«Напомни через минуту», или Как считать время в браузере

Никита Дубко, Яндекс.Поиск

Кто я?

- доброжелюбный бородач
- редактор новостей в Веб-стандартах
- пятый голос одноименного подкаста
- разработчик интерфейсов в Яндекс.Поиске

Задачи со временем

- ⏱️ замеры производительности;
- 🩺 медицинские таймеры;
- 📈 акции и котировки;
- ⚡ период полураспада цезия на АЭС.

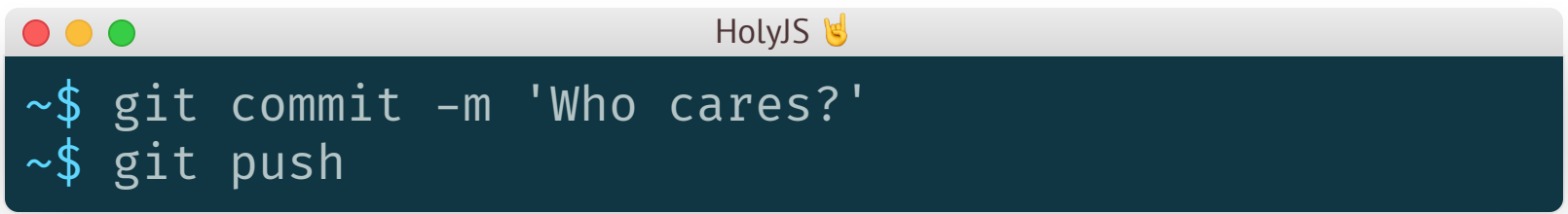
JavaScript?

- ⏱️ замеры производительности;
- ~~🩺 медицинские таймеры;~~
- ~~📈 акции и котировки;~~
- ~~⚡ период полураспада цезия на АЭС.~~

Задача 👇

показывать через заданное пользователем время попап с напоминалкой, что время прошло.

```
const ms = await getTime();
setTimeout(showPopup, ms);
```


HolyJS 🤘
~$ git commit -m 'Who cares?'
~$ git push

Спасибо за внимание!

Задача 👇

показывать **через заданное** пользователем **время** попап с напоминалкой, что время прошло.

Задача 👇

показывать через заданное пользователем время **попап** с напоминалкой, что время прошло.

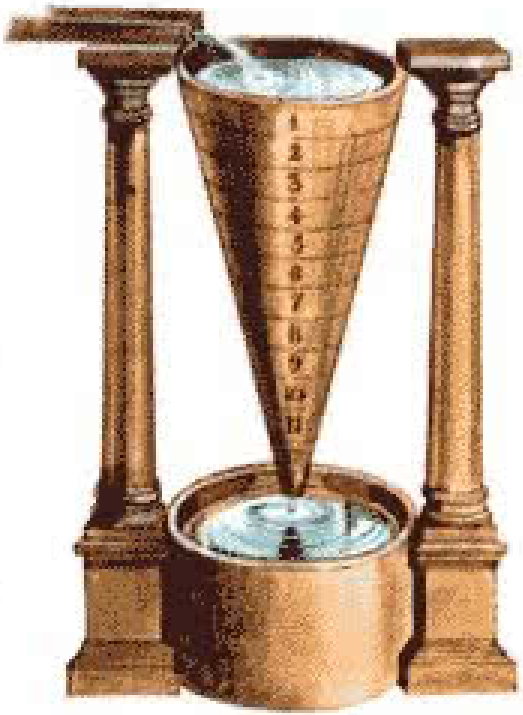
Алгоритм 🤓

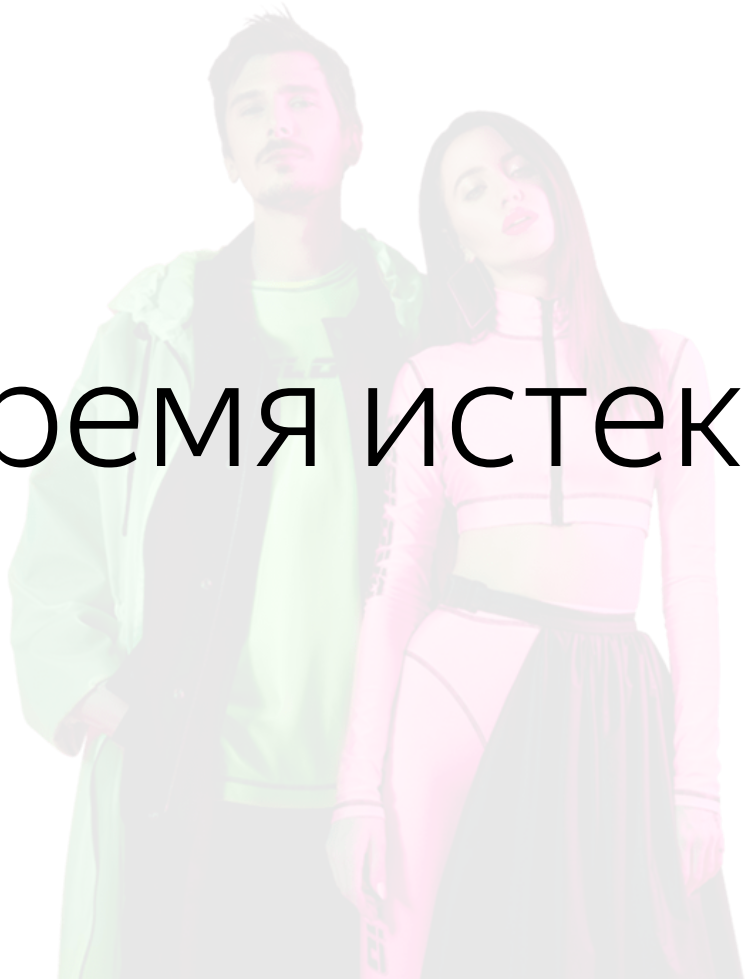
1. Получить текущее значение времени.
2. Отмерить промежуток времени от текущего значения.
3. Показать попап.

