Computer Programming Fundamentals

CS 152
Professor: Leah Buechley
TAs: Melody Horn, Noah Garcia, Andrew Geyko, Juan Ormaza
Time: MWF 10:00-10:50am
https://handandmachine.cs.unm.edu/classes/CS152_Fall2021/

WHERE WE ARE IN THE SEMESTER

ASSIGNMENT 4 MOSTLY GRADED

MIDTERM GRADES BY FRIDAY

questions?

CREATE A NEW PROJECT
“Week9”

CREATE A NEW CLASS
“MouseInteraction”

COPY AND PASTE SAMPLE CODE MyPanel.java AND MyFrame.java INTO THE SAME FILE

https://handandmachine.cs.unm.edu/classes/CS152_Fall2021/assignments/MyPanel.java

https://handandmachine.cs.unm.edu/classes/CS152_Fall2021/sampleCode/MyFrame.java

ENTIRE PROGRAM

```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;

public class MyPanel extends JPanel implements KeyListener {
    int width;
    int height;
    char keyPressed;
    int keyCode;

    MyPanel(int w, int h) {
        width = w;
        height = h;
        Dimension d = new Dimension(width, height);
        setPreferredSize(d);
        addKeyListener(this);
        setFocusable(true);
        requestFocusInWindow();
        setVisible(true);
    }

    public static void main(String[] args) {
        MyPanel panel = new MyPanel(500,500);
        MyFrame f = new MyFrame(panel);
        panel.animate(60);
    }

    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        //do your drawing here
    }

    void delay(int time) {
        try{ Thread.sleep(time); }
        catch (Exception exc){}
    }

    void animate (int framerate) {
        int delay = 1000/framerate;
        while(true) {
            repaint();
            delay(delay);
        }
    }

    @Override
    public void keyTyped(KeyEvent e) {
        keyPressed = e.getKeyChar();
        System.out.println(keyPressed);
    }

    @Override
    public void keyPressed(KeyEvent e) {
        keyCode = e.getKeyCode();
    }

    @Override
    public void keyReleased(KeyEvent e) {}
}
```

MyPanel class

```java
public class MyFrame {
    private JFrame j;
    private JPanel panel;

    MyFrame (JPanel p) {
        panel = p;
        j = new JFrame();
        j.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        j.add(panel);
        j.pack();
        j.setVisible(true);
    }
}
```

MyFrame class

DELETE KEYWORD “public” FROM MyFrame CLASS

```
~~public~~ class MyFrame {
    private JFrame j;
    private JPanel panel;

    MyFrame (JPanel p) {
        panel = p;
        j = new JFrame();
        j.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        j.add(panel);
        j.pack();
        j.setVisible(true);
    }
}
```

RENAME MyPanel MouseInteraction

```
public class MyPanel extends JPanel implements KeyListener {
    int width;
    int height;
    char keyPressed;
    int keyCode;

    MyPanel(int w, int h) {
        width = w;
        height = h;
        Dimension d = new Dimension(width, height);
        setPreferredSize(d);
        addKeyListener(this);
        setFocusable(true);
        requestFocusInWindow();
        setVisible(true);
    }

    public static void main(String[] args) {
        MyPanel panel = new MyPanel(500,500);
        MyFrame f = new MyFrame(panel);
        panel.animate(60);
    }
```

RENAME MyPanel MouseInteraction

```
public class MouseInteraction extends JPanel implements KeyListener {
    int width;
    int height;
    char keyPressed;
    int keyCode;

    MyPanel(int w, int h) {
        width = w;
        height = h;
        Dimension d = new Dimension(width, height);
        setPreferredSize(d);
        addKeyListener(this);
        setFocusable(true);
        requestFocusInWindow();
        setVisible(true);
    }

    public static void main(String[] args) {
        MouseInteraction panel = new MouseInteraction(500,500);
        MyFrame f = new MyFrame(panel);
        panel.animate(60);
    }
```

COMPILE AND RUN

YOU CAN DEFINE MORE THAN
ONE CLASS IN A FILE

ONLY ONE CAN BE PUBLIC
THE ONE WITH SAME NAME AS FILE

questions?

