# ~~Воруем~~ Добываем данные из интернета используя Node.js

Малькевич Егор

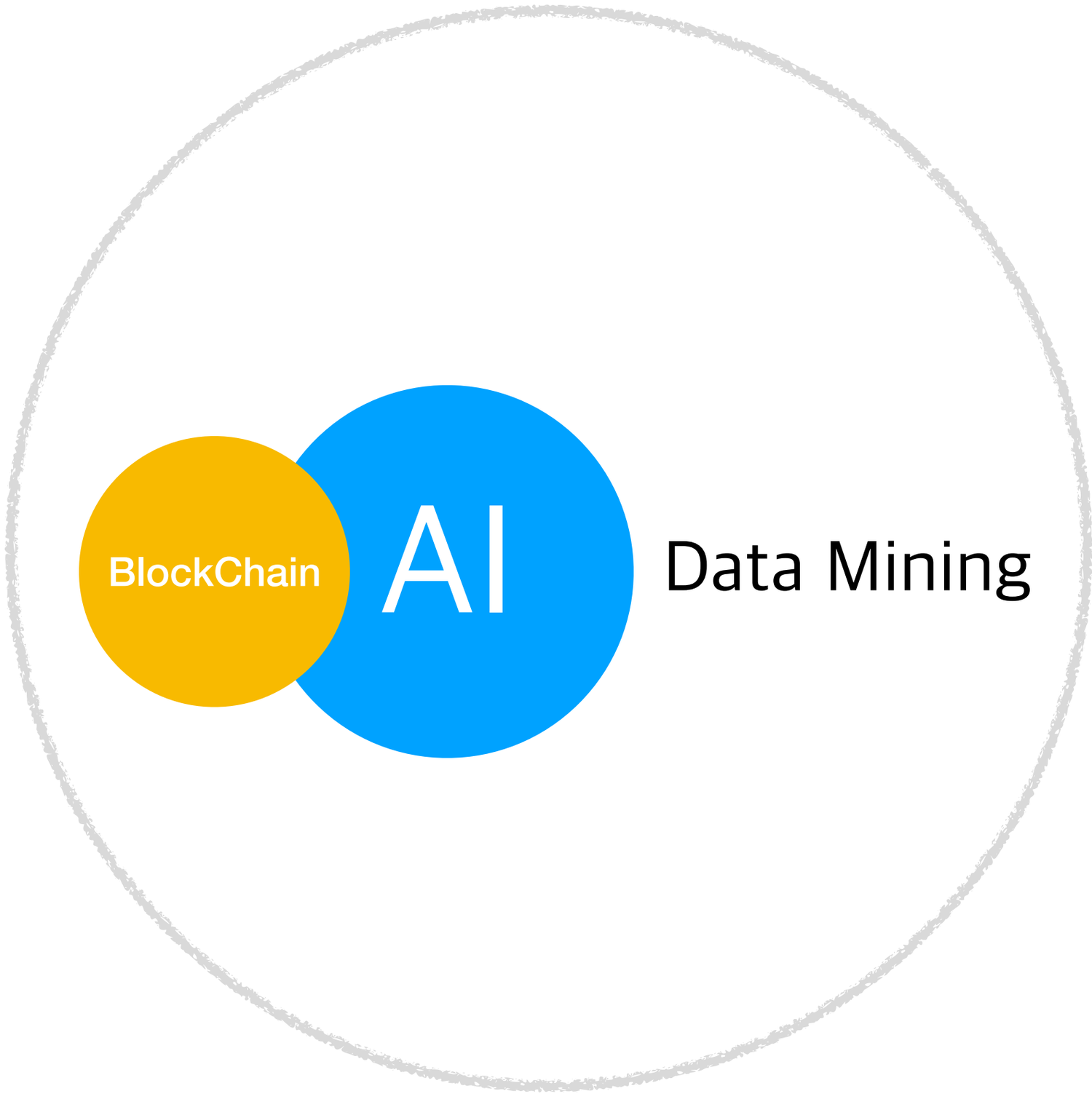
2017

от работы время я изучаю возможности получить разного рода данные из интернета причем таким образом что бы сервер ценой в 5$ на Digital Ocean мог справиться с этой работой. Для меня это MVP.

Собственно и сегодня я здесь перед вами, с целью поделиться накопленными знаниями, и показать что добывать данные нужно и нужно это всем и каждому. Пускай это будет новый голубой океан

https://malkevich.com/

# Что сейчас Main Stream?




Парсеры

| Followers | YouTube | facebook | Instagram |
|---|---|---|---|
| 100k - 500k | $12,500 | $6,250 | $5,000 |
| 500k - 1m | $25,000 | $12,500 | $10,000 |
| 1m - 3m | $125,000 | $62,500 | $50,000 |
| 3m - 7m | $187,500 | $93,750 | $75,000 |
| over 7m | $300,000 | $187,500 | $150,000 |

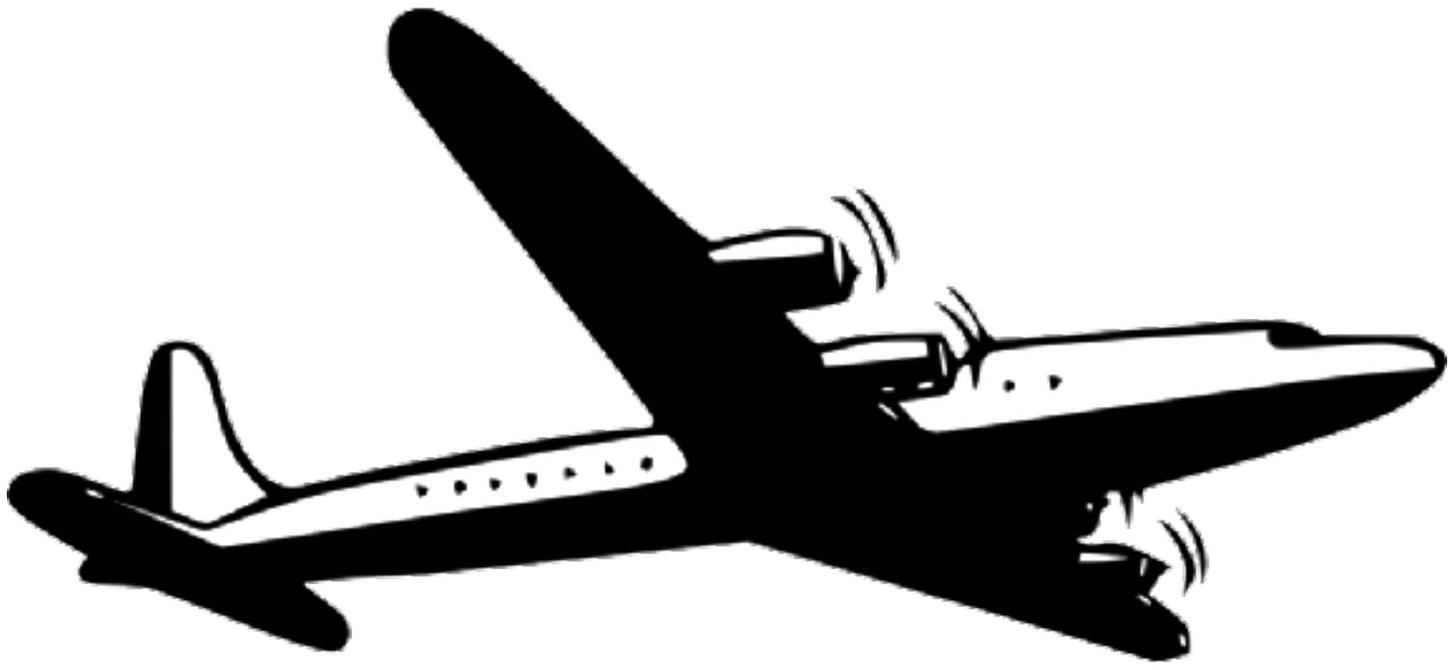
| Followers | snapchat | Vine | twitter |
|---|---|---|---|
| 100k - 500k | $5,000 | $3,750 | $2,000 |
| 500k - 1m | $10,000 | $7,500 | $4,000 |
| 1m - 3m | $50,000 | $37,500 | $20,000 |
| 3m - 7m | $75,000 | $56,250 | $30,000 |
| over 7m | $150,000 | $112,500 | $60,000 |

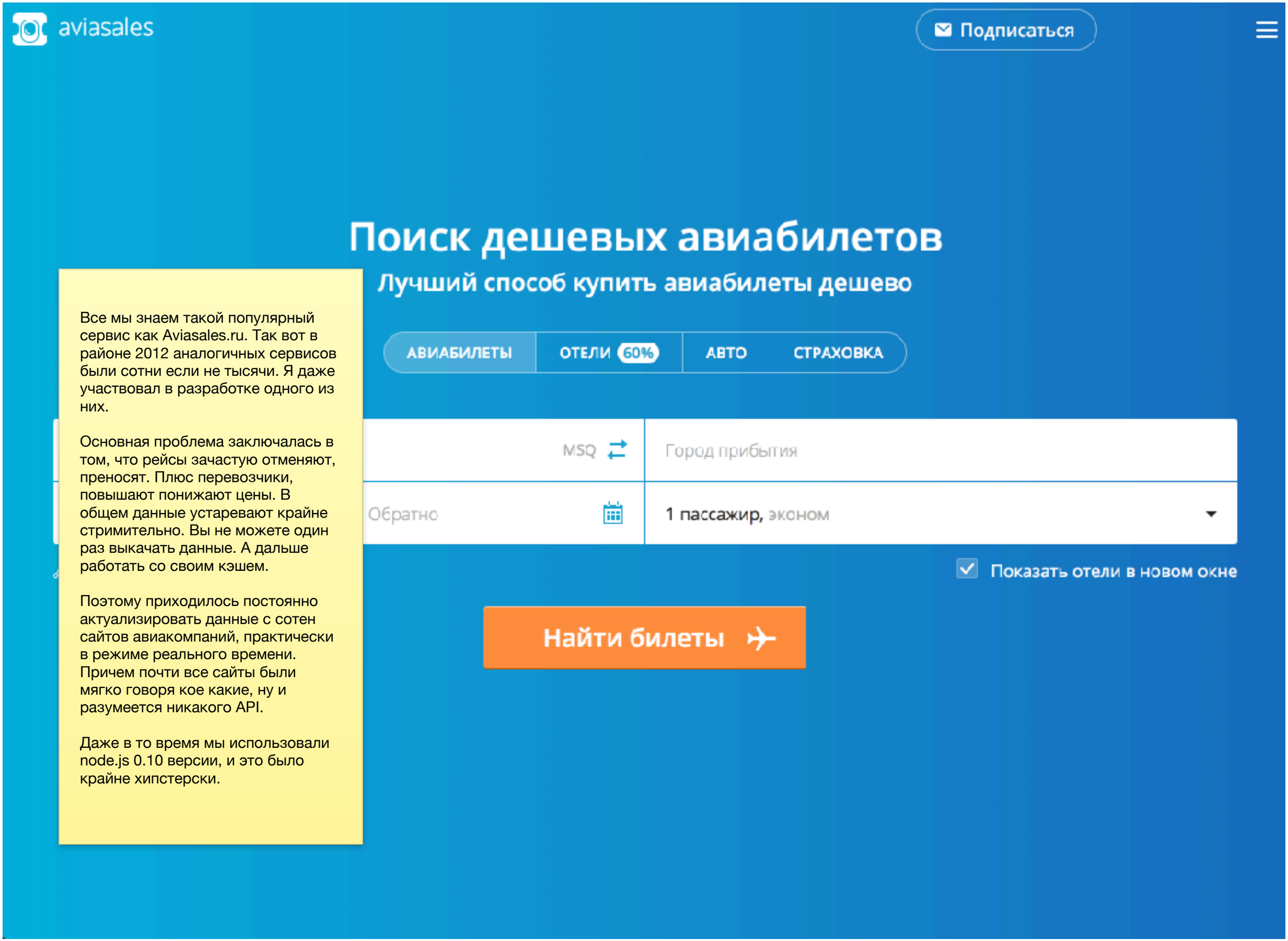
Source: Captiv8

Economist.com

# Пару историй

# Самолеты



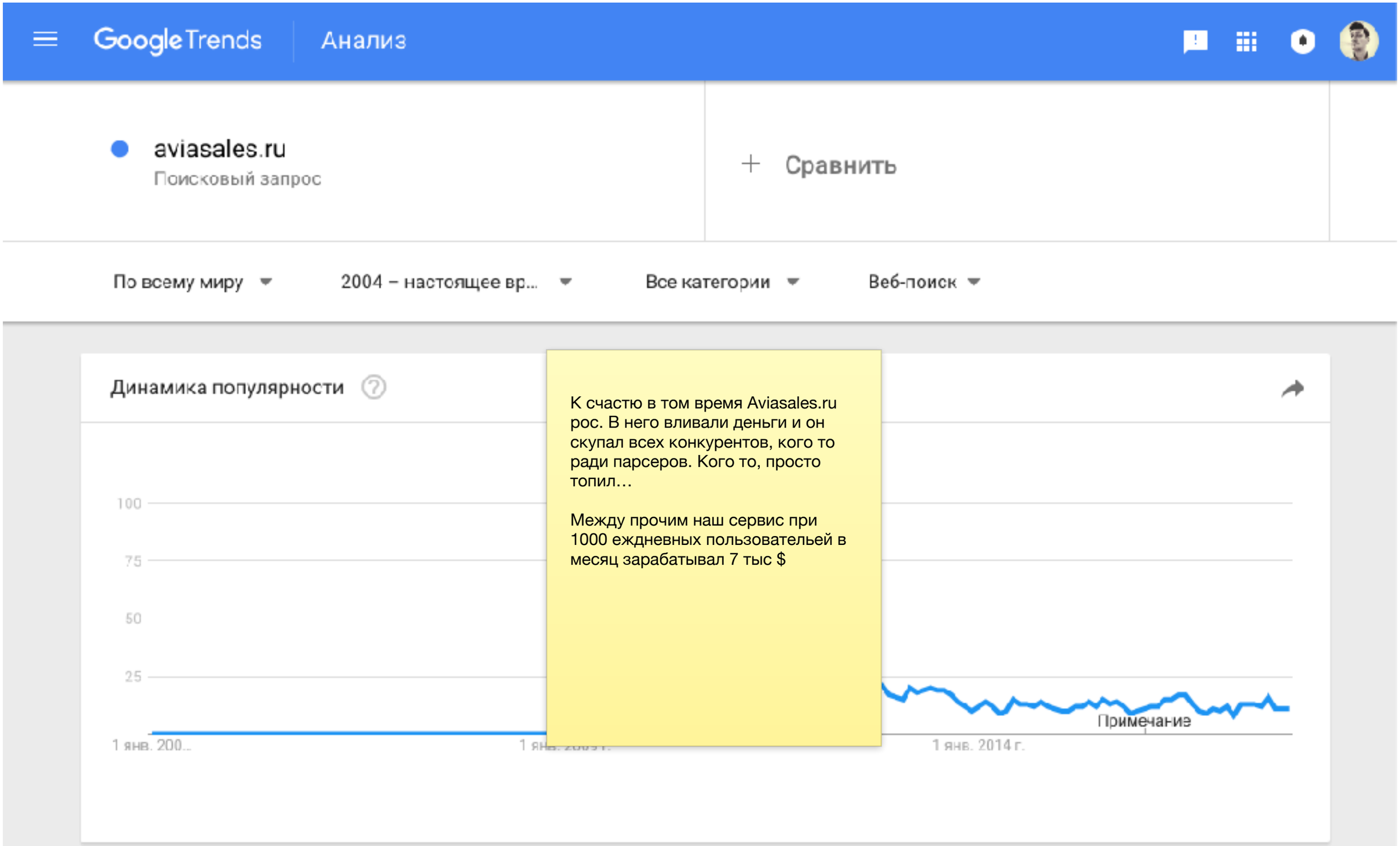



Все мы знаем такой популярный сервис как Aviasales.ru. Так вот в районе 2012 аналогичных сервисов были сотни если не тысячи. Я даже участвовал в разработке одного из них.

Основная проблема заключалась в том, что рейсы зачастую отменяют, переносят. Плюс перевозчики, повышают понижают цены. В общем данные устаревают крайне стримительно. Вы не можете один раз выкачать данные. А дальше работать со своим кэшем.

Поэтому приходилось постоянно актуализировать данные с сотен сайтов авиакомпаний, практически в режиме реального времени. Причем почти все сайты были мягко говоря кое какие, ну и разумеется никакого API.

Даже в то время мы использовали node.js 0.10 версии, и это было крайне хипстерски.




К счастью в том время Aviasales.ru рос. В него вливали деньги и он скупал всех конкурентов, кого то ради парсеров. Кого то, просто топил…

Между прочим наш сервис при 1000 ежедневных пользователей в месяц зарабатывал 7 тыс $

Google Авиабилеты

Войти

Туда и обратно | Один пассажир | Эконом-класс

Москва SVO DME VKO, Курс... | Минск MSQ | чт, 11 янв. | вс, 14 янв.

Минск – выберите рейс | Обратно: Москва, Жуковский | Сведения о путешествии

Прямой | Цена | Время | ЮТэйр | Ещё

## Советы по поиску рейсов

**ДАТЫ**

В выбранные дни билеты обычно стоят дешево

ЕЩЁ...

**ЦЕН**

Минск
меняются
п

**АЭРОПОРТЫ**

к: билеты на рейсы в
рты неподалеку могут
тоить дешевле.

ЕЩЁ...

**СОВЕТ**

Минск: спла
путеш

ЕЩЁ...

Если вдруг вы решите делать подобный сервис сегодня, обратите внимание на google.ru/flights

Потому что гугл уже парсит и индексирует миллионы подобных сервисов за вас.

## Рейсы вылета

Указана итоговая цена за один взрослый билет с учет
багажа и другие комиссии.

сборы за провоз

Упорядочить по:

| | | | | |
|---|---|---|---|---|
| 10:00 – 11:20<br>ЮТэйр | 1 ч. 20 мин.<br>VKO–MSQ | Прямой рейс | 4 408 ₽<br>туда и обратно | |
| 14:40 – 16:00<br>ЮТэйр | 1 ч. 20 мин.<br>VKO–MSQ | Прямой рейс | 4 408 ₽<br>туда и обратно | |

# Машины


Все же сталкивались с перекупами? Которые скупают тачки подешевле, шаманят и продают по дороже. Так вот, до появления таможенного союза, эти мастера продаж, могли зарабатывать влегкую по несколько тыс $ с машины.

На сегодняшний день, ихняя маржа упала. Скажем до 300$ с машины в среднем.

Сайтов по продаже авто стало меньше. Найти и купить самостоятельно авто - проще.

И вот забавно, но факт. Перекупы эволюционировали и нашли возможности в «высокочастотном трэйдинге».

Все марки

503 576 объявлений

Сохранить поиск

Любой регион

Все параметры

Показать 503 576 объявлений

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BMW | 22209 | Hyundai | 32863 | Mercedes-Benz | 26016 | Renault | 16523 |
| Chevrolet | 18195 | Kia | 21515 | Mitsubishi | 15596 | Toyota | 35500 |
| Ford | 24023 | LADA (ВАЗ) | 82341 | Nissan | 27204 | Volkswagen | 27991 |



Продажа автомобилей

По актуальности

За все время

₽

**Peugeot 308**

1.6 AT (120 л.с

привод, хэтчб

00 ₽

2009

69 800 км

Москва

**Suzuki Gran**

2.0 AT (140 л.с

привод, внедо

00 ₽

2005

193 095 км

Подольск

Схема такая, вы выкидываете машину на продажу, перекупы сами или исполььюзуя ботов сканируют доски объявлений, и стараются первыми посмотреть тачку. И иногда успевают купить и продать в тот же день. Посути как работая как брокеры на бирже.

У автору есть платный сервис, который позволяет вам видеть объявления раньше других. Но зачастую дешевле написать свой маленький парсер, что многие и делают.

К примеру, в среднем в Минске выкидывают 30-60 новых объявлений о продаже машины в день.

То-есть если добавить туда маржу в 300$ - максимальный суточный доход может достигать 90000$

# Видео и сериалы

Никто так не парсит контент, как пиратские сайты с видео.

Они пишут парсеры
- на группы вконтакте,
- сканируют торренты,
- сканируют своих конкурентов,
- сканируют сайты студий озвучки
- парсят еще кучу сомнительных сайтов
- они анализируют оффициальную ленту релизов фильмов и выгинеривают фэйковые страницы с ожидаемыми фильмами и сериалами, что бы попадать в высокочастотные запросы

Я не говорю про Seasonvar.ru, а говорю в целом про нишу пиратских видео.

Вы бы видели какие гавносайты являются, донорами этих агрегаторов.

Между прочим, даже средний такой пиратский сайт может приносить +-3k$ при наличии 10k пользователей, на рекламе или пожертвованиях.

Мы с ребяами даже создали сервис, Gendalf TV. В первую очередь провести ряд экспериментов по парсингу видео.

Во вторую дать людям возможность стримить видео с любого сайта на Smart TV, без необходимости держать бук, или телефон залоченым.

Собственно в момент разработки этого сервиса я в серьез увлекся разработкой парсеров. Много эксперементировал в поиске дешового решения, для разработки собственных парсеров.

В общем об этом опыте я и хочу с вами поделиться

https://gendalf.tv

# API которым делятся и которое нужно

# http://seasonvar.ru/api.txt

Ответ возвращается в формате JSON.