Глава 1

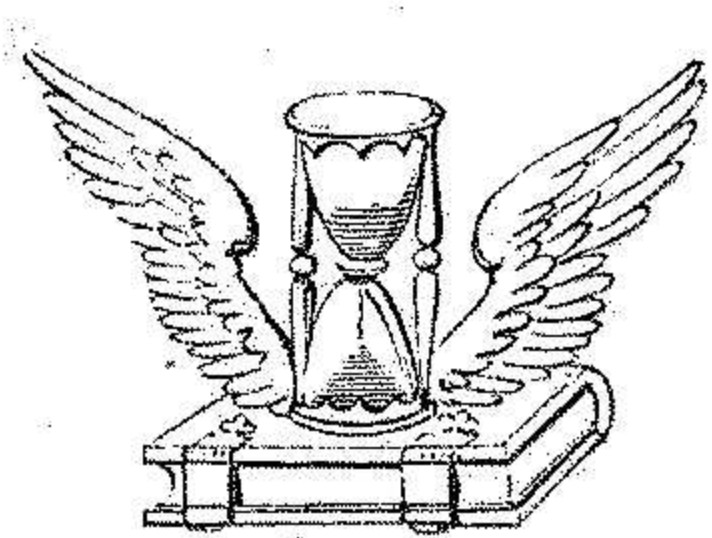
Время


GROW
ALONG
ME THE BEST
OLD
WITH
IS YET TO BE
GROW
ALONG
ME THE BEST
OLD
WITH
IS YET TO BE


1
2
3
4
5
6
7
8
9
10
11



«Время истекло»



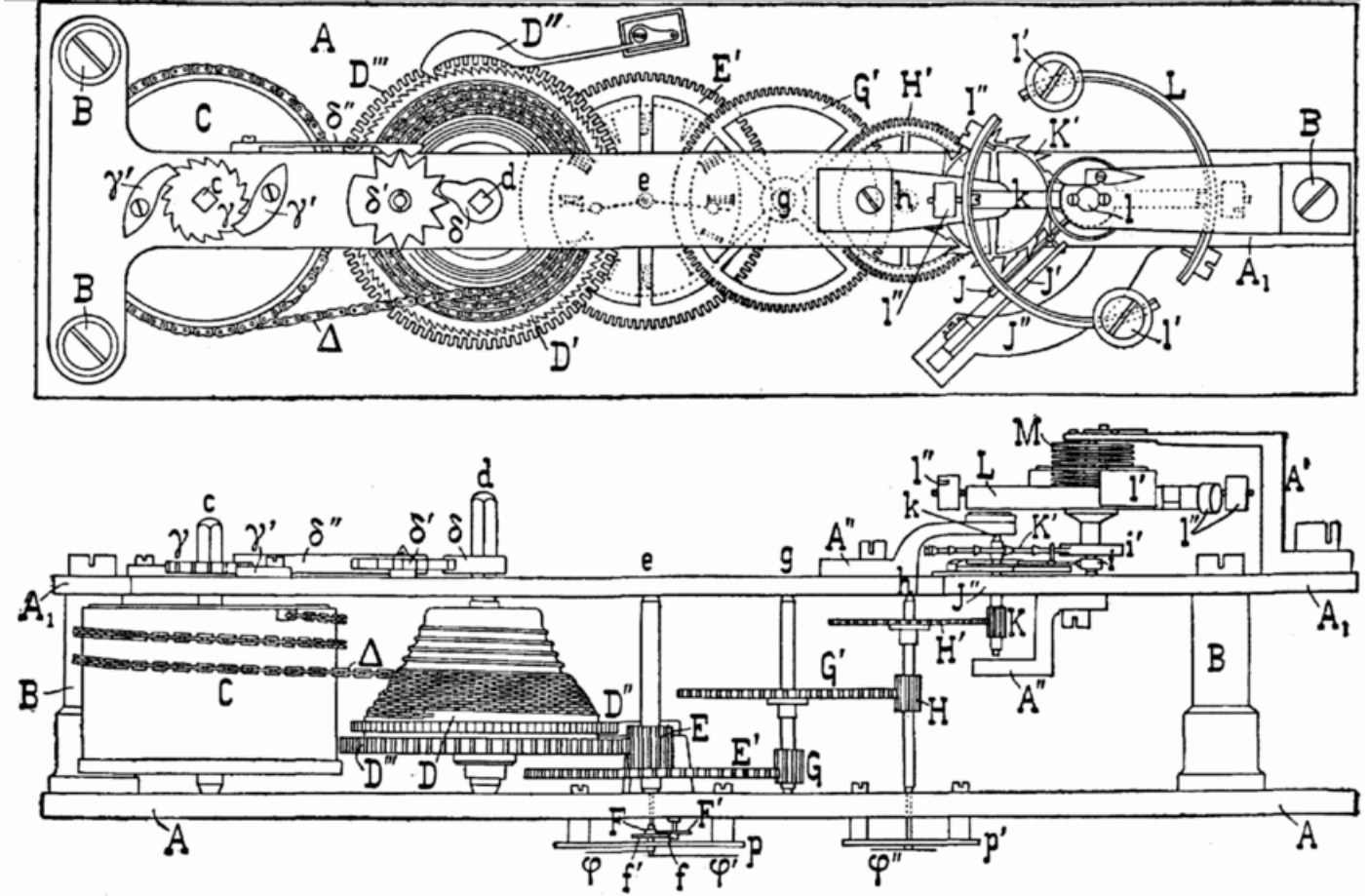
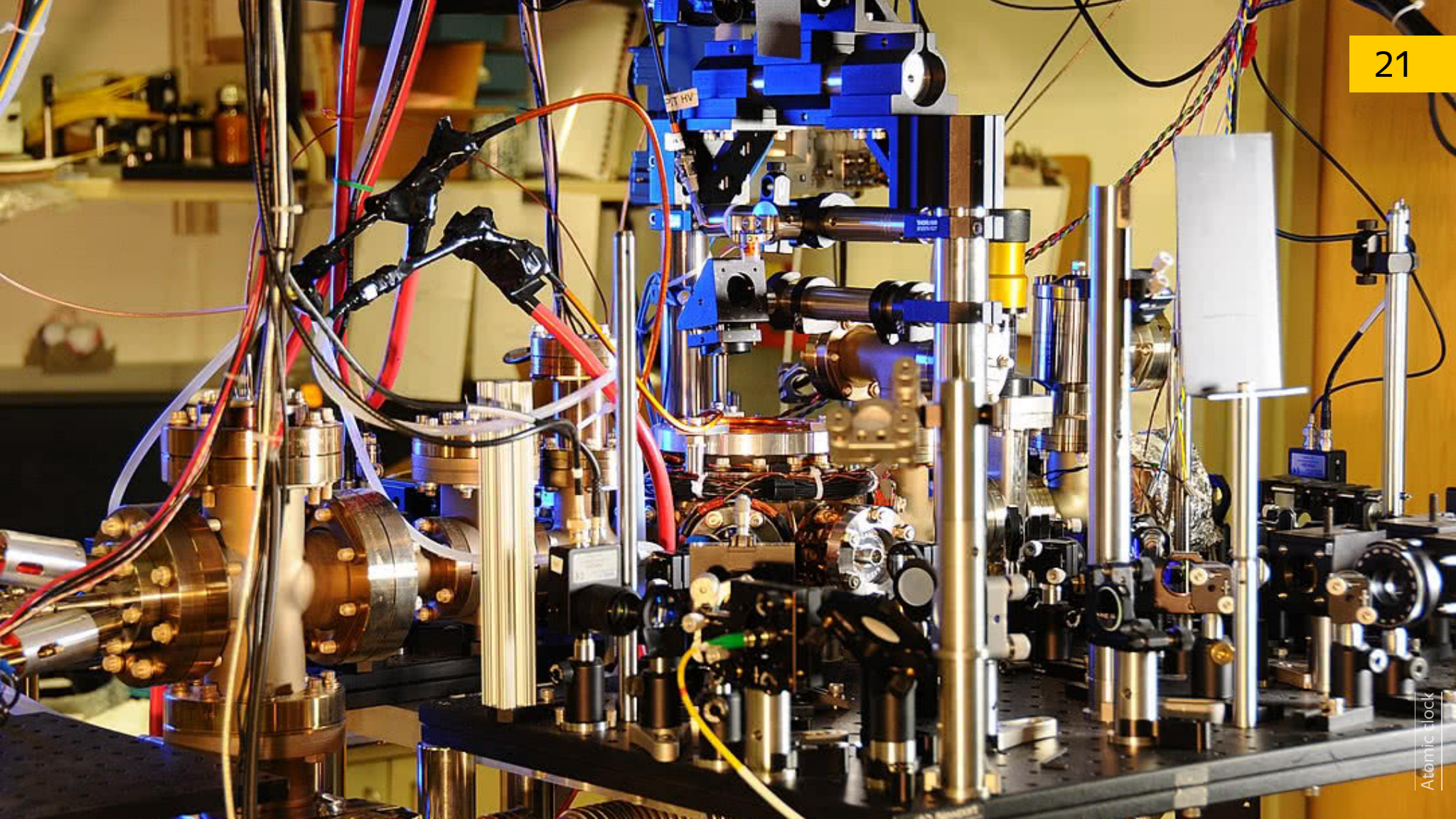