TODAY: MOUSE INTERACTION

USING MouseMotionListener

ADD MOUSE INTERACTION

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseMotionListener;

public class MouseInput extends JPanel implements KeyListener, MouseMotionListener {
```

tells compiler you'll be using mouse input
you'll be "listening" for mouse movement

ADD 2 METHODS

```
@Override
public void mouseDragged(MouseEvent e) {

}

@Override
public void mouseMoved(MouseEvent e) {

}
```

code that will run when the mouse is dragged

code that will run when the mouse is moved

note: I will not expect you to remember these on any exam or quiz

COMPILE & RUN TO CHECK CODE

ADD 2 VARIABLES FOR MOUSE LOCATION INFORMATION

```
public class MouseInput extends JPanel implements KeyListener,
MouseMotionListener {
    int width;
    int height;
    char keyPressed;
    int keyCode;
    int mouseX, mouseY;
```

will store the location of the mouse pointer

CAN DEFINE MULTIPLE VARIABLES ON ONE LINE AS LONG AS THEY'RE THE SAME TYPE

```
public class MouseInput extends JPanel implements KeyListener,
MouseMotionListener {
    int width;
    int height;
    char keyPressed;
    int keyCode;
    int mouseX, mouseY;
```

will store the location of the mouse pointer

WHAT HAPPENS IF WE DON’T INITIALIZE THEM IN THE CONSTRUCTOR?

WHAT VALUES DO THEY HAVE?

IN CONSTRUCTOR

```
MouseInput(int w, int h) {
    width = w;
    height = h;
    Dimension d = new Dimension(width, height);
    setPreferredSize(d);
    addKeyListener(this);
    addMouseMotionListener(this);
    setFocusable(true);
    requestFocusInWindow();
    setVisible(true);
    System.out.println("The initial value of mouseX is:" +mouseX);
    System.out.println("The initial value of mouseY is:" +mouseY);
}
```

```
The initial value of mouseX is:0
The initial value of mouseY is:0
```

THE VALUE STORED IN AN int VARIABLE THAT IS NOT INITIALIZED IS ZERO

THE DEFAULT VALUE OF AN int VARIABLE IS ZERO

questions?

BACK TO OUR CODE USING THE VARIABLES

EDIT THE mouseMoved METHOD

```
@Override
public void mouseMoved(MouseEvent e) {
    mouseX = e.getX();
    mouseY = e.getY();
    System.out.println("The location of the mouse is: " +mouseX + ", " +mouseY );
}
```

store the mouse pointer location into our mouseX and mouseY variables

COMPILE & RUN TO CHECK CODE

```
The location of the mouse is: 334, 359
The location of the mouse is: 333, 357
The location of the mouse is: 333, 357
The location of the mouse is: 332, 355
The location of the mouse is: 332, 355
The location of the mouse is: 332, 354
The location of the mouse is: 332, 354
```

questions?

USING THE MOUSE TO CONTROL GRAPHICS

EDIT THE paintComponent METHOD

```
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    setBackground(Color.WHITE);
    int size = 50;
    g.setColor(Color.BLUE);
    g.fillOval(mouseX-size/2,mouseY-size/2,size,size);
}
```

what does this code do?

COMPILE & RUN

MORE PLAY

```
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    setBackground(Color.WHITE);
    int size = 50;
    if (mouseX<width/2)
        g.setColor(Color.MAGENTA);
    else
        g.setColor(Color.ORANGE);
    g.fillOval(mouseX-size/2,mouseY-size/2,size,size);
}
```

what does this code do?

EVENT DRIVEN DRAWING

STOP ANIMATING

```
public static void main(String[] args) {
    MouseInput panel = new MouseInput(500,500);
    MyFrame f = new MyFrame(panel);
    ~~panel.animate(60);~~
}
```

COMPILE & RUN

repaint() IN mouseMoved()

