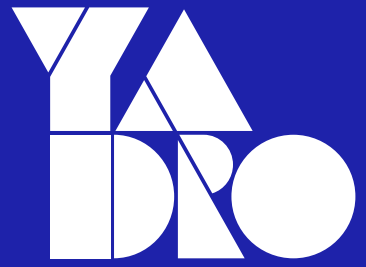
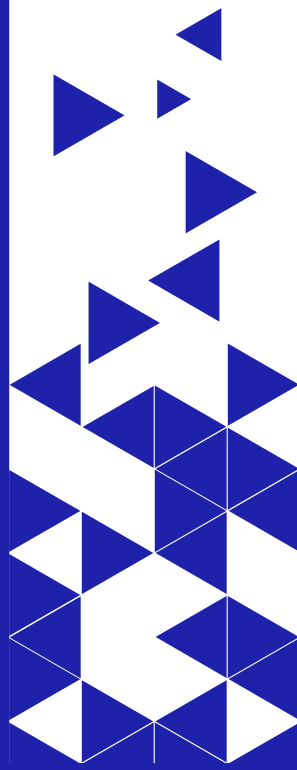


Боремся с **90***-секундным зависанием* **Chrome** *при рендеринге графиков на* **50***к точек каждый*



Обо мне

Проблема длиной в 90 секунд

Замеры перфоманса

Оптимизация рендера

А что случилось?

Миша, все плохо, давай по новой!

Фантастические библиотеки и где они обитают

Соколов Андрей

Эксперт по разработке ПО, Yadro

- Full-stack developer
  (TS, React, node.js … java, asp, php)
- Telegram: @AndSklv

Обо мне

Проблема длиной в 90 секунд

Замеры перфоманса

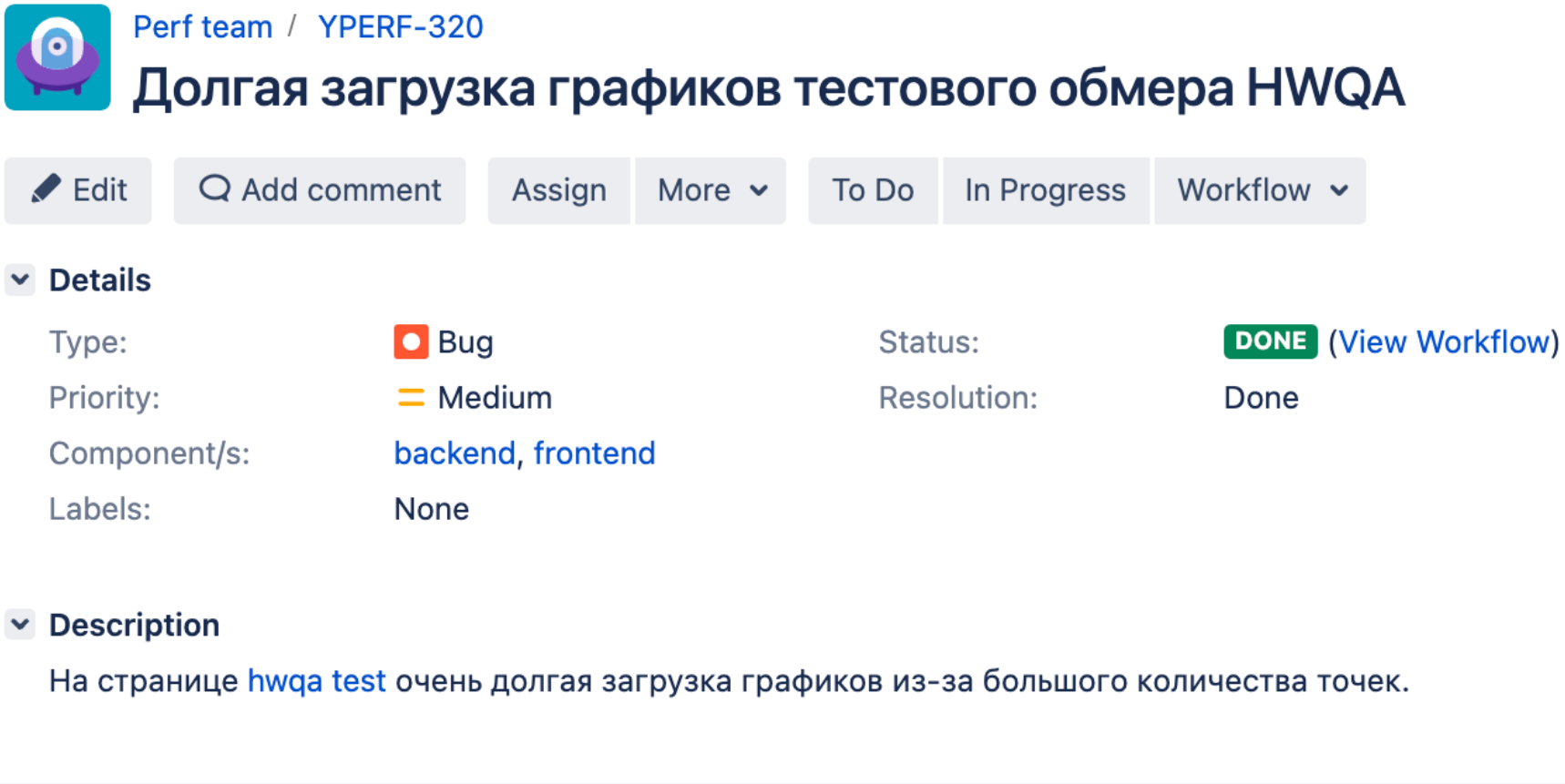
Оптимизация рендера

А что случилось?

Миша, все плохо, давай по новой!

Фантастические библиотеки и где они обитают

Задача



Perf team / YPERF-320

Долгая загрузка графиков тестового обмера HWQA

Edit | Add comment | Assign | More | To Do | In Progress | Workflow

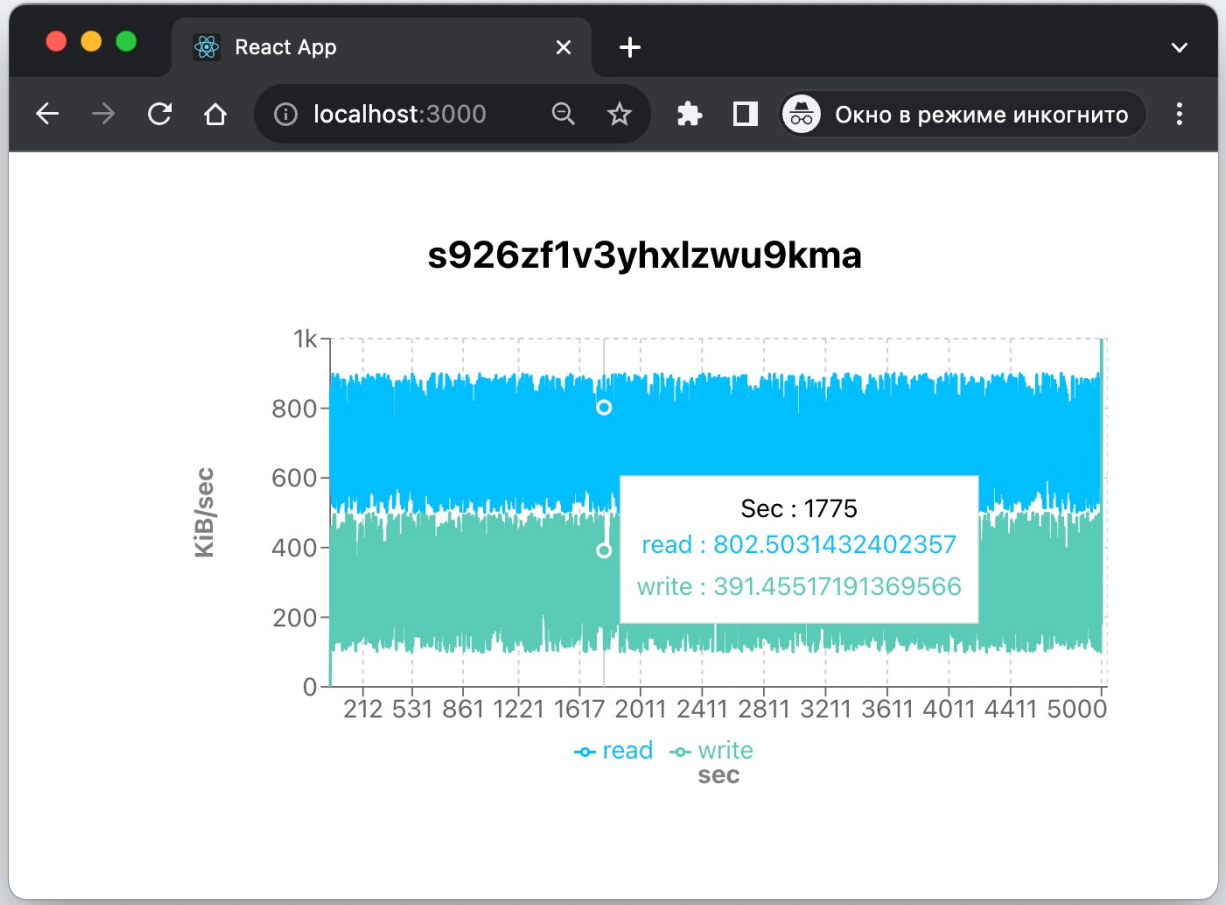
Details

Type:	Bug	Status:	DONE (View Workflow)
Priority:	Medium	Resolution:	Done
Component/s:	backend, frontend		
Labels:	None		

Description

На странице hwqa test очень долгая загрузка графиков из-за большого количества точек.

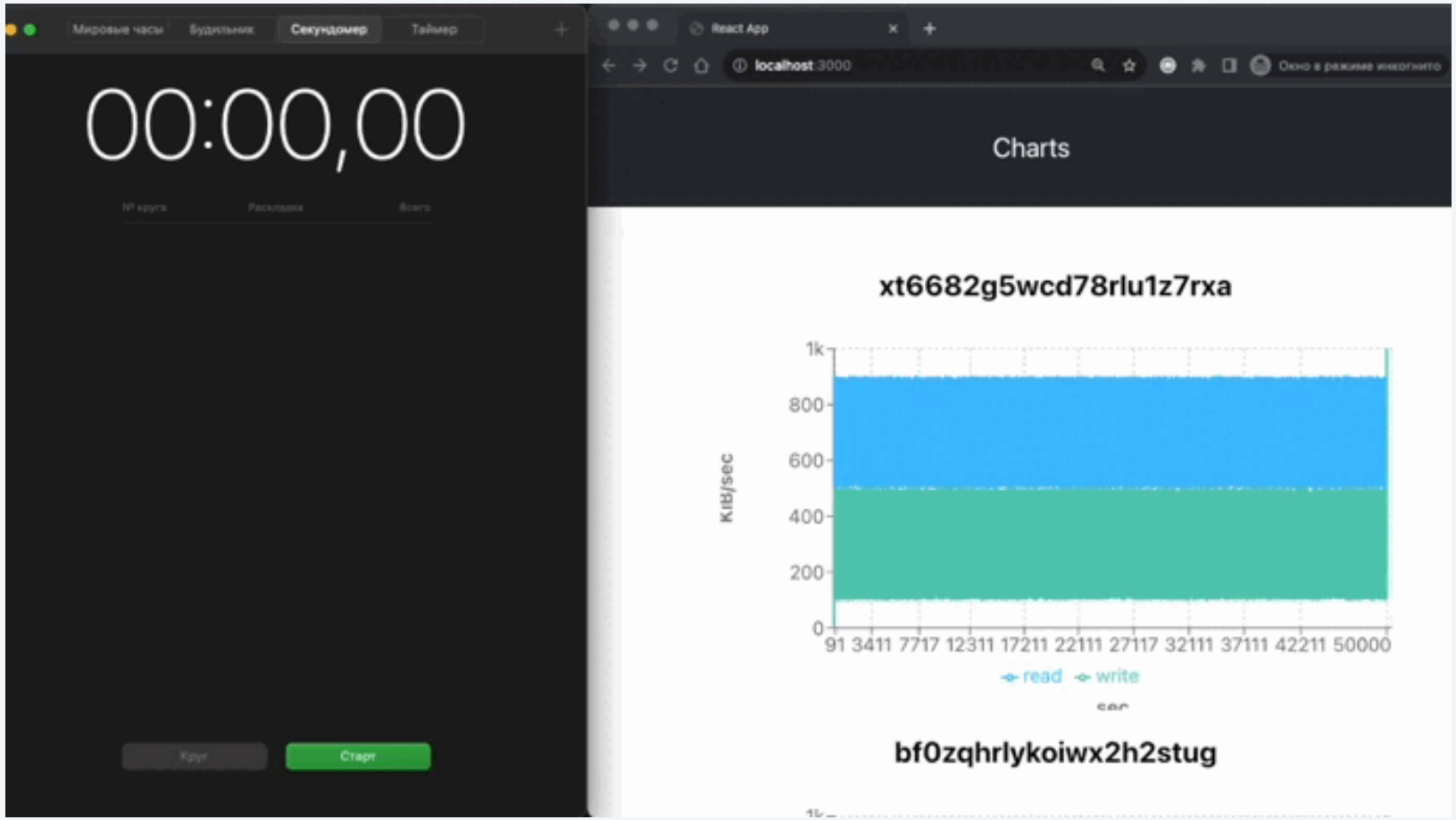
Как выглядит график



Как выглядит код

```
<ResponsiveContainer width='100%' height={320}>
  <LineChart data={data} margin={{ top: 20, bottom: 30, left: 34 }}>
    <CartesianGrid strokeDasharray='3 3' />
    <XAxis dataKey='sec' padding={{ left: 0, right: 4 }}>
      <Label
        value={labelX} position='bottom'
        offset={20} style={{ textAnchor: 'middle'}} />
    </XAxis>
    <YAxis type='number' tickFormatter={(value) => kFormatter(value)}>
      <Label
        value={labelY} position='insideLeft'
        angle={-90} offset={-20} style={{ textAnchor: 'middle'}} />
    </YAxis>
    <Tooltip labelFormatter={(value) => `Sec : ${value}`} />
    <Legend />
    {availableValues.map((field, index) => (
      <Line
        key={index} type='monotone' dataKey={field}
        stroke={generateColor(field)} dot={false} strokeWidth={2}/>
    ))}
  </LineChart>
</ResponsiveContainer>
```

Проблема


Мировые часы
Будильник
Секундомер
Таймер
00:00,00
Круг
Старт
React App
localhost:3000
Charts
xt6682g5wcd78rlu1z7rxa
KiB/sec
1k
800
600
400
200
0
91 3411 7717 12311 17211 22111 27117 32111 37111 42211 50000
read
write
bf0zqhrlykoiwx2h2stug

Обо мне

Проблема длиной в 90 секунд

Замеры перфоманса

Оптимизация рендера

А что случилось?

Миша, все плохо, давай по новой!

