PennDraw

Learning Objectives

- Write a program that draws a specified image
- Judge how lines of code will affect the program output

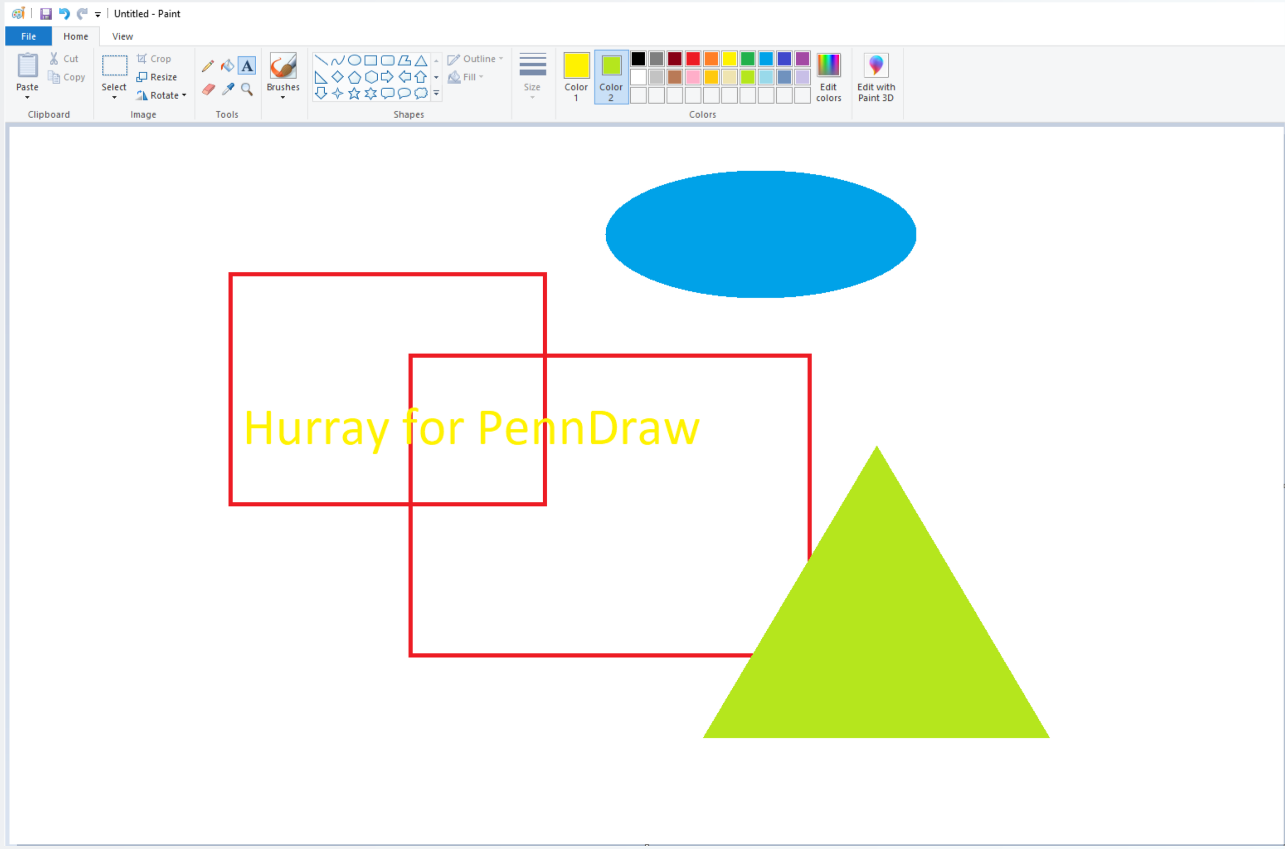
What is PennDraw?

PennDraw

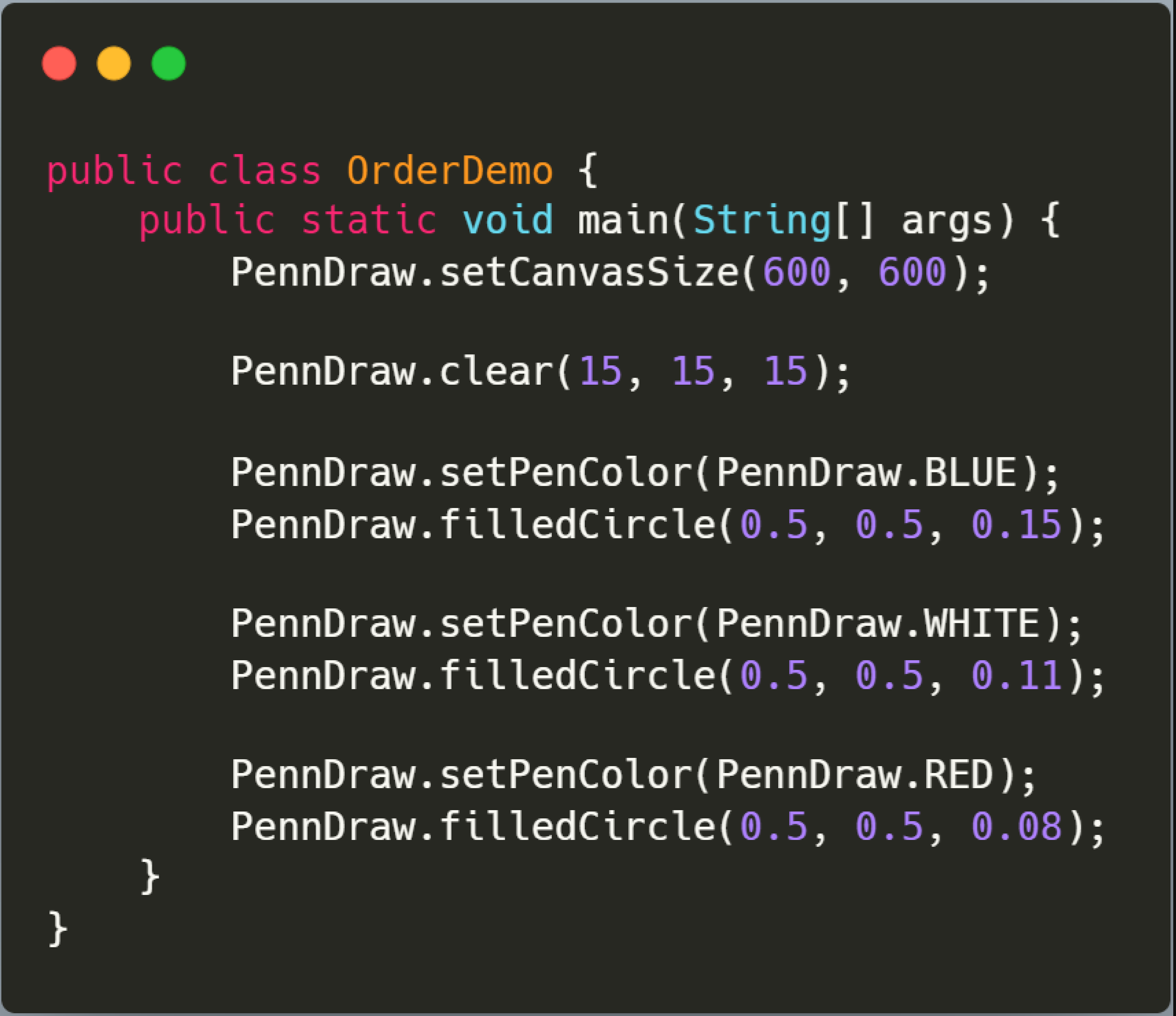
- The name of a group of related drawing tools available for you to use.
    - Adapted from a library called “StdDraw” if you see that anywhere
- Any time we need to draw to the computer’s screen in CIS 1100, we’ll use PennDraw.
- You can access a full listing of PennDraw’s features on the page for **PennDraw** on the course website