Fig. 1. *A* obere Platine. A_1 untere Platine. *A'* Brücke der Unruhe. *A''* Brücke der Hemmung. *B* Pfeiler der Platine. *C* Federhaus. *c* Federwelle. *γ* Sperrad für die Feder. *γ'* Sperrkegel. *D* Schnecke. *D'* Gegenſperrad. *D''* Sperrfeder für das Gegenſperrad. *D'''* Schneckenrad. *d* Schneckenwelle. *Δ* Kette. *δ* Zahn der Stellung. *δ'* Stellungsrad. *δ''* Stellungsfeder. *E* Trieb des Großbodenrads. *E'* Großbodenrad. *e* Achſe des Großbodenrads. *F* Minutenrohr. *F'* Wechſelrad. *f* Trieb des Wechſelrads. *f'* Stundenrad. *φ* Stundenzeiger. *φ'* Minutenzeiger. *φ''* Sekundenzeiger. *G* Trieb des Kleinbodenrads. *G'* Kleinbodenrad. *g* Achſe des Kleinbodenrads. *H* Trieb des Sekundenrads. *H'* Sekundenrad. *h* Achſe des Sekundenrads. *J* Hemmungsfeder. *J'* Anſchlagkloben für die Hemmungsfeder. *J''* Brücke der Hemmungsfeder. *i* Kleine Rolle. *i'* Große Rolle. *K* Trieb des Hemmungsrads. *K'* Hemmungsrad. *k* Steg der Unruhe. *L* Unruhe. *l* Achſe der Unruhe. *l'* Kompenſationsgewichte. *l''* Regulierſchrauben. *M* Spirale.


JAPAN
00234
SEIKO
35A



 GPS

1 секунда

— время, равное 9 192 631 770 периодам излучения, соответствующего переходу между двумя сверхтонкими уровнями основного состояния атома цезия-133.

JavaScript 🙃

setTimeout

⚠️ Не часть ECMAScript

`setTimeout(callback[, delay[, ...args]])`

Added in: v0.0.1

- `callback` `<Function>` The function to call when the timer elapses.
- `delay` `<number>` The number of milliseconds to wait before calling the `callback`. **Default:** `1`.
- `...args` `<any>` Optional arguments to pass when the `callback` is called.
- Returns: `<Timeout>` for use with `clearTimeout()`

Schedules execution of a one-time `callback` after `delay` milliseconds.

🇬🇧 When delay is larger than 2147483647 or less than 1, the delay will be set to 1. Non-integer delays are truncated to an integer.

🇷🇺 Когда задержка больше 2147483647 или меньше 1, значение задержки устанавливается в 1. Не-целые задержки обрезаются до целых.

