YEAH Hours: HangKarel

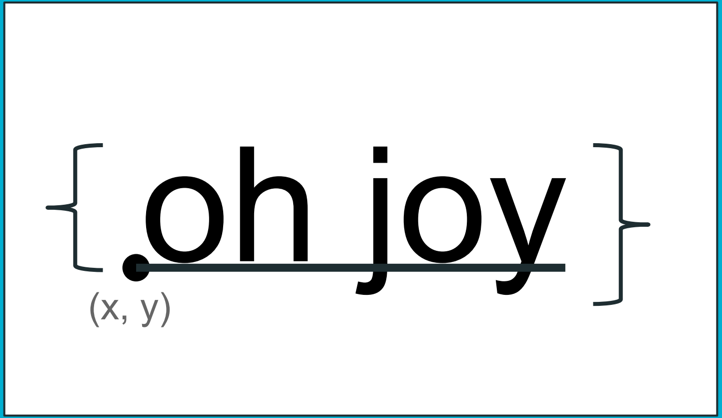
10/31/18

Ryan Eberhardt

Using GLabels

```
let label = GLabel("oh joy");
gw.add(label, WINDOW_WIDTH / 2 - label.getWidth() / 2,
              WINDOW_HEIGHT / 2 + label.getAscent() / 2);
```

