

# Странные АПИ браузера - вдруг да пригодятся?

А может быть и нет

# Кто я?

Меня зовут **Антон Ленёв**

- СТО в стартапе Viewst
- В разработке с 2008 года



# Зачем туда полез?

Может возникнуть вопрос, как я полез в эту тему?

WYSIWYG редактор рекламных креативов ответ

- нужно работать с цветом удобнее (спойлер - EyeDropper)
- эффективнее работать с pooling'ом данных
- улучшение автосохранения работы пользователя
- и прочие навороты что можно выжать

# На чем будем показывать примеры?

- Фронт приложения для продажи билетов на барную подписку на пиво и мероприятия

Но чтобы было интересней, мы будем стараться сделать приложение **коварным, злым и надоедливым** с использованием разных интересных API

Код примеров и материалы



[https://github.com/m0rg0t/unusual\\_web\\_api\\_and\\_examples](https://github.com/m0rg0t/unusual_web_api_and_examples)

# Страница с примером

[https://m0rg0t.github.io/unusual\\_web\\_api\\_and\\_examples/](https://m0rg0t.github.io/unusual_web_api_and_examples/)



# Battery Status API

Что мы можем узнать:

- уровень заряда аккумулятора (от 0 до 1.0)
- заряжается ли аккумулятор
- сколько секунд осталось до зарядки
- сколько секунд осталось до разрядки



# Battery Status API - где пригодится?


- Чаще автосохранять данные чтобы пользователь не потерял свою работу (заполнение формы, редактирование чего-либо)
- Предупреждать перед запуском длительной задачи
- Минимизировать “прожорливые” задачи
- ...
- PROFIT!

# Battery Status API - пример

В примере мы будем делать злобно-коварные вещи, поэтому на странице покупки подписки на пиво мы будем повышать цену, если заряд устройства кончается и устройство не заряжается

## Battery Status API - CR

Method to provide information about the battery status of the hosting device.

Current aligned Usage relative Date relative Filtered All 

Chrome	Edge *	Safari	Firefox	Opera	IE	Chrome for Android	Safari on iOS *	Samsung Internet	Opera Mini *	Opera Mobile *	UC Browser for Android	Android Browser *	Firefox for Android	QQ Browser	Baidu Browser	KaiOS Browser
			2-9													
			10-15													
			16-42													
4-36			43-51	10-24												
37	12-18															
38-114	79-114	3.1-16.4	52-115	25-100	6-10											
115	115	16.5	116	101	11	115	16.5	21	all	73	15.5	115	115	13.1	13.18	2.5
116-118		16.6-TP	117-119				16.6-17									

Usage % of all users ?  
Global 73.59% + 0.01% = 73.6%  
unprefixed: 73.59%





```
1 const initBatteryWork = (price) => {
2     if (navigator.getBattery) {
3         navigator.getBattery().then((battery) => {
4             updatePriceInDomFromBattery(price, battery);
5
6             battery.addEventListener("chargingchange", () =>
7 {
8             updatePriceInDomFromBattery(price, battery);
9
10            battery.addEventListener("levelchange", () => {
11                updatePriceInDomFromBattery(price, battery);
12            });
13        });
14    } else {
15        console.log("Battery status API unavailable");
16    }
17 }
```

# Web Share API

Что мы можем сделать:

- поделиться текстом, ссылкой, картинкой или файликом нативным механизмом твоей операционки

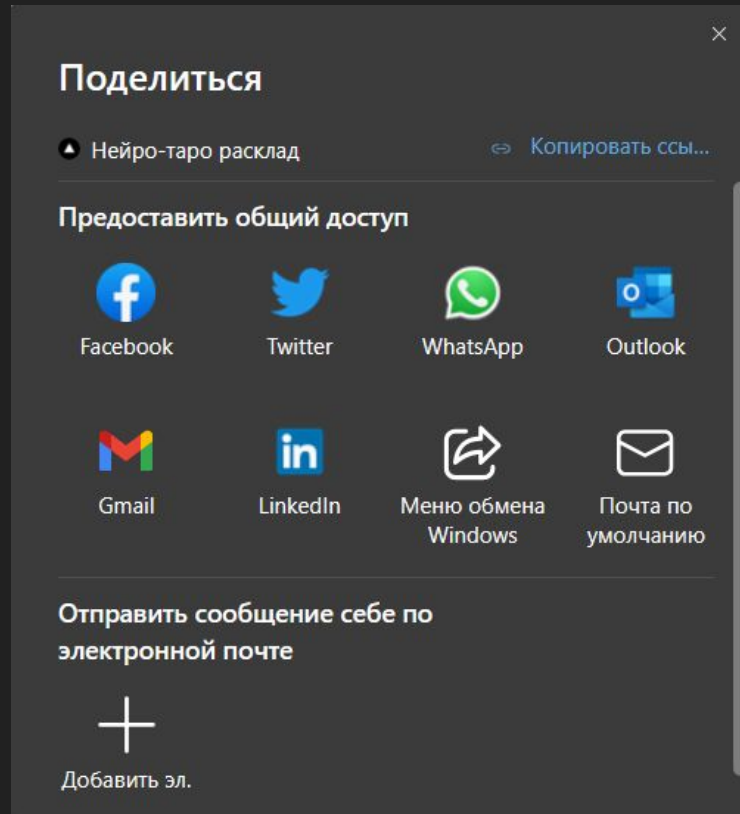
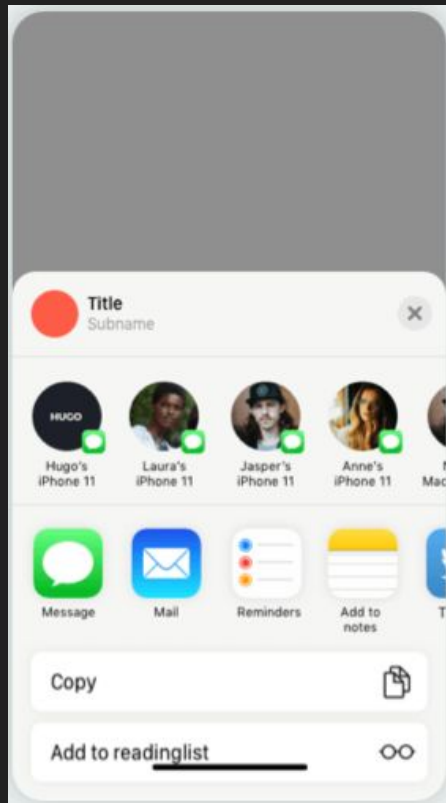