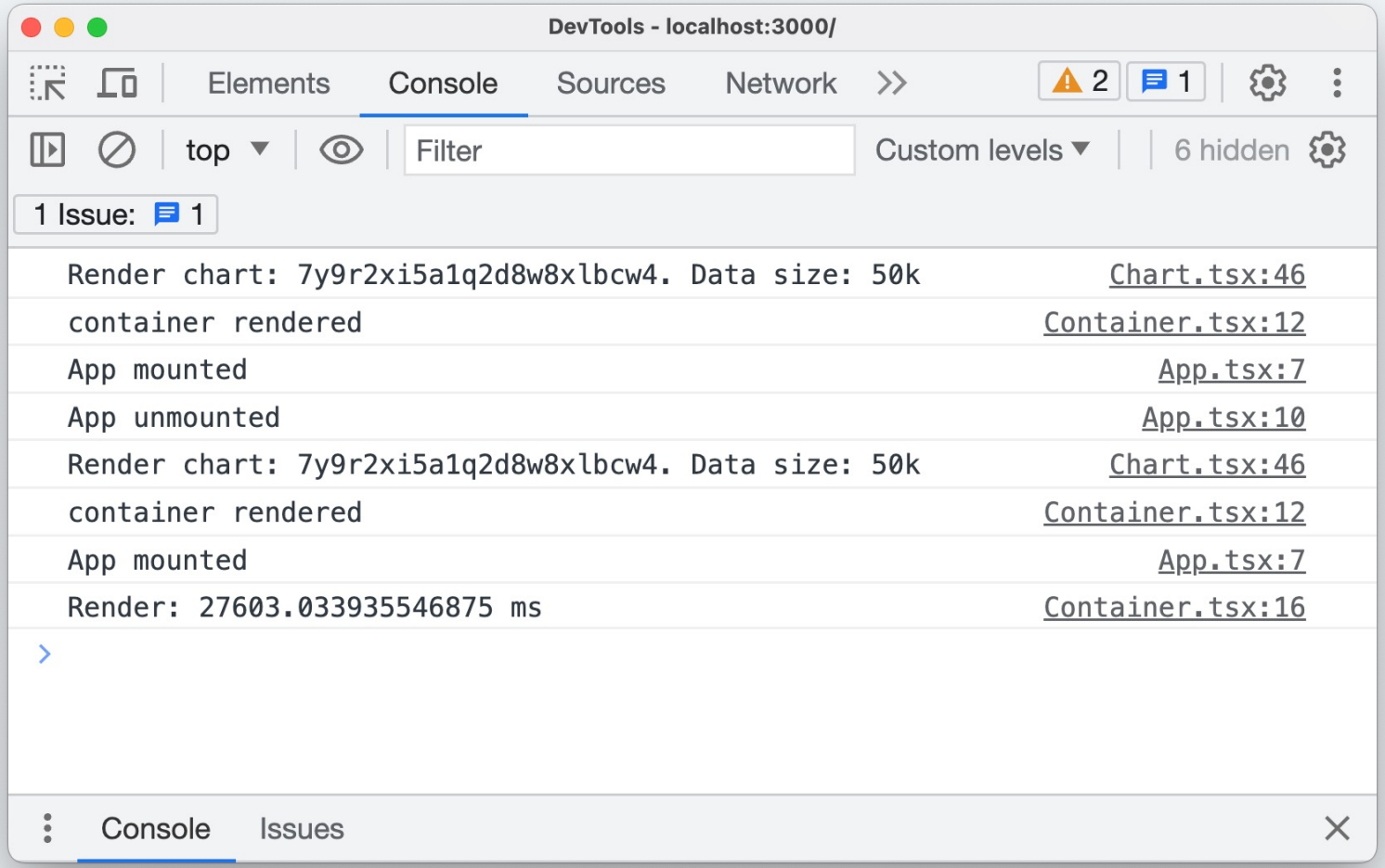
Фантастические библиотеки и где они обитают

Код

charts – Chart.tsx

```
useEffect(() =>
  console.log( {
    `Render chart: ${title}. Data size: ${Math.round(
      data.length / 1000
    )}k`
  );

  const timerId = setInterval(() =>
    const line = document.querySelector('.recharts-line-curve');
    if  line) {
      clearInterval(timerId);
      onEnd();
    }
  }, 250);

  return () =>
    clearInterval(timerId);
  }
}, []);
```

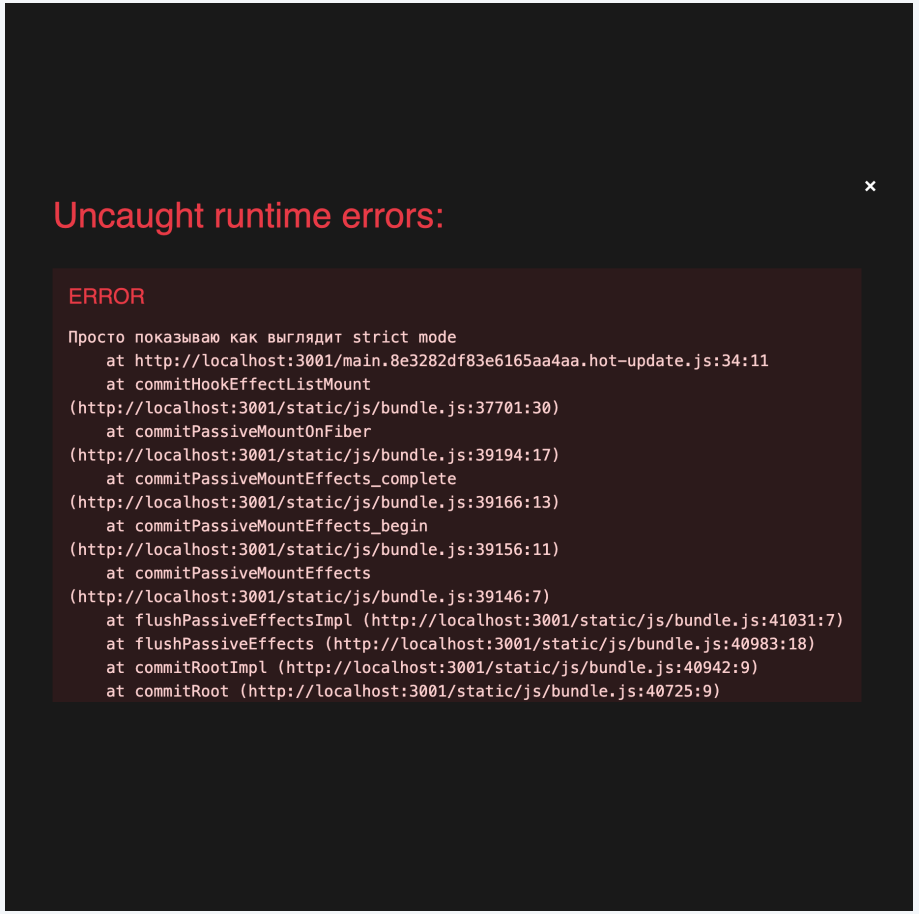
Первый результат



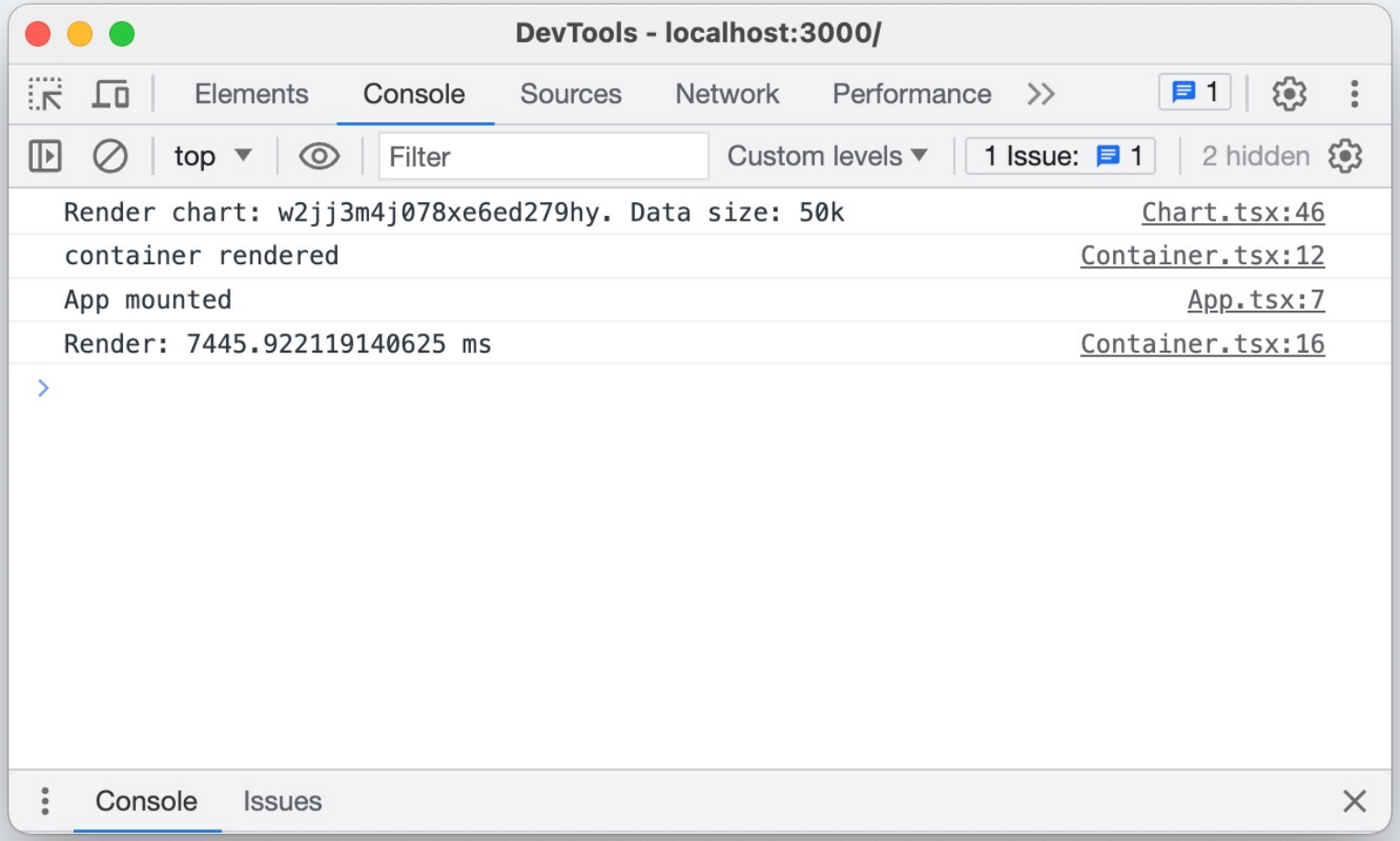
React.StrictMode

```
1  root.render(
2    <React.StrictMode>
3      <App />
4    </React.StrictMode>
5  );
```

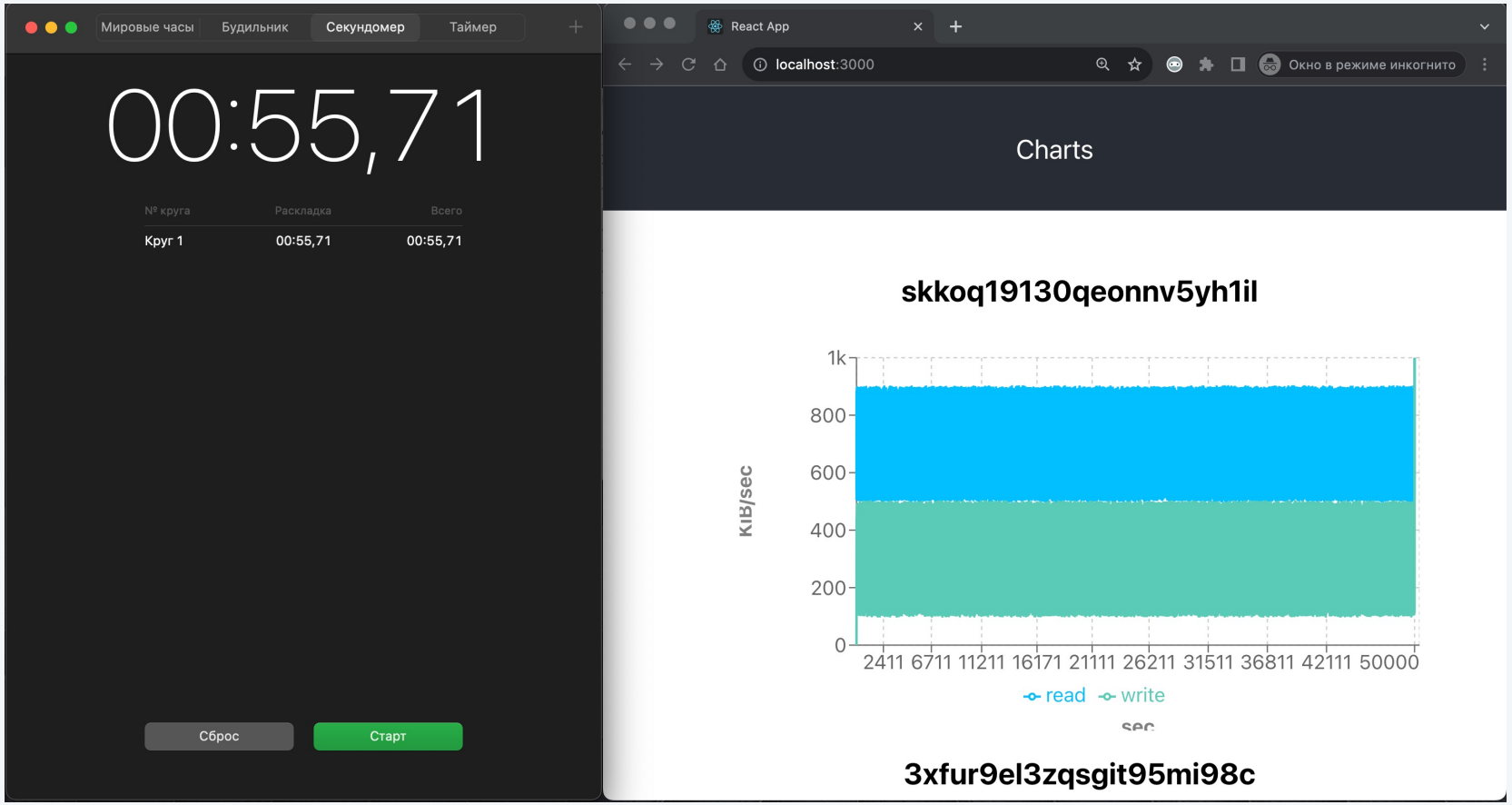
- Инструмент для выявления потенциальных проблем в приложениях React
- Рендерит дважды
- Strict Mode не влияет на продакшн-сборку вашего приложения



Базовое значение



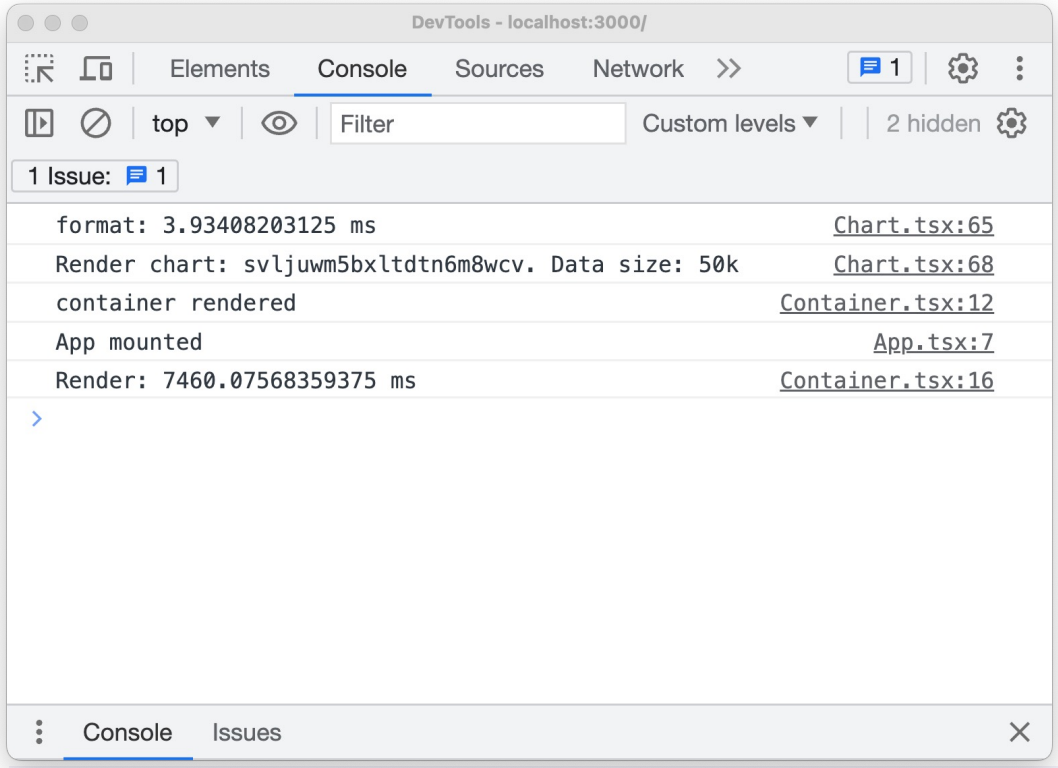
Рендер


Мировые часы
Будильник
Секундомер
Таймер
00:55,71
№ круга
Раскладка
Всего
Круг 1
00:55,71
00:55,71
Сброс
Старт
React App
localhost:3000
Окно в режиме инкогнито
Charts
skkoq19130qeonnv5yh1il
1k
800
600
400
200
0
KiB/sec
2411 6711 11211 16171 21111 26211 31511 36811 42111 50000
read
write
sec
3xfur9el3zqsgit95mi98c