PennDraw: a programmable Microsoft Paint

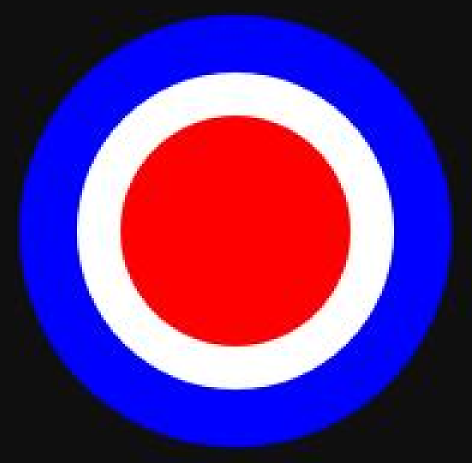
- Uses a set canvas
- Has an imaginary “pen”
    - The pen has a color setting and a weight setting.
- Draw shapes
    - Rectangles, ellipses, arbitrary polygons
- Draw text



PENNDRAW



```
public class OrderDemo {
    public static void main(String[] args) {
        PennDraw.setCanvasSize(600, 600);

        PennDraw.clear(15, 15, 15);

        PennDraw.setPenColor(PennDraw.BLUE);
        PennDraw.filledCircle(0.5, 0.5, 0.15);

        PennDraw.setPenColor(PennDraw.WHITE);
        PennDraw.filledCircle(0.5, 0.5, 0.11);

        PennDraw.setPenColor(PennDraw.RED);
        PennDraw.filledCircle(0.5, 0.5, 0.08);
    }
}
```

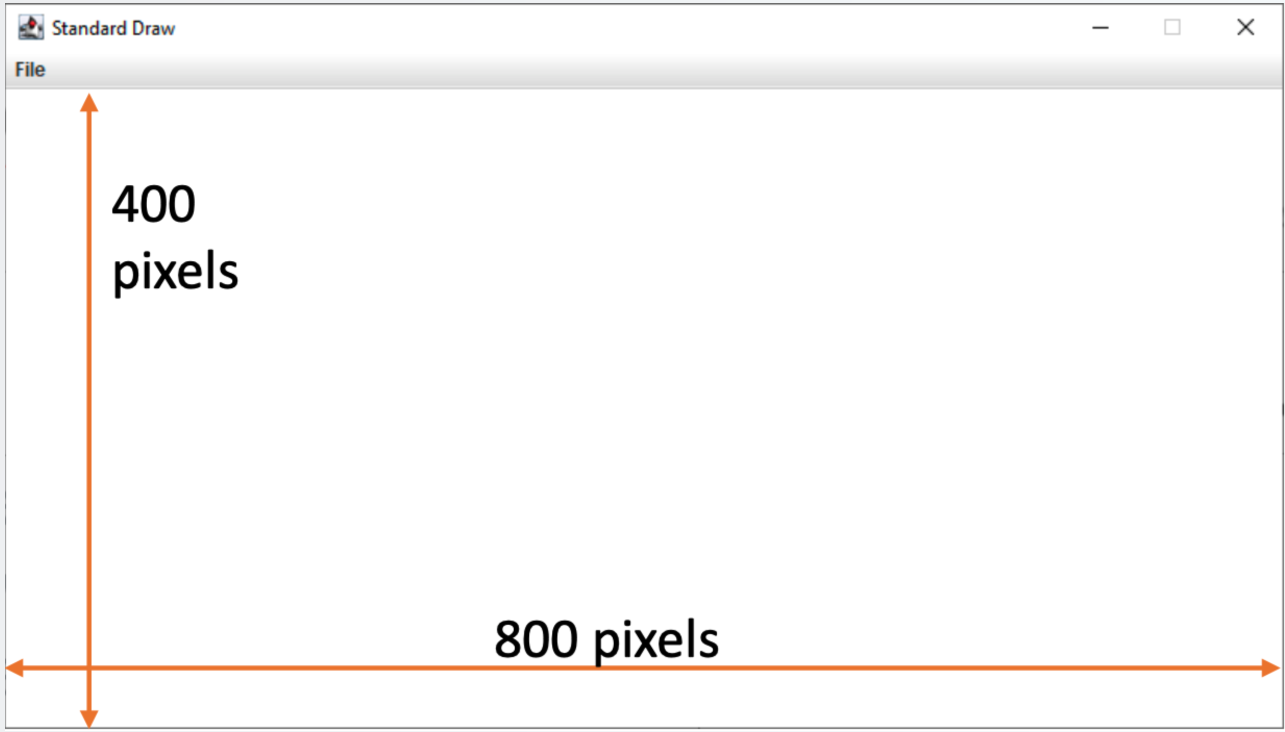


The Canvas

The *canvas* refers to the window of space on which PennDraw can do its drawing.

It has a width and a height, both defined in pixels.

- We usually express the size of a canvas like "*width* by *height*"
- Width is the *x dimension*
- Height is the *y dimension*

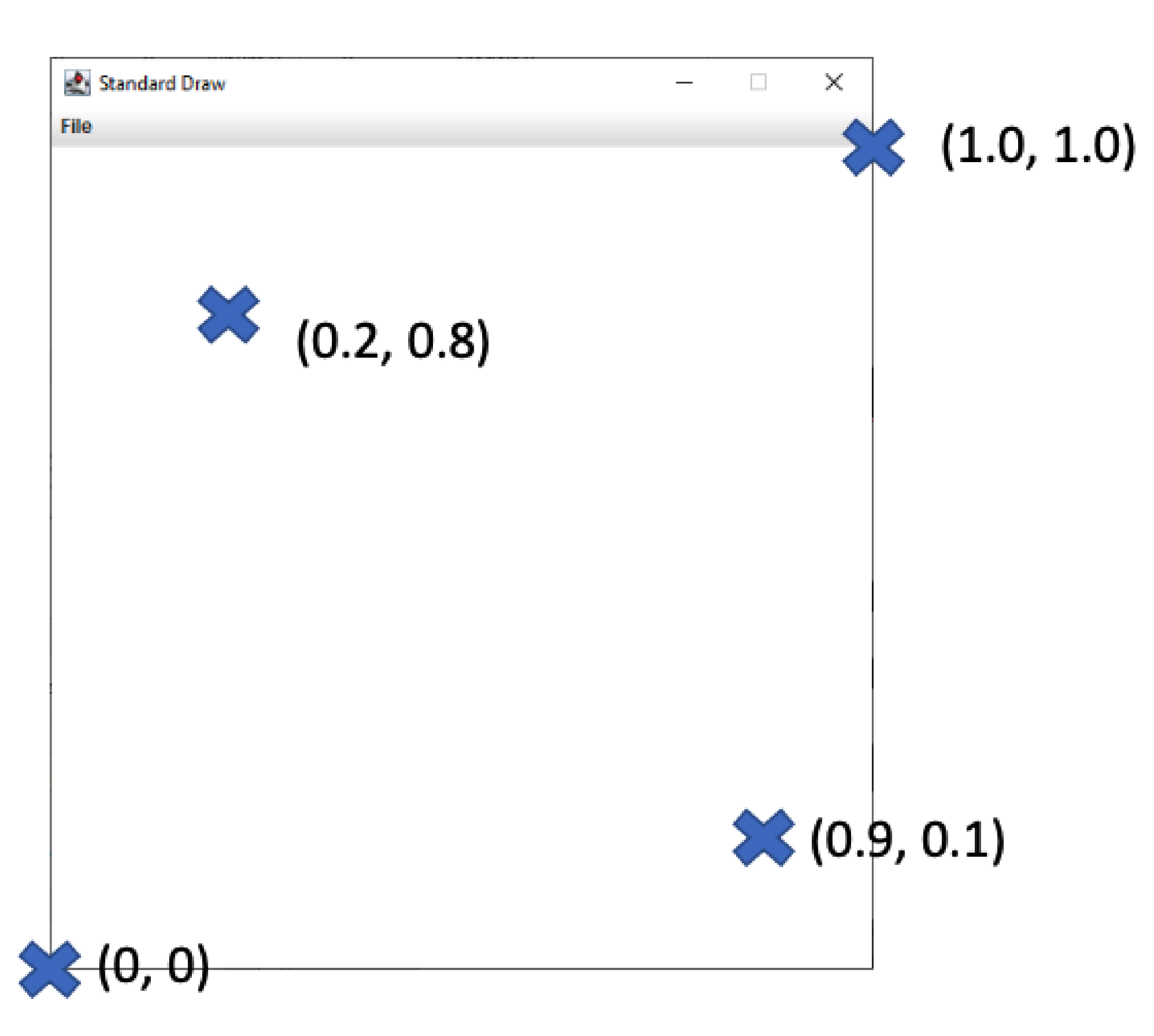


The Coordinate System

Canvas positions are accessed using coordinates.