```java
@Override
public void mouseMoved(MouseEvent e) {
    mouseX = e.getX();
    mouseY = e.getY();
    System.out.println("The location of the mouse is: " +mouseX + ", " +mouseY );
    repaint();
}
```

store the mouse pointer location into our
mouseX and mouseY variables

WILL REDRAW THE SCREEN
ONLY WHEN THE MOUSE MOVES

COMPILE & RUN

questions?

PAINTING ONLY NEW THINGS INSTEAD OF THE ENTIRE SCREEN

NEED A WAY TO ADD STUFF TO GRAPHICS OUTSIDE OF paintComponent()

ADD A VARIABLE FOR GRAPHICS

```
public class MouseInput extends JPanel implements KeyListener, MouseMotionListener
{
    int width;
    int height;
    char keyPressed;
    int keyCode;
    int mouseX, mouseY;
    Graphics g;
```

will store the graphics object we’re drawing everything on

DELETE DRAWING CODE IN paintComponent()

```
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    setBackground(Color.WHITE);
    int size = 50;
    if (mouseX<width/2)
        g.setColor(Color.MAGENTA);
    else
        g.setColor(Color.ORANGE);
    g.fillOval(mouseX-size/2,mouseY-size/2,size,size);
}
```

ADD CODE TO mouseMoved()

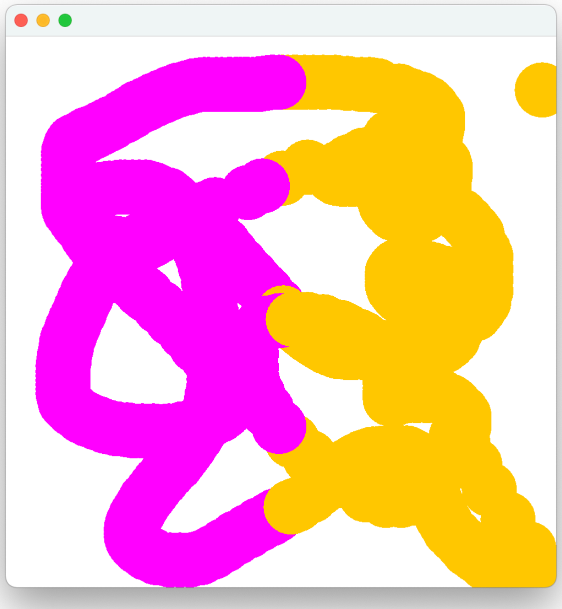
```java
@Override
public void mouseMoved(MouseEvent e) {
    mouseX = e.getX();
    mouseY = e.getY();
    System.out.println("The location of the mouse is: " +mouseX + ", " +mouseY );
    g = getGraphics();
}
```

First, get the current graphics object to draw on

ADD DRAWING CODE TO mouseMoved()

```
@Override
public void mouseMoved(MouseEvent e) {
    mouseX = e.getX();
    mouseY = e.getY();
    System.out.println("The location of the mouse is: " +mouseX + ", " +mouseY );
    g = getGraphics();
    int size = 50;
    if (mouseX<width/2)
        g.setColor(Color.MAGENTA);
    else
        g.setColor(Color.ORANGE);
    g.fillOval(mouseX-size/2,mouseY-size/2,size,size);
}
```

ADD DRAWING CODE TO mouseMoved()



NEW STUFF IS DRAWN
ON TOP OF OLD STUFF
SCREEN IS NEVER CLEARED

LET’S CLEAR THE SCREEN WHEN SPACEBAR IS PRESSED

WHERE DO WE WRITE THIS CODE?

keyTyped() method!

ADD CODE TO keyTyped()

```
@Override
public void keyTyped(KeyEvent e) {
    keyPressed = e.getKeyChar();
    System.out.println(keyPressed);
    if (keyPressed==' ') {
        //??
    }
}
```

what do we do to clear the screen?

repaint()

ADD CODE TO keyTyped()