ЗАМЕРЫ ПЕРФОМАНСА

Оптимизация данных

- 0%

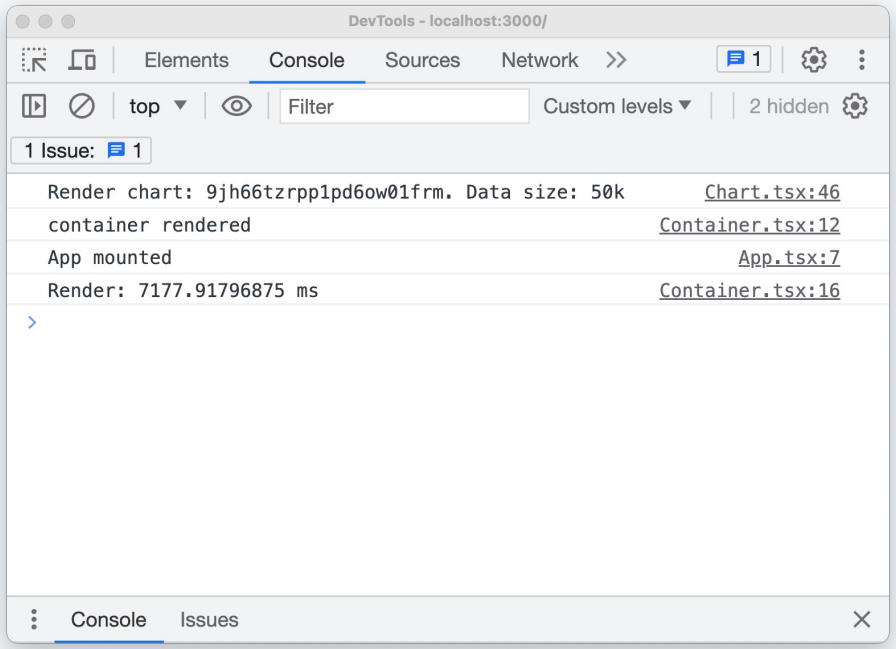


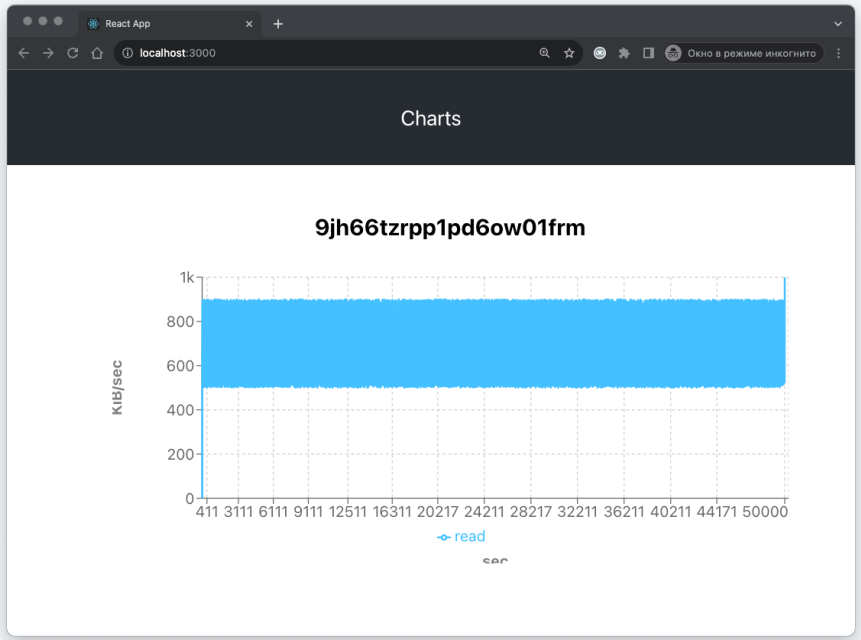
charts – Chart.tsx

```tsx
const formated: Record<string, string>  = {};

const addToFormat = (val?: number) => {
  if (val === undefined) {
    return;
  }
  const rounded = Math.round(val)
  const roundedString = rounded.toString();
  if (formated[roundedString]) {
    return;
  }
  formated[roundedString] = kFormatter(rounded);
}

console.time('format');
data.forEach((item) => {
  const {read, write} = item;
  addToFormat(read);
  addToFormat(write);
});
console.timeEnd('format');
```