By default, coordinates range from 0 to 1 in both the x dimension and the y dimension.

- The coordinate (0,0) refers to the bottom left position of the canvas.
- Coordinate (1,1) is found at the top right of the canvas.



The Pen

PennDraw works in a model where the programmer (you!) gives a series of instructions, one by one, to a computer

Some instructions are responsible for changing how shapes will be drawn

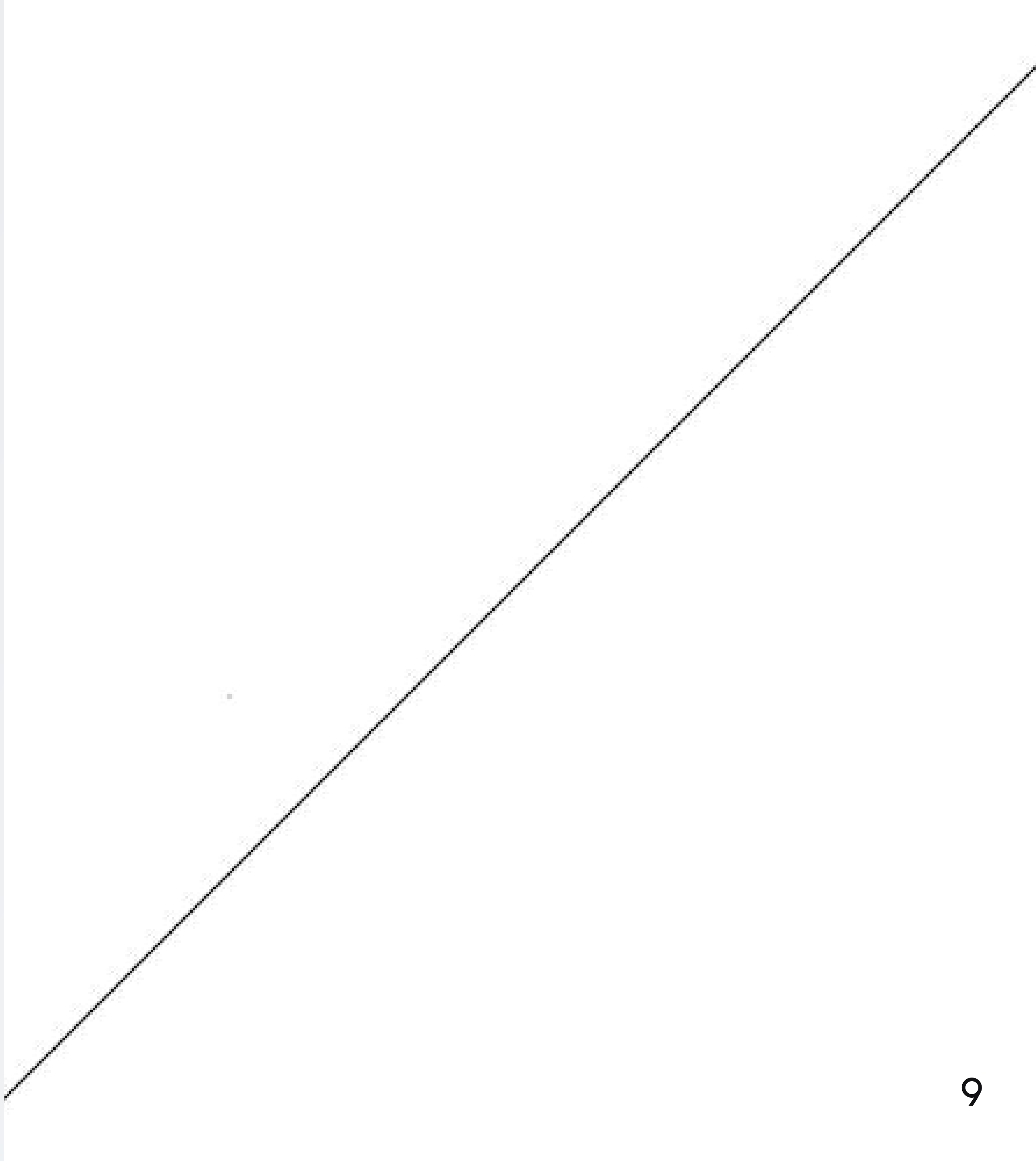
- “changing the settings of the pen”
- Settings include radius and color

The instructions change the pen settings until the next time the settings are explicitly modified.

Radius

Whenever we ask PennDraw to draw e.g. a point or line on the screen, these marks will appear with a certain thickness determined by the current setting for the radius of the pen.

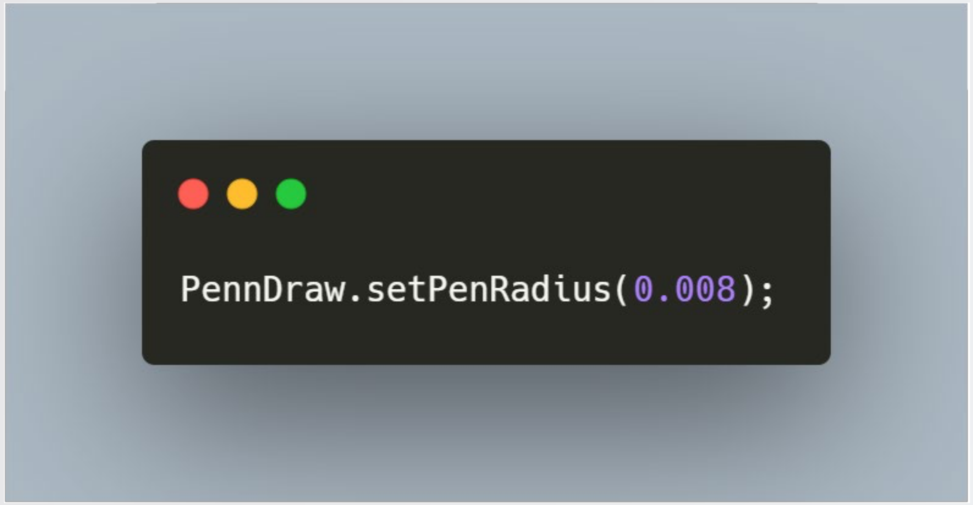
Pictured: a point and a line drawn with a default radius setting of 0.002



