9:17 - 10:17

Morning drills

--

Get into a problem-solving mindset

- stretch,
- get going &
- get coding

morning drills

- wake up!
- start thinking like software developers
- develop interview skills
  - get used to white-boarding
  - get used to talking about code

Kyrel

after this discussion, you will be able to:

- Use the methods in kyrel

- work alone or with a partner to solve the day-1 exercises

Game play

We always start with an array of 5 elements, like:

`[ '.', '.', '.', '.', '.' ]` ==

We're always given a goal like:

`[ '.', '.', 'b', '.', '.' ]`

Cursor Movement

We always start in the left-most element of the array.

```
[ '.', '.', '.', '.', '.' ]
```

We can:

- moveRight();
- moveLeft();

Drawing

useGreen(); // switches to **green** color

useBlue(); // switches to **blue** color

draw(); // draws a mark using the **current color**

erase(); // removes a mark

So how can we get from:

```
[ '.', '.', '.', '.', '.' ]
```

To:

```
[ '.', '.', 'g', '.', '.' ]
```

Using the functions: (commands)

moveRight(); moveLeft(); useGreen();

useBlue(); draw(); erase();

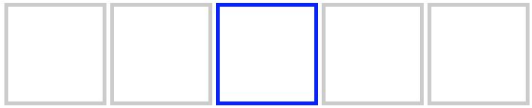
goal: [‘.’, ‘.’, ‘g’, ‘.’, ‘.’]

moveRight();

goal: [‘.’, ‘.’, ‘g’, ‘.’, ‘.’]

moveRight();

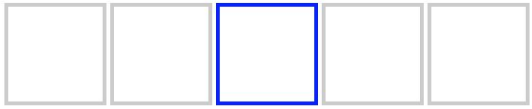
moveRight();



goal: `[ '.', '.', 'g', '.', '.' ]`

moveRight();

moveRight();



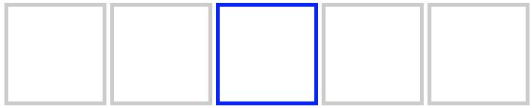
useGreen();

draw();

goal: ['.', '.', 'g', '.', '.']

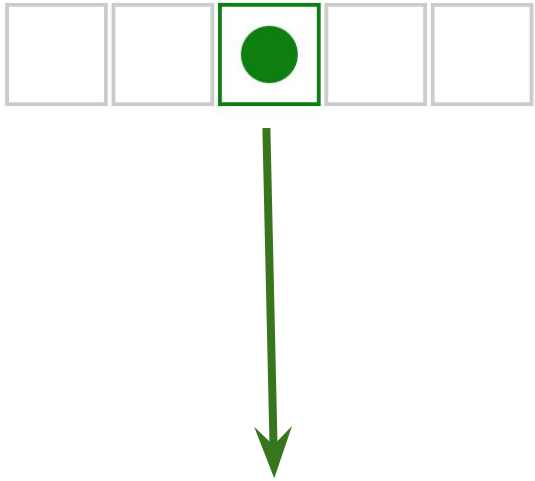
moveRight();

moveRight();



useGreen();

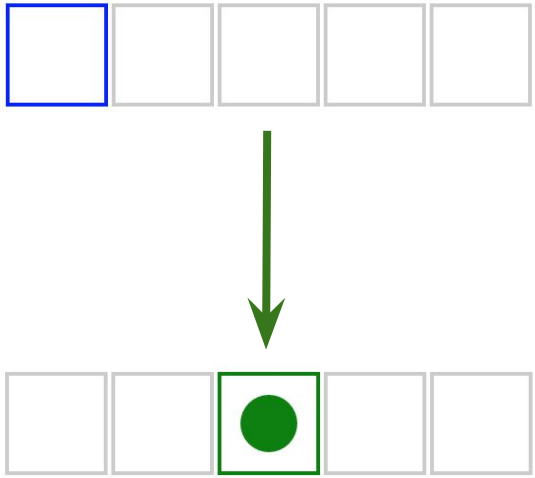
draw();



goal: [‘.’, ‘.’, ‘g’, ‘.’, ‘.’]

Final solution

```
function main() {
    moveRight();
    moveRight();
    useGreen();
    draw();
  }
```



Right?

once again, life has *rules*

- We always begin at the **LEFT**
- You cannot leave the board / row.
- Always set the **color** before you draw.
- You CAN overwrite.....
  - you don’t have to erase first
- sorry, no adding new ***functions***

EXERCISES

2) erase cell 3 start: ['b', 'b', 'b', 'b', 'b'] finish: ['b', 'b', '.', 'b', 'b']	3) erase every other start: ['b', 'b', 'b', 'b', 'b'] finish: ['b', '.', 'b', '.', 'b']	4) every other blue start: ['g', 'g', 'g', '.', '.'] finish: ['g', 'b', 'g', 'b', '.']

