




Variables and Functions

+ Entry Survey

- 18 of you filled out the survey



+ Questions about me

- Where did you grow up?
- How did you get started in computer science/programming? What is your computer science background?
- What aspect of programming are you most interested in?
- Why do you teach? What is the most important thing to have in a classroom?
- What kind of experience and work have you done in computer science?
- What are your research interests and what drew you to this type of research?
- What was your most proud accomplishment in the field of computer science or programming.

+ Questions about me

- What is the most thrilling moment of success you've ever had, after having put a lot of effort into designing something or writing a program?
- Are you very good at math? Do you have to be especially skilled in math to learn computer science?
- Are you strict about minor errors in assignments?
- What advice would you give to a student who wanted to be successful in your program?

+ Your Experience

5

- Past programming
 - Processing (2/18)
 - JavaScript (2/18)
 - Java (1/18)
 - Python (5/18)
- Computers
 - web, email, word processing (83%)
 - use when I have to (5%)
 - spreadsheets and powerpoint (83%)
 - other math tools (28%)
- Class year
 - First (17%)
 - Second (44%)
 - Third or more (39%)
- Operating System
 - Mac (9/18)
 - Windows (8/18)
- Major
 - Computer Science?
 - Math
 - None/Undecided
 - Physics
 - Economics
 - Geology
 - Chemistry
 - History/Spanish

+ What you hope to learn

- To be able to write working scripts and how to be comfortable writing code
- discover if I want to major in Computer Science alongside Math. ■ improve at coding and learn more about other languages-preparation for a possible CS major
- I hope to learn more about how computers work to be able to design and create my own websites. ■ tools that will help me easily learn more about computing on my own.
- The basics of programming

+ What you hope to learn



- I hope to learn some more about Java and object oriented languages, and how to make art from computing.
- Basics of coding and an ability to replicate them in other languages.
- I hope to become comfortable enough in coding that I can easily learn many coding languages
- the basic logic of computers.
- To learn a coding language well enough to perform basic tasks.

+ What you hope to learn



- To know my way around a computer program and fix simple bugs.
- I would like to learn basic coding and how data structure works.
- scope of what a computer can create and do.
- something I could apply the theoretical knowledge I have learned in math and physics

+ Concerns

- Not being able to attend the computer labs due to conflict in schedule
- I am a complete beginner to computing, so I might need extra help through tutoring and/or office hours.
- I wonder what the process of transferring this to Haverford will be like
- I have heard that it is really hard, but I'm excited about the material covered and assignments.
- I'm worried about understanding how hexadecimal numbers, object-oriented languages, and scope work
- There should be more of a process for teaching concepts. I.e. one exercise completed by the whole class does very poorly to connect with many students and provides little replicable working understanding of the concepts at hand.

+ Concerns

- I am very bad at memorizing, so I need a lot of time to practice and tinker. I am worried the class isn't structured such that I will be able to keep up, and that I will perform very badly on tests.
- I am concerned that the course may move at a fast pace. I have almost no programming experience besides a bit of Java from high school, which I have long since forgotten. I am also not a particularly artistic person.
- none
- I hope that my lack of experience will not be a hindrance, and that I will be able to wrap my head around Processing. I am, however, enthusiastic!
- I'm not really good at math...
- I had very little experience with programming before.
- I have never used any programming before and I have completely no idea about those languages.

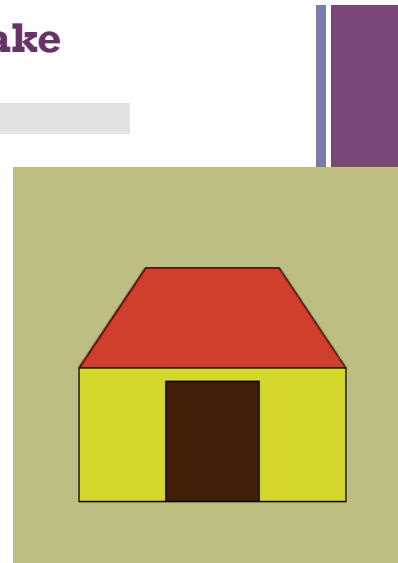
+What variables will make this scalable?

```
// Draw a barn
// Create and set canvas
size(300, 300);
smooth();
background(187, 193, 127);

// wall
fill(206, 224, 14);
rect(50, 150, 200, 100);

// Draw Door
fill(72, 26, 2);
rect(115, 160, 70, 90);

// Draw roof
fill(224, 14, 14);
quad(50, 150, 100, 75, 200, 75, 250, 150);
```



+What variables will make this scalable?