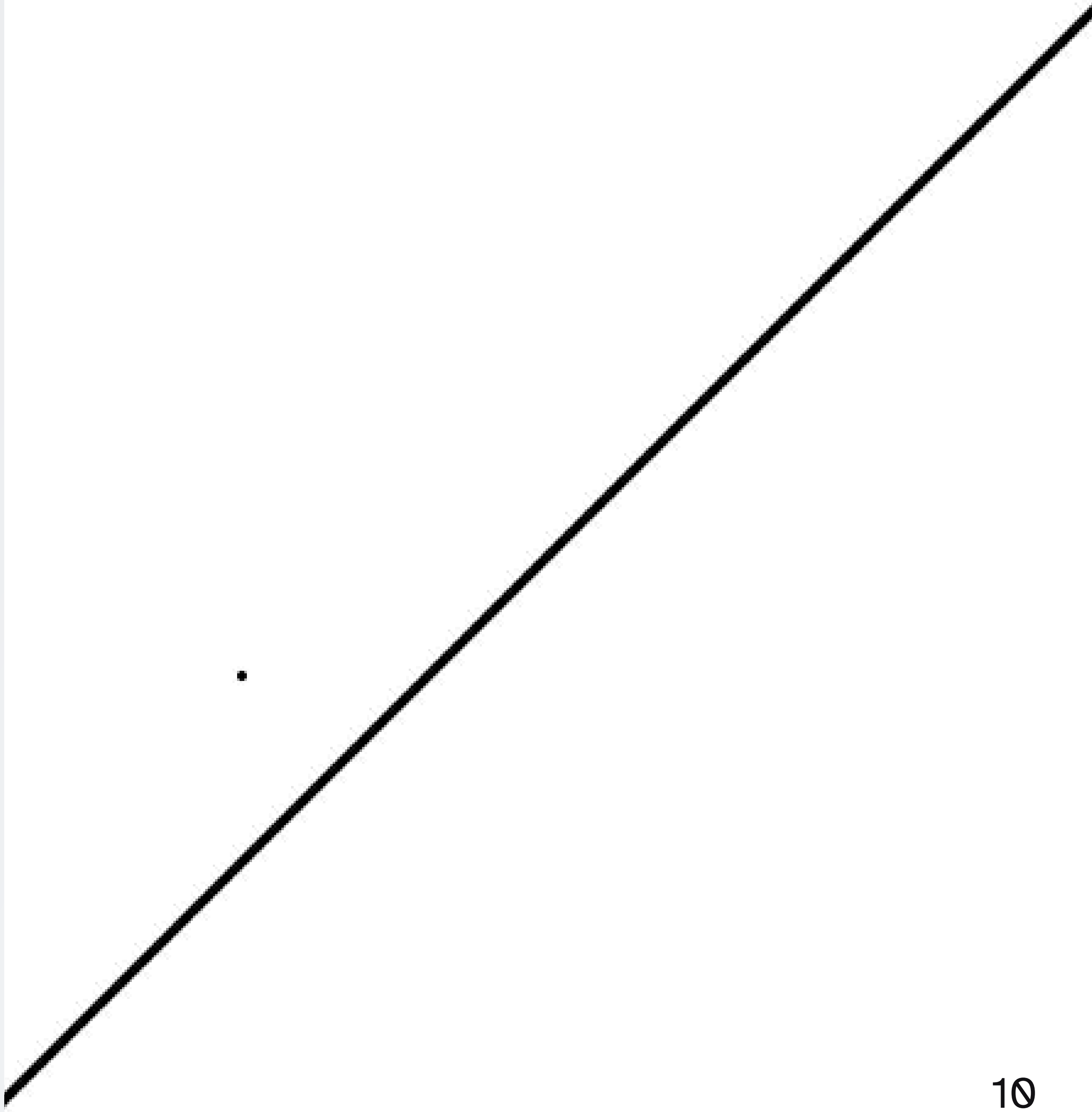
Radius

On right is the same drawing with the pen radius set to 0.008 four times the default setting.

To change the pen radius:



```
PennDraw.setPenRadius(0.008);
```

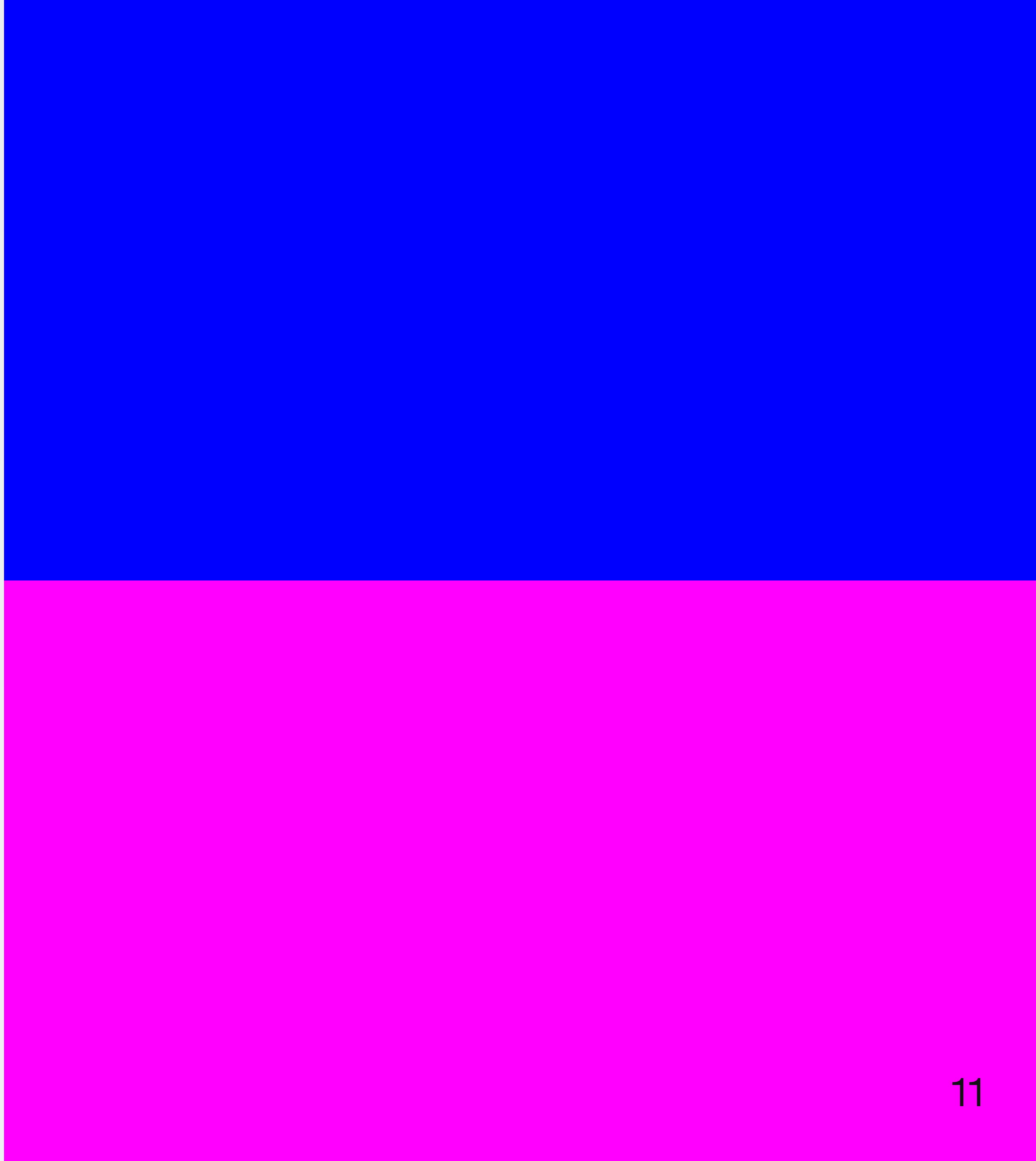


Color

Two ways to set the pen color.

First: referring to some of them by name

```
PennDraw.setPenColor(PennDraw.BLUE);
PennDraw.setPenColor(PennDraw.MAGENTA);
```

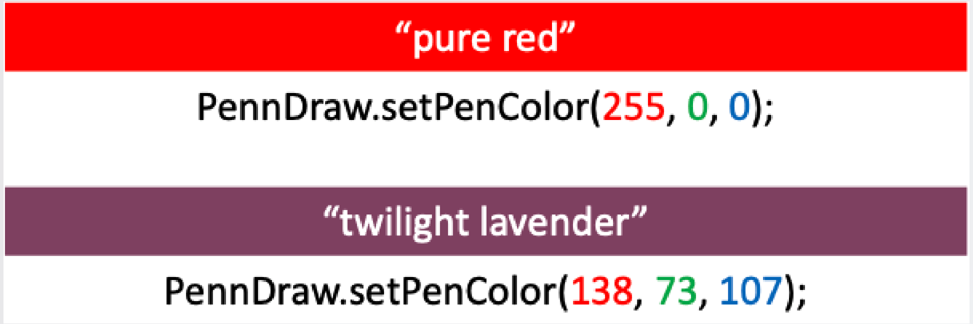


Color

Two ways to set the pen color

Second: specifying the red, green, and blue values of the color as integers from 0-255 each

Allows for fine-grained color control!

