CMPU 101 § 53 · Computer Science I

Reactive Programs

27 February 2024

DataFest 2024 @ Vassar

Friday April 5 - Sunday April 7, 2024

5PM Friday to 5PM Sunday
Kenyon Hall and Blodgett Auditorium

Your challenge for the weekend: ask questions and draw insights from the data. But the data will remain a secret until Friday's opening ceremony! To participate, you only need an enthusiasm for data and friendly competition. Prizes for the winners; food and swag for everyone. Registration is FREE but limited to the first 80 students.
Register by Friday March 1 (before Spring Break).

To register and for more information, visit https://pages.vassar.edu/datafest/ or scan the QR code:

DataFest 2024 @ Vassar

Friday April 5 - Sunday April 7, 2024

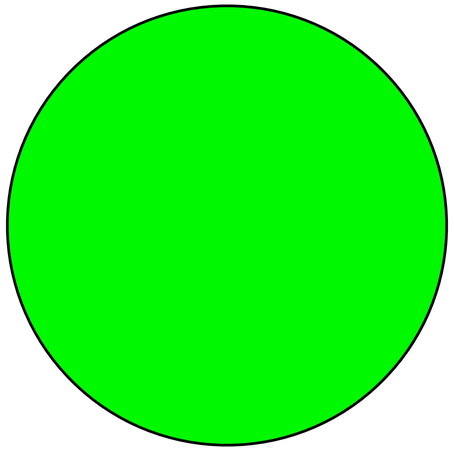
5PM Friday to 5PM Sunday
Kenyon Hall and Blodgett Auditorium

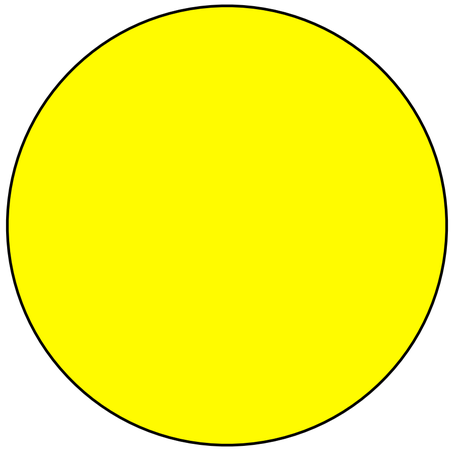
Your challenge for the weekend: ask questions and draw insights from the data. But the data will remain a secret until Friday's opening ceremony! To participate, you only need an enthusiasm for data and friendly competition. Prizes for the winners; food and swag for everyone. Registration is FREE but limited to the first 80 students.

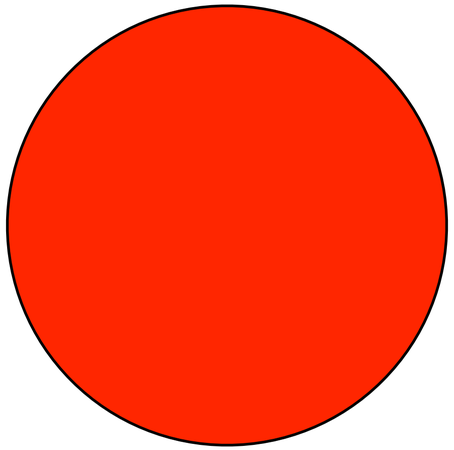
Register by Friday March 1 (before Spring Break).

To register and for more information, visit https://pages.vassar.edu/datafest/
or scan the QR code:

Where are we?







All traffic lights are the same size and position on the screen.

All traffic lights are the same size and position on the screen.

What distinguishes them?

All traffic lights are the same size and position on the screen.

What distinguishes them?

Asking this helps us think about data

All traffic lights are the same size and position on the screen.

All traffic lights are the same size and position on the screen.

How do we get from one to the other?

All traffic lights are the same size and position on the screen.

How do we get from one to the other?