getAscent() returns the distance from the baseline to the top of the label

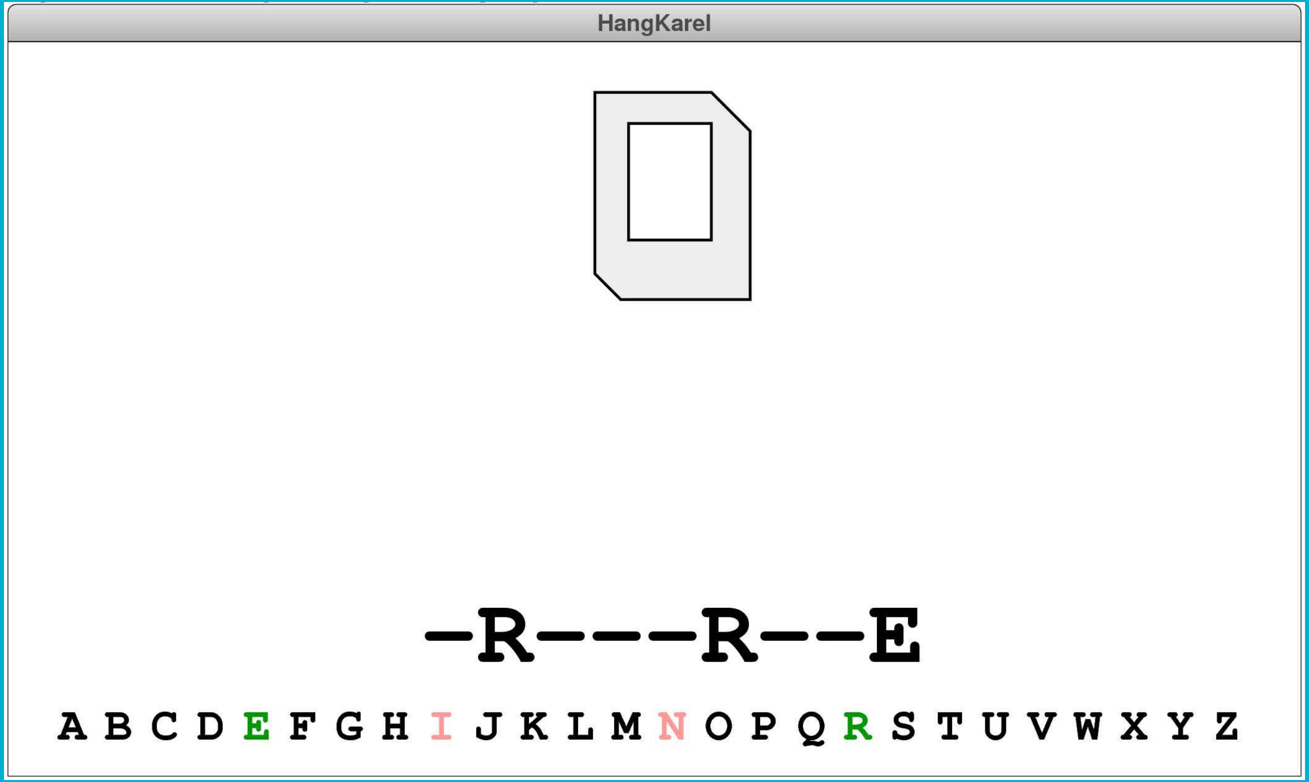


getHeight() returns the full height, including the parts below the baseline

Using GLabels

- The "baseline" is the line that most letters sit on top of (but letters like j, y, and g hang over the bottom of the baseline)
- Note: The anchor/reference point of a GLabel is the left **baseline**, not the top left that we are used to!
- Useful functions:
  - label.getWidth() and label.getAscent()
  - label.setFont(fontName)
  - label.getText()
  - label.setText(newText)
  - label.setColor(color)
  - label.setLocation(x, y)

HangKarel



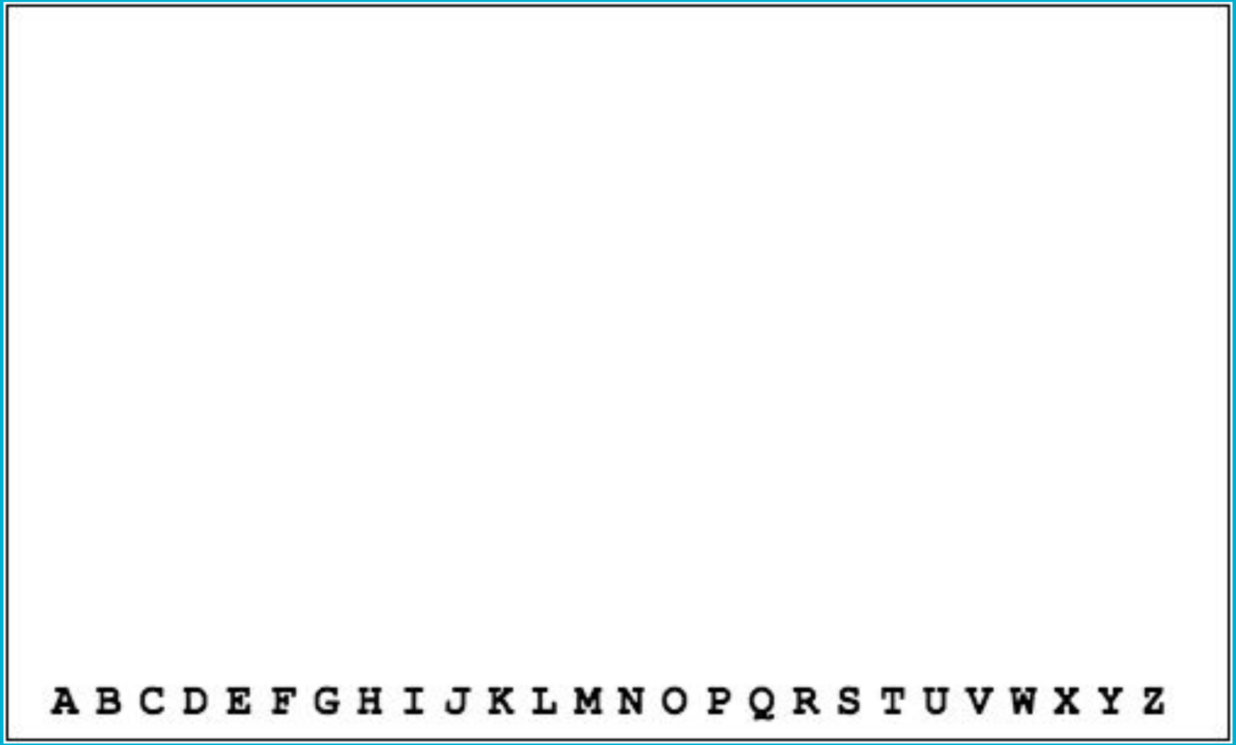
Logistics

- *Individual* assignment
- Also broken into milestones
- Due Monday, Nov 6

Warning: Be careful about types!