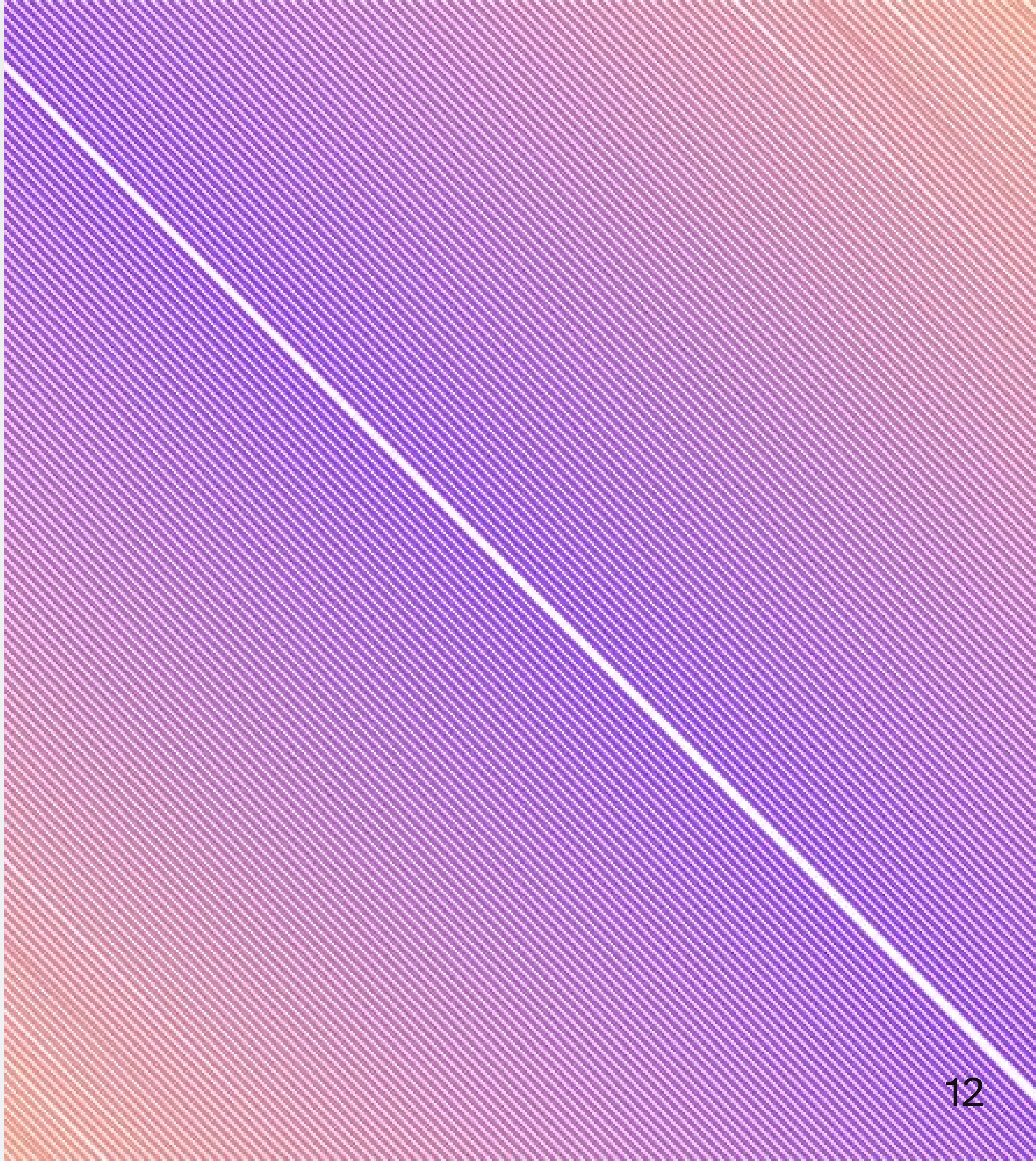


“pure red”

```
PennDraw.setPenColor(255, 0, 0);
```

“twilight lavender”

```
PennDraw.setPenColor(138, 73, 107);
```



Running (and Stopping) PennDraw Programs

- For `HelloWorld.java`, we had a simple loop: *edit, compile, run, repeat*
    - The program was finished running once it ran out of instructions to execute
- For programs that use PennDraw, the program will continue to run so that you can see the drawing you made!
- Before re-compiling, you need to stop the program execution one of two ways:
    i. Close the drawing window
    ii. Press Control-C on your keyboard in the terminal

Demo: Running (and Stopping) a Program

Let's type, compile, and execute the following program:

```java
public class HelloSquare {
    public static void main(String[] args) {
        PennDraw.square(0.5, 0.5, 0.25);
    }
}
```

You should see a black square in the center of the screen!

Demo: Running (and Stopping) a Program

Let's make a change to the program:

```
public class HelloSquare {
    public static void main(String[] args) {
        PennDraw.setPenColor(PennDraw.GREEN);
        PennDraw.square(0.5, 0.5, 0.25);
    }
}
```

Then, we need to **stop the execution** of the program, recompile, and run the program again.

You should see a **green** square in the center of the screen!

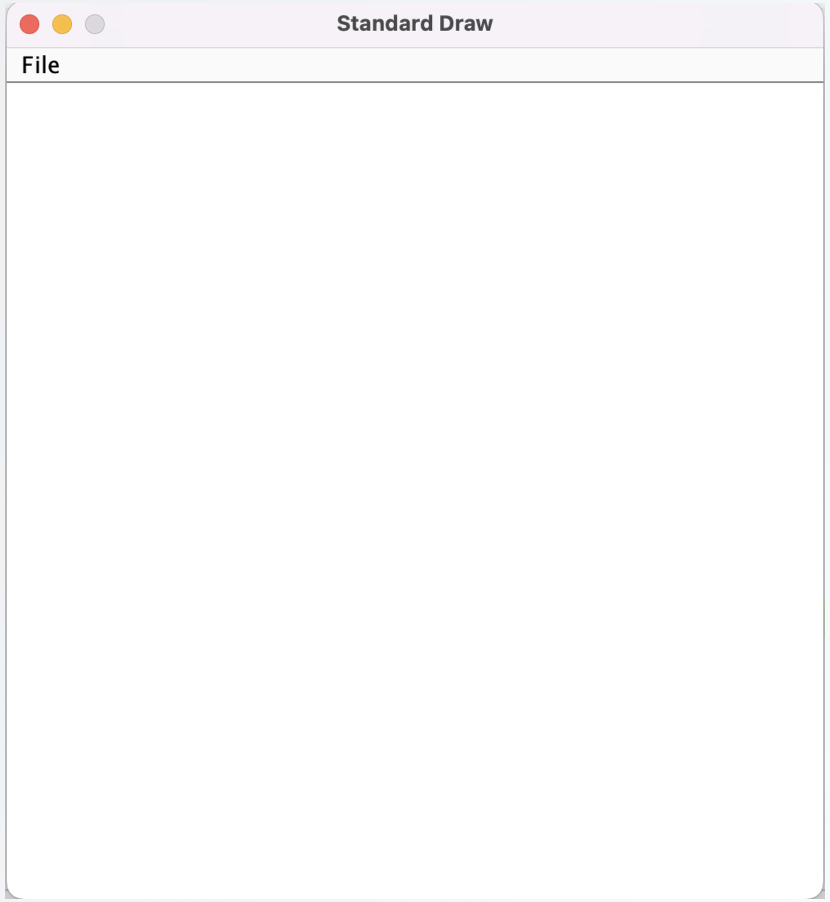
MyHouse.java

Inventory

- `PennDraw.setCanvasSize(width, height)`
  - Sets the canvas to a certain `width` & `height` in pixels
- `PennDraw.clear(color)`
  - Clears the screen and colors the background in the provided `color`.
  - `color` can be provided by name or by passing in three integers
    - e.g. `PennDraw.BLUE` or `(0, 0, 255)`
- `PennDraw.setPenColor(color)`
  - Sets the `color` that future shapes will be drawn in
  - Ditto bullet two from `PennDraw.clear()`