2147483647 секунд

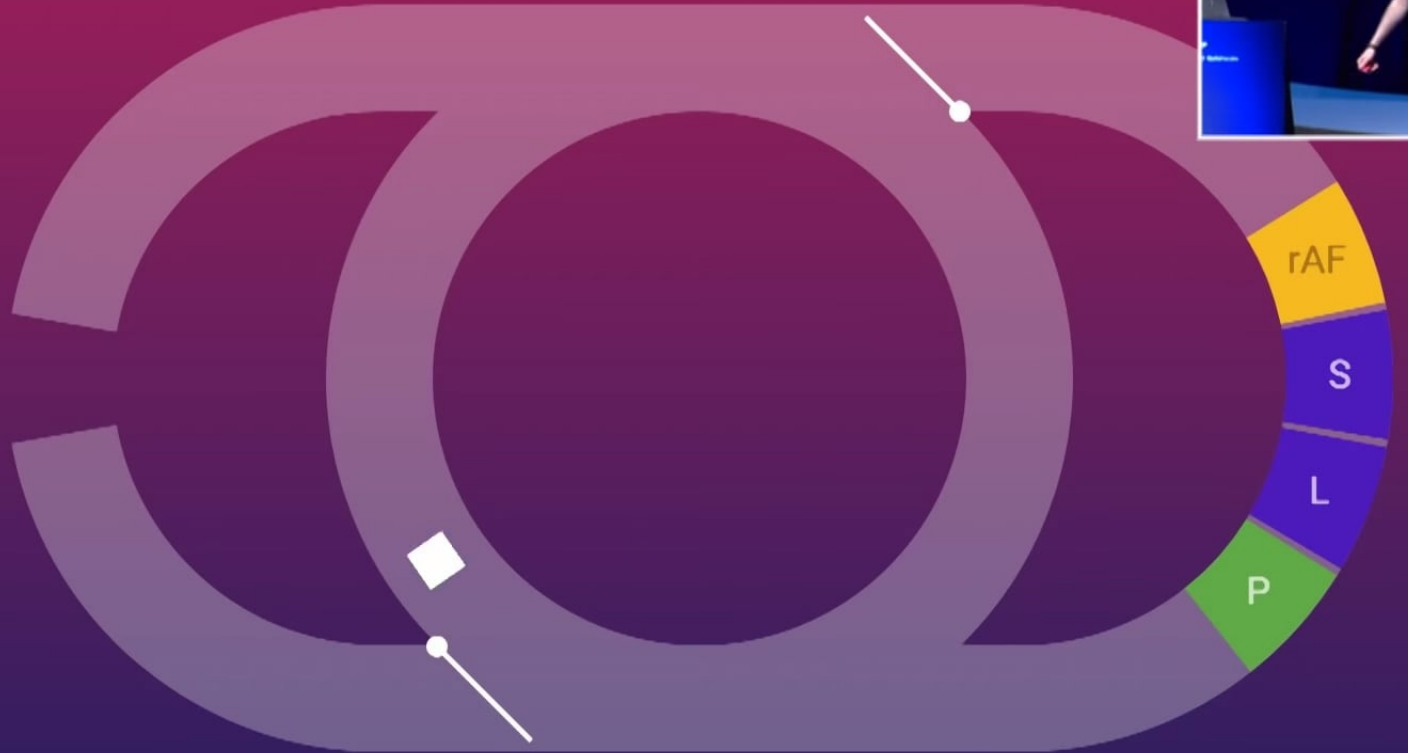
=

24,8551348 дня

```
void Shell::SetTimeout(const v8::FunctionCallbackInfo<v8::Value>& args) {
    Isolate* isolate = args.GetIsolate();
    args.GetReturnValue().Set(v8::Number::New(isolate, 0));
    if (args.Length() == 0 || !args[0]->IsFunction()) return;
    Local<Function> callback = args[0].As<Function>();
    Local<Context> context = isolate->GetCurrentContext();
    PerIsolateData::Get(isolate)->SetTimeout(callback, context);
}

void PerIsolateData::SetTimeout(Local<Function> callback,
    Local<Context> context) {
    set_timeout_callbacks_.emplace(isolate_, callback);
    set_timeout_contexts_.emplace(isolate_, context);
}
```

```
bool Shell::CompleteMessageLoop(Isolate* isolate) {
    auto get_waiting_behaviour = [isolate]() {
        base::MutexGuard guard(isolate_status_lock_.Pointer());
        DCHECK_GT(isolate_status_.count(isolate), 0);
        bool should_wait = (options.wait_for_background_tasks &&
                            isolate->HasPendingBackgroundTasks()) ||
                            isolate_status_[isolate] ||
                            isolate_running_streaming_tasks_[isolate] > 0;
        return should_wait ? platform::MessageLoopBehavior::kWaitForWork
                           : platform::MessageLoopBehavior::kDoNotWait;
    };
    return ProcessMessages(isolate, get_waiting_behaviour);
}
```

```
while (true) {
    // кручу-верчу, таски выполнить хочу
    // ...

    bool ran_set_timeout = false;
    if (!RunSetTimeoutCallback(isolate, &ran_set_timeout)) {
        return false;
    }

    if (!ran_set_timeout) return true;
}
```



@jaffathecake

Node.js 15.0.0

```
const { setTimeout } = require('timers/promises');

async function myPromise(delay) {
    await setTimeout(delay);
    return new Promise((resolve) => {
        resolve({
            data: `The data from ${delay} ms delay`,
        });
    });
}
```

⚠️ Stability: Experimental

Date

✅ Стандарт

ECMAScript® Language Specification

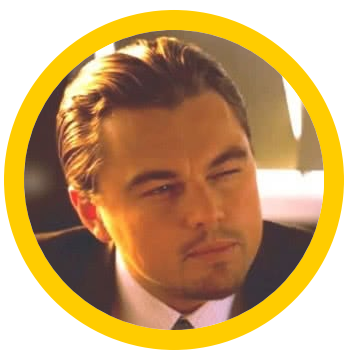
```
const currentDate = new Date();
const timestamp = currentDate.getTime();

// OR (IE9+)

const timestamp = Date.now();
```

```
double JSDate::CurrentTimeValue(Isolate* isolate) {
    if (FLAG_log_internal_timer_events) LOG(isolate, CurrentTimeEvent());
    if (FLAG_correctness_fuzzer_suppressions) return 4.2;

    // According to ECMA-262, section 15.9.1, page 117, the precision of
    // the number in a Date object representing a particular instant in
    // time is milliseconds. Therefore, we floor the result of getting
    // the OS time.
    return std::floor(V8::GetCurrentPlatform()->CurrentClockTimeMillis());
}
```

```
double DefaultPlatform::CurrentClockTimeMillis() {
    return base::OS::TimeCurrentMillis();
}

double OS::TimeCurrentMillis() {
    return Time::Now().ToJsTime();
}
```

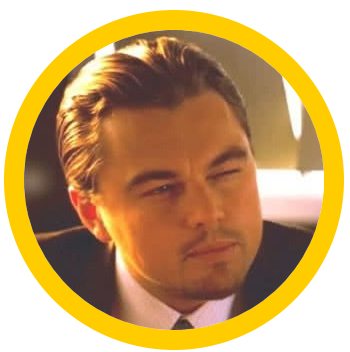
```
// WIN

Time Now() {
    // …

    // Determine current time and ticks.
    TimeTicks ticks = GetSystemTicks();
    Time time = GetSystemTime();

    // …
}
```

```
// POSIX

Time Time::Now() {
    struct timeval tv;
    int result = gettimeofday(&tv, nullptr);
    DCHECK_EQ(0, result);
    USE(result);
    return FromTimeval(tv);
}
```



DateTime under the hood, Andrey Akinshin

```
; gettimeofday

0x00000034f408c2d4 : mov    $0xffffffffff600000,%rax
0x00000034f408c2db : callq  *%rax
```