*Asking this helps us think about **functions***

Data

Data definition

Examples

Template

Functions

Signature

Docstring

Examples

Body

Data

Functions

Data definition

Examples

Template

Signature

Docstring

Examples

Body

```
data TrafficLight:
  ...
end
```

```
data TrafficLight:
  | green
  | yellow
  | red
end
```

Data

Functions

Data definition

Examples

Template

Signature

Docstring

Examples

Body

```
data TrafficLight:
  | green
  | yellow
  | red
end

TL-GREEN = green
TL-YELLOW = yellow
TL-RED = red
```

For this data definition, the examples are so trivial we can skip them, but you saw in the pipeline lab how helpful it can be to have examples when you have a lot of possibilities!

```
data TrafficLight:
  | green
  | yellow
  | red
end
```

Data

Data definition

Examples

Template

Functions

Signature

Docstring

Examples

Body

```
data TrafficLight:
  | green
  | yellow
  | red
end
```

```
data TrafficLight:
  | green
  | yellow
  | red
end
#|
fun trafficlight-fun(tl :: TrafficLight) -> ...:
|#
```

```
data TrafficLight:
  | green
  | yellow
  | red
end
#|
fun trafficlight-fun(tl :: TrafficLight) -> ...:
  doc: "TrafficLight template"
  cases (TrafficLight) tl:
    | green => ...
    | yellow => ...
    | red => ...
  end
where:
  trafficlight-fun(green) is ...
  trafficlight-fun(yellow) is ...
  trafficlight-fun(red) is ...
end |#
```

Data

Functions

Data definition

Examples

Template

Signature

Docstring

Examples

Body

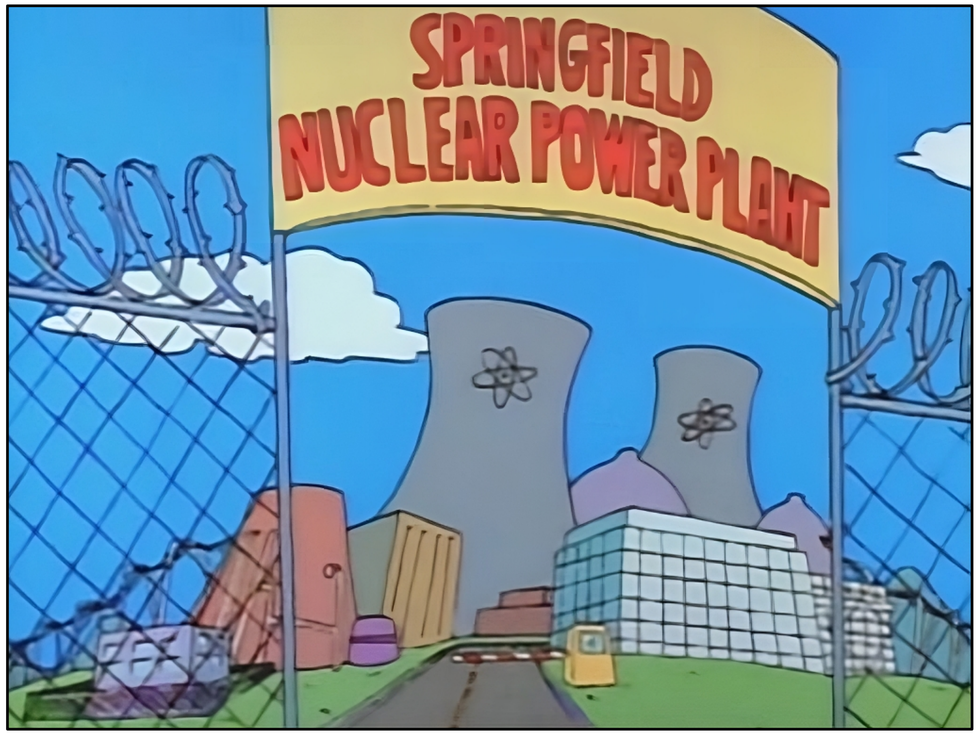
As we saw last class, Pyret has a mechanism for supporting interactive programs, called a **`reactor`**.

