Building Go Services With Bazel

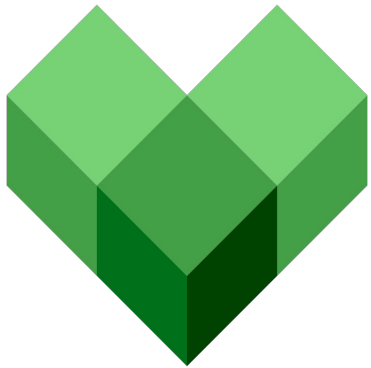
~Four Years Later

Paul MacKay - February 21, 2023

April 2019

Building Go Services With Bazel

What has happened since then?



A lot of things!

April 2019 - Version 0.23.0

October 2019 - Version 1.0.0

Feb 2023 - Version 6.0.0 LTS

Bazel is used in production in hundreds of companies worldwide

Bazelcon 2019-2022

Multiple Companies Built Around Bazel

But first, what is Bazel again?

Bazel is derived from Google's internal build system

- Open-source 2015 (port of Google’s Blaze build system)
- 100% reproducible builds in sandboxed environment
- Supports any language (an extension language - Starlark)
- Supports scaling to lots of files (100k’s)
- Quick builds and tests via caching

Why should you use Bazel?

Use Bazel When You:

- Want a build system that is platform independent
- Have multiple languages in you organization
- Have a monorepo
- Have a large code base and building and testing take forever
- Don’t want to leave “side-effects” from the build process
- Have run out of CPU and Memory on your laptop

{Fast, Correct} - Choose Two

Bazel is fast

- Bazel creates a dependency tree for each build target
- Each node of the dependency tree is cached
- Unit tests are cached if they have already been run
- File changes are detected by content digest not timestamps

Bazel correct builds every time - Deterministic

- Each build target declares its sources and dependencies
- Every build step happens in a sandbox area (“hermetically sealed”)
  - Cryptographic hashing of all inputs - files, toolchain and environment
- Every build produces the correct output each time

Bazel is an Artifact-Based Build System

- Bazel is an artifact-based build system rather than a task-based build system (e.g. Make, Maven)

- Two basic concerns:
  - Artifacts
  - Actions

Where do I begin?

The new way to install Bazel is via:

`bazelisk`

To install Bazel on a mac:

```
brew install bazelisk
```

To specify which version of Bazel to use:

- If the environment variable `USE_BAZEL_VERSION` is set
- Else if a `.bazeliskrc` file exists in the workspace root and contains the USE_BAZEL_VERSION variable
- Else if a `.bazelversion` file exists in the current directory or recursively any parent directory, it will read the file and use the version specified in it
- Else the latest version

Bazel Terminology

Workspace - A directory tree of source files to build

Repositories - A directory containing a WORKSPACE.bazel file is the root of the main repository, also called “@”. Other, (external) repositories are defined in the WORKSPACE.bazel file using workspace rules.

WORKSPACE/WORKSPACE.bazel - May be empty, or may contain references to external dependencies required to build the outputs

Packages - Is defined as a directory containing a BUILD.bazel and zero or more related files

Bazel Terminology (cont.)

Targets - Elements of a package (files and rules)

Labels - Uniquely identifies a target

Rules - Specifies relationships between input and output

BUILD/BUILD.bazel - Files containing the rules used to build the target

To build a program with Bazel, type bazel build followed by the target you want to build

What Does Bazel Do When I Run It?

When running a build or a test, Bazel does the following:

1. **Loads** the BUILD.bazel files relevant to the target
2. **Analyzes** the inputs and their dependencies, applies the specified build rules, and produces an action graph.
3. **Executes** the build actions on the inputs until the final build outputs are produced.

A Simple Go App Using Bazel

```
/
|- WORKSPACE.bazel
|- BUILD.bazel
|- cmd/
  |- myapp1/
    |- BUILD.bazel
    |- main.go
    |- internal/
    |- internal/BUILD.bazel
    |- internal/app.go
  |- myapp2/
    |- BUILD.bazel
    |- main.go
    |- internal/
    |- internal/BUILD.bazel
    |- internal/app.go
```

DEMO

- Simple Build
- Container Image Build

Questions?

Thank You!

This presentation and other information is found at:

https://github.com/whizard/bazel-talk-2023

References

- https://bazel.build/install
- https://bazel.build/configure/best-practices
- https://github.com/bazelbuild/bazel-gazelle
- https://github.com/bazelbuild/rules_go
- https://github.com/bazelbuild/bazel-skylib
- https://earthly.dev/blog/bazel-build/
- https://www.buildbuddy.io/blog/whats-new-in-bazel-6-0/
- https://opensourcelive.withgoogle.com/events/bazelcon2022
- https://sluongng.hashnode.dev/bazel-caching-explained-pt-1-how-bazel-works
- https://docs.aspect.build/guides/bazelrc
- https://github.com/buchgr/bazel-remote
- https://github.com/buildbarn

References (cont.)

- https://github.com/bazelbuild/bazel-buildfarm
- https://medium.com/@simontoth/golang-with-bazel-2b5310d4ce48
- https://www.youtube.com/watch?v=toPWLiUq5Ps
- https://www.youtube.com/watch?v=KZIYdxsRp4w
- https://github.com/systemlogic/learn-bazel
- https://www.aspect.dev
- https://www.buildbuddy.io
- https://www.engflow.com
- https://github.com/twitter/scoot
- https://www.tweag.io/blog/2021-09-08-rules_go-gazelle/
- https://medium.com/@helbingxxx/building-monorepo-with-bazel-59e3488d611e
- https://www.brandongreeley.com/an-intro-to-building-a-go-application-with-bazel
- https://github.com/adobe/rules_gitops
- https://github.com/kriscfoster/multi-language-bazel-monorepo

References (cont.)

- https://github.com/salrashid123/go-grpc-bazel-docker
- https://levelup.gitconnected.com/build-and-run-your-first-go-application-with-bazel-ab83acb747f5
- https://github.com/ash2k/bazel-tools
- https://www.xenonstack.com/insights/bazel
- https://dev.to/schoren/create-container-images-with-bazel-47am
- https://wiki.crdb.io/wiki/spaces/CRDB/pages/1380090083/How+to+ensure+your+code+builds+with+Bazel
- https://www.sabre.com/insights/delivering-software-faster-%E2%80%AFis-bazel-the-best-build-tool-for-monorepos/
- https://awesomebazel.com
- https://www.wix.engineering/post/virtual-monorepo-for-bazel
- https://www.stevenengelhardt.com
- https://www.uber.com/blog/how-we-halved-go-monorepo-ci-build-time
- https://github.com/ajbouh/bazel-zig-cc