basil.js Cheatsheet v2

based on basil.js v2 dev 20180924

JAVASCRIPT

```
var myNumber = 5;  // integer or whole number
var myNumber = 3.14;  // floating-point decimal number
var myText = "Hello World";  // string of text
var myChar = 'a';  // single character
var mySwitch = true;  // boolean (true or false), used for if statements
var obj = *basil.js function*;  // use var's in code to reference items later in code

// this is a single line comment  // won't be executed
/*
   this is a multiline comment.
   nothing between here will be run or executed
*/
```

META

```
size("A4", LANDSCAPE);  // resize document to known sizes, can also feed exact values
canvasMode(MARGIN);  // limit coordinates to PAGE, MARGIN, BLEED, FACING_PAGES, ....
units(MM);  // set unit system to MM, IN, CM, PX, PT
colorMode(CMYK);  // use RGB (0-255 values), or CMYK (0-100)
guideX(200);  // new guide at the given x position, there's also guideY();
layer("output");  // create/set layer for generated items
doc();  // returns the current document
page();  // returns the current page. with a number, goes to that page
clear();  // clear page, layer, document
remove();  // remove pageItem, page, layer, swatch, etc.
width  // variable refers to canvas width, use 'width/2' for horizontal center
height  // variable refers to canvas height, use 'height/2' for vertical center
```

STYLE

```
color(255, 255, 0);  // create color in RGB, use color(255) for grayscale, see colorMode(CMYK)
fill(255, 0, 0);  // similar to color, but applies it directly as fill, can also pass color() variable
fill("blah");  // fill using predefined swatch
noFill();  // removes fill
stroke(150);  // set stroke to gray, similar as color() and fill() above
noStroke();  // removes stroke
strokeWeight(5);  // set thickness of stroke
rectMode(CENTER);  // draw from CENTER or CORNER (default), see ellipseMode(), imageMode()
opacity(obj, 50);  // set opacity of object, 0 - 100
property(obj, "fillColor", value);  // post-style change, see Jongware for list
applyObjectStyle(obj, style);  // pass it pageItem and style as var or "name"
```

SHAPES

```
line(x1, y1, x2, y2);  // draw line from x1, y1 to x2, y2
rect(x, y, w, h);  // draw rectangle at given postition and size
ellipse(x, y, w, h);  // draw ellipse at given postition and size
beginShape();  // start complex shape
    vertex(x, y);  // use as many as needed
endShape();  // end complex form. use endShape(CLOSE); to automatically close shape
```

TYPOGRAPHY

```
textFont("Helvetica", "Bold");  // set font family and cut – these go before text()
textSize(48);  // set text size
textAlign(Justification.CENTER_ALIGN);  // see reference for options
text("text", x, y, w, h);  // create text block at given position and size
typo(obj, "pointSize", 72);  // post-style change, see Jongware + typo cheatsheet for list
paragraphs(obj);  // returns array with all paragraphs of text in obj, see Modifying Type tutorial
lines(obj);  // returns array with all lines of text in obj
words(obj);  // returns array with all words of text in obj
characters(obj);  // returns array with all characters of text in obj
placeholder(obj);  // fill with placeholder text (lorem ipsum)
applyCharacterStyle(text, style);  // pass it text or block and style as var or "name"
applyParagraphStyle(text, style);  // pass it text or block and style as var or "name"
```

SELECTION

```
selection();  // returns single selected item
selections();  // returns array of selected items
nameOnPage(name);  // returns first item on active page with name in Layers window
labels(name);  // returns array of items on active page with name set in 'Script Label' window
items(page());  // returns array of items found on document, page, layer, group
```

TRANSFORMATION

```
transform(obj, "position", [x, y]);  // move pageItem to new position
transform(obj, "size", [w, h]);  // resize pageItem, or just "width" or "height"
referencePoint(CENTER);  // set reference point for any transformations
transform(obj, "rotation", 45);  // rotate pageItem
bounds(obj);  // returns object with left, right, top, bottom, width, height + baseline, xHeight for textFrame
```

RANDOM

```
random(100);  // generate a random number from 0 to 100
random(75, 100);  // generates a random number from 75 to 100
randomSeed(42);  // locks each request of random to that values 'gear' = consistant random
```

MATH

```
+ - * /  // add, subtract, multiply, divide = basic math operations
foo = foo + 5;  // value = it's current value + 5
foo += 5;  // same as above, but less code!
foo++;  // similar to above, however only adds 1 each time (also works with --)
round();  // convert a float into an int, normal rounding rules apply
floor();  // convert a float into an int, force rounding down
map();  // scale value from one range to another, ie: map(input, oldMin, oldMax, newMin, newMax);
abs();  // absolute value, useful when comparing two numbers with subtraction
```

CONDITIONALS

```
if(a > b){
        // executes this code if a is bigger than b
}else{
        // optionally this code will run
}
```

RELATIONAL OPERATORS

```
a == b  // a is EQUAL to b (note two == signs)
a != b  // a is NOT EQUAL to b
a >  b  // a is GREATER than b
a <  b  // a is SMALLER than b
a >= b  // a is GREATER or EQUAL to b
a <= b  // a is SMALLER or EQUAL to b
```

LOGICAL OPERATORS

```
if(a == b && b == c){...}  // AND / BOTH statements must be true
if(a == b || b == c){...}  // OR / EITHER statement must be true
```

LOOPS

```
for (int i = 0; i < 50; i++){  // abstract for loops: for( start; stop; counter ){...}
   println(i);  // this code runs x times (50 in this case)
}
```

INPUT

```
loadStrings("data.txt");  // load data, ideally in 'data' folder next to InDesign document
files(folder("~/Pictures"));  // load all files within a given folder
image("name.jpg", x, y, w, h);  // x can be rect/ellipse/polygon, then ignore y, w, h
```

OUTPUT

```
savePDF(timestamp() + ".pdf");  // add ', true' to adjust export settings
savePNG(timestamp() + ".png");  // add ', true' to adjust export settings
println(foo);  // print value to the console, used to debug variable's value
```