ThriftUsageHaskell

Simple networked client-server example program:

Compile and install Thrift

Don't have full requirements for compiling Thrift off the top of my head; it requires Network, Binary and possibly some other things.

Create a thrift file

test.thrift

```
namespace hs test

enum Operation {
  ADD = 1,
  SUBTRACT = 2,
  MULTIPLY = 3,
  DIVIDE = 4
}

struct Work {
  1: i32 num1 = 0,
  2: i32 num2,
  3: Operation op,
  4: optional string comment,
}
```

Compile the thrift file

```
$ thrift --gen hs test.thrift
```

Write your Haskell program

Requirements

File test.hs

```
module Main where

import Data.List
import IO
import Network
import System (getArgs)

-- Thrift libraries
import Thrift
import Thrift.Transport.Handle
import Thrift.Protocol
import Thrift.Protocol.Binary
import Thrift.Server

-- Generated Thrift modules
import Test_Types
```

Constants

```
port :: PortNumber
port = 4390

testdata :: Work
testdata = Work {
  f_Work_num1 = Just 1,
  f_Work_num2 = Just 2,
  f_Work_op = Just ADD,
  f_Work_comment = Just "Foo!"
  }

testdata2 :: Work
testdata2 = Work {
  f_Work_num1 = Just 10,
  f_Work_num2 = Just 20,
  f_Work_op = Just SUBTRACT,
  f_Work_comment = Just "Bar!"
  }
```

Functions

```
serverFunc :: a -> (BinaryProtocol Handle, BinaryProtocol Handle)
              -> IO Bool
serverFunc a (h1,h2) = do
  let t1 = getTransport h1
  let t2 = getTransport h2

  putStrLn "Server go!"
  dat <- read_Work h1
  putStrLn "Recieved data:"
  print dat
  write_Work h1 testdata2
  tFlush t1
  putStrLn "Data written"

  return False

clientFunc :: HostName -> PortNumber -> IO ()
clientFunc host p = do
  putStrLn "Client go!"
  h <- connectTo host $ PortNumber p
  let proto = BinaryProtocol h
  write_Work proto testdata
  tFlush h
  putStrLn "Data sent, receiving."
  w <- read_Work proto
  putStrLn "Recieved:"
  print w
  tClose h

main :: IO ()
main = do
  a <- getArgs
  if elem "client" a then do clientFunc "127.0.0.1" port
    else do
    runBasicServer () serverFunc port
    putStrLn "Server stopped"
```

Compile your Haskell program

```
$ ghc --make gen-hs/*.hs test.hs
```

Run it

```
# Start server
host1$ ./test

# Run client
host2$ ./test client
```

Issues with this

1. In serverFunc, whether you read/write to/from h1 or h2 does not seem to matter. What's up with that?
2. Does not demonstrate implementing services
3. Does not demonstrate maps, constants, etc...
4. runBasicServer listens on IPv4 only
5. Compiling Thrift is a little painful and could use more explanation; also, the version in Hackage is 0.5.0 and the current (and version used here) is 0.6.0.