```diff
- Cascading Style Sheets
+ Code Style Sheets
```

PAT 4/11/23, Sam Cohen

```
shoutGreeting names =
  concatWith " " (map uppercase "hello:names)

assert (shoutGreeting [ "people"
                      , "tech"
                      , "seminar"]
           == "HELLO PEOPLE TECH SEMINAR")
```

```
shoutGreeting names =
  concatWith " " (map uppercase "hello:names)

assert (shoutGreeting [ "people"
                      , "tech"
                      , "seminar"]
           == "HELLO PEOPLE TECH SEMINAR")
```

```
shoutGreeting names =
  concatWith " " (map uppercase "hello":names)

assert (shoutGreeting [ "people"
                      , "tech"
                      , "seminar"]
           == "HELLO PEOPLE TECH SEMINAR")
```

```
shoutGreeting names =
  concatWith " " (map uppercase "hello":names)

assert (shoutGreeting [ "people"
                      , "tech"
                      , "seminar"]
           == "HELLO PEOPLE TECH SEMINAR")
```

```
shoutGreeting names =
  concatWith " " (map uppercase "hello":names)

assert (shoutGreeting [ "people"
                      , "tech"
                      , "seminar"]
           == "HELLO PEOPLE TECH SEMINAR")
```

```
patPrez.hs:7:23: error:
   • Couldn't match type ‘Char’ with ‘[Char]’
     Expected: [String]
       Actual: String
   • In the second argument of ‘concatMap’, namely ‘"hello, "’
     In the first argument of ‘(:)’, namely
       ‘concatMap uppercase "hello, "’
     In the expression: concatMap uppercase "hello, " : names
  |
7 |   concatMap uppercase "hello, ":names
  |                       ^^^^^^^^^
```

```
shoutGreeting names =
  concatWith " " (map uppercase "hello":names)

assert (shoutGreeting [ "people"
                      , "tech"
                      , "seminar"]
           == "HELLO PEOPLE TECH SEMINAR")
```

```
patPrez.hs:7:23: error:
   • Couldn't match type ‘Char’ with ‘[Char]’
     Expected: [String]
       Actual: String
   • In the second argument of ‘concatMap’, namely ‘"hello, "’
     In the first argument of ‘(:)’, namely
       ‘concatMap uppercase "hello, "’
     In the expression: concatMap uppercase "hello, " : names
  |
7 |   concatMap uppercase "hello, ":names
  |                       ^^^^^^^^^
```

```haskell
shoutGreeting names =
  concatWith " " (map uppercase ("hello":names))

assert (shoutGreeting [ "people"
                      , "tech"
                      , "seminar"]
           == "HELLO PEOPLE TECH SEMINAR")
```

```
patPrez.hs:7:23: error:
   • Couldn't match type ‘Char’ with ‘[Char]’
     Expected: [String]
       Actual: String
   • In the second argument of ‘concatMap’, namely ‘"hello, "’
     In the first argument of ‘(:)’, namely
       ‘concatMap uppercase "hello, "’
     In the expression: concatMap uppercase "hello, " : names
  |
7 |   concatMap uppercase "hello, ":names
  |                       ^^^^^^^^^
```

Two goals of CSS:

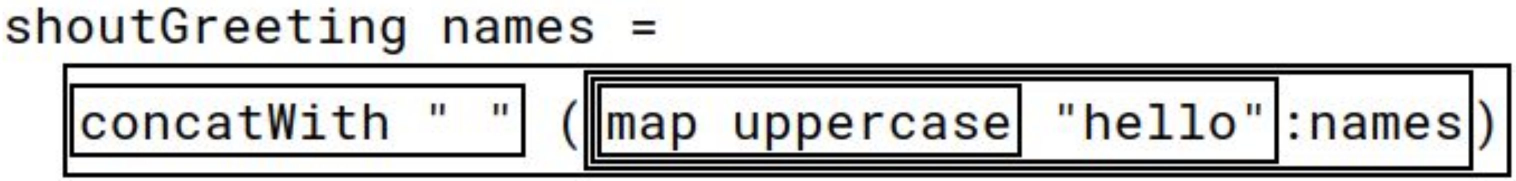
- *Rich Styles*: Code representations should use color, but also grouping, hierarchy, and meaningful typography.

Two goals of CSS:

- *Rich Styles*: Code representations should use color, but also grouping, hierarchy, and meaningful typography.