Обработчик требует наличия двух обязательных параметров:
1. key - ключ авторизации, уникальный для каждого пользователя
2. command - название команды

Список команд (method):
1. getSerialList - список всех сериалов.
   1. country (массив)
      Список стран.
      Пример: array('сша', 'россия ')
   2. genre (массив)
      Список жанров.
      Пример: array('анимационные', 'комедия')
   3. locale (строка).
      Два вида - domestic(отечественные), foreign(иностранные)
   4. sort (массив)
      Принимает два параметра: order и method
      -order (массив)
      -kinopoisk - сортировка по оценкам кинопоиска
      -imdb - сортировка по оценкам imdb
      -popular - сортировка по популярным сериалам
      -year - сортировка по году
      -method(строка)
      -desc -выставляет DESC (по умолчанию ASC)

      Пример:
      array( 'order' => 'popular','method' => 'DESC')

      -lastSeasonInfo (bool)
      Дополнительная информация к последнему сезону
      По умолчанию - false.
   5. letter (string) - по букве (или части слова, начиная с начала)

1. getSeason - информацию по указанному сезону. Требует наличия дополнительного параметра season_id (id запрашиваемого сезона).

https://habrahabr.ru/post/302150/



raiym 30 мая 2016 в 12:03 Разработка

# Что делать если Instagram не дал доступ к API?

PHP, Open source, API

Из песочницы

1 июня 2016 года Instagram отключит от своего API все приложения, которые не прошли модерацию. Что делать если вы в их числе?

## Предыстория

Мы делаем сервис для постинга в Instagram по расписанию и используем API для получения информации об аккаунтах. Самим постингом занимаются телефоны в автоматическом режиме. Нам отказали в доступе к API после 1 июня (пробовали пройти модерацию два раза) поэтому было решено найти замену.

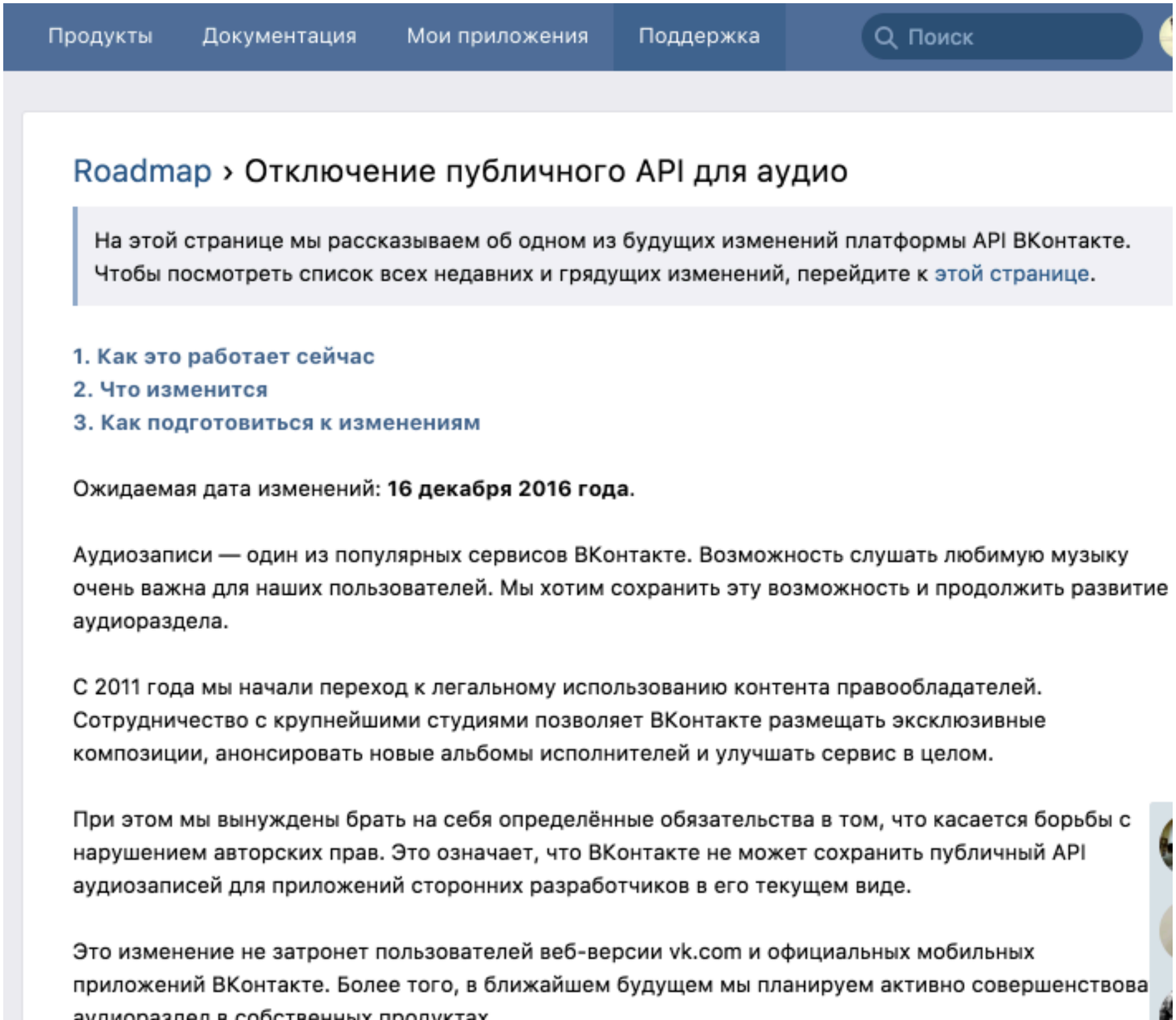
Сначала расскажу как мы использовали официальный API:

1. При добавлении аккаунта забираем из Instagram информацию об аккаунте: имя, фото профайла, количество постов, подписчиков, подписок.

2. Перед тем как опубликовать фото/видео мы запрашиваем количество постов, и тоже самое после публикации, если число постов увеличилось считаем публикацию успешной.

3. Если публикация прошла успешно забираем ссылку на последнее фото в профайле

**https://vk.com/dev.php?method=audio_api**



Продукты Документация Мои приложения Поддержка Поиск

Roadmap › Отключение публичного API для аудио

> На этой странице мы рассказываем об одном из будущих изменений платформы API ВКонтакте.
> Чтобы посмотреть список всех недавних и грядущих изменений, перейдите к этой странице.

1. Как это работает сейчас
2. Что изменится
3. Как подготовиться к изменениям

Ожидаемая дата изменений: **16 декабря 2016 года**.

Аудиозаписи — один из популярных сервисов ВКонтакте. Возможность слушать любимую музыку очень важна для наших пользователей. Мы хотим сохранить эту возможность и продолжить развитие аудиораздела.

С 2011 года мы начали переход к легальному использованию контента правообладателей. Сотрудничество с крупнейшими студиями позволяет ВКонтакте размещать эксклюзивные композиции, анонсировать новые альбомы исполнителей и улучшать сервис в целом.

При этом мы вынуждены брать на себя определённые обязательства в том, что касается борьбы с нарушением авторских прав. Это означает, что ВКонтакте не может сохранить публичный API аудиозаписей для приложений сторонних разработчиков в его текущем виде.

Это изменение не затронет пользователей веб-версии vk.com и официальных мобильных приложений ВКонтакте. Более того, в ближайшем будущем мы планируем активно совершенствова
аудиораздел в собственных продуктах.

# Coding Time

# What will we need?

- request-promise—Request is a simple HTTP client that allows us to make quick and easy HTTP calls.

- cheerio—jQuery for Node.js. Cheerio makes it easy to select, edit, and view DOM elements.

```
1   import request from 'request-promise';  1.1M (gzipped: 320.5K)
2   import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
3
4   const options = {
5     uri: "https://www.google.com",
6     transform: (body) => cheerio.load(body)
7   };
8
9   const scrape = async (options) => {
10    return await request(options);
11  };
12
13  export default (params)=> scrape({...options, ...params });
14
```

Example 1

```
1   import request from 'request-promise';  1.1M (gzipped: 320.5K)
2   import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
3
4   const options = {
5     uri: "https://www.google.com",
6     transform: (body) => cheerio.load(body)
7   };
8
9   const scrape = async (options) => {
10    return await request(options);
11  };
12
13  export default (params)=> scrape({...options, ...params });
14
```

Example 1

```
import request from 'request-promise';  1.1M (gzipped: 320.5K)
import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)

const options = {
  uri: "https://www.google.com",
  transform: (body) => cheerio.load(body)
};

const scrape = async (options) => {
  return await request(options);
};

export default (params)=> scrape({...options, ...params });
```

Example 1

```
import request from 'request-promise';  1.1M (gzipped: 320.5K)
import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)

const options = {
  uri: "https://www.google.com",
  transform: (body) => cheerio.load(body)
};


const scrape = async (options) => {
  return await request(options);
};

export default (params)=> scrape({...options, ...params });

```

Example 1

```
1   import request from 'request-promise';  1.1M (gzipped: 320.5K)
2   import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
3
4   const options = {
5     uri: "https://www.google.com",
6     transform: (body) => cheerio.load(body)
7   };
8
9   const scrape = async (options) => {
10    return await request(options);
11  };
12
13  export default (params)=> scrape({...options, ...params });
14
```

Example 1

```
1   import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
2
3   const $ = cheerio.load(`
4   <ul id="cities">
5       <li class="large">New York</li>
6       <li id="c-medium">Portland</li>
7       <li class="small">Salem</li>
8   </ul>
9
10  <ul id="towns">
11      <li class="large">Bend</li>
12      <li id="t-medium">Hood River</li>
13      <li class="small">Madras</li>
14  </ul>`);
15
16  export default $;
```

Example 2

```
1   import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
2
3   const $ = cheerio.load(`
4   <ul id="cities">
5       <li class="large">New York</li>
6       <li id="c-medium">Portland</li>
7       <li class="small">Salem</li>
8   </ul>
9
10  <ul id="towns">
11      <li class="large">Bend</li>
12      <li id="t-medium">Hood River</li>
13      <li class="small">Madras</li>
14  </ul>`);
15
16  export default $;
```

Example 2

```
1   import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
2
3   const $ = cheerio.load(`
4   <ul id="cities">
5       <li class="large">New York</li>
6       <li id="c-medium">Portland</li>
7       <li class="small">Salem</li>
8   </ul>
9
10  <ul id="towns">
11      <li class="large">Bend</li>
12      <li id="t-medium">Hood River</li>
13      <li class="small">Madras</li>
14  </ul>`);
15
16  export default $;
```

Example 2

```
$('#cities').find('.small').text() // Salem
$('#towns').find('.small').text() // Madras

// or all

$('.small').map((index, el)=> $(el).text()); // ["Salem",
"Madras"]
```

Example 2

# Limitations

MOST websites modify the DOM using JavaScript.

Unfortunately Cheerio doesn't resolve parsing a modified DOM. Dynamically generated content from procedures leveraging AJAX, client-side logic, and other async procedures are not available to Cheerio.

# At the end of 2017

PhantomJS' development has stopped, Headless Chrome is in the spotlight—and people love it

- There are a lot of libraries for controlling Chrome, pick the one you like;

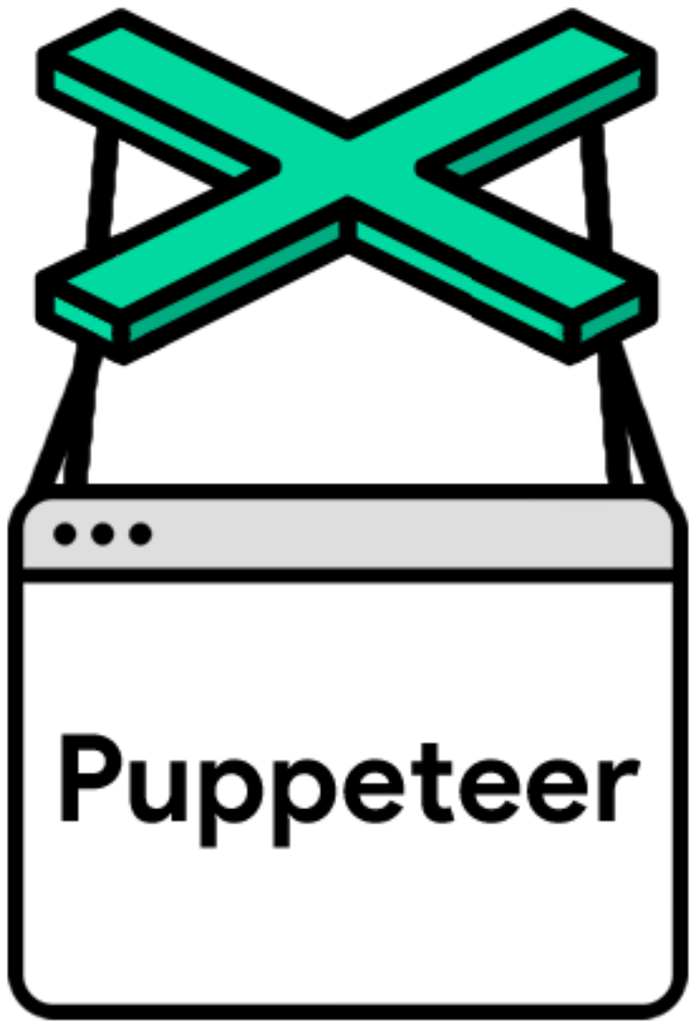
- Web scraping with Headless Chrome is easy, even more so when you're aware of these tips & tricks;

- Headless browser visitors can be detected but nobody does it.

You'll need to have Node 8+ installed on your computer.

| NPM package | chrome-remote-interface | chromeless | puppeteer | nickjs |
| --- | --- | --- | --- | --- |
| GitHub stars | ~2k | ~10k | ~11k | ~0.03k (yeah!) |
| Maintainer | Andrea Cardaci | Graphcool | Google's DevTools team | Phantombuster |
| API | Generic low level mapping between NodeJS and the DevTools protocol. | Mostly complete API with command chaining (imitates NigthmareJS). Allows for an easy AWS Lambda integration. | Google's take on a generic NodeJS mapping of the DevTools protocol. Complete API that tries to simplify Chrome automation. | Very simple API that gets out of your way as much as possible. Is specifically made with scraping in mind. |

To do this, we'll use Puppeteer. [Puppeteer] is a Node library API that allows us to control headless Chrome


Puppeteer

```
1   import puppeteer from 'puppeteer';
2
3   const scrape = async() => {
4       const browser = await puppeteer.launch({
5           headless: false
6       });
7       const page = await browser.newPage();
8       await page.setViewport({
9           width: 1240,
10          height: 680
11      });
12      await page.goto('https://google.com');
13      await page.screenshot({
14          path: 'google.png'
15      });
16      await browser.close();
17  }
18
19  export default () => scrape();
20
```

Example 3

```
import puppeteer from 'puppeteer';

const scrape = async() => {
    const browser = await puppeteer.launch({
        headless: false
    });
    const page = await browser.newPage();
    await page.setViewport({
        width: 1240,
        height: 680
    });
    await page.goto('https://google.com');
    await page.screenshot({
        path: 'google.png'
    });
    await browser.close();
}

export default () => scrape();
```

Example 3

```
import puppeteer from 'puppeteer';

const scrape = async() => {
    const browser = await puppeteer.launch({
        headless: false
    });
    const page = await browser.newPage();
    await page.setViewport({
        width: 1240,
        height: 680
    });
    await page.goto('https://google.com');
    await page.screenshot({
        path: 'google.png'
    });
    await browser.close();
}

export default () => scrape();

```

Example 3

```
1   import puppeteer from 'puppeteer';
2
3   const scrape = async() => {
4       const browser = await puppeteer.launch({
5           headless: false
6       });
7       const page = await browser.newPage();
8       await page.setViewport({
9           width: 1240,
10          height: 680
11      });
12      await page.goto('https://google.com');
13      await page.screenshot({
14          path: 'google.png'
15      });
16      await browser.close();
17  }
18
19  export default () => scrape();
20
```

Example 3

```
import puppeteer from 'puppeteer';

const scrape = async() => {
    const browser = await puppeteer.launch({
        headless: false
    });
    const page = await browser.newPage();
    await page.setViewport({
        width: 1240,
        height: 680
    });
    await page.goto('https://google.com');
    await page.screenshot({
        path: 'google.png'
    });
    await browser.close();
}

export default () => scrape();
```

Example 3

```
import puppeteer from 'puppeteer';

const scrape = async() => {
    const browser = await puppeteer.launch({
        headless: false
    });
    const page = await browser.newPage();
    await page.setViewport({
        width: 1240,
        height: 680
    });
    await page.goto('https://google.com');
    await page.screenshot({
        path: 'google.png'
    });
    await browser.close();
}

export default () => scrape();
```

Example 3

Anything that has real and lasting value is always a gift from within.

(Franz Kafka)

все, что имеет реальную и прочную ценность, всегда является подарком изнутри

# Fallback to the first example

```
1   import request from 'request-promise';  1.1M (gzipped: 320.5K)
2   import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
3
4   const options = {
5     uri: "https://www.google.com",
6     transform: (body) => cheerio.load(body)
7   };
8
9   const scrape = async (options) => {
10    return await request(options);
11  };
12
13  export default (params)=> scrape({...options, ...params });
14
```

Example 4

```
1   import puppeteer from 'puppeteer';
2   import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
3
4   const options = {
5     uri: `http://books.toscrape.com/`,
6     transform: (body) => cheerio.load(body)
7   };
8
9   const scrape = async (options) => {
10      const browser = await puppeteer.launch(options);
11      const page = await browser.newPage();
12      await page.goto(options.uri);
13      let content = await page.content();
14      let $ = options.transform(content);
15      browser.close();
16      return $;
17  };
18
19  export default (params)=> scrape({...options, ...params });
20
```

Example 4

```
import puppeteer from 'puppeteer';
import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)

const options = {
  uri: `http://books.toscrape.com/`,
  transform: (body) => cheerio.load(body)
};

const scrape = async (options) => {
    const browser = await puppeteer.launch(options);
    const page = await browser.newPage();
    await page.goto(options.uri);
    let content = await page.content();
    let $ = options.transform(content);
    browser.close();
    return $;
};

export default (params)=> scrape({...options, ...params });

```

Example 4

```
import puppeteer from 'puppeteer';
import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)

const options = {
  uri: `http://books.toscrape.com/`,
  transform: (body) => cheerio.load(body)
};

const scrape = async (options) => {
    const browser = await puppeteer.launch(options);
    const page = await browser.newPage();
    await page.goto(options.uri);
    let content = await page.content();
    let $ = options.transform(content);
    browser.close();
    return $;
};

export default (params)=> scrape({...options, ...params });

```

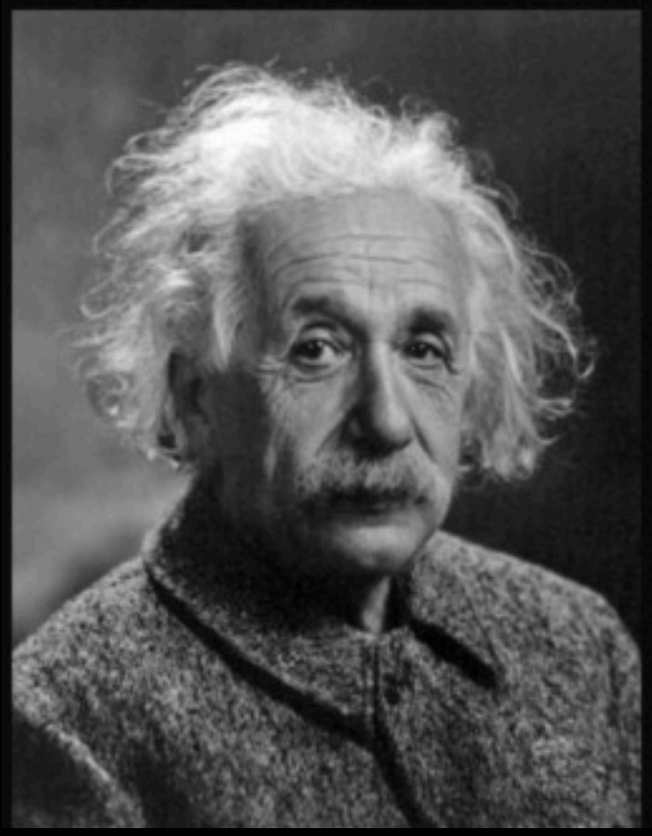
Example 4

```
import puppeteer from 'puppeteer';
import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)

const options = {
  uri: `http://books.toscrape.com/`,
  transform: (body) => cheerio.load(body)
};

const scrape = async (options) => {
    const browser = await puppeteer.launch(options);
    const page = await browser.newPage();
    await page.goto(options.uri);
    let content = await page.content();
    let $ = options.transform(content);
    browser.close();
    return $;
};

export default (params)=> scrape({...options, ...params });

```

Example 4

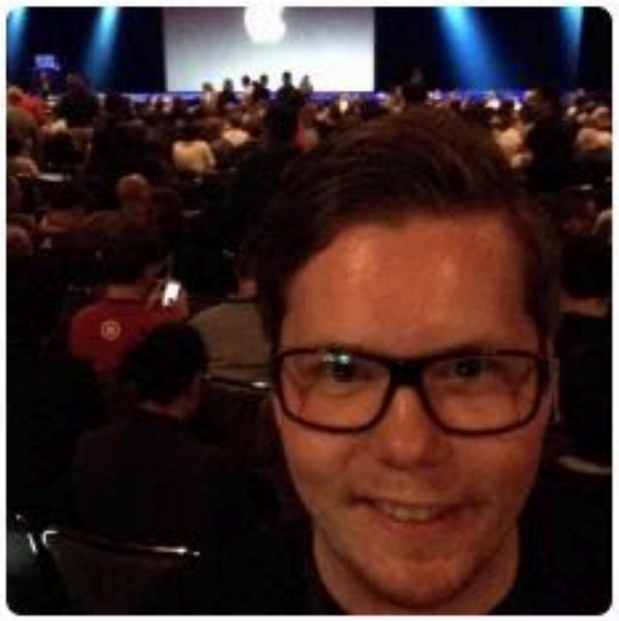
```
1   import puppeteer from 'puppeteer';
2   import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
3
4   const options = {
5     uri: `http://books.toscrape.com/`,
6     transform: (body) => cheerio.load(body)
7   };
8
9   const scrape = async (options) => {
10      const browser = await puppeteer.launch(options);
11      const page = await browser.newPage();
12      await page.goto(options.uri);
13      let content = await page.content();
14      let $ = options.transform(content);
15      browser.close();
16      return $;
17  };
18
19  export default (params)=> scrape({...options, ...params });
20
```

Example 4



Everything should be made as simple as possible, but not one bit simpler.

