Arction
Ultimate Data Visualization Solutions

JavaScript Charts Performance Comparison / Heatmap Charts

Test date September 2nd, 2021

Foreword

This is a performance test / comparison of JavaScript charts. The tests are focusing on **heat map charts**' performance in different scenarios – visualizing static data sets, refreshing data sets and real-time data sources. The charts selected in this test, are the major manufacturers who claim their charts to be ***high-performance oriented*** or ***the fastest***, and some open-source libraries. There are also other charts available, which are either end-of-life, not supported anymore, or simply don't work. They were excluded. We are confident we have selected all the fastest charts in the comparison, and if we didn't, inform us to get it added in this test.

The heat map charts in these tests should cover almost all application fields for heat map charts. To mention some examples, **spectrogram monitoring/analysis, geographical data sets visualization**.

Ultimate Data Visualization Solutions

Test procedure

The test project is published as open-source project in Github.

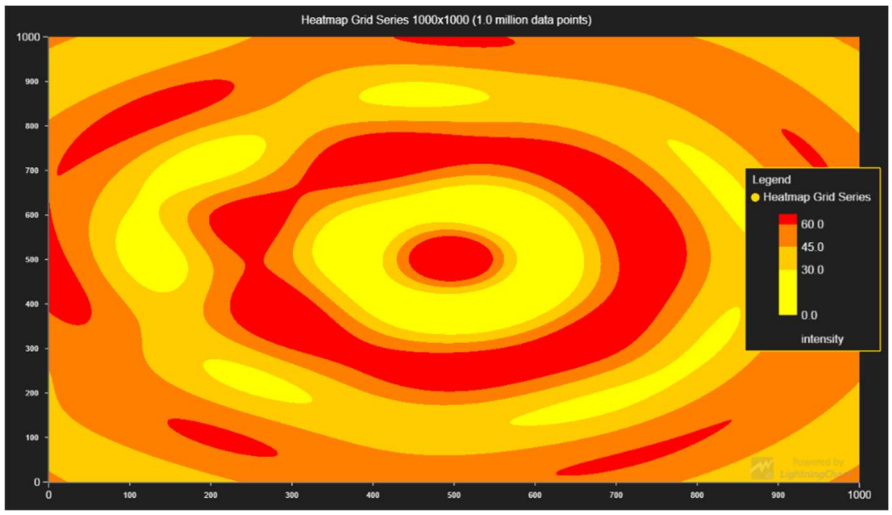
There are three different types of test scenarios, one for static heat map applications (data doesn't change during test) and two for real-time heat map applications (data changes during test).

For all the different test types, the tests are repeated with increasing stress levels until the application is no longer responsive. Stress level is defined as the size of heat map, which is equal to number of columns (values along X axis) times number of rows (values along Y axis). For example, 100x100 heat map will always have total of 10 000 data points in it.

The tests were made according to examples and tutorials. We strongly believe all of them have been made in optimal way for all of them, and if you notice any problem or wrong configuring of them, please contact us for a fix.

Static heat map tests

In this test, a data set is loaded and rendered as heat map based on a color look-up table.



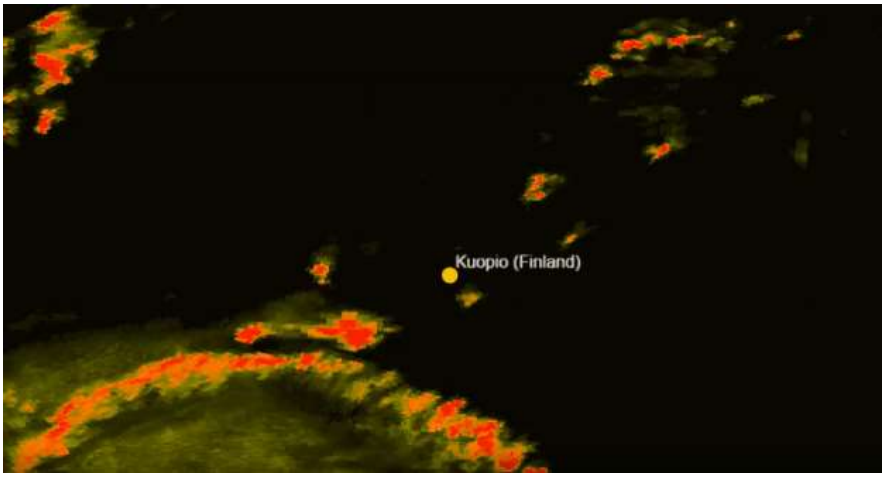
This test collects two performance measurements:

- Is the chart able to function? When the amount of data exceeds the capabilities of a particular chart, it can end up in browser crashing, running out of memory or simply hanging forever.

- Load-up speed. This is the time (seconds) from initiating the chart to when the heat map chart becomes fully visible. Lower values are better.