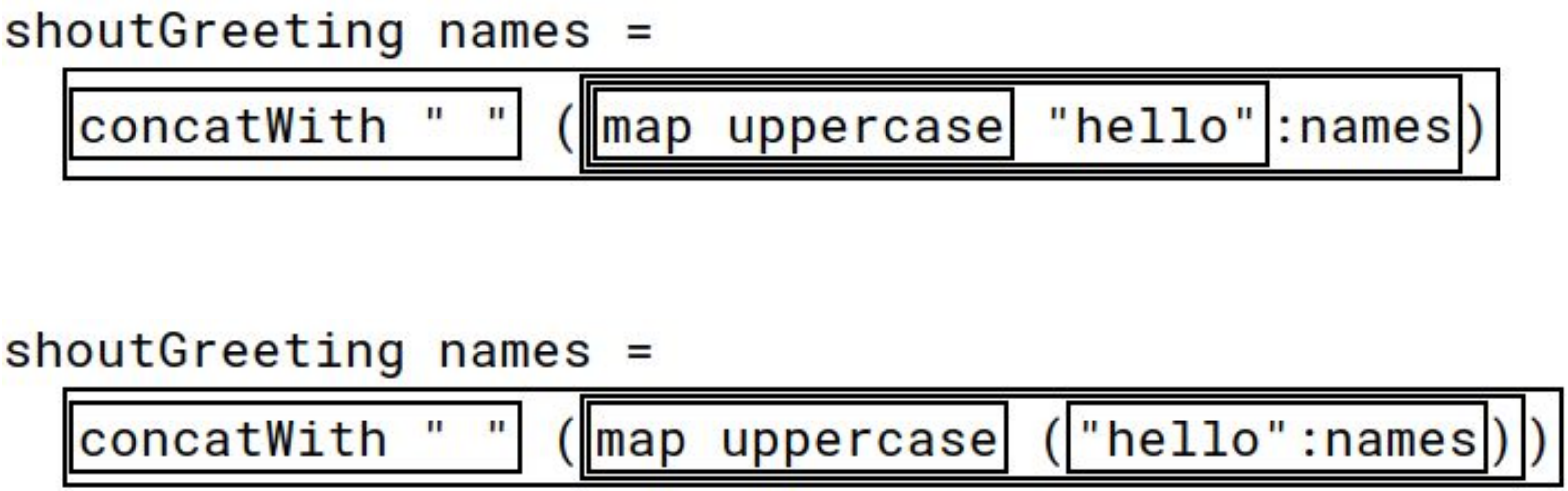
- *Configurable Styles*: Visual representation should be user or task controlled.

Example: Expose program structure



```
shoutGreeting names =
  concatWith " " (map uppercase "hello":names)
```

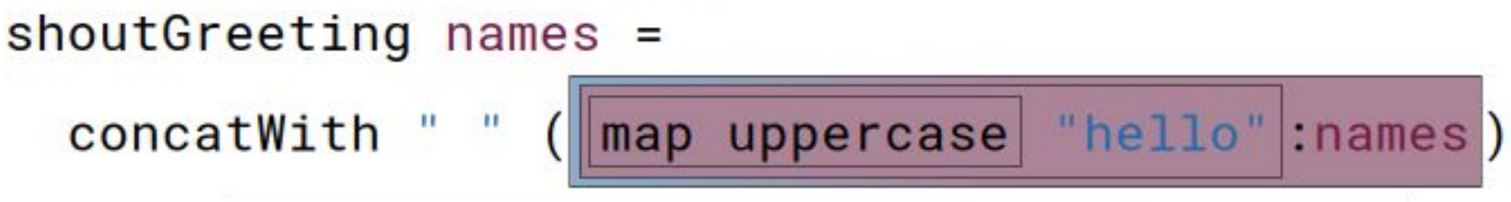
Example: Expose program structure



```
shoutGreeting names =
  concatWith " " (map uppercase "hello":names)
```

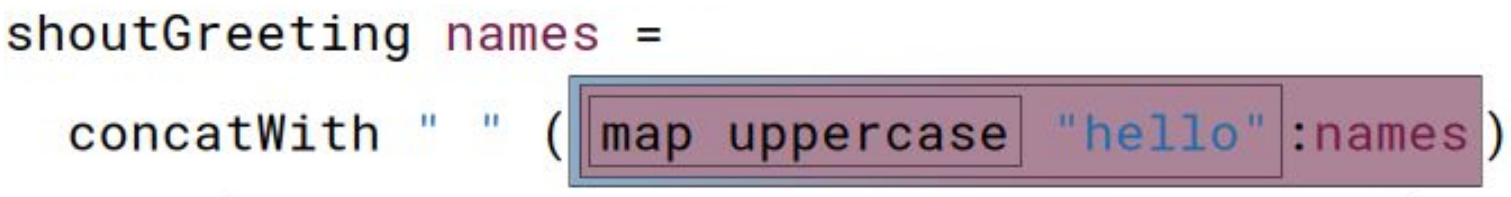
```
shoutGreeting names =
  concatWith " " (map uppercase ("hello":names))
```