(Albert Einstein)



# John Sundell

JohnSundell

I build apps, games and Swift developer tools! Passionate about open source & developer productivity. You can follow me on Twitter **@JohnSundell**.

**Follow**

Block or report user

Poland / Norway / Sweden

john@sundell.co

https://twitter.com/johnsundell

## Organizations



## Pinned repositories

### Unbox

The easy to use Swift JSON decoder

Swift ★ 1.7k 123

### Wrap

The easy to use Swift JSON encoder

Swift ★ 548 39

### Marathon

Marathon makes it easy to write, run and manage your Swift scripts 🏃

Swift ★ 1.3k 49

### ImagineEngine

A project to create a blazingly fast Swift game engine that is a joy to use 🚀

Swift ★ 972 40

### Files

A nicer way to handle files & folders in Swift

Swift ★ 983 52

### TestDrive

Quickly try out any Swift pod or framework in a playground

Swift ★ 1.2k 28

## 1,455 contributions in the last year

Block or report user

Poland / Norway / Sweden

john@sundell.co

https://twitter.com/johnsundell

## Organizations

## 74,511 users

Sort: Best match

**john** **John** McGrath

I'm a solutions architect at AWS and prior to that co-founded Entelo. Interested in renewable energy, media, and democracy.

San Francisco, CA

Follow

**JohnSundell** **John** Sundell

I build apps, games and Swift developer tools! Passionate about open source & developer productivity. You can follow me on Twitter @johnsundell.

Poland / Norway / Sweden john@sundell.co

Follow

jr0cket **John** Stevenson

Speaker, author, conference organiser & community obsessed developer. Loves Clojure, Emacs, Cats, Cycling & Agile development.

London, UK john@jr0cket.co.uk

Follow

jdegoes **John** A. De Goes

Boulder, CO john@degoes.net

Follow

### Languages

| | |
|---|---|
| JavaScript | 5,645 |
| Java | 4,034 |
| Python | 3,082 |
| HTML | 2,202 |
| Ruby | 1,925 |
| PHP | 1,787 |
| C# | 1,497 |
| CSS | 1,349 |
| C++ | 1,342 |
| C | 1,170 |

# Sign in to GitHub

**Username or email address**

**Password** Forgot password?

**Sign in**

New to GitHub? Create an account.

Terms Privacy Security Contact GitHub

```
Elements  Console  Sources  Network  »  3
<div id="js-pjax-container" data-pjax-container>
  ▼<div class="auth-form px-3" id="login">
    <!-- '"` -->
    <!-- </textarea></xmp> -->
    ▼<form accept-charset="UTF-8" action="/session" method="post">
      ▶<div style="margin:0;padding:0;display:inline">…</div>
      ▶<div class="auth-form-header p-0">…</div>
       <div id="js-flash-container">
       </div>
      ▼<div class="auth-form-body mt-3">
         <label for="login_field">
                  Username or email address
                </label>
...      <input autocapitalize="off" autocorrect="off" autofocus=
         "autofocus" class="form-control input-block" id="login_fie
         name="login" tabindex="1" type="text"> == $0
       ▶<label for="password">…</label>
         <input class="form-control form-control input-block" id=
         "password" name="password" tabindex="2" type="password">
         <input class="btn btn-primary btn-block" data-disable-with
         "Signing in…" name="commit" tabindex="3" type="submit" val
         "Sign in">
       </div>
     </form>
    ▶<p class="create-account-callout mt-3">…</p>
   </div>
  </div>
  <div class="modal-backdrop js-touch-events"></div>
 </div>
▶<div class="footer container-lg p-responsive py-6 mt-6 f6" role=
"contentinfo">…</div>
▶<div id="ajax-error-message" class="ajax-error-message flash flash-
…</div>
 <script crossorigin="anonymous" integrity="sha256-
 F7VsAbjYEuEdAvWDOjCP6snWeqx5tkyuQYm7fPXaSw0=" src="https://assets-
 cdn.github.com/assets/frameworks-17b56c0….js"></script>
```

```
... div  #js-pjax-container  #login  form  div  input#login_field.form-control.input-
```

```
Styles  Event Listeners  DOM Breakpoints  Properties  Accessibility Properties  Aureli

Filter                                              :hov  .cls  +

element.style {
}

.auth-form-body .input-block {                  github-15d10d5c…e792d414.css:6
  margin-top: 5px;
  margin-bottom: 15px;
}

.input-block {                                  frameworks-d713…5465cab.css:14
  display: block;
  width: 100%;
```

```
102  const scrape = async (options) => {
103      const browser = await puppeteer.launch(options.launch);
104      const page = await browser.newPage();
105
106      await page.setViewport(options.setViewport);
107      await login(page, options);
108      await search(page, options);
109
110      const numPages = await getNumPages(page, options);
111
112      let users = await numPages.times(async (index)=> {
113          return await receiveUserMeta(index + 1, page, options);
114      });
115
116      browser.close();
117
118      return [].concat(...users);
119  }
```

Example 5

```
102  const scrape = async (options) => {
103      const browser = await puppeteer.launch(options.launch);
104      const page = await browser.newPage();
105
106      await page.setViewport(options.setViewport);
107      await login(page, options);
108      await search(page, options);
109
110      const numPages = await getNumPages(page, options);
111
112      let users = await numPages.times(async (index)=> {
113          return await receiveUserMeta(index + 1, page, options);
114      });
115
116      browser.close();
117
118      return [].concat(...users);
119  }
```

Example 5

```
13  const login = async(page, options) => {
14      const CREDS = {
15          username: options.username,
16          password: options.password
17      };
18
19      const USERNAME_SELECTOR = '#login_field';
20      const PASSWORD_SELECTOR = '#password';
21      const BUTTON_SELECTOR = '#login > form > div.auth-form-body.mt-3 >
22
23      await page.goto('https://github.com/login');
24      // dom element selectors
25      await page.click(USERNAME_SELECTOR);
26      await page.keyboard.type(CREDS.username);
27      await page.click(PASSWORD_SELECTOR);
28      await page.keyboard.type(CREDS.password);
29      await page.click(BUTTON_SELECTOR);
30      await page.waitForNavigation();
31  }
32
```

Example 5

```
13  const login = async(page, options) => {
14      const CREDS = {
15          username: options.username,
16          password: options.password
17      };
18
19      const USERNAME_SELECTOR = '#login_field';
20      const PASSWORD_SELECTOR = '#password';
21      const BUTTON_SELECTOR = '#login > form > div.auth-form-body.mt-3 >
22
23      await page.goto('https://github.com/login');
24      // dom element selectors
25      await page.click(USERNAME_SELECTOR);
26      await page.keyboard.type(CREDS.username);
27      await page.click(PASSWORD_SELECTOR);
28      await page.keyboard.type(CREDS.password);
29      await page.click(BUTTON_SELECTOR);
30      await page.waitForNavigation();
31  }
32
```

Example 5

```
13  const login = async(page, options) => {
14      const CREDS = {
15          username: options.username,
16          password: options.password
17      };
18
19      const USERNAME_SELECTOR = '#login_field';
20      const PASSWORD_SELECTOR = '#password';
21      const BUTTON_SELECTOR = '#login > form > div.auth-form-body.mt-3 >
22
23      await page.goto('https://github.com/login');
24      // dom element selectors
25      await page.click(USERNAME_SELECTOR);
26      await page.keyboard.type(CREDS.username);
27      await page.click(PASSWORD_SELECTOR);
28      await page.keyboard.type(CREDS.password);
29      await page.click(BUTTON_SELECTOR);
30      await page.waitForNavigation();
31  }
32
```

Example 5

```
13  const login = async(page, options) => {
14      const CREDS = {
15          username: options.username,
16          password: options.password
17      };
18
19      const USERNAME_SELECTOR = '#login_field';
20      const PASSWORD_SELECTOR = '#password';
21      const BUTTON_SELECTOR = '#login > form > div.auth-form-body.mt-3 >
22
23      await page.goto('https://github.com/login');
24      // dom element selectors
25      await page.click(USERNAME_SELECTOR);
26      await page.keyboard.type(CREDS.username);
27      await page.click(PASSWORD_SELECTOR);
28      await page.keyboard.type(CREDS.password);
29      await page.click(BUTTON_SELECTOR);
30      await page.waitForNavigation();
31  }
32
```

Example 5

```
13  const login = async(page, options) => {
14      const CREDS = {
15          username: options.username,
16          password: options.password
17      };
18
19      const USERNAME_SELECTOR = '#login_field';
20      const PASSWORD_SELECTOR = '#password';
21      const BUTTON_SELECTOR = '#login > form > div.auth-form-body.mt-3 >
22
23      await page.goto('https://github.com/login');
24      // dom element selectors
25      await page.click(USERNAME_SELECTOR);
26      await page.keyboard.type(CREDS.username);
27      await page.click(PASSWORD_SELECTOR);
28      await page.keyboard.type(CREDS.password);
29      await page.click(BUTTON_SELECTOR);
30      await page.waitForNavigation();
31  }
32
```

Example 5

```
13  const login = async(page, options) => {
14      const CREDS = {
15          username: options.username,
16          password: options.password
17      };
18
19      const USERNAME_SELECTOR = '#login_field';
20      const PASSWORD_SELECTOR = '#password';
21      const BUTTON_SELECTOR = '#login > form > div.auth-form-body.mt-3 >
22
23      await page.goto('https://github.com/login');
24      // dom element selectors
25      await page.click(USERNAME_SELECTOR);
26      await page.keyboard.type(CREDS.username);
27      await page.click(PASSWORD_SELECTOR);
28      await page.keyboard.type(CREDS.password);
29      await page.click(BUTTON_SELECTOR);
30      await page.waitForNavigation();
31  }
32
```

Example 5

```
13  const login = async(page, options) => {
14      const CREDS = {
15          username: options.username,
16          password: options.password
17      };
18
19      const USERNAME_SELECTOR = '#login_field';
20      const PASSWORD_SELECTOR = '#password';
21      const BUTTON_SELECTOR = '#login > form > div.auth-form-body.mt-3 >
22
23      await page.goto('https://github.com/login');
24      // dom element selectors
25      await page.click(USERNAME_SELECTOR);
26      await page.keyboard.type(CREDS.username);
27      await page.click(PASSWORD_SELECTOR);
28      await page.keyboard.type(CREDS.password);
29      await page.click(BUTTON_SELECTOR);
30      await page.waitForNavigation();
31  }
32
```

Example 5

```
33  const getSearchUrl = (options) => {
34      const userToSearch = options.search || 'john';
35      return `https://github.com/search?q=${userToSearch}&type=Users
36  }
37
```

Example 5

```
33  const getSearchUrl = (options) => {
34      const userToSearch = options.search || 'john';
35      return `https://github.com/search?q=${userToSearch}&type=Users
36  }
37
```

Example 5

```
42  async function getNumPages(page, options) {
43      const NUM_USER_SELECTOR = '#js-pjax-container > div.container > d
44
45      let content = await page.content();
46      let $ = options.transform(content);
47
48       // format is: "69,803 users"
49      let inner = ($(NUM_USER_SELECTOR)
50                      .html() || '')
51                      .replace(',', '')
52                      .replace('users', '')
53                      .trim();
54      const numUsers = parseInt(inner);
55      /**
56       * GitHub shows 10 resuls per page, so
57       */
58      let num = Math.ceil(numUsers / 10);
59      return num > options.maxPage ? options.maxPage || 100 : num;
60  }
```

Example 5

```
62  const receiveUserMeta = async(index, page, options) => {
63      const LENGTH_SELECTOR_CLASS = '.user-list-item';
64
65      const LIST_USERNAME_SELECTOR = '.user-list-info.ml-2 > a';
66      const LIST_EMAIL_SELECTOR = '.octicon.octicon-mail + a.muted-link';
67
68      let pageUrl = getSearchUrl(options) + '&p=' + index;
69
70      await page.goto(pageUrl, {waitUntil: ['domcontentloaded']});
71
72      let content = await page.content();
73      let $ = options.transform(content);
74
75      return $(LENGTH_SELECTOR_CLASS).map(function() {
76          return {
77              username: $(this).find(LIST_USERNAME_SELECTOR).eq(0).text(),
78              email: $(this).find(LIST_EMAIL_SELECTOR).eq(0).text()
79          };
80      }).get().filter(e => e.email);
81  }
82
```

Example 5

```
const receiveUserMeta = async(index, page, options) => {
    const LENGTH_SELECTOR_CLASS = '.user-list-item';

    const LIST_USERNAME_SELECTOR = '.user-list-info.ml-2 > a';
    const LIST_EMAIL_SELECTOR = '.octicon.octicon-mail + a.muted-link';

    let pageUrl = getSearchUrl(options) + '&p=' + index;

    await page.goto(pageUrl, {waitUntil: ['domcontentloaded']});

    let content = await page.content();
    let $ = options.transform(content);

    return $(LENGTH_SELECTOR_CLASS).map(function() {
        return {
            username: $(this).find(LIST_USERNAME_SELECTOR).eq(0).text(),
            email: $(this).find(LIST_EMAIL_SELECTOR).eq(0).text()
        };
    }).get().filter(e => e.email);
}

```

Example 5

```
62  const receiveUserMeta = async(index, page, options) => {
63      const LENGTH_SELECTOR_CLASS = '.user-list-item';
64
65      const LIST_USERNAME_SELECTOR = '.user-list-info.ml-2 > a';
66      const LIST_EMAIL_SELECTOR = '.octicon.octicon-mail + a.muted-link';
67
68      let pageUrl = getSearchUrl(options) + '&p=' + index;
69
70      await page.goto(pageUrl, {waitUntil: ['domcontentloaded']});
71
72      let content = await page.content();
73      let $ = options.transform(content);
74
75      return $(LENGTH_SELECTOR_CLASS).map(function() {
76          return {
77              username: $(this).find(LIST_USERNAME_SELECTOR).eq(0).text(),
78              email: $(this).find(LIST_EMAIL_SELECTOR).eq(0).text()
79          };
80      }).get().filter(e => e.email);
81  }
82
```

Example 5

```
62  const receiveUserMeta = async(index, page, options) => {
63      const LENGTH_SELECTOR_CLASS = '.user-list-item';
64
65      const LIST_USERNAME_SELECTOR = '.user-list-info.ml-2 > a';
66      const LIST_EMAIL_SELECTOR = '.octicon.octicon-mail + a.muted-link';
67
68      let pageUrl = getSearchUrl(options) + '&p=' + index;
69
70      await page.goto(pageUrl, {waitUntil: ['domcontentloaded']});
71
72      let content = await page.content();
73      let $ = options.transform(content);
74
75      return $(LENGTH_SELECTOR_CLASS).map(function() {
76          return {
77              username: $(this).find(LIST_USERNAME_SELECTOR).eq(0).text(),
78              email: $(this).find(LIST_EMAIL_SELECTOR).eq(0).text()
79          };
80      }).get().filter(e => e.email);
81  }
82
```

Example 5

```
const receiveUserMeta = async(index, page, options) => {
    const LENGTH_SELECTOR_CLASS = '.user-list-item';

    const LIST_USERNAME_SELECTOR = '.user-list-info.ml-2 > a';
    const LIST_EMAIL_SELECTOR = '.octicon.octicon-mail + a.muted-link';

    let pageUrl = getSearchUrl(options) + '&p=' + index;

    await page.goto(pageUrl, {waitUntil: ['domcontentloaded']});

    let content = await page.content();
    let $ = options.transform(content);

    return $(LENGTH_SELECTOR_CLASS).map(function() {
        return {
            username: $(this).find(LIST_USERNAME_SELECTOR).eq(0).text(),
            email: $(this).find(LIST_EMAIL_SELECTOR).eq(0).text()
        };
    }).get().filter(e => e.email);
}

```

Example 5

```
const receiveUserMeta = async(index, page, options) => {
    const LENGTH_SELECTOR_CLASS = '.user-list-item';

    const LIST_USERNAME_SELECTOR = '.user-list-info.ml-2 > a';
    const LIST_EMAIL_SELECTOR = '.octicon.octicon-mail + a.muted-link';

    let pageUrl = getSearchUrl(options) + '&p=' + index;

    await page.goto(pageUrl, {waitUntil: ['domcontentloaded']});

    let content = await page.content();
    let $ = options.transform(content);

    return $(LENGTH_SELECTOR_CLASS).map(function() {
        return {
            username: $(this).find(LIST_USERNAME_SELECTOR).eq(0).text(),
            email: $(this).find(LIST_EMAIL_SELECTOR).eq(0).text()
        };
    }).get().filter(e => e.email);
}

```

Example 5

```
import puppeteer from 'puppeteer';
import cheerio from 'cheerio';

var timesFunction = function(callback) {
};

String.prototype.times = timesFunction;
Number.prototype.times = timesFunction;

const options = {
    url: `https://www.google.com`,
    transform: (body) => cheerio.load(body),
};

const login = async(page) => {
    const CREDS = {
        username: '****',
        password: '****'
    };

    const USERNAME_SELECTOR = '#login_field';
    const PASSWORD_SELECTOR = '#password';
    const BUTTON_SELECTOR = '#login > form > div.auth-form-body.mt-3 > input.btn.btn-primary.btn-block';

    await page.goto('https://github.com/login');
    // dom element selectors
    await page.click(USERNAME_SELECTOR);
    await page.keyboard.type(CREDS.username);
    await page.click(PASSWORD_SELECTOR);
    await page.keyboard.type(CREDS.password);
    await page.click(BUTTON_SELECTOR);
    await page.waitForNavigation();
}

const search = async (page)=> {
    const userToSearch = 'jonn';
    const searchUrl = 'https://github.com/search?q=${userToSearch}&type=Users&utf8=%E2%9C%93';
    await page.goto(searchUrl);
    await page.waitFor(2 * 1000);
}

async function getNumPages(page, options) {
    const NUM_USER_SELECTOR = '#js-pjax-container > div.container > div > div.column.three-fourths.codesearch-results.pr-6 > div.d-flex.flex-justify-between.border-bottom.pb-3 > h3';
    let content = await page.content;
    let $ = options.transform(content);

    // format is: "69,883 users"
    let inner = $(NUM_USER_SELECTOR)
                .html()
                .replace(',', '')
                .replace('users', '')
                .trim();

    const numUsers = parseInt(inner);
    console.log('numUsers: ', numUsers);
    /*
     * GitHub shows 10 results per page. so
     */
    return Math.ceil(numUsers / 10);
}

const recieveUserMeta = async(index, page, options) => {
    const LIST_USERNAME_SELECTOR = '#user_search_results > div.user-list > div:nth-child(INDEX) > div.d-flex > div > a';
    const LIST_EMAIL_SELECTOR = '#user_search_results > div.user-list > div:nth-child(INDEX) > div.d-flex > div > ul > li:nth-child(2) > a';
    const LENGTH_SELECTOR_CLASS = 'user-list-item';

    let pageUrl = searchUrl + '&p=' + index;
    await page.goto(pageUrl);
    let content = await page.content;
    let $ = options.transform(content);

    let listLength = $(sel).length;

    return listLength.times((index)=> {
        let usernameSelector = LIST_USERNAME_SELECTOR.replace("INDEX", index + 1);
        let emailSelector = LIST_EMAIL_SELECTOR.replace("INDEX", index + 1);

        let username = $(usernameSelector).attr('href').replace('/', '');
        let email = $(emailSelector).html();
        // not all users have emails visible
        if (!email)
            return null;

        return {
            username: username,
            email: email
        };
    });
}

const scrape = async () => {
    const browser = await puppeteer.launch();
    const page = await browser.newPage();

    await login(page);
    await search(page);

    const numPages = await getNumPages(page, options);
    let users = await Promise.all(numPages.times(async (index)=> {
        return await recieveUserMeta(index + 1, page, options);
    }));

    browser.close();

    return users;
}

scrape(options);
```

All together 121 line of code

index.test.js — presentation-one

index.js .../puppeteer-login-github

JS index.test.js .../puppeteer-login-github

```
1  import scrape from './index';
2  import config from '../config';
3  jasmine.DEFAULT_TIMEOUT_INTERVAL = 999999;
   1 inconclusive
4  describe('puppeteer login github', () => {
       Inconclusive | Debug
5      test('should return array', async () => {
6          let users = await scrape({
7              username: config.github.username,
8              password: config.github.password,
9              search: "bitcoin",
10             maxPage: 20
11         });
12
13         const resp = {
14             username: expect.stringMatching(/.*/),
15             email: expect.stringMatching(/@/),
16         };
17
18         expect(Array.isArray(users)).toBe(true)
19         expect(users.length).toBeGreaterThan(0);
20
21         return expect(users[0]).toMatchObject(resp);
22     });
23 });
24
```

1: zsh

```
# macbook at abc in ~/sandbox/presentation/p
resentation-one on git:master ✖ [1:45:12]
→ npm test ./src/puppeteer-login-github
```

index.test.js — presentation-one

index.js .../puppeteer-login-github

JS index.test.js .../puppeteer-login-github

```
1  import scrape from './index';
2  import config from '../config';
3  jasmine.DEFAULT_TIMEOUT_INTERVAL = 999999;
   1 Inconclusive
4  describe('puppeteer login github', () => {
       Inconclusive | Debug
5      test('should return array', async () => {
6          let users = await scrape({
7              username: config.github.username,
8              password: config.github.password,
9              search: "bitcoin",
10             maxPage: 20
11         });
12
13         const resp = {
14             username: expect.stringMatching(/.*/),
15             email: expect.stringMatching(/@/),
16         };
17
18         expect(Array.isArray(users)).toBe(true)
19         expect(users.length).toBeGreaterThan(0);
20
21         console.log(users);
22         return expect(users[0]).toMatchObject(resp);
23     });
24 });
25
```

1: node

```
# macbook at abc in ~/sandbox/presentation/p
resentation-one on git:master ✖ [1:45:12]
→ npm test ./src/puppeteer-login-github

> crawler-tests@1.0.0 test /Users/macbook/sa
ndbox/presentation/presentation-one
> jest "./src/puppeteer-login-github"

 PASS  src/puppeteer-login-github/index.test
.js (20.401s)
  puppeteer login github
    ✓ should return array (19915ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        20.924s
Ran all test suites matching /.\/src\/puppet
eer-login-github/i.

# macbook at abc in ~/sandbox/presentation/p
resentation-one on git:master ✖ [1:46:31]
→ npm test ./src/puppeteer-login-github

> crawler-tests@1.0.0 test /Users/macbook/sa
ndbox/presentation/presentation-one
> jest "./src/puppeteer-login-github"

 RUNS  ...ppeteer-login-github/index.test.js
```

A famous example of that is LinkedIn. Setting the **li_atcookie** will guarantee your scraper bot access to their social network (please note: we encourage you to respect your target website ToS).

