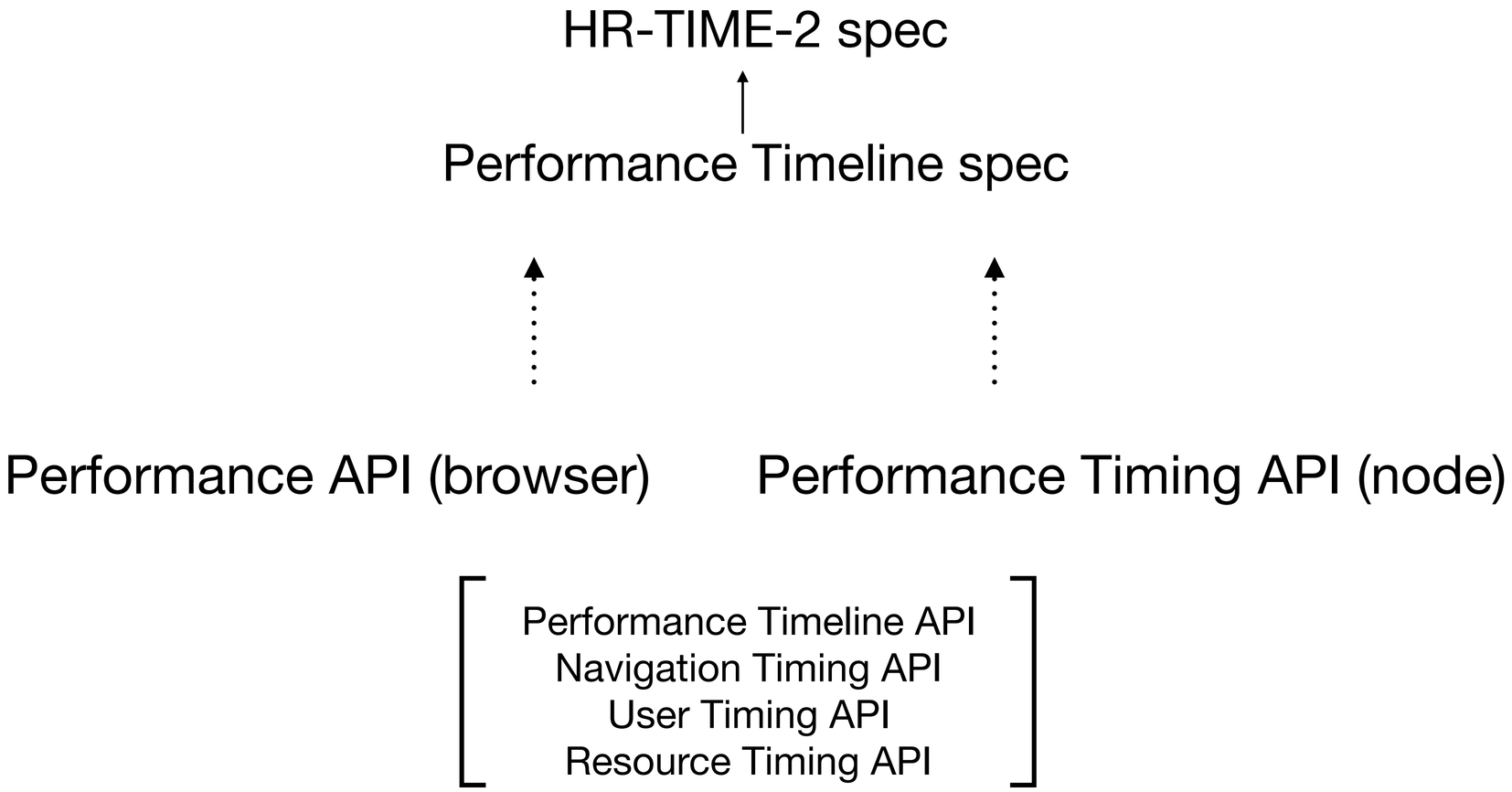
Node
Performance Timing API

Вера Никитинская P4262

Спецификации



Performance Timing API

perf_hooks – модуль ядра node

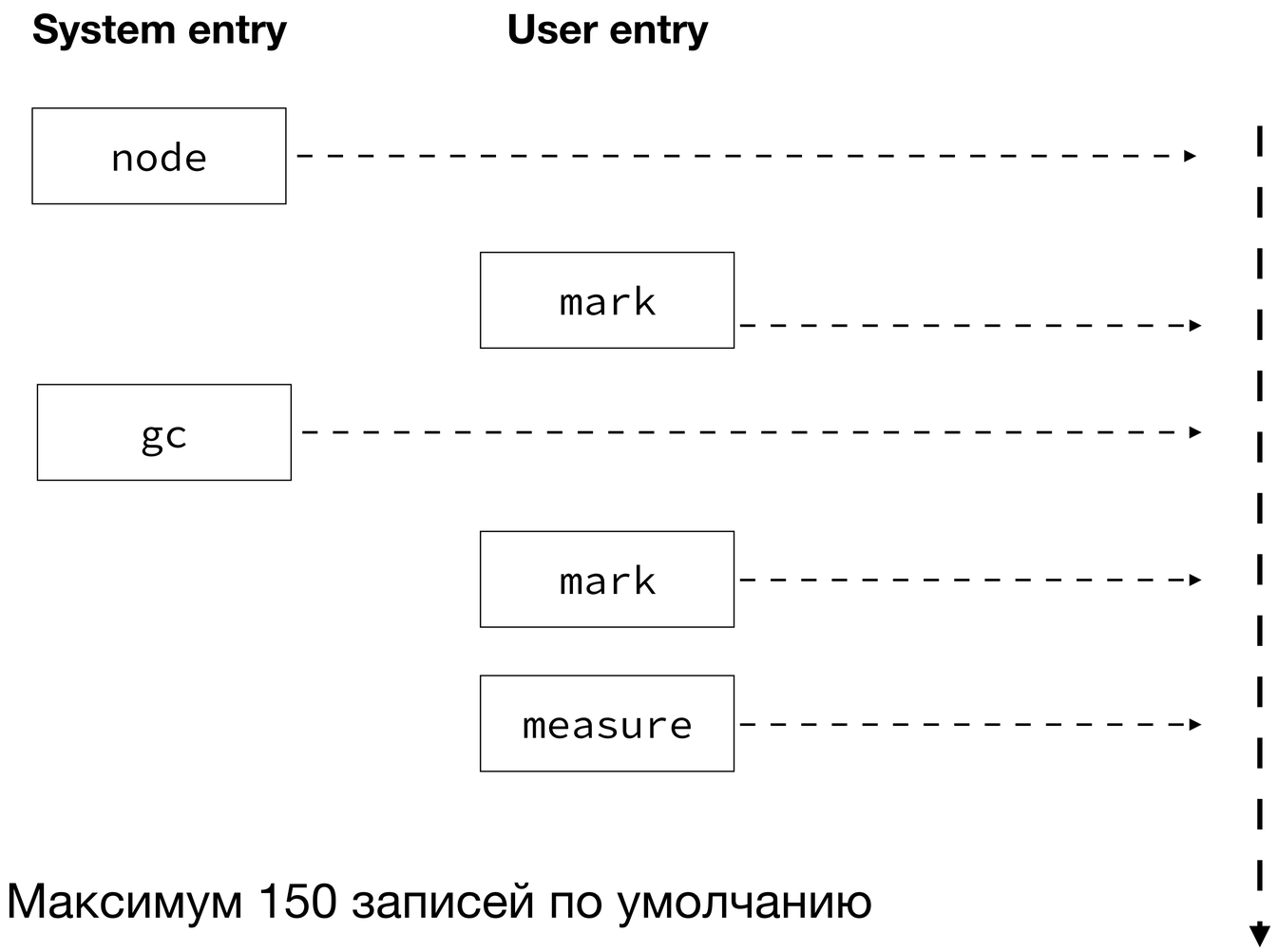
Начиная с node 8.5.0, август 2017

```
const {
    performance,
    PerformanceObserver
} = require('perf_hooks';
```

Как работает

- *perf_hooks.performance* возвращает объект Performance
- Performance дает доступ к Performance Timeline
  - связанный список последовательно появляющихся в таймлайне объектов PerformanceEntry

Performance Timeline



Максимум 150 записей по умолчанию

PerformanceEntry

```
console.log(performance.getEntries()); // только до node 9

[ PerformanceEntry {
    name: 'A to B',
    entryType: 'measure',
    startTime: 205.157366,
    duration: 25.734086 },
  PerformanceEntry {
    name: 'A',
    entryType: 'mark',
    startTime: 205.157366,
    duration: 0 },
  } ... ]
```

entryType

6 видов записей PerformanceEntry:

– mark
– measure
– node
– gc
– function
– http2

entryType: node

Записывается первой, автоматически

```
{ name: 'node',
  entryType: 'node',
  startTime: 0,
  duration: 217.51575499773026,
  nodeStart: 30.049373984336853,
  v8Start: 42.87327700853348,
  bootstrapComplete: -1,
  environment: 71.10232901573181,
  loopStart: -1,
  loopExit: -1,
  thirdPartyMainStart: -1,
  thirdPartyMainEnd: -1,
  clusterSetupStart: -1,
  clusterSetupEnd: -1,
  moduleLoadStart: 174.59000700712204,
  moduleLoadEnd: 174.6149640083313,
  preloadModuleLoadStart: 174.61514300107956,
  preloadModuleLoadEnd: 174.65145403146744 }
```

entryType: function

Оборачиваем функцию в performance.timerify() и замеряем длительность её выполнения

```
function someFunction() {
  console.log('hello world');
}

const wrapped = performance.timerify(someFunction);

wrapped(); // Здесь будет создана запись в таймлайне
```

Пример с таймерами из д/з

Подключаем модули

```
const {
  performance,
  PerformanceObserver } = require(‘perf_hooks’);

const Timer = require(‘timerpromise');
```

```
const obs = new PerformanceObserver((items) => {
  console.log(items.getEntries());
});

obs.observe({ entryTypes: ['mark', 'measure'] });

(async () => {
  performance.mark(‘t0');

  await new Timer(3).start;

  performance.mark('t1');
  performance.measure('t0 to t1', 't0', ‘t1');

  await new Timer(2).start;

  performance.mark('t2');
  performance.measure('t1 to t2', 't1', 't2');
  performance.measure('t0 to t2', 't0', 't2');
})();
```

Расставляем метки

```
const {
  performance,
  PerformanceObserver } = require(‘perf_hooks’);

const Timer = require(‘timerpromise');


const obs = new PerformanceObserver((items) => {
  console.log(items.getEntries());
});

obs.observe({ entryTypes: ['mark', 'measure'] });

(async () => {
  performance.mark(‘t0');

  await new Timer(3).start;

  performance.mark('t1');
  performance.measure('t0 to t1', 't0', ‘t1');

  await new Timer(2).start;

  performance.mark('t2');
  performance.measure('t1 to t2', 't1', 't2');
  performance.measure('t0 to t2', 't0', 't2');
})();
```

Используем observer, чтобы посмотреть таймлайн

PerformanceObserver реагирует
на добавление в таймлайн новых PerformanceEntry

```
const {
  performance,
  PerformanceObserver } = require(‘perf_hooks’);

const Timer = require(‘timerpromise');

const obs = new PerformanceObserver((items) => {
  console.log(items.getEntries());
});

obs.observe({ entryTypes: ['mark', 'measure'] });

(async () => {
  performance.mark(‘t0');

  await new Timer(3).start;

  performance.mark('t1');
  performance.measure('t0 to t1', 't0', ‘t1');

  await new Timer(2).start;

  performance.mark('t2');
  performance.measure('t1 to t2', 't1', 't2');
  performance.measure('t0 to t2', 't0', 't2');
})();
```

Таймлайн

```
[ PerformanceEntry {
    name: 't0',
    entryType: 'mark',
    startTime: 89.609569,
    duration: 0 } ]
[ PerformanceEntry {
    name: 't1',
    entryType: 'mark',
    startTime: 3097.107409,
    duration: 0 } ]
[ PerformanceEntry {
    name: 't0 to t1',
    entryType: 'measure',
    startTime: 91.321641,
    duration: 3004.504551 } ]

//...
```

Используем observer, чтобы посмотреть только интервалы
Получаем только duration каждой записи типа measure

```
const {
  performance,
  PerformanceObserver } = require(‘perf_hooks’);

const Timer = require(‘timerpromise');

const obs = new PerformanceObserver((items) => {
  console.log(items.getEntries()[0].duration);
});

obs.observe({ entryTypes: ['measure'] });

(async () => {
  performance.mark(‘t0');

  await new Timer(3).start;

  performance.mark('t1');
  performance.measure('t0 to t1', 't0', ‘t1');

  await new Timer(2).start;

  performance.mark('t2');
  performance.measure('t1 to t2', 't1', 't2');
  performance.measure('t0 to t2', 't0', 't2');
})();
```

Итог

```
❯ node index.js
3004.815459
2010.499412
5015.314871
```

Ссылки

- Спецификации
  https://www.w3.org/TR/hr-time-2/#the-performance-interface
  https://w3c.github.io/performance-timeline/

- Документация (для node 10)
  https://nodejs.org/dist/latest-v10.x/docs/api/perf_hooks.html

- С чего всё началось
  https://medium.com/the-node-js-collection/timing-is-everything-6d43fc9fd416

- Исходный код
  https://github.com/nodejs/node/blob/master/lib/perf_hooks.js