Убираем линию

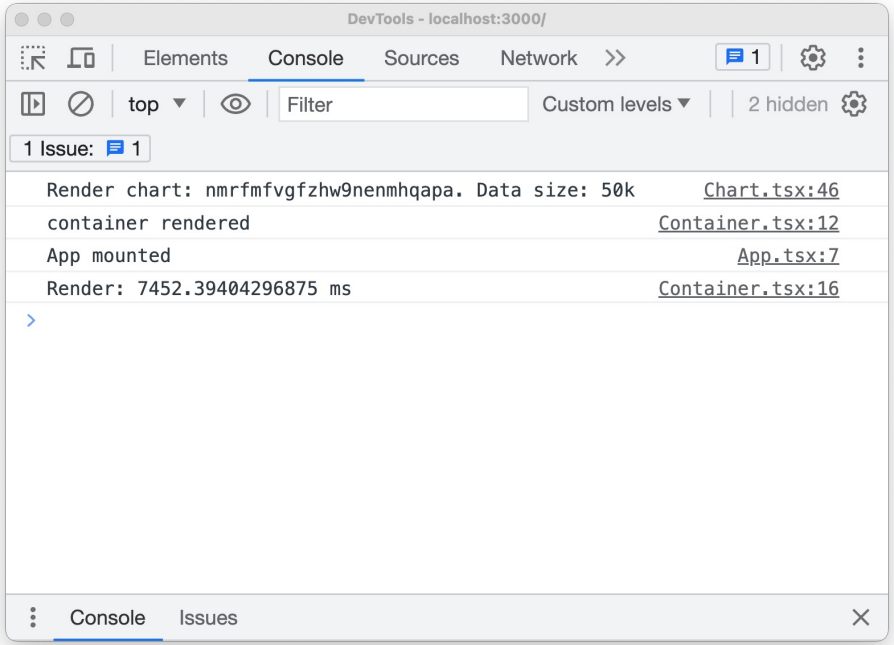
- 2%

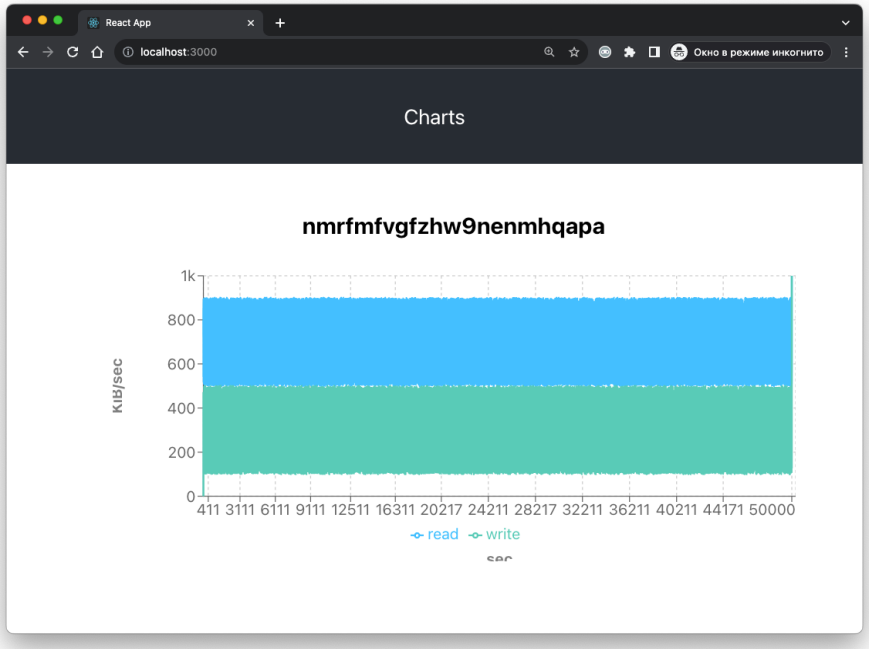




Убираем тултип

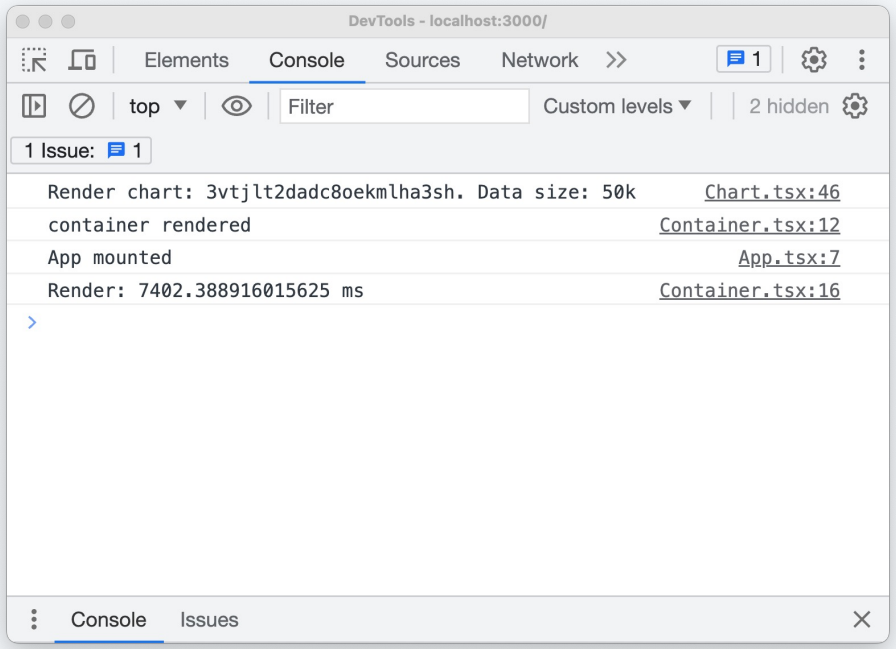
- 0%

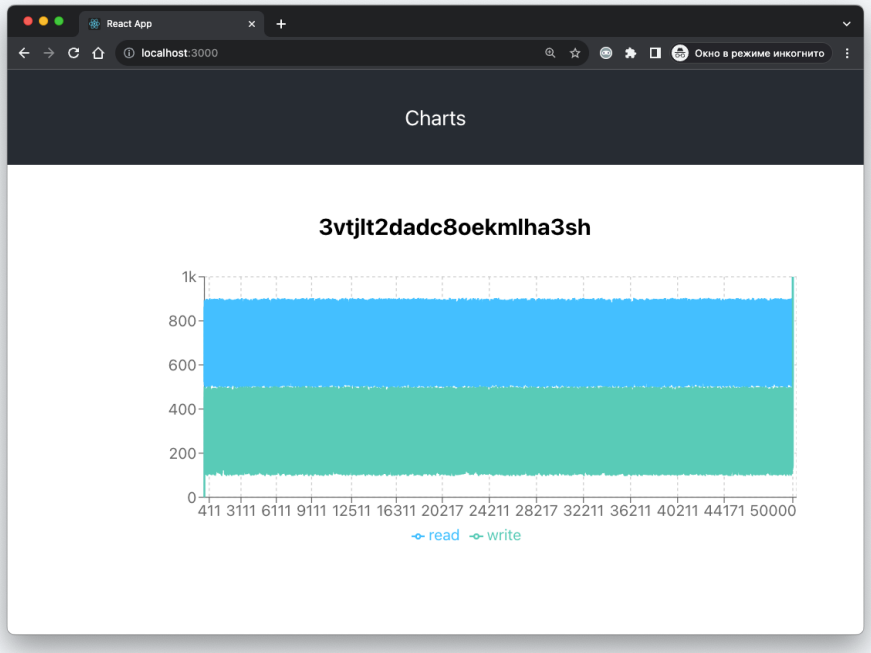




Убираем подписи осей

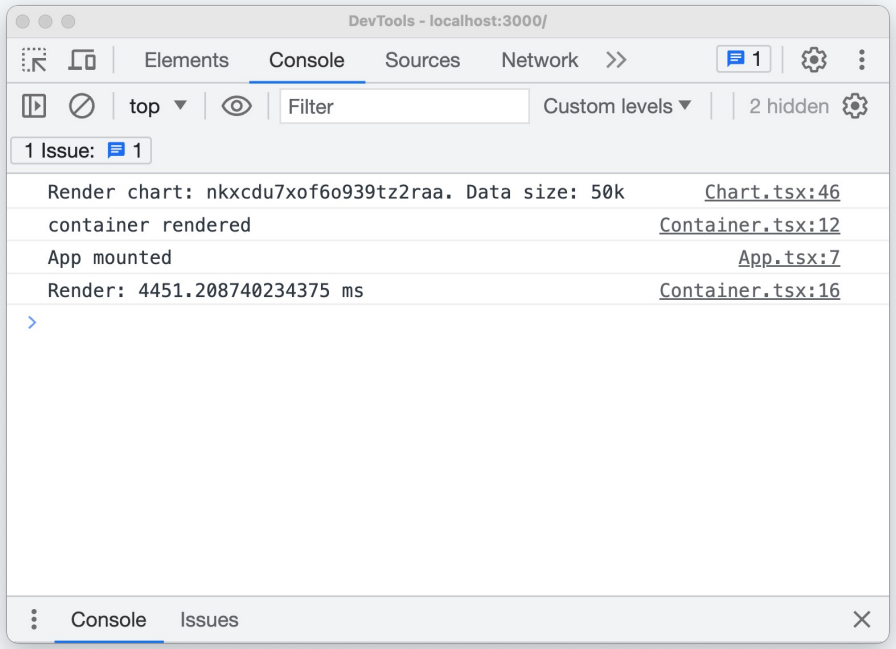
- 0%

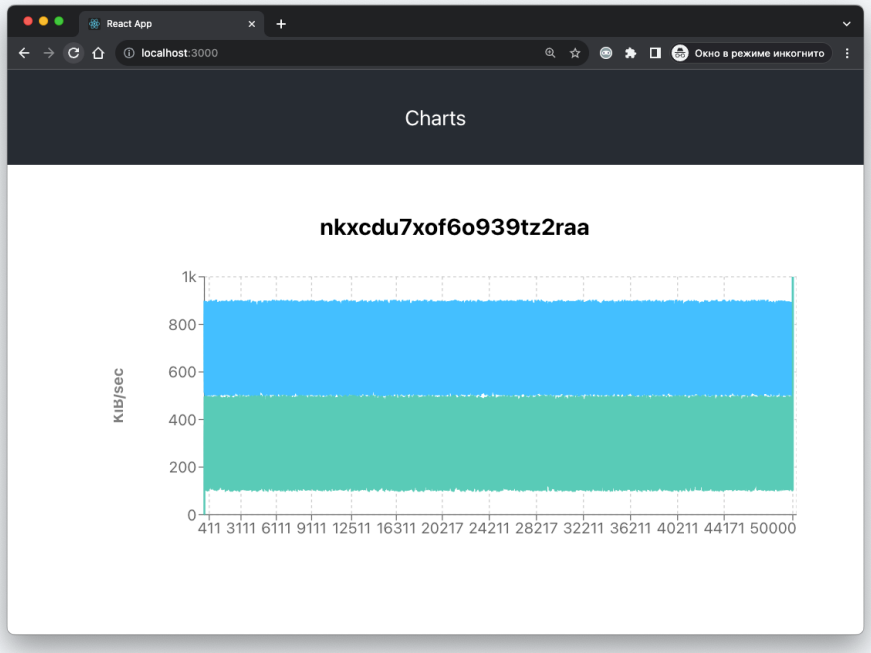




Убираем легенду

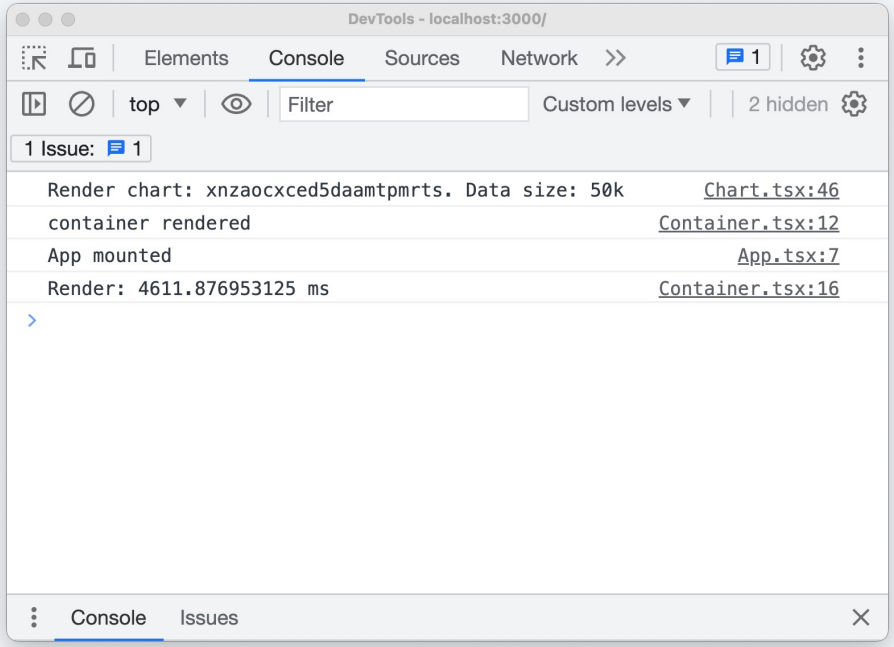
- 40%

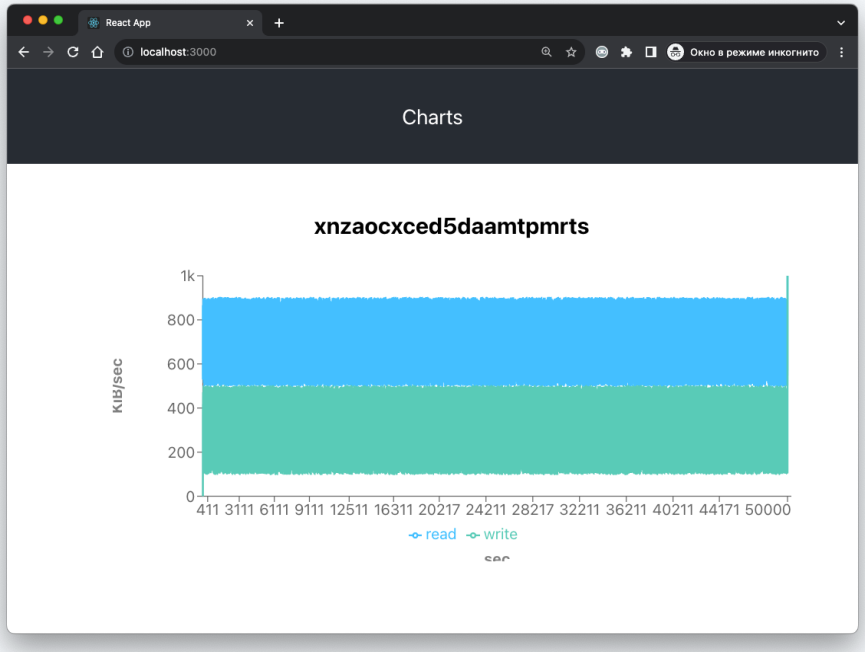




Убираем сетку

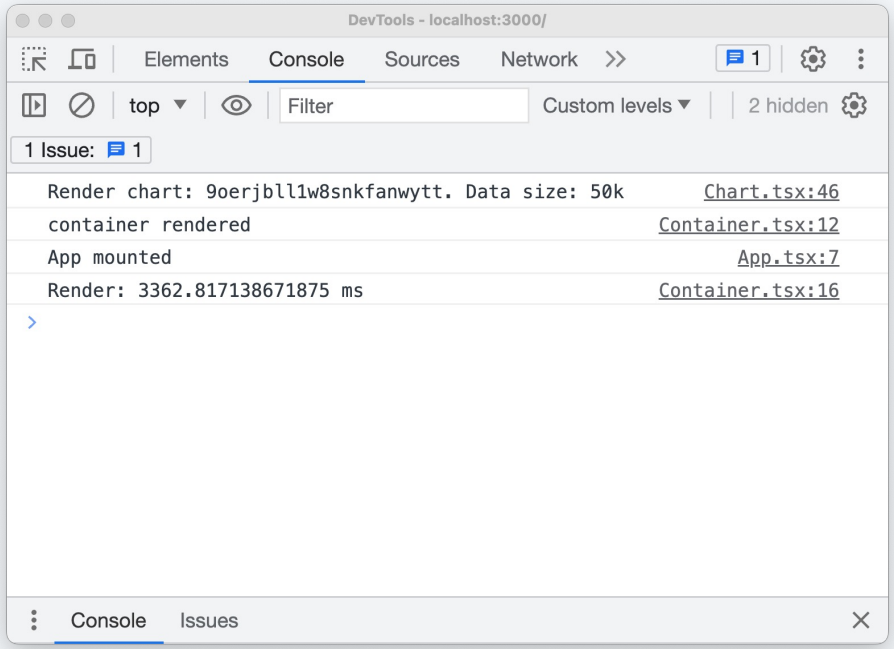
- 38%

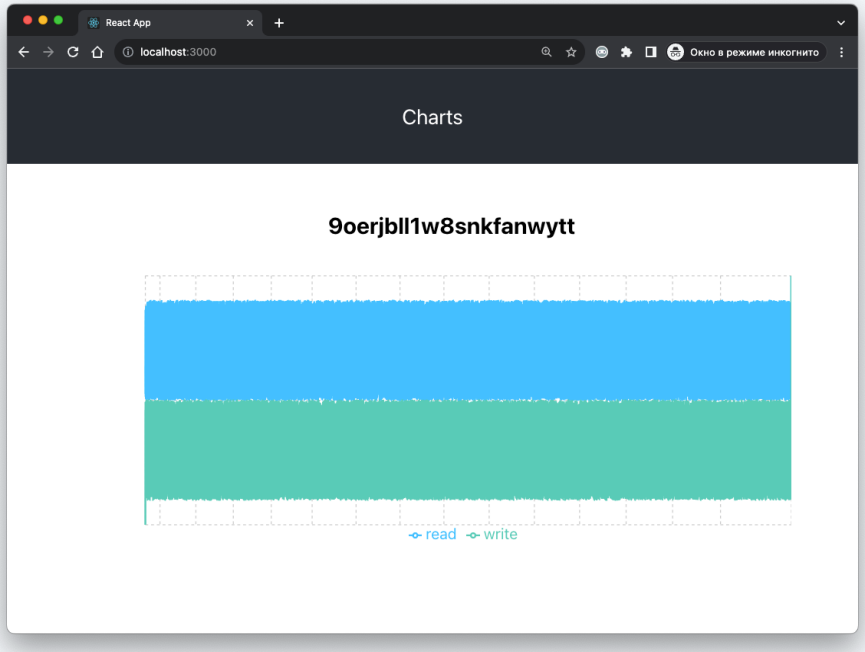




Убираем оси???

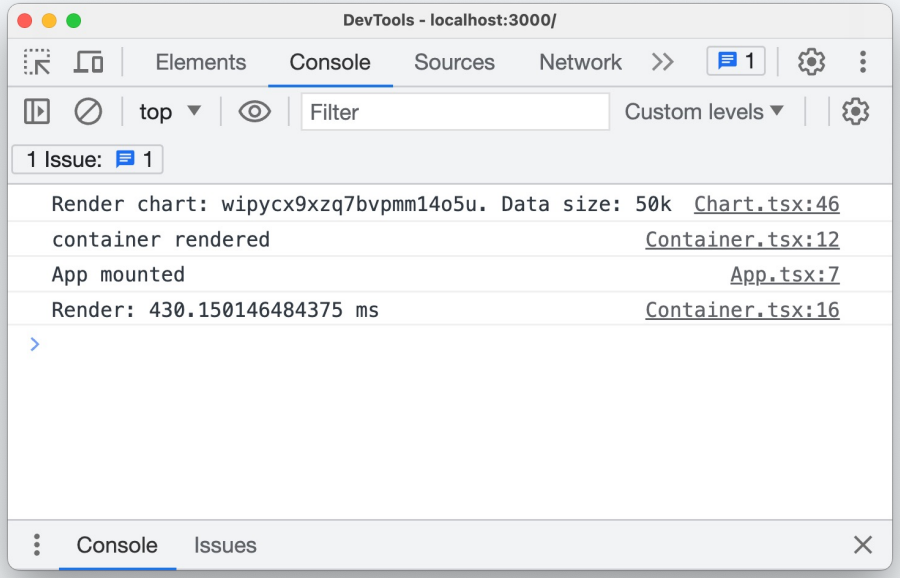
- 54%

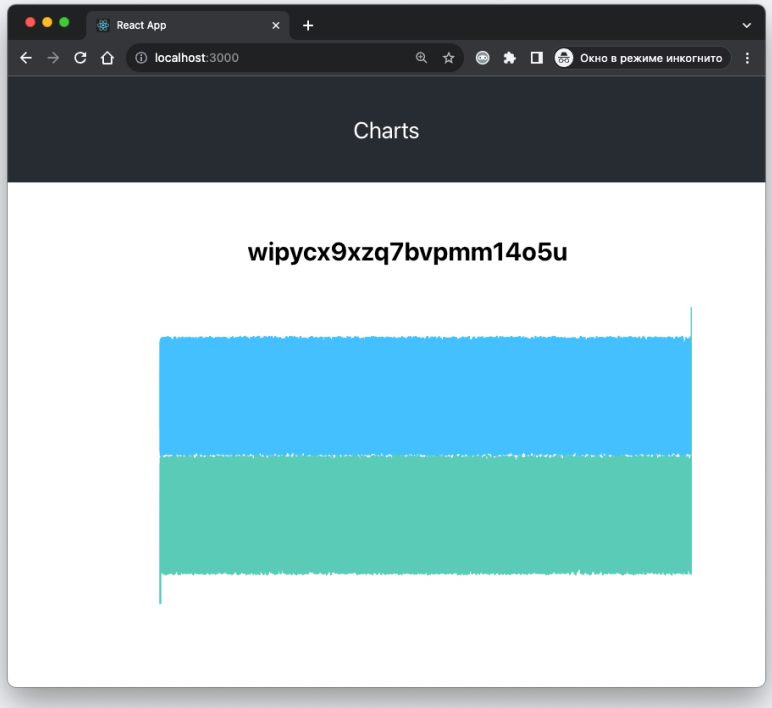




Убираем все

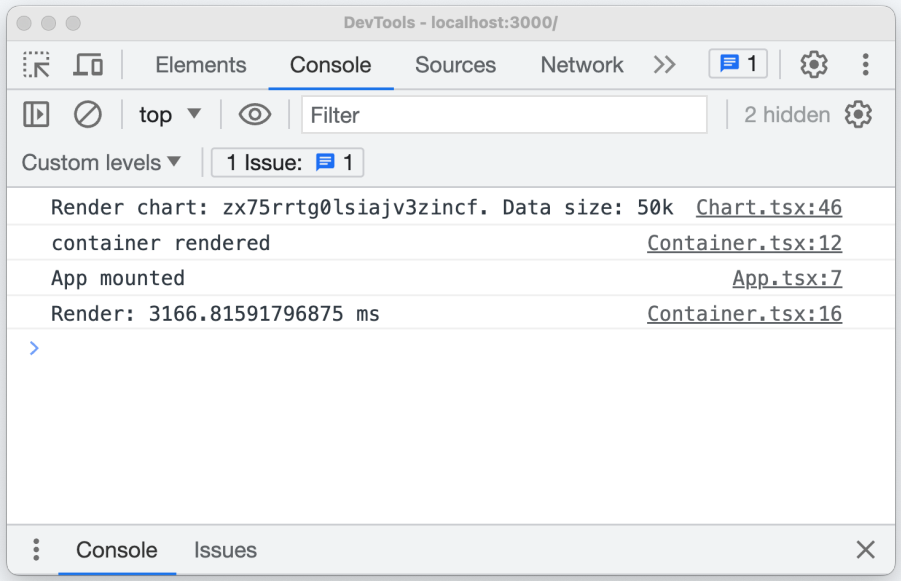
- 94%

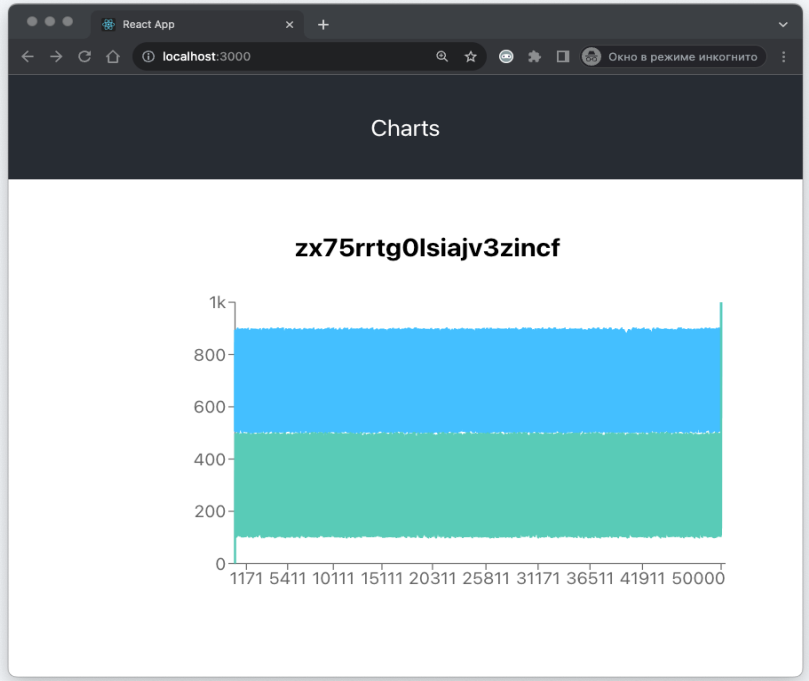




Убираем необязательное

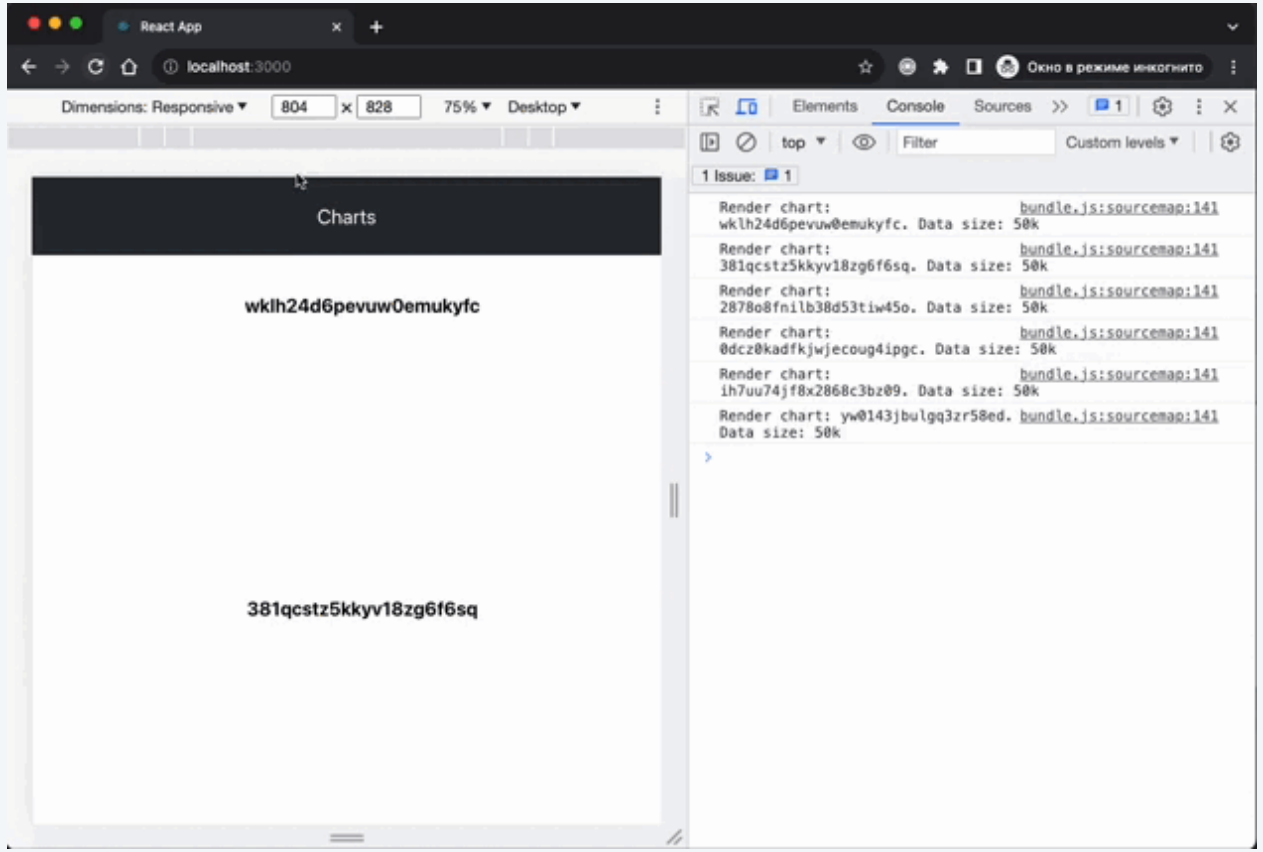
- 57%





Итог

- *20 сек LCP*
- *20 сек* до первого значимого рендера
- нет умирания хрома
- не очень *user friendly*



Обо мне

Проблема длиной в 90 секунд

Замеры перфоманса

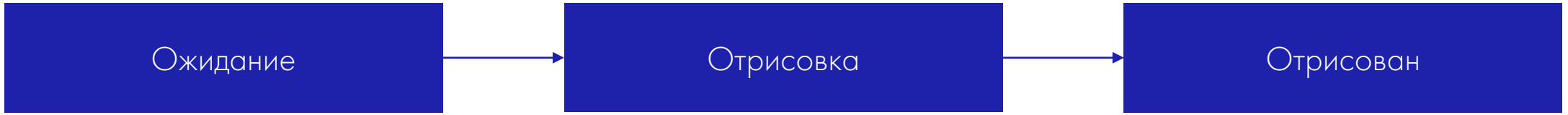
Оптимизация рендера

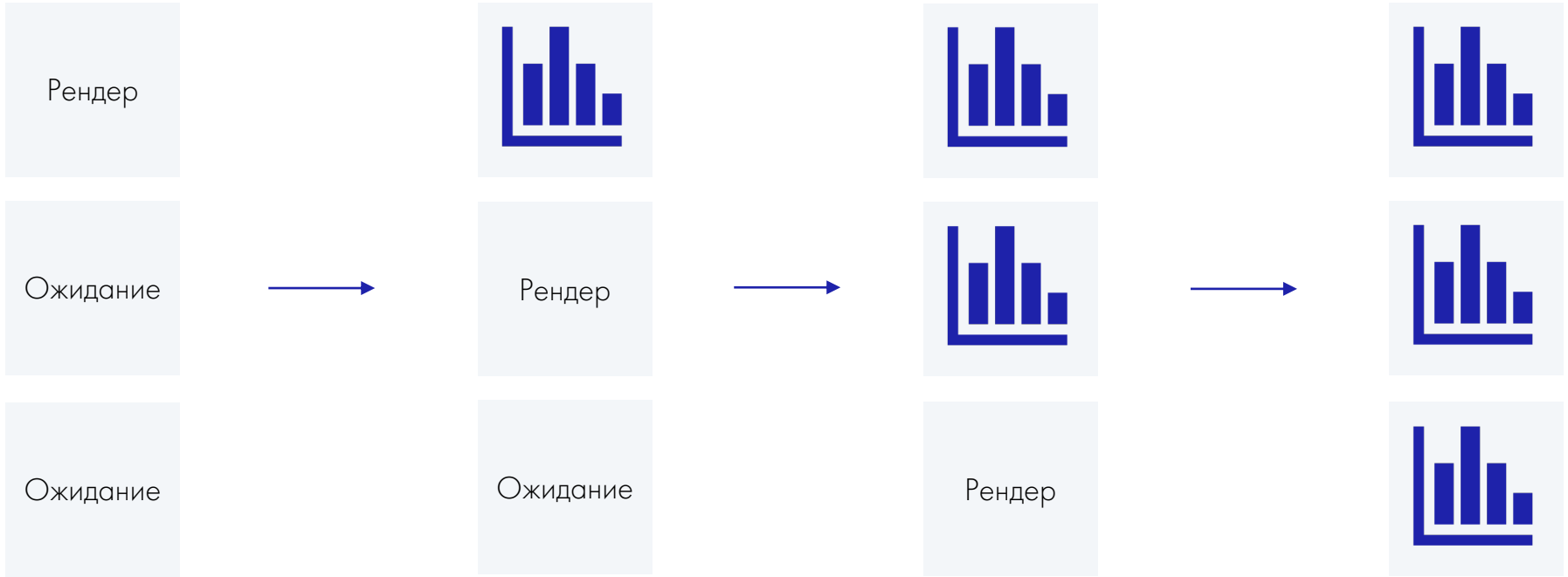
А что случилось?