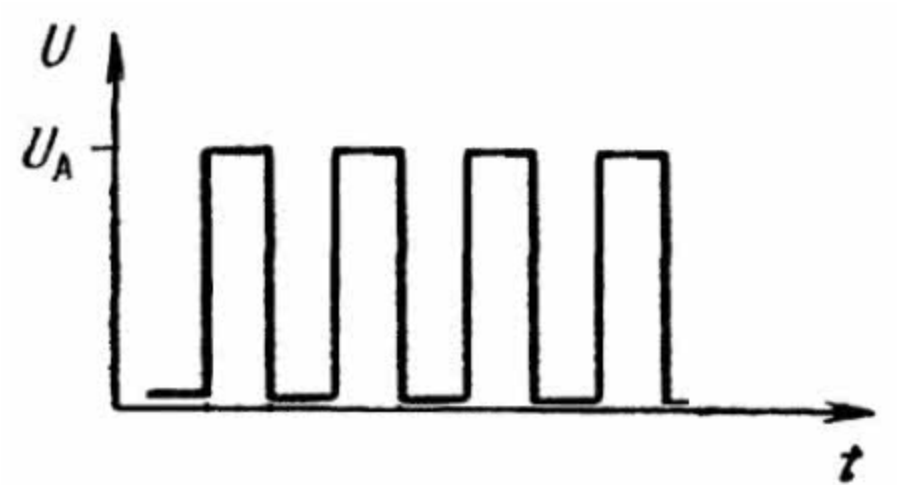
vsyscall, vDSO

System calls in the Linux kernel. Part 3.

System Time

Спикер может наврать, потому что не является дипломированным сварщиком. Воспринимайте фантазии спикера с разумной долей критики. Доклад не является публичной

офертой. Вы видите свой нос прямо сейчас.


U
U_A
t


16.000

10 МГц

=

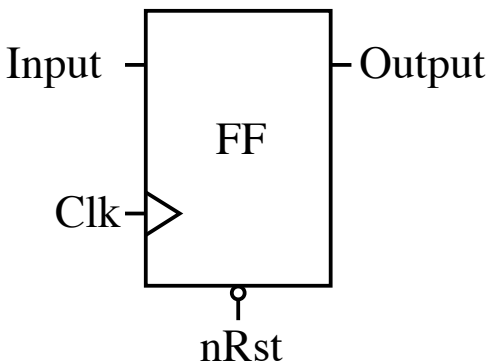
10 000 000
колебаний в секунду

10 000 000 колебаний

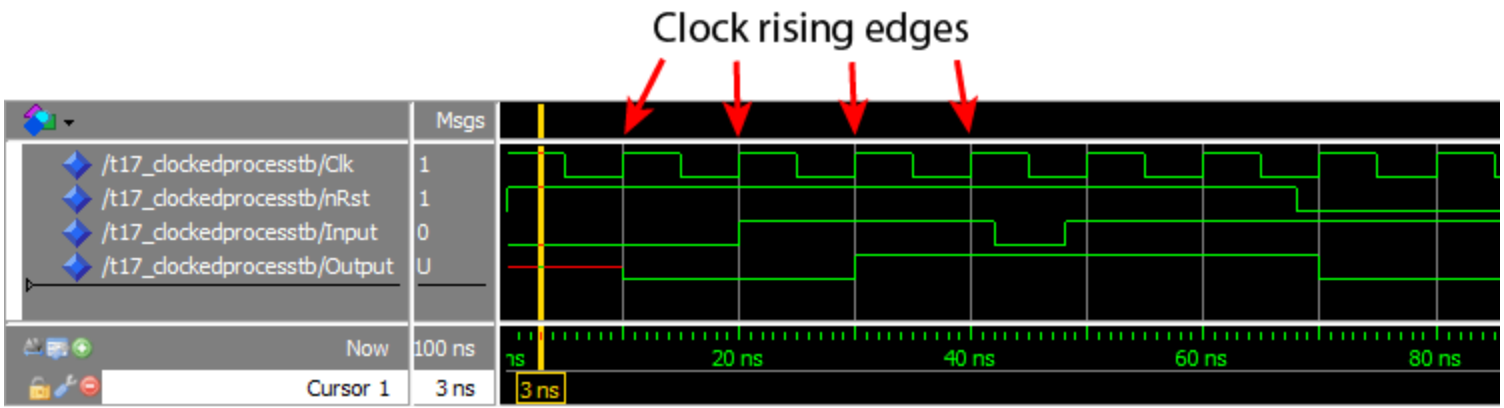
=

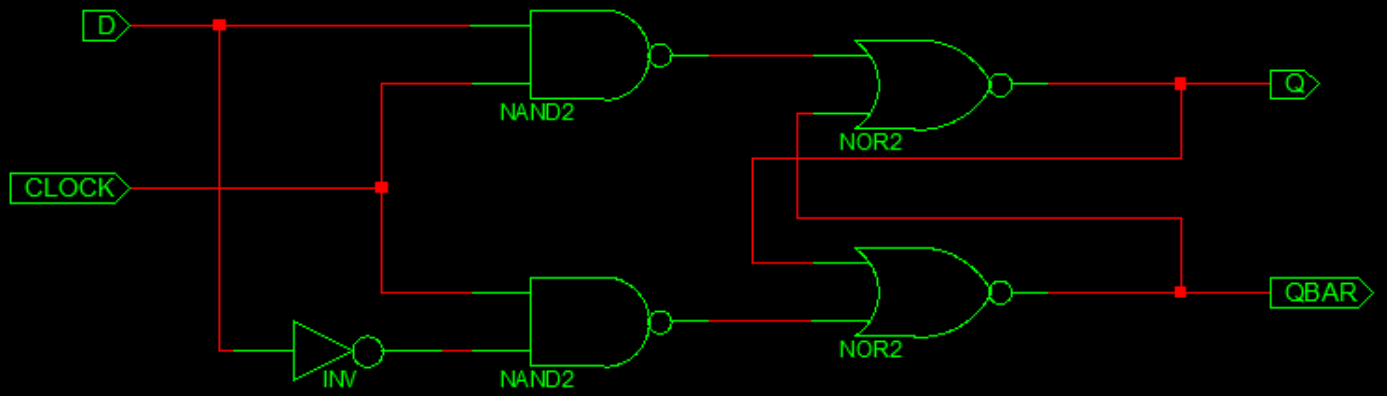
1 секунда

 VHDL


Input
Output
FF
Clk
nRst

```
entity D_FF is
    PORT(
        Input, Clk: in std_logic;
        Output: out std_logic
    );
end D_FF;

architecture behavioral of D_FF is
begin
    process(Clk)
    begin
        if(Clk='1' and Clk'EVENT) then
            Output <= Input;
        end if;
    end process;
end behavioral;
```