Example: Expose type information

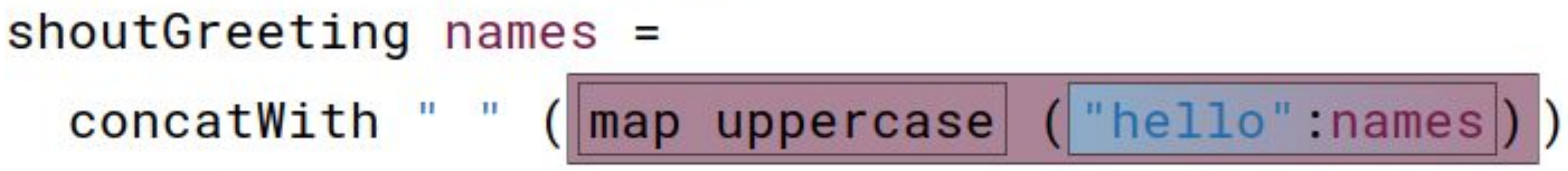


```
shoutGreeting names =
  concatWith " " (map uppercase "hello":names)
```

Example: Expose type information



```
shoutGreeting names =
  concatWith " " (map uppercase "hello":names)
```



```
shoutGreeting names =
  concatWith " " (map uppercase ("hello":names))
```

Example: Task specific highlights

Task 1: Syntactical Error

```
shoutGreeting names =
  concatWith " " (map uppercase "hello:names)

assert (shoutGreeting [ "people"
                      , "tech"
                      , "seminar"]
           == "HELLO PEOPLE TECH SEMINAR")
```

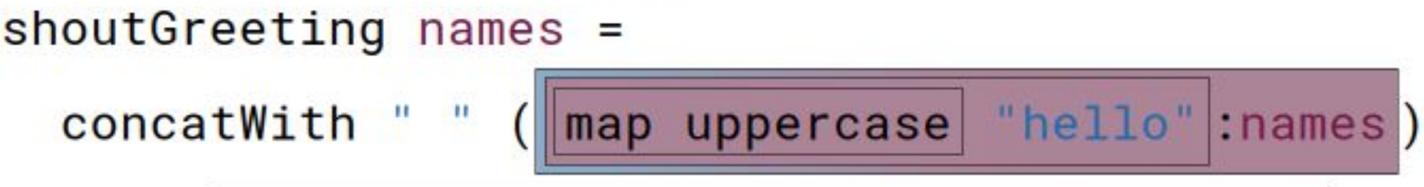
Example: Task specific highlights

Task 1: Syntactical Error

```
shoutGreeting names =
  concatWith " " (map uppercase "hello:names)

assert (shoutGreeting [ "people"
                      , "tech"
                      , "seminar"]
           == "HELLO PEOPLE TECH SEMINAR")
```

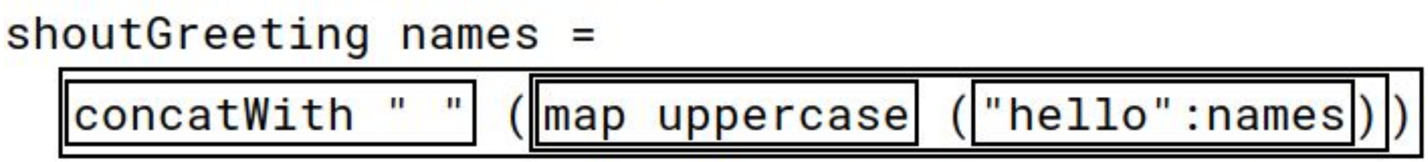
Task 2: Type Error



```
shoutGreeting names =
  concatWith " " (map uppercase "hello":names)
```

User Controlled, Language Agnostic

```css
.HsApp {
  border-width: 2px;
  padding: 2px;
}
```

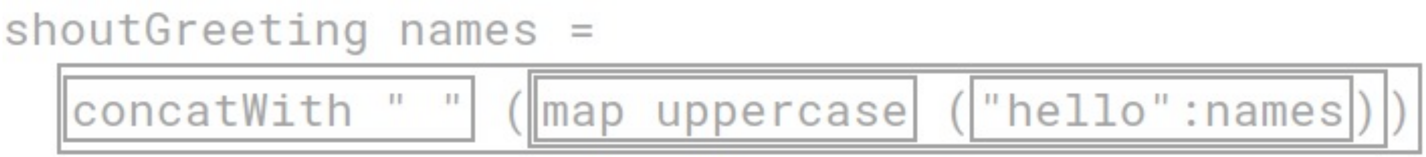


```haskell
shoutGreeting names =
  concatWith " " (map uppercase ("hello":names))
```