Refreshing heat map tests

In this test, a new data set is loaded for every displayed frame, meaning that one set of data is only displayed for as little time as possible. This is an extremely stressful test, which aims to find how fast the chart can refresh entire heat map data sets.



This test collects two performance measurements:

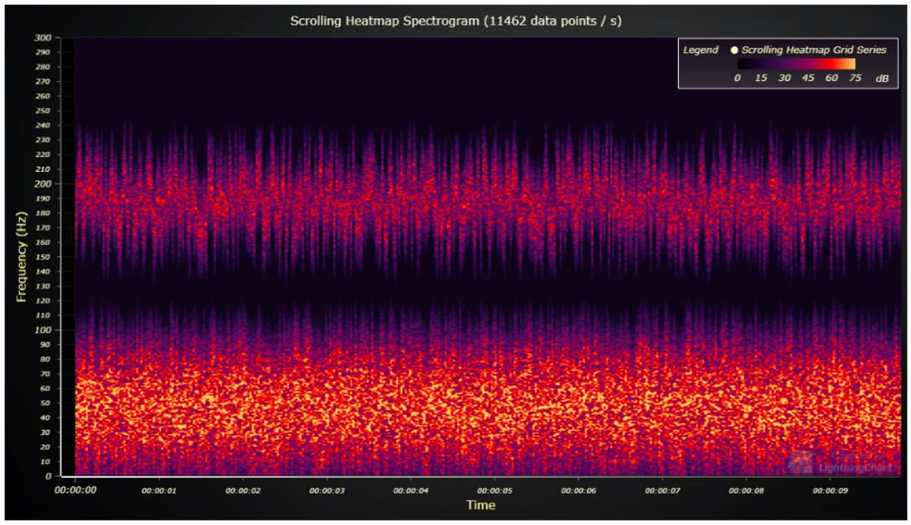
- Refresh rate or frames per second. How many times the data set is refreshed per second. Higher values are better.
- CPU usage. This is a % value between 0 and 100, which indicates how much processing power the chart is using. Lower values are better.

Appending heat map tests

In this test, a single data sample is added to the heat map after every displayed frame. This data sample contains a single value for each row of the heat map. For example, for a 500x500 heat map this means that every frame 500 data points will be added to the heat map. Adding data into the heat map will shift the oldest data out.

Arction

Ultimate Data Visualization Solutions



This test collects two performance measurements:

- Refresh rate or frames per second. How many times the chart is refreshed (redrawn) per second. Also tells the number of samples that are added every second, as they are added once per frame.
- CPU usage. This is a % value between 0 and 100, which indicates how much processing power the chart is using. Lower values are better.

Test hardware setups

The tests were carried out with the following fast desktop PC setup:

- CPU: AMD Ryzen 9 5900X processor
- GPU: Nvidia GeForce RTX 3080
- RAM 16 GB
- 2560 x 1440 resolution, 165 Hz

Google Chrome and Mozilla Firefox web browsers were used to collect the performance results. Only the best result between the two was written down.

Tested chart libraries (in no particular order)

- LightningChart® JS v.3.1
- LightningChart® JS v.3.0
- Highcharts v.9.1.0 (Boost module enabled)
- ECharts v.5
- ZingChart v.2.9.3
- SciChart JS v.1.4.1633

Results

The results were output to Google Chrome browser Console and imported into Excel.

The green color represents good value, yellow mediocre, and red… it indicates either slowness or bright red a failure.

The FPS rates are visualized with bars inside the cells, too.

