In Deliverable Four, I made modification of three methods which are Cell.toString(), MainPanel.runContinuous() and MainPanel.convertToInt() to improve the performance. These three methods are easily identified as "hot spots" of the application when using VisualVM as profiler to determine methods that consume most resources. To test the function of method doesn't change after modification. I used unit tests for Cell.toString() and MainPanel.convertToInt (), but manual testing for MainPanel.runContinuous() as this method has no return value.

Method One: Cell.toString()

After clicking the cells to change their state, I click "Run" button and then click "Write" button. According to the screenshot, it is obvious that Cell.toString() is a hot spot method, so I choose to modify this method. There is an unnecessary for loop in the method, as it only extract the first character of the string. I removed the for loop and keep the local fields in case they will be used in future. After modification, Cell.toString() is no longer in the Hot Spots-Method list.

Before:
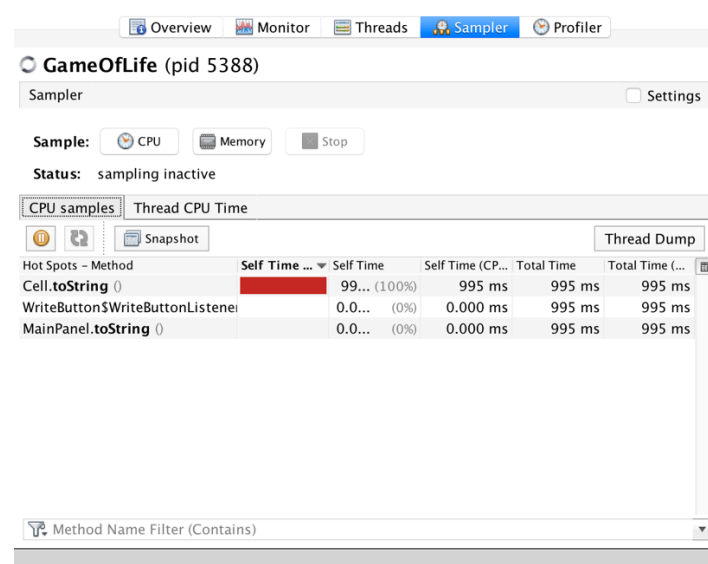
After:
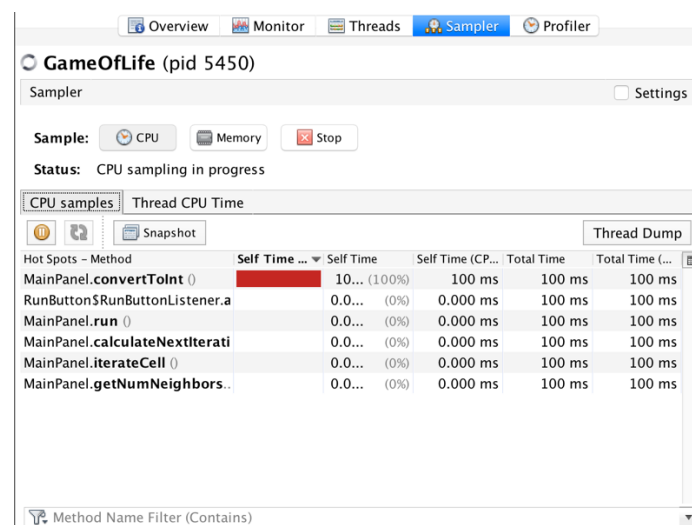
After clicking the cells to change their state, I click "Run Continuous" button, I found that MainPanel.runContinuous() and MainPanel.convertToInt() are two hot spot methods, and both of them have unnecessary loops which occupy the CPU. I first modified MainPanel.convertToInt() method. After testing the modification and comparing the performance of this method before and after modification, I ensured that the modification works. Then, I began to change MainPanel.runContinuous().
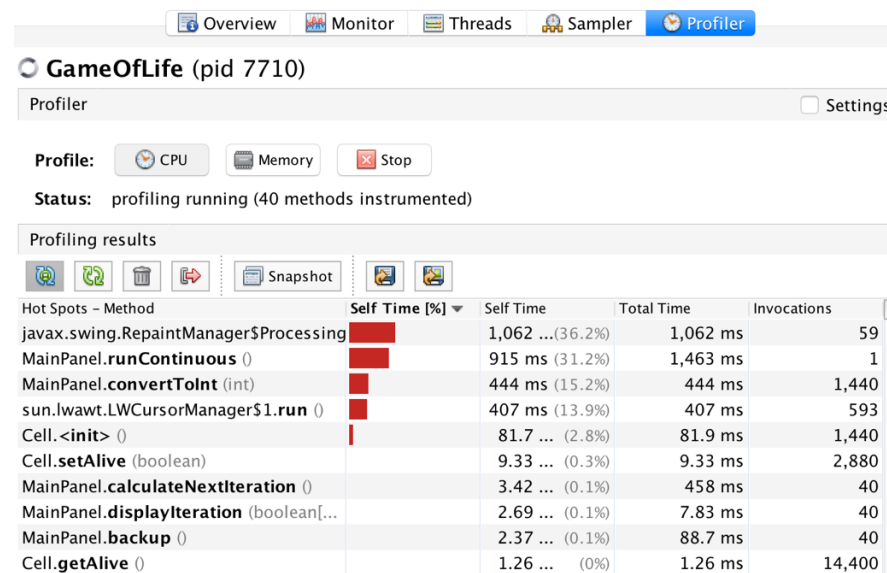
Mehthod Two: MainPanel.convertToInt()

