Cloud Functions in Go must provide all of their dependencies via either Go modules with a go.mod file, or a vendor directory. Your function cannot specify dependencies using both Go modules and a vendor directory at the same time.

The Go runtime includes a number of <u>system packages</u> (/functions/docs/reference/go-system-packages) in the execution environment. If your function uses a dependency that requires a system package that is not listed, you can <u>request a package</u>

(/functions/docs/reference/go-system-packages#requesting_a_package).

The content in this document applies to both Go 1.11 and Go 1.13.

Cloud Functions support Go's experimental Modules functionality

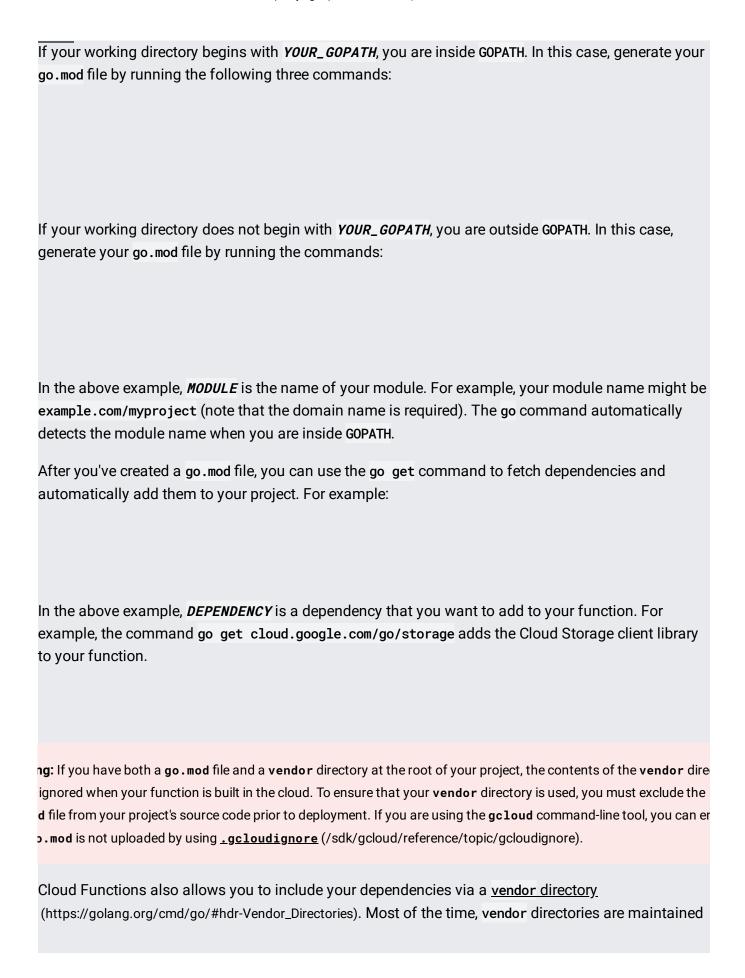
(https://golang.org/doc/go1.11#modules), which enables you to specify dependencies in a go.mod file at the root of your project. When you deploy your function, dependencies specified in the go.mod file will be fetched and built automatically.

The behavior of Go modules differs depending on whether you are developing inside or outside of GOPATH. To determine whether you are inside GOPATH:

- 1. Navigate to your project directory.
- 2. Find your GOPATH by running the command:

This outputs a line similar to:

3. Find your current working directory by running:



with a dependency manager. You can use any dependency manager you like. For example, you can use Go's Modules functionality to create a vendor directory from your go.mod file.

If you have a <code>go.mod</code> file and a <code>vendor</code> directory, the <code>vendor</code> directory will be ignored when you deploy your function. You can use a <code>.gcloudignore</code> (/sdk/gcloud/reference/topic/gcloudignore) file to avoid uploading your <code>go.mod</code> and <code>go.sum</code> files, in which case the contents of your <code>vendor</code> directory will be respected:

1. Create a .gcloudignore file at the root of your project directory with the following contents:

2. Create a vendor directory using the contents of your go.mod file by running the following command:

If your function's dependencies are hosted in a repository that is not publicly accessible, you must use a vendor directory to fetch your dependencies before deploying your function. If you plan to use a go.mod file, see the instructions above to avoid conflicts between the go.mod file and the vendor directory.