Ultimate Data Visualization Solutions

Static heat map results

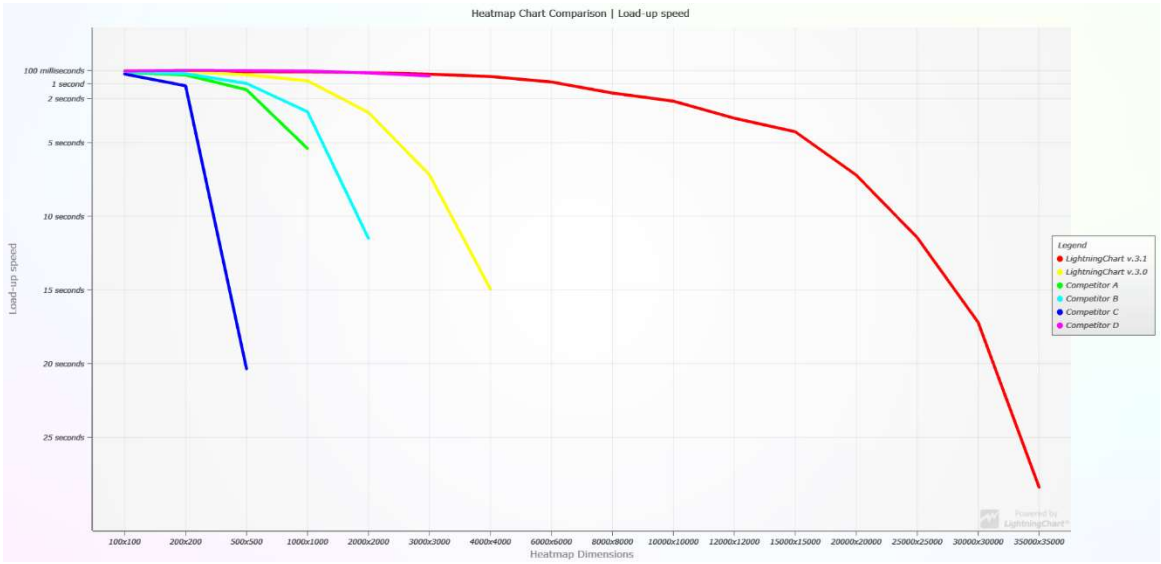
Page 1

Heat map size	Data points	JavaScript chart library	Initial rendering delay (ms)
50x50	2500	LightningChart v.3.1	150
		Competitor D	110
		LightningChart v.3.0	107
		Competitor A	146
		Competitor B	6
		Competitor C	108
100x100	10000	LightningChart v.3.1	155
		Competitor D	130
		LightningChart v.3.0	107
		Competitor A	220
		Competitor B	190
		Competitor C	340
200x200	40000	LightningChart v.3.1	170
		Competitor D	96
		LightningChart v.3.0	180
		Competitor A	405
		Competitor B	314
		Competitor C	1140
500x500	250000	LightningChart v.3.1	180
		Competitor D	110
		LightningChart v.3.0	391
		Competitor A	1405
		Competitor B	980
		Competitor C	20360
1000x1000	1000000	LightningChart v.3.1	190
		Competitor D	140
		LightningChart v.3.0	802
		Competitor A	5400
		Competitor B	2900
		Competitor C	FAIL
2000x2000	4000000	LightningChart v.3.1	250
		Competitor D	265
		LightningChart v.3.0	2950
		Competitor A	FAIL
		Competitor B	11500
		Competitor C	FAIL
3000x3000	9000000	LightningChart v.3.1	350
		Competitor D	465
		LightningChart v.3.0	7180
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL
4000x4000	16000000	LightningChart v.3.1	510
		Competitor D	FAIL
		LightningChart v.3.0	14950
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL

Arction
Ultimate Data Visualization Solutions

Heat map size	Data points	JavaScript chart library	Initial rendering delay (ms)
6000x6000	36000000	LightningChart v.3.1	880
		Competitor D	FAIL
		LightningChart v.3.0	FAIL
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL
8000x8000	64000000	LightningChart v.3.1	1630
		Competitor D	FAIL
		LightningChart v.3.0	FAIL
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL
10000x10000	100000000	LightningChart v.3.1	2180
		Competitor D	FAIL
		LightningChart v.3.0	FAIL
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL
12000x12000	144000000	LightningChart v.3.1	3340
		Competitor D	FAIL
		LightningChart v.3.0	FAIL
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL
15000x15000	225000000	LightningChart v.3.1	4260
		Competitor D	FAIL
		LightningChart v.3.0	FAIL
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL
20000x20000	400000000	LightningChart v.3.1	7200
		Competitor D	FAIL
		LightningChart v.3.0	FAIL
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL
25000x25000	625000000	LightningChart v.3.1	11440
		Competitor D	FAIL
		LightningChart v.3.0	FAIL
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL
30000x30000	900000000	LightningChart v.3.1	17200
		Competitor D	FAIL
		LightningChart v.3.0	FAIL
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL
35000x35000	1225000000	LightningChart v.3.1	28400
		Competitor D	FAIL
		LightningChart v.3.0	FAIL
		Competitor A	FAIL
		Competitor B	FAIL
		Competitor C	FAIL

Arction
Ultimate Data Visualization Solutions



Visualization of static heat map benchmark results. X axis = heat map data set size, Y axis = load-up speed (higher is better). Range along X axis indicates the maximum size of heat map with the web chart.