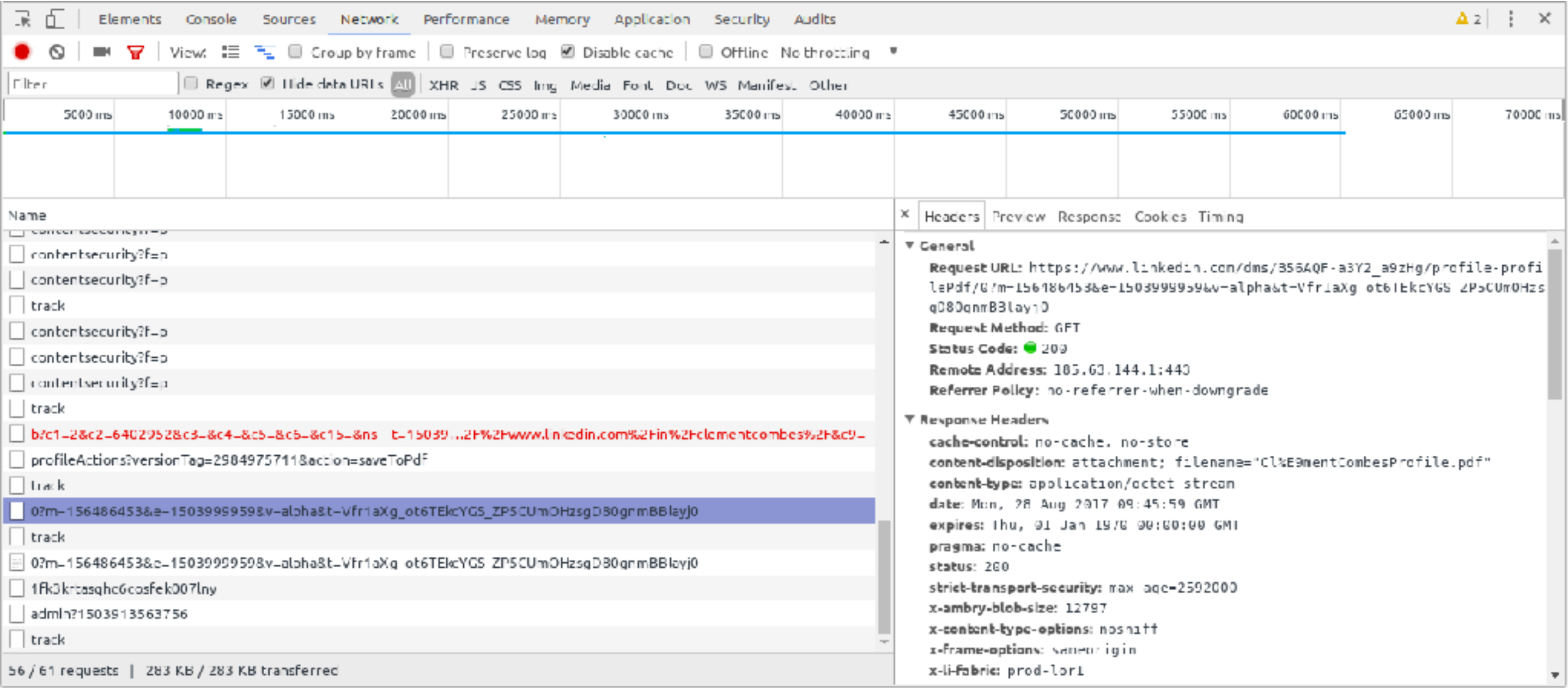
https://phantombuster.com/

```
import puppeteer from 'puppeteer';

const scrape = async () => {
    const browser = await puppeteer.launch();
    const page = await browser.newPage();

    await page.setCookie({
        name: "li_at",
        value: "a session cookie value copied from your DevTools",
        domain: "www.linkedin.com"
    });
    // make something awesome here
}
```

**page.setCookie(...cookies)**

- `...cookies` <...Object>
  - `name` <string> **required**
  - `value` <string> **required**
  - `url` <string>
  - `domain` <string>
  - `path` <string>
  - `expires` <number> Unix time in seconds.
  - `httpOnly` <boolean>
  - `secure` <boolean>
  - `sameSite` <string> `"Strict"` or `"Lax"`.
- returns: <Promise>

https://phantombuster.com/




Because we're using the DevTools API, the code we write has the equivalent power of a human using Chrome's DevTools. That means your bot can intercept, examine and even modify or abort any network request.

index.test.js — presentation-one

JS index.test.js .../puppeteer-linkedin-cookie-demo ✖ | JS config.js | MD doc.md

```
10  //             cookie: config.linkedIn.token,
11  //         });
12
13  //         return expect($("[data-control-name='identity_welcome_me
14  //     });
15  // });
16
17
    1 Inconclusive
18  describe('puppeteer github cookie', () => {
        Inconclusive | Debug
19      test('should login using mine cookie', async () => {
20          let $ = await scrape({
21              uri: "https://github.com/search?q=bitcoin&type=Users&ut
22              cookie: config.github.token,
23              setCookie: {
24                  name: "user_session",
25                  domain: "github.com"
26              },
27          });
28
29          let nickname = $(".css-truncate-target").text();
30          console.log(nickname);
31
32          return expect(nickname).toMatch(/wegorich/);
33      });
34  });
35
```

1: zsh

```
# macbook at abc in ~/sandbox/presentatio
n/presentation-one on git:master ✖ [2:04:
37]
➜
```


In the same vein, we can speed up our scraping by blocking unnecessary requests. Analytics, ads and images are typical targets. However, you have to keep in mind that it will make your bot less human-like (for example LinkedIn will not serve their pages properly if you block all images—we're not sure if it's deliberate or not).

```
import puppeteer from 'puppeteer';
import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)

const options = {
    launch: { headless: true },
    blacklist: [
        /.*collector\.githubapp.*/,
        /.*fsitouchzoom\.js/,
        /.*api\.github\.com\/_private\/browser.*/,
        /.*google.*/
    ],
    setViewport: { width: 1240, height: 680 },
    setCookie: {
        name: "li_at",
        domain: "www.linkedin.com"
    },
    transform: (body) => cheerio.load(body)
};
```

Example 8

```
import puppeteer from 'puppeteer';
import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)

const options = {
    launch: { headless: true },
    blacklist: [
        /.*collector\.githubapp.*/,
        /.*fsitouchzoom\.js/,
        /.*api\.github\.com\/_private\/browser.*/,
        /.*google.*/
    ],
    setViewport: { width: 1240, height: 680 },
    setCookie: {
        name: "li_at",
        domain: "www.linkedin.com"
    },
    transform: (body) => cheerio.load(body)
};
```

Example 8

```
20  const writeFileInterceptor = ({blacklist})=> (e) => {
21      if (blacklist.find(item => item.test(e.url))) {
22          e.abort();
23      } else {
24          e.continue();
25      }
26  }
```

Example 8

```
28  const scrape = async (options) => {
29      const browser = await puppeteer.launch(options.launch);
30      const page = await browser.newPage();
31
32      await page.setViewport(options.setViewport);
33      await page.setRequestInterception(true);
34      page.on('request', writeFileInterceptor(options));
35      //"a session cookie value copied from your DevTools",
36      options.setCookie.value = options.cookie || options.setCookie.value;
37      await page.setCookie(options.setCookie);
38      await page.goto(options.uri, {waitUntil: ['domcontentloaded']});
39
40      let content = await page.content();
41      let $ = options.transform(content);
42      browser.close();
43      return $;
44      // make something awesome here
45  };
```

Example 8

```
26      let start = null;
27
28      // first item loads 100ms longer usually
29      // and 1.2s longer first run
30
31      start = now();
32      await scrape(Object.assign({}, options, {blacklist: []}));
33      let speedDefault = now() - start;
34
35      start = now();
36      await scrape(options);
37      let speedAdBlock = now() - start;
38
39      console.log(`AdBlocked speed - ${speedAdBlock}, by default - ${speedDefault}`);
40      return expect(speedAdBlock).toBeLessThanOrEqual(speedDefault);
```

Example 8

```
26      let start = null;
27
28      // first item loads 100ms longer usually
29      // and 1.2s longer first run
30
31      start = now();
32      await scrape(Object.assign({}, options, {blacklist: []}));
33      let speedDefault = now() - start;
34
35      start = now();
36      await scrape(options);
37      let speedAdBlock = now() - start;
38
39      console.log(`AdBlocked speed - ${speedAdBlock}, by default - ${speedDefault}`);
40      return expect(speedAdBlock).toBeLessThanOrEqual(speedDefault);
```

Example 8

```
26      let start = null;
27
28      // first item loads 100ms longer usually
29      // and 1.2s longer first run
30
31      start = now();
32      await scrape(Object.assign({}, options, {blacklist: []}));
33      let speedDefault = now() - start;
34
35      start = now();
36      await scrape(options);
37      let speedAdBlock = now() - start;
38
39      console.log(`AdBlocked speed - ${speedAdBlock}, by default - ${speedDefault}`);
40      return expect(speedAdBlock).toBeLessThanOrEqual(speedDefault);
```

Example 8

index.test.js — presentation-one

index.js .../puppeteer-adblock-demo

JS index.test.js .../puppeteer-adblock-demo

```
1  import scrape from './index';
2  import config from '../config';
3  import now from 'performance-now';  2.7K (gzipped: 1.1K)
4
5  jasmine.DEFAULT_TIMEOUT_INTERVAL = 999999;
6
   1 Inconclusive
7  describe('puppeteer adblock', () => {
       Inconclusive | Debug
8      test('addblock should be faster', async () => {
9          let options = {
10             uri: "https://github.com/search?q=bitcoin&type=Users&ut
11             cookie: config.github.token,
12         setCookie: {
13             name: "user_session",
14             domain: "github.com"
15         },
16         blacklist: [
17             /.*assets-cdn\.github.*/,
18             /.*githubusercontent.*/,
19             /.*collector\.githubapp.*/,
20             /.*fsitouchzoom\.js/,
21             /.*api\.github\.com\/_private\/browser.*/,
22             /.*google.*/
23         ]
24     };
25
26     let start = null;
```

1: zsh

```
# macbook at abc in ~/sandbox/presentatio
n/presentation-one on git:master ✖ [2:22:
05]
→ npm test ./src/puppeteer-adblock-demo
```

# EasyList

```
[Adblock Plus 2.0]
! Version: 201712102233
! Title: EasyList
! Last modified: 10 Dec 2017 22:33 UTC
! Expires: 4 days (update frequency)
! Homepage: https://easylist.to/
! Licence: https://easylist.to/pages/licence.html
!
! Please report any unblocked adverts or problems
! in the forums (https://forums.lanik.us/)
! or via e-mail (easylist.subscription@gmail.com).
! GitHub issues: https://github.com/easylist/easylist/issues
! GitHub pull requests: https://github.com/easylist/easylist/pulls
!
! ----------------------General advert blocking filters----------------
! *** easylist:easylist/easylist_general_block.txt ***
&act=ads_
&ad.vid=$~xmlhttprequest
&ad_box_
&ad_channel=
&ad_classid=
&ad_height=
&ad_ids=
&ad_keyword=
```

https://easylist.to/easylist/easylist.txt

# abp-filter-parser

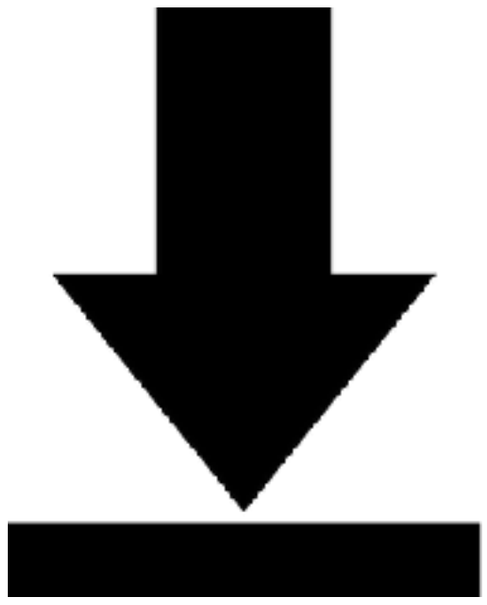
JavaScript Adblock Plus filter parser for lists like EasyList.

Parses filter rules as per:

https://adblockplus.org/en/filters
https://adblockplus.org/en/filter-cheatsheet

npm install abp-filter-parser

# Download File



# Установите Диск для Мас

Папка Яндекс.Диска выглядит так же, как обычная папка на компьютере, но хранит ваши файлы ещё и на сервере Яндекса. Они доступны только вам и тем, кого вы сами выберете. А добраться до них можно с любого устройства с интернетом.

Диск для Mac OS

Узнайте о бета-версии экспериментального Яндекс.Диска 2.0

```
15  const writeFileInterceptor = ({file})=> (e) => {
16      if (file.regexp.test(e.url)) {
17          e.buffer().then(buffer => {
18              fs.writeFileSync(file.path, Buffer.from(buffer, 'base64'));
19          });
20      }
21  }
```

Яндекс Диск

Компьютер | Планшет | Смартфон | Smart TV

# Установите Диск для Мас

Папка Яндекс.Диска выглядит так же, как обычная папка на компьютере, но хранит ваши файлы ещё и на сервере Яндекса. Они доступны только вам и тем, кого вы сами выберете. А добраться до них можно с любого устройства с интернетом.

Диск для Mac OS

Узнайте о бета-версии экспериментального Яндекс.Диска 2.0

Request.js

+

Puppeteer.js

## page.setContent(html)

- `html` <string> HTML markup to assign to the page.
- returns: <Promise>

### page.goto(url, options)

- `url` <string> URL to navigate page to. The url should include scheme, e.g. `https://` .
- `options` <Object> Navigation parameters which might have the following properties:
  - `timeout` <number> Maximum navigation time in milliseconds, defaults to 30 secon
  - `waitUntil` <string|Array<string>> When to consider navigation succeeded, defaul strings, navigation is considered to be successful after all events have been fired. Ev
    - `load` - consider navigation to be finished when the `load` event is fired.
    - `domcontentloaded` - consider navigation to be finished when the `DOMContentL`
    - `networkidle0` - consider navigation to be finished when there are no more tha `500` ms.
    - `networkidle2` - consider navigation to be finished when there are no more tha

```
import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
import { URL } from 'url';
import requestDriver from './request-driver';
import puppeteerDriver from './puppeteer-driver';

const options = {
    uri: 'https://ab.onliner.by/',
    selector: '.autoba-table-imp h2 strong',
    launch: { headless: true },
    blacklist: [
        /.*css/,
        /.*google.*/
    ],
    setViewport: { width: 1240, height: 680 },
    setCookie: {
        name: "li_at",
        value: "10",
        domain: "www.linkedin.com"
    },
    isDynamic: false,
    transform: (body) => cheerio.load(body)
};

```

```
import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
import { URL } from 'url';
import requestDriver from './request-driver';
import puppeteerDriver from './puppeteer-driver';

const options = {
    uri: 'https://ab.onliner.by/',
    selector: '.autoba-table-imp h2 strong',
    launch: { headless: true },
    blacklist: [
        /.*css/,
        /.*google.*/
    ],
    setViewport: { width: 1240, height: 680 },
    setCookie: {
        name: "li_at",
        value: "10",
        domain: "www.linkedin.com"
    },
    isDynamic: false,
    transform: (body) => cheerio.load(body)
};

```

```
import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
import { URL } from 'url';
import requestDriver from './request-driver';
import puppeteerDriver from './puppeteer-driver';

const options = {
    uri: 'https://ab.onliner.by/',
    selector: '.autoba-table-imp h2 strong',
    launch: { headless: true },
    blacklist: [
        /.*css/,
        /.*google.*/
    ],
    setViewport: { width: 1240, height: 680 },
    setCookie: {
        name: "li_at",
        value: "10",
        domain: "www.linkedin.com"
    },
    isDynamic: false,
    transform: (body) => cheerio.load(body)
};

```

```
const isDynamicUrls = new Map();

const scrape = async (options) => {
    let origin = (new URL(options.uri)).origin;
    let $ = null;
    if (options.isDynamic) {
        isDynamicUrls.set(origin, true);
    }

    if (!isDynamicUrls.has(origin)) {
        $ = await requestDriver(options);
        let items = $(options.selector);
        if (items.length) {
            return items.map(function() {return $(this).text();}).get();
        }

        isDynamicUrls.set(origin, true);
    }

    $ = await puppeteerDriver(options);
    return $(options.selector).map(function() {return $(this).text();}).get();
};
```

```
const isDynamicUrls = new Map();

const scrape = async (options) => {
    let origin = (new URL(options.uri)).origin;
    let $ = null;
    if (options.isDynamic) {
        isDynamicUrls.set(origin, true);
    }

    if (!isDynamicUrls.has(origin)) {
        $ = await requestDriver(options);
        let items = $(options.selector);
        if (items.length) {
            return items.map(function() {return $(this).text();}).get();
        }

        isDynamicUrls.set(origin, true);
    }

    $ = await puppeteerDriver(options);
    return $(options.selector).map(function() {return $(this).text();}).get();
};
```

```
const isDynamicUrls = new Map();

const scrape = async (options) => {
    let origin = (new URL(options.uri)).origin;
    let $ = null;
    if (options.isDynamic) {
        isDynamicUrls.set(origin, true);
    }

    if (!isDynamicUrls.has(origin)) {
        $ = await requestDriver(options);
        let items = $(options.selector);
        if (items.length) {
            return items.map(function() {return $(this).text();}).get();
        }

        isDynamicUrls.set(origin, true);
    }

    $ = await puppeteerDriver(options);
    return $(options.selector).map(function() {return $(this).text();}).get();
};
```

```
const isDynamicUrls = new Map();

const scrape = async (options) => {
    let origin = (new URL(options.uri)).origin;
    let $ = null;
    if (options.isDynamic) {
        isDynamicUrls.set(origin, true);
    }

    if (!isDynamicUrls.has(origin)) {
        $ = await requestDriver(options);
        let items = $(options.selector);
        if (items.length) {
            return items.map(function() {return $(this).text();}).get();
        }

        isDynamicUrls.set(origin, true);
    }

    $ = await puppeteerDriver(options);
    return $(options.selector).map(function() {return $(this).text();}).get();
};
```

```
const isDynamicUrls = new Map();

const scrape = async (options) => {
    let origin = (new URL(options.uri)).origin;
    let $ = null;
    if (options.isDynamic) {
        isDynamicUrls.set(origin, true);
    }

    if (!isDynamicUrls.has(origin)) {
        $ = await requestDriver(options);
        let items = $(options.selector);
        if (items.length) {
            return items.map(function() {return $(this).text();}).get();
        }

        isDynamicUrls.set(origin, true);
    }

    $ = await puppeteerDriver(options);
    return $(options.selector).map(function() {return $(this).text();}).get();
};
```



Все расчеты на территории Беларуси между гражданами Республики Беларусь осуществляются исключительно в белорусских рублях.
Цена в иностранной валюте (в случае ее наличия в объявлении граждан) указана справочно.

# Автобарахолка

Пройти диагностику

Разместить объявление всего за 90 секунд!

**32193** объявления

Актуальные | Последние добавленные

Год | Пробег | Цена, р.

**Цена** $ € р.

Любая – Любая

Обмен 14613

С диагностикой NEW 11

**Местонахождение**

Все страны

Все области

Все города

Растаможен 31824

**Марка**

Все марки

Все модели

Добавить марку

**Тип кузова**

Седан 11392

Универсал 4233

Хетчбэк 6493

Минивэн 2411

## Porsche 911 Carrera S

2013 | 72 000 км | **159 881** | 78 500 $ | 66 810 €

купе, бензин 3.8 л, автомат, задний привод, кондиционер, кожаный салон, легкосплавные диски, ксенон, парктроник, подогрев сидений, система контроля стабилизации, навигация, громкая связь.

