

$\{ \sim \}$

exercism

Bob







Bob







🤘 "Sure."



👉 "Sure."

👉 "Whoa, chill out!"



👉 "Sure."

👉 "Whoa, chill out!"

👉 "Calm down, I know what I'm doing!"



👉 "Sure."

👉 "Whoa, chill out!"

👉 "Calm down, I know what I'm doing!"

👉 "Fine. Be that way!"



👉 "Sure."

👉 "Whoa, chill out!"

👉 "Calm down, I know what I'm doing!"

👉 "Fine. Be that way!"

👉 "Whatever."

Bob

Bob

Bob

🤘 Distinguish questions

Bob

👉 Distinguish questions

👉 Distinguish yelling

Bob

👉 Distinguish questions

👉 Distinguish yelling

👉 Distinguish a question yelled

Bob

👉 Distinguish questions

👉 Distinguish yelling

👉 Distinguish a question yelled

👉 Understand silence

Bob

- 👉 Distinguish questions
- 👉 Distinguish yelling
- 👉 Distinguish a question yelled
- 👉 Understand silence
- 👉 Provide a fallback



Take 1

```
module Bob exposing (hey)
```

```
hey : String -> String
```

```
hey message =
```


Distinguish Silence

```
hey : String -> String
hey message =
  if isSilence message then
    "Fine. Be that way!"
```

Distinguish Silence

```
hey : String -> String
hey message =
    if isSilence message then
        "Fine. Be that way!"
```

```
isSilence : String -> Bool
isSilence message =
    String.trim message == ""
```

Distinguish a Question Yelled

```
hey : String -> String
hey message =
  if isSilence message then
    "Fine. Be that way!"
  else if isYelling message && isQuestion message then
    "Calm down, I know what I'm doing!"
```

Distinguish a Question Yelled

```
hey : String -> String
hey message =
  if isSilence message then
    "Fine. Be that way!"
  else if isYelling message && isQuestion message then
    "Calm down, I know what I'm doing!"

isYelling : String -> Bool
isYelling message =
  (String.toUpper message == message)
```

Distinguish a Question Yelled

```
hey : String -> String
hey message =
  if isSilence message then
    "Fine. Be that way!"
  else if isYelling message && isQuestion message then
    "Calm down, I know what I'm doing!"

isYelling : String -> Bool
isYelling message =
  (String.toUpperCase message == message)
  && not (onlyDigitsOrNonWords message)
```


Distinguish a Question Yelled

```
isYelling : String -> Bool
isYelling message =
    (String.toUpper message == message)
    && not (onlyDigitsOrNonWords message)

onlyDigitsOrNonWords : String -> Bool
onlyDigitsOrNonWords message =
    let
        regex =
            "^[0-9]|^[a-zA-Z]+$"
            |> Regex.fromString
            |> Maybe.withDefault Regex.never
    in
    Regex.contains regex message
```

Distinguish a Question Yelled

```
hey : String -> String
hey message =
  if isSilence message then
    "Fine. Be that way!"
  else if isYelling message && isQuestion message then
    "Calm down, I know what I'm doing!"
```

Distinguish a Question Yelled

```
hey : String -> String
hey message =
  if isSilence message then
    "Fine. Be that way!"
  else if isYelling message && isQuestion message then
    "Calm down, I know what I'm doing!"
```

```
isQuestion : String -> Bool
isQuestion message =
  message
    |> String.trim
    |> String.endsWith "?"
```

Distinguish Yelled Question

```
hey : String -> String
hey message =
  if isSilence message then
    "Fine. Be that way!"
  else if isYelling message && isQuestion message then
    "Calm down, I know what I'm doing!"
```

Distinguish Yelling

```
hey : String -> String
hey message =
  if isSilence message then
    "Fine. Be that way!"
  else if isYelling message && isQuestion message then
    "Calm down, I know what I'm doing!"
  else if isYelling message then
    "Whoa, chill out!"
```






