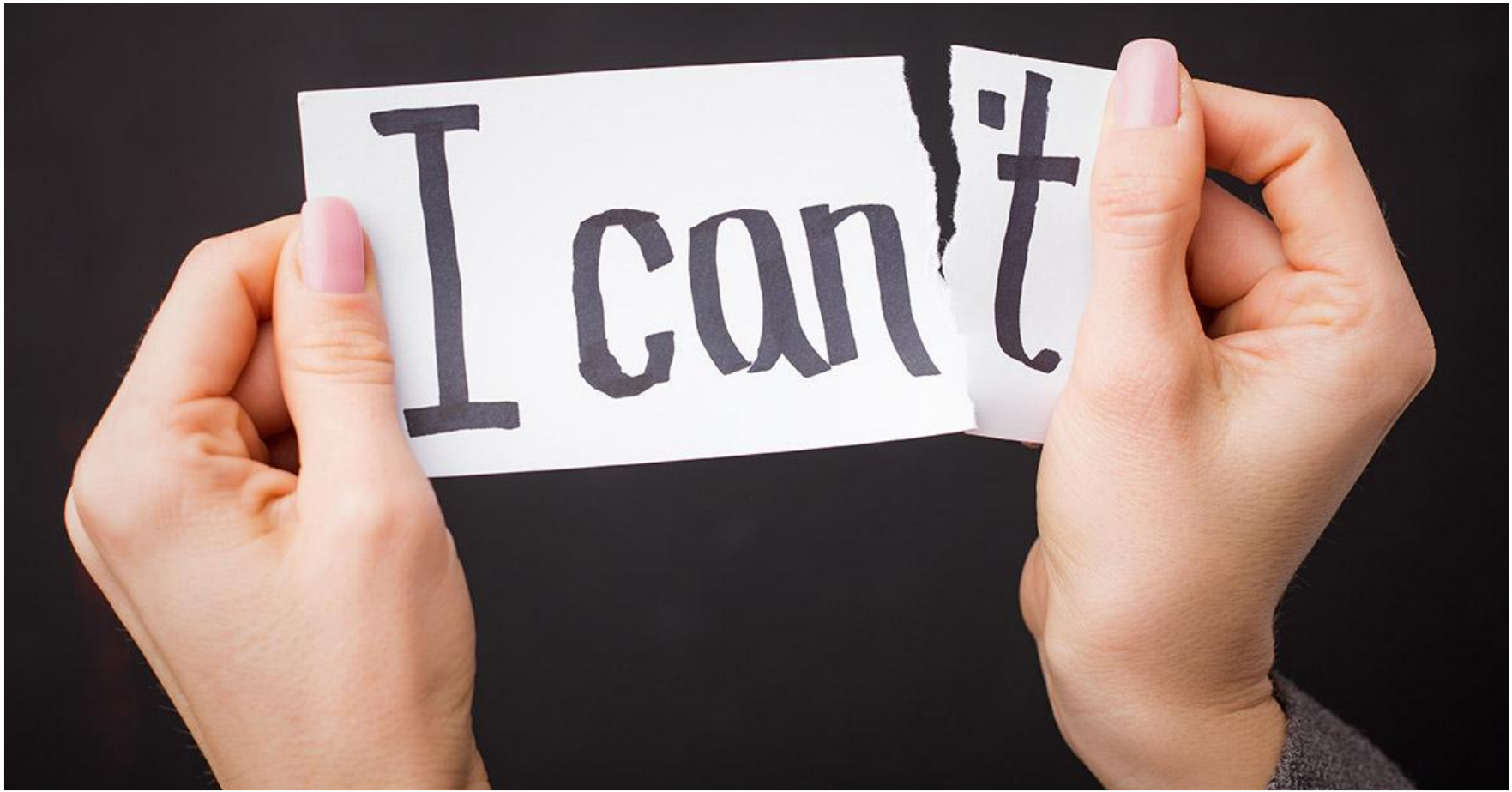
Introduction to Haskell and its REPL


I can't

1.

```
$ docker run -it --rm \
    mitchty/alpine-ghc ghci
```

1.

```
$ docker run -it --rm \
    mitchty/alpine-ghc ghci
```

docker pull mitchty/alpine-ghc:latest

So... what now?

- 1 + 2
- 2^1000
- (+) 1 2
- :t (+)
- :i (+)

Functions Everywhere

- (+) :: a -> a -> a

```
In [1]: def add2Integers(a, b):
   ...:     return a + b
   ...:

In [2]: add2Integers(1, 2)
Out[2]: 3
```

List

- [1, 2, 3]
- [1..]
- [1..10]
- [1, 1.25 .. 4.0]

List

- (:) :: a -> [a] -> [a]
- (++) :: [a] -> [a] -> [a]

List

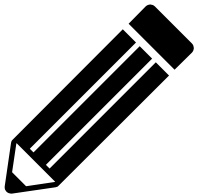
- head :: [a] -> a
- tail :: [a] -> [a]
- take :: Int -> [a] -> [a]
- drop :: Int -> [a] -> [a]

List

- map :: (a -> b) -> [a] -> [b]
- filter :: (a -> Bool) -> [a] -> [a]

GHCi

- :t <espression> ← type inspection
- :i <espression> ← info
- :l <filename> ← load .hs file
- :r ← reload files



BYO Editor

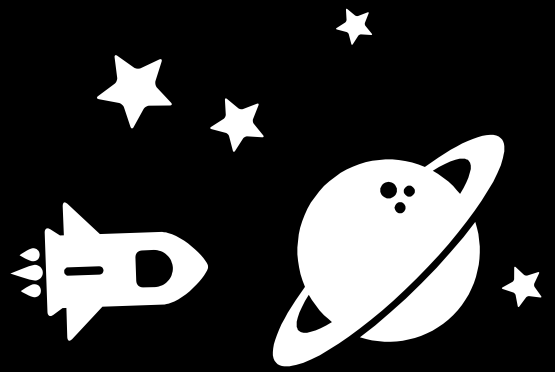
- Spacemacs http://spacemacs.org/
- Atom https://atom.io/
- Visual Studio https://code.visualstudio.com/

2.

```
$ docker run -it --rm \
    -v `pwd`:/root \
    mitchty/alpine-ghc ghci
```

Wanna try an exercise together?

Let’s reverse a list



Theorems for free!

– Philip Walder, 1989
https://people.mpi-sws.org/~dreyer/tor/papers/wadler.pdf

What’s the only possible implementation of `foo`?

- foo :: a -> a
- foo2 :: b -> a
- foo3 :: a -> b -> a


A journey of a thousand miles
BEGINS WITH
A SINGLE STEP.
Lao Tzu

Thanks!

Any questions?

@filippovitale