|  |
| --- |
| Cell |
| -cellNum:int  -state:boolean  -nextState:boolean  -count:int |
| +Cell(int, boolean)  +Cell(int)  +set() methods for cellNum, state and nextState  +get() methods for cellNum, state and nextState  +isAlive(): boolean  +isDead(): boolean  +paint(): void  +resetCell(): void |

|  |
| --- |
| JFrame |
|  |

|  |
| --- |
| JButton |
|  |

1..\*

|  |
| --- |
| GameGUI |
| -size:int  +sizeSqrt:int  -cellboard:Cell[]  -running:boolean  -boardPanel:JPanel  -buttonPanel:JPanel  -stepButton:JButton  -resetButton:JButton  -randomizeButton:JButton  -runButton:JToggleButton |
| +GameGUI(int)  +setBoardSize(int):void  +initBoard():void  +reset():void  +randomizeBoard(): void  +updateBoard(): void  +gameLogic(): void  +isNeighborAlive(int):boolean  +backgroundRun(): void |

Game of life is a game with cells. You either turn the cells on or off and when you press start, they either live or they die based on four basic rules.

1. If the cell has two or less live neighbors, it dies
2. If the cell has two or three live neighbors, it lives until the next generation
3. If the cell has more than three live neighbors, it dies
4. If a dead cell has three live neighbors, it becomes alive.

The user can choose to change cells manually between turns or can let the cells be and see if they change next time the rules are run.

There are four buttons in the game. Stepbutton continues the game for one generation, resetbutton kills all the cells, randomize button randomizes the cell board and runbutton runs the rules with threads so the game does not slow down.