Distinguish Questions

```
hey : String -> String
hey message =
  if isSilence message then
    "Fine. Be that way!"
  else if isYelling message && isQuestion message then
    "Calm down, I know what I'm doing!"
  else if isYelling message then
    "Whoa, chill out!"
  else if isQuestion message then
    "Sure."
```

Provide a Fallback

```
hey : String -> String
hey message =
  if isSilence message then
    "Fine. Be that way!"
  else if isYelling message && isQuestion message then
    "Calm down, I know what I'm doing!"
  else if isYelling message then
    "Whoa, chill out!"
  else if isQuestion message then
    "Sure."
  else
    "Whatever."
```


Elm Analyse

-  Dashboard
-  All Messages
-  Tree
-  Dependencies
-  Modules



1

Modules




0

Imports



0

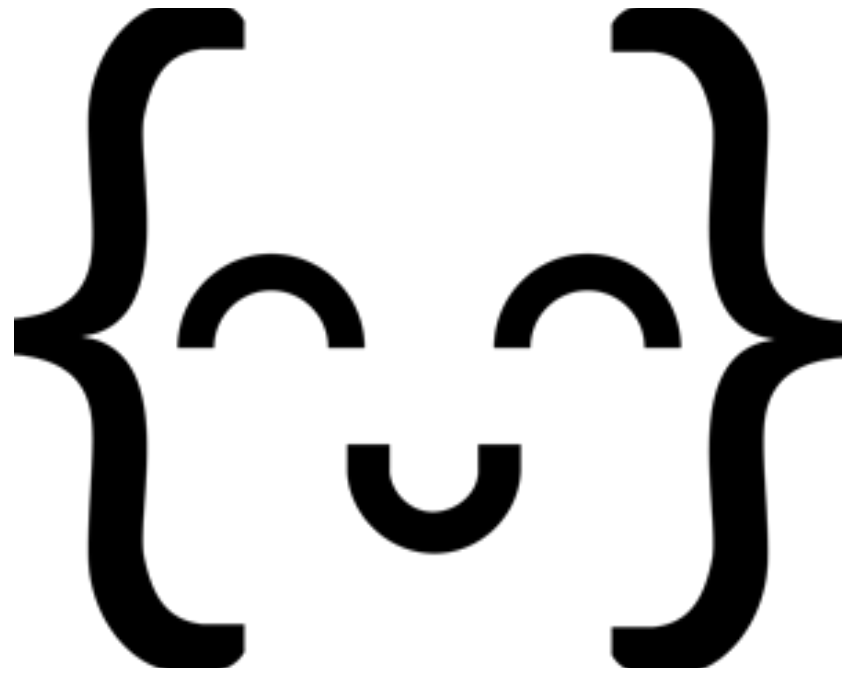
Messages



0

Unused dependencies



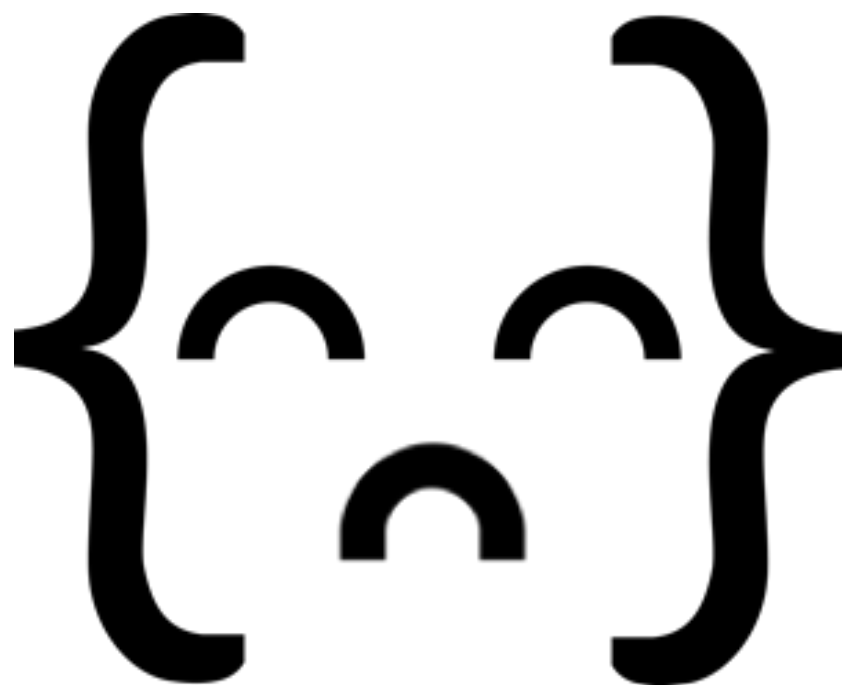


exercism

exercism submit Bob.elm

{~}

Done!



Done?





👉 Repetition in
condition checking



👉 Repetition in
condition checking

👉 Repetition with
`String.trim`



👉 Repetition in
condition checking

👉 Repetition with
`String.trim`

👉 Do we really need a
`regex`?



👉 Repetition in
condition checking

👉 Repetition with
`String.trim`

👉 Do we really need a
regex?

👉 Big if statements
don't feel right



Take 2

Distinguish Silence

```
hey : String -> String  
hey input =
```

Distinguish Silence

```
hey : String -> String
hey input =
  let
    message =
      String.trim input
```


Distinguish Silence

```
hey : String -> String
hey input =
  let
    message =
      String.trim input

    isSilence =
      message == ""
```

Distinguish Silence

```
hey : String -> String
hey input =
  let
    message =
      String.trim input

    isSilence =
      message == ""
  in
  if isSilence then
    "Fine. Be that way!"

  else
    respondToVerbalMessage message
```

Distinguish Question

```
respondToVerbalRemark : String -> String  
respondToVerbalRemark message =
```

Distinguish Question

```
respondToVerbalRemark : String -> String
respondToVerbalRemark message =
  let
    isQuestion =