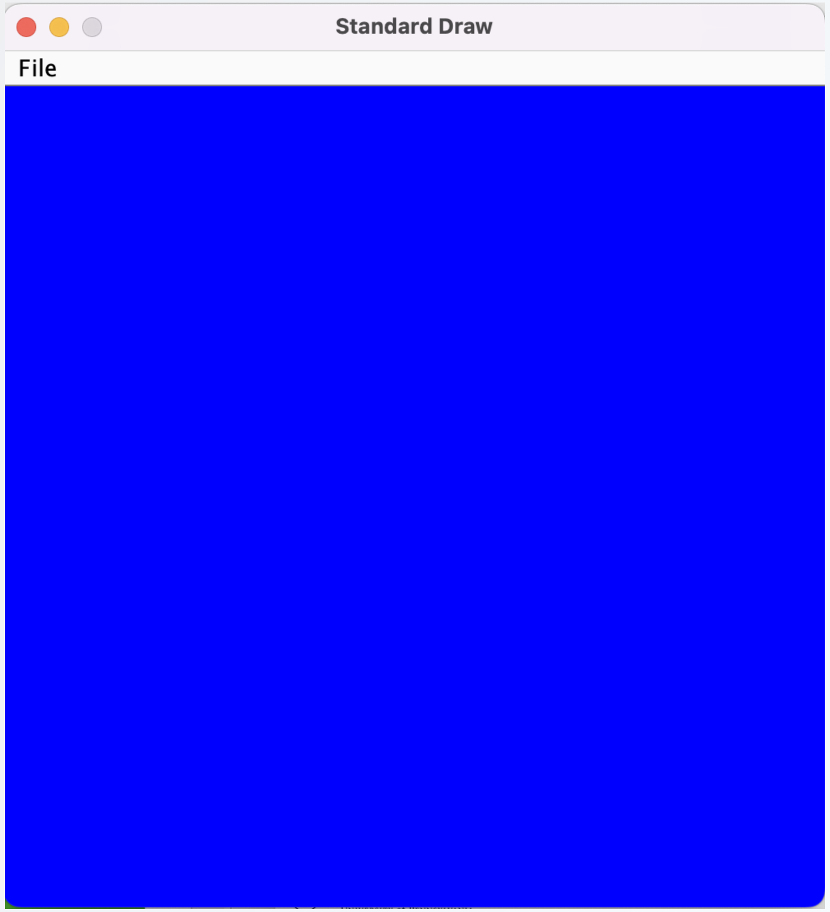
PENNDRAW

```java
public class MyHouse {
    public static void main(String[] args) {

        // Make our window 500 pixels wide and 500 pixels tall.
        PennDraw.setCanvasSize(500, 500);

    }
}
```



PENNDRAW

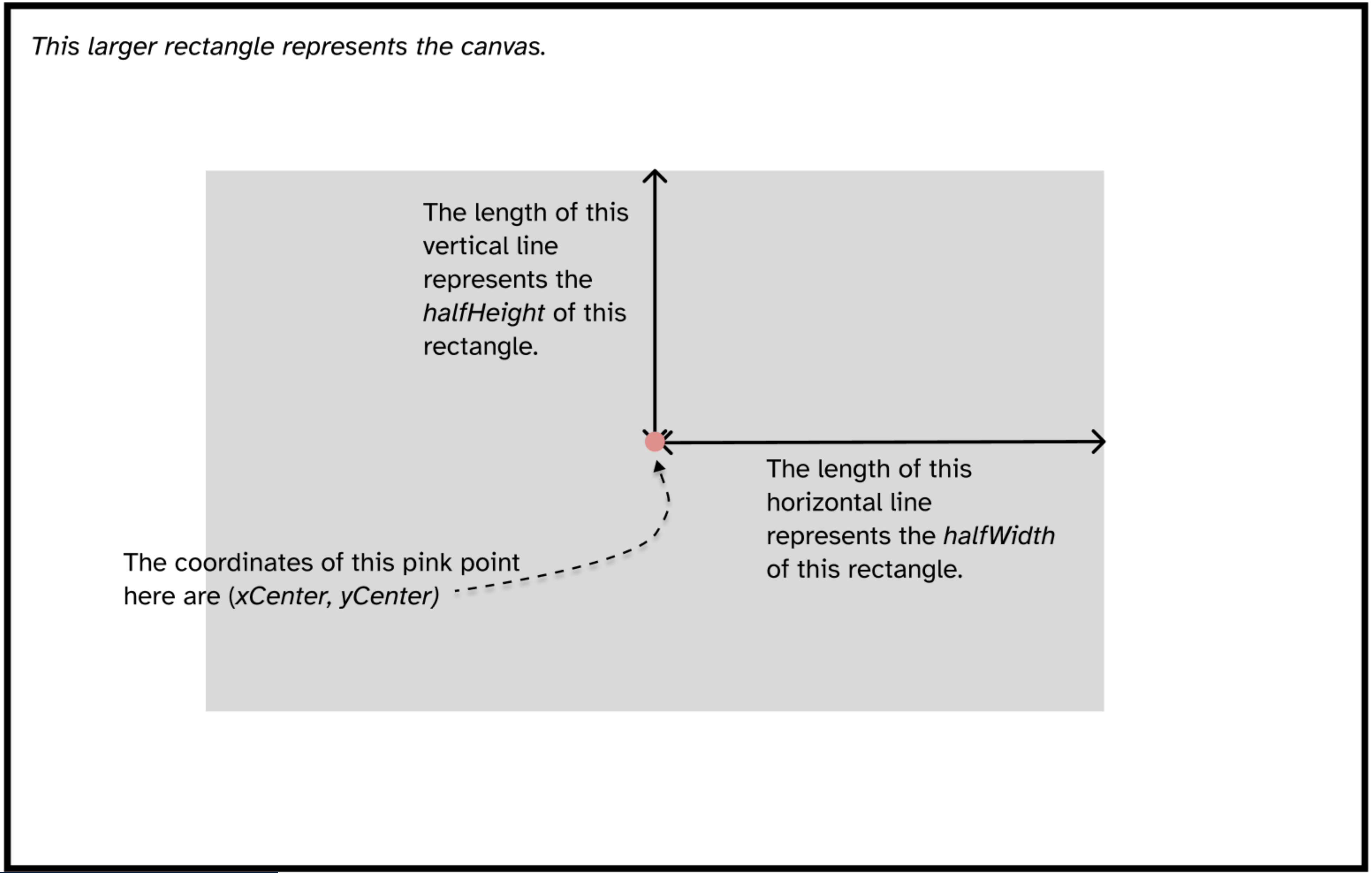
```
public class MyHouse {
    public static void main(String[] args) {
        // Make our window 500 pixels wide and 500 pixels tall.
        PennDraw.setCanvasSize(500, 500);
        // Draw a blue background for our sky
        PennDraw.clear(PennDraw.BLUE);

    }
}
```



`PennDraw.filledRectangle(...)`

This function takes four arguments:

- `xCenter`: the x coordinate of the center of the rectangle
- `yCenter`: the y coordinate of the center of the rectangle
- `halfWidth`: the horizontal distance between the side of the rectangle and its center
- `halfHeight`: the vertical distance between the top of the rectangle and its center



This larger rectangle represents the canvas.