Arction
Ultimate Data Visualization Solutions

Refreshing heat map results

Page 1

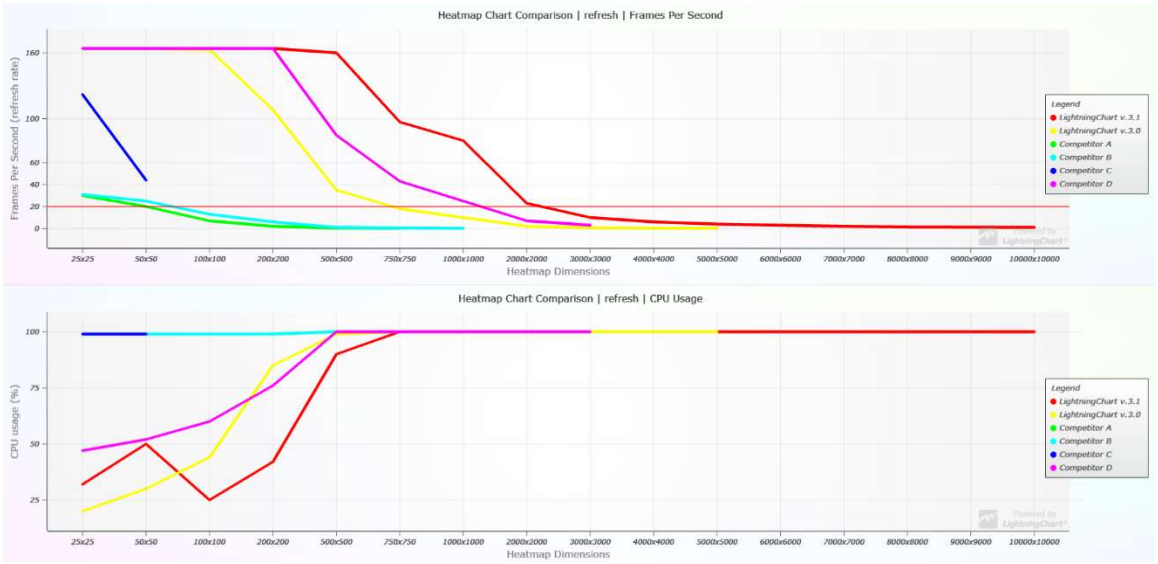
Heat map size	JavaScript chart library	FPS	CPU usage
25x25	**LightningChart v.3.1**	164	32
25x25	Competitor D	164	47
25x25	LightningChart v.3.0	164	20
25x25	Competitor A	30	99
25x25	Competitor B	31	99
25x25	Competitor C	122	99
50x50	**LightningChart v.3.1**	164	50
50x50	Competitor D	164	52
50x50	LightningChart v.3.0	164	30
50x50	Competitor A	20	99
50x50	Competitor B	25	99
50x50	Competitor C	44	99
100x100	**LightningChart v.3.1**	164	25
100x100	Competitor D	164	60
100x100	LightningChart v.3.0	163	44
100x100	Competitor A	7	99
100x100	Competitor B	13	99
100x100	Competitor C	FAIL	FAIL
200x200	**LightningChart v.3.1**	164	42
200x200	Competitor D	164	76
200x200	LightningChart v.3.0	108	85
200x200	Competitor A	2	99
200x200	Competitor B	6	99
200x200	Competitor C	FAIL	FAIL
500x500	**LightningChart v.3.1**	160	90
500x500	Competitor D	85	100
500x500	LightningChart v.3.0	35	99
500x500	Competitor A	0.4	100
500x500	Competitor B	1	100
500x500	Competitor C	FAIL	FAIL
750x750	**LightningChart v.3.1**	97	100
750x750	Competitor D	43	100
750x750	LightningChart v.3.0	18	100
750x750	Competitor A	0.2	100
750x750	Competitor B	0.5	100
750x750	Competitor C	FAIL	FAIL
1000x1000	**LightningChart v.3.1**	80	100
1000x1000	Competitor D	25	100
1000x1000	LightningChart v.3.0	10	100
1000x1000	Competitor A	FAIL	FAIL
1000x1000	Competitor B	0.2	100
1000x1000	Competitor C	FAIL	FAIL
2000x2000	**LightningChart v.3.1**	23	100
2000x2000	Competitor D	7	100
2000x2000	LightningChart v.3.0	2	100
2000x2000	Competitor A	FAIL	FAIL
2000x2000	Competitor B	FAIL	FAIL
2000x2000	Competitor C	FAIL	FAIL