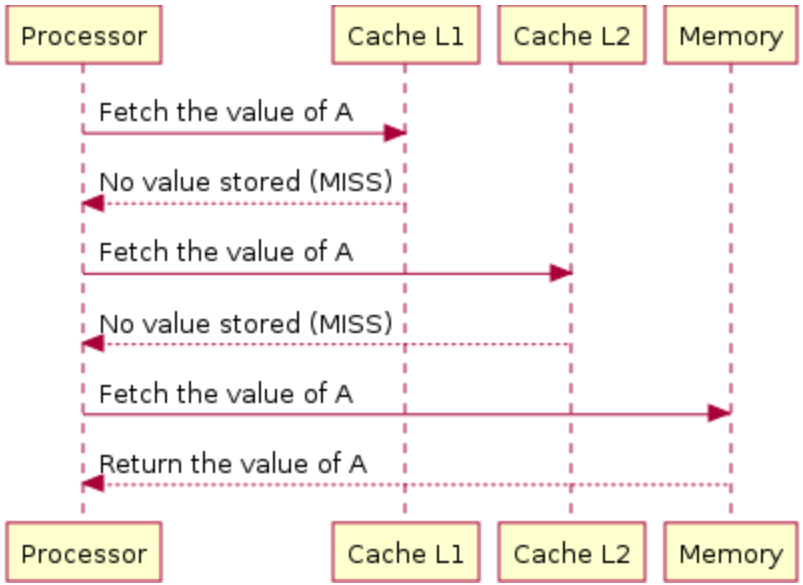

Clock rising edges
Msgs
/t17_clockedprocesstb/Clk
/t17_clockedprocesstb/nRst
/t17_clockedprocesstb/Input
/t17_clockedprocesstb/Output
1
1
0
U
Now
100 ns
Cursor 1
3 ns
ns
20 ns
40 ns
60 ns
80 ns
3 ns


D
CLOCK
NAND2
INV
NAND2
NOR2
NOR2
Q
QBAR

Реактивное программирование ⚡

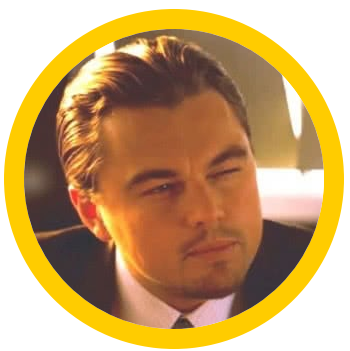
over**clock**ing 

Замеры времени — быстрые 🚀


Processor
Cache L1
Cache L2
Memory
Fetch the value of A
No value stored (MISS)
Fetch the value of A
No value stored (MISS)
Fetch the value of A
Return the value of A
Processor
Cache L1
Cache L2
Memory

Браузерное расширение

Выныриваем 🤿



Задача 👇

показывать **через заданное** пользователем **время** попап с напоминалкой, что время прошло.

```
const start = Date.now();
const timeout = 15000;

function check() {
    const now = Date.now();
    if (now - start >  timeout) {
        showPopup();
    } else {
        setTimeout(check, 1);
    }
}

setTimeout(check, 1);
```

```
function sleep(milliseconds) {
    const date = Date.now();
    let currentDate = null;
    do {
        currentDate = Date.now();
    } while (currentDate - date < milliseconds);
}

sleep(15000);
```

Web Worker

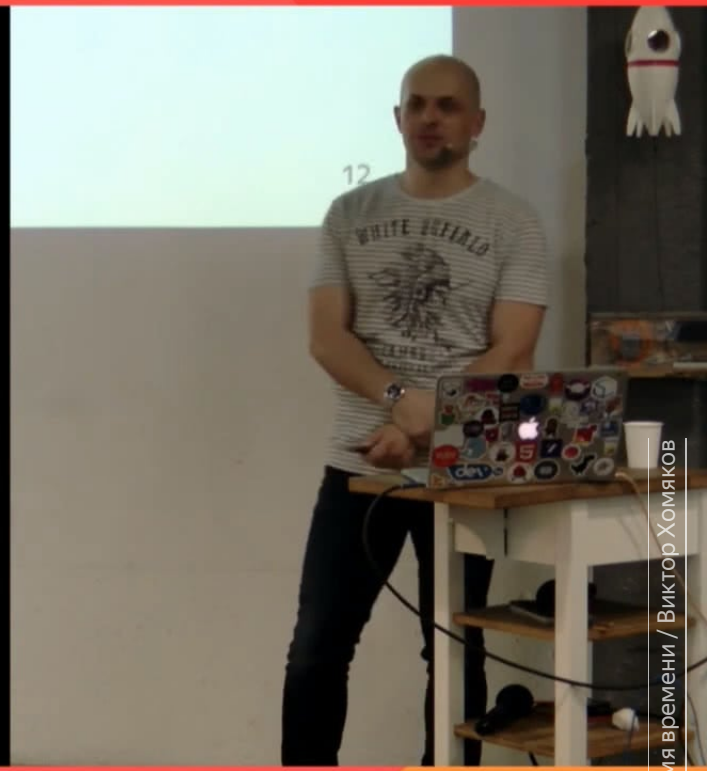
```
// worker.js

let time = Date.now();
const start = time;

while (true) {
    if (Date.now() - start > 15000) {
        postMessage('killme');
        break;
    }
}
```

```
// main.js

function go() {
    const worker = new Worker('/worker.js');

    worker.onmessage = message => {
        if (message.data === 'killme') {
            worker.terminate();
        }
    };
}
```