- In many places in this assignment, you will have strings and labels
  - A string contains the text that you want to appear
  - A GLabel is the actual thing that is showing on the screen
  - They are different!! You can't concatenate two GLabels together, and you can't set the color of a string
- Be careful about how you name these variables
  - I use names like dashedStr and dashedLabel to clearly distinguish between types

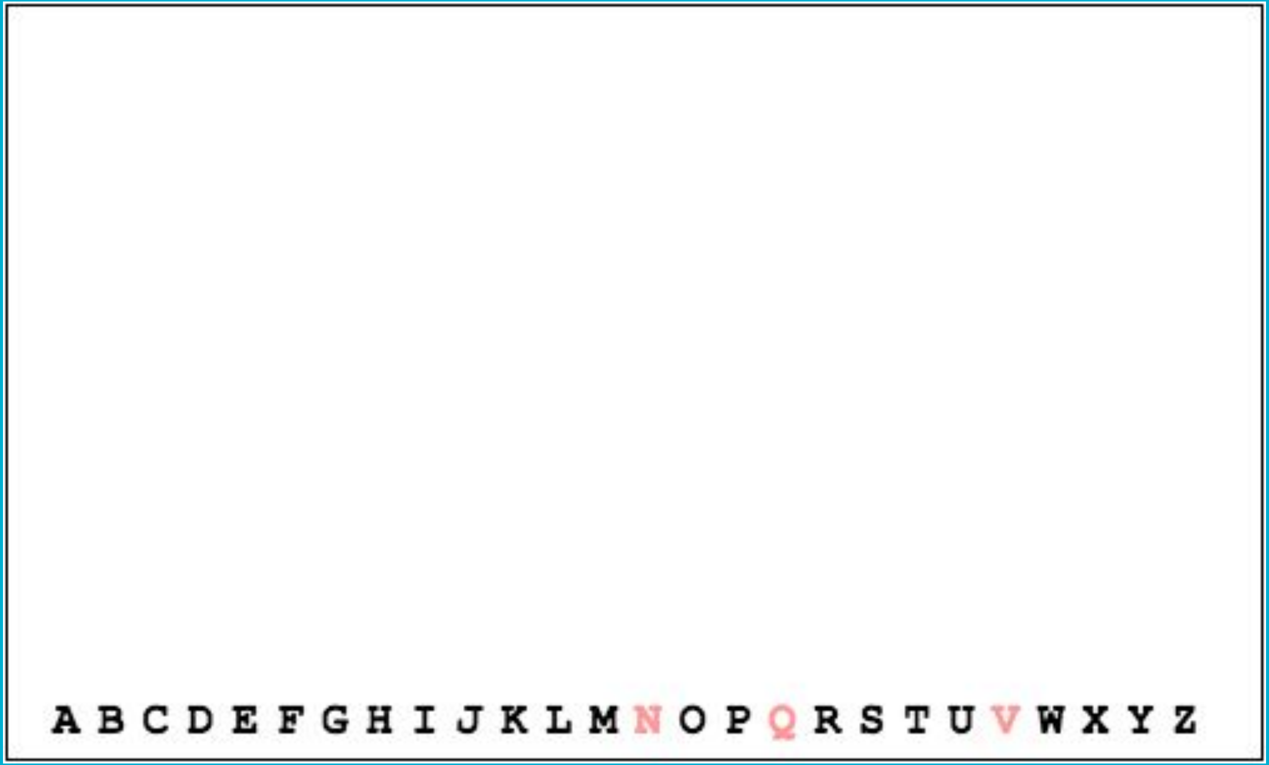
Milestone 1: Display letters at bottom of window


A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Milestone 1: Display letters at bottom of window