The purpose of function convertToInt() is to covert the data type of input to int, however, the data type of input is restricted as int type. As this method is good for segmenting code and in case it will be useful in the future, so I just remove the unnecessary while loop instead of removing the whole method. After modification, MainPanel.convertToInt() is no longer in the Hot Spots-Method list.

Before:

| Overview | Monitor | Threads | Sampler | Profiler |

GameOfLife (pid 7710)

Profiler   ☐ Settings

Profile: ⏱ CPU   🎞 Memory   ❌ Stop

Status: profiling running (40 methods instrumented)

Profiling results

🔄 ⟳ 🗑 ⇨   📋 Snapshot   🖼 🖼

Hot Spots – Method	Self Time [%] ▼	Self Time	Total Time	Invocations
javax.swing.RepaintManager$Processing		1,062 ...(36.2%)	1,062 ms	59
MainPanel.**runContinuous** ()		915 ms (31.2%)	1,463 ms	1
MainPanel.**convertToInt** (int)		444 ms (15.2%)	444 ms	1,440
sun.lwawt.LWCursorManager$1.**run** ()		407 ms (13.9%)	407 ms	593
Cell.**<init>** ()		81.7 ... (2.8%)	81.9 ms	1,440
Cell.**setAlive** (boolean)		9.33 ... (0.3%)	9.33 ms	2,880
MainPanel.**calculateNextIteration** ()		3.42 ... (0.1%)	458 ms	40
MainPanel.**displayIteration** (boolean[...		2.69 ... (0.1%)	7.83 ms	40
MainPanel.**backup** ()		2.37 ... (0.1%)	88.7 ms	40
Cell.**getAlive** ()		1.26 ... (0%)	1.26 ms	14,400

After:

| Overview | Monitor | Threads | Sampler | Profiler |

GameOfLife (pid 7715)

Profiler   ☐ Settings

Profile: ⏱ CPU   🎞 Memory   ❌ Stop

Status: profiling running (40 methods instrumented)

Profiling results

🔄 ⟳ 🗑 ⇨   📋 Snapshot   🖼 🖼

Hot Spots – Method	Self Time [%] ▼	Self Time	Total Time	Invocations
MainPanel.**runContinuous** ()		1,776 ...(46.6%)	1,939 ms	1
javax.swing.RepaintManager$Processing		1,425 ...(37.4%)	1,425 ms	51
sun.lwawt.LWCursorManager$1.**run** ()		448 ms (11.7%)	448 ms	652
Cell.**<init>** ()		129 ms (3.4%)	130 ms	2,808
Cell.**setAlive** (boolean)		12.3 ... (0.3%)	12.3 ms	5,616
MainPanel.**backup** ()		5.96 ... (0.2%)	143 ms	78
MainPanel.**calculateNextIteration** ()		5.91 ... (0.2%)	19.8 ms	78
MainPanel.**displayIteration** (boolean[...		2.83 ... (0.1%)	8.60 ms	78
MainPanel.**getNumNeighbors** (int, int)		2.47 ... (0.1%)	4.22 ms	2,808
Cell.**getAlive** ()		2.23 ... (0.1%)	2.23 ms	28,080