Логируем транзакции

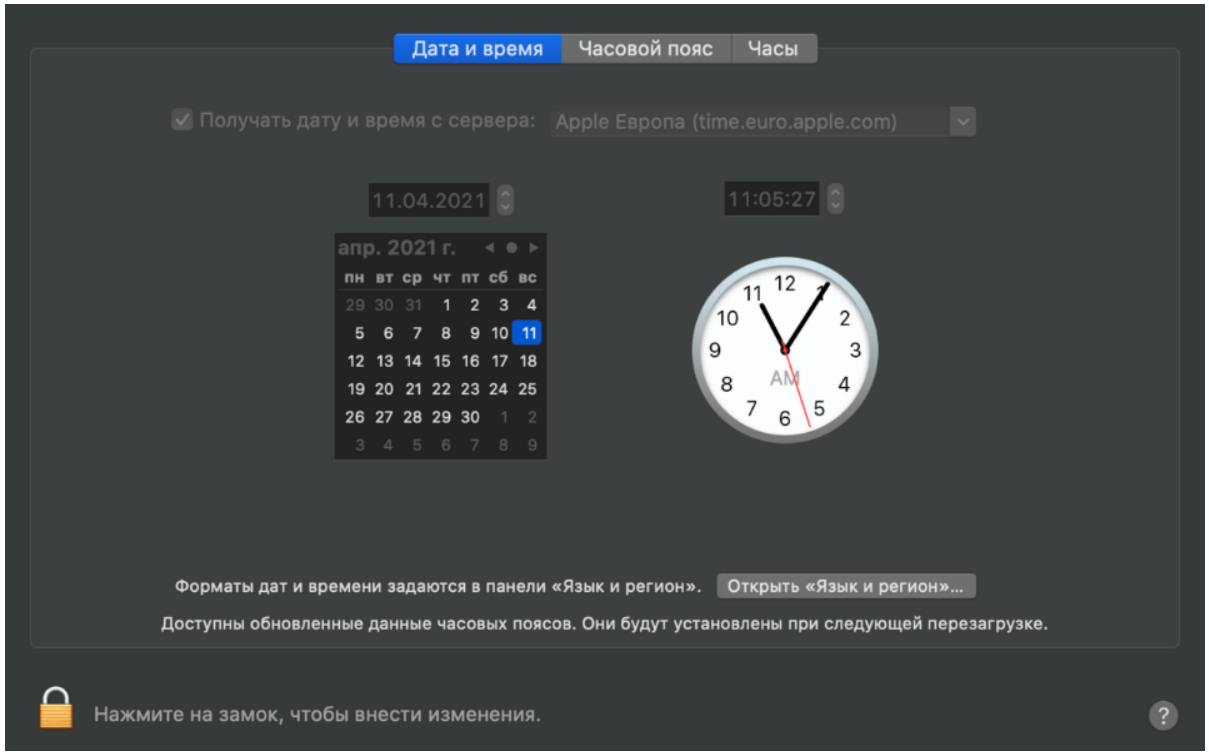
```
const start = Date.now();                          // 1550000000000
transaction();
const finish = Date.now();                         // 1549999999000
log(`Транзакция длилась ${finish - start}мс`); // Транзакция длилась -1000мс
```

12



NTP Daemon

Network Time Protocol (NTP) daemon


Дата и время
Часовой пояс
Часы
Получать дату и время с сервера:
Apple Европа (time.euro.apple.com)
11.04.2021
11:05:27
апр. 2021 г.
пн вт ср чт пт сб вс
29 30 31 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 1 2
3 4 5 6 7 8 9
AM
Форматы дат и времени задаются в панели «Язык и регион».
Открыть «Язык и регион»...
Доступны обновленные данные часовых поясов. Они будут установлены при следующей перезагрузке.
Нажмите на замок, чтобы внести изменения.

Страны и территории, применяющие летнее время [править | править код]

Приведены сведения по состоянию на 2019 год — 63 государства и 10 территорий[55].

- Австралия[56]
- Австрия
- Аландские острова
- Албания
- Андорра
- Багамские Острова
- Бельгия
- Бермуды
- Болгария
- Босния и Герцеговина
- Ватикан
- Великобритания
- Венгрия
- Гаити
- Германия
- Гернси
- Гибралтар
- Государство Палестина
- Гренландия[56]
- Греция
- Дания
- Джерси
- Израиль
- Иордания
- Иран
- Ирландия
- Испания
- Италия
- Канада[56]
- Кипр
- Республика Косово
- Куба
- Латвия
- Ливан
- Литва
- Лихтенштейн
- Люксембург
- Северная Македония
- Мальта
- Мексика[56]
- Молдавия
- Монако
- Нидерланды
- Новая Зеландия
- Норвегия
- Остров Мэн
- Парагвай
- Польша
- Португалия
- Румыния
- САДР
- Самоа
- Сан-Марино
- Сен-Пьер и Микелон
- Сербия
- Сирия
- Словакия
- Словения
- США[56]
- Теркс и Кайкос
- Тонга
- Украина[56]
- Фарерские острова
- Фиджи
- Финляндия
- Франция
- Хорватия
- Черногория
- Чехия
- Чили[56]
- Швейцария
- Швеция
- Эстония

Daylight saving time

performance

```
void Shell::PerformanceNow(const v8::FunctionCallbackInfo<v8::Value>& args) {
    if (i::FLAG_verify_predictable) {
        args.GetReturnValue().Set(g_platform->MonotonicallyIncreasingTime());
    } else {
        base::TimeDelta delta =
            base::TimeTicks::HighResolutionNow() - kInitialTicks;
        args.GetReturnValue().Set(delta.InMillisecondsF());
    }
}
```

```
DEFINE_BOOL(verify_predictable, false,
    "this mode is used for checking that V8 behaves predictably")


const base::TimeTicks Shell::kInitialTicks =
    base::TimeTicks::HighResolutionNow();


TimeTicks TimeTicks::HighResolutionNow() {
    return TimeTicks::Now();
}
```

```
// We use timeGetTime() to implement TimeTicks::Now().  This can be problematic
// because it returns the number of milliseconds since Windows has started,
// which will roll over the 32-bit value every ~49 days.  We try to track
// rollover ourselves, which works if TimeTicks::Now() is called at least every
// 48.8 days (not 49 days because only changes in the top 8 bits get noticed).
```

QPC — QueryPerformanceCounter

QueryPerformanceCounter function

DOMHighResTimeStamp

DOMHighResTimeStamp

The time, given in milliseconds, should be accurate to 5 μs (microseconds), with the fractional part of the number indicating fractions of a millisecond.

To offer protection against timing attacks and fingerprinting, the precision of time stamps might get rounded depending on browser settings. In Firefox, the privacy.reduceTimerPrecision preference is enabled by default and defaults to 20 µs in Firefox 59; in 60 it will be 2ms.

In Firefox, you can also enable `privacy.resistFingerprinting`, the precision will be 100ms or the value of `privacy.resistFingerprinting.reduceTimerPrecision.microseconds`, whichever is larger.

```
// reduced time precision (2ms) in Firefox 60
event.timeStamp
// 1519211809934
// 1519211810362
// 1519211811670
// ...

// reduced time precision with `privacy.resistFingerprinting`
event.timeStamp;
// 1519129853500
// 1519129858900
// 1519129864400
// ...
```

requestAnimationFrame

The callback function is passed one single argument, a **DOMHighResTimeStamp** similar to the one returned by `performance.now()`, indicating the point in time when `requestAnimationFrame()` starts to execute callback functions.