Arction
Ultimate Data Visualization Solutions

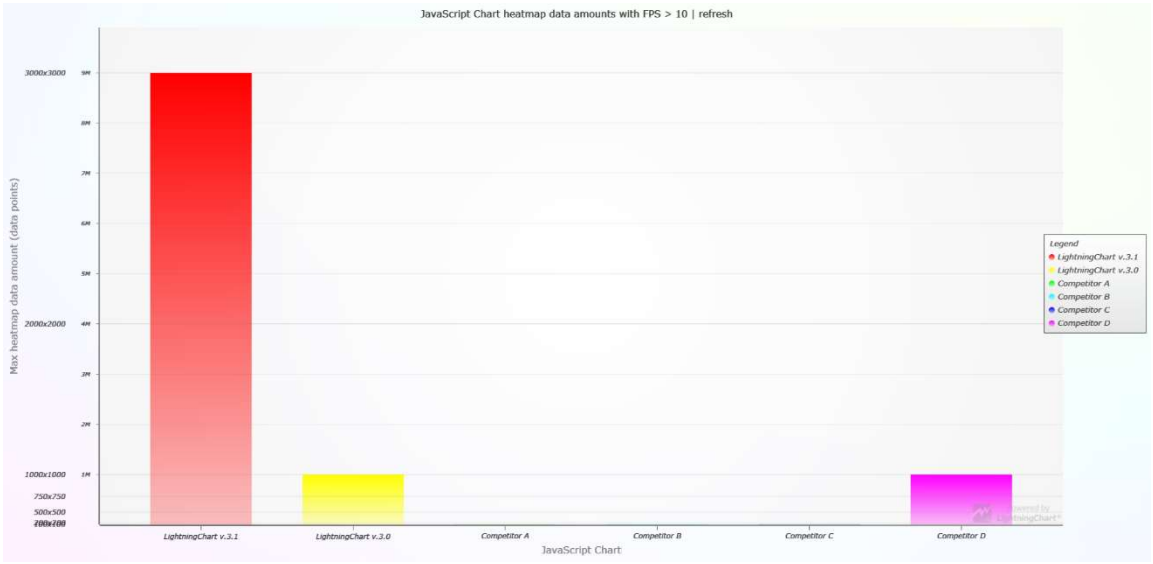
Heat map size	JavaScript chart library	FPS	CPU usage
3000x3000	**LightningChart v.3.1**	10	100
3000x3000	Competitor D	3	100
3000x3000	LightningChart v.3.0	0.7	100
3000x3000	Competitor A	FAIL	FAIL
3000x3000	Competitor B	FAIL	FAIL
3000x3000	Competitor C	FAIL	FAIL
4000x4000	**LightningChart v.3.1**	6	100
4000x4000	Competitor D	FAIL	FAIL
4000x4000	LightningChart v.3.0	0.3	100
4000x4000	Competitor A	FAIL	FAIL
4000x4000	Competitor B	FAIL	FAIL
4000x4000	Competitor C	FAIL	FAIL
5000x5000	**LightningChart v.3.1**	4	100
5000x5000	Competitor D	FAIL	FAIL
5000x5000	LightningChart v.3.0	0.2	100
5000x5000	Competitor A	FAIL	FAIL
5000x5000	Competitor B	FAIL	FAIL
5000x5000	Competitor C	FAIL	FAIL
6000x6000	**LightningChart v.3.1**	3	100
6000x6000	Competitor D	FAIL	FAIL
6000x6000	LightningChart v.3.0	FAIL	FAIL
6000x6000	Competitor A	FAIL	FAIL
6000x6000	Competitor B	FAIL	FAIL
6000x6000	Competitor C	FAIL	FAIL
7000x7000	**LightningChart v.3.1**	2	100
7000x7000	Competitor D	FAIL	FAIL
7000x7000	LightningChart v.3.0	FAIL	FAIL
7000x7000	Competitor A	FAIL	FAIL
7000x7000	Competitor B	FAIL	FAIL
7000x7000	Competitor C	FAIL	FAIL
8000x8000	**LightningChart v.3.1**	1.4	100
8000x8000	Competitor D	FAIL	FAIL
8000x8000	LightningChart v.3.0	FAIL	FAIL
8000x8000	Competitor A	FAIL	FAIL
8000x8000	Competitor B	FAIL	FAIL
8000x8000	Competitor C	FAIL	FAIL
9000x9000	**LightningChart v.3.1**	1.2	100
9000x9000	Competitor D	FAIL	FAIL
9000x9000	LightningChart v.3.0	FAIL	FAIL
9000x9000	Competitor A	FAIL	FAIL
9000x9000	Competitor B	FAIL	FAIL
9000x9000	Competitor C	FAIL	FAIL
10000x10000	**LightningChart v.3.1**	1	100
10000x10000	Competitor D	FAIL	FAIL
10000x10000	LightningChart v.3.0	FAIL	FAIL
10000x10000	Competitor A	FAIL	FAIL
10000x10000	Competitor B	FAIL	FAIL
10000x10000	Competitor C	FAIL	FAIL

Arction

Ultimate Data Visualization Solutions



Visualization of refreshing heat map benchmark results. X axis = heat map data set size, Upper Y axis = Refresh rate (higher value is better), Lower Y axis = CPU usage (lower value is better).



Visualization of refreshing heat map benchmark results. Bar height indicates the maximum heat map size where the library functioned with satisfactory performance (refresh rate > 10 / second).