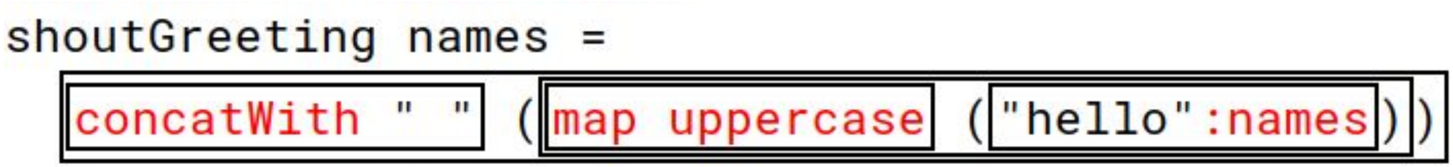
User Controlled, Language Agnostic

```
.HsApp {
  border-width: 2px;
  padding: 2px;
}
```

```
.HsVar {
  color: red;
}
```



```
shoutGreeting names =
  concatWith " " (map uppercase ("hello":names))
```



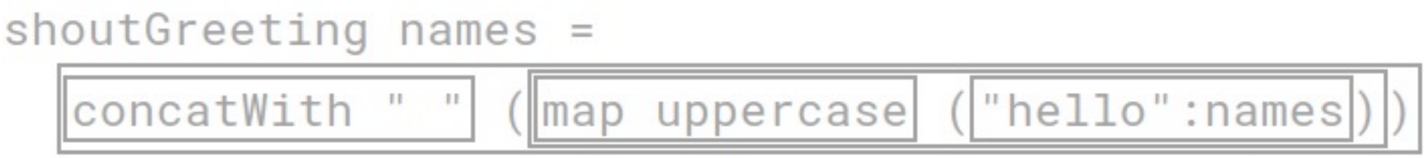
```
shoutGreeting names =
  concatWith " " (map uppercase ("hello":names))
```

User Controlled, Language Agnostic

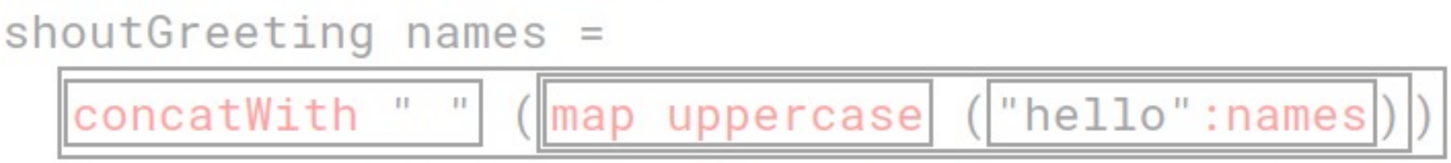
```
.HsApp {
  border-width: 2px;
  padding: 2px;
}
```

```
.HsVar {
  color: red;
}
```

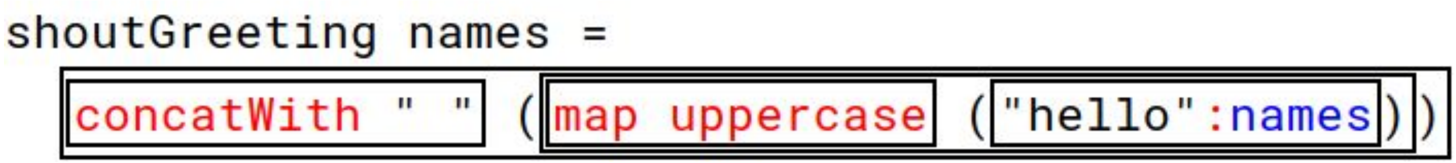
```
.HsVar String List {
  color: blue;
}
```



```
shoutGreeting names =
  concatWith " " (map uppercase ("hello":names))
```



```
shoutGreeting names =
  concatWith " " (map uppercase ("hello":names))
```



```
shoutGreeting names =
  concatWith " " (map uppercase ("hello":names))
```

Direction

- *Short term:* Conduct a survey to evaluate our visualizations and interactions
- *Medium term:* Continue to build-out our prototype, use static visualizations in CS 223 course materials
- *Long term:* Dynamic visualizations and editor integration

Direction

Thank you!

- *Short term:* Conduct a survey to evaluate our visualizations and interactions
- *Medium term:* Continue to build-out our prototype, use static visualizations in CS 223 course materials
- *Long term:* Dynamic visualizations and editor integration