Word Processing

You can add text with fonts, thats pretty cool! What about if you want to actively add text to your sketch. Enter the data type called a string. This allows us to add characters, subtract characters using our keyboard!

This was typed in processing!

PROCESSING
SKILL REQUIREMENTS
DIFFICULTY 1-5

ELECTRICAL PROTOTYPING	○○○○○
ROBOTICS	○○○○○
SOLDERING	○○○○○
PROGRAMMING	●●○○○
DIY	○○○○○

MATERIALS LIST

- Computer
- Processing
- Graph Paper
- Colored Pencils
- Super fine tip sharpie

STEP 1: Keyboard Variables

We have been using a keyboard this whole time to write code. We are now going to look at how to use the keyboard as an input device like we have been using the mouse. There are a few built in variables around the keyboard, a couple of them require an `if()` statement to be really useful. Here is the list of the variables and their descriptions:

`key` -returns the ASCII value of that key that was pressed when used as a number but also can be used as a character input.
`keyCode` -returns non-ASCII value of the key pressed such as the up arrow, enter, etc.
`keyPressed` -returns a boolean (true/ false) if a key, any key is pressed.
`CODED` -Used to check if a key is coded (non-ASCII) or not coded (ASCII)

STEP 2: Keyboard Functions

There are also a number of keyboard functions as well! Just like the mouse functions these are event functions and have their own void. To stress the point, event functions allow for clean code and allow you to interrupt the top down flow of the code and have something happen “out of order” Here are a few keyboard event functions:

`keyPressed()` -Event triggered when a key on the keyboard is pressed (ASII or non-ASII)
`keyReleased()` -Event is triggered when a key on the keyboard is released (ASII or non-ASII)
`keyTyped()` -Event is triggered when a key is typed (pressed and released)

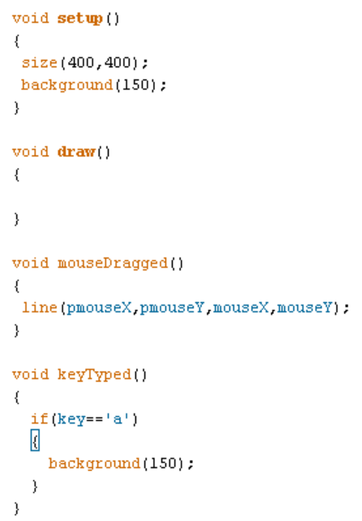
Remember that each one of these should have a void in front of them and their own set of curly brackets.

STEP 3: Keyboard Event Function in action!

Probably the most useful of the event functions for the keyboard is the `keyTyped()` function. The others are also pretty self explanatory if you got a handle on event functions with the mouse. We used the same example code from the Processing the Mouse HotSheet but we replaced the `mouseClicked()` event function with the `keyTyped()` event function. The `if()` statement inside the event checks what letter was pressed and if it is the lower case ‘a’ it redraws the background.

We need to note that when you are checking a key value in terms of characters you use the double equal sign and the letter or character you are looking for is in ‘ticks’ the single ticks note a character vs. a string which is in “quotes”. You can check for capital letters as well!

This setup will only look for ASCII characters and not non-ASCII characters like the up arrow, etc. For that we need to add a layer of complexity and check if a key is coded!



```
void setup()
{
 size(400,400);
 background(150);
}

void draw()
{

}

void mouseDragged()
{
 line(pmouseX,pmouseY,mouseX,mouseY);
}

void keyTyped()
{
  if(key=='a')
  {
    background(150);
  }
}
```

STEP 4: Key==CODED???

If you want to be able to use keys that are not ASCII keys such as up, down, left, right, etc. then you need to be able to check if the key you type is coded or not. We can do that using an `if()` statement to check if its coded and then a series of other if statements to check for the keyCode of that key. See the example for the syntax of how to setup coded keys for use. Here are some basic coded keys you can use:

`UP,DOWN,LEFT,RIGHT,ALT,CONTROL,SHIFT`

STEP 5: DIY Word Processor?!

With what you have learned by now you can create your own basic word processing application in Processing. AWESOMENESS!!! Remember the concept of a string? We have briefly talked about them on and off. A string is just that a string of characters, and we can add characters to a string. Where do we get those characters? From the keyboard!

We have the basic code to the right, but we will step you through it the best we can! First of all we create a String object and name it. We also say that there is nothing in the string for now. We then set our `size()` and `background()`. In our `draw()` code we have 2 functions the `fill()` for the text color and the text function. The data for the text is going to be our string data, and we are placing it in the upper left hand corner.

The last step is to start adding characters to our string. We do that using the `keyTyped()` event function. When a key is typed it will take what is currently in the string object and add the character key that was just pressed! On its own, this code reads spaces, tabs, and carriage returns (Enter), but it does not delete anything! Change and add this code to make it your own! add a font, font size, background colors etc.

```
String myString= "";

void setup()
{
 size(400,400);
 background(255);
}

void draw()
{
fill(0);
text(myString,10,15);
}


void keyTyped()
{
  myString=myString + key;
}
```

STEP 8: Share your work!

If you would like to share your work...which you do! You can sign up at openprocessing.org. Open Processing allows you to share your code, view the drawing online, as well as allows others to fork or borrow your code to manipulate and change it on their own while still keeping your sketch intact for others to check out.

OpenProcessing

TAKING IT FURTHER

- Check out some examples of Processing at www.processing.org/exhibition/
- What other geometry could we draw using Processing...Hint: Reference!
- Draw a second line going from the upper right corner to the lower left corner
- change the size of your window!