Ultimate Data Visualization Solutions

Appending heat map results

Page 1

Heat map size	JavaScript chart library	FPS	CPU usage (%)
25x25	LightningChart v.3.1	165	35
25x25	Competitor D	164	45
25x25	LightningChart v.3.0	159	17
25x25	Competitor A	28	98
25x25	Competitor B	27	97
25x25	Competitor C	119	100
50x50	LightningChart v.3.1	165	55
50x50	Competitor D	164	53
50x50	LightningChart v.3.0	161	25
50x50	Competitor A	19	98
50x50	Competitor B	23	97
50x50	Competitor C	43	100
100x100	LightningChart v.3.1	165	27
100x100	Competitor D	164	59
100x100	LightningChart v.3.0	163	20
100x100	Competitor A	7	99
100x100	Competitor B	14	98
100x100	Competitor C	FAIL	FAIL
200x200	LightningChart v.3.1	164	34
200x200	Competitor D	164	82
200x200	LightningChart v.3.0	163	35
200x200	Competitor A	2	100
200x200	Competitor B	5	96
200x200	Competitor C	FAIL	FAIL
500x500	LightningChart v.3.1	164	50
500x500	Competitor D	85	100
500x500	LightningChart v.3.0	138	98
500x500	Competitor A	0.4	100
500x500	Competitor B	1	100
500x500	Competitor C	FAIL	FAIL
750x750	LightningChart v.3.1	165	62
750x750	Competitor D	44	100
750x750	LightningChart v.3.0	57	100
750x750	Competitor A	FAIL	FAIL
750x750	Competitor B	0.5	100
750x750	Competitor C	FAIL	FAIL
1000x1000	LightningChart v.3.1	165	25
1000x1000	Competitor D	25	100
1000x1000	LightningChart v.3.0	35	100
1000x1000	Competitor A	FAIL	FAIL
1000x1000	Competitor B	FAIL	FAIL
1000x1000	Competitor C	FAIL	FAIL
2000x2000	LightningChart v.3.1	165	30
2000x2000	Competitor D	7	100
2000x2000	LightningChart v.3.0	5	100
2000x2000	Competitor A	FAIL	FAIL
2000x2000	Competitor B	FAIL	FAIL
2000x2000	Competitor C	FAIL	FAIL

Arction
Ultimate Data Visualization Solutions

Heat map size	JavaScript chart library	FPS	CPU usage (%)
3000x3000	LightningChart v.3.1	165	40
3000x3000	Competitor D	3	100
3000x3000	LightningChart v.3.0	2	100
3000x3000	Competitor A	FAIL	FAIL
3000x3000	Competitor B	FAIL	FAIL
3000x3000	Competitor C	FAIL	FAIL
4000x4000	LightningChart v.3.1	163	42
4000x4000	Competitor D	FAIL	FAIL
4000x4000	LightningChart v.3.0	0.6	100
4000x4000	Competitor A	FAIL	FAIL
4000x4000	Competitor B	FAIL	FAIL
4000x4000	Competitor C	FAIL	FAIL
5000x5000	LightningChart v.3.1	163	53
5000x5000	Competitor D	FAIL	FAIL
5000x5000	LightningChart v.3.0	0.3	100
5000x5000	Competitor A	FAIL	FAIL
5000x5000	Competitor B	FAIL	FAIL
5000x5000	Competitor C	FAIL	FAIL
6000x6000	LightningChart v.3.1	163	59
6000x6000	Competitor D	FAIL	FAIL
6000x6000	LightningChart v.3.0	FAIL	FAIL
6000x6000	Competitor A	FAIL	FAIL
6000x6000	Competitor B	FAIL	FAIL
6000x6000	Competitor C	FAIL	FAIL
7000x7000	LightningChart v.3.1	163	61
7000x7000	Competitor D	FAIL	FAIL
7000x7000	LightningChart v.3.0	FAIL	FAIL
7000x7000	Competitor A	FAIL	FAIL
7000x7000	Competitor B	FAIL	FAIL
7000x7000	Competitor C	FAIL	FAIL
8000x8000	LightningChart v.3.1	162	73
8000x8000	Competitor D	FAIL	FAIL
8000x8000	LightningChart v.3.0	FAIL	FAIL
8000x8000	Competitor A	FAIL	FAIL
8000x8000	Competitor B	FAIL	FAIL
8000x8000	Competitor C	FAIL	FAIL
9000x9000	LightningChart v.3.1	161	78
9000x9000	Competitor D	FAIL	FAIL
9000x9000	LightningChart v.3.0	FAIL	FAIL
9000x9000	Competitor A	FAIL	FAIL
9000x9000	Competitor B	FAIL	FAIL
9000x9000	Competitor C	FAIL	FAIL
10000x10000	LightningChart v.3.1	164	30
10000x10000	Competitor D	FAIL	FAIL
10000x10000	LightningChart v.3.0	FAIL	FAIL
10000x10000	Competitor A	FAIL	FAIL
10000x10000	Competitor B	FAIL	FAIL
10000x10000	Competitor C	FAIL	FAIL

