Nested For Loops and Intro to Graphics

Lecture 7

CS106A, Summer 2019

Sarai Gould && Laura Cruz-Albrecht

Announcements

- Assignment 1 was due at **10am this morning.**
- Assignment 2 is going out right after lecture!
  - Assignment 2 is due at **10am on Wednesday, July 10th**.

- No lecture or sections Thursday for 4th of July.
  - If your section in cancelled, please try to attend a Wednesday section or Friday's section (11:30am in Skilling Auditorium)
- No LaIR Wednesday, July 3rd due to Holiday.

Announcements

- When emailing instructors, *always email both of us!*
  - We have a joint email: cs106a-sum1819-staff@lists.stanford.edu

- If you don't have a discussion section, please email both instructors ASAP to be placed in a section.

Plan for Today

- Review: Methods
- Nested For Loop with Ascii
- Intro to Graphics Program - Our First Graphics Program!
- Checkerboard
- Infinite Loops

Review: Making Sandwiches

```
/* We will make Java a sandwich.
* Pre: n/a
* Post: We will have a completed sandwich.
*/
private food makeASandwich(food bread, food veg, food protein, food spread){
     food sandwich = bread + veg + protein + spread + bread;

     return sandwich;
}
```

Review: Parameters and Return Statements

```
/* We will do x, y, and z actions and return methodType data
* Pre: What we assume is true beforehand.
* Post: What we promise is true afterwards.
*/
visibility methodType methodName(paramType1 paramName1, paramType2 paramName2){

     commands;

     return dataThatIsMethodType;
}
```

Review: Return Statements

```
public void run(){
     double average = calculateAverage(5, 10);
     println("The average is: " + average);
}

private double calculateAverage(double num1, double num2){
     println("I can also do actions!");
     println("I also HAVE to return a double because that's the method type!");

     double sum = num1 + num2;
     return sum / 2; // This will equal 7.5
}
```

7.5

calculateAverage(5, 10);

ParameterReturnExample [completed]

```
I can also do actions!
I also HAVE to return a double because that's the method type!
The average is 7.5
```

Review: Return Statements

```
public void run(){
     double average = 7.5;
     println("The average is: " + average);
}

private double calculateAverage(double num1, double num2){
     println("I can also do actions!");
     println("I also HAVE to return a double because that's the method type!");

     double sum = num1 + num2;
     return sum / 2; // This will equal 7.5
}
```

ParameterReturnExample [completed]

```
I can also do actions!
I also HAVE to return a double because that's the method type!
The average is 7.5
```

Review: Scope

Scope is the idea that variables only exist inside a certain block of code.

In Java, a variable is **born when it is declared.**

A variable **terminates when it hits the ending bracket** of the code block in which it was declared.

```
public void run(){
     int newVariable = 0; // I am born!
     ...
     ...
     ...
} // I hope I haven't bored you. Goodbye.
```

A Variable Love Story: Scope

The variable only exists from its declaration to the end of its current code block.

```
while (conditionAIsTrue){

     if (conditionBIsTrue){
          ...
          int newVariable = 0; // I am born!
          ...

     } // I'm bored with it all. Goodbye.

}
```

A Variable Love Story: Scope

It doesn't exist before it's declared, and it doesn't exist outside of its current code block!

```
while (conditionAIsTrue){

     if (conditionBIsTrue){
          ...
          int newVariable = 0; // I am born!
          ...

     } // I'm bored with it all. Goodbye.
     println(newVariable);
}
```

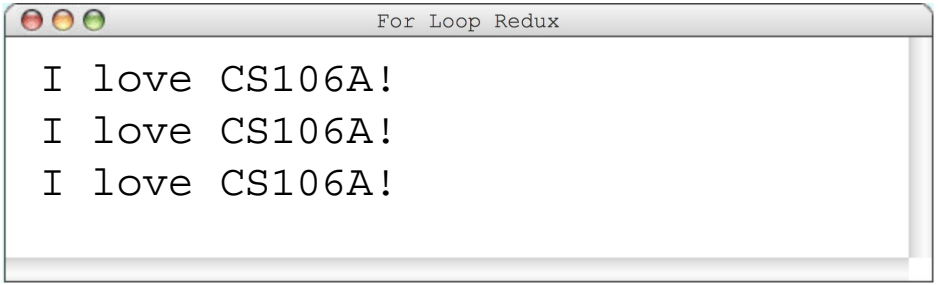
Review: For Loop

This code is run once, just before the for loop starts

Repeats the loop if this condition passes

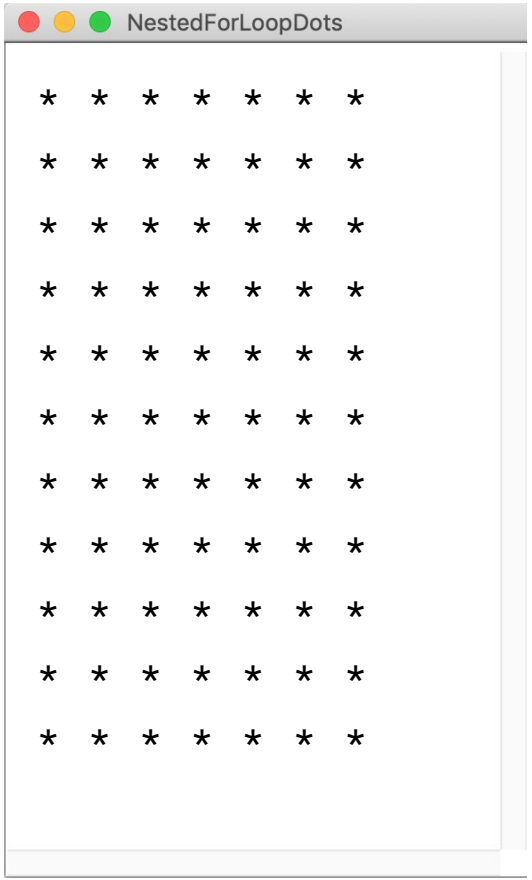
This code is run each time the code gets to the end of the "body"

```
for (int i = 0; i < 3; i++) {
   println("I love CS106A!");
}
```



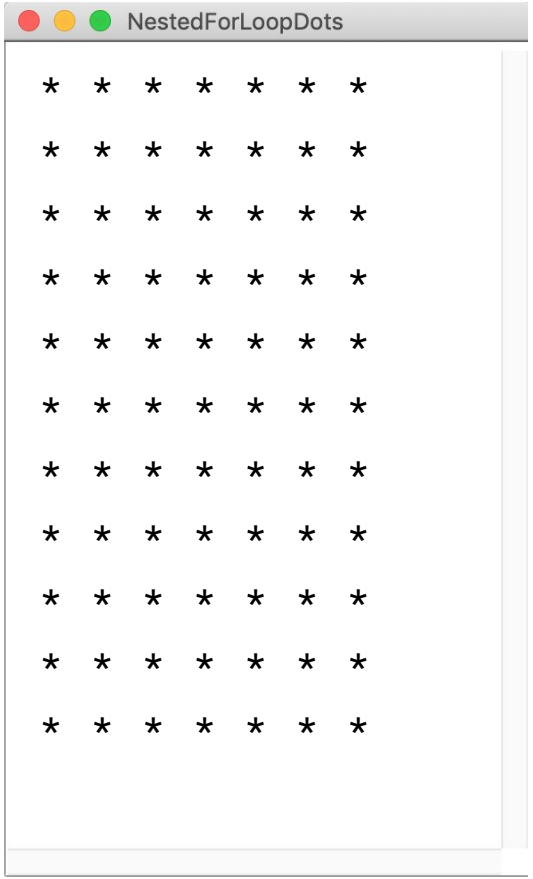
Let’s Make Some Art!

Dot Art



Dot Art

How can we do this?

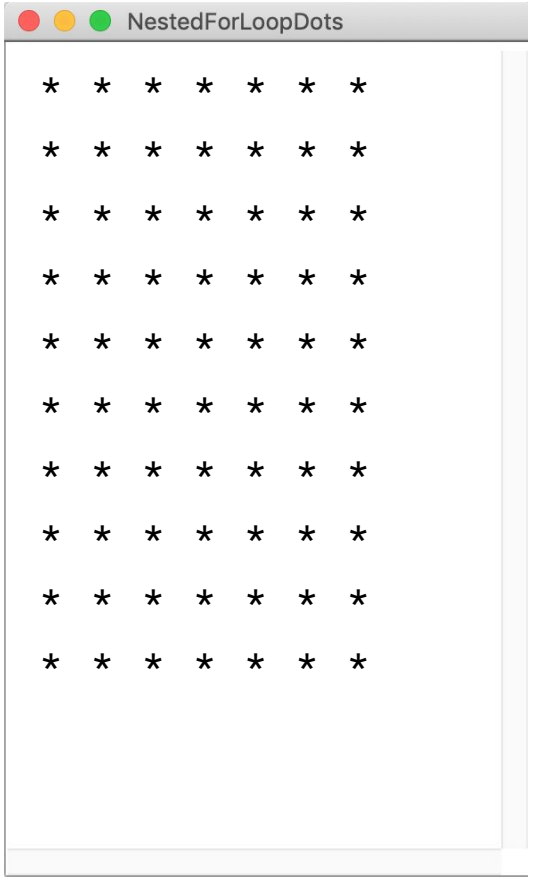


Dot Art

How can we do this?