```java
@Override
public void keyTyped(KeyEvent e) {
    keyPressed = e.getKeyChar();
    System.out.println(keyPressed);
    if (keyPressed==' ') {
        repaint();
    }
}
```

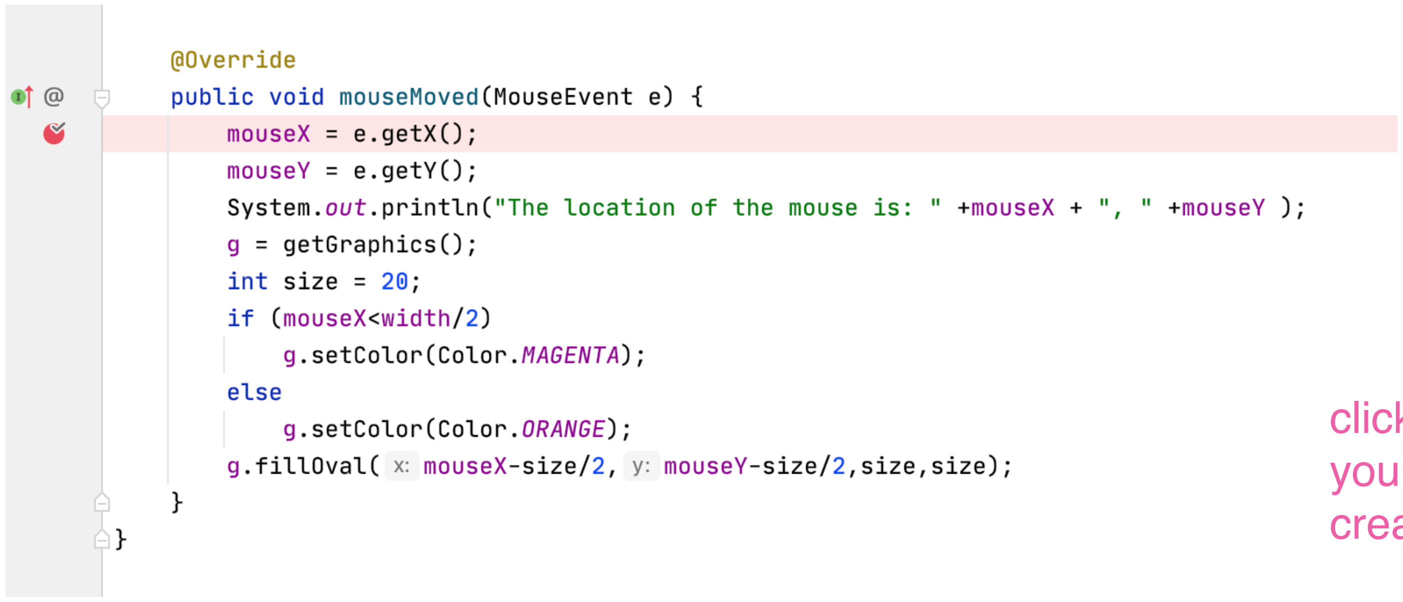
questions?

THE DEBUGGER

THE DEBUGGER

- Can stop your program anywhere and look at variable values
- Can execute your program one line of code at a time and see the values of variables as you step through the program
- Can be overwhelming or confusing at first, but try it out!
- If it doesn’t seem useful yet, stick to print statements

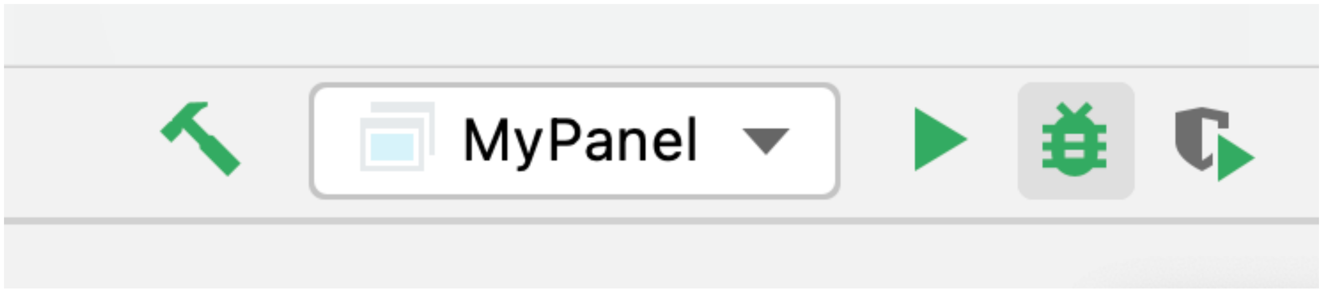
USING THE DEBUGGER CREATE A “BREAKPOINT”



