```
module Task where
-- safely convert a maybe value
-- to a value of the inner type
fromMaybe' ma x =
    case ma of
        Nothing \rightarrow x
        Just a \rightarrow a
-- Feed zero if the maybe value is nothing
justOrZero n = fromMaybe' 0 n
-- add two maybe values
x = let a = justOrZero (Just 3)
        b = justOrZero Nothing
    in a + b
```

ChameleonIDE The expression from Maybe' can be either one of the two types: fromMaybe'::Maybe a→a→a fromMaybe'::Int→a→a

```
ChameleonIDE
module Task where
                                                     The expression from Maybe' can
-- safely convert a maybe value
                                                     be either one of the two types:
-- to a value of the inner type
fromMaybe' ma x =
                                                      fromMaybe'::Maybe a→a→a
    case ma of
         Nothing \rightarrow x
         Just a \rightarrow a
                                                      fromMaybe'::Int→a→a
-- Feed zero if the maybe value is nothing
justOrZero n = <a href="fromMaybe">fromMaybe</a> o n
-- add two maybe values
x = let a = justOrZero (Just 3)
         b = justOrZero Nothing
    in a + b
```