17 UP! 2 часа назад

## BMW X6 XDRIVE 30D

2015 | 26 307 км | **119 962** | 58 900 $ | 50 129 € | ОБМЕН

внедорожник, дизель 3 л, автомат, полный привод, кондиционер, кожаный салон, легкосплавные диски, ксенон, парктроник, подогрев сидений, система контроля стабилизации, навигация, громкая связь.

8 UP! 4 часа назад

## Mini Cooper

2016 | 22 000 км | **52 529** | 25 792 $ | 21 950 € | ОБМЕН

хетчбэк, бензин 1.5 л, автомат, передний привод, кондиционер, легкосплавные диски, ксенон, парктроник, подогрев сидений, система контроля стабилизации.

8 UP! 4 часа назад

## BMW X5 50i

2010 | 118 992 км | **47 659** | 23 400 $

внедорожник, бензин 4.4 л, автомат, полный

**АВТОРАЗДЕЛЫ В УСЛУГАХ**

Ремонт и обслуживание авто

Перетяжка и ремонт салона

Шиномонтаж

Автомойка

Химчистка

Другие автоуслуги

*Все разделы Услуг*

Все расчеты на территории Беларуси между гражданами Республики Беларусь осуществляются исключительно в белорусских рублях. Цена в иностранной валюте (в случае ее наличия в объявлении граждан) указана справочно.

```
14      test('the code should detect dynamic pages, the cost should be permanent',
15          let urls = [
16              { uri: "https://ab.onliner.by/" },
17              { uri: 'https://ab.onliner.by/' },
18              { uri: 'https://ab.onliner.by/', isDynamic: true  },
19          ];
20
21          let otherResults = await mapSeries(urls, async (e)=> {
22              return await getPerformace(()=> scrape(e));
23          });
24
25          console.log(otherResults.map(e => e.speed));
26
```

Inconclusive | Debug

```
36  test('static sites shoud be superfast', async () => {
37      let urls = [
38          { uri: "https://ab.onliner.by/", selector: '.project-navigation__sign' },
39          { uri: 'https://ab.onliner.by/', selector: '.project-navigation__sign' },
40          { uri: 'https://ab.onliner.by/', selector: '.project-navigation__sign', isDynamic: true  },
41      ];
42
43      let otherResults = await mapSeries(urls, async (e)=> {
44          return await getPerformace(()=> scrape(e));
45      });
46
47      console.log(otherResults.map(e => e.speed));
48
```

index.test.js — presentation-one

t.js .../puppeteer-download-demo

JS index.test.js .../request-puppeteer-together-demo

```
6
7   const getPerformace = async (fn)=> {
8       let start = now()
9       let result = await fn();
10      return { result, speed: now() - start};
11  }
12
    2 Inconclusive
13  describe('puppeteer should work with request js', () => {
        Inconclusive | Debug
14      test('the code should detect dynamic pages, the cost should be perm
15          let urls = [
16              { uri: "https://ab.onliner.by/" },
17              { uri: 'https://ab.onliner.by/' },
18              { uri: 'https://ab.onliner.by/', isDynamic: true  },
19          ];
20
21          let otherResults = await mapSeries(urls, async (e)=> {
22              return await getPerformace(()=> scrape(e));
23          });
24
25          console.log(otherResults.map(e => e.speed));
26
27          otherResults.map(e=> e.speed - otherResults[0].speed).forEach(e
28              expect(e).toBeLessThanOrEqual(1500);
29          });
30
31          otherResults.map(e=> e.result).forEach(result=> {
32              expect(result.length).toBeGreaterThanOrEqual(1);
```

1: node

```
# macbook at abc in ~/sandbox/presen
tation/presentation-one on git:maste
r ✖ [3:00:33]
→ npm test ./src/request-puppeteer-t
ogether-demo
```

# Speed Up

# Puppeteer.js

```
let browser = null;
let browserTimeout = null;

const createBrowser = async ({launch})=> {
    clearTimeout(browserTimeout);
    if (!browser) {
        browser = await puppeteer.launch(launch);
    }

    return browser;
}

const closeBrowser = ()=> {
    clearTimeout(browserTimeout);
    browserTimeout = setTimeout(() => {
        browser.close();
        browser = null;
    }, 15000);
}
```

```
1   import scrape from './index';
2   import now from 'performance-now';  2.7K (gzipped: 1.1K)
3   import { mapSeries } from 'p-iteration';  3.4K (gzipped: 1K)
4
5   jasmine.DEFAULT_TIMEOUT_INTERVAL = 999999;
6
7   const getPerformace = async (fn)=> {
8       let start = now()
9       await fn();
10      return now() - start;
11  }
12
    1 Inconclusive
13  describe('puppeteer optimization test', () => {
        Inconclusive | Debug
14      test('running browser should cost enought of time', async () => {
15          let initTime = await getPerformace(()=> scrape({
16              uri: "https://www.google.com/search?num=50&ei=tMQqWqKZIM-S6QT7lIHIDQ&q=%D0%B2%D0%B8%D0%BA%D1%8
17          }));
18
19          let urls = [
20              'https://www.google.com/search?num=50&ei=tMQqWqKZIM-S6QT7lIHIDQ&q=%D0%B2%D0%B8%D0%BA%D1%82%D0%
21              'https://www.google.com/search?q=%D0%B2%D0%B8%D0%BA%D1%82%D0%BE%D1%80+%D0%BA%D0%BE%D0%BC%D0%B0
22              'https://www.google.com/search?q=%D0%B2%D0%B8%D0%BA%D1%82%D0%BE%D1%80+%D0%BA%D0%BE%D0%BC%D0%B0
23              'https://www.google.com/search?q=%D0%B2%D0%B8%D0%BA%D1%82%D0%BE%D1%80+%D0%BA%D0%BE%D0%BC%D0%B0
24              'https://www.google.com/maps?q=%D0%B2%D0%B8%D0%BA%D1%82%D0%BE%D1%80+%D0%BA%D0%BE%D0%BC%D0%B0%D
25              'https://www.google.com/search?q=%D0%B2%D0%B8%D0%BA%D1%82%D0%BE%D1%80+%D0%BA%D0%BE%D0%BC%D0%B0
26              'https://www.google.com/flights/',
27          ],
```

# Cost

**Note**: When you install Puppeteer, it downloads a recent version of Chromium (~71Mb Mac, ~90Mb Linux, ~110Mb Win) that is guaranteed to work with the API.

# Limitations

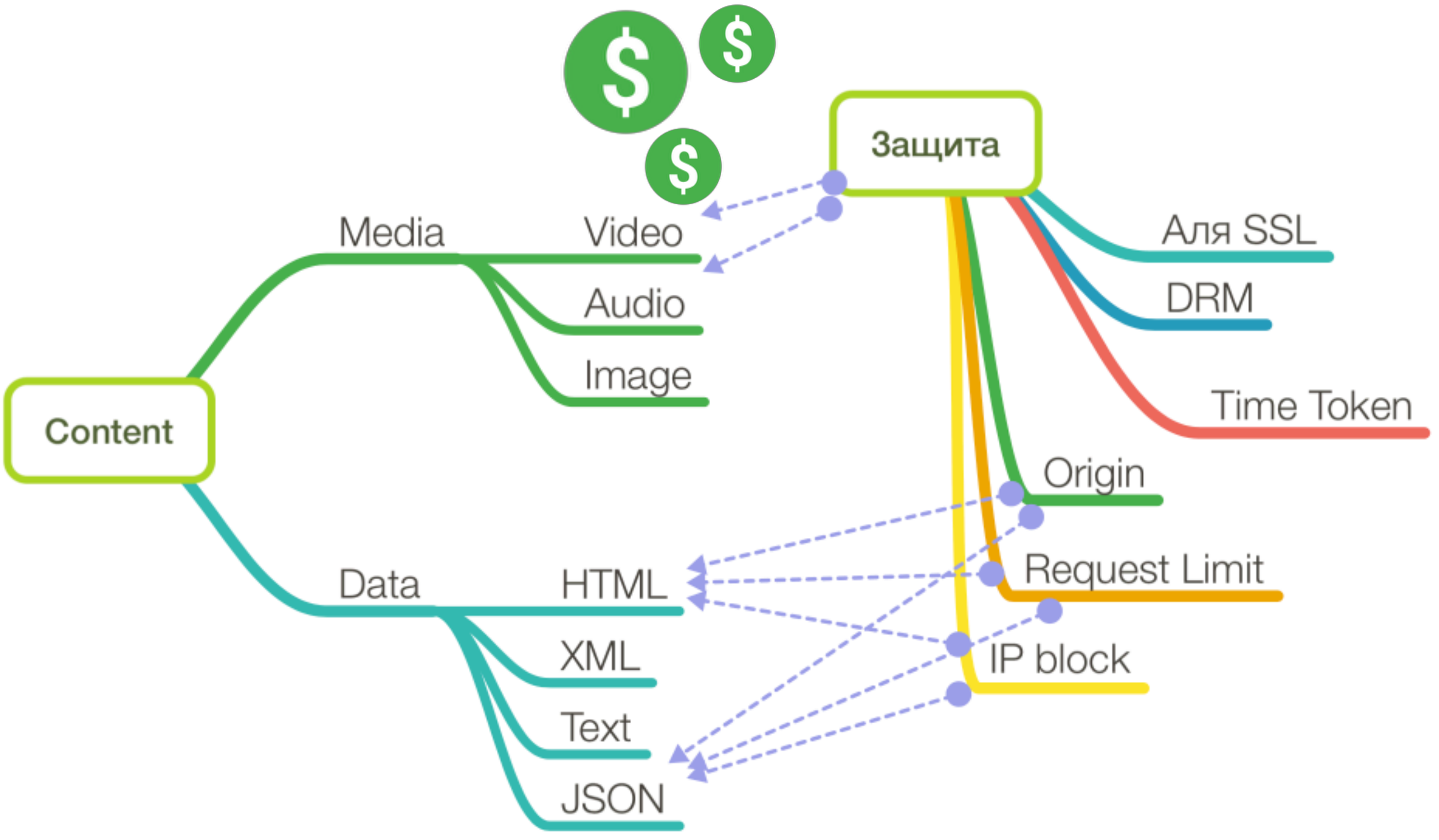
**Encodings -** Your text editor, browser, word processor or whatever else that's trying to read the document is assuming the wrong encoding (windows-1251).

**Query limits:**

Rule 1: Every request per user increments an internal count. When the count exceeds the limit, the requests are denied with a HTTP 429 Too Many Requests

Rule 2: The only way for count to go away, is for an internal expiration time to expire, called the expiry, and is measured in seconds

# Content


Защита
Media
Video
Audio
Image
Content
Data
HTML
XML
Text
JSON
Аля SSL
DRM
Time Token
Origin
Request Limit
IP block

# Text

Это никому не интересно, ну серьезно

# Images

You could simply use this expression to match an img tag as in the example :

```
<img([\w\W]+?)/>
```

# Audio

SOUNDCLOUD

Sign in | Create account

# Connect on SoundCloud

Discover, stream, and share a constantly expanding mix of music from emerging and major artists around the world.

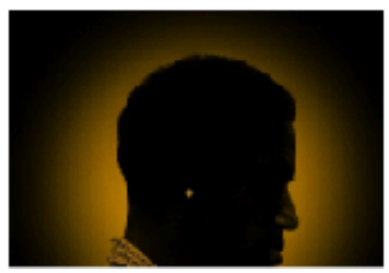
Sign up for free

Search for artists, bands, tracks, podcasts

or Upload your own

## Hear what's trending for free in the SoundCloud community




Audio – using soundcloud.com as example

**page.waitFor(selectorOrFunctionOrTimeout[, options])**

- `selectorOrFunctionOrTimeout` <string|number|function> A selector, predicate or timeout to wait for
- `options` <Object> Optional waiting parameters
- returns: <Promise>

This method behaves differently with respect to the type of the first parameter:

- if `selectorOrFunctionOrTimeout` is a `string`, than the first argument is treated as a selector to wait for and the method is a shortcut for page.waitForSelector
- if `selectorOrFunctionOrTimeout` is a `function`, than the first argument is treated as a predicate to wait for and t method is a shortcut for page.waitForFunction().
- if `selectorOrFunctionOrTimeout` is a `number`, than the first argument is treated as a timeout in milliseconds and method returns a promise which resolves after the timeout
- otherwise, an exception is thrown

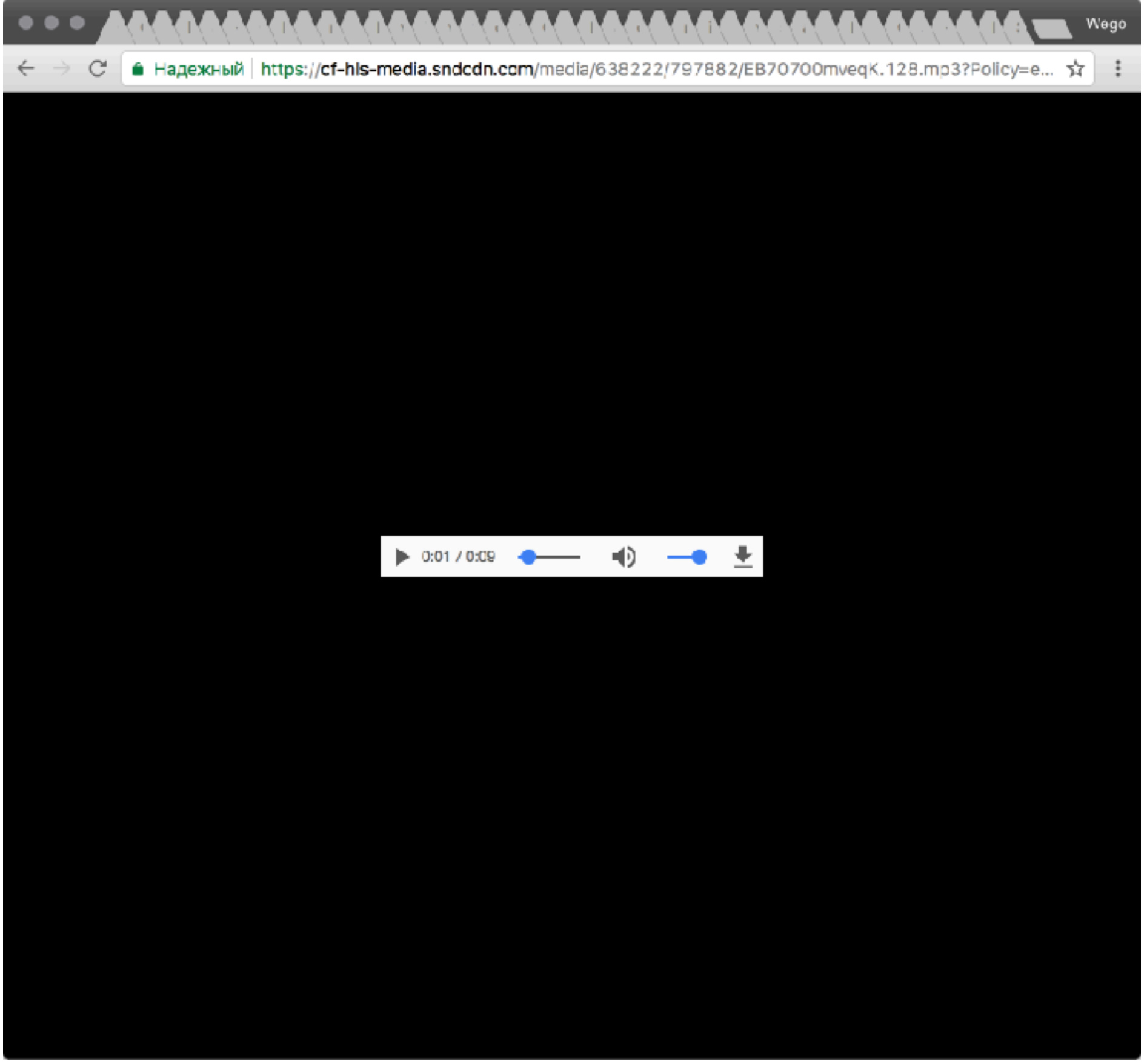
Shortcut for page.mainFrame().waitFor(selectorOrFunctionOrTimeout[, options]).

Audio – using soundcloud.com as example

```
▼<article itemscope itemtype="http://schema.org/MusicRecording">
  ▼<header>
    ▶<h1 itemprop="name">…</h1>
     "
         published on "
     <time pubdate>2015-07-27T11:54:04Z</time>
     <meta itemprop="duration" content="PT00H05M41S">
     <meta itemprop="genre" content="Remix">
     <meta itemprop="interactionCount" content="UserLikes:263437">
     <meta itemprop="interactionCount" content="UserDownloads:0">
     <meta itemprop="interactionCount" content="UserComments:1604">
    ▼<div itemscope itemprop="audio" itemtype="http://schema.org/
     AudioObject">
         <meta itemprop="embedUrl" content="https://w.soundcloud.com/
         player/?
         url=https%3A%2F%2Fapi.soundcloud.com%2Ftracks%2F216542711&auto_p
         lay=false&show_artwork=true&visual=true&origin=schema.org"> == $
         <meta itemprop="height" content="400px">
      </div>
     ▶<div itemscope itemprop="byArtist" itemtype="http://schema.org/
     MusicGroup">…</div>
     ▶<div itemscope itemprop="provider" itemtype="http://schema.org/
     Organization">…</div>
     </header>
  ▶<p>…</p>
```


Audio - using soundcloud.com as example

Audio – using soundcloud.com as example




Audio – using soundcloud.com as example

# Video

seasonvar.ru/serial-14131-Tik.html

SEASONVAR
сериалы - тут

Поиск сериала...

Меню | На чем остановились




## Сериал Тик/The Tick онлайн

Вернуться

Главный герой этого сериала - Тик. Простой, скромный, но при этом невероятно сильный и отважный парень борется с преступностью. Он уже всех пленил своими голубыми глазами, но врагов главный герой сражает далеко не их красотой. Парень прыгает по крышам огромных домов и использует свою суперсилу для того, чтобы останавливать преступные группировки, которые пытаются провернуть очередное мошенническое дело. На кону стоят не только огромные деньги, но и жизни людей, поэтому главный герой не теряет времени даром. Правда, когда дело подходит к минуте отдыха, Тик не отказывается от общения с милыми дамами.

**Оригинал:** The Tick
**Альтернативное название:** Человек-клещ

**Жанр:** комедия, фантастические
**Страна:** США

**Вышел:** 2016
**Режиссер:** Уолли Пфистер

**Рейтинг IMDB:** 7.30 3239
**Рейтинг КиноПоиск:** 6.15 223

**Теги (top 10):** • супергерои • чудеса • деньги • суперспособности + Добавить тег

**В ролях**

- Питер Серафинович
- Гриффин Ньюман
- Вэлори Керри
- Брендан П. Хайнс
- Джеки Эрл Хейли

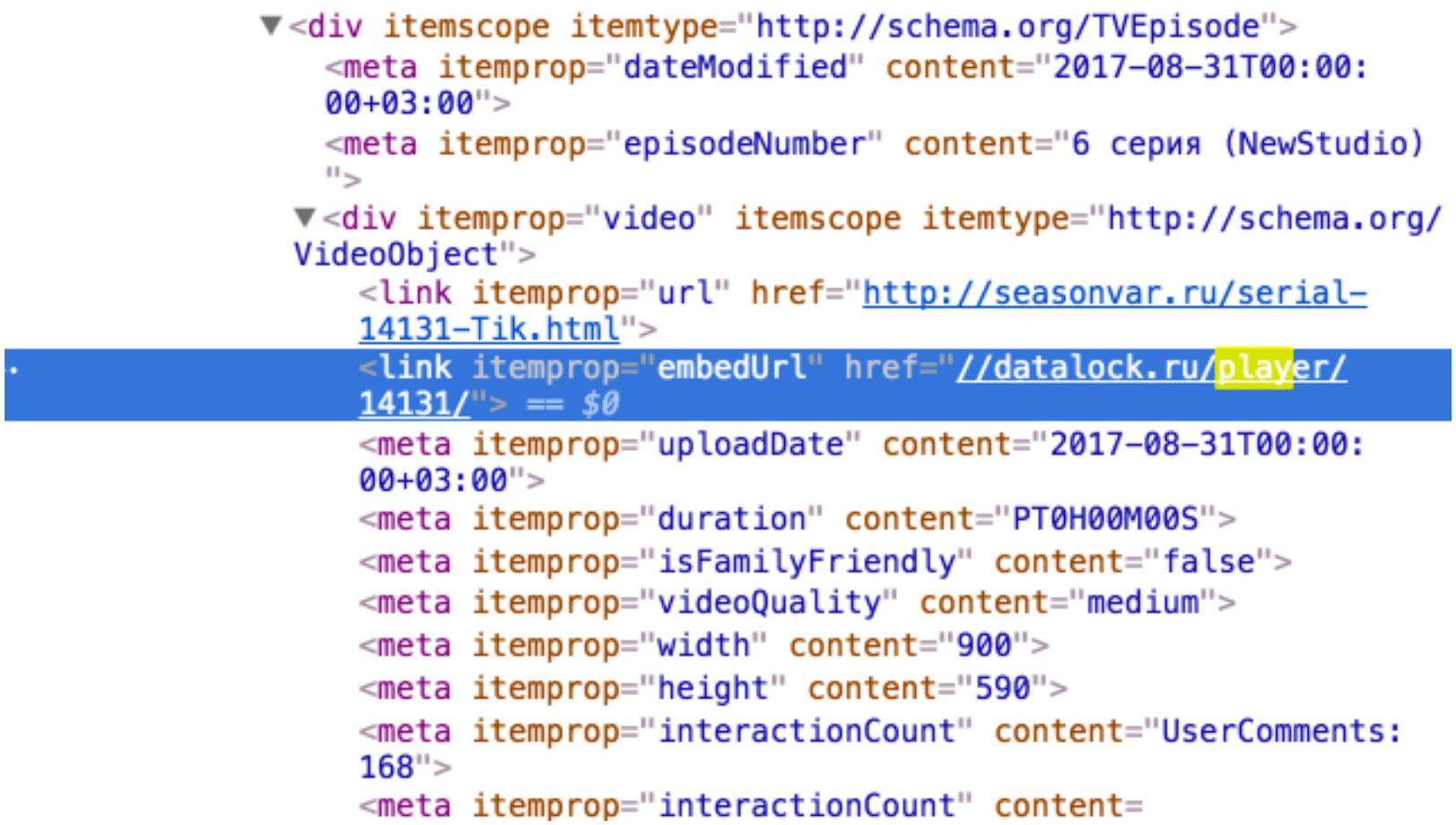
Сезоны

>>> Сериал Тик/The Tick
(31.08.2017 6 серия (NewStudio) из ??)