      String.endsWith "?" message
```

Distinguish Question

```
respondToVerbalRemark : String -> String
respondToVerbalRemark message =
  let
    isQuestion =
      String.endsWith "?" message

    hasLetters =
      String.any Char.isAlpha message
```

Distinguish Yelling

```
respondToVerbalRemark : String -> String
```

```
respondToVerbalRemark message =
```

```
  let
```

```
    isQuestion =
```

```
      String.endsWith "?" message
```

```
    hasLetters =
```

```
      String.any Char.isAlpha message
```

```
    isYelling =
```

```
      hasLetters && String.toUpper message == message
```


Distinguish Question/Yelling Combos

```
respondToVerbalRemark : String -> String
respondToVerbalRemark message =
  let
    -- ...
  in
  case ( isQuestion, isYelling ) of
```

Distinguish Question/Yelling Combos

```
respondToVerbalRemark : String -> String
respondToVerbalRemark message =
  case ( isQuestion, isYelling ) of
    ( True, True ) ->
      "Calm down, I know what I'm doing!"
```

Distinguish Question/Yelling Combos

```
respondToVerbalRemark : String -> String
respondToVerbalRemark message =
  case ( isQuestion, isYelling ) of
    ( True, True ) ->
      "Calm down, I know what I'm doing!"
    ( True, False ) ->
      "Sure."
```

Distinguish Question/Yelling Combos

```
respondToVerbalRemark : String -> String
respondToVerbalRemark message =
  case ( isQuestion, isYelling ) of
    ( True, True ) ->
      "Calm down, I know what I'm doing!"
    ( True, False ) ->
      "Sure."
    ( False, True ) ->
      "Whoa, chill out!"
```

Distinguish Question/Yelling Combos

```
respondToVerbalRemark : String -> String
respondToVerbalRemark message =
  case ( isQuestion, isYelling ) of
    ( True, True ) ->
      "Calm down, I know what I'm doing!"
    ( True, False ) ->
      "Sure."
    ( False, True ) ->
      "Whoa, chill out!"
    ( False, False ) ->
      "Whatever."
```


You'll rarely get it
right the first time

Submit often,
get feedback

Read other
people's code

Refactor until
you're happy



github.com/paulfioravanti/exercism

Thanks!
@paulfioravanti

