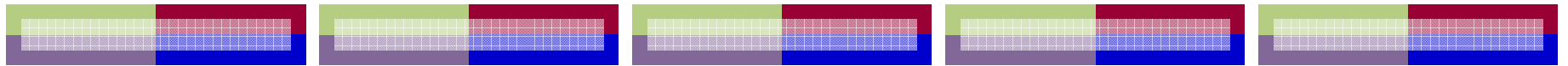


CS101

# Problem Solving and Object-Oriented Programming

## L14: Randomness and Conditionals



# Recall Colors

```
import java.awt.Color;    // tells java where Color is defined
```

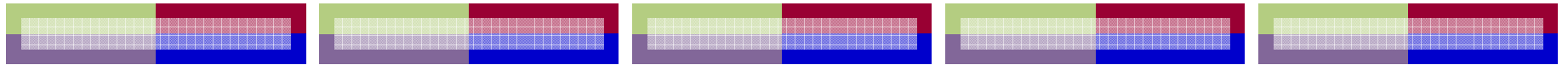
Pre-defined Colors:

BLACK, BLUE, CYAN, DARK\_GRAY, GRAY, GREEN, LIGHT\_GRAY,  
MAGENTA, ORANGE, PINK, RED, WHITE, YELLOW

● Color defined by intensities of red, blue, and green it contains

- Values range between 0 and 255      Why 255?
  - Example: Red = 129, Green = 202, Blue = 60
- Can be used to define new colors
  - e.g. `private static final Color DK_BROWN = new Color (95, 83, 61);`

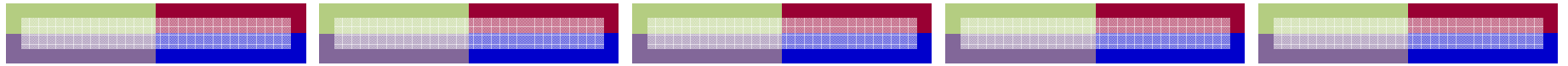




## Why 255?

- Computer memory uses binary
- In binary, there are just 2 values: 0 and 1
- Easy to implement physically: positive or negative magnetic charge
- A bit is a unit of memory that can hold a single binary digit (0 or 1)
- A byte is a unit of memory that can hold 8 bits, the values 00000000 to 11111111
  - What is binary 00000000 in decimal?
  - What is binary 11111111 in decimal?





# Decimal & Binary

## ● Decimal:

- Digits range from 0 to 9
- What is 623?
- from elementary school: 6 in 100's place, 2 in 10's place, and 3 in one's place
- Position determines power of 10
- $623 = 6 * 10^2 + 2 * 10^1 + 3 * 10^0$   
 $= 6 * 100 + 2 * 10 + 3 * 1$





# Decimal & Binary

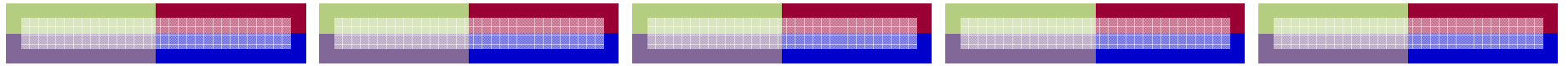
## Binary:

- Digits range from 0 to 1
- Position,  $p$ , determines power of 2
  - rightmost bit (least significant bit or lsb) is bit 0 or  $2^0$
  - each bit to left gets next successive number

For 8-bits: 0 0 1 0 1 0 1 1  
p7 p6 p5 p4 p3 p2 p1 p0

- What should binary 101 represent?
  - $101_2 = 1 * 2^2 + 0 * 2^1 + 1 * 2^0 = 1*4 + 0*2 + 1*1 = 5$  in decimal
- Why do color intensities range from 0-255?
  - $11111111 = 1 * 2^7 + 1 * 2^6 + 1 * 2^5 + 1 * 2^4 + 1 * 2^3 + 1 * 2^2 + 1 * 2^1 + 1 * 2^0 = 255$  in decimal





# Randomness

● In Alice, world functions generated random values:

- Random boolean:

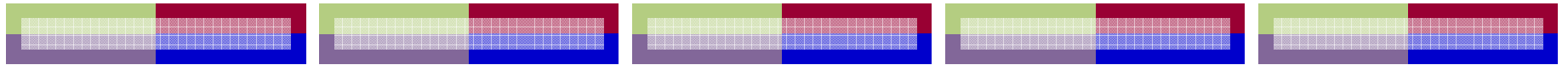
```
if (Random.nextBoolean())
```

- Random number:

```
randomNum.set (value, Random.nextDouble()
```

```
    minimum = 1 maximum = 11 integerOnly = true);
```





# Randomness in Java takes 2 steps

1. Construct a random number generator:

```
RandomIntGenerator generator = new RandomIntGenerator (1, 10);
```

2. Generate a random number:

```
int randomNum = generator.nextValue();
```

Example:

```
// object that represents a single die
```

```
private RandomIntGenerator die = new RandomIntGenerator(1,NUM_SIDES);
```

```
// Value of each die on a given roll
```

```
private int roll1, roll2;
```

```
// Roll the dice with each click
```

```
public void onMouseClick (Location point) {
```

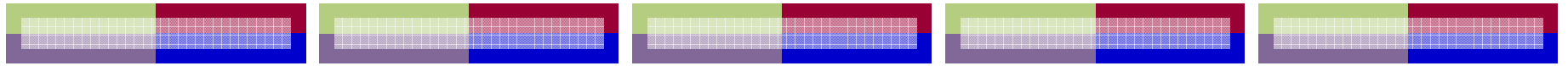
```
    roll1 = die.nextValue();
```

```
    roll2 = die.nextValue();
```

```
    result.setText("You rolled a " + roll1 + " and a " + roll2 + " for a total of " + (roll1 + roll2));
```

```
}
```