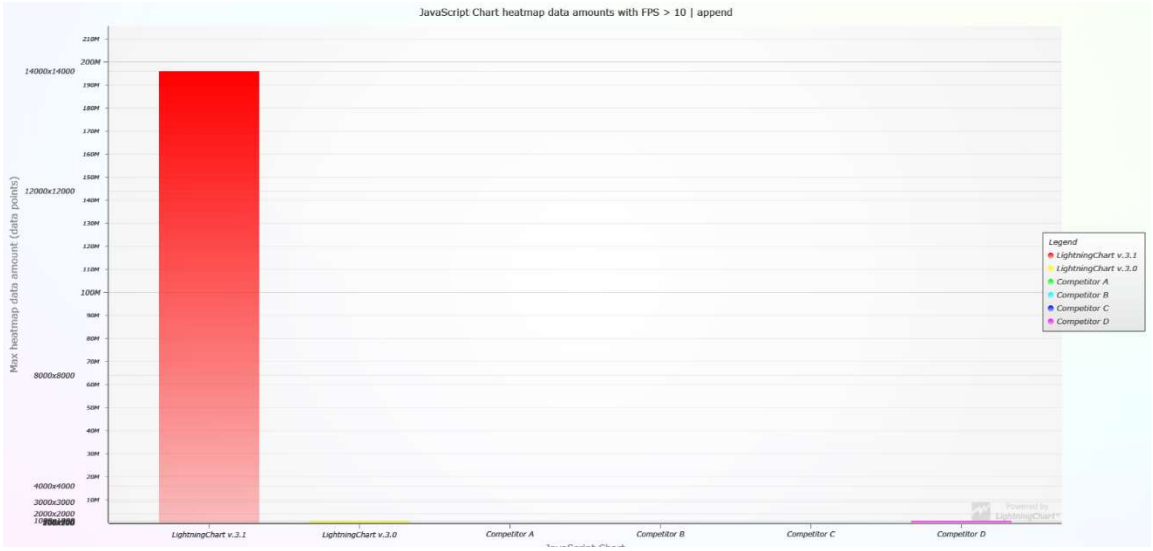
Arction
Ultimate Data Visualization Solutions

Heat map size	JavaScript chart library	FPS	CPU usage (%)
11000x11000	LightningChart v.3.1	145	32
11000x11000	Competitor D	FAIL	FAIL
11000x11000	LightningChart v.3.0	FAIL	FAIL
11000x11000	Competitor A	FAIL	FAIL
11000x11000	Competitor B	FAIL	FAIL
11000x11000	Competitor C	FAIL	FAIL
12000x12000	LightningChart v.3.1	125	38
12000x12000	Competitor D	FAIL	FAIL
12000x12000	LightningChart v.3.0	FAIL	FAIL
12000x12000	Competitor A	FAIL	FAIL
12000x12000	Competitor B	FAIL	FAIL
12000x12000	Competitor C	FAIL	FAIL
13000x13000	LightningChart v.3.1	120	44
13000x13000	Competitor D	FAIL	FAIL
13000x13000	LightningChart v.3.0	FAIL	FAIL
13000x13000	Competitor A	FAIL	FAIL
13000x13000	Competitor B	FAIL	FAIL
13000x13000	Competitor C	FAIL	FAIL
14000x14000	LightningChart v.3.1	115	46
14000x14000	Competitor D	FAIL	FAIL
14000x14000	LightningChart v.3.0	FAIL	FAIL
14000x14000	Competitor A	FAIL	FAIL
14000x14000	Competitor B	FAIL	FAIL
14000x14000	Competitor C	FAIL	FAIL

Arction

Ultimate Data Visualization Solutions

Visualization of appending heat map benchmark results. X axis = heat map data set size, Upper Y axis = Refresh rate (higher value is better), Lower Y axis = CPU usage (lower value is better).



Visualization of appending heat map benchmark results. Bar height indicates the maximum heat map size where the library functioned with satisfactory performance (refresh rate > 10 / second).

Ultimate Data Visualization Solutions

Conclusion – which is the fastest JavaScript chart?

Different chart charting libraries have their own strengths and selling points, but LightningChart's strength definitely is the exceptional rendering performance, allowing to build very advanced and data-intensive applications, visualizing heat map charts in just about any type of application.

Static heat map charts

With small heat map sizes (< 10000 total data samples) there is little variation in load-up speed. However, after 500x500 threshold is passed the difference in hardware accelerated libraries speed is clear, being **ready on average ~10x faster** than the other JavaScript chart libraries.

In **5 second**s, LightningChart JS v.3.1 can display a 13000x13000 heat map with a whopping 170 million data points. This is **395 times more data** than the average competitor without hardware acceleration, and **20** times more data than the closest hardware accelerated competitor.

The available heat map dimensions range is crucial in real-life applications, because if it is not enough it means you have to down-sample your data which results in precision loss! This can be a showstopper in many cases.

