Haskell Testing Demo

Josh Cough

https://github.com/joshcough/HaskellTestingDemo

Wat?

TestFramework

- HUnit
- QuickCheck

Tasty

- HUnit
- QuickCheck
- SmallCheck

Test Framework HUnit

```
import Test.Framework.Providers.API
import Test.Framework.Providers.HUnit
import Test.Framework.Runners.Console
import Test.HUnit

main = defaultMain $ [testGroup "Main" tests]
tests = [
  testCase "a passing test!" $ 5 @?= 5
 ,testCase "a failing test!" $ 5 @?= 6 ]
```

Test Framework HUnit Cabal

```
-- HUnit tests
test-suite unit-tests
  type:           exitcode-stdio-1.0
  main-is:        UnitTestsMain.hs
  hs-source-dirs: .
  build-depends:
    base,
    HUnit,
    test-framework,
    test-framework-hunit
```

Test Framework QuickCheck

```haskell
{-# LANGUAGE TemplateHaskell #-}
import Test.Framework.Providers.QuickCheck2
import Test.Framework.Runners.Console
import Test.Framework.TH
import Test.QuickCheck

main = defaultMain [tests]
prop_list_reverse_reverse :: [Int] -> Bool
prop_list_reverse_reverse list = list == reverse (reverse list)
tests = $testGroupGenerator
```

Test Framework QuickCheck Cabal

```
-- QuickCheck tests
test-suite properties
  type:           exitcode-stdio-1.0
  main-is:        PropertiesMain.hs
  hs-source-dirs: .
  build-depends:
    base,
    QuickCheck                >= 2.4,
    test-framework            >= 0.6,
    test-framework-quickcheck2 >= 0.2,
    test-framework-th         >= 0.2
```

Test Framework Everything

```haskell
{-# LANGUAGE TemplateHaskell #-}
import Test.Framework.Providers.API
import Test.Framework.Providers.HUnit
import Test.Framework.Providers.QuickCheck2
import Test.Framework.Runners.Console
import Test.Framework.TH
import Test.HUnit
import Test.QuickCheck

main = defaultMain $ [unitTests, props]

unitTests = testGroup "HUnit tests" [
  testCase "a passing test!" $ 5 @?= 5 ,testCase "a failing test!" $ 5 @?= 6 ]

prop_list_reverse_reverse :: [Int] -> Bool
prop_list_reverse_reverse list = list == reverse (reverse list)
props = $testGroupGenerator
```

Test Framework Everything Cabal

```
-- QuickCheck tests
test-suite properties
  type:           exitcode-stdio-1.0
  main-is:        PropertiesMain.hs
  hs-source-dirs: .
  build-depends:
    base,
    HUnit,
    QuickCheck                 >= 2.4,
    test-framework             >= 0.6,
    test-framework-hunit,
    test-framework-quickcheck2 >= 0.2,
    test-framework-th          >= 0.2
```

Tasty HUnit

```
import Data.List
import Test.Tasty
import Test.Tasty.HUnit

main = defaultMain unitTests

unitTests = testGroup "Unit tests"
  [ testCase "List comparison (different length)" $
      [1, 2, 3] `compare` [1,2] @?= GT ]
```

Tasty QuickCheck

```
import Data.List
import Test.Tasty
import Test.Tasty.QuickCheck as QC

main = defaultMain qcProps

qcProps = testGroup "(checked by QuickCheck)"
  [ QC.testProperty "sort == sort . reverse" $
      \list -> sort (list :: [Int]) == sort (reverse list) ]
```

Tasty SmallCheck

```
import Data.List
import Test.Tasty
import Test.Tasty.SmallCheck as SC

main = defaultMain scProps

scProps = testGroup "(checked by SmallCheck)"
  [ SC.testProperty "sort == sort . reverse" $
      \list -> sort (list :: [Int]) == sort (reverse list) ]
```

Tasty Everything

```
import Data.List
import Test.Tasty
import Test.Tasty.SmallCheck as SC
import Test.Tasty.QuickCheck as QC
import Test.Tasty.HUnit

main = defaultMain $ testGroup "Tests" [properties, unitTests]
properties = testGroup "Properties" [scProps, qcProps]

scProps = testGroup "(checked by SmallCheck)"
  [ SC.testProperty "sort == sort . reverse" $ \list -> sort (list :: [Int]) == sort (reverse list) ]

qcProps = testGroup "(checked by QuickCheck)"
  [ QC.testProperty "sort == sort . reverse" $ \list -> sort (list :: [Int]) == sort (reverse list) ]

unitTests = testGroup "Unit tests"
  [ testCase "List comparison (different length)"$ [1, 2, 3] `compare` [1,2] @?= GT ]
```

Tasty Cabal

```
-- Tasty Tests
test-suite tasty
  type:           exitcode-stdio-1.0
  main-is:        TastyMain.hs
  hs-source-dirs: .
  build-depends:
    base,
    tasty >= 0.5.2,
    tasty-hunit,
    tasty-quickcheck,
    tasty-smallcheck
```

Executing Tests

```
cabal install --enable-tests
```

That’s it!