[https://developer.mozilla.org/en-US/docs/Web/API/Web\\_Share\\_API](https://developer.mozilla.org/en-US/docs/Web/API/Web_Share_API)

# Web Share API - как выглядит

В зависимости от системы  
выглядит по разному  
конечно же, но думаю со  
своим sharing меню вы  
вполне знакомы

(кроме винды, где вы такое  
видели вообще)







```
1 const shareTicket = async (ticketUrl) => {
2   const shareData = {
3     title: "Пивная подписка - купи и ты!",
4     text: "Подписку на пиво купили, а ты не купил? Купи!",
5     url: ticketUrl,
6   };
7
8   if (navigator.canShare) {
9     try {
10      await navigator.share(shareData);
11    } catch (err) {
12      console.log(`Error: ${err}`);
13    }
14  } else {
15    console.log(`Your browser doesn't support the Web Share
16 API.`);
17 }
```

# Page visibility API

Что мы можем узнать:

- видим или нет наш документ

Для чего:

- выключить звук
- не пулить или не пушить данные с\на сервер
- и прочие полезные штуки







```
1 const initPageVisibility = (callback) => {  
2   const visibilityChange = () => {  
3     if (document.hidden) {  
4       console.log('Page is hidden');  
5       callback && callback(false);  
6     } else {  
7       console.log('Page is visible');  
8       callback && callback(true);  
9     }  
10  }  
11  
12  document.addEventListener('visibilitychange', visibilityChange,  
13 false);
```





```
1 const initPageVisibility = (callback) => {  
2   const visibilityChange = () => {  
3     if (document.hidden) {  
4       console.log('Page is hidden');  
5       callback && callback(false);  
6     } else {  
7       console.log('Page is visible');  
8       callback && callback(true);  
9     }  
10  }  
11  
12  document.addEventListener('visibilitychange', visibilityChange,  
13 false);
```

# Screen Wake Lock API

Что мы можем узнать:

- “попросить” не отключать или блокировать экран устройства



[https://developer.mozilla.org/en-US/docs/Web/API/Screen\\_Wake\\_Lock\\_API](https://developer.mozilla.org/en-US/docs/Web/API/Screen_Wake_Lock_API)

# Screen Wake Lock API - пример

Когда открываем страницу с билетом - просим не тушить экран, наверняка пользователь хочет его на входе показать. Хотя давайте и при покупке также делать, чтобы пользователь видел что еще не дал нам денег

[illegible]



```
1 const initWakeLock = async () => {
2     if ("wakeLock" in navigator) {
3         console.log("Screen Wake Lock API supported!");
4
5         let wakeLock = null;
6
7         // create an async function to request a wake lock
8         try {
9             wakeLock = await navigator.wakeLock.request("screen");
10            console.log("Wake Lock is active!");
11        } catch (err) {
12            console.error(err);
13        }
14    } else {
15        console.log("Wake lock is not supported by this
16 browser.");
17 }
```

# Vibration API

Что мы можем:


- можем вибрировать, а можем не вибрировать

Для чего:

- Когда хотите привлечь внимание пользователя без звука и прочих спецэффектов







```
const initUpdateHowMuchPeoplesWatchingSubscription = () => {
  let peoplesWatchingCount = 1;
  const vibrate = () => {
    if (window.navigator.vibrate) {
      console.log('bzzzzzzz');
      window.navigator.vibrate([100, 100, 100]);
    } else {
      console.log("vibrate none available in your
browser");
    }
    setInterval(() => {
      vibrate();
      peoplesWatchingCount++;
      document.getElementById(
        "peoplesWatching"
      ).innerText = String(peoplesWatchingCount);
    }, 10000);
  }
}
```

# Что еще интересного есть, но время поджимает

- **Screen Orientation API** - ориентация устройства
- **Contact Picker API** - можно попросить выбрать контакт пользователя
- **EyeDropper API** - пипетка для того чтобы выбрать цвет, радость создателям WYSIWYG редакторов
- **Sensors API** - акселерометры, магнитометры, датчики освещенности и прочие интересные штуки
- **Web Speech API** - можем генерировать голос, а можем распознавать
- **Gamepad API** - если вы делаете игры, вы про него знаете
- **Web NFC API** - читать и писать NFC метки, можно сделать много удобного с взаимодействием с реальным миром
- и много чего интересного



# Всем спасибо!

Контакты для связи:

<https://m0rg0t.github.io/alenev/>

