Builders

Build configs (at the top) and host configs (bottom) are listed here:

https://farmer.golang.org/builders

A builder runs on a certain host type. (e.g. `linux-386-387` is a build type. It runs on `host-linux-kubestd`, a Kubernetes-based linux/amd64 host)

They come from the file https://github.com/golang/build/blob/master/dashboard/builders.go

For design details about the coordinator, see https://go.dev/s/builderplan

How to set up a builder

1. talk to golang-dev@ to get a builder host type & hash (they can get one from using the `golang.org/x/build/cmd/genbuilderkey` tool), and put that in `~/.gobuildkey` or `~/.gobuildkey-host-foo-bar` or the file pointed to by env var `$GO_BUILD_KEY_PATH`.
2. define your new builder in https://github.com/golang/build/blob/master/dashboard/builders.go with a new HostConfig and BuildConfig.
3. have golang-dev deploy the build coordinator rebuilt with the dashboard/builders.go change
4. have golang-dev modify golang.org/x/build/cmd/buildlet/Makefile to add your port and to uploads its buildlet binary to Google Cloud Storage (you can do this step out of order if your compiler changes aren't yet upstream)
5. verify you can see the new host & build configs at https://farmer.golang.org/builders
6. (Interm/testing step) Test that your builder key works and you can register:
    1. `go get -u golang.org/x/build/cmd/buildlet`
    2. `buildlet -coordinator=farmer.golang.org -reverse-type=host-foo-bar -reboot=false`
    3. verify it shows up at https://farmer.golang.org/#pools in "Reverse pool summary" and "Reverse pool machine detail"
7. Modify the golang.org/x/build/cmd/buildlet/stage0 binary if/as needed to pass the right flags to the buildlet binary.
8. Put your stage0 binary on your builder, run in a loop under your operating system's process supervisor (systemd, etc). The stage0 binary is responsible for conditionally re-downloading the buildlet binary from Google Cloud Storage for each build. (This lets us evolve the build system without involving each machine owner)

For WIP ports, the steps above can be done out of order as needed. But as a port matures, be sure each step above is done. In particular, make sure that you're not just running a fixed copy of the buildlet binary in a loop forever. We need to be able to update it over time without your involvement. You should be running the stage0 binary (or equivalent shell script or similar for your platform) in a loop instead.

Builder Requirements

- internet connection (at least be able to access Google and https://farmer.golang.org)
- preferably with two or more (V)CPUs
- at least 512MiB of memory (1GB or more highly recommended. 512MB might need a small `GOGC` setting to avoid thrashing.)

Security notes

Generally, community-run builders only run code that's already been reviewed & submitted. We only enable pre-submit testing for builders run by the Go team that have a lot of hardware available. However, the Gomote tool is available for a number of people on the Go team and in the Go community that lets them have arbitrary access to the builders for development & debugging.

For paranoia reasons, you might want to run your builder in an isolated network that can't access any of your internal resources.