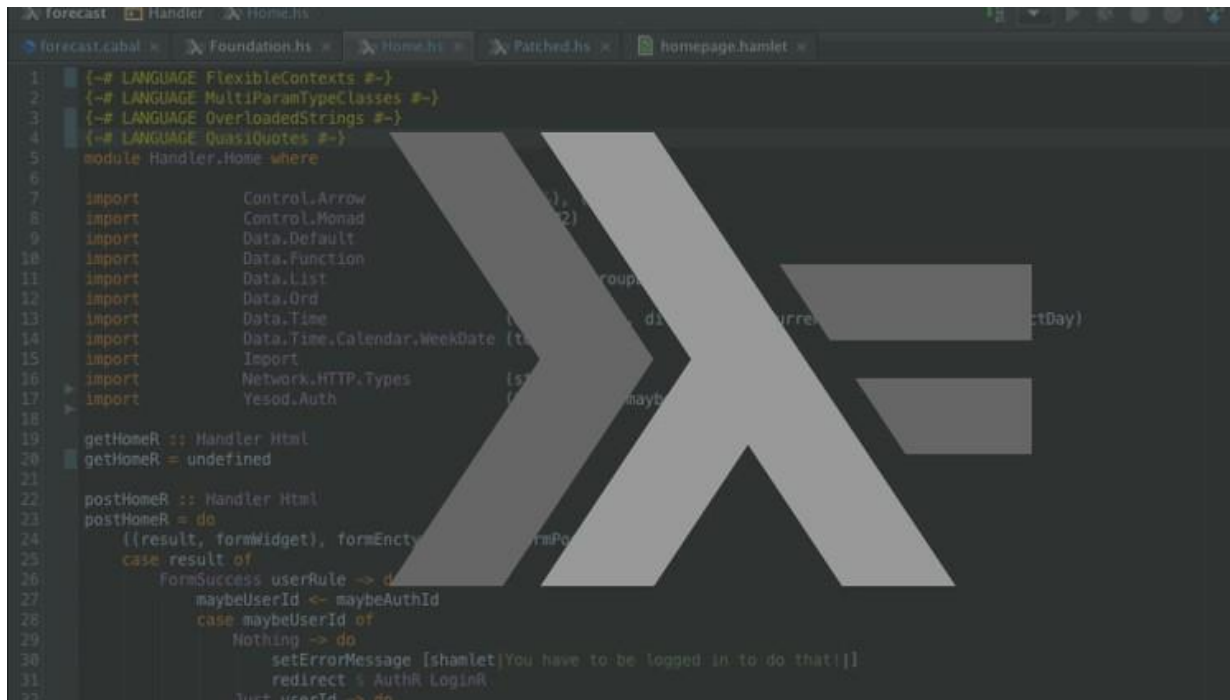


Implementation of <*>



```
1 {-# LANGUAGE FlexibleContexts #-}
2 {-# LANGUAGE MultiParamTypeClasses #-}
3 {-# LANGUAGE OverloadedStrings #-}
4 {-# LANGUAGE QuasiQuotes #-}
5 module Handler.Home where
6
7 import Control.Arrow
8 import Control.Monad
9 import Data.Default
10 import Data.Function
11 import Data.List
12 import Data.Ord
13 import Data.Time
14 import Data.Time.Calendar.WeekDate (toDayOfWeek)
15 import Import
16 import Network.HTTP.Types
17 import Yesod.Auth
18
19 getHomeR :: Handler Html
20 getHomeR = undefined
21
22 postHomeR :: Handler Html
23 postHomeR = do
24   ((result, formWidget), formEnctype) <- runFormPost
25   case result of
26     FormSuccess userRule -> do
27       maybeUserId <- maybeAuthId
28       case maybeUserId of
29         Nothing -> do
30           setErrorMessage [shamlet|You have to be logged in to do that!|]
31           redirect % AuthR.LoginR
32         Just userId -> do
```

Applicatives



We already know how to apply functions:

```
λ> (+3) 2
```

👉 5

And we know how to do it on containers:

```
λ> fmap (+3) (Just 2)
```

👉 Just 5

But what if the function is in a container?

```
λ> (Just (+3)) (Just 2)
```

❌

In this case, we can use `<*>`! (it is read *app*)

```
λ> Just (+3) <*> Just 2
λ> Just (+3) <*> Nothing
λ> Nothing <*> Just (+3)
λ> Nothing <*> Nothing
```

👉 Just 5
👉 Nothing
👉 Nothing
👉 Nothing

```
λ> Right (+3) <*> Right 2
λ> Right (+3) <*> Left "err"
λ> Left "err" <*> Right 2
λ> Left "err1" <*> Left "err2"
```

👉 Right 5
👉 Left "err"
👉 Left "err"
👉 Left "err1 "

```
λ> [(+2), (+2)] <*> [1, 2, 3]
```

👉 [2, 4, 6, 3, 4, 5]

Implementation of $\langle * \rangle$



The operator $\langle * \rangle$ is an operation of the class `Applicative` (which must also be functor):

```
class Functor f => Applicative f where
  (<*>) :: f (a -> b) -> (f a -> f b)
  pure  :: a -> f a
```

- $\langle * \rangle$ applies a function inside a container to values inside a container. Containers are generic and of the same type.
- `pure` constructs a container with a value.