```
    @Override
    public void mouseMoved(MouseEvent e) {
        mouseX = e.getX();
        mouseY = e.getY();
        System.out.println("The location of the mouse is: " +mouseX + ", " +mouseY );
        g = getGraphics();
        int size = 20;
        if (mouseX<width/2)
            g.setColor(Color.MAGENTA);
        else
            g.setColor(Color.ORANGE);
        g.fillOval( x: mouseX-size/2, y: mouseY-size/2,size,size);
    }
}
```

click next to the line where you want to stop and create a red dot

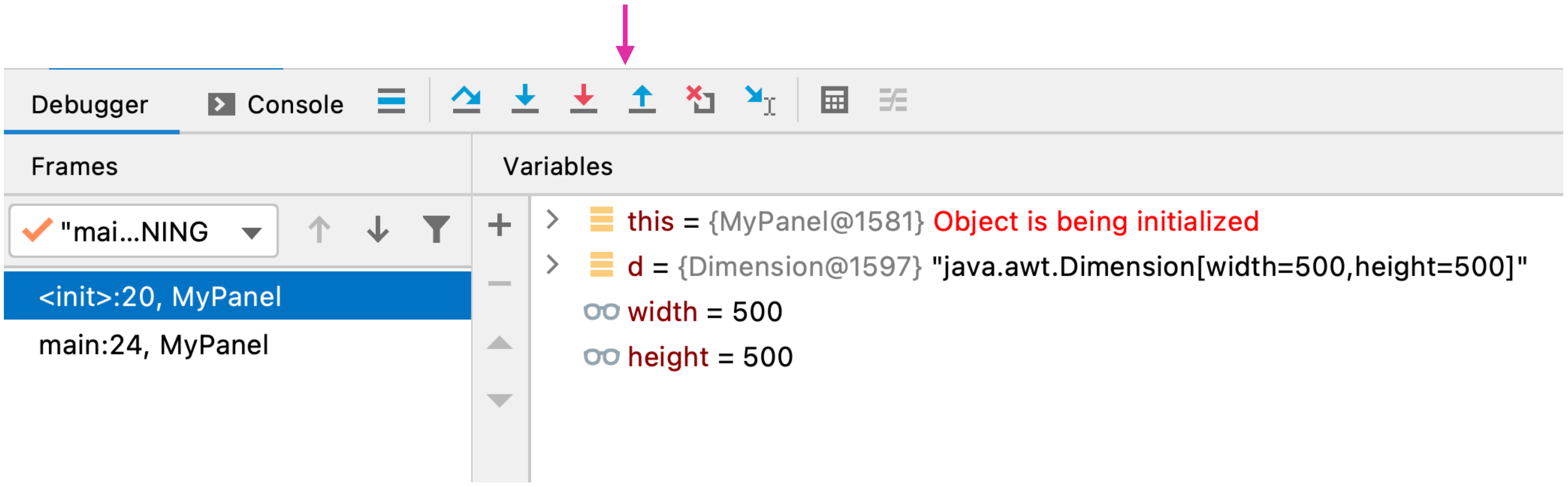
RUN IN DEBUGGING MODE



click on the bug icon at the top of your window

PROGRAM WILL STOP AT BREAKPOINT

can now use debugging tools to control flow through your program