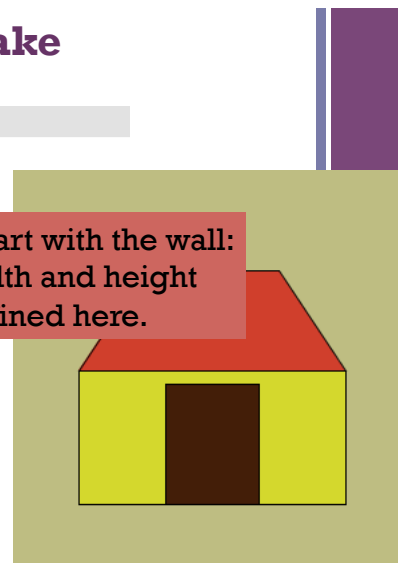
```
// Draw a barn
// Create and set canvas
size(300, 300);
smooth();
background(187, 193, 127);

// wall
fill(206, 224, 14);
rect(50, 150, 200, 100);

// Draw Door
fill(72, 26, 2);
rect(115, 160, 70, 90);

// Draw roof
fill(224, 14, 14);
quad(50, 150, 100, 75, 200, 75, 250, 150);
```

Let's start with the wall:
the width and height
are defined here.



+ What variables will make this scalable?

```
// Draw a barn
// Create and set canvas
size(300, 300);
smooth();
background(187, 193, 127);

// wall
fill(206, 224, 14);
rect(50, 150, 200, 100);

// Draw Door
fill(72, 26, 2);
rect(115, 160, 70, 90);

// Draw roof
fill(224, 14, 14);
quad(50, 150, 100, 75, 200, 75, 250, 150);
```

Let's start with the wall:
the width and height
are defined here.
We can define wallWidth
and wallHeight as ints:
int wallWidth;
int wallHeight;

+ What variables will make this scalable?

```
// Draw a barn
// Create and set canvas
size(300, 300);
smooth();
background(187, 193, 127);

// wall
fill(206, 224, 14);
rect(50, 150, 200, 100);

// Draw Door
fill(72, 26, 2);
rect(115, 160, 70, 90);

// Draw roof
fill(224, 14, 14);
quad(50, 150, 100, 75, 200, 75, 250, 150);
```

Let's start with the wall:
the width and height
are defined here.
We can define wallWidth
and wallHeight as ints:
int wallWidth;
int wallHeight;

How does drawing the
door change relative to
the width and height of the wall?

+ Drawing proportions

- When considering two shapes that are drawn in relation to each other, one shape can be positioned relative to the other shape.
 - We've already seen this when considering your display rectangle as the reference shape:
 - 0,0 is the reference point
 - displayWidth, and displayHeight are the reference width and reference height.
 - In this case the wall is the reference shape, and the door is relative to it.

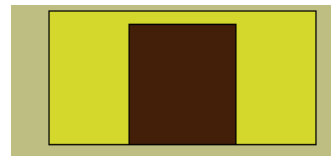


+ A Simpler barn (with wall as reference)

```
// reference values for the wall
int refX = 50; // the left side of the wall
int refY = 250; // the bottom of the wall
int refWidth = 200;
int refHeight = 100;

// wall
fill(206, 224, 14);
rect(refX, refY-refHeight, refWidth, refHeight);

// Draw Door
fill(72, 26, 2);
rect(refX + 0.3*refWidth, refY - 0.9*refHeight,
    0.4 * refWidth, 0.9 * refHeight);
```



+ Variables in functions

- A variable, just like a literal can be passed into a function.
- For instance:
 - `color(255,255,255);`
 - is the same as
 - `int full = 255;`
`color(full,full,full);`
- also:
 - `rect(3,3,20,20);`
 - is the same as
 - `int p = 3, s = 20;`
`rect(p,p,s,s);`

+ Arithmetic with **int** and **float** values

```
int x = 42;           vs    int x = 42.0;
float x = 42.0       vs    float x = 42;
float x = 7/2;       vs    float x = 7.0/2.0;
```

+ Arithmetic with **int** and **float** values

```
int x = 42;           vs   int x = 42.0;      // error
float x = 42.0       vs   float x = 42;          // same 42.0

float x = 7/2;       vs   float x = 7.0/2.0;    // 3.0 vs 3.5
```

- Type of variable is important and determines the value that can be assigned to it.
- Result of division depends upon operands

| | |
|-------------|--------------------------|
| int/int | yields an integer result |
| float/int | yields a float result |
| int/float | yields a float result |
| float/float | yields a float result |

+ Processing: Predefined Variables

- **width, height**
The width & height of the canvas used in the sketch

- **PI, HALF_PI, TWO_PI**
For different values of π . Note that

```
HALF_PI = PI/2
TWO_PI = 2*PI
```

- **displayWidth, displayHeight**
The width and height of the monitor being used. This is useful in running fullscreen sketches using:

```
size(displayWidth, displayHeight);
```

- **mouseX, mouseY**
The current mouse location in sketch (...coming soon!)

