UNIVERSITY OF HULL

Department of Computer Science and Technology

Applicant Experience - Asteroids in Processing Sheet 3: Text

Now that we're able to clear the screen, and draw elements in new places and update. We can start thinking about other aspects of the game we eventually want to make. Part of almost every computer program is some kind of interface. So far, we've been dealing with visuals - our circle and the background - but most of the time we want to convey meaning to the user. In video games, this is usually statistics such as score, or health; however, all of the computers you use right now have a rich interface which needed programming.

Circles are relatively easy to create. It's a single point, and we fill in all points around it up to some radius, r. Text is a different kind of problem. Thankfully, most computer programming languages now make things easy.

Task 6 - We're now going to look at how we can write things on our main screen, rather than in the console. For this we're going to make a classic Frames Per Second counter.
(You may find https://processing.org/reference/text_.html a useful resource)

To do this, we're going to use `text()`, `textSize()`, and `fill()` for displaying text, and something else called `frameRate` for the actual value to put in. You guessed it, Processing gives us these out-of-the-box to use. It makes life a lot easier when others have done all the complex heavy lifting and mathematics for us. Let's explain what each of these does.

textSize() - This tells Processing how big the text we're about to draw is. Therefore this comes before we draw any text to the screen. If we don't do this Processing will assume a font size for us (Just like it assumes an actual Font to use).

E.g `textSize(32);`

text() - This function accepts some text (between " ") just like our `print()`, and `println()` functions. However, we need to specify **where** to draw the text, this is done by an x-coordinate, and a y-coordinate for where the text begins.

E.g text("The quick brown fox jumps over the lazy dog", 10, 10); // Top left position 10px away from origin.

fill() - This just tells Processing what colour to make the text we're about to draw. Again, this needs to be before actually putting any text to screen.

frameRate - This **variable** (just like width, heigh, mouseX, mouseY) can give us a value for the number of frames per second that we're drawing.

Putting it all together we want the following:

1. Optionally set the textSize, let's try 32pt size.
2. Set the fill colour, 255 should suffice.
3. Construct the text we want. E.g "Frame Rate: " + frameRate.
    a. This will add the number as text at the end of our "Frame Rate: " text.
4. Pass this constructed text to the text() function, specifying an x and y position to place our text. We'll use (10,10).

UNIVERSITY OF HULL

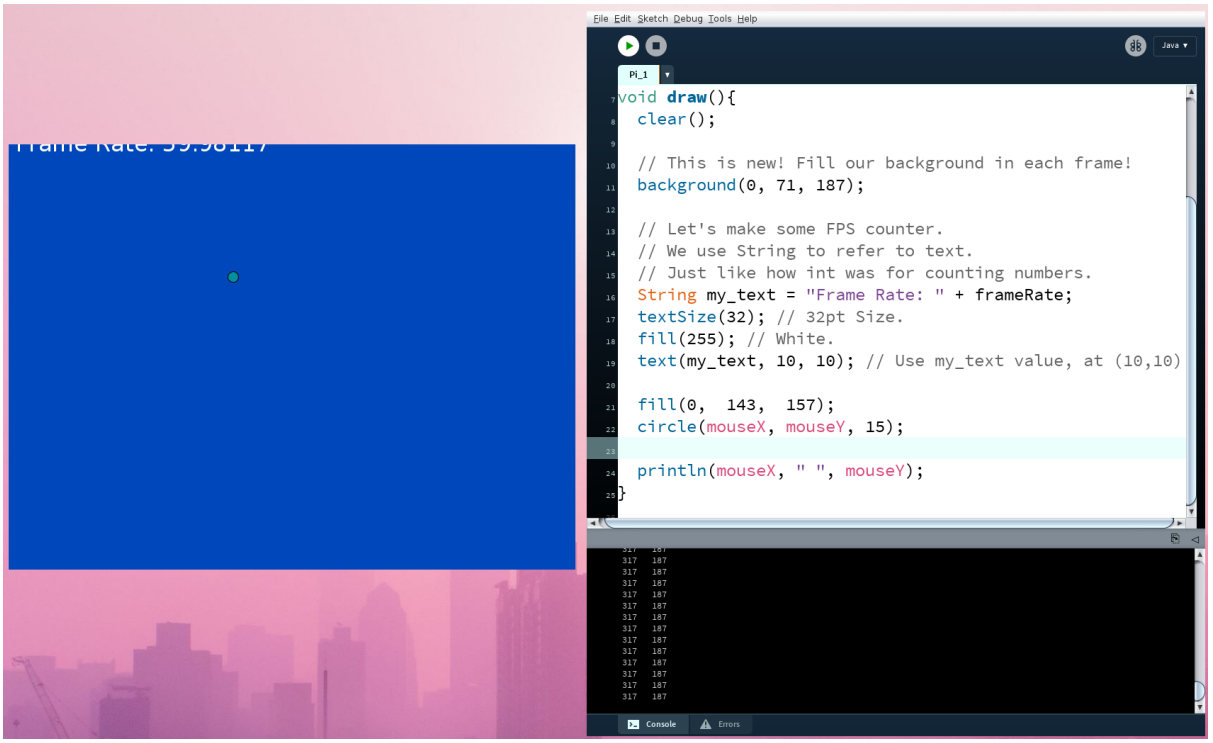
```
void draw(){
  clear();

  // This is new! Fill our background in each frame!
  background(0, 71, 187);

  // Let's make some FPS counter.
  // We use String to refer to text.
  // Just like how int was for counting numbers.
  String my_text = "Frame Rate: " + frameRate;
  textSize(32); // 32pt Size.
  fill(255); // White.
  text(my_text, 10, 10); // Use my_text value, at (10,10)

  fill(0,  143,  157);
  circle(mouseX, mouseY, 15);

  println(mouseX, " ", mouseY);
}
```