We use for loops to repeat an action a set number of times.

Let's try it!

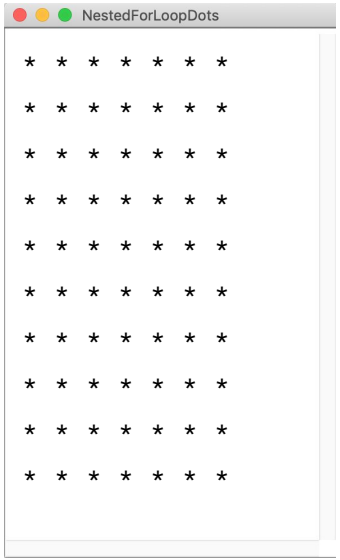


Dot Art

Pseudocode:

Repeat 10 times:

    Draw a line of 7 stars



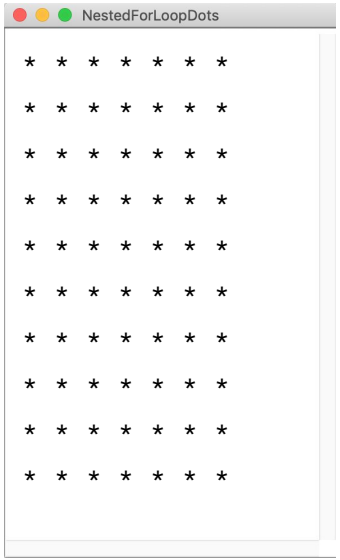

```
public void run(){
     drawDotArt();
}

private void drawDotArt(){


}
```

Dot Art

Pseudocode:

Repeat 10 times:

Draw a line of 7 stars



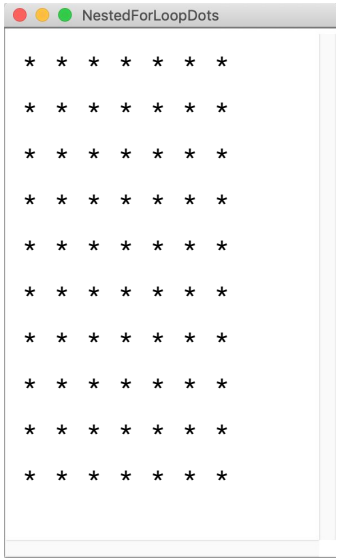

```
public void run(){
     drawDotArt();
}

private void drawDotArt(){

     for (int i = 0; i < 10; i++) {

     }

}
```

Dot Art

Pseudocode:

Repeat 10 times:

Draw a line of 7 stars



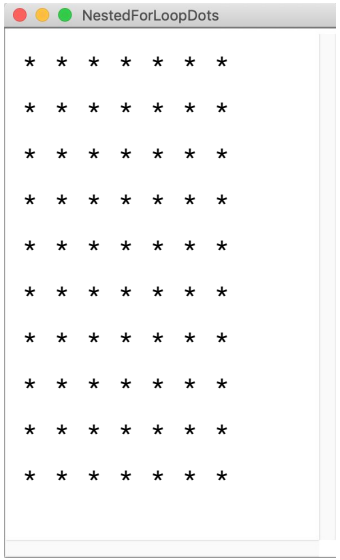

```
public void run(){
     drawDotArt();
}

private void drawDotArt(){

     for (int i = 0; i < 10; i++) {
          println("* * * * * * *");
     }

}
```

Dot Art

Pseudocode:

Repeat 10 times:

Draw a line of 7 stars




```
public void run(){
     drawDotArt();
}

private void drawDotArt(){

     for (int i = 0; i < 10; i++) {
          println("* * * * * * *");
     }

}
```

This line seems a little repetitive.

How can we use code to make this less repetitive?

Dot Art: Nested For Loops

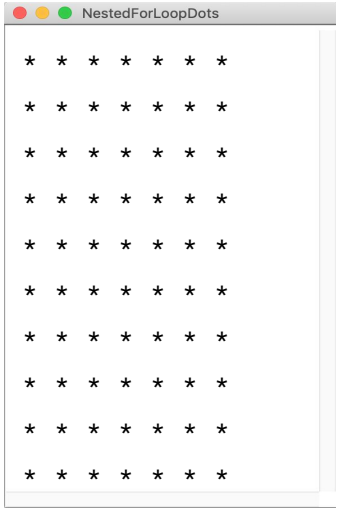
Pseudocode:

Repeat 10 times:

Repeat 7 times:

Draw a star

Make a new line




```
public void run(){
     drawDotArt();
}

private void drawDotArt(){

     for (int i = 0; i < 11; i++) {
          println("* * * * * * *");
     }

}
```

Dot Art: Nested For Loops

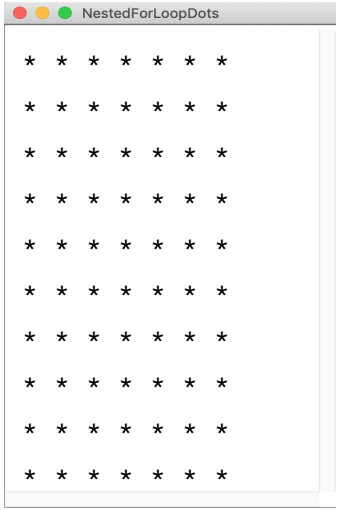
Pseudocode:

Repeat 10 times:

    Repeat 7 times:

        Draw a star

    Make a new line

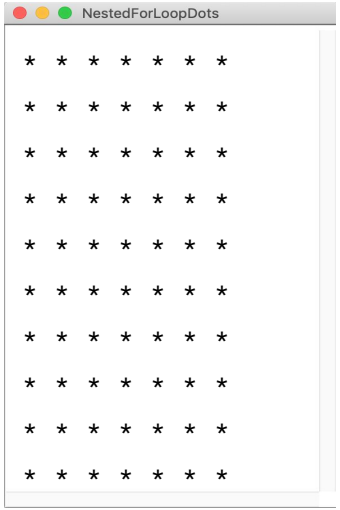



```
public void run(){
     drawDotArt();
}

private void drawDotArt(){

     for (int i = 0; i < 11; i++) {
          for (int j = 0; j < 7; j++) {

          }
     }

}
```

Dot Art: Nested For Loops

Pseudocode:

Repeat 10 times:
    Repeat 7 times:
        Draw a star
    Make a new line

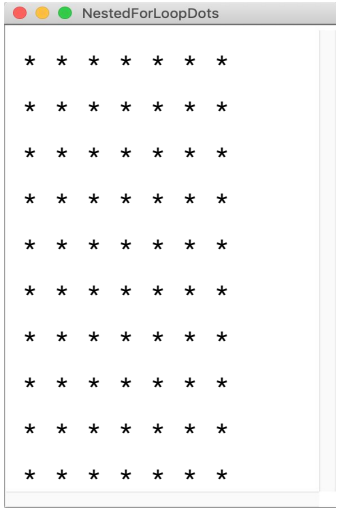



```
public void run(){
     drawDotArt();
}

private void drawDotArt(){

     for (int i = 0; i < 11; i++) {
          for (int j = 0; j < 7; j++) {
               print("*");
          }
     }

}
```

Dot Art: Nested For Loops

Pseudocode:

Repeat 10 times:
    Repeat 7 times:
        Draw a star
    Make a new line




```
public void run(){
     drawDotArt();
}

private void drawDotArt(){

     for (int i = 0; i < 11; i++) {
          for (int j = 0; j < 7; j++) {
               print("*");
          }
          println();
     }

}
```

Let’s Code It!

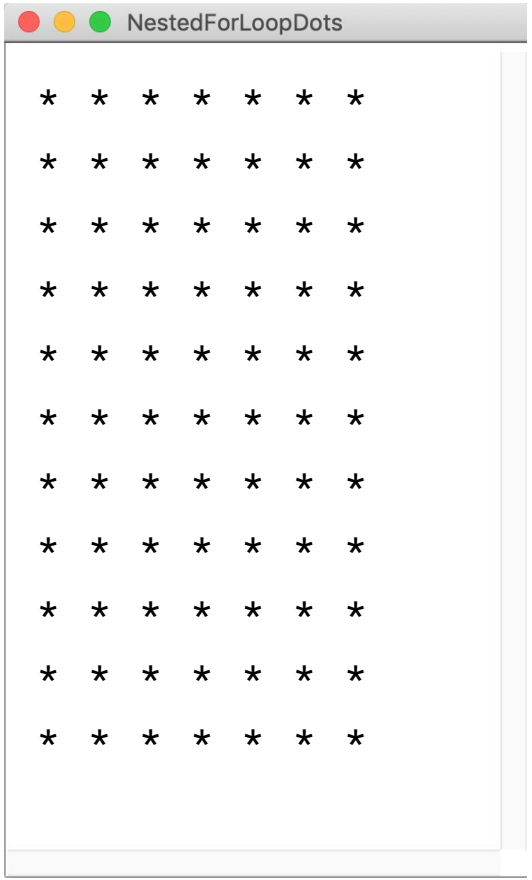
Dot Art: Nested For Loops

We've added parameters so we can take our user input and pass it into our method.

