Basic Methods (DrawingGizmo class)

- **Update Methods**
  - public void draw()
    - post: The DrawingGizmo is set to drawing mode (and is colored green).
  - public void dontDraw()
    - post: The DrawingGizmo is set to moving mode (and is colored red).
  - public void turnClockwise()
    - post: The DrawingGizmo is rotated 30 degrees clockwise from its current heading.
  - public void turnCounterclockwise()
    - post: The DrawingGizmo is rotated 30 degrees counterclockwise from its current heading.

- Some methods have **post-conditions only**
  - Nothing special is required for them to work
  - Only need DrawingGizmo object instance to call them

Methods with Both Pre- and Post-conditions

- public void moveForward()
  - pre: The DrawingGizmo should be at least 20 pixels from the edge of the screen towards which it is pointing; if it is not, then the object and part of the line it draws will appear off-screen.
  - post: The DrawingGizmo moves in the direction of its arrow by 20 pixels. If it is in drawing mode, a line segment will be drawn between its current and prior position.

  **The preconditions on the moveForward() method tell us what must be true before the method can properly run.**
  (If these are not satisfied, then the method may not work properly, or even work at all.)

  **The postconditions tell us what will be true after the method has finished running properly.**
More Complex Methods

- **DrawingGizmo** object has simple methods, each of which does the same thing, the same way, every time.

- But it also has some more complex methods, which:
  - Take input parameters
  - Change what they do based on those parameters

A Parameterized Method

The **turnBy( int )** method wants us to insert an integer value ("int" in Java) as an input parameter.

We must give it an integer when we call it, or it won’t work!

```java
ten.turnBy( 90 );
ten.turnBy( 9000 );
ten.turnBy( -180 );
ten.turnBy( 3.6 );
ten.turnBy();
```

Two More Parameterized Methods

- These new **DrawingGizmo** methods change the color of the background window and drawing line.

- E.g., for a yellow line on a blue background:

```java
ten.setBackground( java.awt.Color.blue );
ten.setForeground( java.awt.Color.yellow );
```
Exercise

- List 3 methods you have used this semester that do not take parameters.
- List 3 methods you have used this semester that do take at least one parameter.

31 March 2020
Software Design I (CS 120) 9

Void vs Non-void Methods

- Void Method: outputs something or changes one or more objects but does not return a value
  - System.out.println()
  - window.setBackground(Color.blue)
- Non-void Method: may make some change to object but primary effect is that it returns a value
  - window.getWidth()
  - oval.getBackground()

In general:
- Setters are void methods
- Getters are non-void methods

31 March 2020
Software Design I (CS 120) 10

Methods in the String Class

<table>
<thead>
<tr>
<th>String</th>
<th>Strings are objects with many methods</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Returns character at input position first char at: 0</td>
</tr>
<tr>
<td></td>
<td>last char at: length() - 1</td>
</tr>
<tr>
<td></td>
<td>Returns number of chars in String</td>
</tr>
<tr>
<td></td>
<td>Returns sub-part of String, starting from input position, going all the way to end</td>
</tr>
<tr>
<td></td>
<td>Returns subpart of String, starting from first input position, going to second - 1</td>
</tr>
</tbody>
</table>

Returns lower/upper case version (Note: does not change original)

Full list online at:
http://docs.oracle.com/javase/8/docs/api/java/lang/String.html

31 March 2020
Software Design I (CS 120) 11

Calling Void and Non-void Methods

- Void method: standalone call
  ```java
  win.setWidth(500);
  win.setHeight(600);
  System.out.println("Window size set.");
  ```
- Non-void method: make use of returned value
  ```java
  int w = win.getWidth();
  if(win.getHeight() > w)
  System.out.println("Portrait mode.");
  ```

31 March 2020
Software Design I (CS 120) 12
Methods in the String Class

String first, last;
first = "Ronald";
last = "McDonald";
System.out.print( first.charAt( 1 ) ); // o
System.out.print( last.length() ); // 8
System.out.print( last.substring( 2 ) ); // Donald
System.out.print( first.substring( 0, 3 ) ); // Ron
first = first.toLowerCase();
last = last.toUpperCase();
System.out.println( first + " " + last ); // ronald MCDONALD

Non-Void Methods and Operators

These methods have non-void return types, meaning that after they have completed running, they output a value.

The basic rule of such methods is that they can be called and used anywhere in our code that a value of the return type can be used.

This Week & Next

Meetings this week:
- Monday/Wednesday: Lab assignments
- Tuesday/Friday: Recorded lectures

Reading 05: Ch. 6 due Friday April 3 at 5:00 PM

Program 04: due Tuesday April 7 at 11:59 PM

Office Hours: via the interwebs
- Monday/Tuesday/Wednesday/Friday: 9:00 AM–11:00 AM
- https://kube-0.cs.uwlax.edu:8443/ZombieApocalypseOfficeHours