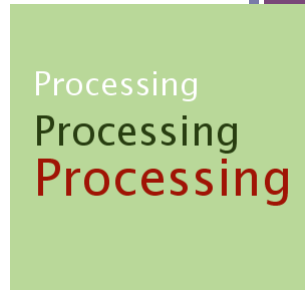
+ Extra: Drawing Text

`text(string, x, y);`
 Draws string with bottom left corner at
 x, y

`textSize(fontSize);`
 Can be used to specify font size

`fill()` can be used to specify color

See Reference for using fonts and other
 options.



```
size(300, 300);
background(185, 216, 153);

textSize(32);
text("Processing", 25, 100);
textSize(40);
fill(40, 62, 17);
text("Processing", 25, 150);
textSize(50);
fill(160, 20, 5);
text("Processing", 25, 200);
```

+ Mathematical Functions

$$y = f(x)$$

$$y = \textit{twice}(x) = 2x$$

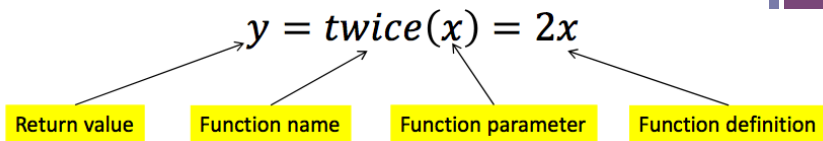
$$a = \textit{area}(\textit{radius}) = \pi r^2$$

$$y = f(x) = \begin{cases} 1 & \text{if } x > \theta \\ 0 & \text{otherwise} \end{cases}$$

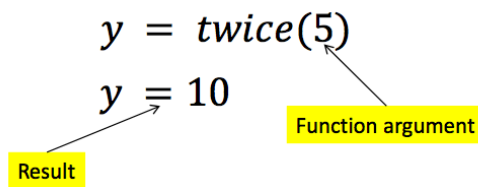
$$y = \textit{sum}(n) = \sum_{i=1}^n i$$

+ Functions: Terminology

23



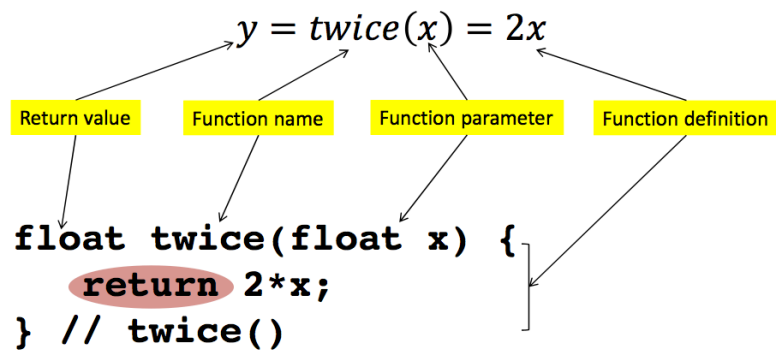
Function application:



GXX2013

+ Processing: Defining Functions

24



GXX2013

+ Processing: Defining Functions

25

Syntax:

```
returnType functionName(parameters) {
    ...
    return expression;
}
```

Example:

```
float twice(float x) {
    return 2*x;
} // twice()
```

Use:

```
y = twice(5);
```

GXX2013

+ Defining Functions: void

26

Use **void** as *returnType* when no value is returned.

Syntax:

```
void functionName(parameters) {
    ...
    return;
}
```

Example:

```
void square(float x, float y, float side) {
    rectMode(CORNER);
    rect(x, y, side, side);
} // square()
```

Use:

```
square(50, 50, 100); // draws a 100x100 square at 50, 50
```

GXX2013

+ Program Structure: Functions

27

```

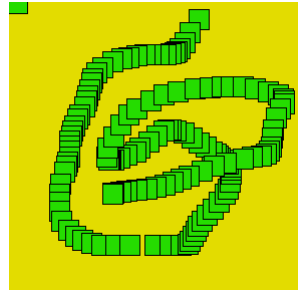
color color1 = color(227, 220, 0);
color color2 = color(37, 220, 0);

void setup() {
  // create and set up canvas
  size(300, 300);
  smooth();
  background(color1);
} // setup()

void draw() {
  fill(color2);
  square(mouseX, mouseY, 20);
} // draw()

void square(float x, float y, float side) {
  rectMode(CORNER);
  rect(x, y, side, side);
} // square()

```



GXX2013

+ Variables & Scope

28

```

color color1 = color(227, 220, 0);
color color2 = color(37, 220, 0);

void setup() {
  // create and set up canvas
  size(300, 300);
  smooth();
  background(color1);
} // setup()

void draw() {
  fill(color2);
  square(mouseX, mouseY, 20);
} // draw()

void square(float x, float y, float side) {
  rectMode(CORNER);
  rect(x, y, side, side);
} // square()

```

Global Variables

**Either pre-defined
Or defined at top**

**Are visible everywhere
In the program**

GXX2013

+ Variables & Scope

29

```

color color1 = color(227, 220, 0);
color color2 = color(37, 220, 0);

void setup() {
  // create and set up canvas
  size(300, 300);
  smooth();
  background(color1);
} // setup()

void draw() {
  fill(color2);
  square(mouseX, mouseY, 20);
} // draw()

void square(float x, float y, float side) {
  rectMode(CORNER);
  rect(x, y, side, side);
} // square()

```

Local Variables

**Either
parameters
Or defined
inside blocks**

**Are visible
ONLY
in the block
After they are
defined**

GXX2013