Build Tools

Effective Programming in Scala

Developer Workflow

As a developer, when you work on a program, you need to:

- compile
- run
- test
- deploy

Some of these tasks:

- can also be performed by a continuous integration server,
- may be automatically triggered on source changes.

Compiling

What is it to compile a program?

- invoking the compiler on all the source files to turn them into executable JVM bytecode

But it also requires:

- constructing the application classpath by fetching library dependencies,
- generating parts of the source code or resources (e.g., assets, data type serializers, etc.)

Running and Testing

Building an execution environment (ie, a JVM with the correct classpath), invoking the entry point(s).

Requires compiling the program.

Deploying and Publishing

Can take various forms:

- publish an artifact on a library repository,
- package the program and its dependencies as a single .jar file,
- publish a Docker image,
- publish documentation,
- ...

Requires compiling the program.

Build Tools

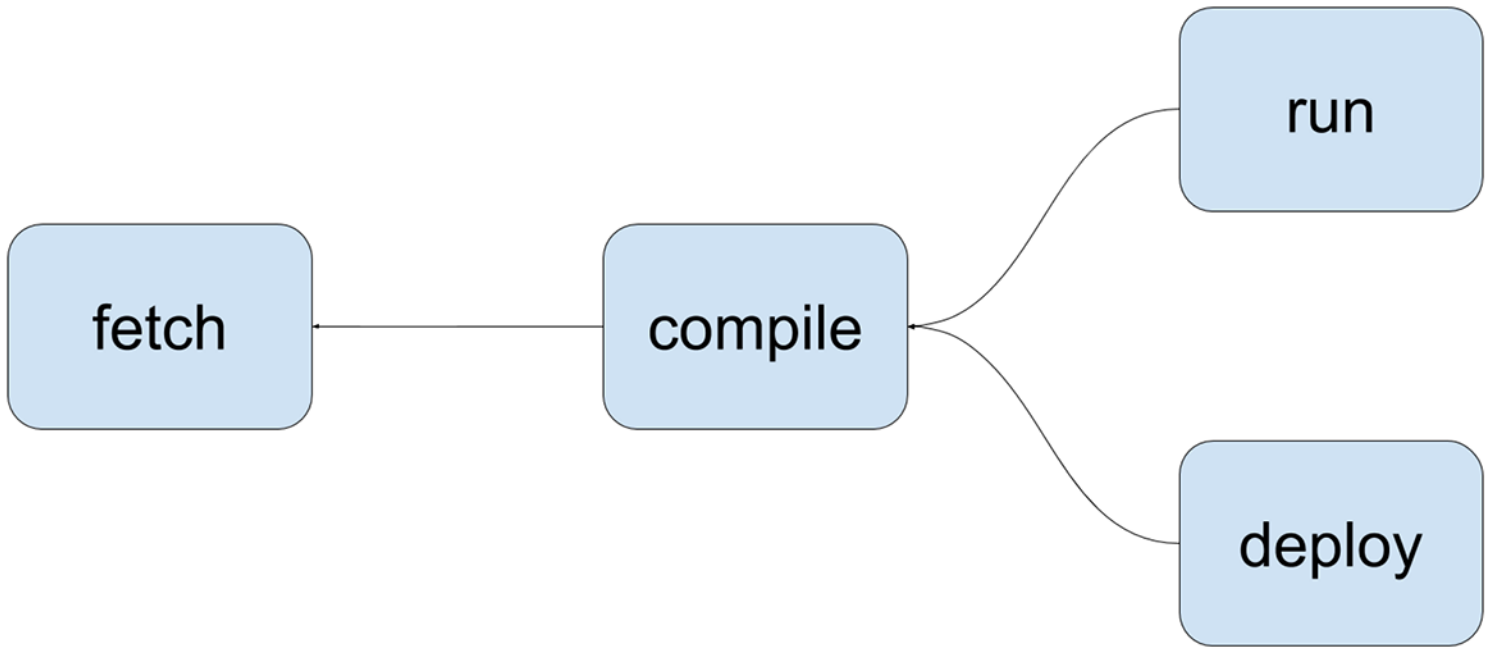
Manually invoking these tasks (compiling, fetching dependencies, etc.) in the right order is cumbersome.

Instead, **build tools** can manage these tasks for you.

Idea: you indicate where your program source files are located, which libraries your program depends on, etc., and the build tool takes care of fetching the dependencies, invoking the compiler, running the tests, etc.

Task Engine

Generally, build tools are not limited to a specific developer workflow but implement a generic task engine able to coordinate the execution of a graph of tasks.



Summary

Working on a program involves performing various interdependent tasks such as compiling, running, and deploying the program.

Build tools aim at simplifying the coordination of these tasks.