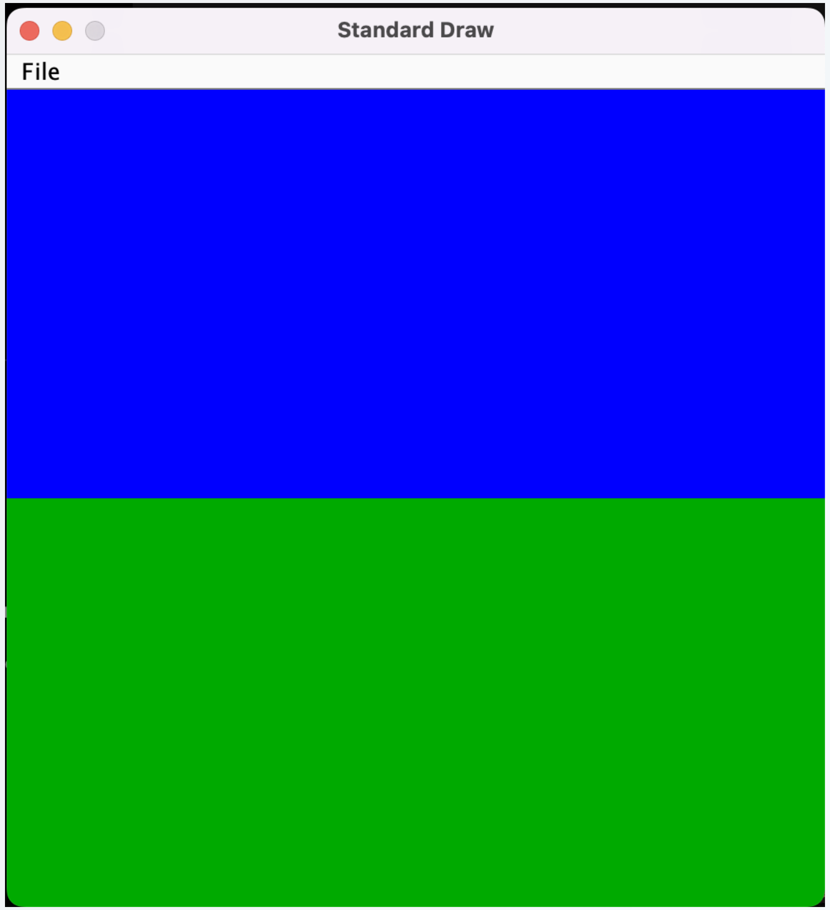
The length of this vertical line represents the *halfHeight* of this rectangle.

The length of this horizontal line represents the *halfWidth* of this rectangle.

The coordinates of this pink point here are (*xCenter, yCenter)*

PENNDRAW

```java
public class MyHouse {
    public static void main(String[] args) {

        // Make our window 500 pixels wide and 500 pixels tall.
        PennDraw.setCanvasSize(500, 500);
        // Draw a blue background for our sky
        PennDraw.clear(PennDraw.BLUE);
        // Draw a green rectangle as the ground
        PennDraw.setPenColor(0, 170, 0);
        PennDraw.filledRectangle(0.5, 0.25, 0.5, 0.25);
    }
}
```



`PennDraw.filledPolygon(...)`

This function takes n pairs of (x, y) coordinates to draw an n-gon with vertices at the specified coordinates.

- For a triangle (3-gon), provide three coordinate pairs (six numbers)
- For a hexagon (6-gon), provide six coordinate pairs (twelve numbers)

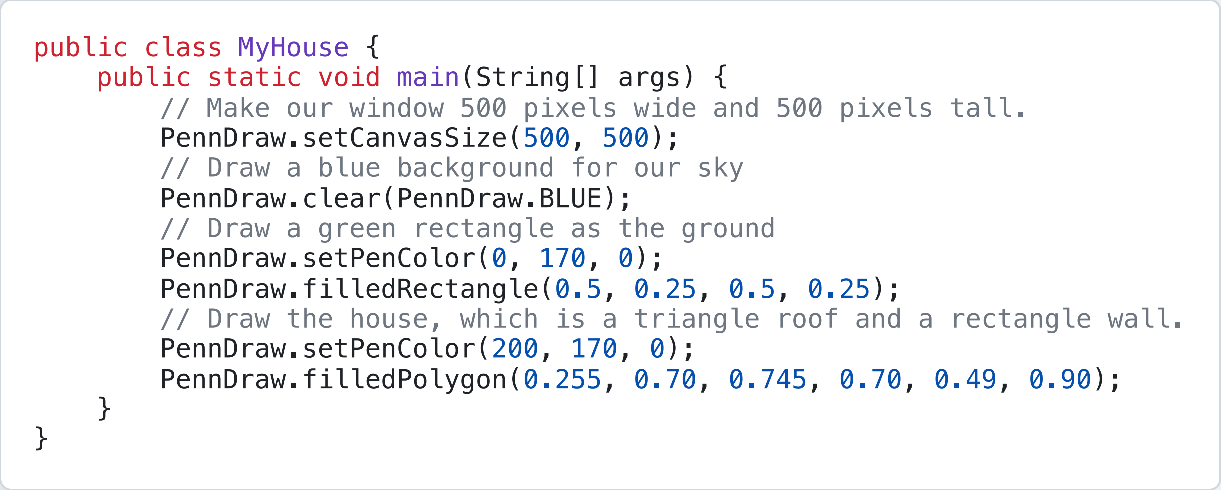
Drawing a Roof

`PennDraw.filledPolygon(0.255, 0.70, 0.745, 0.70, 0.49, 0.90);`

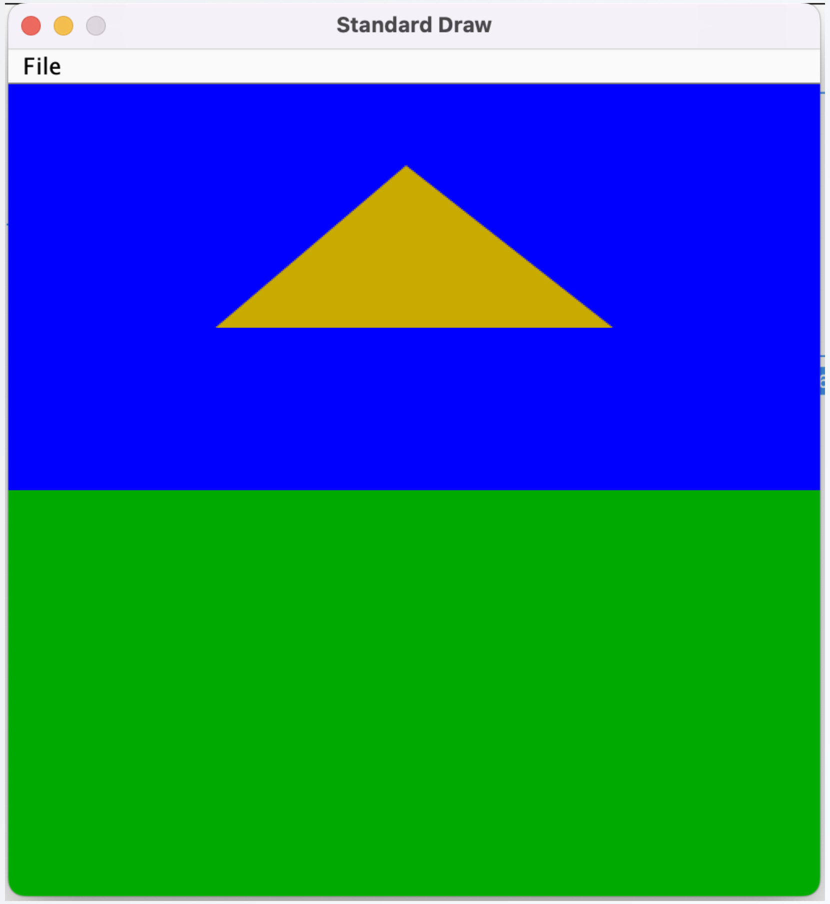
draws a polygon with vertices at coordinates:

