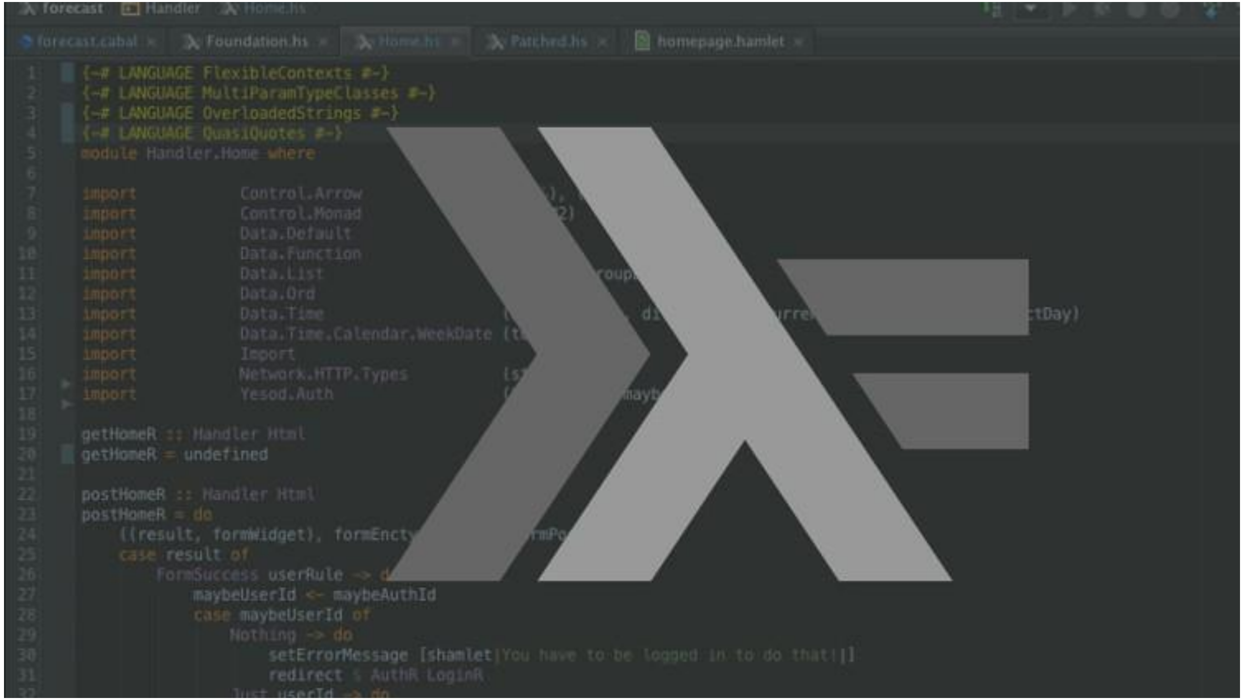
Fundamentals of Binary Trees



```
{-# LANGUAGE FlexibleContexts #-}
{-# LANGUAGE MultiParamTypeClasses #-}
{-# LANGUAGE OverloadedStrings #-}
{-# LANGUAGE QuasiQuotes #-}
module Handler.Home where

import           Control.Arrow                  ), (
import           Control.Monad                  2)
import           Data.Default
import           Data.Function
import           Data.List                      roup
import           Data.Ord
import           Data.Time                      (      , di      urre               ctDay)
import           Data.Time.Calendar.WeekDate (t
import           Import
import           Network.HTTP.Types             (s
import           Yesod.Auth                     (             mayb

getHomeR :: Handler Html
getHomeR = undefined

postHomeR :: Handler Html
postHomeR = do
    ((result, formWidget), formEncty          rmPo
    case result of
        FormSuccess userRule -> d
            maybeUserId <- maybeAuthId
            case maybeUserId of
                Nothing -> do
                    setErrorMessage [shamlet|You have to be logged in to do that!|]
                    redirect $ AuthR LoginR
                Just userId -> do
```