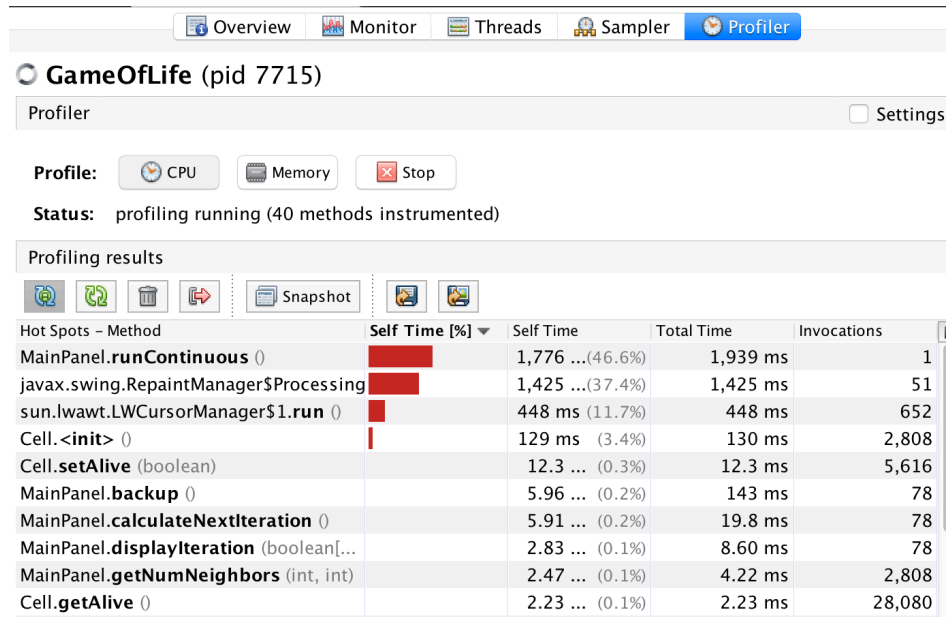## Method Three: MainPanel.runContinuous()

There is an unnecessary loop, as although the value of global parameter _r is changed during the loop, but it was changed back to its origin value after the loop, so I remove the for loop and try...catch which have thread.sleep() in it to improve the performance. After modification, MainPanel.runContinuous() is no longer in the Hot Spots-Method list.

As **MainPanel.runContinuous()** method **returns nothing**, I am unable to use Unit Test to test the outcome, so I use manual test to test the function is not changed.

Before:

| | Overview | Monitor | Threads | Sampler | Profiler |

GameOfLife (pid 7715)

Profiler ☐ Settings

Profile: CPU | Memory | Stop

Status: profiling running (40 methods instrumented)

Profiling results

Snapshot

Hot Spots – Method	Self Time [%] ▼	Self Time		Total Time	Invocations
MainPanel.**runContinuous** ()		1,776 ...	(46.6%)	1,939 ms	1
javax.swing.RepaintManager$Processing		1,425 ...	(37.4%)	1,425 ms	51
sun.lwawt.LWCursorManager$1.**run** ()		448 ms	(11.7%)	448 ms	652
Cell.**<init>** ()		129 ms	(3.4%)	130 ms	2,808
Cell.**setAlive** (boolean)		12.3 ...	(0.3%)	12.3 ms	5,616
MainPanel.**backup** ()		5.96 ...	(0.2%)	143 ms	78
MainPanel.**calculateNextIteration** ()		5.91 ...	(0.2%)	19.8 ms	78
MainPanel.**displayIteration** (boolean[...		2.83 ...	(0.1%)	8.60 ms	78
MainPanel.**getNumNeighbors** (int, int)		2.47 ...	(0.1%)	4.22 ms	2,808
Cell.**getAlive** ()		2.23 ...	(0.1%)	2.23 ms	28,080

After:

Start Page | GameOfLife (pid 2410)

| | Overview | Monitor | Threads | Sampler | Profiler |

GameOfLife (pid 2410)

Profiler ☐ Settings

Profile: CPU | Memory | Stop

Status: profiling running (40 methods instrumented)

Profiling results

Snapshot

Hot Spots – Method	Self Time [%] ▼	Self Time		Total Time	Invocations
javax.swing.RepaintManager$ProcessingR		1,613 ...	(47.6%)	1,613 ms	68
Cell.**<init>** ()		1,077 ...	(31.8%)	1,083 ms	108,675
sun.lwawt.LWCursorManager$1.**run** ()		409 ms	(12.1%)	409 ms	622
Cell.**setAlive** (boolean)		72.0 ms	(2.1%)	72.0 ms	217,350
Cell.**getAlive** ()		49.7 ms	(1.5%)	49.7 ms	1,086,750
MainPanel.**getNumNeighbors** (int, int)		47.2 ms	(1.4%)	90.4 ms	108,675
MainPanel.**backup** ()		42.4 ms	(1.3%)	1,179 ms	483
MainPanel.**displayIteration** (boolean[][])		25.5 ms	(0.8%)	49.7 ms	483
MainPanel.**calculateNextIteration** ()		20.8 ms	(0.6%)	178 ms	483
MainPanel.**iterateCell** (int, int)		13.1 ms	(0.4%)	108 ms	108,675
MainPanel.**runContinuous** ()		9.26 ms	(0.3%)	1,367 ms	1