Checking the color

onBlue(); // true if you’re on a blue

onGreen(); // true if you’re on a green

5) move start to finish

case 1:	case 2:	case 3:
start: ['b', '.', '.', '.', '.']	start: ['g', '.', '.', '.', '.']	start: ['.', '.', '.', '.', '.']
finish: ['.', '.', '.', '.', 'b']	finish: ['.', '.', '.', '.', 'g']	finish: ['.', '.', '.', '.', '.']

use same code to complete all three cases

```
if( onBlue() ) {
   useBlue();
} else if ( onGreen() ){
   useGreen();
}

erase();
moveRight();
moveRight();
moveRight();
moveRight();
draw();
```


FIN

Kyrel

day 2

```
moveRight();
moveLeft();
                    useGreen();
                     useBlue();

draw();
erase();
                          onBlue();
___                 onGreen();
```

Last time, on Kyrel...

5) move start to finish

case 1:	case 2:	case 3:
start: ['b', '.', '.', '.', '.']	start: ['g', '.', '.', '.', '.']	start: ['.', '.', '.', '.', '.']
finish: ['.', '.', '.', '.', 'b']	finish: ['.', '.', '.', '.', 'g']	finish: ['.', '.', '.', '.', '.']

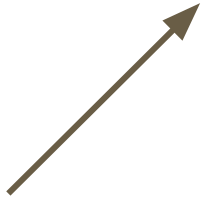
use same code to complete all three cases

```
if( onBlue() ) {
   useBlue();
} else if ( onGreen() ){
   useGreen();
}

erase();
moveRight();
moveRight();
moveRight();
moveRight();
draw();
```

```
if( onBlue() ) {
   useBlue();
} else if ( onGreen() ){
   useGreen();
}

erase();
moveRight();
moveRight();
moveRight();
moveRight();
draw();
```



better solution?

```
if( onBlue() ) {
    useBlue();
} else if ( onGreen() ) {
    useGreen();
}
if( onBlue() || onGreen() ) {
    erase();
    moveRight();
    moveRight();
    moveRight();
    moveRight();
    draw();
}
```

can we think of broader solutions to the problem: ”move start to finish” ?

```
if( onBlue() ) {
    useBlue();
} else if ( onGreen() ) {
    useGreen();
}
if( onBlue() || onGreen() ) {
    erase();
    moveRight();
    moveRight();
    moveRight();
    moveRight();
    draw();
}
```

even better solutions?

As you work through these…. (goals, technique)

- Abstract the problem
    a. Some problems will have more than one “case”
        i. ensure that your code will work for **ALL** “cases”.
- "Don't repeat yourself" (DRY)
    a. If you see a lot of repetition in your code, **refactor it**.
- Be efficient.
    a. How many steps do your instructions take?
    b. How does the number of steps compare to the number of cells in the row?

Day 2 Problem 1

- all blue -

start: ['.', '.', '.', '.', '.']

finish: ['b', 'b', 'b', 'b', 'b']

Day 2 Problem 1 -- *try to use for(...) { }*

all blue

moveRight();

moveLeft();

draw();

erase();

useGreen();

useBlue();

onBlue();

onGreen();

start: ['.', '.', '.', '.', '.']

finish: ['b', 'b', 'b', 'b', 'b']

Day 2 Problem 1 Solution

```
useBlue();
for(var j=0; j<5; j++) {
  draw();
  moveRight();
}
```

all blue

start: ['.', '.', '.', '.', '.']

finish: ['b', 'b', 'b', 'b', 'b']

As you work through these…. (goals, technique)

- Abstract the problem
    a. Some problems will have more than one “case”
        i. ensure that your code will work for **ALL** “cases”.
- "Don't repeat yourself" (DRY)
    a. If you see a lot of repetition in your code, **refactor it**.
- Be efficient.
    a. How many steps do your instructions take?
    b. How does the number of steps compare to the number of cells in the row?

Day 2 Problem 2

- all first color -

start: ['b', '.', '.', '.', '.']

finish: ['b', 'b', 'b', 'b', 'b']

start: ['g', '.', '.', '.', '.']

finish: ['g', 'g', 'g', 'g', 'g']

Day 2 Problem 2

start: ['b', '.', '.', '.', '.']

finish: ['b', 'b', 'b', 'b', 'b']

```
if ( onBlue() ) {
  useBlue()
} else if ( onGreen() ) {
  useGreen();
}
for(var j=0; j<5; j++) {
  draw();
  moveRight();
}
```

Any other solutions?

start: ['b', '.', '.', '.', '.']

finish: ['b', 'b', 'b', 'b', 'b']

```
if ( onBlue() ) {
  useBlue()
} else if ( onGreen() ) {
  useGreen();
}
for(var j=0; j<5; j++) {
  draw();
  moveRight();
}
```

Let's get started

Browse to: **github.com/your-class/kyrel**

to clone the repo to your computer:

cd ~/dev

git clone cloneURL

-- **don't use the solutions!**

work on **kyrel/challenges/day2**

use your browser

try not to peek at the solutions

refresh whenever you change kyrel.js


FIN

This slide intentionally blank

Kyrel

day 3

```
moveRight();
moveLeft();
                    useGreen();
                     useBlue();

draw();
erase();
                          onBlue();
___                    onGreen();
```

Interview prep / Goals

Using:

- whiteboarding
- your voices and kyrel knowledge

Be able to explain a problem and a solution

Be able to walk us through your code

This is all good practice for interviews!

In pairs

Prepare a problem solution

Test your solution on your laptops to make sure it solves the problem

Put your solution on the wall

then....

Explain it to the class

Rules

- whiteboard required
- both partners must speak
- we'll go around the room, to everyone