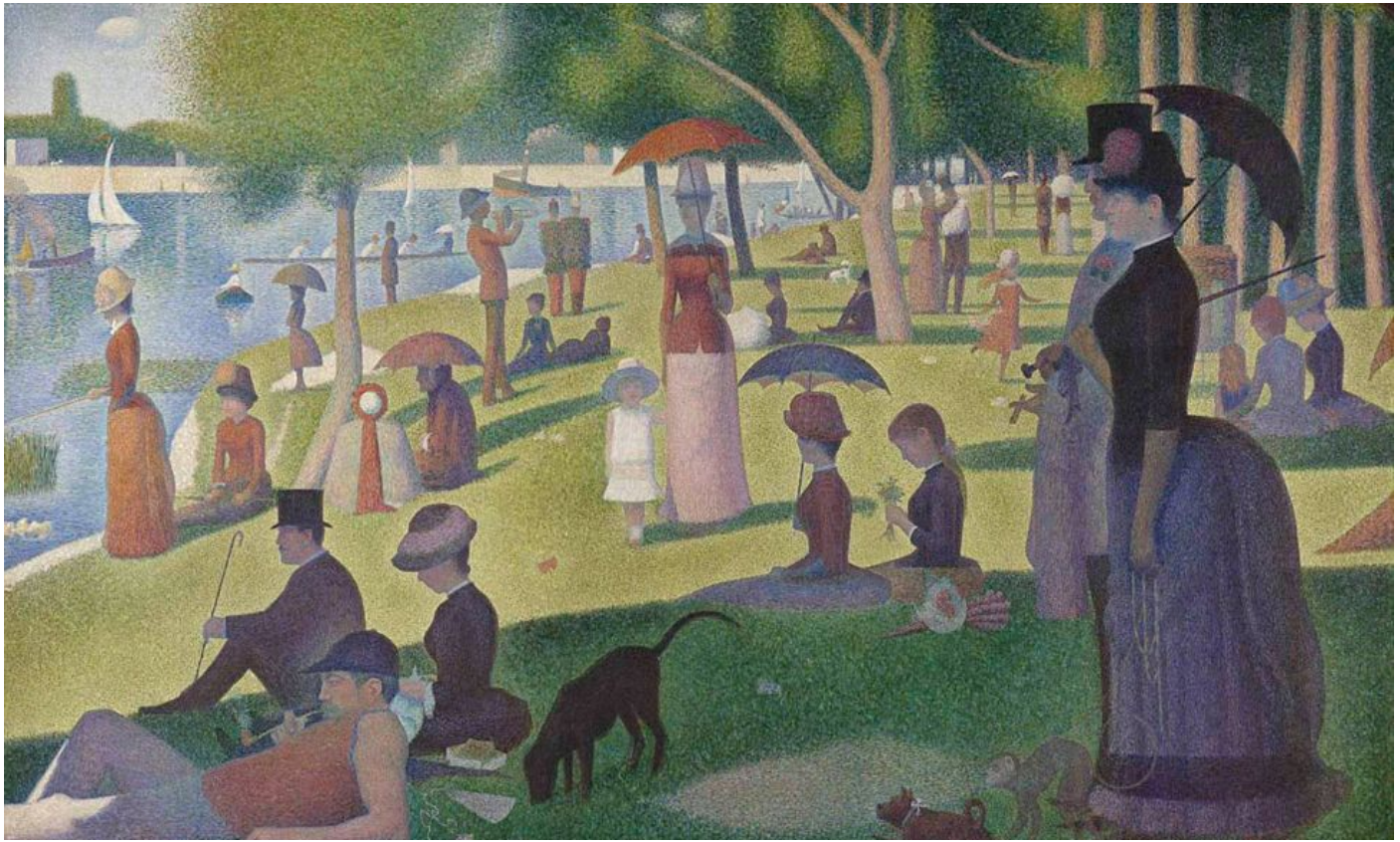
`r` will equal `rows` and `c` will equal `cols`.

```
public void run(){
     int rows = readInt("How many rows?");
     int cols = readInt("How many cols?");
     drawDotArt(rows, cols);
}

private void drawDotArt(int r, int c){

     for (int i = 0; i < r; i++) {
          for (int j = 0; j < c; j++) {
               print("*");
          }
          println();
     }

}
```

Can Dots be Art?



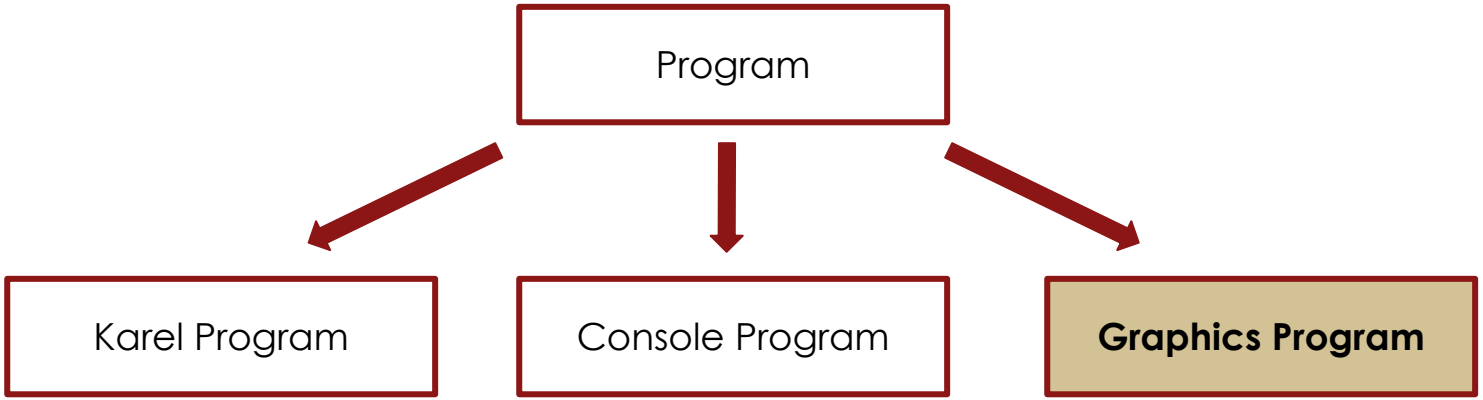
Isn’t This Art???



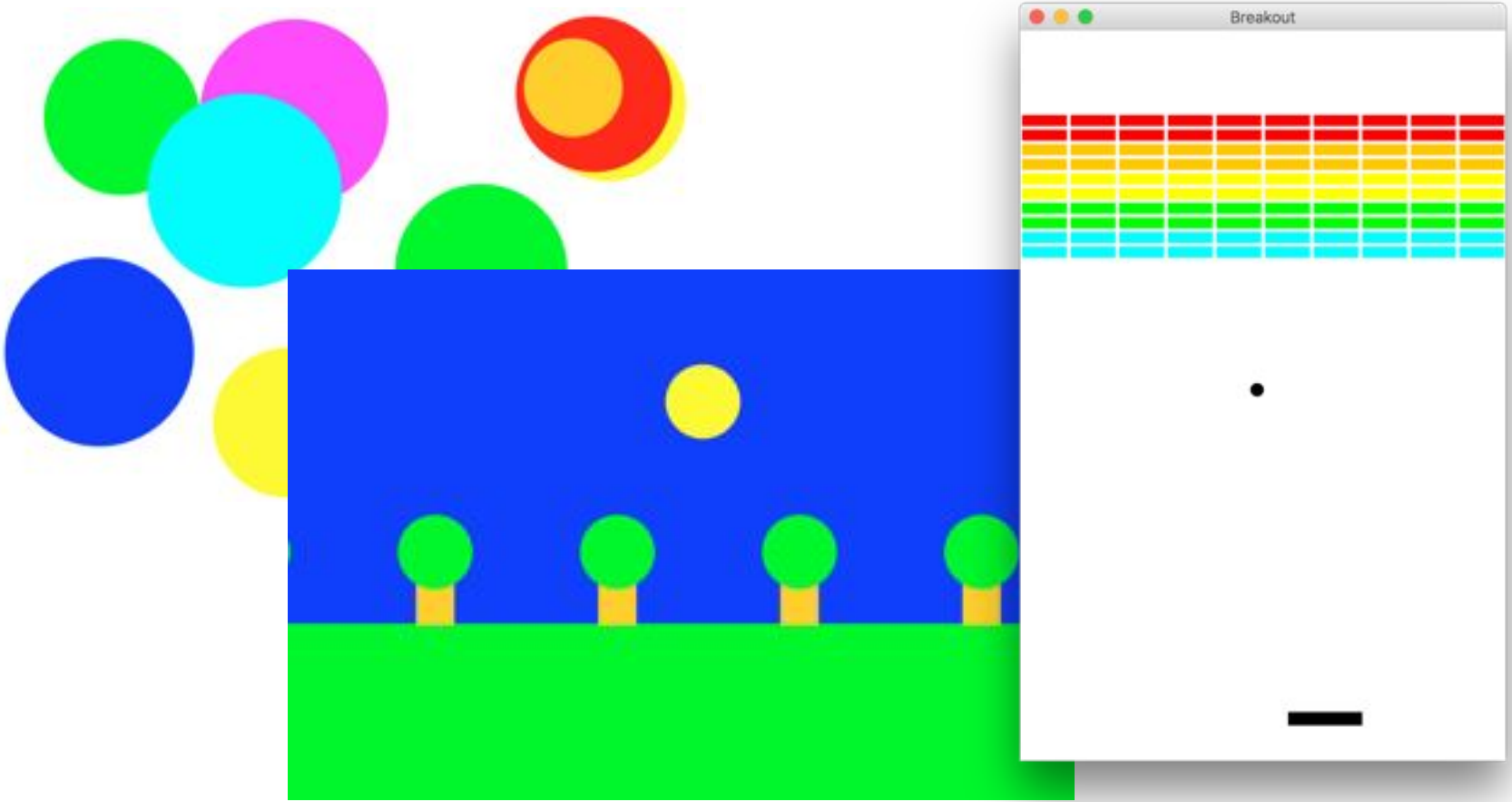
Not really the same, but okay...

