NobleProg

Lab Exercise 1 – C++ Build using Bazel

(Single Target, Single Package)

It's time to build the first part of the project. For a visual reference, the structure of the Stage 1 section of the project is:

```
└──NobleProg1
   ├── main
   │   ├── BUILD
   │   └── hello-world.cc
   └── WORKSPACE
```

Run the following to move to the cpp-tutorial/stage1 directory:

```
cd NobleProg1
```

Next, run:

```
bazel build //main:hello-world
```

In the target label, the //main: part is the location of the BUILD file relative to the root of the workspace, and hello-world is the target name in the BUILD file.

Bazel produces something that looks like this:

```
INFO: Found 1 target...
Target //main:hello-world up-to-date:
  bazel-bin/main/hello-world
INFO: Elapsed time: 2.267s, Critical Path: 0.25s
```

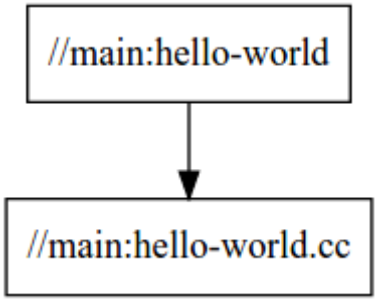
You just built your first Bazel target. Bazel places build outputs in the bazel-bin directory at the root of the workspace.

Now test your freshly built binary, which is:

```
bazel-bin/main/hello-world
```

This results in a printed “Hello world” message.

Here’s the dependency graph of NobleProg1:



Summary: NobleProg1

Now that you have completed your first build, you have a basic idea of how a build is structured. In the next stage, you will add complexity by adding another target.