To use it, first write

```
include reactors
```

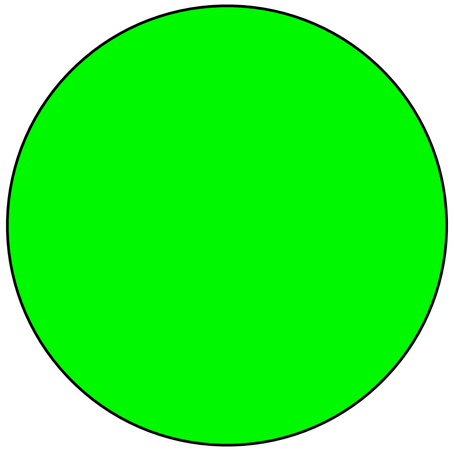
```
reactor:
  init: initial-state,
  to-draw: draw-function,
  event-type: event-function
end
```

```
reactor:
  init: initial-state,
  to-draw: draw-function,
  event-type: event-function
end
```



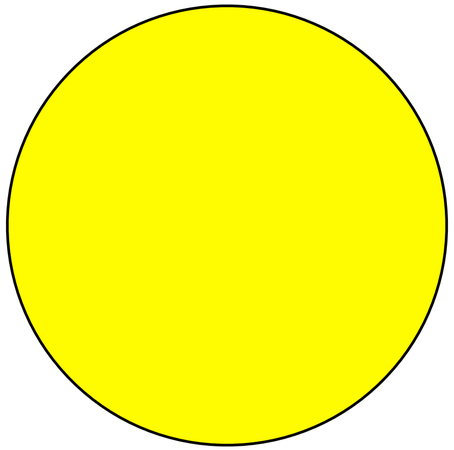
Less nuclear reactor; more person-that-reacts to something.

`reactor` puts all the pieces together to start things going.

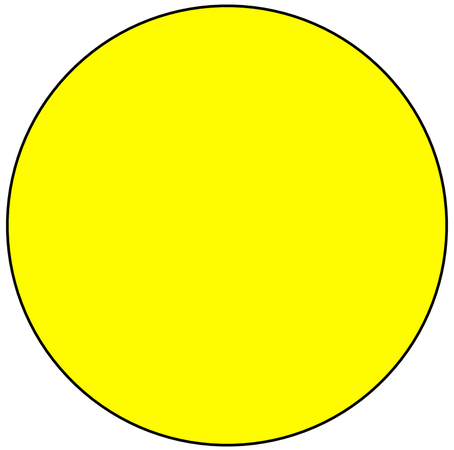


initial state

some event happens…



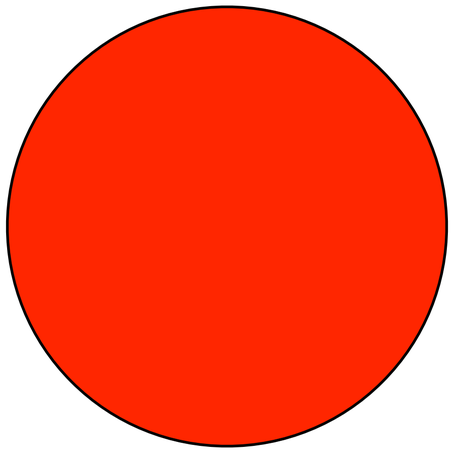
next state



~~*next state*~~

now the current state

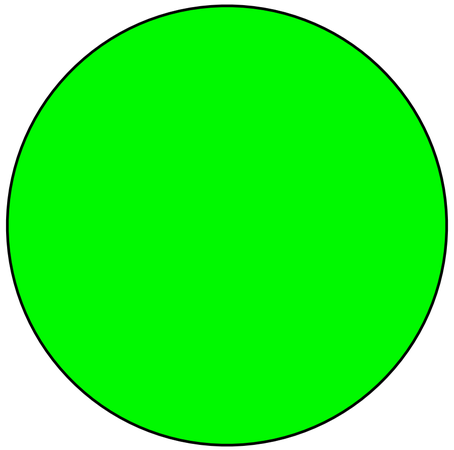
some event happens…



~~*next state*~~

now the current state

some event happens…



~~*next state*~~

now the current state

```
reactor:
  init: initial-state,
  to-draw: draw-function,
  event-type: event-function
end
```

```
reactor:
  init: green,
  to-draw: draw-function,
  event-type: event-function
end
```

```
reactor:
  init: green,
  to-draw: draw-light,
  event-type: event-function
end
```

```
reactor:
  init: green,
  to-draw: draw-light,
  event-type: event-function
end
```

We haven't written this;

add it to our wishlist!

```
reactor:
  init: green,
  to-draw: draw-light,
  on-tick: next-light
end
```

```
reactor:
  init: green,
  to-draw: draw-light,
  on-tick: next-light
end
```

Another function for the wishlist!