Speaking of Art...

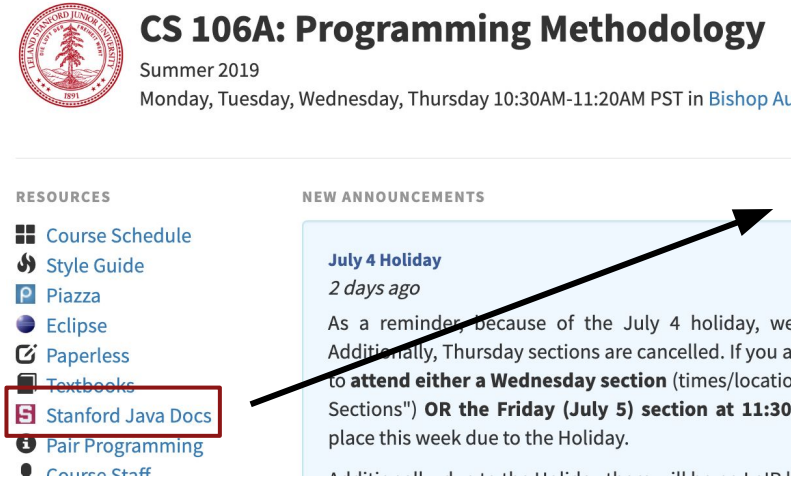
Let's Look at Some Graphics Programs!

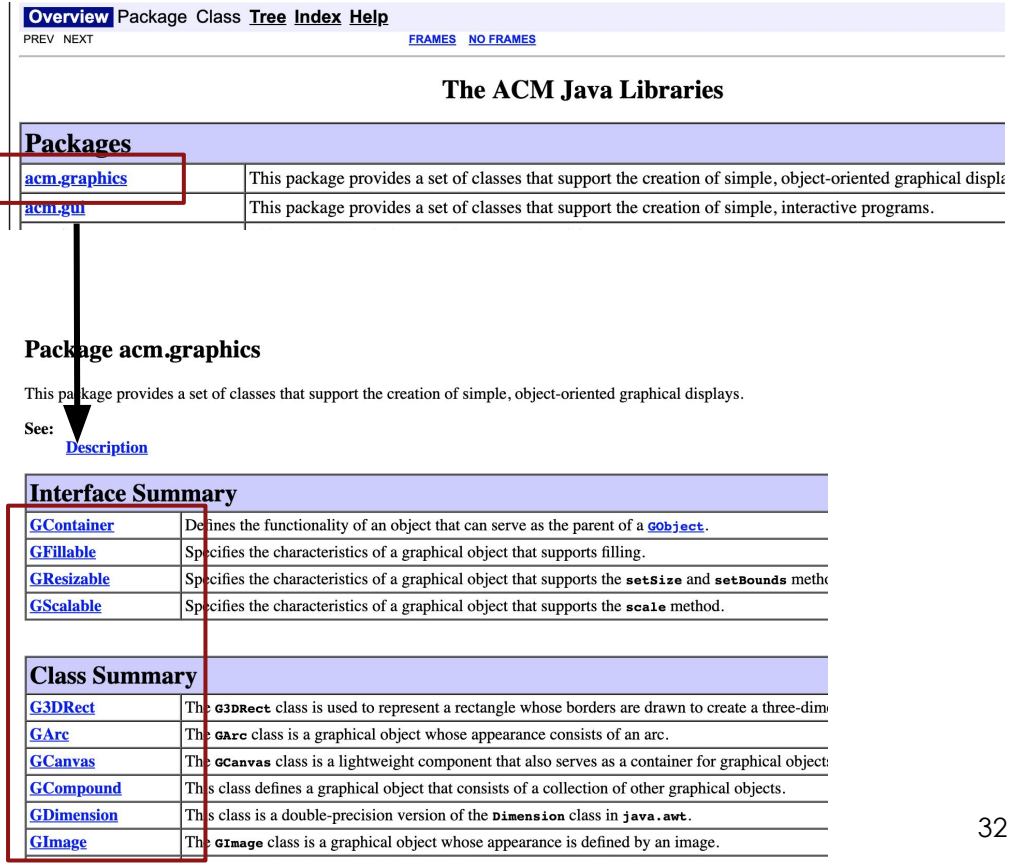


Let's Look at Some Graphics Programs!

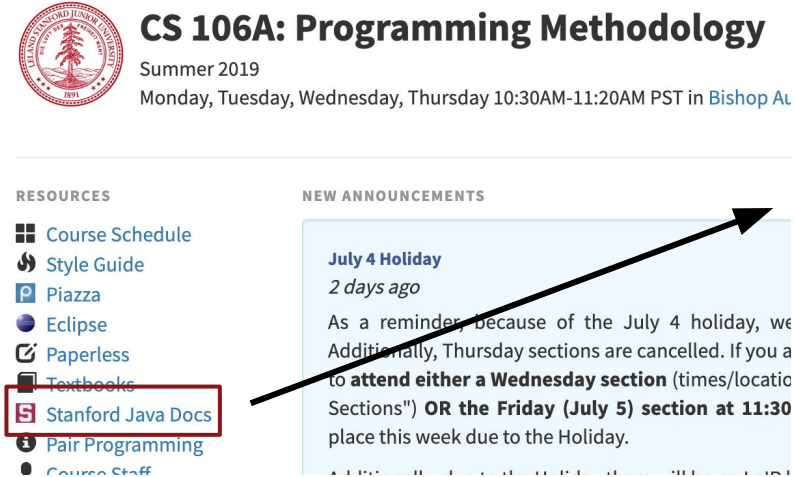


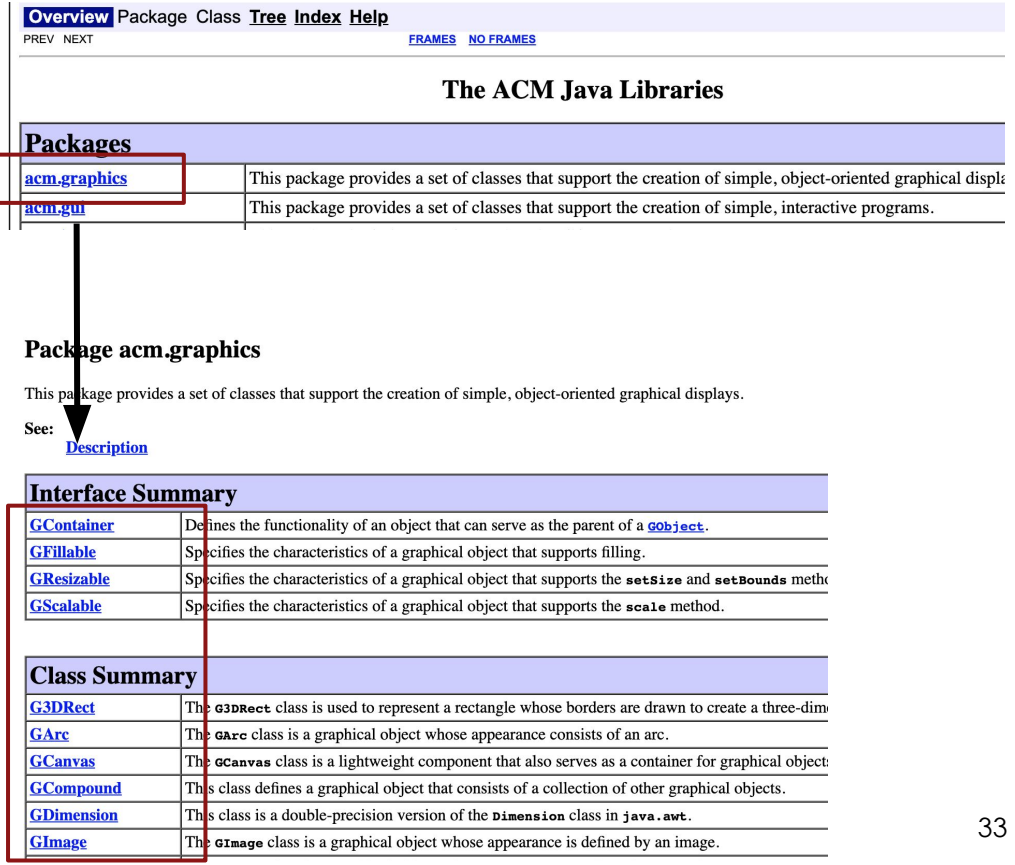
As We're Learning: JavaDocs!





