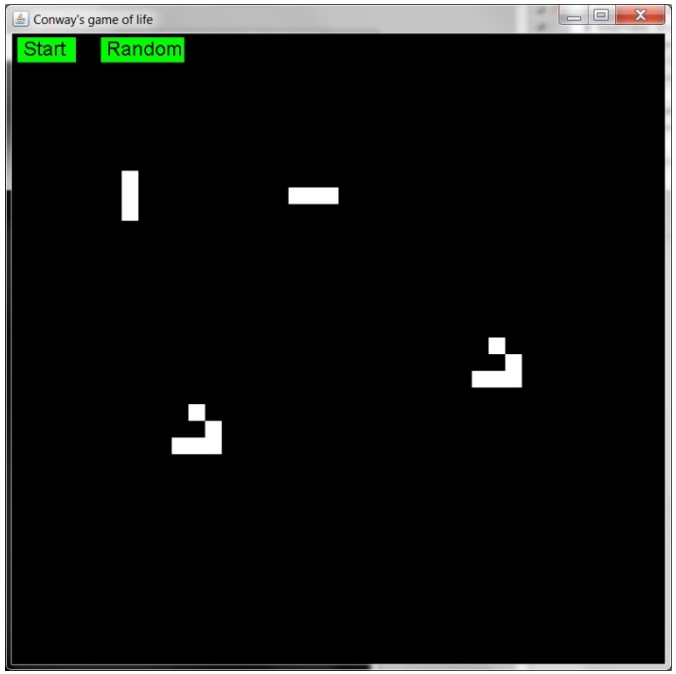
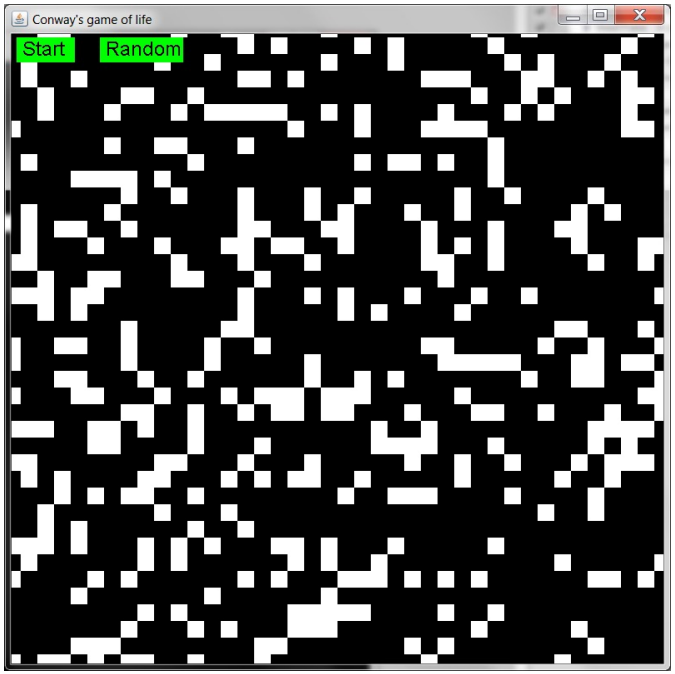
CT255
NGT2 – 2D Games in Java

Week 8
[2D Games in Java]

Dr. Sam Redfern
sam.redfern@universityofgalway.ie

Last week's assignment
Conway's Game of Life





- Add game states (playing and not playing)
- When not playing, render two rectangles as 'buttons'
- Modify the mousePressed method so that it checks for clicks on the button's regions
  - Start – switches the game state to 'playing'
  - Random – randomises the game state
- When in playing state, apply the rules of Conway's Game of Life at each repaint (see next slide)

Topics this week

- Loading and saving using text files
- Mouse move events
- Introducing A* pathfinding

Reading from text files

- The **java.io** package provides file handling classes
- **FileReader** to read from a text file
- **BufferedFileReader** to do so more efficiently (reads larger blocks and buffers/caches them)
- *Exception handling is required..*
- **BufferedFileReader:**
  - Use **FileReader** class constructor to open a file
  - Use **readLine()** method to read a line of text (returns a String)
  - Use **close()** method to close file

