Compile and install the application

In this last topic, you'll learn a couple new `go` commands. While the `go run` command is a useful shortcut for compiling and running a program when you're making frequent changes, it doesn't generate a binary executable.

This topic introduces two additional commands for building code:

- The `go build` command compiles the packages, along with their dependencies, but it doesn't install the results.
- The `go install` command compiles and installs the packages.

> **Note:** This topic is part of a multi-part tutorial that begins with Create a Go module.

1. From the command line in the hello directory, run the `go build` command to compile the code into an executable.

   ```
   $ go build
   ```

2. From the command line in the hello directory, run the new `hello` executable to confirm that the code works.

   Note that your result might differ depending on whether you changed your greetings.go code after testing it.

   - On Linux or Mac:

     ```
     $ ./hello
     map[Darrin:Great to see you, Darrin! Gladys:Hail, Gladys! Well met! Samantha:Hail, Samantha! Well met!]
     ```

   - On Windows:

     ```
     $ hello.exe
     map[Darrin:Great to see you, Darrin! Gladys:Hail, Gladys! Well met! Samantha:Hail, Samantha! Well met!]
     ```

   You've compiled the application into an executable so you can run it. But to run it currently, your prompt needs either to be in the executable's directory, or to specify the executable's path.

   Next, you'll install the executable so you can run it without specifying its path.

3. Discover the Go install path, where the `go` command will install the current package.

   You can discover the install path by running the `go list` command, as in the following example:

   ```
   $ go list -f '{{.Target}}'
   ```

   For example, the command's output might say `/home/gopher/bin/hello`, meaning that binaries are installed to /home/gopher/bin. You'll need this install directory in the next step.

4. Add the Go install directory to your system's shell path.

   That way, you'll be able to run your program's executable without specifying where the executable is.

   - On Linux or Mac, run the following command:

     ```
     $ export PATH=$PATH:/path/to/your/install/directory
     ```

   - On Windows, run the following command:

     ```
     $ set PATH=%PATH%;C:\path\to\your\install\directory
     ```

   As an alternative, if you already have a directory like `$HOME/bin` in your shell path and you'd like to install your Go programs there, you can change the install target by setting the `GOBIN` variable using the `go env` command:

   ```
   $ go env -w GOBIN=/path/to/your/bin
   ```

   or

   ```
   $ go env -w GOBIN=C:\path\to\your\bin
   ```

5. Once you've updated the shell path, run the `go install` command to compile and install the package.

   ```
   $ go install
   ```

6. Run your application by simply typing its name. To make this interesting, open a new command prompt and run the `hello` executable name in some other directory.

   ```
   $ hello
   map[Darrin:Hail, Darrin! Well met! Gladys:Great to see you, Gladys! Samantha:Hail, Samantha! Well met!]
   ```

That wraps up this Go tutorial!

< Add a test

Conclusion and links to more information >