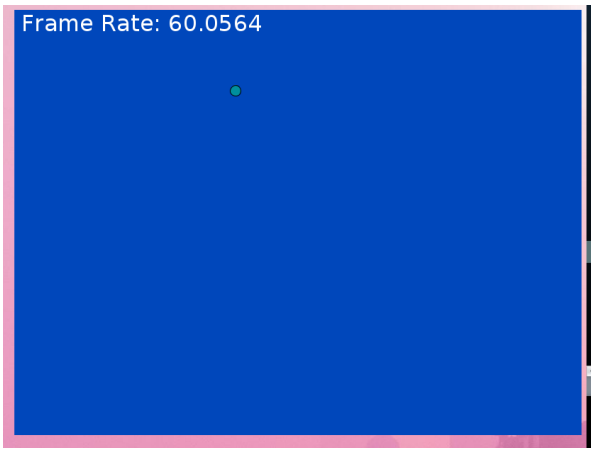
UNIVERSITY OF HULL

Task 7 - We were so close. But there are a few issues here:

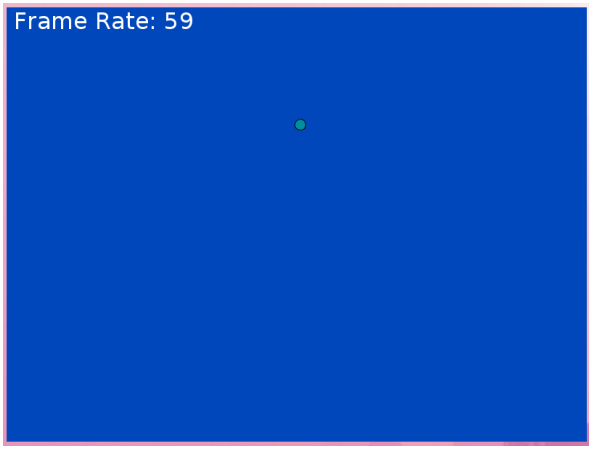
- The text is off the screen! It's a nice distance from the left side, but we're missing half the line off the top!
- The number (what we can see of it), is in the format 59.98117. Which, whilst accurate, is perhaps not the most user-friendly number to read, and it starts taking up most of our screen. As humans 59 or 60 would be a good way to represent that.

Time to fix it! Perform the following:

1. Change the y-coordinate of our text() call to something higher. Remember, our y-coordinate is the distance from the top of the screen (0). You may need to re-run the sketch multiple times to find some numbers you are happy with.
2. The harder task. Recall how we used (int) to force something to be a counting number. E.g random(n) produces values from 0 up to (and excluding) n. This comes out with decimal points (what we call 'floats'). Add (int) to the front of frameRate.




```
String my_text = "Frame Rate: " + (int)frameRate;
```



UNIVERSITY OF HULL

Sub-Task (Things to make you think)

1. What happens if we put `fill()`, or `textSize()` AFTER our `text()` call?
2. Should we draw our UI text before, or after we draw our circle? Try moving your cursor over near the text. Does your circle go behind, or in front of the text? Which way do you think makes the most sense? Try both out and see the differences, this can be done by just moving the lines of code down after our `circle()` call. Remember, our code will execute top-down in order for each block.

Text is a really useful way to convey information. Certain “debug” modes in videogames help developers understand what’s going on in their program. It can provide an overlay of information to peer into the inner workings.

For example, perhaps you’d like to experiment with different things you can display on your screen. What about your mouse coordinates, so you can ensure things are placed correctly? Feel free to move the text around, change the length of it. What would happen if my text was really long? Does it go off the screen? These are all considerations that people have to think about when writing programs themselves (and it’s not always an easy solution!).