CSE 8B Today

Don’t forget: Fill out your CAPES, TA evaluations, and your survey

Just 3 more days of class!
Today: More Threads and Exceptions
Wednesday: Finish Threads and Exceptions
Friday: Wrap up and review

Discussions this week:
Today and Wednesday: PSA 8
Friday: Final exam review (partial—it’s only 1 hour!!)

How’s PSA8 going?
A. I haven’t started at all
B. I’ve read it and started thinking about it
C. I’ve started some coding
D. I’m done

The final exam will look a lot like the interm exams, only longer
Designed to take about 1.5-2 hours (you will have 3)
Study by RE-DOING clicker questions and PSAs!!
1. When does a runtime error occur?

A. While your program is executing.

B. When you compile your program.

C. When you write your program.

D. Before your program runs.
2. What is a catch block used for?

A. It isn’t used for anything.

B. It lets you run code that would normally contain compile errors.

C. It handles exceptions that are thrown at runtime.

D. It tries to run code that might throw an exception.
3. You can define your own exceptions by extending which class?

A. Error
B. Exception
C. Throws
D. TryCatch

B. Exception
public class ConnectFourAnimationDemo extends JFrame implements Runnable {
    private boolean running; // Should the animation run?
    public ConnectFourAnimationDemo() {
        ...
        new Thread( this ).start();
    }

    public void run() {
        try {
            int panelNum = 0;
            while( true ) {
                if ( running ) {
                    if ( board[panelNum] == ' ' )
                        board[panelNum] = turn;
                    else
                        board[panelNum] = ' ';
                    panelNum = (panelNum + 1) % 3;
                    repaint();
                }
                Thread.sleep( 500 );
            }
        } catch (InterruptedException ex) { }
    }
}

Can be run in a separate thread

A class variable sets the “state” of the animation

Launch the run method in a separate thread.

repaint asks the main GUI thread to repaint. But sleep happens in a separate thread so is doesn’t affect the main thread (in fact it allows it to run again!)
class StartStopListener implements ActionListener
{
    public void actionPerformed( ActionEvent e )
    {
        JButton src = (JButton)e.getSource();
        if (src.getText().equals( "Start" )) {
            running = true;
        }
        else if (src.getText().equals( "Stop" )) {
            running = false;
        }
    }
}
public void run() {
    try {
        int panelNum = 0;
        while (true) {
            if (running) {
                if (board[panelNum] == ' ') {
                    board[panelNum] = turn;
                } else {
                    board[panelNum] = ' ';
                }
                panelNum = (panelNum + 1) % 3;
                repaint();
            }
            Thread.sleep(500);
        }
    } catch (InterruptedException ex) {}
}

_startStopListener_.implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        JButton src = (JButton)e.getSource();
        if (src.getText().equals("Start")) {
            running = true;
        } else if (src.getText().equals("Stop")) {
            running = false;
        }
    }
}
Currently, our ConnectFourAnimationDemo object IS our task object

```java
public class ConnectFourAnimationDemo extends JFrame implements Runnable {
    ...
    public void run() {
        try {
            int panelNum = 0;
            while (true) {
                if (running) {
                    if (board[panelNum] == ' ') {
                        board[panelNum] = turn;
                    } else {
                        board[panelNum] = ' ';
                    }
                    panelNum = (panelNum + 1) % 3;
                    repaint();
                }
                Thread.sleep(500);
            }
        } catch (InterruptedException ex) { }
    }
}
```

In the ConnectFourAnimationDemo constructor, what line of code will execute the ConnectFourAnimationDemo’s run method in a new Thread?

A. `Thread.start( this );`  
B. `new Thread( this ).start();`  
C. `this.run();`  
D. `ConnectFourAnimationDemo.run();`  

**B. new Thread( this ).start();**
PanelAnimator is an inner class of ConnectFourAnimationDemo class:

```java
class PanelAnimator implements Runnable {
    int panelNum;
    int delay;
    public PanelAnimator(int num, int delayIn) {
        panelNum = num;
        delay = delayIn;
    }
    public void run() {
        try {
            while (true) {
                if (running) {
                    if (board[panelNum] == ' ') {
                        board[panelNum] = turn;
                    } else {
                        board[panelNum] = ' ';
                    }
                    repaint();
                }
                Thread.sleep(delay);
            }
        } catch (InterruptedException ex) { }
    }
}
```

// In the ConnectFourAnimationDemo Constructor...
PanelAnimator p0 = new PanelAnimator(0, 500);

// What line of code will start this PanelAnimator running in a new Thread?