- Create many GLabels, one for each letter
    - How do you get the letter for each label?
    - You can create an ALPHABET string and get the letter at index `i`, or use character codes to generate the character for each loop iteration
- Set the font on the GLabel to a monospaced font
    - An example of setting the label on a font:
      `label.setFont("bold 20px 'Courier'");`
    - Be sure to use the POINTSIZE constants for the font size
- Position the labels so they are centered across the screen
    - You can get the width of a label as `label.getWidth()`. Note that if you use a monospaced font, all letter labels will have the same width
    - You can declare a spacing constant and center the letters like you centered the bricks in Breakout

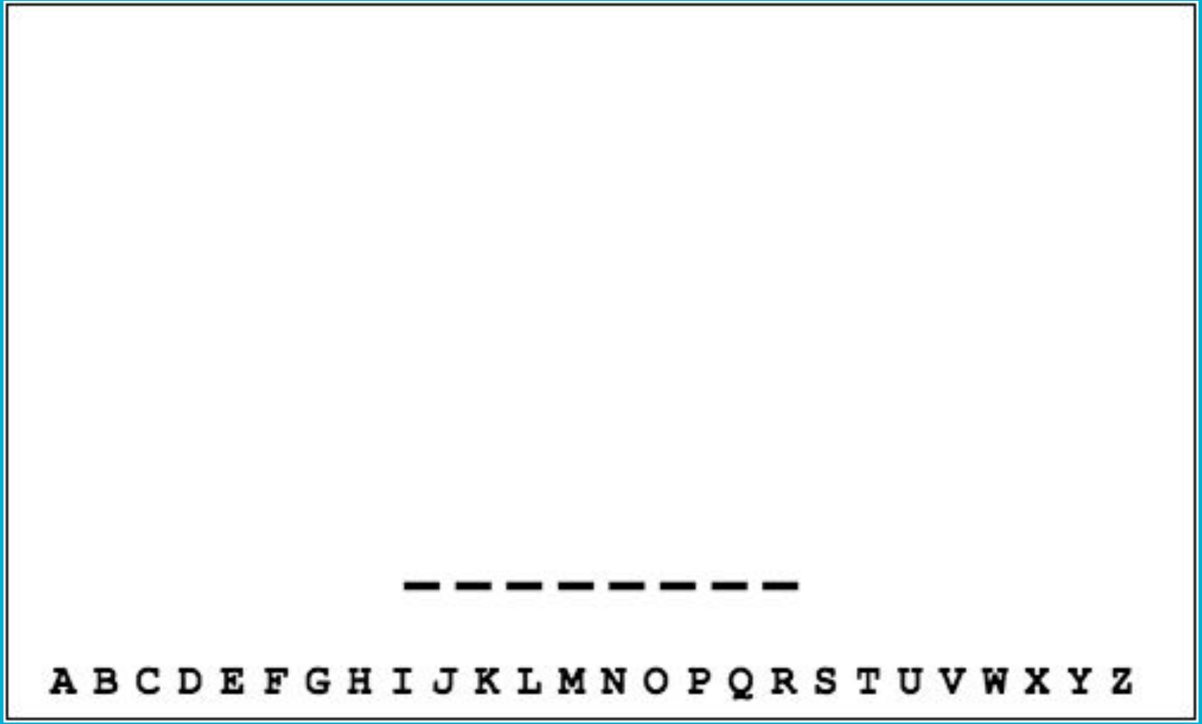
Milestone 2: Detecting mouse clicks on letters



Milestone 2: Detecting mouse clicks on letters

- When the user clicks a label, set the label's color to INCORRECT_COLOR
- How can we tell what the user clicked?
  - let label = gw.getElementAt(e.getX(), e.getY())
  - Make sure that the user actually clicked something (and didn't just click empty space on the screen!)
- How can we change the color?
  - Use label.setColor(...)
- Caveat: Later in the program, we will be drawing Karel body parts and other labels, and we don't want to handle clicks on those letters
  - Define a constant that is the y coordinate of the top of the letters, and only respond to clicks below that coordinate

Milestone 3: Choose a random secret word and display it in its hidden form



```
- - - - - - - -

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
```