As We're Learning: JavaDocs!



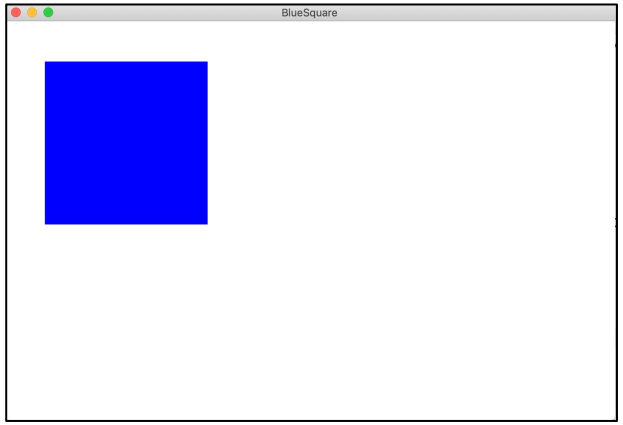


We don't expect you to memorize everything! Use the graphics library documentation to look up methods you can use in graphics programs!

Our First Graphics Object: GRect

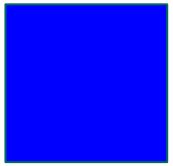
A GRect is a variable type that stores a rectangle.

As an example, the following code displays a blue square.

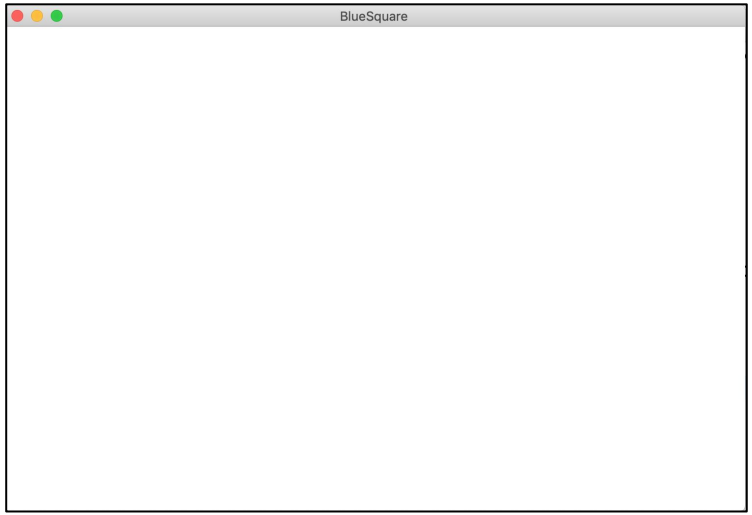



```
public void run(){
     drawBlueSquare();
}

private void drawBlueSquare(){
     GRect rect = new GRect(200, 200);
     rect.setFilled(true);
     rect.setColor(Color.BLUE);
     add(rect, 50, 50);
}
```

rect

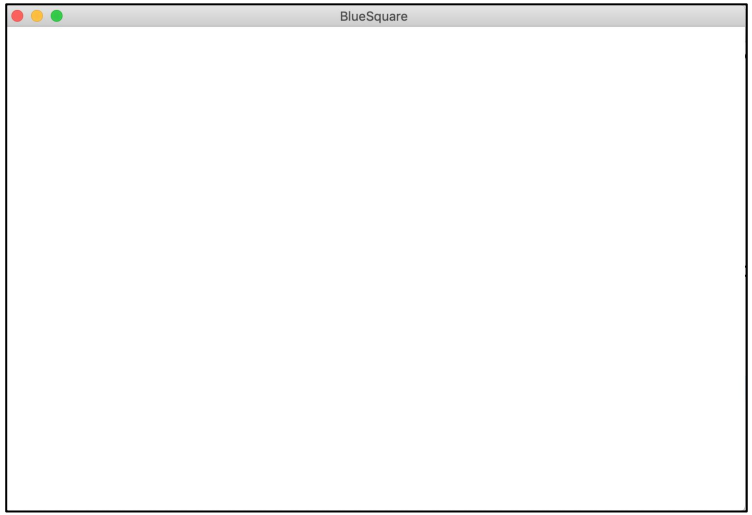


Our First Graphics Object: GRect



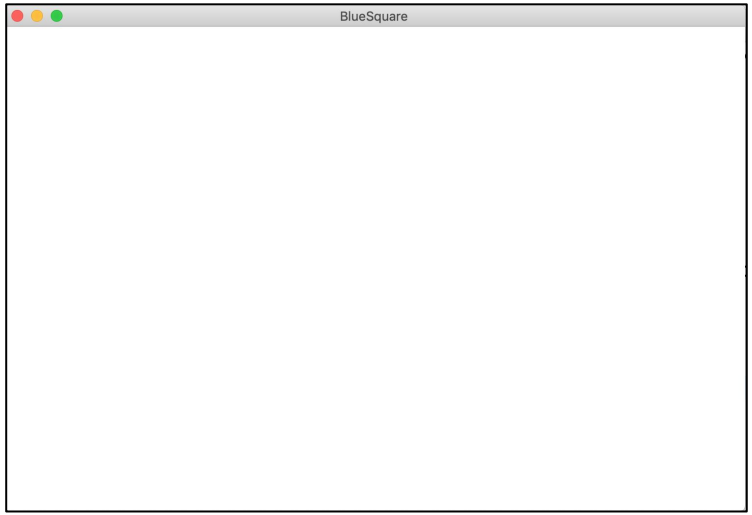

```
public void run(){
     drawBlueSquare();
}

private void drawBlueSquare(){
     GRect rect = new GRect(200, 200);
     rect.setFilled(true);
     rect.setColor(Color.BLUE);
     add(rect, 50, 50);
}
```

Our First Graphics Object: GRect

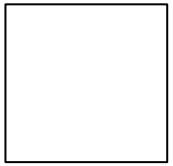



```
public void run(){
     drawBlueSquare();
}

private void drawBlueSquare(){
     GRect rect = new GRect(200, 200);
     rect.setFilled(true);
     rect.setColor(Color.BLUE);
     add(rect, 50, 50);
}
```

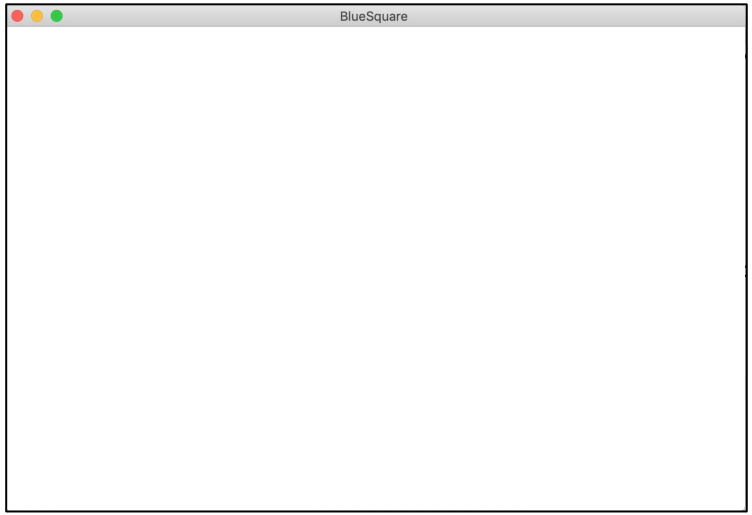
Our First Graphics Object: GRect




```
public void run(){
     drawBlueSquare();
}

private void drawBlueSquare(){
     GRect rect = new GRect(200, 200);
     rect.setFilled(true);
     rect.setColor(Color.BLUE);
     add(rect, 50, 50);
}
```