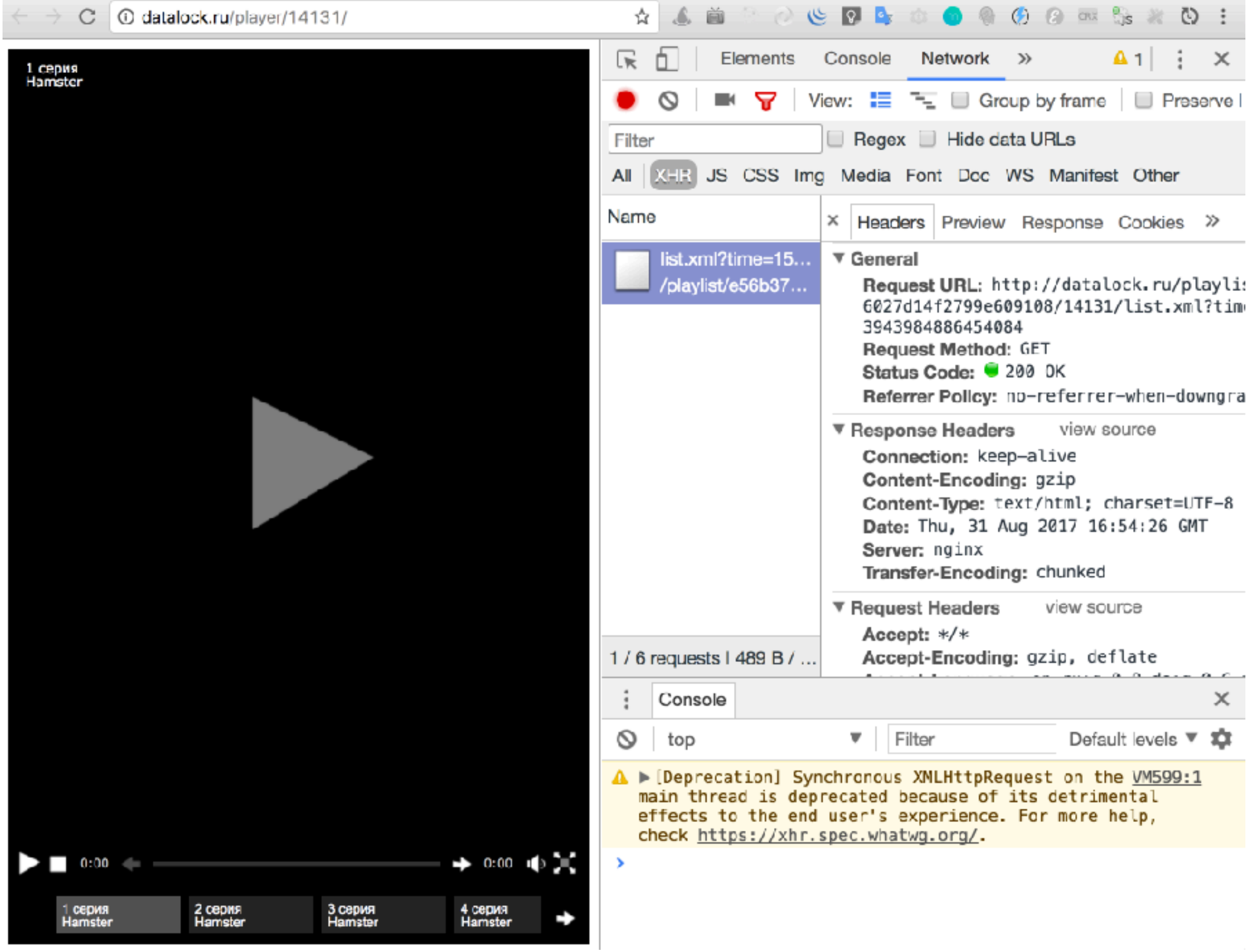
Отметка на серии | Отметка на моменте | Хочу посмотреть

HTML5 | Flash

1 серия SD/HD
Hamster




Video – using seasonvar.ru as example




Video – using seasonvar.ru as example

```
</script>
▼<script type="text/javascript">
    ;eval(function(y,t,u,p){var lIll=0;var ll1I=0;var
    Il1l=0;var ll1l=[];var l1lI=[];while(true)
    {if(lIll<5)l1lI.push(y.charAt(lIll));else
    if(lIll<y.length)ll1l.push(y.charAt(lIll));lIll++;if(ll1
    I<5)l1lI.push(t.charAt(ll1I));else
    if(ll1I<t.length)ll1l.push(t.charAt(ll1I));ll1I++;if(Il1
    l<5)l1lI.push(u.charAt(Il1l));else
    if(Il1l<u.length)ll1l.push(u.charAt(Il1l));Il1l++;if(y.l
    ength+t.length+u.length+p.length==ll1l.length+l1lI.lengt
    h+p.length)break;}var lI1l=ll1l.join('');var
    I1lI=l1lI.join('');ll1I=0;var l1ll=
    [];for(lIll=0;lIll<ll1l.length;lIll+=2){var
    ll11=-1;if(I1lI.charCodeAt(ll1I)%2)ll11=1;l1ll.push(Stri
    ng.fromCharCode(parseInt(lI1l.substr(lIll,2),36)-
    ll11));ll1I++;if(ll1I>=l1lI.length)ll1I=0;}return
    l1ll.join('');}
    ('91b621u212a2933391a363q01311o27212q1b3x2e1d3q01112m3q0
    1322m3x2u37262v222p11323a251s27352116212x25211e2u2911113
    a251s27352116223a1v1112161115 3x3b2q1931261u3u2v312p113w2
```

Video - using seasonvar.ru as example

http://datalock.ru/playlist/e56b374d433046027d14f2799e609108/14131/list.xml


```
▼ {playlist: [{comment: "1 серия<br>Hamster", streamsend: "sec", galab
  ▼ playlist: [{comment: "1 серия<br>Hamster", streamsend: "sec", gala
    ▶ 0: {comment: "1 серия<br>Hamster", streamsend: "sec", galabel: "
    ▼ 1: {comment: "2 серия<br>Hamster", streamsend: "sec", galabel: "
        comment: "2 серия<br>Hamster"
        file: "http://temp-cdn.datalock.ru/fi2lm/e56b374d433046027d14f2
        galabel: "14131_477597"
        streamsend: "sec"
    ▶ 2: {comment: "3 серия<br>Hamster", streamsend: "sec", galabel: "
    ▶ 3: {comment: "4 серия<br>Hamster", streamsend: "sec", galabel: "
    ▶ 4: {comment: "5 серия<br>Hamster", streamsend: "sec", galabel: "
    ▶ 5: {comment: "6 серия<br>Hamster", streamsend: "sec", galabel: "
```


Video – using seasonvar.ru as example

```
36  // saving files to the downloads folder
37  const requestInterceptor = (options) => (e)=> {
38      if (options.fileExtensions.find(ext => ext.test(e.url))) {
39          console.log(e.url);
40          let arr = e.url.split('/');
41          let fileName = arr[arr.length - 1];
42
43          let storedFilePath = path.join(options.folder, fileName);
44          let writeStream  = fs.createWriteStream(storedFilePath);
45          let stream = request(e.url).pipe(writeStream);
46
47          stream.on('finish', function () { e.continue() });
48      } else {
49          e.continue();
50      }
51  }
52
```

```
36  // saving files to the downloads folder
37  const requestInterceptor = (options) => (e)=> {
38      if (options.fileExtensions.find(ext => ext.test(e.url))) {
39          console.log(e.url);
40          let arr = e.url.split('/');
41          let fileName = arr[arr.length - 1];
42
43          let storedFilePath = path.join(options.folder, fileName);
44          let writeStream  = fs.createWriteStream(storedFilePath);
45          let stream = request(e.url).pipe(writeStream);
46
47          stream.on('finish', function () { e.continue() });
48      } else {
49          e.continue();
50      }
51  }
52
```

```
36  // saving files to the downloads folder
37  const requestInterceptor = (options) => (e)=> {
38      if (options.fileExtensions.find(ext => ext.test(e.url))) {
39          console.log(e.url);
40          let arr = e.url.split('/');
41          let fileName = arr[arr.length - 1];
42
43          let storedFilePath = path.join(options.folder, fileName);
44          let writeStream  = fs.createWriteStream(storedFilePath);
45          let stream = request(e.url).pipe(writeStream);
46
47          stream.on('finish', function () { e.continue() });
48      } else {
49          e.continue();
50      }
51  }
52
```

```
// saving files to the downloads folder
const requestInterceptor = (options) => (e)=> {
    if (options.fileExtensions.find(ext => ext.test(e.url))) {
        console.log(e.url);
        let arr = e.url.split('/');
        let fileName = arr[arr.length - 1];

        let storedFilePath = path.join(options.folder, fileName);
        let writeStream  = fs.createWriteStream(storedFilePath);
        let stream = request(e.url).pipe(writeStream);

        stream.on('finish', function () { e.continue() });
    } else {
        e.continue();
    }
}

```

```
53    const responseFormatter = (options)=> ($) => {
54        return $.first().attr('href');
55    }
56
57    const pageHandler = (options)=> async (page) => {
58        await page.click('#layer');
59    }
```

```
63  const scrape = async (options) => {
64      let origin = (new URL(options.uri)).origin;
65      let $ = null;
66      if (options.isDynamic) {
67          isDynamicUrls.set(origin, true);
68      }
69
70      if (!isDynamicUrls.has(origin)) {
71          $ = await requestDriver(options);
72          let items = $(options.selector);
73          if (items.length) {
74              if (options.responseFormatter) {
75                  return options.responseFormatter(options)(items);
76              }
77
78              return items.map(function() {return $(this).text();}).get();
79          }
80
81          isDynamicUrls.set(origin, true);
82      }
83
84      $ = await puppeteerDriver(options);
85
86      if (options.responseFormatter) {
87          return options.responseFormatter(options)($(options.selector));
88      }
89
90      return $(options.selector).map(function() {return $(this).text();}).get();
```

```
63  const scrape = async (options) => {
64      let origin = (new URL(options.uri)).origin;
65      let $ = null;
66      if (options.isDynamic) {
67          isDynamicUrls.set(origin, true);
68      }
69
70      if (!isDynamicUrls.has(origin)) {
71          $ = await requestDriver(options);
72          let items = $(options.selector);
73          if (items.length) {
            if (options.responseFormatter) {
                return options.responseFormatter(options)(items);
            }
79          }
80
81          isDynamicUrls.set(origin, true);
82      }
83
84      $ = await puppeteerDriver(options);
85
86      if (options.responseFormatter) {
87          return options.responseFormatter(options)($(options.selector));
88      }
89
90      return $(options.selector).map(function() {return $(this).text();}).get();
```

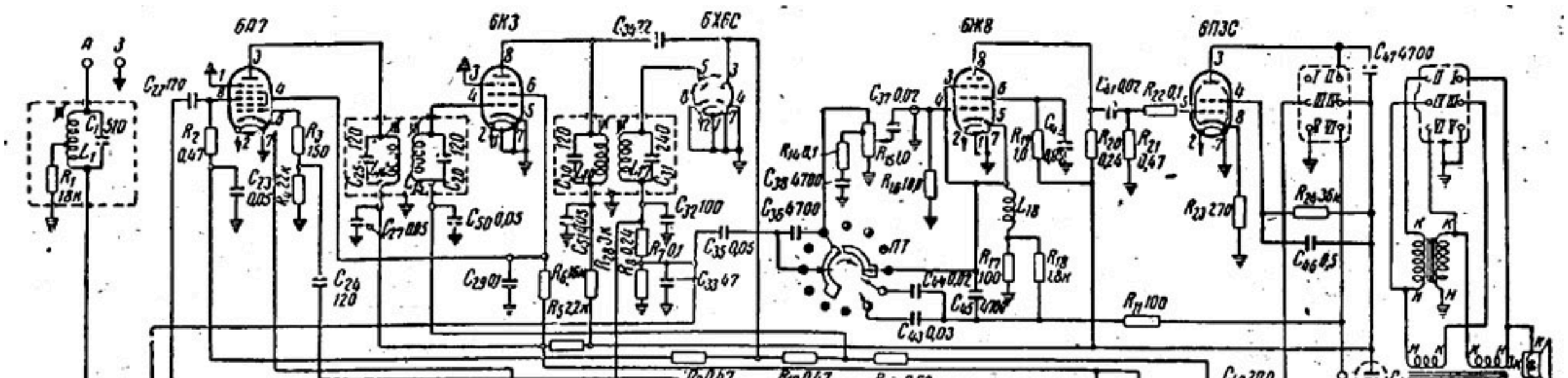
```
18      let interceptor = options.requestInterceptor ?
19          options.requestInterceptor(options) :
20          writeFileInterceptor(options);
21
22      page.on('request', interceptor);
23
```

```
93  const workflow = async (options) => {
94      // get player url
95      let playerUrl = await scrape(Object.assign({},
96          options, {
97              uri: 'http://seasonvar.ru/serial-17080-Grand_Tur-2-season.html',
98              responseFormatter
99          }));
100
101     // get file
102     return await scrape(Object.assign({},
103         options, {
104             uri: `http:${playerUrl}`,
105             requestInterceptor,
106             pageHandler
107         }));
108 }
109
110 export default (params)=> workflow({...options, ...params });
```

```
93  const workflow = async (options) => {
94      // get player url
95      let playerUrl = await scrape(Object.assign({},
96          options, {
97              uri: 'http://seasonvar.ru/serial-17080-Grand_Tur-2-season.html',
98              responseFormatter
99          }));
100
101     // get file
102     return await scrape(Object.assign({},
103         options, {
104             uri: `http:${playerUrl}`,
105             requestInterceptor,
106             pageHandler
107         }));
108 }
109
110 export default (params)=> workflow({...options, ...params });
```

```
3   jasmine.DEFAULT_TIMEOUT_INTERVAL = 999999;
4
    1 Inconclusive
5   describe('puppeteer download video from seasonvar.ru', () => {
        Inconclusive | Debug
6       test('try download file', async () => {
7           let options = {
8               uri: 'http://seasonvar.ru/serial-17080-Grand_Tur-2-season.html',
9               launch: { headless: true },
10              selector: '[itemprop="embedUrl"]',
11              blacklist: [
12              ],
13              setViewport: { width: 1240, height: 680 },
14              isDynamic: false,
15              fileExtensions: [/mp4/, /flv/, /m3u8/, /m4s/],
16              folder: './assets/download',
17          }
18
19          await scrape();
20
21          let filePath = fs.readdirSync(options.folder).find(file => {
22              return options.fileExtensions.find(e=> e.test(file));
23          });
24
25          return expect(filePath).not.toBeUndefined();
26      });
27  });
28
```

# Sitemap, Metadata, SiteSchema



# web-auto-extractor

Parse semantically structured information from any HTML webpage.

Supported formats:

- Encodings that support **Schema.org** vocabularies:
  - Microdata
  - RDFa-lite
  - JSON-LD
- Random Meta tags

Popularly, many websites mark up their webpages with Schema.org vocabularies for better SEO. This library helps you parse that information to JSON

npm install web-auto-extractor

# web-auto-extractor - достаем метадату

```
// IF CommonJS
var WAE = require('web-auto-extractor').default
// IF ES6
import WAE from 'web-auto-extractor'

var parsed = WAE().parse(sampleHTML)
```

# web-auto-extractor - достаем метадату

```
<div itemscope itemtype="http://schema.org/Product">
  <span itemprop="brand">ACME</span>
  <span itemprop="name">Executive Anvil</span>
  <img itemprop="image" src="anvil_executive.jpg" alt="Executive Anvil
  <span itemprop="description">Sleeker than ACME's Classic Anvil, the
    Executive Anvil is perfect for the business traveler
  looking for something to drop from a height.
</span>
Product #: <span itemprop="mpn">925872</span>
<span itemprop="aggregateRating" itemscope itemtype="http://schema.or
  <span itemprop="ratingValue">4.4</span> stars, based on <span itemp
    </span> reviews
</span>

<span itemprop="offers" itemscope itemtype="http://schema.org/Offer">
  Regular price: $179.99
  <meta itemprop="priceCurrency" content="USD" />
```

# web-auto-extractor - достаем метадату


```
{
  "microdata": {
    "Product": [
      {
        "@context": "http://schema.org/",
        "@type": "Product",
        "brand": "ACME",
        "name": "Executive Anvil",
        "image": "anvil_executive.jpg",
        "description": "Sleeker than ACME's Classic Anvil, the\n    Executive Anvil is perfect for the business trave
        "mpn": "925872",
      "aggregateRating": {
        "@context": "http://schema.org/",
        "@type": "AggregateRating",
        "ratingValue": "4.4",
        "reviewCount": "89"
      },
      "offers": {
        "@context": "http://schema.org/",
        "@type": "Offer",
        "priceCurrency": "USD",
        "price": "119.99",
        "priceValidUntil": "5 November!",
        "seller": {
          "@context": "http://schema.org/",
          "@type": "Organization",
          "name": "Executive Objects"
        },
        "itemCondition": "http://schema.org/UsedCondition",
        "availability": "http://schema.org/InStock"
      }
    }
```

# open-graph-scraper

A simple node module for scraping Open Graph and Twitter Card info off a site.

npm install open-graph-scraper

# open-graph-scraper - достаем метадату

```
const ogs = require('open-graph-scraper');
const options = {'url': 'http://ogp.me/', 'timeout': 4000};
ogs(options, function (error, results) {
  console.log('error:', error); // This is returns true or false.
  console.log('results:', results);
});
```

# open-graph-scraper - достаем метадату

```
{
  data: {
    ogTitle: 'Open Graph protocol',
    ogType: 'website',
    ogUrl: 'http://ogp.me/',
    ogDescription: 'The Open Graph protocol enables any web page to become a
    ogImage: {
      url: 'http://ogp.me/logo.png',
      width: '300',
      height: '300',
      type: 'image/png'
    }
  },
  success: true
}
```

# sitemapper

Parse through a sitemaps xml to get all the urls for your crawler

npm install sitemapper --save

# sitemapper - достаем ссылки

```
import Sitemapper from 'sitemapper';

const Google = new Sitemapper({
  url: 'https://www.google.com/work/sitemap.xml',
  timeout: 15000, // 15 seconds
});

Google.fetch()
  .then(data => console.log(data.sites))
  .catch(error => console.log(error));


// or


const sitemapper = new Sitemapper();
sitemapper.timeout = 5000;

sitemapper.fetch('http://wp.seantburke.com/sitemap.xml')
  .then(({ url, sites }) => console.log(`url:${url}`, 'sites:', sites))
  .catch(error => console.log(error));
```

# Где взять Sitemap ссылку?

example.com/robot.txt

OR

example.com/sitemap.xml

OR

google.com + site:example.com inurl:gz inurl:xml inurl:sitemap

???

index.js — presentation-one

dex.js .../seasonvar-demo | JS index.js .../scheme-demo × | JS request-driver.js .../request-puppeteer-together-demo

```
1  import request from 'request-promise';  1.1M (gzipped: 320.5K)
2  import cheerio from 'cheerio';  395.8K (gzipped: 113.3K)
3  import WAE from 'web-auto-extractor';  483K (gzipped: 142.2K)
4
5  const options = {
6    uri: "https://www.google.com",
7    transform: (body) => ({
8        scheme: WAE().parse(body),
9        $: cheerio.load(body)
10     })
11 };
12
13 const scrape = async (options) => {
14     try {
15         // console.log('uri', options.uri);
16         let result = await request(options);
17         // console.log(result.scheme);
18         return result;
19     } catch (err) {
20         return {
21             scheme: {},
22             body: {}
23         }
24     }
25 };
26
27 export default (params)=> scrape({...options, ...params });
28
```

1: zsh

```
# macbook at abc in ~/sandbox/present
ation/presentation-one on git:master
* [18:20:08]
➜
```

# The top 500 sites on the web

Global By Country **By Category**

## Sub-Categories ( 22 )

- Alternative ( 81 )
- Analysis and Opinion ( 274 )
- Breaking News ( 87 )
- By Category ( 0 )
- By Region ( 0 )
- By Subject ( 0 )
- Chats and Forums ( 7 )
- Colleges and Universities ( 747 )
- Current Events ( 110 )
- Directories ( 26 )
- Extended Coverage ( 21 )
- Headline Links ( 58 )
- Internet Broadcasts ( 25 )
- Journalism ( 1,051 )
- Magazines and E-zines ( 68 )
- Media Industry ( 431 )
- Museums and Archives ( 60 )
- Newspapers ( 2,984 )
- Personalized News ( 12 )
- Satire ( 45 )
- Weather ( 146 )
- Weblogs ( 59 )

Showing 50 of 500 results



| | Site | Daily Time on Site | Daily Pageviews per Visitor | % of Traffic From Search | Tot |
|---|---|---|---|---|---|
| 1 | Https://www.reddit.com/<br>User-generated news links. Votes promote stories to the front page. | 15:54 | 10.22 | 16.50% | |
| 2 | Nytimes.com | 3:49 | 2.14 | 25.10% | |

# How often are sitemap.xml checked for updates by crawlers?

Let's try an empirical approach.