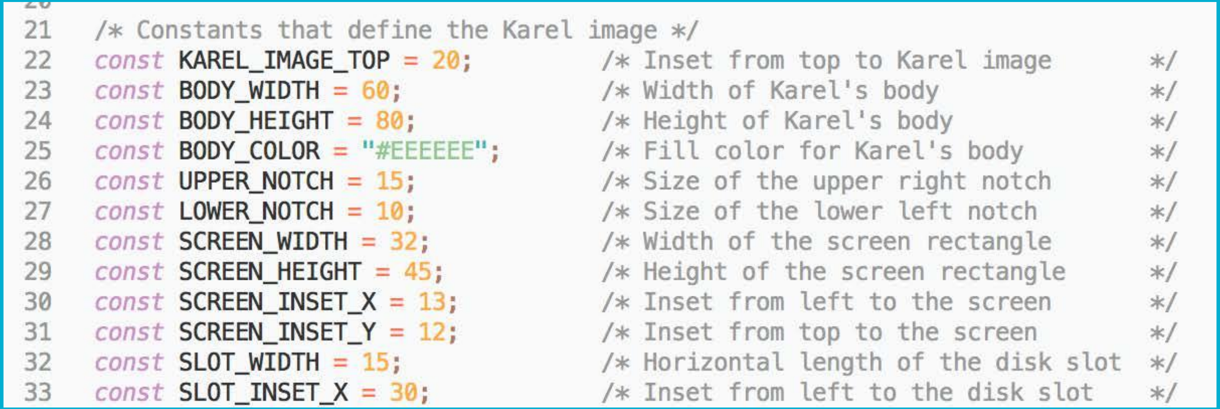
Milestone 3: Choose a random secret word and display it in its hidden form

- At the beginning, choose a random word that the player is supposed to guess (the code for doing this is provided for you in the handout)
- You need to take this string and convert it into a string of dashes, to hide the word from the user (but show its length and the correctly-guessed characters)
  - For each letter in the original string, add a dash to some new string
- Create a GLabel from the hidden string, set it to a monospaced font, and add it to your window

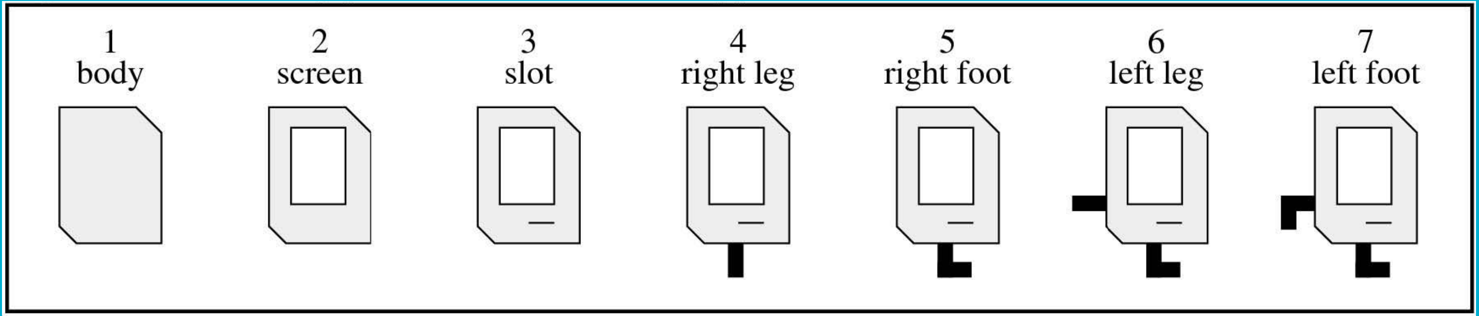
Milestone 4: Implement the code that updates correctly guessed letters

- Check whether the guessed letter is correct or not
- If it is correct, generate a new hidden word string that now shows the guessed letter
    - There are many ways to do this, and this is one of the most important conceptual parts of the assignment, so I'm going to avoid giving a solution. Think hard!
- Update the hidden word label using the new string
    - label.setLabel(text)