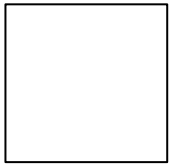


Our First Graphics Object: GRect

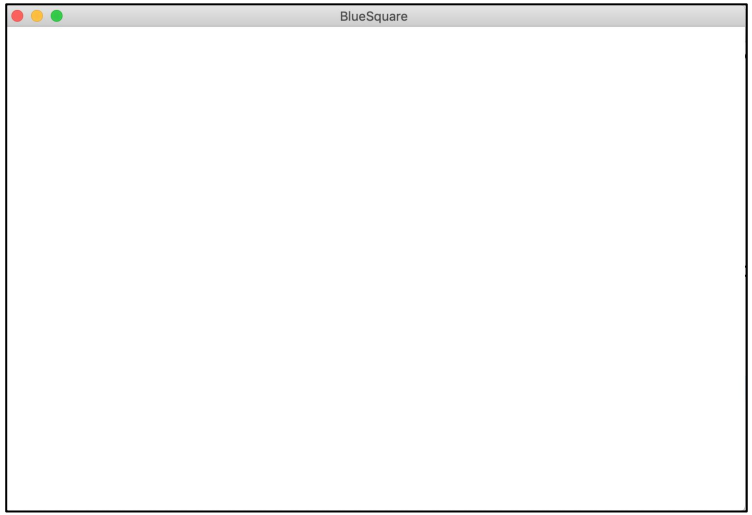



```
public void run(){
     drawBlueSquare();
}

private void drawBlueSquare(){
     GRect rect = new GRect(200, 200);
     rect.setFilled(true);
     rect.setColor(Color.BLUE);
     add(rect, 50, 50);
}
```

rect



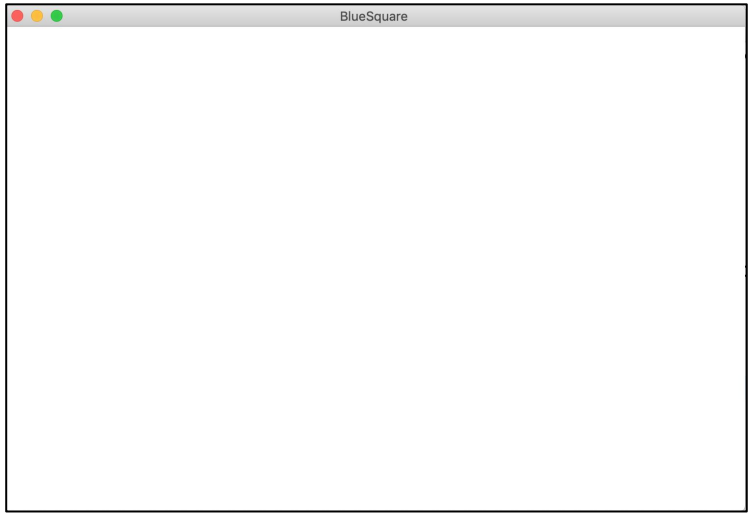
Our First Graphics Object: GRect




```
public void run(){
     drawBlueSquare();
}

private void drawBlueSquare(){
     GRect rect = new GRect(200, 200);
     rect.setFilled(true);
     rect.setColor(Color.BLUE);
     add(rect, 50, 50);
}
```

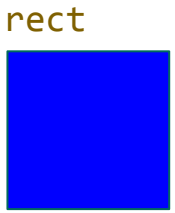
rect

Our First Graphics Object: GRect

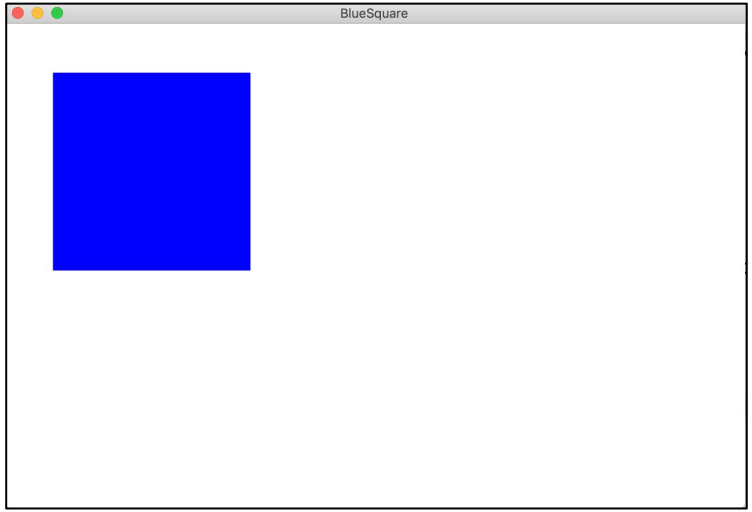



```
public void run(){
     drawBlueSquare();
}

private void drawBlueSquare(){
     GRect rect = new GRect(200, 200);
     rect.setFilled(true);
     rect.setColor(Color.BLUE);
     add(rect, 50, 50);
}
```

Note: The `GRect` is *created,* but hasn't been *added to the screen.*

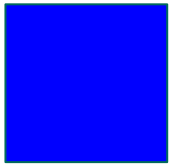


Our First Graphics Object: GRect



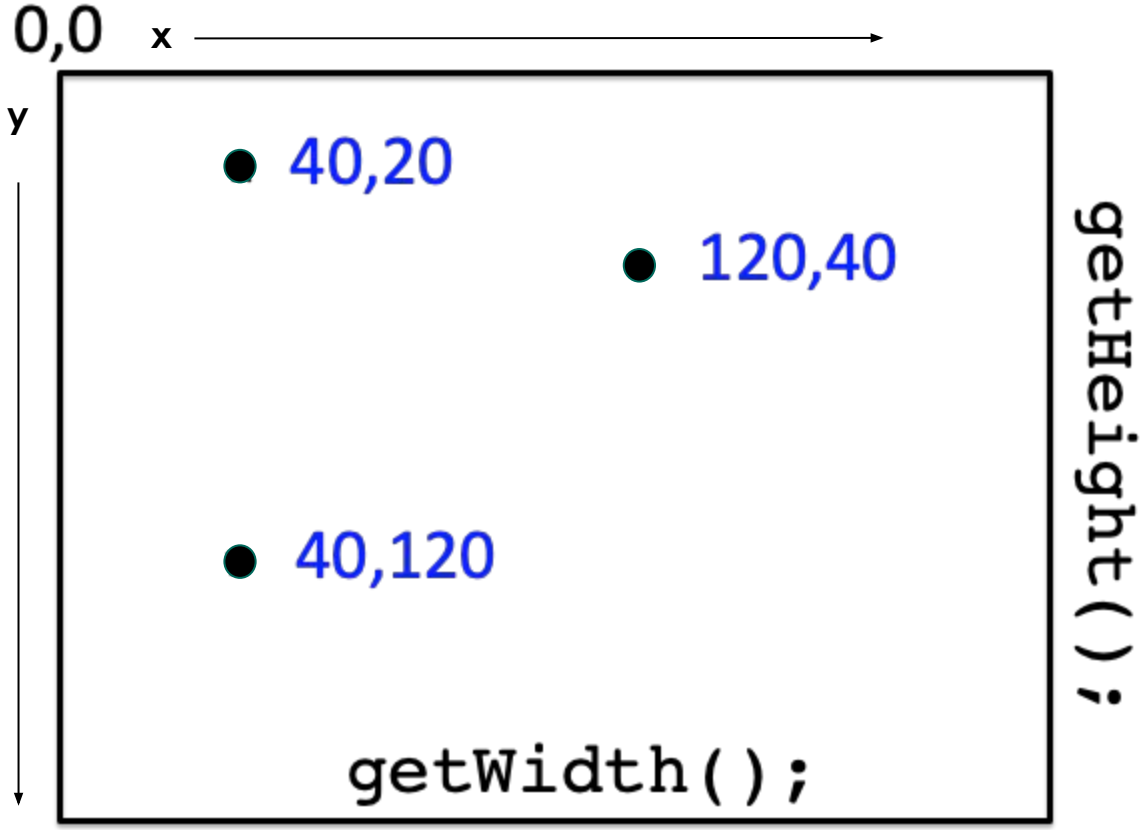

```
public void run(){
     drawBlueSquare();
}

private void drawBlueSquare(){
     GRect rect = new GRect(200, 200);
     rect.setFilled(true);
     rect.setColor(Color.BLUE);
     add(rect, 50, 50);
}
```

