Линзы / Lenses

Роман Чепляка

21 июля 2012
Kiev::fprog

Haskell's record system

```haskell
data TimeOfDay = TimeOfDay {
    todHour :: Int,
    todMin  :: Int,
    todSec  :: Pico
}

getHour :: TimeOfDay -> Int
getHour = todHour

setHour :: Int -> TimeOfDay -> TimeOfDay
setHour hour time = time { todHour = hour }
```

Haskell's record system

```haskell
data TimeOfDay = TimeOfDay {
    todHour :: Int,
    todMin  :: Int,
    todSec  :: Pico
}

data LocalTime = LocalTime {
    localDay       :: Day,
    localTimeOfDay :: TimeOfDay
}

data ZonedTime = ZonedTime {
    zonedTimeToLocalTime :: LocalTime,
    zonedTimeZone :: TimeZone
}
```

Haskell's record system

```haskell
getHour :: ZonedTime -> Int
getHour =
  todHour . localTimeOfDay . zonedTimeToLocalTime
```

Haskell's record system

```haskell
setHour :: Int -> ZonedTime -> ZonedTime
setHour hour zonedTime = zonedTime'
  where
    zonedTime' =
      zonedTime { zonedTimeToLocalTime = localTime' }
    localTime' =
      localTime { localTimeOfDay = timeOfDay' }
    timeOfDay' =
      timeOfDay { todHour = hour }

    localTime = zonedTimeToLocalTime zonedTime
    timeOfDay = localTimeOfDay localTime
```

Lenses

```haskell
data Lens a b = Lens (a -> b) (b -> a -> a)

(.) :: Lens b c -> Lens a b -> Lens a c
Lens get1 set1 . Lens get2 set2 = Lens get set
  where
    get s = get1 $ get2 s
    set n s = set2 (set1 n $ get2 s) s
```

Operations

```
getL :: Lens a b -> (a -> b)
setL :: Lens a b -> (b -> a -> a)
modL :: Lens a b -> ((b -> b) -> (a -> a))
```

Operations

```haskell
getHour :: ZonedTime -> Int
getHour = getL $
  todHour . localTimeOfDay . zonedTimeToLocalTime

setHour :: Int -> ZonedTime -> ZonedTime
setHour = setL $
  todHour . localTimeOfDay . zonedTimeToLocalTime
```

Implementation and features

- Рекомендуемая реализация: data-lens (Russell O'Connor, Edward Kmett, Tony Morris)

Implementation and features

- Рекомендуемая реализация: data-lens (Russell O'Connor, Edward Kmett, Tony Morris)
- deriving

Implementation and features

- Рекомендуемая реализация: data-lens (Russell O'Connor, Edward Kmett, Tony Morris)
- deriving
- State monad

Implementation and features

- Рекомендуемая реализация: data-lens (Russell O'Connor, Edward Kmett, Tony Morris)
- deriving
- State monad
- time-lens

Ad #1

Haskell в Севастополе

Вакансия на jobs.dou.ua

Ad #2

Haskell в Одессе

Odessa Haskell user group
odhug @ google groups