// PanelAnimator is an inner class of ConnectFourAnimationDemo

// In the ConnectFourAnimationDemo Constructor...
PanelAnimator p0 = new PanelAnimator(0, 500);

// What line of code will start this PanelAnimator running in a new Thread?

```java
A. p0.run();
B. p0.start();
C. new Thread(this).start();
D. new Thread(p0).start();
E. new Thread(p0).run();
```
PanelAnimator is an inner class of ConnectFourAnimationDemo class

PanelAnimator implements Runnable {
    int panelNum;
    int delay;
    public PanelAnimator( int num, delayIn ) {
        panelNum = num;
        delay = delayIn;
    }
    public void run() {
        try {
            while( true ) {
                if ( running ) {
                    if ( board[panelNum] == ' ' )
                        board[panelNum] = turn;
                    else
                        board[panelNum] = ' ';
                    repaint();
                }
                Thread.sleep( delay );
            }
        } // panelNum doesn't change
        catch ( InterruptedException ex) { } }
    }

// In the ConnectFourAnimationDemo constructor...
PanelAnimator p0 = new PanelAnimator( 0, 500 );
new Thread(p0).start();
} // End of C4AD constructor

// PanelAnimator is an inner class of ConnectFourAnimationDemo class
What will the GUI do when the user clicks “Start”?
A. All three circles will change colors automatically, like they did before.
B. Only the first circle will change colors automatically
C. Nothing will happen
class PanelAnimator implements Runnable {
    int panelNum;
    int delay;
    public PanelAnimator( int num, delayIn ) {
        panelNum = num;
        delay = delayIn;
    }
    public void run() {
        try {
            while( true ) {
                if ( running ) {
                    if ( board[panelNum] == ' ' )
                        board[panelNum] = turn;
                    else
                        board[panelNum] = '/';
                    repaint();
                }
                Thread.sleep( delay );
            }
        } catch (InterruptedException ex) { } }
}

// In the ConnectFourAnimationDemoConstructor...
PanelAnimator p0 = new PanelAnimator( 0, 500 );
PanelAnimator p1 = new PanelAnimator( 1, 700 );
PanelAnimator p2 = new PanelAnimator( 2, 500 );
new Thread(p0).start();
new Thread(p1).start();
new Thread(p2).start();

Add code to make the three circles change color at different speeds
What is an IOException?

Essentially, an exception is just an error that can be passed around (thrown). If `sourceFile` does not exist, the Scanner’s constructor will throw an exception.
Handling exceptions

// In class StringPlay
public String readIt( String filename )
{
    File sourceFile = new File( filename );
    Scanner input = null;
    try {
        input = new Scanner( sourceFile );
    }
    catch ( IOException e ) {
        System.out.println( e.getMessage() );
        return "";
    }
    String allText = "";
    while ( input.hasNextLine() )
    {
        String s1 = input.nextLine();
        allText = allText.concat( s1 );
    }
    System.out.println( allText );
    return allText;
}
public String readIt( String filename )
{
    File sourceFile = new File( filename );
    Scanner input = null;
    try {
        input = new Scanner( sourceFile );
    }
    catch ( IOException e ) {
        System.out.println( e.getMessage() );
        // return ""; NO MORE RETURN HERE
    }
    String allText = "";
    while ( input.hasNextLine() )
    {
        String s1 = input.nextLine();
        allText = allText.concat( s1 );
    }
    System.out.println( allText );
    return allText;
}
Handling exceptions

// In class StringPlay
public String readIt( String filename )
{
    File sourceFile = new File( filename );
    Scanner input = null;
    try {
        input = new Scanner( sourceFile );
    }
    catch ( IOException e ) {
        System.out.println( e.getMessage() );
        // return ""; NO MORE RETURN HERE
    }
    String allText = "";
    while ( input.hasNextLine() )
    {
        String s1 = input.nextLine();
        allText = allText.concat( s1 );
    }
    System.out.println( allText );
    return allText;
}

Why will this code compile, since it appears to throw an exception which is not caught or thrown?

A. It will only compile if we give the java compiler a special flag
B. A NullPointerException is an unchecked exception which doesn’t need to be caught
C. The exception is thrown within the try/catch block that is already in the code
Complete the code to prompt the user to enter a new filename until the file exists.

```java
public String readIt( String filename ) {
    File sourceFile = new File( filename );
    Scanner input = null;
    Scanner userInput = new Scanner( System.in );
    while ( input == null ) {
        try {
            input = new Scanner( sourceFile );
        } catch ( IOException e ) {
            System.out.println( e.getMessage() );
        }
        catch ( IOException e ) { // This catch block is unnecessary
            System.out.println( e.getMessage() );
        }
    }
    String allText = "";
    while ( input.hasNextLine() ) {
        String s1 = input.nextLine();
        allText = allText.concat( s1 );
    }
    System.out.println( allText );
    return allText;
}
```
Java Exceptions: Main Goal

• When you encounter a situation that won’t let your code act “normally”
  – Don’t “print an error” (printing to the screen isn’t the only way we interact with computers)
  – Throw an exception back to the location that called this method
  – It will then catch it and decide what to do

• Why throw it?
  – Scanner (written by Sun) doesn’t get to dictate the way YOU handle an exceptional circumstance