Миша, все плохо, давай по новой!

Фантастические библиотеки и где они обитают

Три состояния





Контейнер

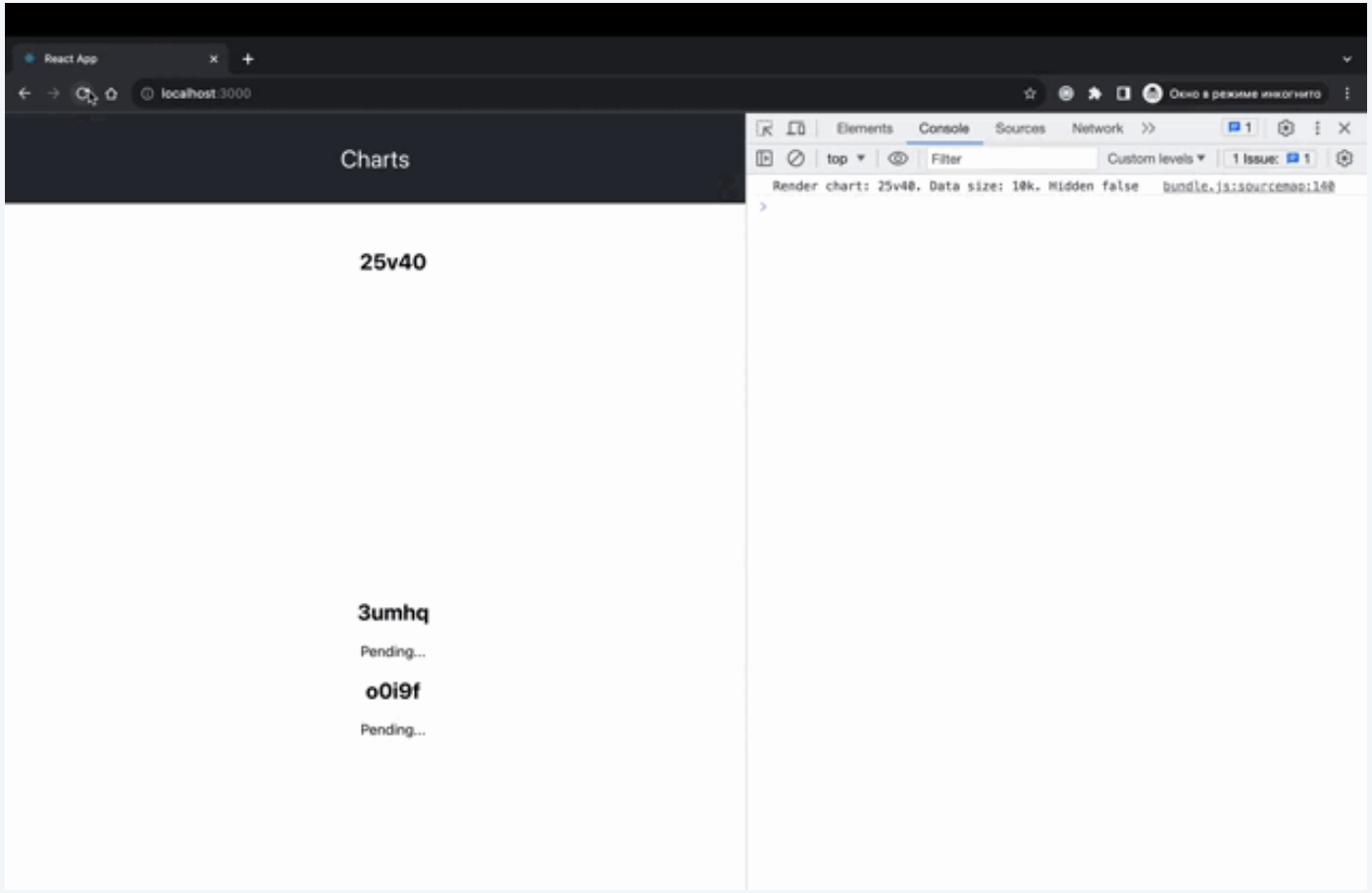
```jsx
const [renderIndex, setRenderIndex] = useState(0);

const handleRenderEnd = () => {
  setRenderIndex(renderIndex + 1);
};

return (
  <ul className='container'>
    {Object.entries(data).map(
      ([key, item]: [string, ChartData], index) => (
        <li key={key} className='chart'>
          <ChartBlock
            data={item.chartData}
            title={key}
            availableValues={item.availableValues}
            labelY={item.yAxisName}
            labelX={item.xAxisName}
            onEnd={handleRenderEnd}
            isHidden={index > renderIndex}
          />
        </li>
      )
    )}
  </ul>
);
```

График

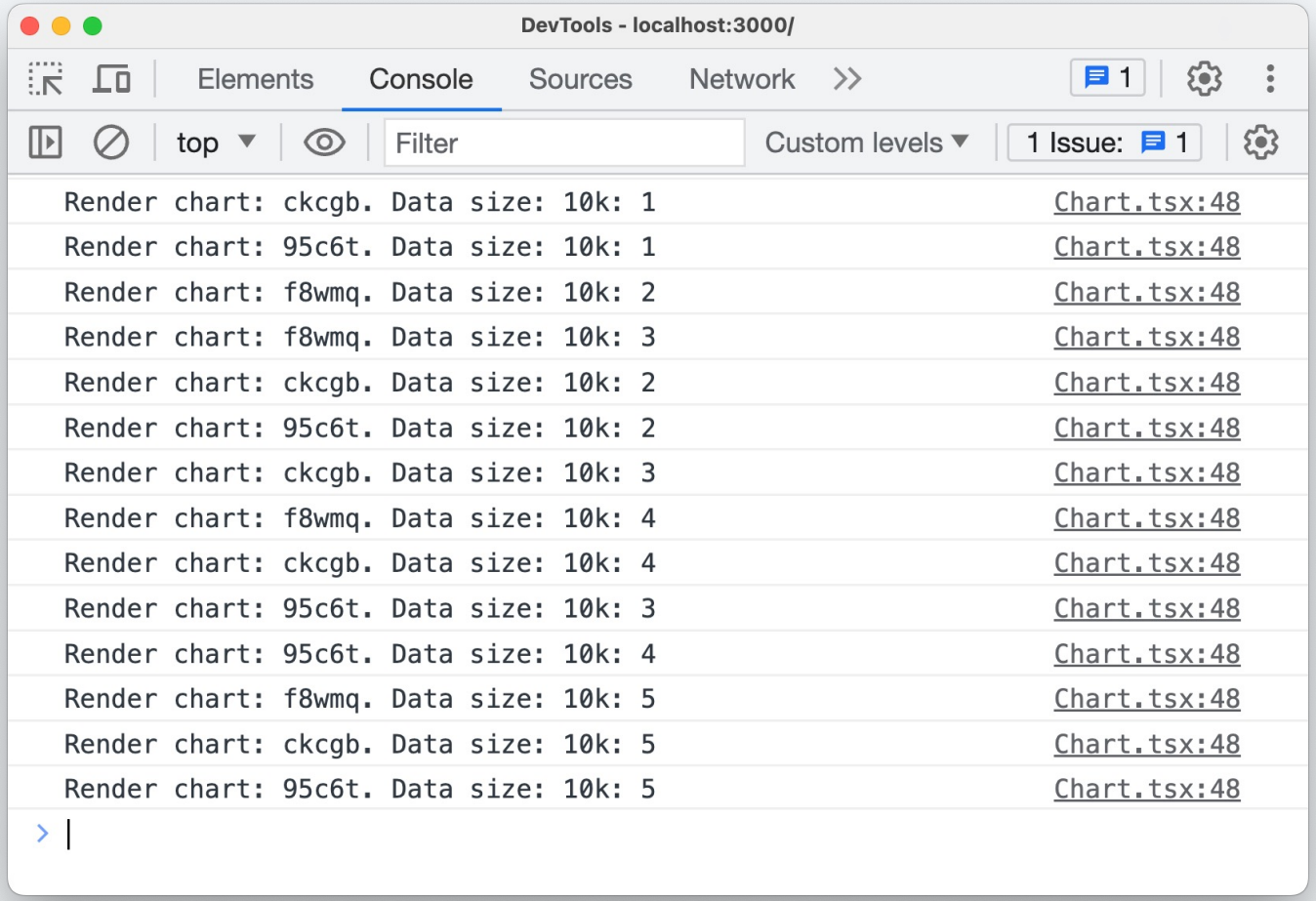
```tsx
const chatRef = useRef<HTMLDivElement>(null);
const [isRendering, setIsRendering] = React.useState(false);

useEffect(() => {
  if (isHidden) {
    return;
  }

  setIsRendering(true);

  const timerId = setInterval(() => {
    const line = chatRef.current?.querySelector('.recharts-line-curve');
    if (line) {
      clearInterval(timerId);
      setIsRendering(false);
      onEnd();
    }
  }, 250);

  return () => {
    clearInterval(timerId);
  };
}, [isHidden]);

if (isHidden) {
  return (
    <div>
      <h2>{title}</h2>
      Pending...
    </div>
  );
}
```

```tsx
return (
    <div ref={chatRef}>
      <h2>{title}</h2>
      {isRendering && <div className='render-block'>Rendering ...</div>}
      <div style={{ opacity: isRendering ? 0 : 1 }}>
        <ResponsiveContainer width='100%' height={320}>
          <LineChart
            data={data}
          >
            <XAxis dataKey='sec'/>
            <YAxis
              type='number'
              domain={['dataMin', 'auto']}
              allowDecimals={true}
              tickCount={6}
              tickFormatter={(value) => kFormatter(value)}
            />
            <Tooltip labelFormatter={(value) => `Sec : ${value}`} />
            {availableValues.map((field, index) => (
              <Line
                key={index}
                type='monotone'
                dataKey={field}
                stroke={generateColor(field)}
                dot={false}
                strokeWidth={2}
              />
            ))}
          </LineChart>
        </ResponsiveContainer>
      </div>
    </div>
  );
```

Первый драфт



Количество рендеров



Изменяющийся пропс

```
1  const [renderIndex, setRenderIndex] = useState(0);
2
3  const handleRenderEnd = () => {
4    setRenderIndex(renderIndex + 1);
5  };
6
7  return (
8    <ul className='container'>
9      {Object.entries(data).map(
10       ([key, item]: [string, ChartData], index) => (
11         <li key={key} className='chart'>
12           <ChartBlock
13             data={item.chartData}
14             title={key}
15             availableValues={item.availableValues}
16             labelY={item.yAxisName}
17             labelX={item.xAxisName}
18             onEnd={handleRenderEnd}
19             isHidden={index > renderIndex}
20           />
21         </li>
22       )
23     )}
24   </ul>
25 );
```

useCallback

Неправильный вариант

charts – Container.tsx

```
const handleRenderEnd = useCallback(() => {
  setRenderIndex(renderIndex + 1)
}, []);
```

Правильный вариант

charts – Container.tsx

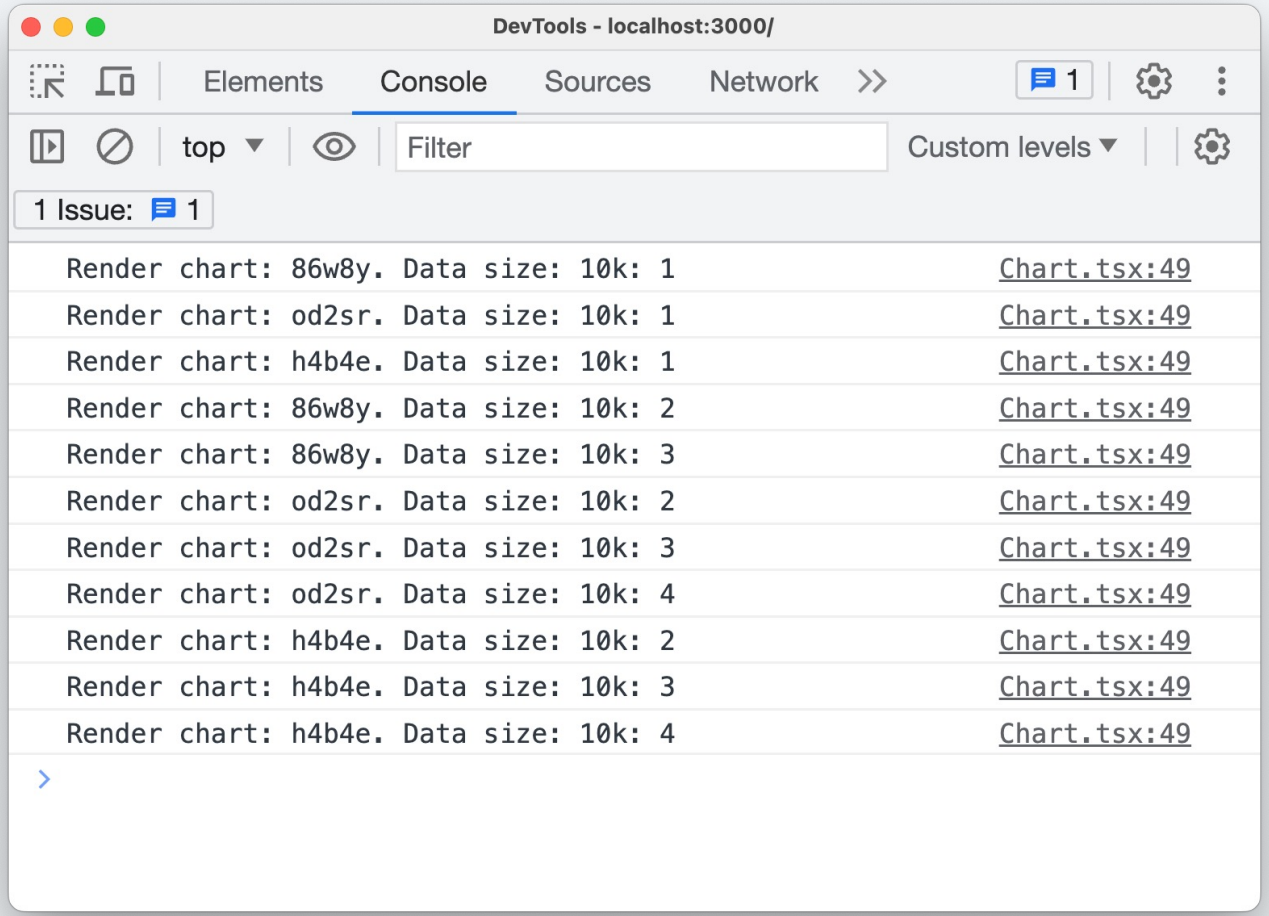
```
const handleRenderEnd = useCallback(() => {
  setRenderIndex(prev => prev + 1)
}, []);
```

Мемоизация графика

charts - Chart.tsx

```tsx
export const ChartBlock = React.memo(ChartBlockIn, (prev, next) => {
  return prev.isHidden === next.isHidden;
});
```

