Functional programming, Seminar No. 1

Danya Rogozin
Lomonosov Moscow State University,
Serokell OÜ

Higher School of Economics
Faculty of Computer Science

General words on Haskell

- The language is named after American logician Haskell Curry
- First implementation: 1990
- The language standard: Haskell2010
- Default compiler: Glasgow Haskell compiler
- Haskell is a strongly-typed, polymorphic, and purely functional programming language

The Haskell Platform installation

There are several ways to install the Haskell platform on Mac:

- Download the `.pkg` file and install the corresponding package
- Run the script `curl -sSL https://get.haskellstack.org/ | sh`
- Install ghc, stack, and cabal via Homebrew

Choose any way you prefer. All those ways are equivalence to each other.

I'm a Mac user, but I believe that you'll manage to install the Haskell Platform on NixOs/Windows/Linux/etc quite quickly.

GHC

- GHC is a default Haskell compiler as we told above
- GHC is an open-source project. Don't hesistate to contribute!
- GHC is mostly implemented on Haskell
- GHC development is produced under the GHC Steering committee control
- Very roughly, compiling pipeline is arranged as follows:
  parsing $\Rightarrow$ compile-time (type-checking mostly) $\Rightarrow$ runtime (program execution)

GHCi

- GHCi is a Haskell interpreter based on GHC
- One may run GHCi with a quite simple command `ghci` on a shell
- You play with GHCi as a calculator, the ordinary arithmetical operators are written in a usual way
- Take a look at the GHCi chapter in the GHC User's Guide to be familiar with GHCi closely

Cabal

- Cabal is a system of library and dependency management
- A `.cabal` file describe the version of a package and its dependencies
- Cabal is also a packaging tool
- Cabal is known as a reason of so-called dependency hell

Stack

- Stack is a cross-platform build tool for Haskell projects
- Stack allows one to
  - install packages and version of GHC (and their concrete versions) you need
  - build, execute, and test projects
  - reproduce builds
  - create an isolated location

Snapshots

- Snapshot is a curated package set used by Stack
- Stackage is a stable repository that stores snapshots
- Resolver is a reference to a required snapshot
- Let us take a look at the screenshot from Stackage:
  TODO: Snapshots.jpg

Ecosystem encapsulation

The Haskell ecosystem encapsulation might be described as the following sequence:
TODO: visualise this story somehow

Creating a Haskell project via Stack

- Figure out how to call your project and run the script `stack new <projectname>`
- You will see the following story after the command `tree .` in the project directory:
  TODO: Tree.jpg

stack.yaml

Let us discuss dependencies files in a Haskell project. First of all, we observe the `stack.yaml` file:
TODO: StackYaml.jpg

Cabal file

As we told above, the `.cabal` file describe the relevant version of a project and its dependencies:
TODO: Cabal.jpg

package.yaml

The `package.yaml` generates automatically from the `stack.yaml` and `.cabal` files:
TODO: PackageYaml.jpg

Building and running a project

The following commands are crucially important:

- `stack build`
- `stack run`
- `stack exec`
- `stack ghci`
- `stack clean`

Their roles follow from names which are quite self-explanatory.

Hackage

According to its description, 'Hackage is the Haskell community's central package archive of open source software'.

- Webpage: `https://hackage.haskell.org`
- Browsing packages
- Simplified package search
- Current uploads

TODO: screenshot

Hoogle

Hoogle is a sort of Haskell search engine. Webpage: `https://hoogle.haskell.org`.
TODO: Hoogle.jpg

Summary

We observed today such topics as

1. General aspects of GHC and GHCi
2. The Haskell Platform installation
3. Dependency management via Stack and Cabal

On the next seminar, we will discuss:

1. The basic Haskell syntax
2. The underlying aspects of the Haskell type system
3. Functions and lambdas
4. Immutability and Laziness