In the access logs for my site, I see 55 sitemap requests over the last 33 days. Out of those 55, 30 are from Googlebot, 21 from msnbot and the remaining four are from Exabot. (I've only submitted the sitemap manually to Google; the others have found it through robots.txt.)

So that's one data point for about "every day", at least for Google and for a smallish site like mine. Although I should note that I regenerate the sitemap daily, so it's possible that Googlebot is simply observing a pattern in the last modification timestamps and adapting to it.

If you want to directly inform search engines that your sitemap has been updated, you can do so by sending an HTTP ping. This may make the search engines reload your sitemap sooner, although of course there are no guarantees.

https://webmasters.stackexchange.com/questions/28781/how-often-are-sitemap-xml-checked-for-updates-by-crawlers

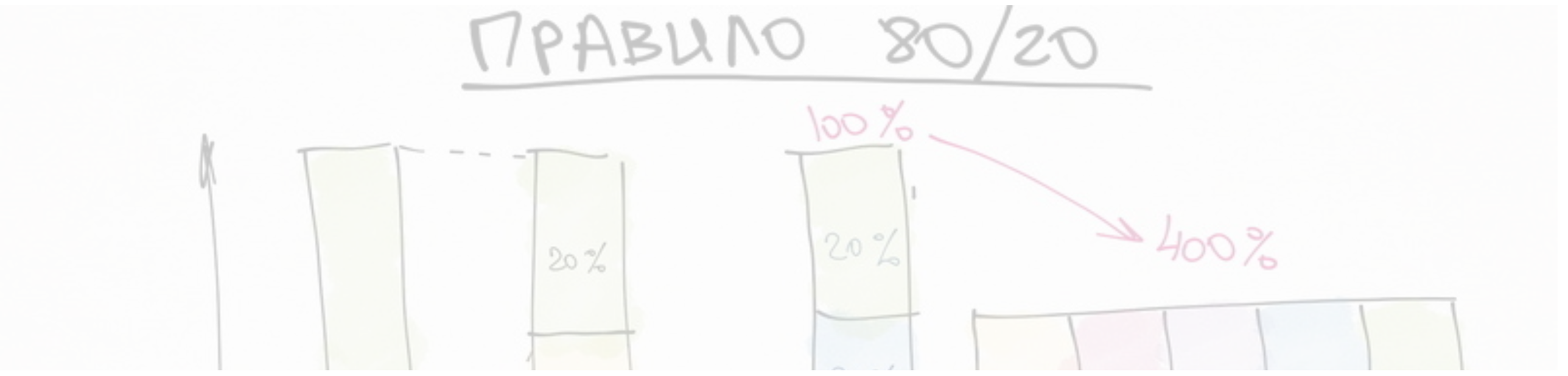
# То есть

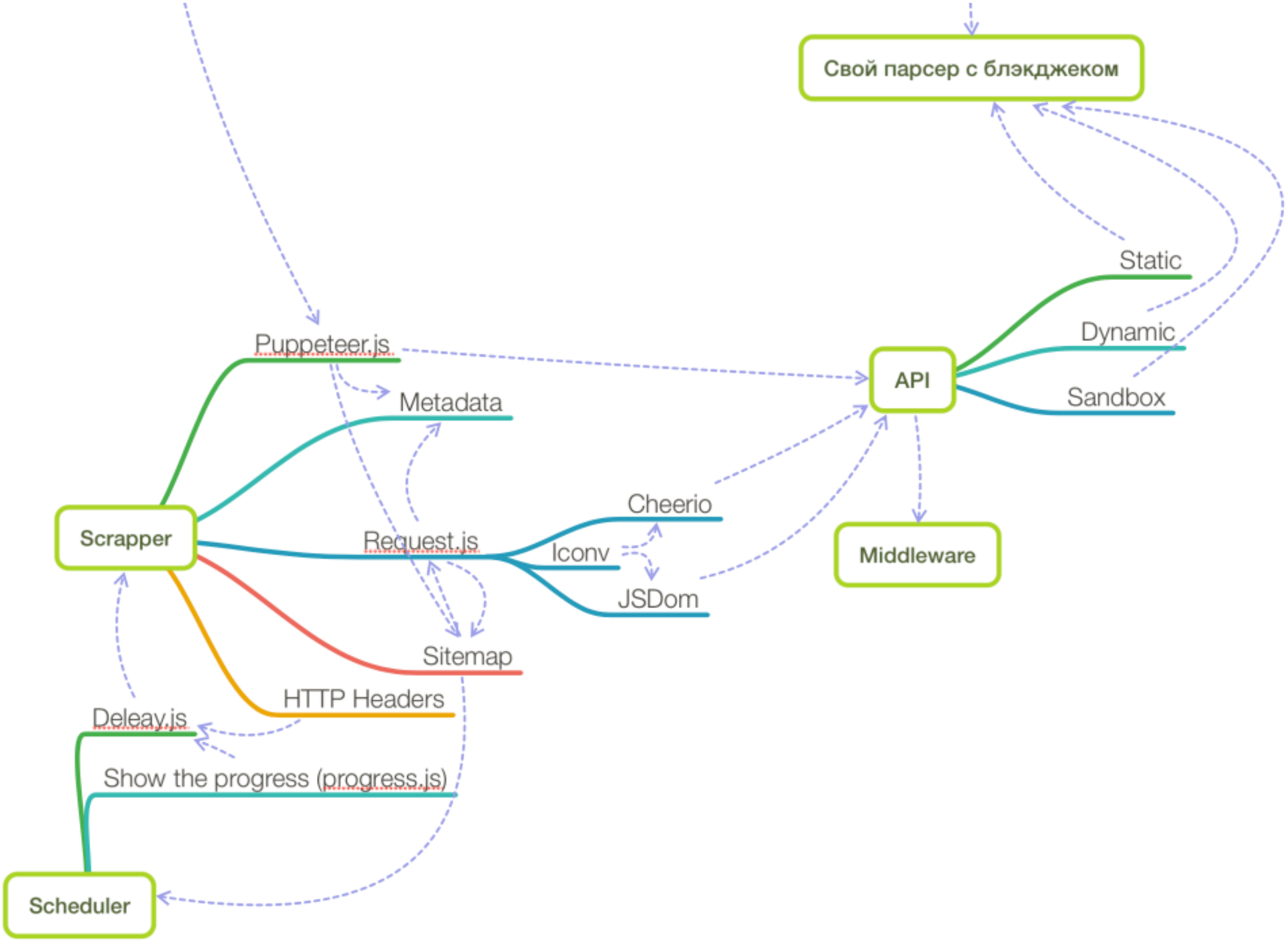
| | Last modification date | Change freq. | *Priority* |
|---|---|---|---|
| | 2017-12-07T02:59:47+03:00 | daily | 1.00 |
| | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| | 2017-12-07T02:59:47+03:00 | daily | 0.50 |

**Google Sitemap http://**
**This Google Sitemap file wa**
**Number of URLs** in this **Goo**
Click on the table headers to change sorting

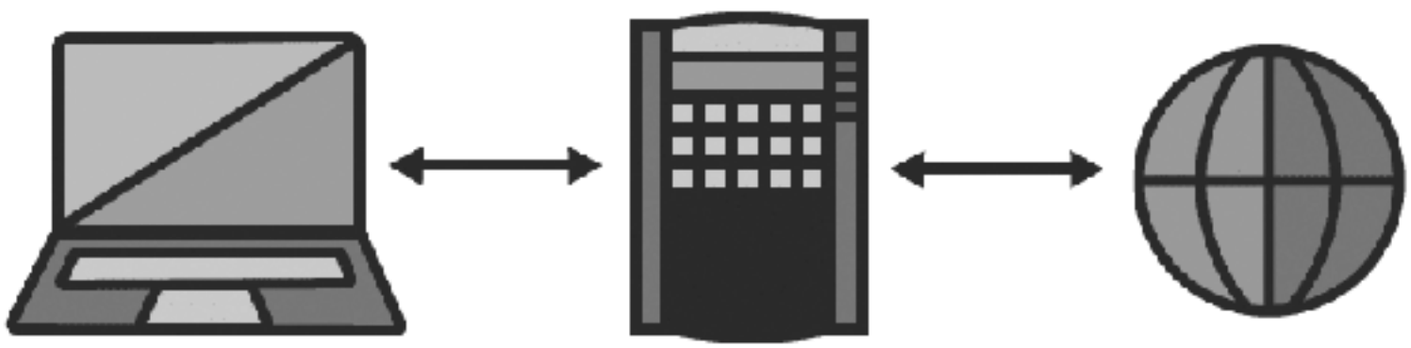
| Sitemap URL | | | |
|---|---|---|---|
| http://seasonvar.ru/ | | | |
| http://seasonvar.ru/rss.php | | | |
| http://seasonvar.ru/?mod=login | | | |
| http://seasonvar.ru/s/online_besplatno | | | |
| http://seasonvar.ru/serial-4105-Dol_sha | | | |
| http://seasonvar.ru/s/smotret_onlayn_b | | | |
| http://seasonvar.ru/s/posmotr_onlayn | | | |
| http://seasonvar.ru/s/smotret_serialy_b | | | |
| http://seasonvar.ru/s/smotret_serialy_o | | | |
| http://seasonvar.ru/s/serialy_onlayn_besplatno.html | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| http://seasonvar.ru/s/onlayn_bezplatno.html | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| http://seasonvar.ru/s/onlayn_besplatno_bez_registracii.html | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| http://seasonvar.ru/s/smotret_online.html | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| http://seasonvar.ru/s/posmotret_serial.html | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| http://seasonvar.ru/s/novye_serialy.html | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| http://seasonvar.ru/s/serials.html | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| http://seasonvar.ru/s/besplatno_serialy.html | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| http://seasonvar.ru/s/posmotret_onlayn.html | 2017-12-07T02:59:47+03:00 | daily | 0.50 |
| http://seasonvar.ru/serial-1-4400-1-season.html | 1970-01-01T03:00:00+03:00 | daily | 0.50 |
| http://seasonvar.ru/serial-2-4400-2-season.html | 1970-01-01T03:00:00+03:00 | daily | 0.50 |
| http://seasonvar.ru/serial-3-4400-3-season.html | 1970-01-01T03:00:00+03:00 | daily | 0.50 |
| http://seasonvar.ru/serial-4-4400-4-season.html | 2017-09-25T00:00:00+03:00 | daily | 0.50 |
| http://seasonvar.ru/serial-5-Heroes-1-season.html | 2017-08-03T00:00:00+03:00 | daily | 0.50 |
| http://seasonvar.ru/serial-6-Gerci-2-season.html | 1970-01-01T03:00:00+03:00 | daily | 0.50 |
| http://seasonvar.ru/serial-7-Gerci-3-season.html | 1970-01-01T03:00:00+03:00 | daily | 0.50 |

# Full workflow




Свой парсер с блэкджеком
Static
Dynamic
Sandbox
API
Middleware
Puppeteer.js
Metadata
Scrapper
Request.js
Cheerio
Iconv
JSDom
Sitemap
HTTP Headers
Deleay.js
Show the progress (progress.js)
Scheduler

# Proxy



# superagent-proxy

`Request#proxy(uri)` superagent extension

This module extends `superagent`'s `Request` class with a `.proxy(uri)` function. This allows you to proxy the HTTP request through a proxy of some kind.
It is backed by the `proxy-agent` module, so see its README for more details.

```
npm install superagent-proxy
```

# auto-proxy

AutoProxy is a nodejs module that ships with web-scrapers out-of-the-box!
That means you don't have to spend hours goog'ling proxysites.
Just fire up AutoProxy and let it search the proxies for you.
However if you've still got one of these old-style `IP:PORT` lists on your 16MB flash-drive, AutoProxy can handle that too.

npm install auto-proxy

# How to add Tor proxy when scraping using node.io?

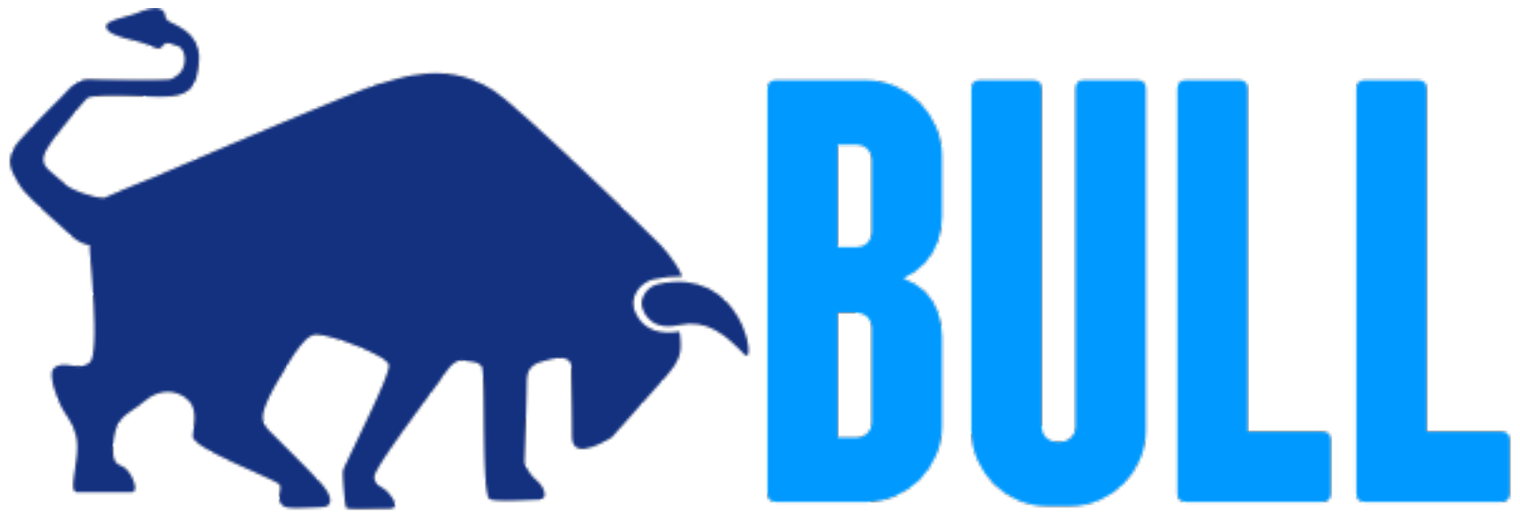
Install tor and polipo. Polipo to connect to Tor and Node.IO will use http proxy polipo provide. It seem simple than what I think. And set proxy for scraper

https://stackoverflow.com/questions/19221903/how-to-add-proxy-like-tor-when-scraping-using-node-io

```
const browser = await puppeteer.launch({
    args: [ '--proxy-server=127.0.0.1:9876' ]
});
```

https://github.com/GoogleChrome/puppeteer/issues/678

# Scheduler




npm install bull

```
import Queue from 'bull';

let store = new Map();

const add = (name, process) => {
    let item = new Queue(name, 'redis://127.0.0.1:6379');
    item.process(process);
    store.set(name, item);

    return item;
}

const get = (name) => {
    return store.get(name);
}

const toQueue = (name, object) => {
    store.get(name).add(object);
}

const remove = (name) => {
    store.get(name).close();
    store.delete(name);
}

export default {
    add,
    get,
    toQueue,
    remove
}
```

```
import Queue from 'bull';

let store = new Map();

const add = (name, process) => {
    let item = new Queue(name, 'redis://127.0.0.1:6379');
    item.process(process);
    store.set(name, item);

    return item;
}
```

```
const processItem = async (params) => {
    let item = await findItem(params);
    let result = await sendEmail({to: item.email});
    return result;
};

let faxQueue = queue.add('items que', processItem);


const someApiAction = async ({body}, next) => {
    let item = await addItem(body);

    faxQueue.add({id: item._id});
};
```

# node-schedule

Node Schedule is a flexible cron-like and not-cron-like job scheduler for Node.js. It allows you to schedule jobs (arbitrary functions) for execution at specific dates, with optional recurrence rules.

It only uses a single timer at any given time (rather than reevaluating upcoming jobs every second/minute).

npm install node-schedule

```
import { scheduleJob } from 'node-schedule';

let jobs = [];

function startJob(job, schedule = '*/40 * * * * *') {
    jobs.push(scheduleJob(schedule, job));
}

function stopJobs() {
    jobs.forEach(e=> e.cancel());
    jobs.length = 0;
}

export default {
    startJob,
    stopJobs
}

```

```
const processItemsDailyAt4AM = async (params) => {
    let item = await findItem(params);
    let result = await sendEmail({to: item.email});
    return result;
};

startJob(processItemsDailyAt4AM, '*/40 * * * *');
```

# История про 3ГГБ запрос

```
process.stdin.resume();

/* your app */

function exitHandler(type, err) {
    console.log(err);
    if (type) {
        server.close(function() {
            console.log("Finished all requests");
        });

        /* PLACE TO CLOSE EVERYTHING OUTSIDE THE APP */
    } else {
        process.exit();
    }
}

[
    { name: 'exit', type: 'cleanup' },
    { name: 'SIGINT' },
    { name: 'SIGUSR1' },
    { name: 'SIGUSR2' },
    { name: 'uncaughtException' }
].forEach((e) => {
    process.on(e.name, exitHandler.bind(null, e.type));
});
```

```
18  [
19      { name: 'exit', type: 'cleanup' },
20      { name: 'SIGINT' },
21      { name: 'SIGUSR1' },
22      { name: 'SIGUSR2' },
23      { name: 'uncaughtException' }
24  ].forEach((e) => {
25      process.on(e.name, exitHandler.bind(null, e.type));
26  });
```