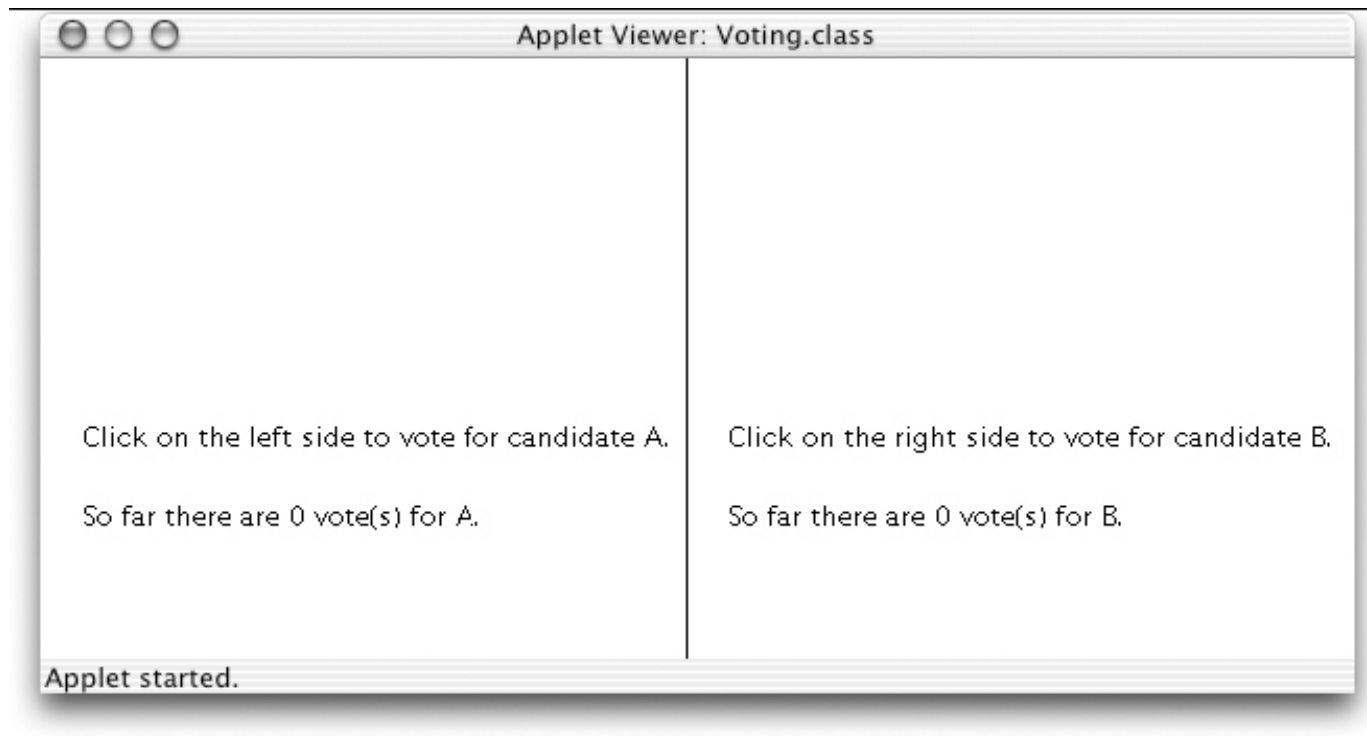
# Conditional Statements

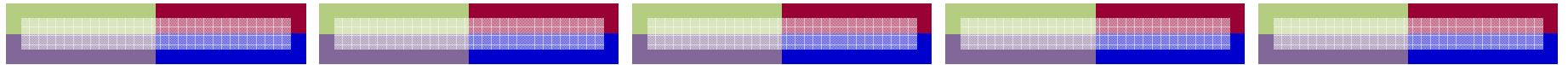
- Consider a program that does the following:
  - if the mouse location is contained in the rectangle,  
displays message “success”
- You’ve seen something like this before when you used Alice.  
What was it?
- Some programming constructs can choose between  
blocks of code to execute e.g. if-else





# A Vote Counting Example





## Code to Update Votes:

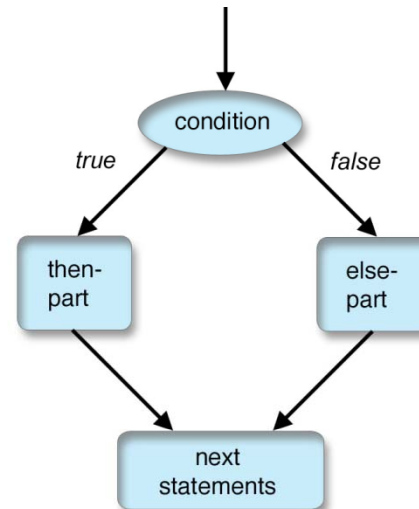
```
// Update votes and display vote counts
public void onMouseClick( Location point ) {
    if ( point.getX() < MID_X ) {
        countA++;
        infoA.setText( "So far there are " + countA +
            " vote(s) for A." );
    } else {
        countB++;
        infoB.setText( "So far there are " + countB +
            " vote(s) for B." );
    }
}
```



# Syntax of the `if` Statement

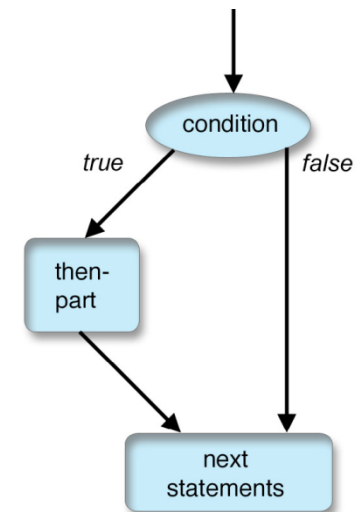
- The condition is evaluated. If true, the bracketed code following `if` is executed; otherwise, the bracketed code after `else` is executed

```
if (condition) {  
    //do something  
} else {  
    //do something else  
}
```



- The else part can be omitted

```
if (condition) {  
    //do something here  
}
```





## if Statement and 2D Objects

To check if the mouse is within a 2D object:


```
public void onMouseClick ( Location point ) {  
    if ( anObject.contains ( point ) ) {  
        //do something here  
    }  
}
```