rect



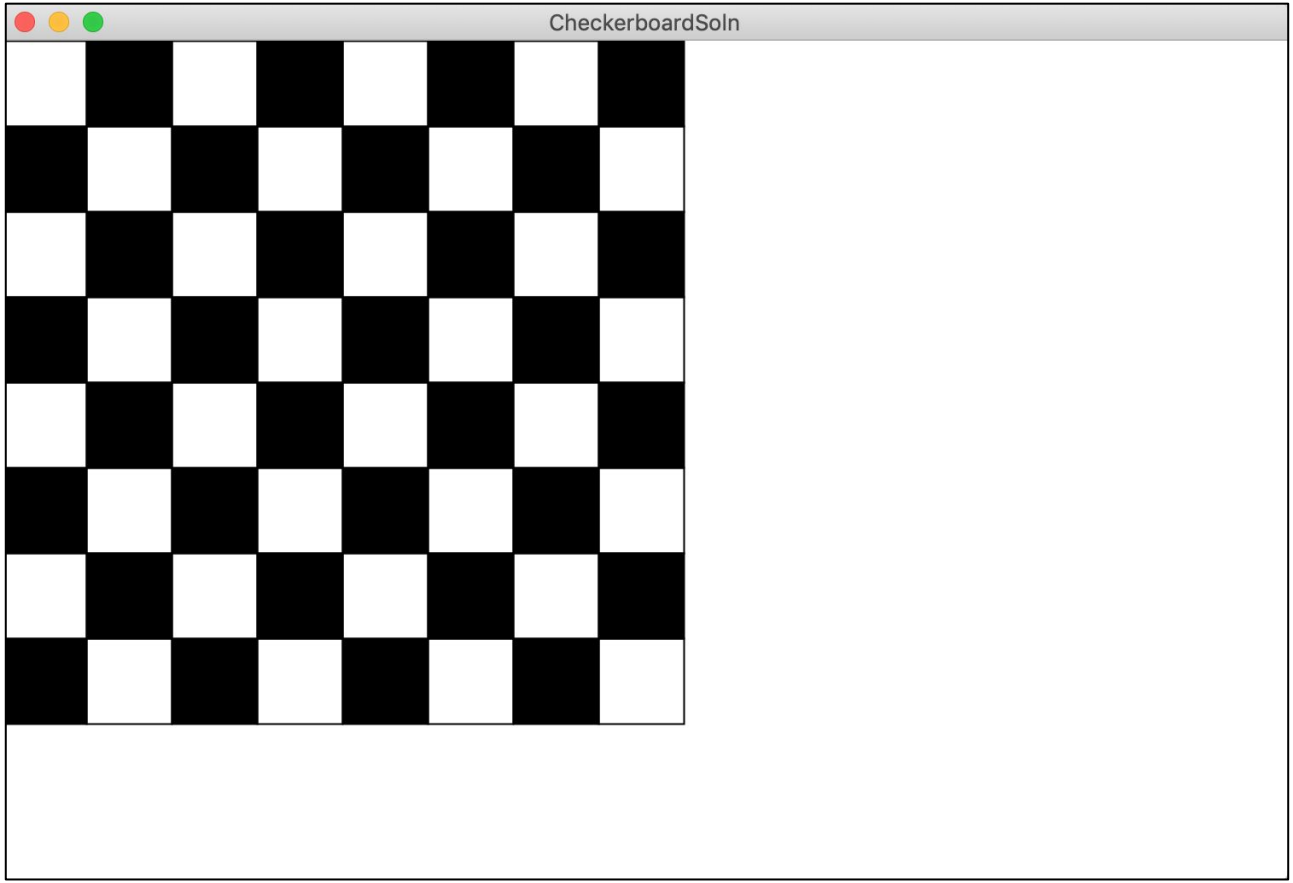
Java Coordinate System

Note:
The y coordinate gets *bigger* as we go *down*. This is opposite of how it acts in most math classes!

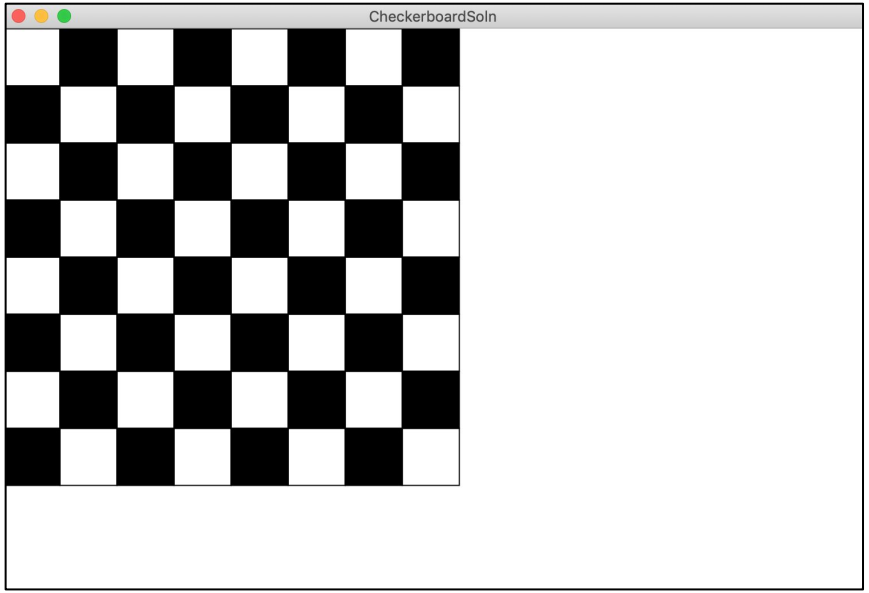


Our First Graphics Challenge

Checkerboard

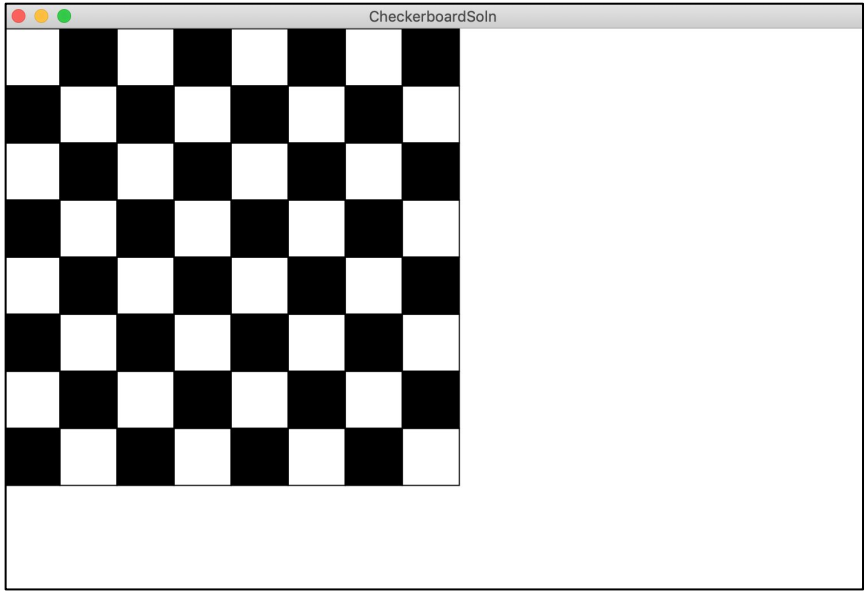


Our First Graphics Challenge



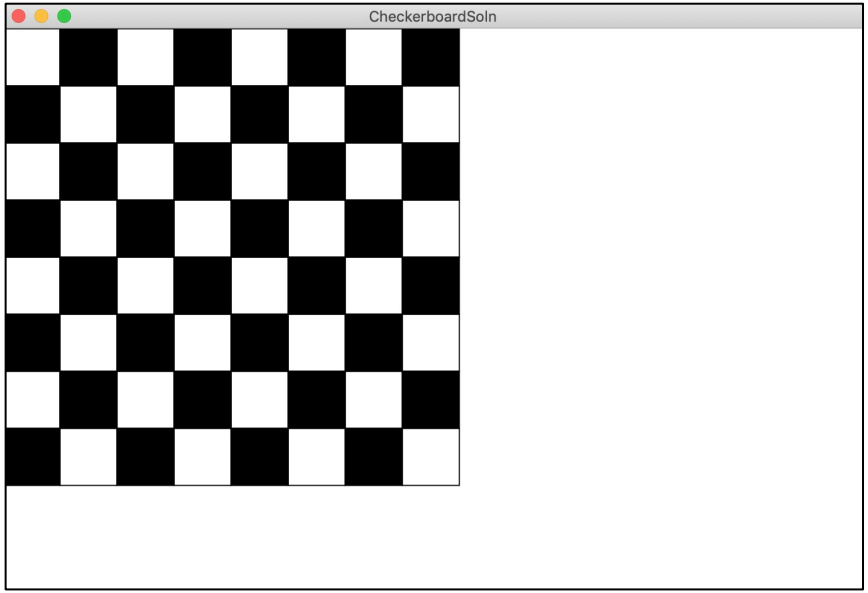
What's the Pseudocode?

Our First Graphics Challenge



What's the Pseudocode for drawing the first row of boxes?

