

A (Very) Tiny Taste of Vinyl

Jon Sterling
FOBO

May 17, 2014

Validating Records

Validating Records

bob :: Rec [Name, Email Work]

Validating Records

bob :: Rec [Name, Email Work]

bob = Name =: "Robert_Harper"

⊕ Email Work =: "rwh@cs.cmu.edu"

Validating Records

bob :: Rec [Name, Email Work]

bob = Name =: "Robert_LHarper"

⊕ Email Work =: "rwh@cs.cmu.edu"

validateName :: **String** → **Either** Error **String**

validateEmail :: **String** → **Either** Error **String**

validatePhone :: [N] → **Either** Error [N]

Validating Records

bob :: Rec [Name, Email Work]

bob = Name =: "Robert_LHarper"

⊕ Email Work =: "rwh@cs.cmu.edu"

validateName :: **String** → **Either** Error **String**

validateEmail :: **String** → **Either** Error **String**

validatePhone :: [N] → **Either** Error [N]

unnnnnnhhh...

Validating Records

bob :: Rec [Name, Email Work]

bob = Name =: "Robert_LHarper"

⊕ Email Work =: "rwh@cs.cmu.edu"

validateName :: **String** → **Either** Error **String**

validateEmail :: **String** → **Either** Error **String**

validatePhone :: [N] → **Either** Error [N]

unnnnnnhhh...

validateContact

:: Rec [Name, Email Work]

→ **Either** Error (Rec [Name, Email Work])

Welp.

Compositional Validation

bob :: $\text{Rec}_{\mathcal{A}}$ [Name, Email Work]

Compositional Validation

bob :: $\text{Rec}_{\mathcal{A}}$ [Name, Email Work]

bob = Name =: "Robert_Harper"

⊕ Email Work =: "rwh@cs.cmu.edu"

Compositional Validation

bob :: $\text{Rec}_{\mathcal{A}}$ [Name, Email Work]

bob = Name =: "Robert_Harper"

⊕ Email Work =: "rwh@cs.cmu.edu"

Compositional Validation

type Validator a = a \rightarrow **Either** Error a

Compositional Validation

```
type Validator a = a → Either Error a  
validateName :: RecA Validator '[Name]  
validatePhone :: ∀ℓ. RecA Validator '[Phone ℓ]  
validateEmail :: ∀ℓ. RecA Validator '[Email ℓ]
```

Compositional Validation

```
type Validator a = a → Either Error a  
validateName :: RecA Validator '[Name]  
validatePhone :: ∀ℓ. RecA Validator '[Phone ℓ]  
validateEmail :: ∀ℓ. RecA Validator '[Email ℓ]
```

```
type TotalContact =  
  [ Name, Email Home, Email Work  
    , Phone Home, Phone Work ]
```

Compositional Validation

type Validator a = a \rightarrow **Either** Error a
validateName :: Rec_A Validator '[Name]
validatePhone :: $\forall \ell$. Rec_A Validator '[Phone ℓ]
validateEmail :: $\forall \ell$. Rec_A Validator '[Email ℓ]

type TotalContact =
[Name, Email Home, Email Work
 , Phone Home, Phone Work]

validateContact :: Rec_A Validator TotalContact
validateContact = validateName
 \oplus validateEmail
 \oplus validateEmail
 \oplus validatePhone
 \oplus validatePhone

Record Operators

Record Operators

```
newtype Lift o f g x = Lift { runLift :: f x 'o' g x }
```

Record Operators

```
newtype Lift o f g x = Lift { runLift :: f x 'o' g x }
```

```
type Validator = Lift (→) Identity (Either Error)
```

Record Operators

newtype Lift o f g x = Lift { runLift :: f x 'o' g x }

type Validator = Lift (→) Identity (**Either** Error)

(\odot) :: Rec_U (Lift (→) f g) rs → Rec_U f rs → Rec_U g rs

Record Operators

newtype Lift o f g x = Lift { runLift :: f x 'o' g x }

type Validator = Lift (→) Identity (**Either** Error)

(\odot) :: $\text{Rec}_{\mathcal{U}}$ (Lift (→) f g) rs → $\text{Rec}_{\mathcal{U}}$ f rs → $\text{Rec}_{\mathcal{U}}$ g rs

$\text{rdist} :: \text{Applicative } f \Rightarrow \text{Rec}_{\mathcal{U}} f rs \rightarrow f (\text{Rec}_{\mathcal{U}} \text{ Identity } rs)$

Compositional Validation

```
newtype Lift o f g x = Lift { runLift :: f x 'o' g x }  
type Validator = Lift (→) Identity (Either Error)  
(⊛) :: RecU (Lift (→) f g) rs → RecU f rs → RecU g rs  
rdist :: Applicative f ⇒ RecU f rs → f (RecU Identity rs)
```

Compositional Validation

```
newtype Lift o f g x = Lift { runLift :: f x 'o' g x }  
type Validator = Lift (→) Identity (Either Error)  
(⊙) :: RecU (Lift (→) f g) rs → RecU f rs → RecU g rs  
rdist :: Applicative f ⇒ RecU f rs → f (RecU Identity rs)  
  
validateContact :: RecA Validator TotalContact
```

Compositional Validation

```
newtype Lift o f g x = Lift { runLift :: f x 'o' g x }  
type Validator = Lift (→) Identity (Either Error)  
(⊙) :: RecU (Lift (→) f g) rs → RecU f rs → RecU g rs  
rdist :: Applicative f ⇒ RecU f rs → f (RecU Identity rs)  
  
validateContact :: RecA Validator TotalContact  
  
bobValid :: RecA (Either Error) [Name, Email Work]
```

Compositional Validation

```
newtype Lift o f g x = Lift { runLift :: f x 'o' g x }  
type Validator = Lift (→) Identity (Either Error)  
(⊛) :: RecU (Lift (→) f g) rs → RecU f rs → RecU g rs  
rdist :: Applicative f ⇒ RecU f rs → f (RecU Identity rs)
```

```
validateContact :: RecA Validator TotalContact
```

```
bobValid :: RecA (Either Error) [Name, Email Work]  
bobValid = cast validateContact ⊛ bob
```


Compositional Validation

```
newtype Lift o f g x = Lift { runLift :: f x 'o' g x }  
type Validator = Lift (→) Identity (Either Error)  
(⊛) :: RecU (Lift (→) f g) rs → RecU f rs → RecU g rs  
rdist :: Applicative f ⇒ RecU f rs → f (RecU Identity rs)
```

```
validateContact :: RecA Validator TotalContact
```

```
bobValid :: RecA (Either Error) [Name, Email Work]  
bobValid = cast validateContact ⊛ bob
```

```
validBob :: Either Error (RecA Identity [Name, Email Work])
```

Compositional Validation

```
newtype Lift o f g x = Lift { runLift :: f x 'o' g x }  
type Validator = Lift (→) Identity (Either Error)  
(⊛) :: RecU (Lift (→) f g) rs → RecU f rs → RecU g rs  
rdist :: Applicative f ⇒ RecU f rs → f (RecU Identity rs)
```

```
validateContact :: RecA Validator TotalContact
```

```
bobValid :: RecA (Either Error) [Name, Email Work]  
bobValid = cast validateContact ⊛ bob
```

```
validBob :: Either Error (RecA Identity [Name, Email Work])  
validBob = rdist bobValid
```