So far…

```
# TrafficLight data
# - definition
# - examples
# - template

# define reactor

# Wishlist:
# - fun draw-light...
# - fun next-light...
```

Data

Data definition

Examples

Template

Functions

Signature

Docstring

Examples

Body

```
fun draw-light(tl :: TrafficLight) -> Image:
  ...
end
```

```
fun draw-light(tl :: TrafficLight) -> Image:
  ...
end
```

```
fun next-light(tl :: TrafficLight) -> TrafficLight:
  ...
end
```

Data

Data definition

Examples

Template

Functions

Signature

Docstring

Examples

Body

```
fun draw-light(tl :: TrafficLight) -> Image:
  ...
end
```

```
fun next-light(tl :: TrafficLight) -> TrafficLight:
  ...
end
```

```
fun draw-light(tl :: TrafficLight) -> Image:
  doc: "Draw a circle of the given color, rendering a traffic light"
  ...
end
```

```
fun next-light(tl :: TrafficLight) -> TrafficLight:
  ...
end
```

```
fun draw-light(tl :: TrafficLight) -> Image:
  doc: "Draw a circle of the given color, rendering a traffic light"
  ...
end
```

```
fun next-light(tl :: TrafficLight) -> TrafficLight:
  doc: "Produce the next light in the sequence green, yellow, red"
  ...
end
```

Data

Data definition

Examples

Template

Functions

Signature

Docstring

Examples

Body

```
fun draw-light(tl :: TrafficLight) -> Image:
  doc: "Draw a circle of the given color, rendering a traffic light"
  ...
end
```

```
fun next-light(tl :: TrafficLight) -> TrafficLight:
  doc: "Produce the next light in the sequence green, yellow, red"
  ...
end
```

```
fun draw-light(tl :: TrafficLight) -> Image:
  doc: "Draw a circle of the given color, rendering a traffic light"
  ...
where:
  draw-light(green) is circle(50, "solid", "green")
  draw-light(yellow) is circle(50, "solid", "yellow")
  draw-light(red) is circle(50, "solid", "red")
end
```

```
fun next-light(tl :: TrafficLight) -> TrafficLight:
  doc: "Produce the next light in the sequence green, yellow, red"
  ...
end
```

```
fun draw-light(tl :: TrafficLight) -> Image:
  doc: "Draw a circle of the given color, rendering a traffic light"
  ...
where:
  draw-light(green) is circle(50, "solid", "green")
  draw-light(yellow) is circle(50, "solid", "yellow")
  draw-light(red) is circle(50, "solid", "red")
end
```

```
fun next-light(tl :: TrafficLight) -> TrafficLight:
  doc: "Produce the next light in the sequence green, yellow, red"
  ...
where:
  next-light(green) is yellow
  next-light(yellow) is red
  next-light(red) is green
end
```

Data

Data definition

Examples

Template

Functions

Signature

Docstring

Examples

Body

Starter code:

tinyurl.com/2024-02-27-tl-starter

Code:

tinyurl.com/2024-02-27-tl

Screensaver


DVD
VIDEO


DVD
VIDEO

Starter code:

tinyurl.com/2024-02-27-dvd-starter

Code:

tinyurl.com/2024-02-27-dvd

Exercise

More advanced version: Make the logo change color when it hits an edge – or when you click.

Class version with revised data definition for color changing:

`tinyurl.com/2024-02-27-dvd-color`

Acknowledgments

This lecture incorporates material from:

Greg Daniels, *The Office*

Marc Smith, Vassar College

Laney Strange, Northeastern University