```
<init>:20, MyPanel
main:24, MyPanel
```

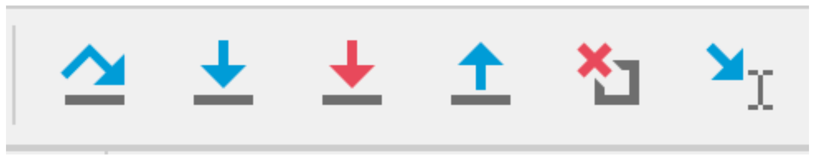
```
> this = {MyPanel@1581} Object is being initialized
> d = {Dimension@1597} "java.awt.Dimension[width=500,height=500]"
  width = 500
  height = 500
```

PROGRAM WILL STOP AT BREAKPOINT

step over: executes the line of code & moves on

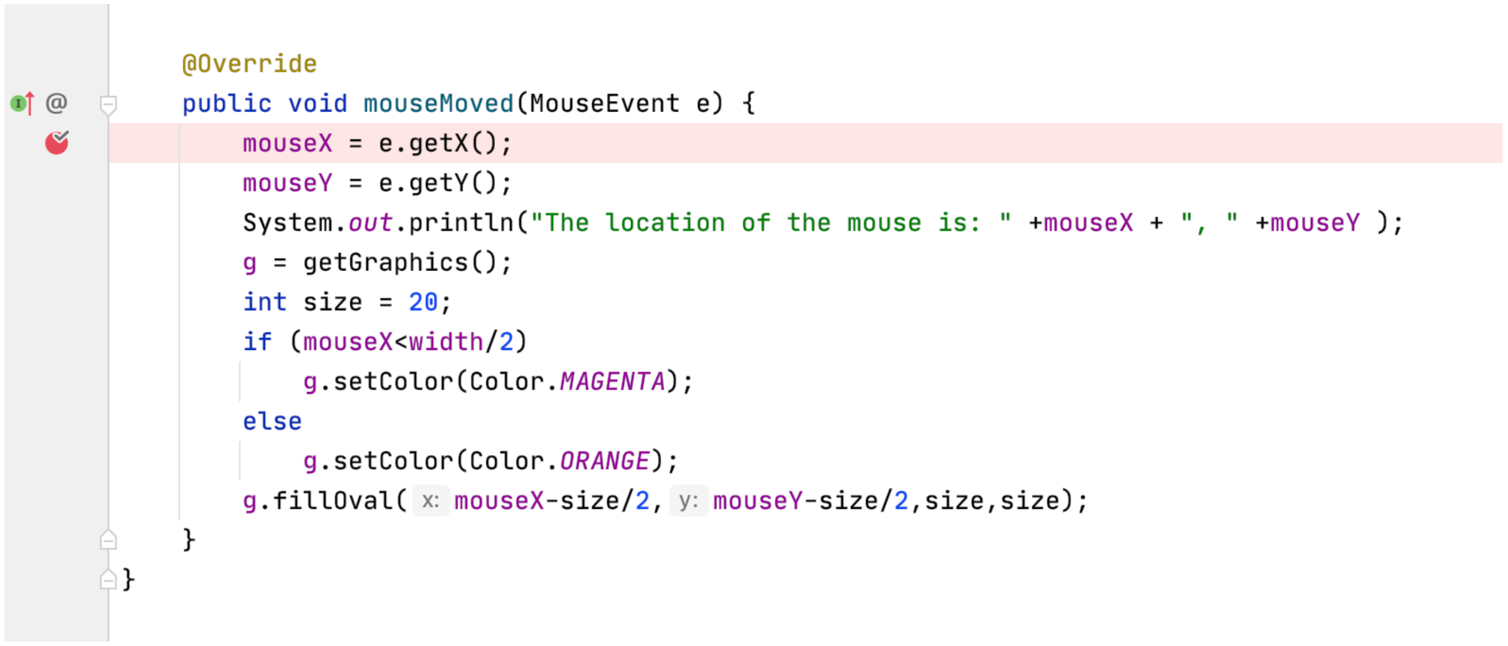
step into: executes the line of code in detail (jumps to methods)

step out: goes back a step or level (jumps out of methods)



run to cursor: executes code up until cursor position then stops

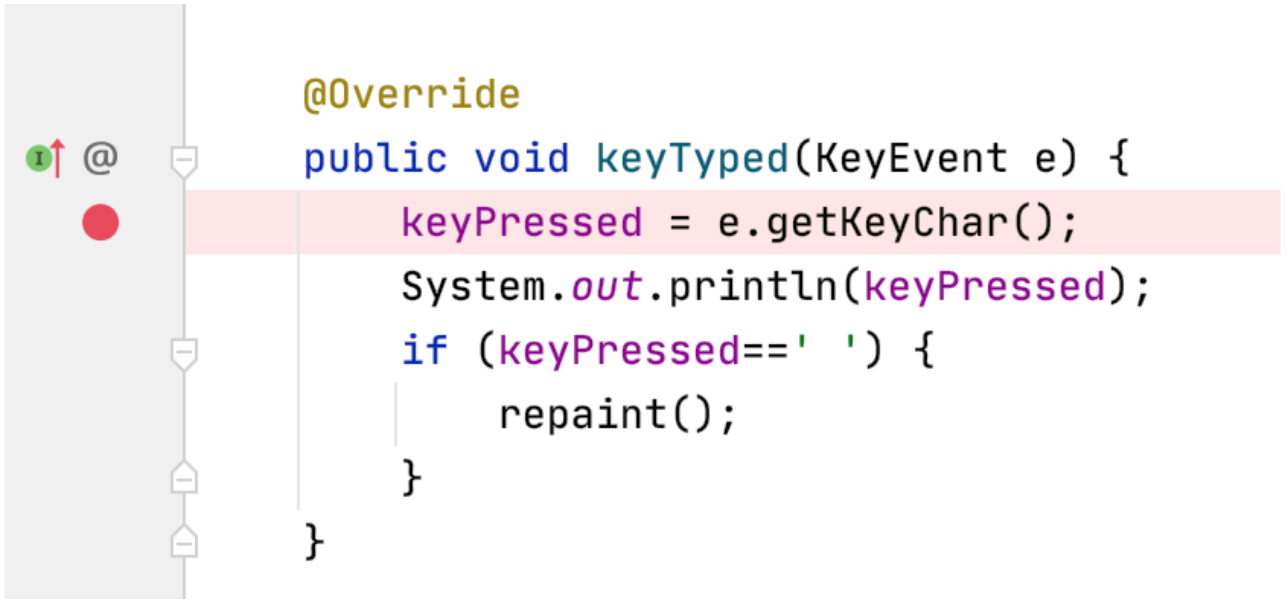
CREATE A “BREAKPOINT” IN mouseMoved()



```
    @Override
    public void mouseMoved(MouseEvent e) {
        mouseX = e.getX();
        mouseY = e.getY();
        System.out.println("The location of the mouse is: " +mouseX + ", " +mouseY );
        g = getGraphics();
        int size = 20;
        if (mouseX<width/2)
            g.setColor(Color.MAGENTA);
        else
            g.setColor(Color.ORANGE);
        g.fillOval( x: mouseX-size/2, y: mouseY-size/2,size,size);
    }
}
```

DEBUG

CREATE A “BREAKPOINT” IN keyTyped()



```
@Override
public void keyTyped(KeyEvent e) {
    keyPressed = e.getKeyChar();
    System.out.println(keyPressed);
    if (keyPressed==' ') {
        repaint();
    }
}
```

DEBUG

questions?

Thank you!

CS 152
Professor: Leah Buechley
TAs: Melody Horn, Noah Garcia, Andrew Geyko, Juan Ormaza
Time: MWF 10:00-10:50am
https://handandmachine.cs.unm.edu/classes/CS152_Fall2021/