Chapter-13: Building Projects.

modules: haskell programs are organized into modules.

Modules contain the datatypes, type synanyms, type classes,

type class instances, and values we define at the top level.

They offer a means to import other modules unto the Scope of our program, and they also centain values that can be exported to other modules.

Primary focus for this chapter is to unduratand how to setup a project in Haskell, use the pkg manager known as Cabal, build the project with Stack, and work with Haskell modules as they are.

making pkgs with stack: The Haskell cabal, or Common Architecture for Building Applications of libraries, is a pkg mgs. A pkg. is a program we're building, including all its modules of dependencies,

A pkg, has dependencies which are the interlinked elements of that program, i.e. the other libraries and pkgs, it may depend on along with any tests and documentation associated with the propect.

cabal exists to help organize all this and make some all dependencies are properly in scope.

Stack is a cross platform program for developing trastell projects, as it helps us manage projects with multiple as well as individual pkgs, whereas Cabal exists primarily to describe a single pkg. with a Cabal file that has a cabal file extension.

1

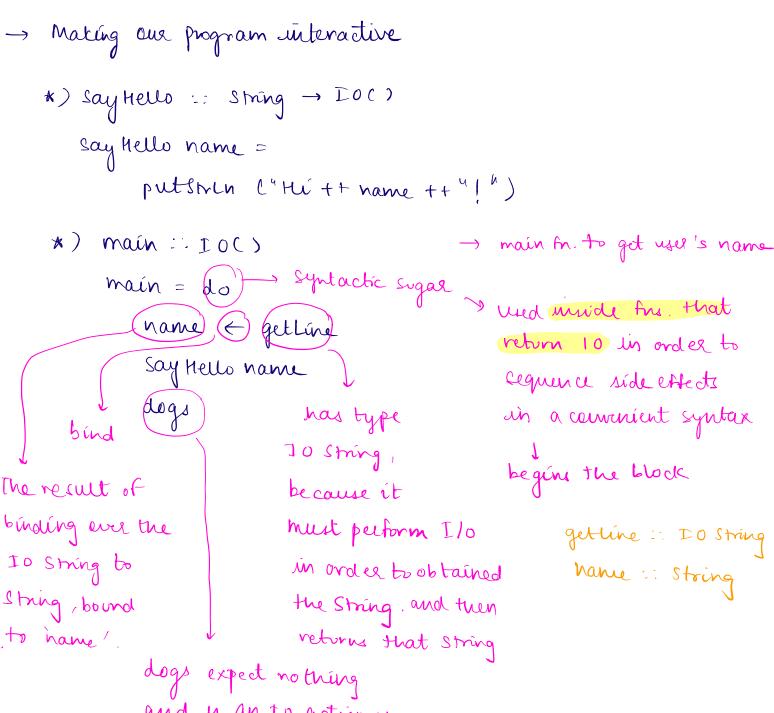
Stack is built on top of Cabal in some imp. sense, hence we would Still be working with cabal files. However, stack simplifies the process somewhat, especially in large projects with multiple dependences, by allowing us to build those large libraries only once and use them across projects.

l

Stack also relies on LTS snapshot of Hasken pkgs from Stackage, that are guaranteed to work together, unlike pkgs from Mackage which may have conflicting dependencies.

The basic structure embodied in Stack templates is the recommended project layout.

- -> Working with a basic project:
  - r) building the project: i) stack build



and is an 10 action of type Io(s, which fits the wealt type of main.