Real-time heat map charts

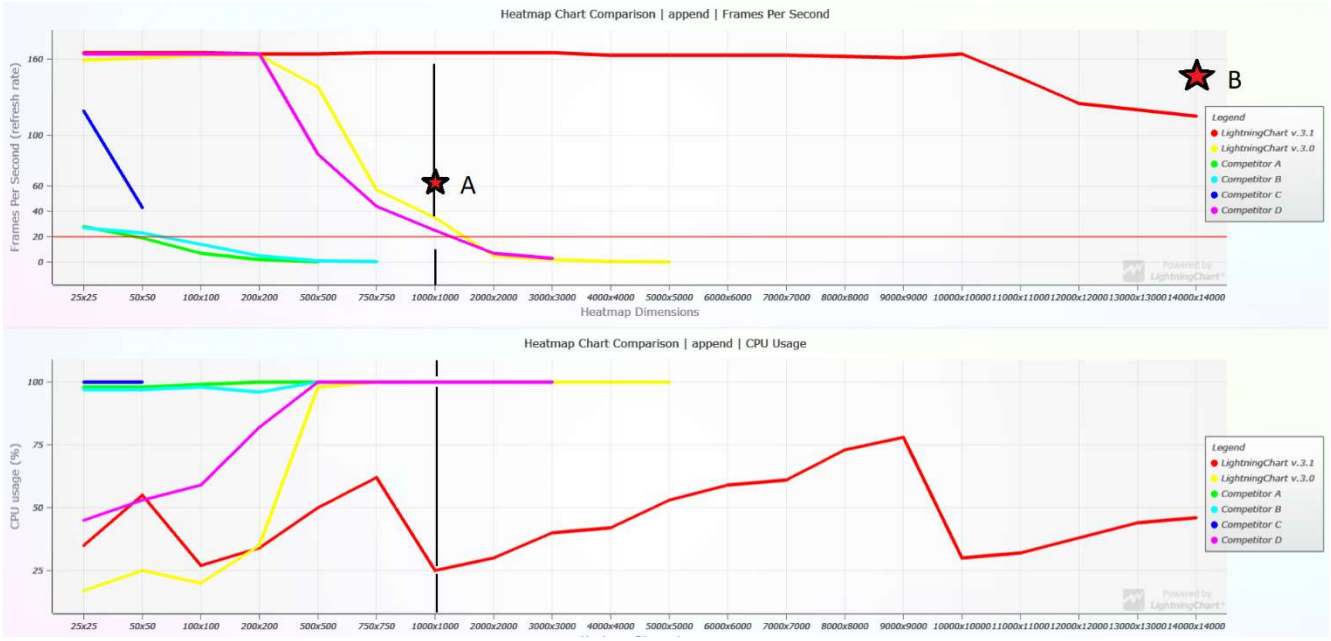
With refreshing and appending heat map applications charts without hardware acceleration can't function when heat map size exceeds even as little as 50x50; These libraries FPS quickly plummets to 5-15 range and even with 25x25 heat map (125 total data samples) they use 100% of available CPU power - **this kind of chart is unusable in an interactive web document**.

In refreshing stress tests, it becomes apparent that even other hardware accelerated charts can't keep up, performing on average **~3 times slower** and finally crashing as the heat map size exceeds 3000x3000. While at this level, the amount of data changes is simply too large to manage in a web browser, Lightning Chart JS persists all the way to 10000x10000 heat maps (100 million data points!!) and further without crashing. To put it into perspective, in this case the chart is receiving data at a rate equal to **100 000 data sources with 1 kHz sample rate.**

However, what really makes LightningChart JS shine above the other charts is the appending stress test. From the results, we can see that LightningChart JS is able to take performance advantage of the fact that with appending heat maps the entire previous data set does not need to be updated - **while the closest hardware accelerated competitor drops at around 1000x1000**, **LightningChart JS is seemingly unaffected** by the increasing data amounts and still performs with a whopping **115 FPS with a 14000x14000** heat map (196 million visible data points!!). Furthermore, the CPU usage at this point is still only at 46% which is less than what competitors require with 25x25 heat maps.

The sheer difference in performance in this test scenario is insane; if we linearly compare data amount and FPS, LightningChart JS is **13800x more efficient** than the average chart without hardware acceleration, and **835x** more efficient than the closest hardware accelerated competitor.

The following visualization can give us an insight on how significant this performance difference is.



Visualization of appending heat map benchmark results. X axis = heat map data set size, Upper Y axis = Refresh rate (higher value is better), Lower Y axis = CPU usage (lower value is better).

The highlighted point A is where every other tool in the field has stopped performing responsively. In the bottom chart you can see the CPU usage of each chart at this test – LightningChart JS v.3.1 only requires 25% of CPU to beat any other competitor.

Furthermore, LightningChart JS continues smoothly operating seemingly unaffected by the increasingly stressful tests. The data set size at point A is 1 million and point B 196 million – even at this stage, LightningChart JS still uses less than 50% of available CPU processing power.

About LightningChart® JS

LightningChart® is registered trademark by Arction Ltd, a pioneer in high-performance charting, who introduced the fastest, GPU accelerated charts, already in 2009 for Microsoft .NET technologies. Before that, and ever since, the LightningChart® team has studied different technologies, prototyped, researched,

innovated new algorithms, which are now part of LightningChart® product lines, to produce the absolute best performance for those advanced applications that really need it. LightningChart® JS product line was released in 2019 and development is full-time by a large team. LightningChart® team is ready to help!