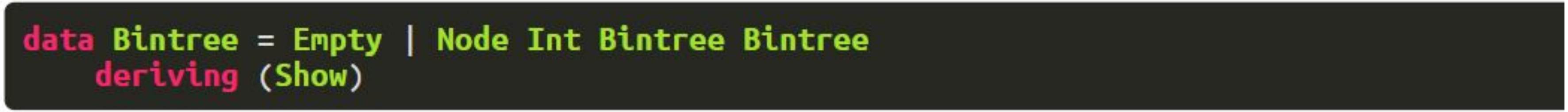
Lucas Bazilio - Udemy

Fundamentals of Binary Trees

Binary Tree: data structure in which each node can have a left child and a right child. They cannot have more than two children.

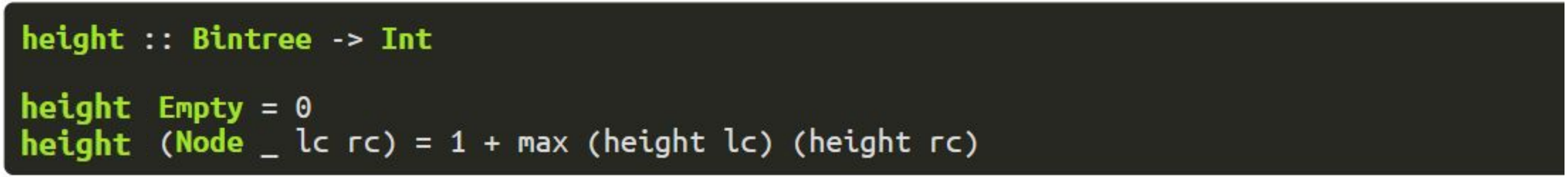
Common **uses** of binary trees are binary search trees, binary heaps, and Huffman coding.

Fundamentals of Binary Trees



```
data Bintree = Empty | Node Int Bintree Bintree
    deriving (Show)
```

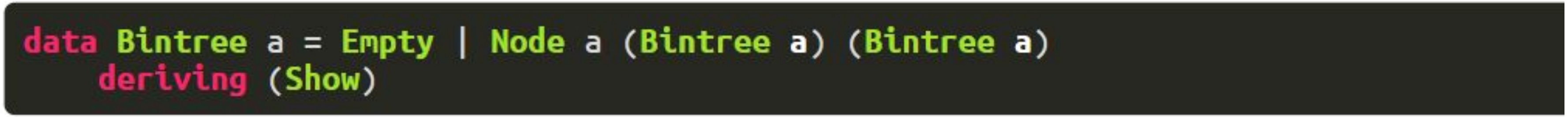
If we wanted to compute the **height** of the tree:



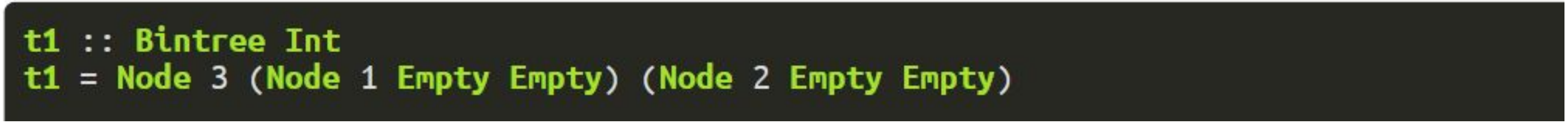
```
height :: Bintree -> Int

height  Empty = 0
height  (Node _ lc rc) = 1 + max (height lc) (height rc)
```

Fundamentals of Binary Trees

Binary Tree of generic data type:



```haskell
data Bintree a = Empty | Node a (Bintree a) (Bintree a)
    deriving (Show)
```



```haskell
t1 :: Bintree Int
t1 = Node 3 (Node 1 Empty Empty) (Node 2 Empty Empty)
```

Instructor Youtube Channel: Lucas Science