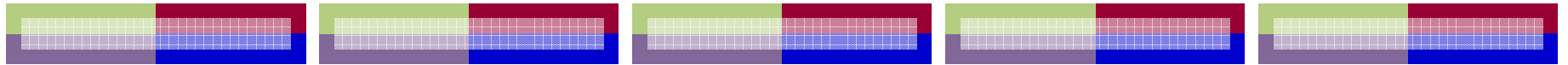




# Comparison Operators

Operator	Description
>	Greater than
<	Less than
==	Equal to
<=	Less than or equal to
>=	Greater than or equal to
!=	Not equal to





## Examples

If  $a=25$ ,  $b=30$

$a < b$  evaluates to true

$a \leq b$  evaluates to true

$a == b$  evaluates to false

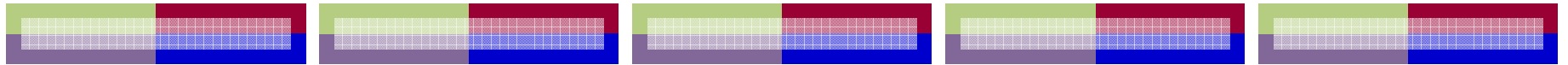
$a != b$  evaluates to true

$a > b$  evaluates to false

$a \geq b$  evaluates to false

Since these expressions are evaluated to either true or false, they are called boolean expressions





# Dragging a Box

*//boolean variable to determine whether the box is grabbed*

```
private boolean boxGrabbed;
```

*// Save starting point and whether point was in box*

```
public void onMousePress( Location point ) {
```

```
    lastPoint = point;
```

```
    boxGrabbed = box.contains( point );
```

```
}
```

*// if mouse is in box, then drag the box*

```
public void onMouseDrag( Location point ) {
```

```
    if ( boxGrabbed ) {
```

```
        box.move( point.getX() - lastPoint.getX(), point.getY() - lastPoint.getY() );
```

```
        lastPoint = point;
```

```
    }
```

```
}
```

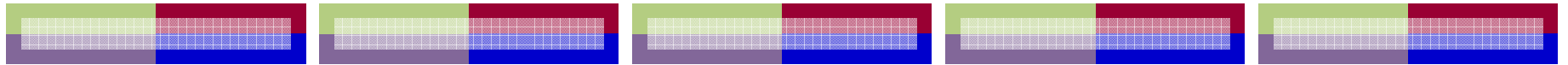
Of key importance:

1. Remember location of mouse

Of key importance:

2. Update mouse location after each drag event





# Dragging a Box

*//What if we do this without using a boolean variable to determine whether the box is grabbed?*

*// Save starting point and whether point was in box*

```
public void onMousePress( Location point ) {  
    lastPoint = point;  
}
```

*// if mouse is in box, then drag the box*

```
public void onMouseDrag( Location point ) {  
    if ( box.contains (point) ) {  
        box.move( point.getX() - lastPoint.getX(), point.getY() - lastPoint.getY() );  
        lastPoint = point;  
    }  
}
```

Yes, but can cause undesired effects:

*if the mouse is moved too quickly, the new location of the mouse may no longer be on the box after moving*





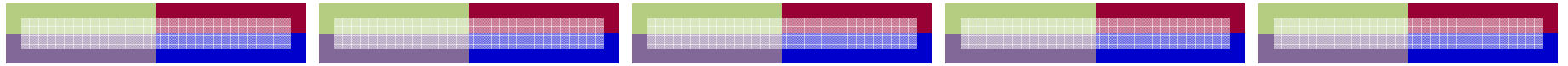
## More Uses of The `if else` Statement

- Picks one choice among many

EX: Converting a score into a letter grade

```
if ( score >= 90 ) {  
    gradeDisplay.setText( "The grade is A" );  
} else if ( score >= 80 ) {  
    gradeDisplay.setText( "The grade is B" );  
} else if ( score >= 70 ) {  
    gradeDisplay.setText( "The grade is C" );  
} else {  
    gradeDisplay.setText( "No credit is given");  
}
```





# Combining Multiple Conditionals

- `&&` (and) combines adjoining conditions in a way that the final result will be **true** only if all are **true**  
Ex: `a && b && c`  
is **true** if a,b,c are **true**
- `||` (or) combines adjoining conditions in a way that if any of them is **true**, the final result will be **true**  
Ex: `a || b || c`  
is **true** if any of a, b, c, is **true**