Количество рендеров

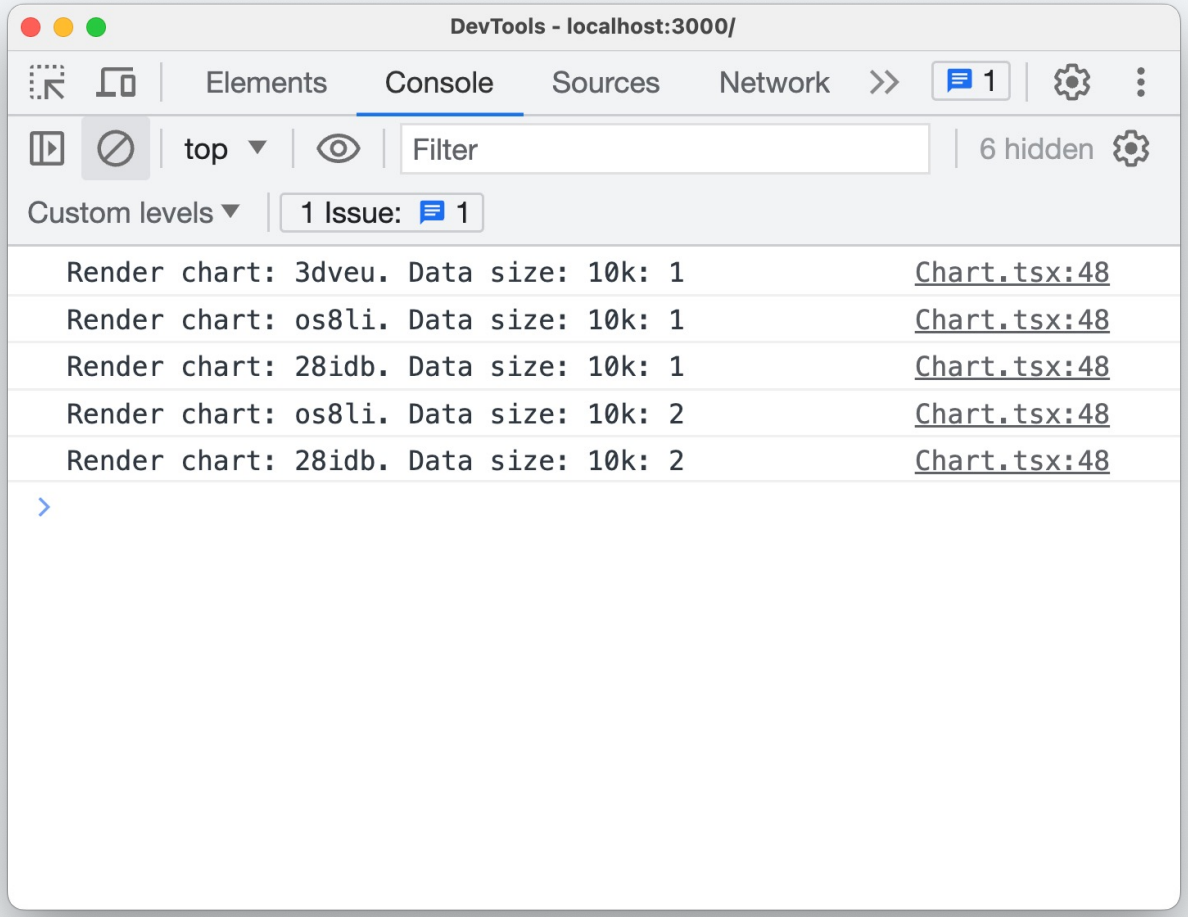


Избавляемся от стейта

```
useEffect(() => {
  if (isHidden) {
    return;
  }

  const timerId = setInterval(() => {
    const line = chatRef.current?.querySelector('.recharts-line-curve');
    if (line) {
      clearInterval(timerId);

      const chart = chatRef.current?.querySelector('.chart-block');
      const renderingText = chatRef.current?.querySelector('.render-t
      if (chart && renderingText) {
        chart.classList.remove('hidden');
        renderingText.classList.add('hidden');
      }
      onEnd();
    }
  }, 250);

  return () => {
    clearInterval(timerId);
  };
}, [isHidden]);
```

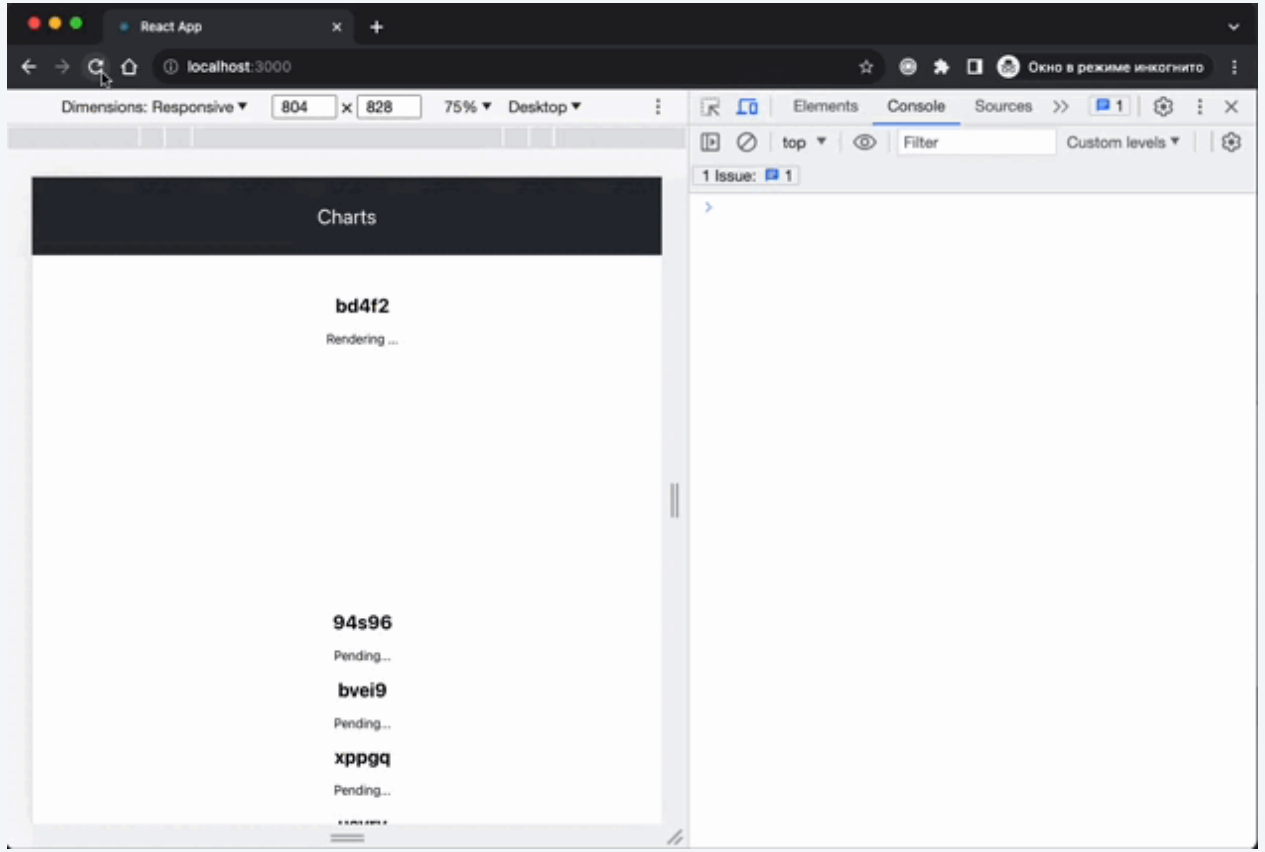
```
return (
  <div ref={chatRef}>
    <h2>{title}</h2>
    <div className='render-text'>Rendering ...</div>
    <div className='chart-block hidden'>
      <ResponsiveContainer width='100%' height={320}>
        <LineChart
          data={data}
          margin={{ top: 20, bottom: 20, left: 34 }}
        >
          //...
```

Количество рендеров



Итог

- Те же 20 сек LCP
- 2 сек до первого значимого рендера
- Пользователь понимает, что вообще происходит



Обо мне

Проблема длиной в 90 секунд

Замеры перфоманса

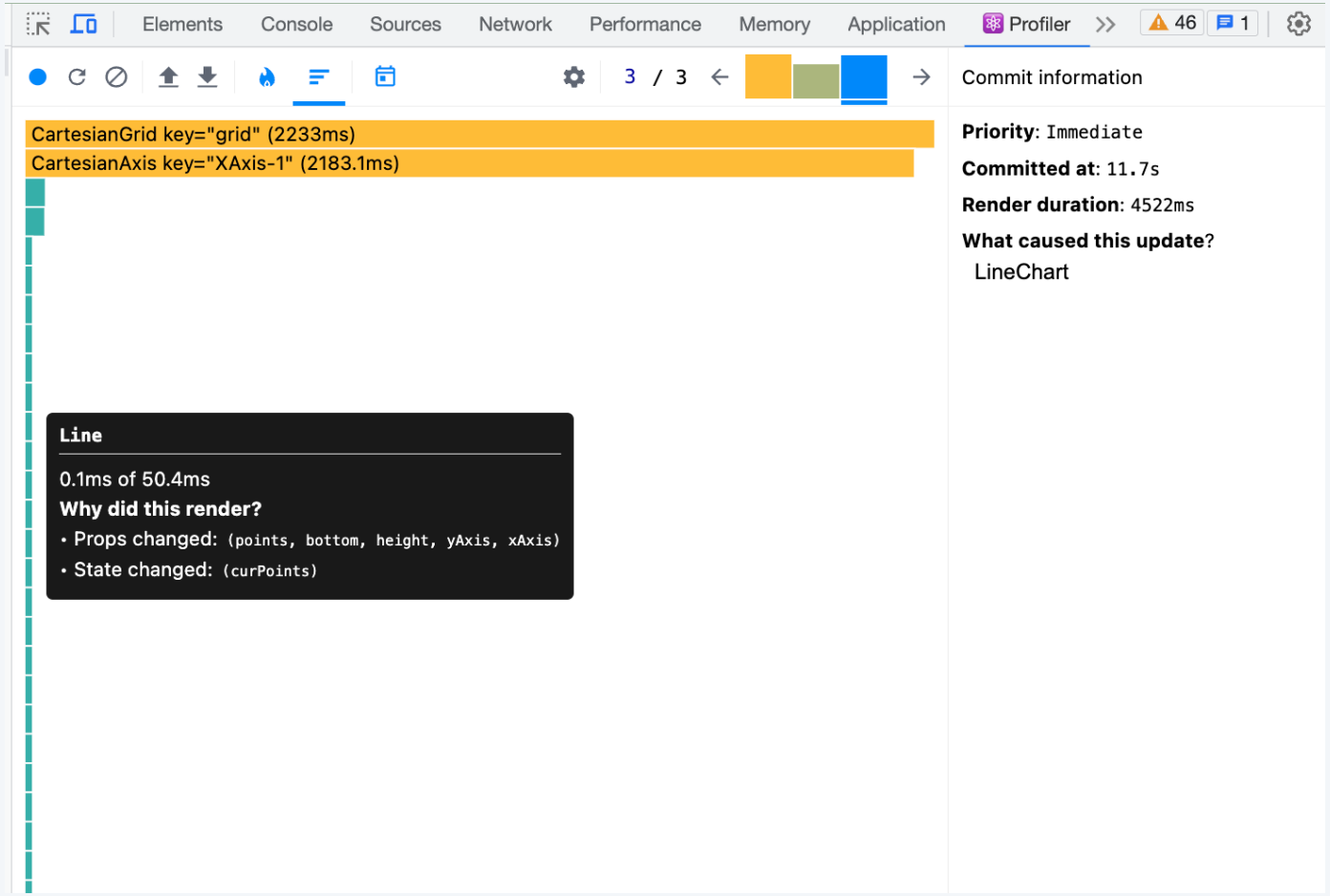
Оптимизация рендера

А что случилось?

Миша, все плохо, давай по новой!

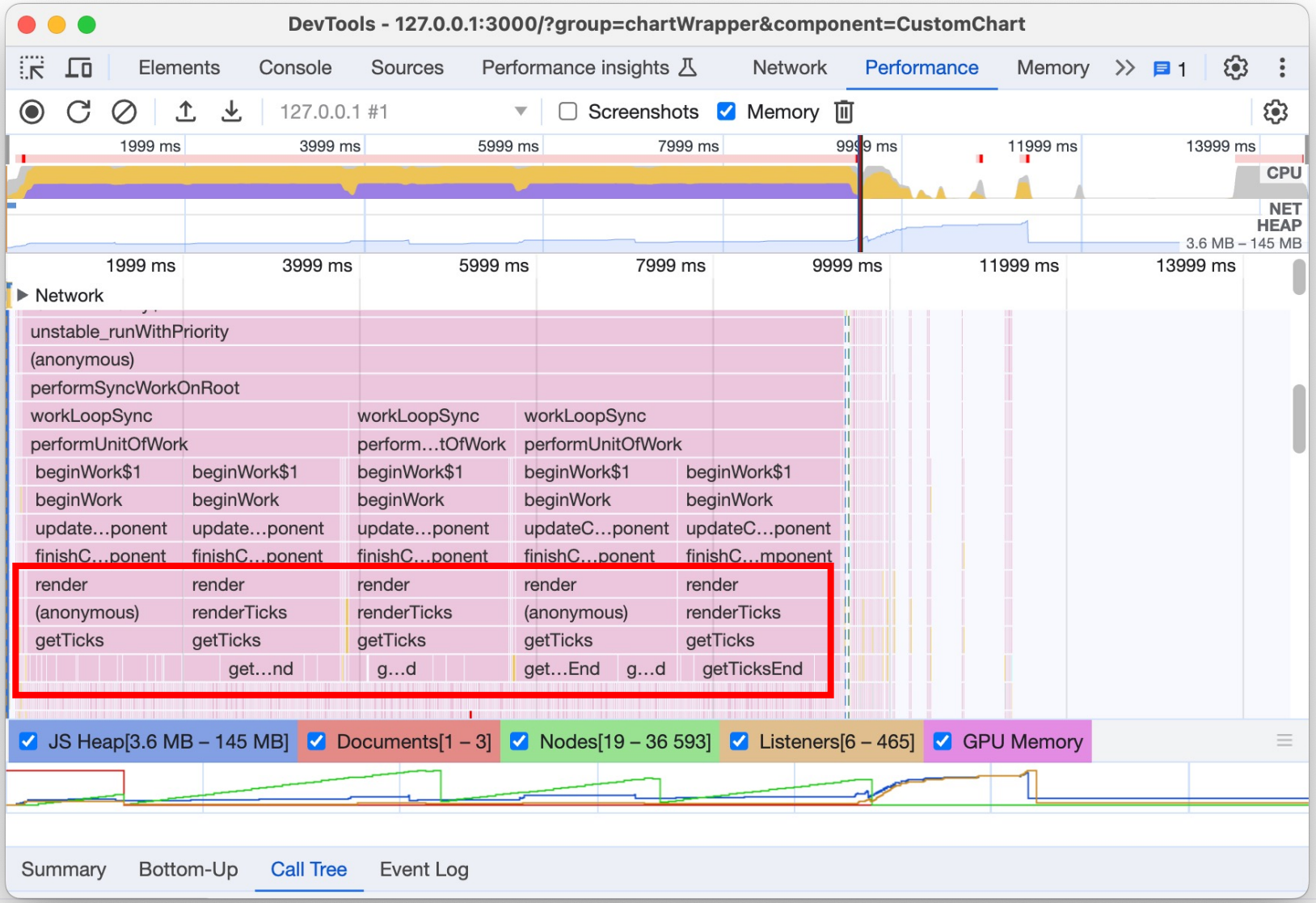
Фантастические библиотеки и где они обитают