Our First Graphics Challenge



What's the Pseudocode for drawing the first row of boxes?

repeat for 8 columns:

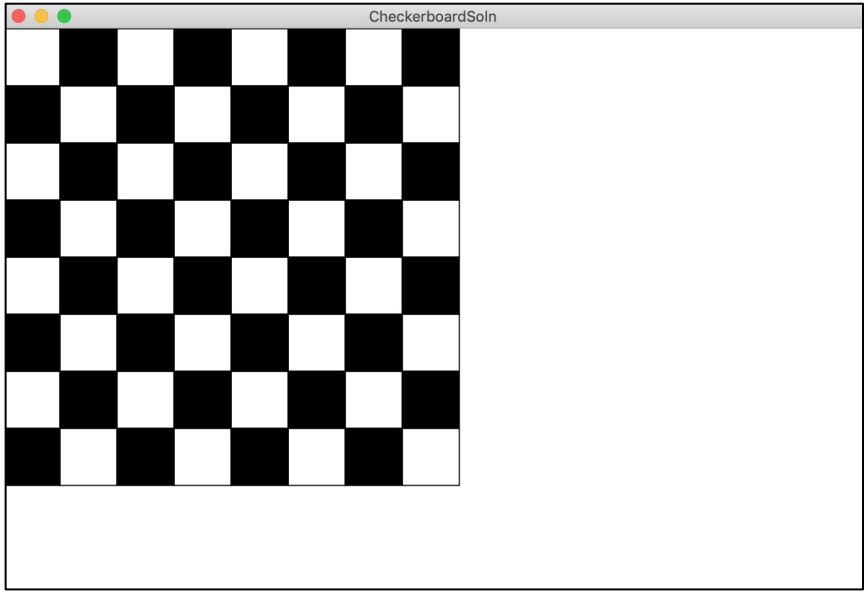
draw a box

*(x location is col# * box size)*

(y location is 0)

add the box to the screen

Our First Graphics Challenge



How can we change the Pseudocode to draw all of the boxes?

repeat for 8 columns:

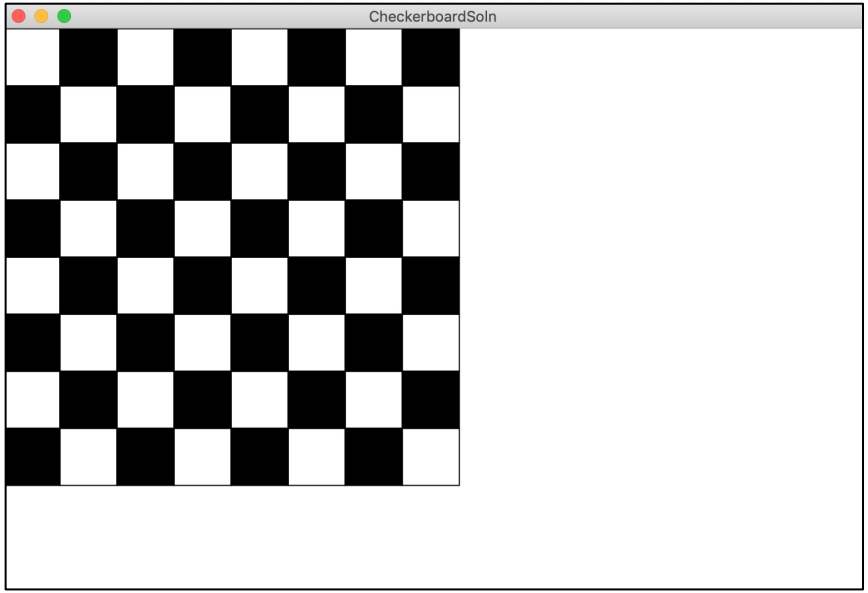
draw a box

*(x location is col# * box size)*

(y location is 0)

add the box to the screen

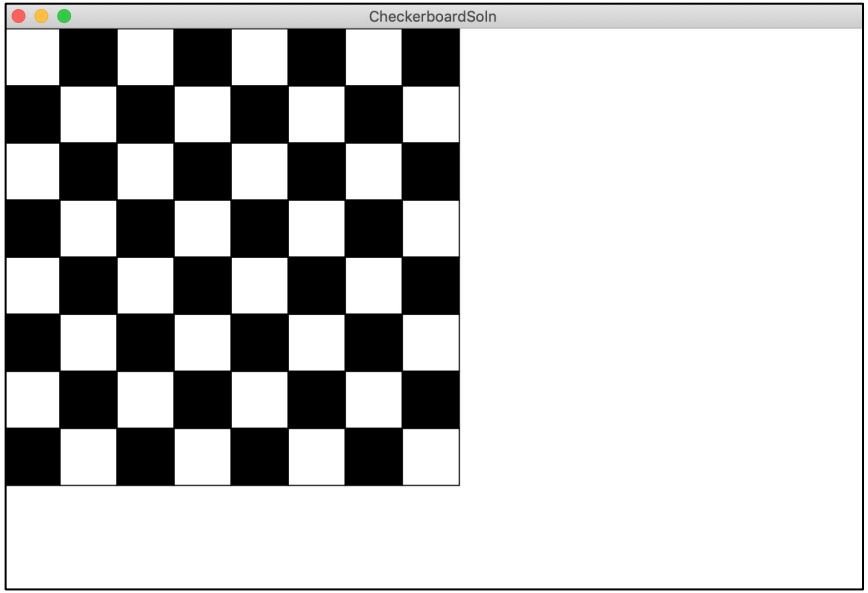
Our First Graphics Challenge



Pseudocode for drawing all of the boxes:

repeat for 8 rows:
    repeat for 8 columns:
        draw a box
        *(x location is col# * box size)*
        (y location is row# * box size)
        add the box to the screen

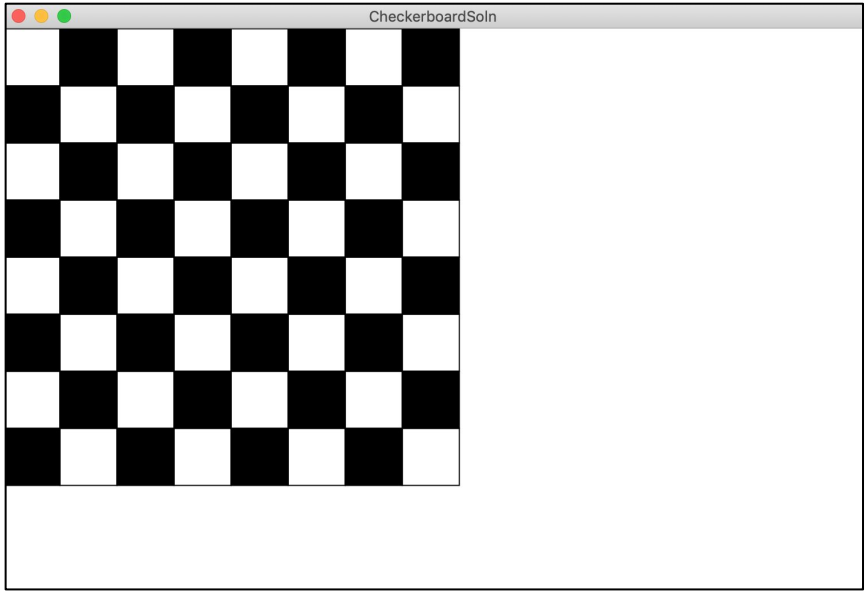
Our First Graphics Challenge



How can we change the Pseudocode to fill in the correct boxes?

repeat for 8 rows:
    repeat for 8 columns:
        draw a box
        *(x location is col# * box size)*
        *(y location is row# * box size)*
        add the box to the screen

Our First Graphics Challenge



Pseudocode for filling the correct boxes:

repeat for 8 rows:
    repeat for 8 columns:
        draw a box
        *(x location is col# * box size)*
        *(y location is row# * box size)*
        If row+column is odd:
            color the box black
        add the box to the screen

Let’s Code It!

Infinite Loops

What's an infinite loop?

Infinite Loops

Infinite loops are loops that *never end*.

Why would this ever happen?

Example: Infinite Loops

What is wrong with this loop?

```
int answer = readInt("What is 1 + 1?");

while (answer != 2){
     println("Wrong answer! Try again");
}
```

Example: Infinite Loops

What is wrong with this loop?

It doesn't give the user a chance to change their answer.

```
int answer = readInt("What is 1 + 1?");

while (answer != 2){
     println("Wrong answer! Try again");
}
```

Example: Infinite Loops

What is wrong with this loop?

It doesn't give the user a chance to change their answer.

This means the while loop will never end! This is effectively a `while(true)` loop!

```
int answer = readInt("What is 1 + 1?");

while (answer != 2){
     println("Wrong answer! Try again");
}
```

Example: Infinite Loops

How can we fix it?

```
int answer = readInt("What is 1 + 1?");

while (answer != 2){
     println("Wrong answer! Try again");
}
```

Example: Infinite Loops

How can we fix it?

We can add a line where we give the user a chance to change their answer!

```
int answer = readInt("What is 1 + 1?");

while (answer != 2){
     println("Wrong answer! Try again");
     answer = readInt("What is 1 + 1?");
}
```

Plan for Today

- Review: Methods
- Nested For Loop with Ascii
- Intro to Graphics Program - Our First Graphics Program!
- Checkerboard
- Infinite Loops

Reminders:

- Assignment 2 is due at **10am on Wednesday, July 10th**.
- No lecture or sections Thursday for 4th of July.
- No LaIR Wednesday, July 3rd due to Holiday.