2012 — requestAnimationFrame API:

now with sub-millisecond precision

```
FrameRequestCallbackCollection::CallbackId
FrameRequestCallbackCollection::RegisterFrameCallback(FrameCallback* callback) {
    FrameRequestCallbackCollection::CallbackId id = ++next_callback_id_;
    callback->SetIsCancelled(false);
    callback->SetId(id);
    frame_callbacks_.push_back(callback);

    DEVTOOLS_TIMELINE_TRACE_EVENT_INSTANT("RequestAnimationFrame",
                                          inspector_animation_frame_event::Data,
                                          context_, id);
    probe::AsyncTaskScheduledBreakable(context_, "requestAnimationFrame",
                                       callback->async_task_id());
    return id;
}
```

```
const timeout = 15000;
let start;

function check(timestamp) {
    if (start === undefined) {
        start = timestamp;
    }
    const elapsed = timestamp - start;

    if (elapsed < timeout) {
        window.requestAnimationFrame(check);
    } else {
        showPopup();
    }
}

window.requestAnimationFrame(check);
```

Как выбрать?

❓ Нужно точно?

✅ performance.now, requestAnimationFrame.

❓ Точность не важна?

✅ Date.now, setTimeout.

❓ Нужен контроль?

✅ Сервер

Long polling

WebSockets

Экзотика 

CSS-анимации

```
.animation.active {
    animation-duration: 1s;
    animation-name: some;
    animation-iteration-count: 1000;
}

@keyframes some {
    from { opacity: 0 }
    to { opacity: 1 }
}
```

```
const animation = document.querySelector('.animation');
let iterationCount = 0;

animation.addEventListener('animationstart', () => { });

animation.addEventListener('animationiteration', () => {
    iterationCount++;
});

animation.addEventListener('animationend', () => { });

animation.classList.toggle('active');
```

Web Animations API

```
function ownSetTimeout(callback, duration) {
    const div = document.createElement('div');
    const keyframes = new KeyframeEffect(div, [], {
        duration,
        iterations: 1
    });
    const animation = new Animation(
        keyframes,
        document.timeline
    );

    animation.play();

    animation.addEventListener('finish', () => {
        callback();
    });
}
```

IE	Edge *	Firefox	Chrome	Safari	Safari on iOS *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			88						
	89	86	89						
11	90	87	90	[5] 14	[5] 14.5	all	89	[2] 12.12	[2] 13.0
		88	91	[5] TP					
		89	92						
			93						

Видео

```
var video = document.createElement('video');
await video.play();

// wait …
console.log(video.currentTime);
```

Глава 2

Уведомление

Браузер экономный 

Оптимизации в фоне 🔧

Policies in place to aid background page performance

Page Lifecycle API

Состояния страниц

- Active 😀
- Passive 😶
- Hidden 🙈
- Frozen 🥶
- Terminated 🤖
- Discarded ❌

Page Freezing Opt-In and Opt-Out

Heuristics for Freezing & Discarding

Не фризит, если ...

- Играет аудио
- Используется WebRTC
- Обновляется title или фавиконка 🙃
- Показывается alert
- Отправляется пуш-нотификация

Аудио 

```
const audio = new Audio('path/to/sound');

if (typeof audio.loop == 'boolean') {
    audio.loop = true;
} else {
    myAudio.addEventListener('ended', function() {
        this.currentTime = 0;
        this.play();
    }, false);
}

audio.play();
```

Заблокировано по умолчанию ❌

Wake Lock API

```
let wakeLock = null;
const requestWakeLock = async () => {
    try {
        wakeLock = await navigator.wakeLock.request();
    } catch (err) {
        console.error(`${err.name}, ${err.message}`);
    }
};

await requestWakeLock();
window.setTimeout(() => {
    wakeLock.release();
    wakeLock = null;
}, 5000);
```

```
const handleVisibilityChange = async () => {
    if (wakeLock !== null &&
        document.visibilityState === 'visible') {
        await requestWakeLock();
    }
};

document.addEventListener(
    'visibilitychange',
    handleVisibilityChange
);
```

IE	Edge *	Firefox	Chrome	Safari	Safari on iOS *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			88						
	1 89	86	89						
11	1 90	87	90	14	14.5	all	89	12.12	13.0
		88	91	TP					
		89	92						
			93						

Push Notifications

Web USB API

Глава 3

Будущее

Currently `performance.now()` and related timestamps are coarsened based on site isolation status. This change will align their coarsening based on cross-origin isolation capability, regardless of platform. That would decrease their resolution on desktop from 5 microseconds to 100 microseconds in non-isolated contexts. It would also increase their resolution on Android from 100 microseconds to 5 microseconds in cross-origin isolated contexts, where it's safe to do so.

mozilla / standards-positions

Watch 105 Star 274 Fork 42

Code Issues 110 Pull requests 3 Actions Projects Security Insights

Aligning high-resolution timer granularity to cross-origin isolated capability #502

New issue

Open yoavweiss opened this issue on Mar 17 · 1 comment

yoavweiss commented on Mar 17 • edited

Request for Mozilla Position on an Emerging Web Specification

- Specification Title: HR-Time
- Specification or proposal URL: https://w3c.github.io/hr-time/#dfn-current-high-resolution-time
- Caniuse.com URL (optional):
- Bugzilla URL (optional):
- Mozillians who can provide input (optional): **@annevk**, **@tomrittervg**

Other information

We recently changed the HR-time spec to better align its resolution clamping with cross-origin isolated capability.

For Chromium, that means that it would be reducing its resolution in non-isolated contexts (regardless of the platform's site-isolation status) to 100 microseconds, and increasing it in cross-origin isolated contexts (even in platforms without site-isolation, e.g. Android) to 5 microseconds.

Assignees

No one assigned

Labels

w3c

Projects

None yet

Milestone

No milestone

Linked pull requests

Successfully merging a pull request may close this issue.

Aligning high-resolution timer granularity to cross-origin isolated capability

Lock Screen API

Chrome Status: Intent to Prototype 🔧

Temporal 😍


EN / День 3 / 10:35 / Зал 1

Temporal.now

```
const timeStamp = Temporal.now.instant();

// Timestamp in Milliseconds
timeStamp.epochMilliseconds;

const duration = Temporal.Duration.from({
    hours: 130,
    minutes: 20
});

duration.total({ unit: 'seconds' }); // => 469200
```

Status?

Stage 3 

«Вебня»

Напоследок

Time you enjoy wasting,
was not wasted.

John Lennon

Спасибо за внимание!

mefody.dev/talks/wait-for-minute/

@dark_mefody

n.a.dubko@gmail.com