Sample code

```
String filename = "C:\\Users\\Sam\\Desktop\\lifegame.txt";
String textinput = null;
try {
   BufferedReader reader = new BufferedReader(new FileReader(filename));
   textinput = reader.readLine();
   reader.close();
}
catch (IOException e) { }
```

This reads just one line from the file (stopping at end of file or when a carriage return is encountered)

Sample Code

```
String line=null;
String filename = "C:\\Users\\Sam\\Desktop\\lifegame.txt";
try {
        BufferedReader reader = new BufferedReader(new FileReader(filename));
        do {
                try {
                        line = reader.readLine();
                        // do something with String here!

                } catch (IOException e) {  }
        }
        while (line != null);

        reader.close();

} catch (IOException e) { }
```

This reads all (CR-separated) lines from the file

Writing to text files

- Use the **FileWriter** and **BufferedWriter** classes
- **BufferedFileWriter:**
  - Use **FileWriter** class constructor to open a file
  - Use **write(String s)** method to write a line to the file (CR appended automatically)
  - Use **close()** method to close file
- E.g., to write a single string to a file:

```
String filename = "C:\\Users\\Sam\\Desktop\\lifegame.txt";
try {
   BufferedWriter writer = new BufferedWriter(new FileWriter(filename));
   writer.write(outputtext);
   writer.close();
 }
catch (IOException e) { }
```

Handling mouse motion events

- As well as mouse button events, we can also receive mouse movement events
- .. This is useful for making it less tedious to manually create a new initial game set-up
- Have the class implement the MouseMotionListener interface as well as MouseListener
- In the application class constructor:

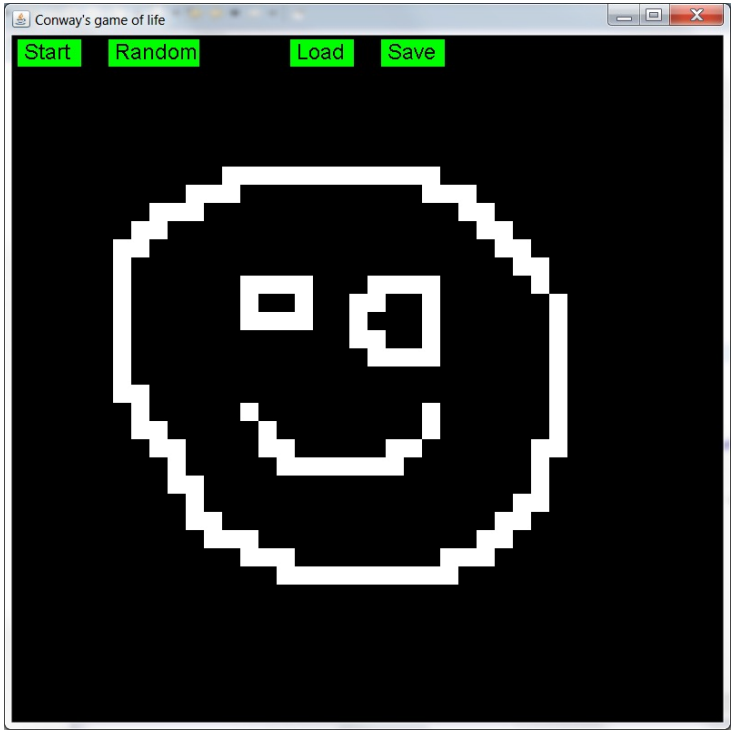
```
addMouseMotionListener(this);
```

- Add these methods (receives same data as the mouse events we have already seen):

```
public void mouseMoved(MouseEvent e)
public void mouseDragged(MouseEvent e)
```

This week's assignment

- Implement mouse dragging for game state setup
- Implement game state loading and saving (via 'buttons' as before)
  - How to encode the game state as string(s) ?
- Read the following A* webpage for next week!



http://www.psychicsoftware.com/AStarForBeginners.html

A* Pathfinding

- An important AI algorithm used in games and elsewhere
- Next week we will discuss implementing this in Java, and will use the Game of Life project as a basis for making a maze-solving AI game
- Read this A* webpage for next week!

  http://www.psychicsoftware.com/AStarForBeginners.html
- These are good introductions too:

  https://www.redblobgames.com/pathfinding/a-star/introduction.html
  https://www.raywenderlich.com/3016-introduction-to-a-pathfinding