- `(0.255, 0.70)`,
- `(0.745, 0.70)`,
- and `(0.49, 0.90)`

PENNDRAW

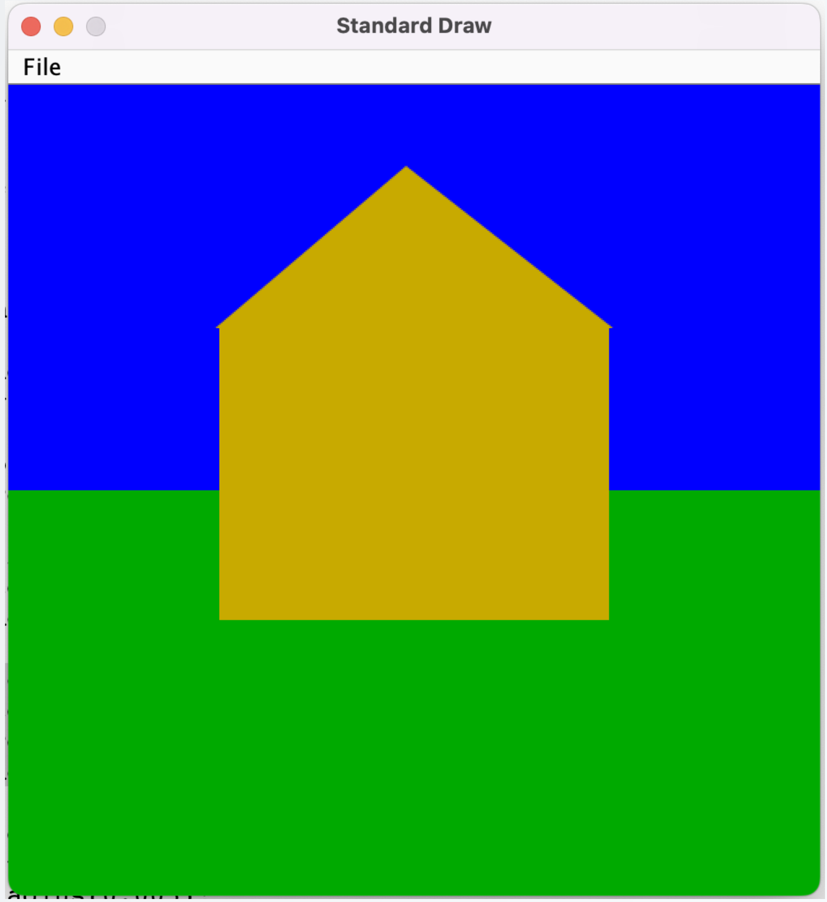


```
public class MyHouse {
    public static void main(String[] args) {
        // Make our window 500 pixels wide and 500 pixels tall.
        PennDraw.setCanvasSize(500, 500);
        // Draw a blue background for our sky
        PennDraw.clear(PennDraw.BLUE);
        // Draw a green rectangle as the ground
        PennDraw.setPenColor(0, 170, 0);
        PennDraw.filledRectangle(0.5, 0.25, 0.5, 0.25);
        // Draw the house, which is a triangle roof and a rectangle wall.
        PennDraw.setPenColor(200, 170, 0);
        PennDraw.filledPolygon(0.255, 0.70, 0.745, 0.70, 0.49, 0.90);
    }
}
```



PENNDRAW

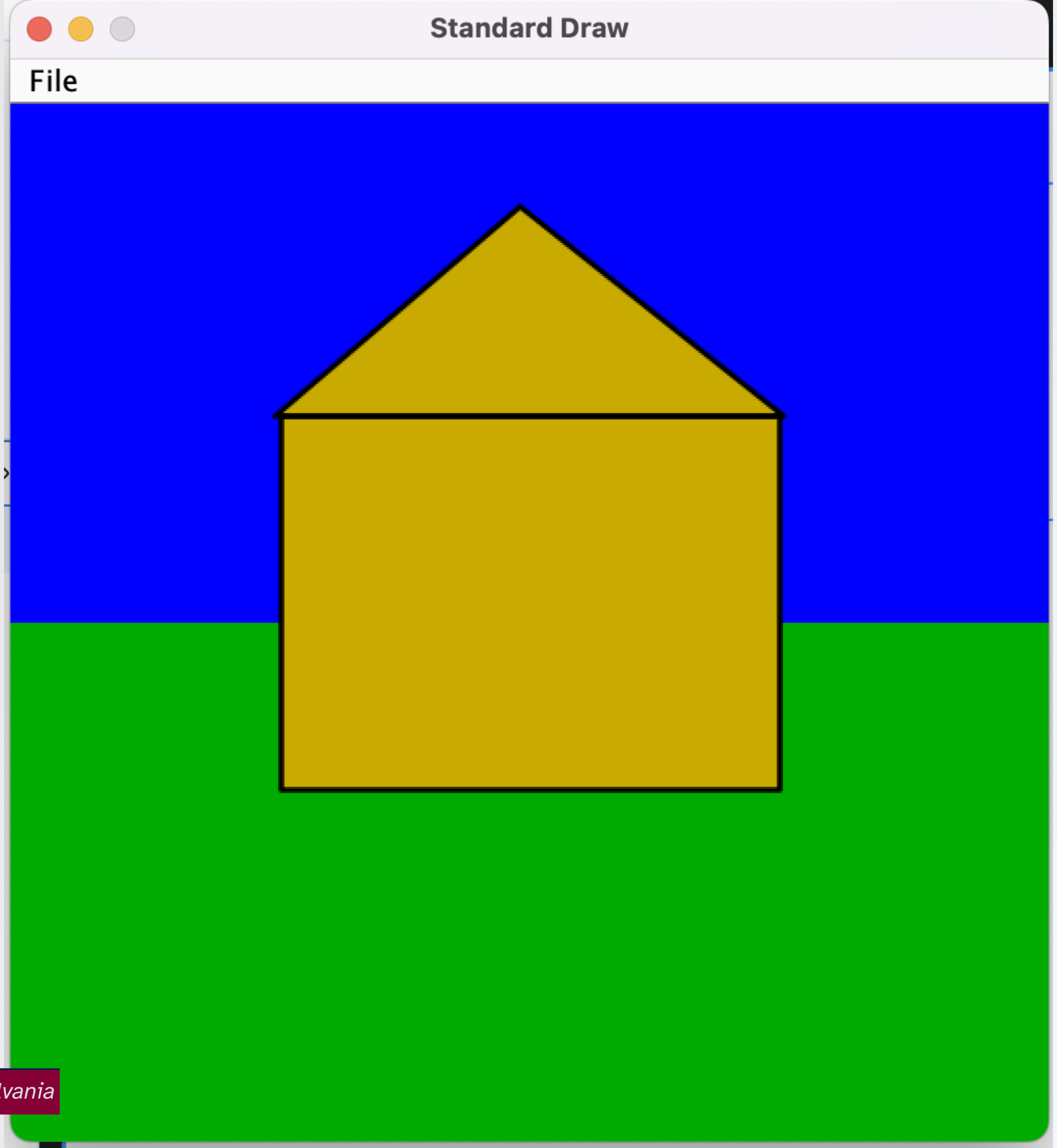
```java
public class MyHouse {
    public static void main(String[] args) {
        // Make our window 500 pixels wide and 500 pixels tall.
        PennDraw.setCanvasSize(500, 500);
        // Draw a blue background for our sky
        PennDraw.clear(PennDraw.BLUE);
        // Draw a green rectangle as the ground
        PennDraw.setPenColor(0, 170, 0);
        PennDraw.filledRectangle(0.5, 0.25, 0.5, 0.25);
        // Draw the house, which is a triangle roof and a rectangle wall.
        PennDraw.setPenColor(200, 170, 0);
        PennDraw.filledPolygon(0.255, 0.70, 0.745, 0.70, 0.49, 0.90);
        PennDraw.filledRectangle(0.5, 0.52, 0.24, 0.18);
    }
}
```



PENNDRAW

```java
public class MyHouse {
    public static void main(String[] args) {

        // Make our window 500 pixels wide and 500 pixels tall.
        PennDraw.setCanvasSize(500, 500);
        // Draw a blue background for our sky
        PennDraw.clear(PennDraw.BLUE);
        // Draw a green rectangle as the ground
        PennDraw.setPenColor(0, 170, 0);
        PennDraw.filledRectangle(0.5, 0.25, 0.5, 0.25);
        // Draw the house, which is a triangle roof and a rectangle wall.
        PennDraw.setPenColor(200, 170, 0);
        PennDraw.filledPolygon(0.255, 0.70, 0.745, 0.70, 0.49, 0.90);
        PennDraw.filledRectangle(0.5, 0.52, 0.24, 0.18);
        // Set the pen color to black and draw outlines around the shapes
        // that make up the house.
        PennDraw.setPenRadius(0.005);
        PennDraw.setPenColor(PennDraw.BLACK);
        PennDraw.polygon(0.255, 0.70, 0.745, 0.70, 0.49, 0.90);
        PennDraw.rectangle(0.5, 0.52, 0.24, 0.18);
    }
}
```

PENNDRAW


Standard Draw
File