React dev tools. Profiler



Profiler

5 рендеров одного компонента - CartesianAxis



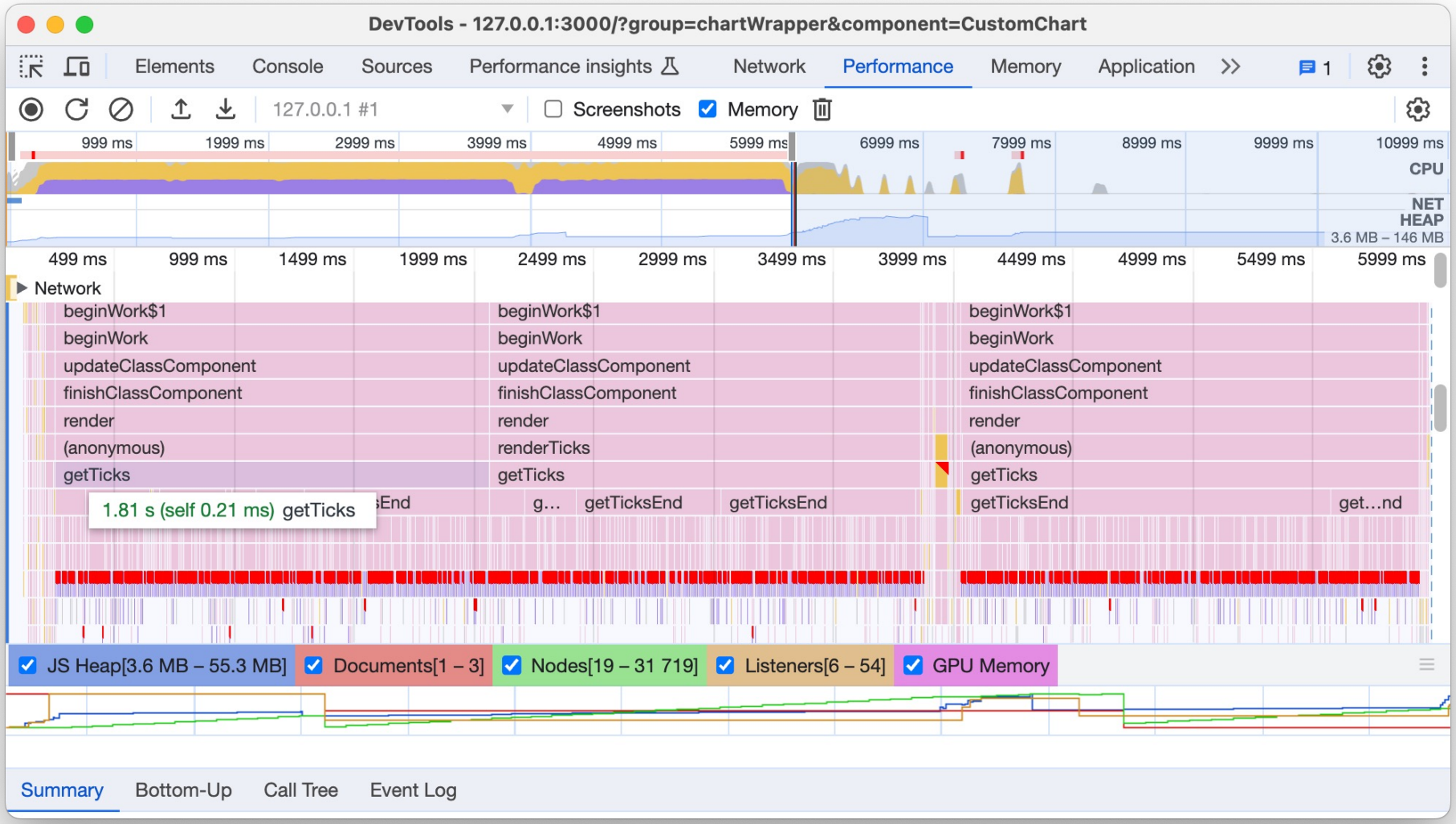
Изменяющиеся пропсы

```
height 221.5 240                                                    CartesianAxis.tsx
scale ƒ scale(x) {                                                  CartesianAxis.tsx
    return x == null || isNaN(x = +x) ? unknown : (output || (output =
piecewise(domain.map(transform), range, interpolate)))(transform(clamp(x)));
  } ƒ scale(x) {
    return x == null || isNaN(x = +x) ? unknown : (output || (output =
piecewise(domain.map(transform), range, interpolate)))(transform(clamp(x)));
  }
domain ▶ (2) [0, 1000] ▶ (2) [0, 1000]                              CartesianAxis.tsx
originalDomain ▶ (2) [0, 'auto'] ▶ (2) [0, 'auto']                  CartesianAxis.tsx
niceTicks ▶ (5) [0, 250, 500, 750, 1000]                            CartesianAxis.tsx

fontSize 16px undefined                                             CartesianAxis.tsx:69
letterSpacing normal undefined                                      CartesianAxis.tsx:69
render: 4                                                           CartesianAxis.tsx:352
```

```
scale ƒ scale(d) {                                                  CartesianAxis.tsx:69
    let i = index.get(d);
    if (i === undefined) {
      if (unknown !== implicit) return unknown;
      index.set(d, i = domain.push(d) - 1);
    }
    return range[i % range.length];
 … ƒ scale(d) {
    let i = index.get(d);
    if (i === undefined) {
      if (unknown !== implicit) return unknown;
      index.set(d, i = domain.push(d) - 1);
    }
    return range[i % range.length];
 …
domain                                                              CartesianAxis.tsx:69
  (20002) [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
  21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 4
▶ 1, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61,
  62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 8
  2, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, …]
  (20002) [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
  21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 4
▶ 1, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61,
  62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 8
  2, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, …]
duplicateDomain                                                     CartesianAxis.tsx:69
  (20002) [0, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
  20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 4
▶ 0, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60,
  61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 8
  1, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, …]
  (20002) [0, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
  20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 4
▶ 0, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60,
  61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 8
  1, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, …]
y 241.5 260                                                         CartesianAxis.tsx:69
```

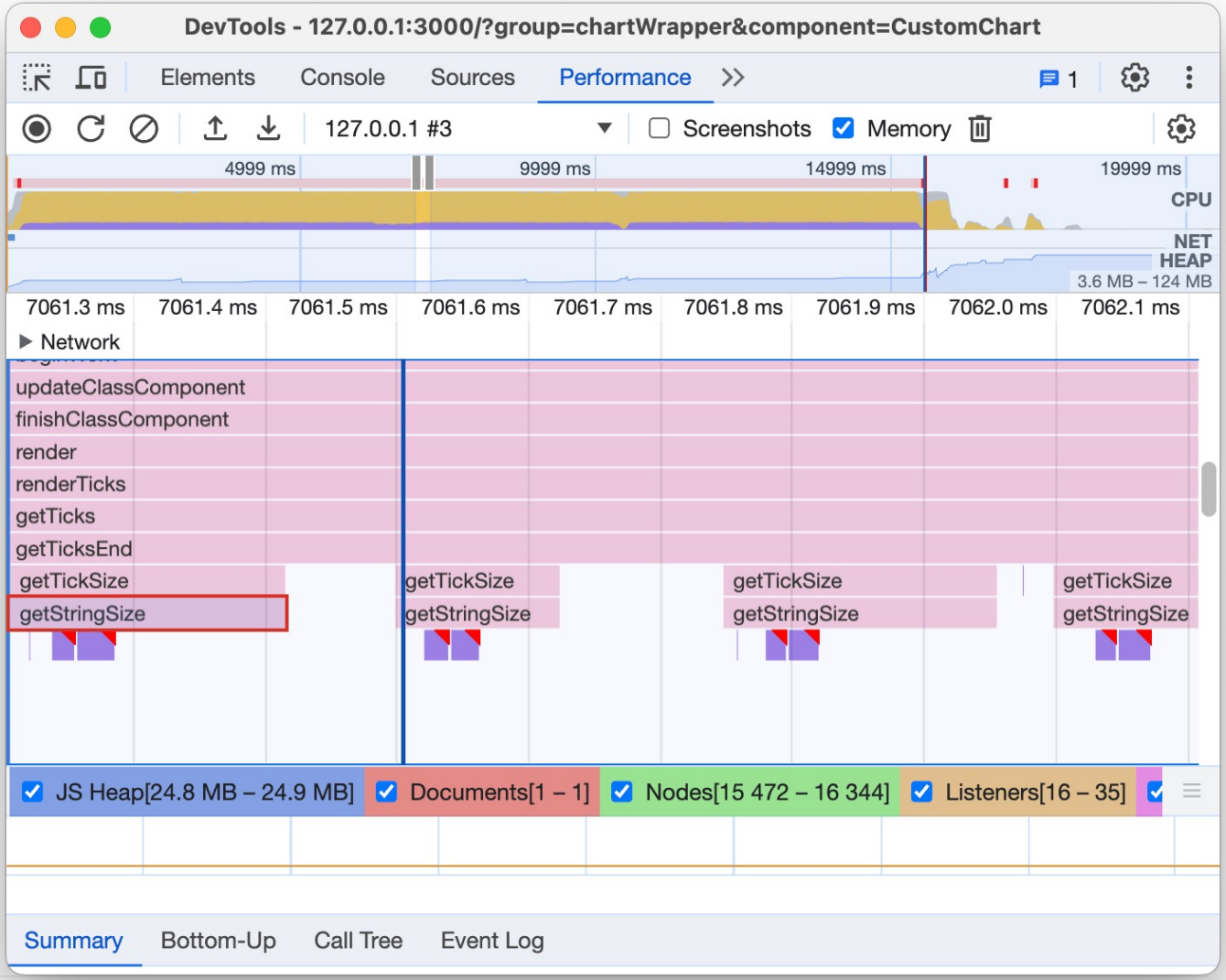
Из чего состоит render

- Render практически полностью состоит из getTicks
- getTicks состоит из нескольких вызовов getTicksEnd



Из чего состоит getTicksEnd

getTicksEnd состоит
из getTickSize/getStringSize
и промежутки на свой
внутренний код



А ЧТО СЛУЧИЛОСЬ?

getTicksEnd

- Проход в цикле по массиву длиной количество точек
- Спреды внутри цикла
- На каждый цикл вызов getTickSize/getStringSize

```
1  function getTicksEnd(
2    sign: Sign,
3    boundaries:{  start: number; end: number},
4    getTickSize:( tick: CartesianTickItem, index: number) => number,
5    ticks: CartesianTickItem[],
6    minTickGap: number,
7  ): CartesianTickItem[] {
8    const result =( ticks ||[]). slice();
9    const len = result.length;
10
11   const{  start}  = boundaries;
12   let{  end}  = boundaries;
13
14   for( let i = len - 1; i >= 0; i--) {
15     let entry = result[i];
16     const size = getTickSize(entry, i);
17     if( i === len - 1) {
18       const gap = sign *( entry.coordinate +( sign * size) / 2 - end);
19       result[i] = entry ={
20         ...entry,
21         tickCoord: gap > 0 ? entry.coordinate - gap * sign : entry.coordinate,
22 };
23 }     else{
24       result[i] = entry ={  ...entry, tickCoord: entry.coordinate};
25 }
26
27     const isShow = isVisible(sign, entry.tickCoord, size, start, end);
28
29     if( isShow) {
30       end = entry.tickCoord - sign *( size / 2 + minTickGap);
31       result[i] ={  ...entry, isShow: true};
32 }
33 }
34
35   return result;
36 }
```

getStringSize

```
1  export const getStringSize =( text: string | number, style: CSSProperties =
   {}) : Size =>{
2    //...
3    try{
4      let measurementSpan = document.getElementById(MEASUREMENT_SPAN_ID);
5
6      if( !measurementSpan) {
7        measurementSpan = document.createElement('span');
8        measurementSpan.setAttribute('id', MEASUREMENT_SPAN_ID);
9        measurementSpan.setAttribute('aria-hidden', 'true');
10       document.body.appendChild(measurementSpan);
11 }
12
13     //...
14
15     measurementSpan.textContent =str;
16
17     const rect = measurementSpan.getBoundingClientRect();
18     const result ={  width: rect.width, height: rect.height};
19
20     //...
```

Итог

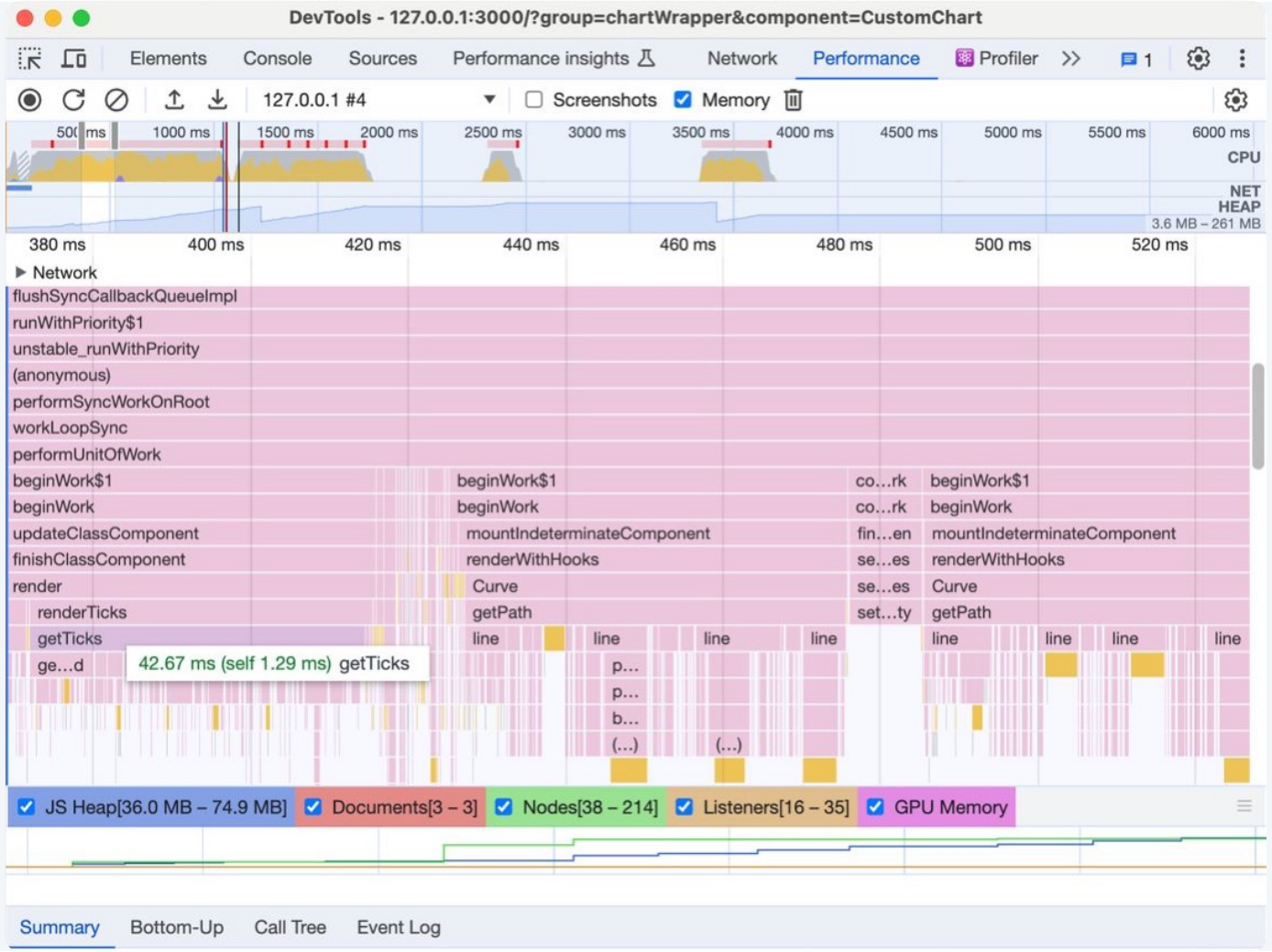
На 1 рендер графика приходится:

- 5 перендеров
- На каждый перендер приходится несколько getTicksEnd
- На каждый getTicksEnd приходится n-точек обращений и изменений DOM

На рендер 6 графиков по 50к точек приходится:

- 6 * 5 * 50000 ≈ 1.5млн обращений к DOM
- Виной всему неоптимальный алгоритм работы библиотеки

Бонус: А что если?



Обо мне

Проблема длиной в 90 секунд

Замеры перфоманса

Оптимизация рендера

А что случилось?

Миша, все плохо, давай по новой!

Фантастические библиотеки и где они обитают

Chart.js

Chart.js

Simple yet flexible JavaScript charting library for the modern web

Подключение

```
import {
  CategoryScale,
  Chart as ChartJS,
  Legend,
  LineElement,
  LinearScale,
  PointElement,
  Title,
  Tooltip,
} from 'chart.js';
import { Line } from 'react-chartjs-2';

ChartJS.register(
  CategoryScale,
  LinearScale,
  PointElement,
  LineElement,
  Title,
  Tooltip,
  Legend
);
```

МИША, ВСЕ ПЛОХО, ДАВАЙ ПО НОВОЙ!

