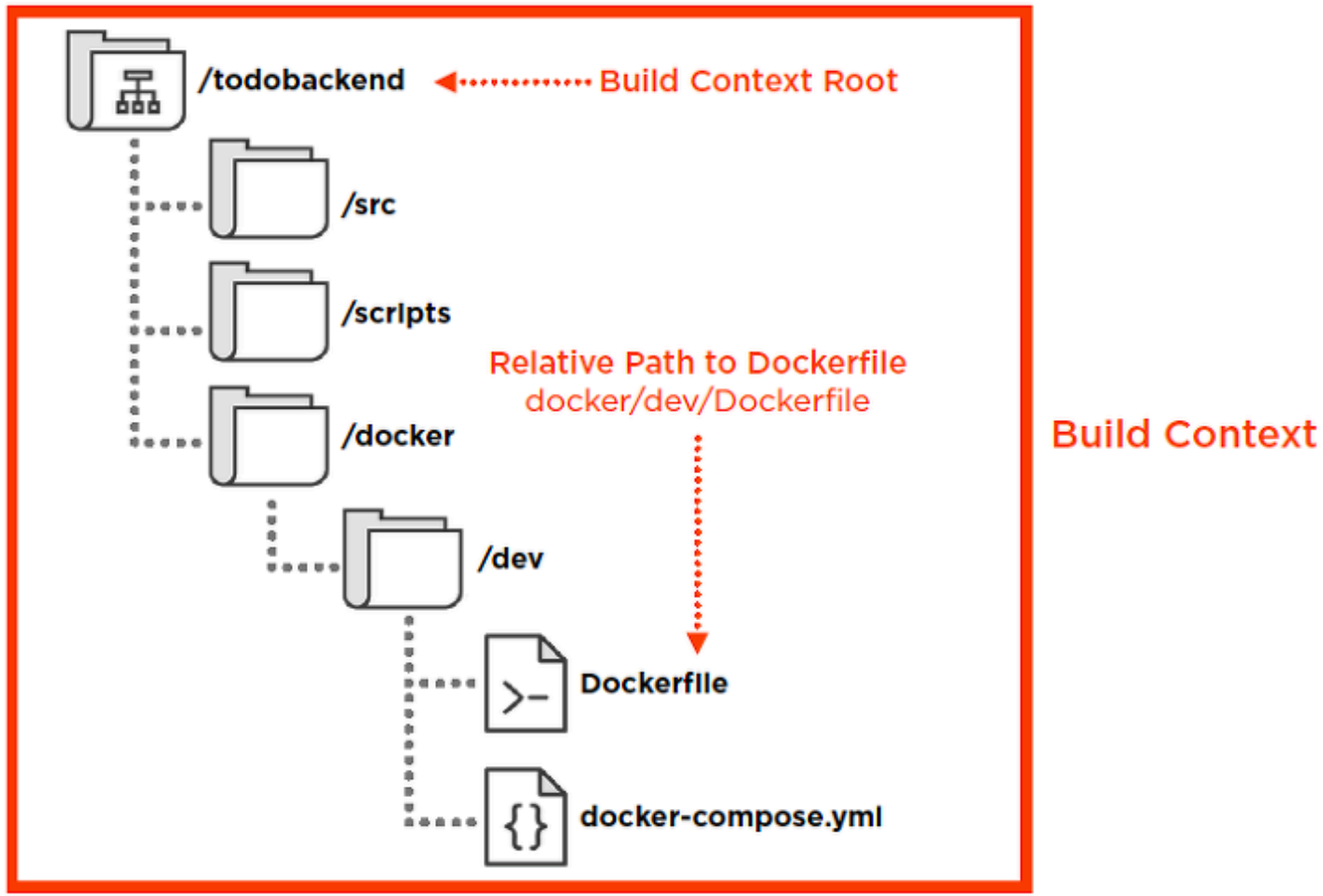
Tip #11: Use multi-stage builds to remove build dependencies (recommended Dockerfile)

```
FROM maven:3.6-jdk-8-alpine AS builder
WORKDIR /app
COPY pom.xml .
RUN mvn -e -B dependency:resolve
COPY src ./src
RUN mvn -e -B package
CMD ["java", "-jar", "/app/app.jar"]

FROM openjdk:8-jre-alpine
COPY --from=builder /app/target/app.jar /
CMD ["java", "-jar", "/app.jar"]
```


docker
Layer C
Layer B
Layer A
Base Image
Update Frequency
Dockerfile
Layer by update frequency

Understand build context



Exclude with .dockerignore

```
# comment
*/temp*
*/*/temp*
temp?
```

This file causes the following build behavior:

Rule	Behavior
`# comment`	Ignored.
`*/temp*`	Exclude files and directories whose names start with `temp` in any immediate subdirectory of the root. For example, the plain file `/somedir/temporary.txt` is excluded, as is the directory `/somedir/temp`.
`*/*/temp*`	Exclude files and directories starting with `temp` from any subdirectory that is two levels below the root. For example, `/somedir/subdir/temporary.txt` is excluded.
`temp?`	Exclude files and directories in the root directory whose names are a one-character extension of `temp`. For example, `/tempa` and `/tempb` are excluded.