Milestone 5: Draw successive body parts of Karel for each incorrect guess



```
/* Constants that define the Karel image */
const KAREL_IMAGE_TOP = 20;            /* Inset from top to Karel image      */
const BODY_WIDTH = 60;                 /* Width of Karel's body              */
const BODY_HEIGHT = 80;                /* Height of Karel's body             */
const BODY_COLOR = "#EEEEEE";          /* Fill color for Karel's body        */
const UPPER_NOTCH = 15;                /* Size of the upper right notch      */
const LOWER_NOTCH = 10;                /* Size of the lower left notch       */
const SCREEN_WIDTH = 32;               /* Width of the screen rectangle      */
const SCREEN_HEIGHT = 45;              /* Height of the screen rectangle     */
const SCREEN_INSET_X = 13;             /* Inset from left to the screen      */
const SCREEN_INSET_Y = 12;             /* Inset from top to the screen       */
const SLOT_WIDTH = 15;                 /* Horizontal length of the disk slot */
const SLOT_INSET_X = 30;               /* Inset from left to the disk slot   */
```

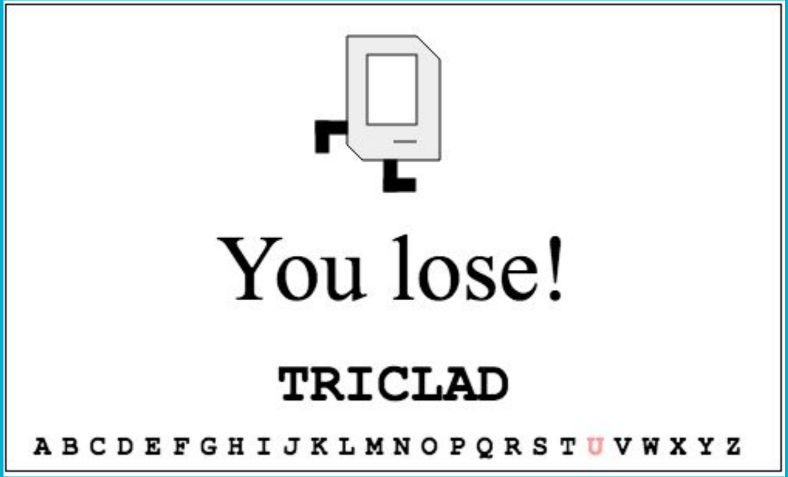


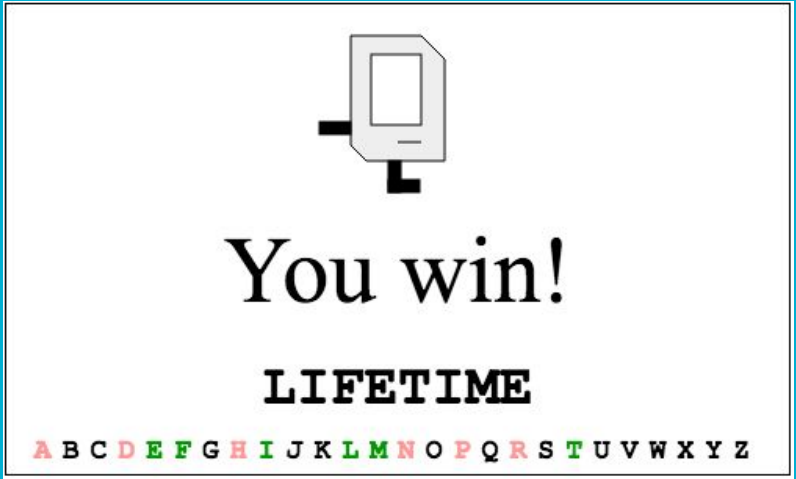
Milestone 5: Draw successive body parts of Karel for each incorrect guess

- Use GPolygon, GRect, and GLine to create the shapes of Karel's body parts
- Make sure to decompose this code. You'll probably end up writing a function for each body part

Milestone 6: Determine when the game is over and display appropriate message

- How can you tell if the user has won the game?
    - Hint: what will the hidden (dashed) string look like?





Extensions

- Be sure to submit a separate file with your extensions
- Possible ideas:
    - Guard a user from guessing the same incorrect letter multiple times
    - Make the graphics fancier (possibly check out the GImage class)
    - Add animations to the graphics
    - Adapt the program for a similar (but different) game