Настройки

```
const chartData = {
  labels: data.map((item) => item.sec),
  datasets: availableValues.map((field) => ({
    label: field,
    data: data.map((item) => item[field]),
    fill: false,
    backgroundColor: generateColor(field),
    borderColor: generateColor(field),
    borderWidth: 2,
  })),
};
```

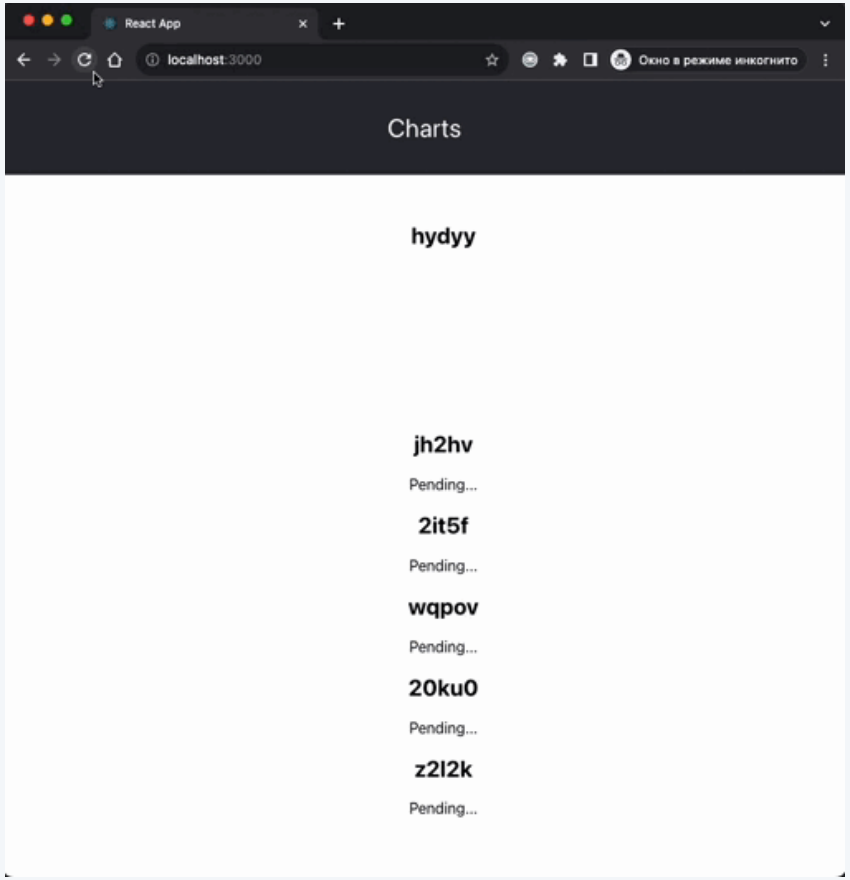
```
const chartOptions = {
    scales: {
      x: {
        title: {
          type: 'linear' as const,
          display: true,
          text: labelX,
          font: {
            weight: 'bold',
          },
        },
      },
      y: {
        type: 'linear' as const,
        display: true,
        position: 'left' as const,
        ticks: {
          callback: (value: any) => kFormatter(value).toString(),
        },
        min: 0,
        title: {
          display: true,
          text: labelY,
          font: {
            weight: 'bold',
          },
        },
      },
    },
    plugins: {
      title: {
        display: false,
      },
      legend: {
        display: true,
      },
    },
    elements: {
      point: {
        radius: 0,
      },
    },
  };
```

Компонент

```jsx
return (
  <div ref={chatRef}>
    <h2>{title}</h2>
    <div className='chart-block'>
      <Line data={chartData} options={chartOptions} />
    </div>
  </div>
);
```

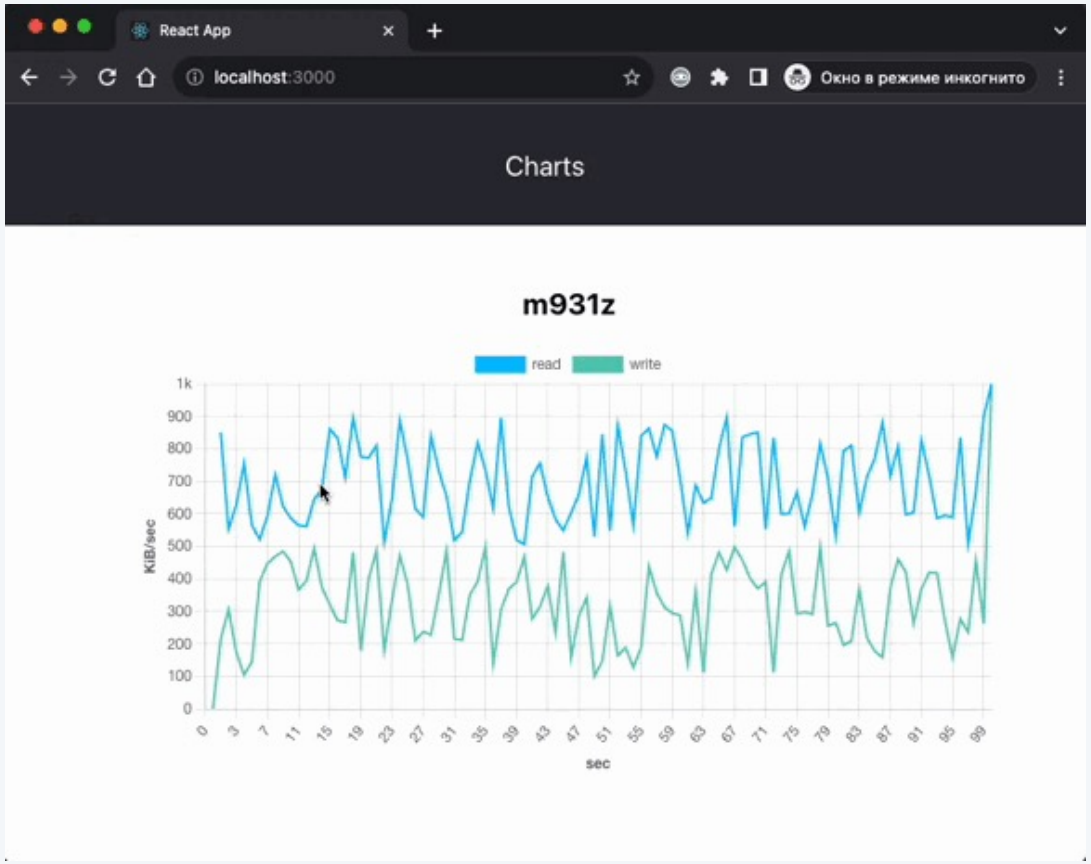
Итог

- 10 сек FCP
- 2 первый значимый рендер
- Хороший отклик
- Большая куча кода для подключения
- Пара проблемок



Проблема тултипа

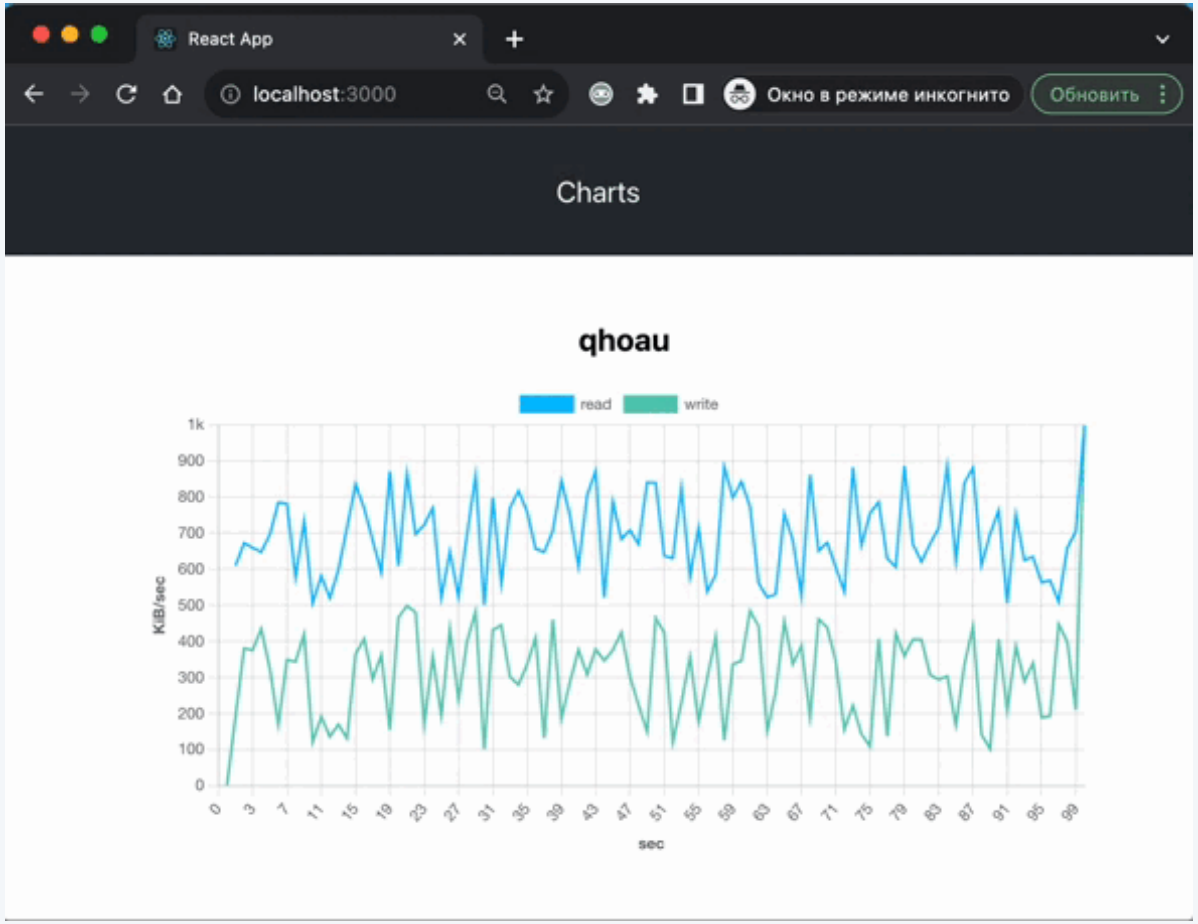
Нужно точно попасть на точку



Решение с тултипом

```
plugins: {
  tooltip: {
    intersect: false,
    callbacks: {
      usePointStyle: true,
      label: (context: any) =>
        return `Sec: ${context{label}, ${context.dataset.label}: ${context.formattedValue}`
      },
    },
  },
  title: {
    display: false,
  },
  legend: {
    display: true,
  },
},
interactions: {
  intersect: false,
  mode: "x" as const,
},
```

Итог


React App
localhost:3000
Окно в режиме инкогнито
Обновить
Charts
qhoau
read
write
KiB/sec
1k
900
800
700
600
500
400
300
200
100
0
0
3
7
11
15
19
23
27
31
35
39
43
47
51
55
59
63
67
71
75
79
83
87
91
95
99
sec

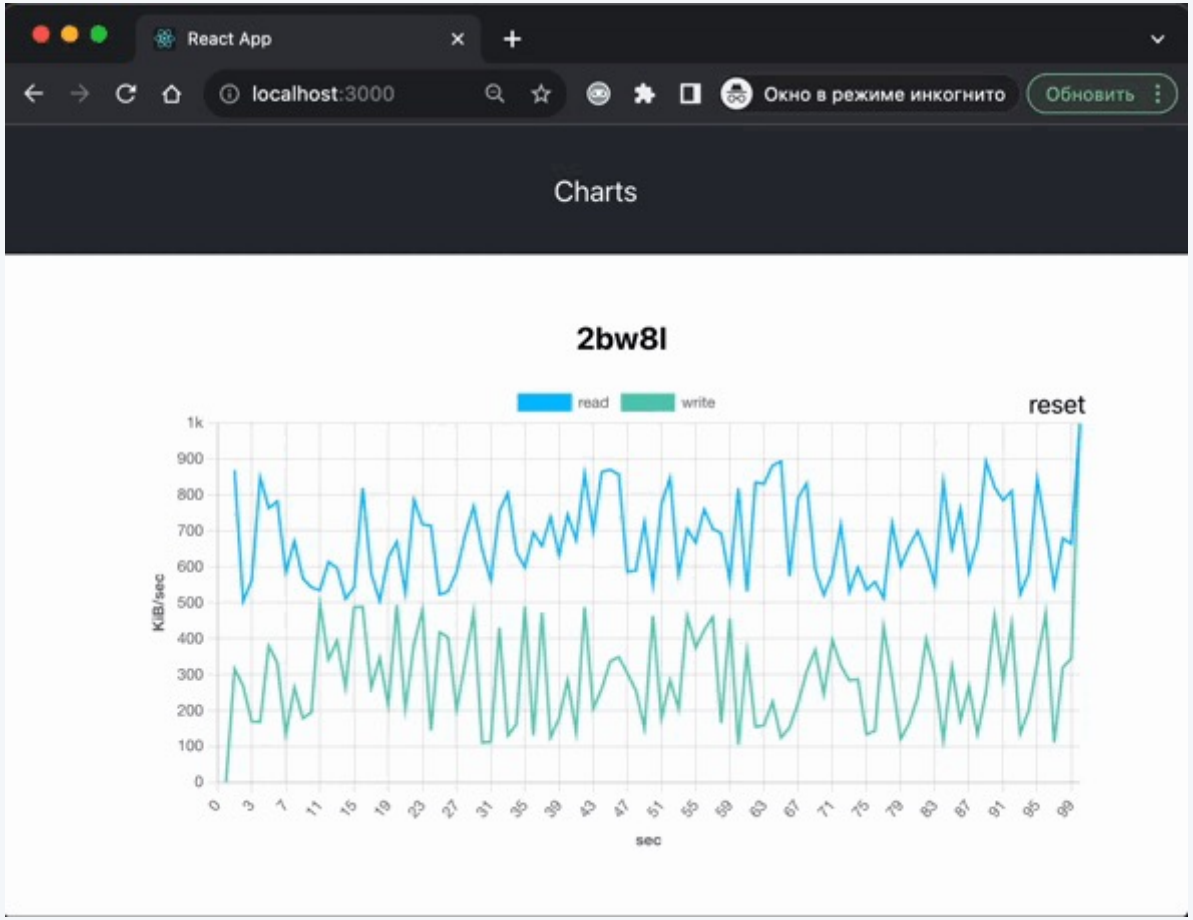
Зум

```
import zoomPlugin from "chartjs-plugin-zoom";
ChartJS.register(
  CategoryScale,
  LinearScale,
  PointElement,
  LineElement,
  Title,
  Tooltip,
  Legend,
  zoomPlugin
);
```

```
plugins: {
  zoom: {
    zoom: {
      mode: 'x' as const,
      drag: {
        enabled: true,
        backgroundColor: 'rgba(0, 0, 0, 0.5)',
      },
    },
  },
```

```
<div className='chart-block'>
  <Line
    data={chartData}
    options={chartOptions}
    ref={chartInstanceRef}
  />
  <div
    className='reset'
    onClick={() => {
      if (chartInstanceRef.current) {
        const chart = chartInstanceRef.current as any;
        chart.resetZoom();
      }
    }}
  >reset</div>
</div>
```

Зум


React App
localhost:3000
Окно в режиме инкогнито
Обновить
Charts
2bw8l
read
write
reset
KiB/sec
1k
900
800
700
600
500
400
300
200
100
0
0 3 7 11 15 19 23 27 31 35 39 43 47 51 55 59 63 67 71 75 79 83 87 91 95 99
sec

Обо мне

Проблема длиной в 90 секунд

Замеры перфоманса

Оптимизация рендера

А что случилось?

Миша, все плохо, давай по новой!

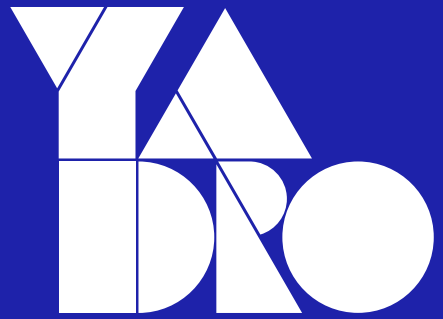
Фантастические библиотеки и где они обитают

Сравнение

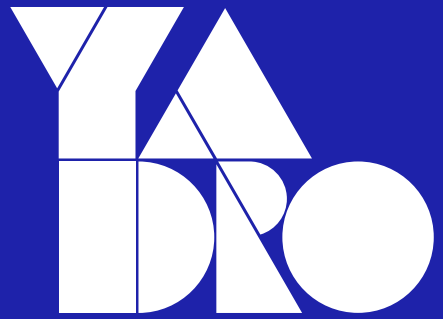
Параметр	Recharts	Chart.js	D3.js	Highcharts	Plotly.js
Простота использования	Высокая	Высокая	Низкая	Средняя	Средняя
Гибкость и настраиваемость	Средняя	Средняя	Высокая	Высокая	Высокая
Интерактивность	Средняя	Средняя	Средняя	Высокая	Высокая
Адаптивность в React	Высокая	Высокая	Средняя	Средняя	Высокая
Документация и сообщество	Средняя	Высокая	Высокая	Средняя	Высокая
Лицензирование и стоимость	Бесплатно	Бесплатно	Бесплатно	Коммерческая	Бесплатно/Коммерческая
Поддержка и обновления	Средняя	Высокая	Высокая	Средняя	Высокая
Тип рендера	*SVG*	*Canvas*	*SVG*	*SVG/Canvas*	*WebGL/SVG*

* этот слайд сделал чатжпт

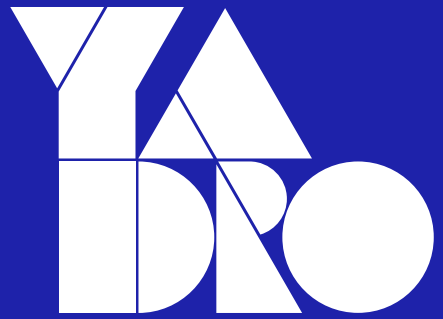
Заключение



design@yadro.com



design@yadro.com



design@yadro.com

Ответ на вопрос

Нет, я не пробовал другие библиотеки.