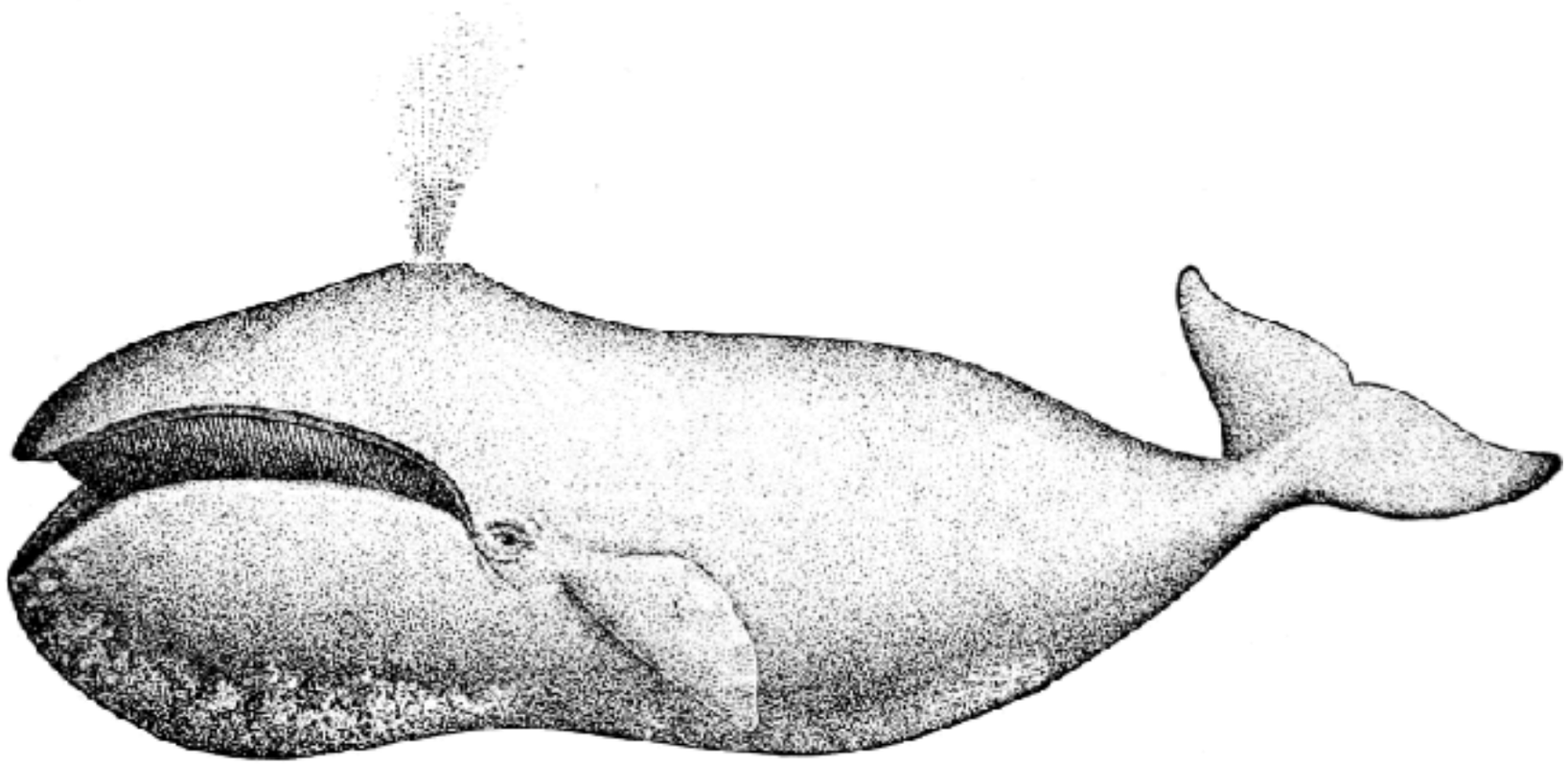

npm install x-ray

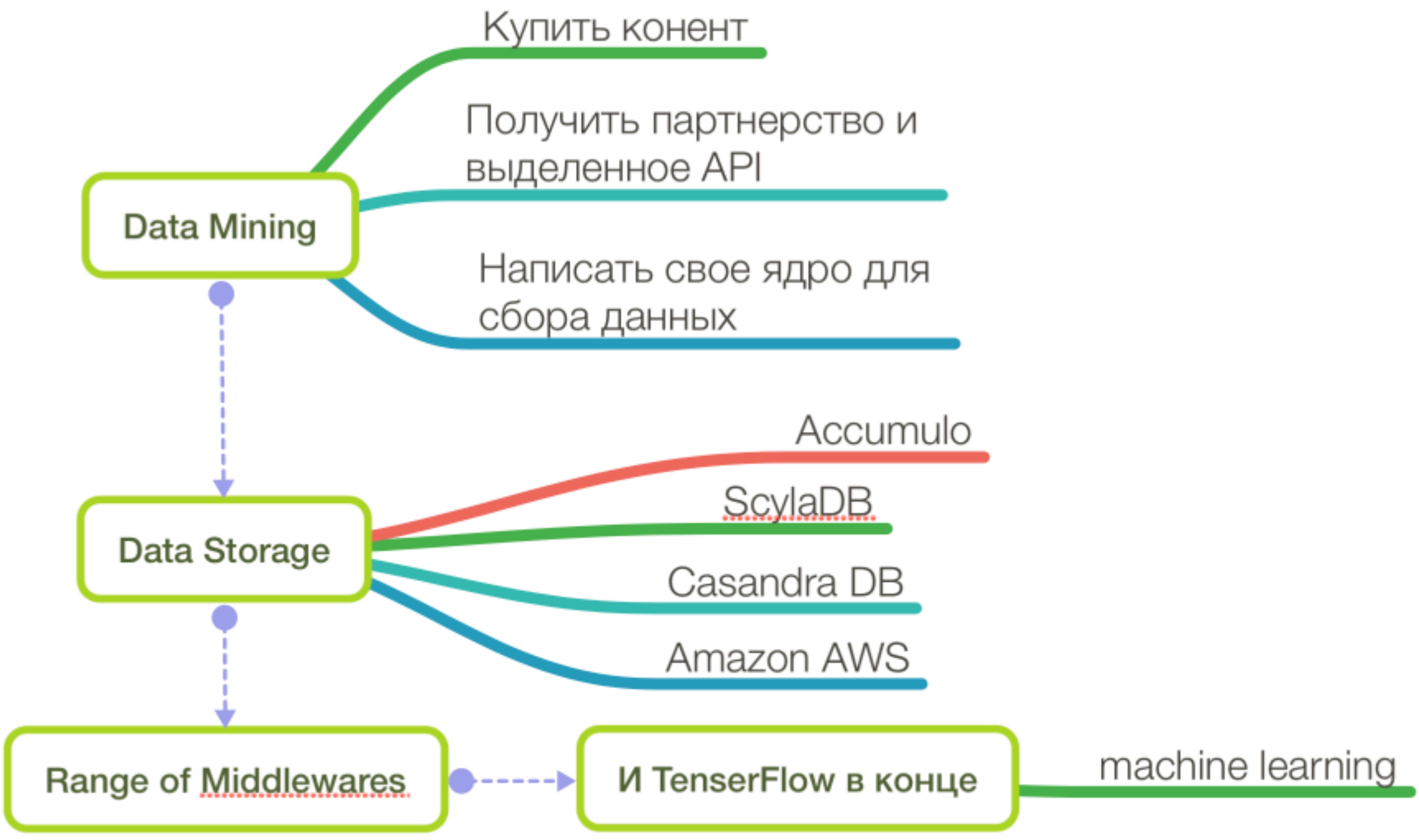
# Features

- **Flexible schema:** Supports strings, arrays, arrays of objects, and nested object structures. The schema is not tied to the structure of the page you're scraping, allowing you to pull the data in the structure of your choosing.
- **Composable:** The API is entirely composable, giving you great flexibility in how you scrape each page.
- **Pagination support:** Paginate through websites, scraping each page. X-ray also supports a request `delay` and a pagination `limit`. Scraped pages can be streamed to a file, so if there's an error on one page, you won't lose what you've already scraped.
- **Crawler support:** Start on one page and move to the next easily. The flow is predictable, following a breadth-first crawl through each of the pages.
- **Responsible:** X-ray has support for concurrency, throttles, delays, timeouts and limits to help you scrape any page responsibly.
- **Pluggable drivers:** Swap in different scrapers depending on your needs. Currently supports HTTP and PhantomJS driver drivers. In the future, I'd like to see a Tor driver for requesting pages through the Tor network.

```
var Xray = require('x-ray');
var x = Xray();

x('https://blog.ycombinator.com/', '.post', [{
  title: 'h1 a',
  link: '.article-title@href'
}])
  .paginate('.nav-previous a@href')
  .limit(3)
  .write('results.json')
```

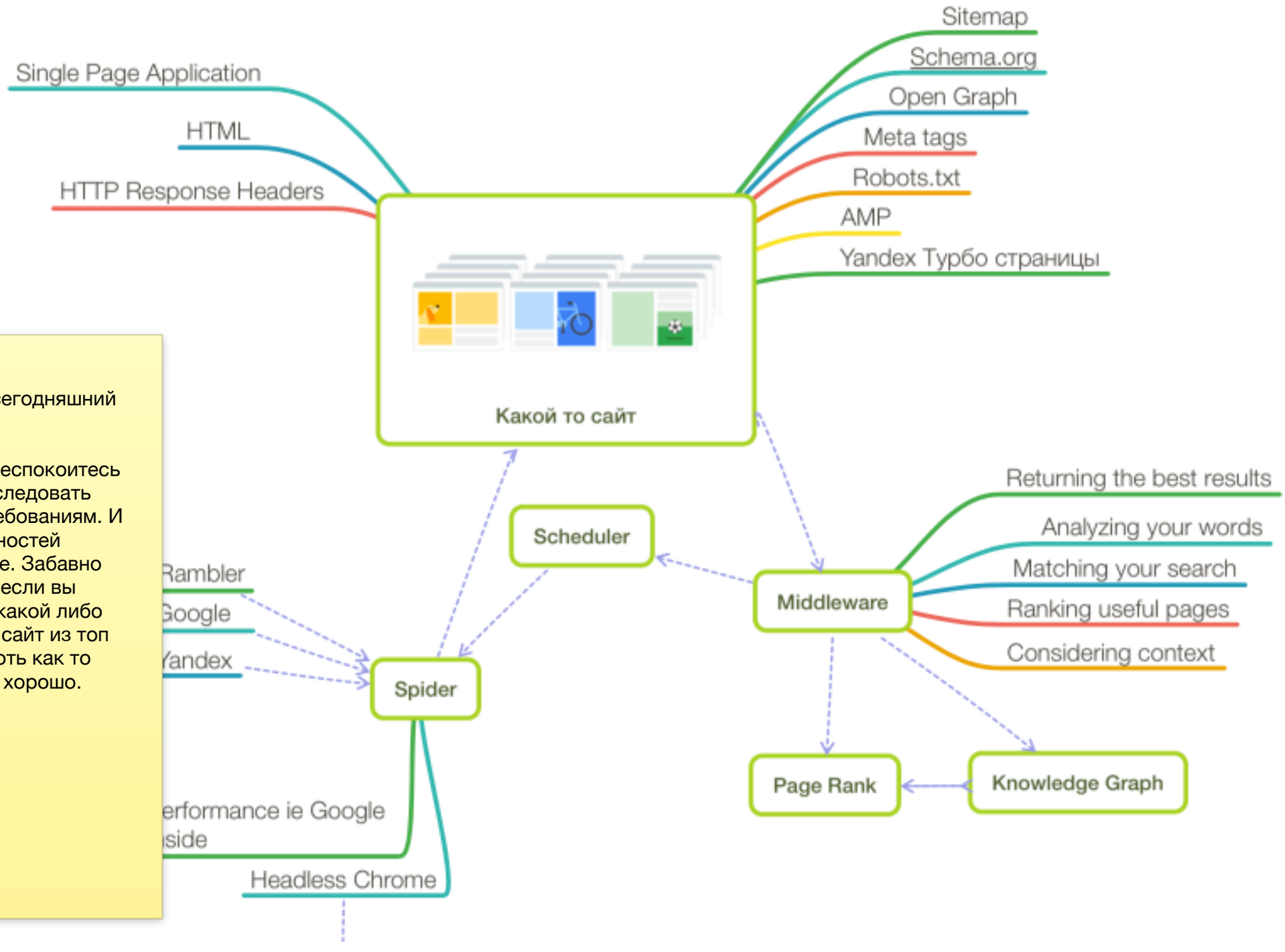
# Как действуют Киты




Data Mining
Купить конент
Получить партнерство и выделенное API
Написать свое ядро для сбора данных
Data Storage
Accumulo
ScylaDB
Casandra DB
Amazon AWS
Range of Middlewares
И TenserFlow в конце
machine learning

Давайте представим сегодняшний интернет.

Поисковики если вы беспокоитесь о сео требуют от вас следовать довольно жестким требованиям. И требований и возможностей становится все больше. Забавно или нет, скорее всего если вы собираетесь спарить какой либо популярный сайт, или сайт из топ 10 выдачи, он будет хоть как то оптимизирован. И это хорошо.



# Как защитить свой контент

https://medium.com/@Jscrambler/protect-your-site-against-web-scraping-ea504a569a77

В первую очередь думайте о своих пользователях


Prevent denial of service (DoS) attacks

Use Cross Site Request Forgery
(CSRF) tokens


Use temporary path to the files

Prevent hotlinking


Blacklist or Whitelist
specific IP addresses


Create “honeypots”

Change DOM structure frequently

attributes/properties

events

Use Shadow Root

<my-element>

DRM, certificates and so on

www.bbc.com/news/av/business-42083002/new-tax-on-dubai-gold

Business | Market Data | Markets | Global Trade | Companies | Entrepreneurship | Technology of Business | More


**New tax on Dubai gold**

24 Nov | Business


**Behind the scenes at the Dubai Airshow**

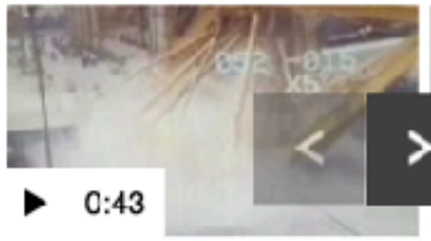
13 Nov | Business


**Fire rips through Dubai's Torch Tower**

04 Aug | Middle East

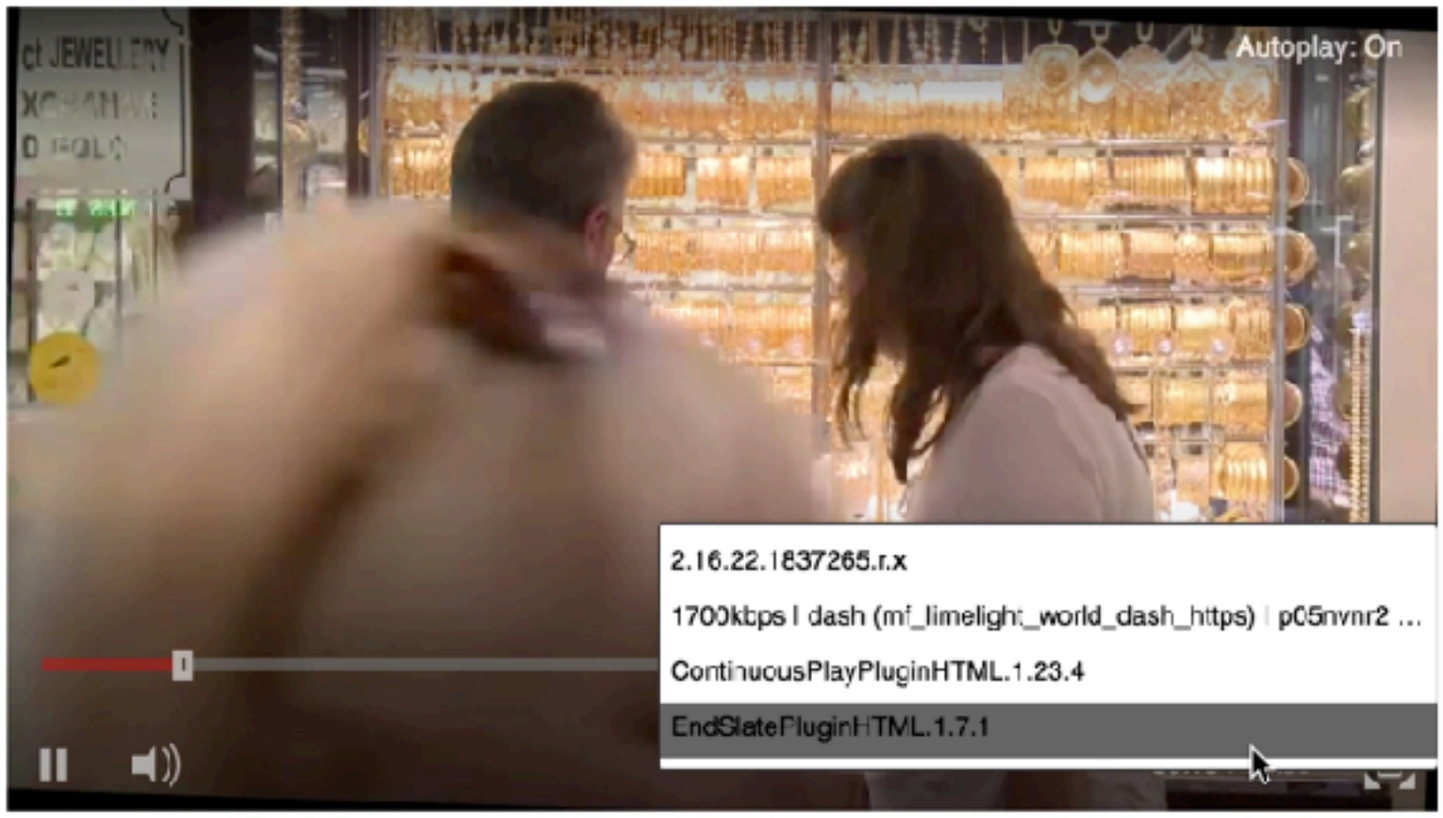

**Khorgos Gateway: Where East meets West**

31 May | Asia




**Moment huge Dubai crane collapses**

09 May | Middle East




# New tax on Dubai gold

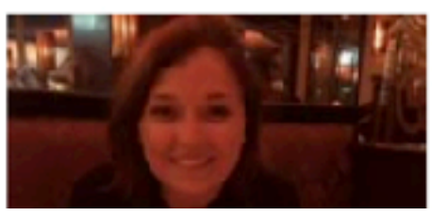
Dubai calls itself City of Gold. Traders there buy and sell $75bn (£56bn) worth of the precious metal every year. Part of the reason for its success is that gold is untaxed, and therefore cheap.

But from the start of next year, the government is imposing a value added tax on gold sales. Jeremy Howell has more.

24 Nov 2017 | Business

Share

**MORE ON:** Dubai



savefrom.net

Установить Вебмастерам Помощь

Просто вставь ссылку

Поддерживаемые ресурсы | Как использовать сервис? | Savefrom.net ВКонтакте

Примечание: чтобы узнать особенности скачивания с определенного ресурса, щелкните по его названию.

- sendspace.com
- youtube.com
- vimeo.com
- smotri.com
- break.com
- sevenload.com
- yandex.video
- dailymotion.com
- mail.ru
- tvigle.ru
- facebook.com
- odnoklassniki.ru *
- veojam.com
- ntv.ru
- autoplustv.ru
- livejournal.com
- soundcloud.com
- 1tv.ru
- vesti.ru
- russiaru.net
- vk.com
- liveinternet.ru

Make the App overcomplicated

Скачать с Контакта, Ютуб, О | YouTube | Wego

Надежный | https://www.youtube.com

YouTube RU | Введите запрос

## Рекомендованные


**Вот почему Tesla Model 3 - крутейшая машина 2017**

Doug DeMuro Русская Версия
189 тыс. просмотра •
6 дней назад


**Good Vibes Radio • MUSIC LIVE STREAM 24/7 - Best**

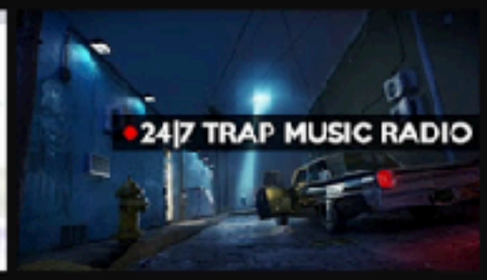
Popularity Music
Зрителей: 192
СЕЙЧАС В ПРЯМОМ ЭФИРЕ


**Christmas Music Mix 🎅 Best Trap - Dubstep - EDM 🎅**

Magic Music
167 тыс. просмотров •
3 дня назад


**COUB лучшее за неделю декабрь 2017 | coub best**

just enjoy
152 тыс. просмотров •
19 часов назад




**TRAP MUSIC RADIO | LIVE STREAM 24/7 - Best & New**

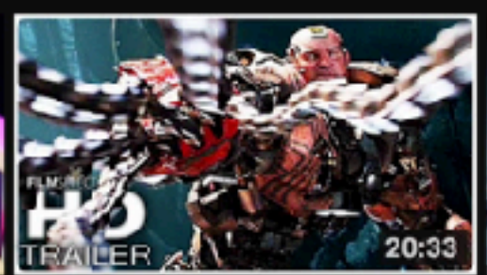
Popularity Music
Зрителей: 83
СЕЙЧАС В ПРЯМОМ ЭФИРЕ


**ПОГРУЖЕНИЕ (2018) [ Официальный**

КиноГуру Все Трейлеры
207 тыс. просмотров •
2 недели назад


**Машинный интеллект и машинное обучение.**

Контент король
38 тыс. просмотров •
Год назад




**ТОП ЛУЧШИХ ФАНТАСТИЧЕСКИХ**

FilmSelect Россия
119 тыс. просмотров •
1 день назад


**Илон Маск. Бесконечная Питьевая вода из**

Iron Cult
32 тыс. просмотров •
1 день назад


**Детский КВН 2017 - 1 сезон 3 выпуск (03.12.2017) ИГРА**

Официальный канал КВН
1,3 тыс. просмотра •
1 час назад

ЕЩЁ

## Миксы YouTube Плейлисты, подобранные по стилю или исполнителю

Скачать с Контакта, Ютуб, О | Машинный интеллект и маш | Wago

Надежный | https://ru.savefrom.net

325px × 730px

Elements Console Sources Network Performance Memory Application Security Audits » 4 2

View: Group by frame Preserve log Disable cache Offline Online

Filter Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other

Name

ru.savefrom.net

ima3.js
imasdk.googleapi...

modernizr-2.8.3.m...
/js/libs

jquery.min.js
ajax.googleapis.c...

scripts_1.19.js?v=1
/js

styles.css?v=1.113
/css

savefrom_6.29.mi...
/js

share12_ru_2.js?v=2

35 requests | 683 KB tran...

Headers Preview Response Cookies Timing

```
<!--/Error message for IE 6-->

  <!--Header-->
  <header class="header header_v2">
    <div class="page-width-inner g-row_inline">
      <div class="header__left g-col c3">
        <span class="menu-btn menu-btn_v2"></span>
        <a name="logo" class="logo" href="">Savefrom.net</a>
      </div>
            <div id="nav_top" class="g-col c9">
          <div class="nav-top-2 right">
            <a href="/user.php?vid=307" class="nav-top-2__item">Установить</a><a href="/webmaster
          </div>
        </div>
  </header>

<div id="main" class="main-block main-block_v2">
  <noscript>
    <div class="wrapper"><p class="javascript-error">SaveFrom.net использует JavaScript для отобр
  </noscript>

        <script>
        function newMainForm (oldForm, modifyFn, callbackFn) {
          var _this = this, helperEnabled = false;

          if (oldForm || _sf.oldMainForm) {
```

Console

top Filter All levels

```
zOqf6+rfY+rJ1krfYRbRYuqRYSYRYuz1zOqfzOEpzOqf5+rfY+rf1frfYtoRYuqRY9NRYuzNzOqfzOzZzOqflbrfY+rJ1krfYtoRYuqRYSYRYuzlzOqfzOE
pzOqfEbrfY+rf1frfYu1RYuqRY9NRYuzlzOqfzOzZzOqfEkrfY+rJ1krfYRrRYuqRYSYRYuz7zOqfzOEpzOqfEkrfY+rf1frfYSARYuqRY9NRYuzOzOqfzO
zZzOqfrfrfY+rJ1krfYSARYuqRYSYRYuq2zOqfzOEpzOqfE+rfY+rf1frfYuMRYuqRY9NRYuq2zOqfzOzZzOqfskrfY+rJ1krfYaYRYuqRYSYRYuzuzOqfz
OEpzOqfskrfY+rf1frfYSSRYuqRY9NRYuzZzOqfzOzZzOqf1frfY+rJ1krfYSSRYuqRYSYRYuzTzOqfzOEpzOqfDkrfY+rf1frfYarRYuqRY9NRYuzTzOqf
zOzZzOqf7+rfY+rJ1krfYR1RYuqRYSYRYuzrzOqfzOEpzOqf7+rfY+rf1frfYSxRYuqRY9NRYuz9zOqfzOzZzOqf6brfY+rJ1krfYSxRYuqRYSYRYuz/zOq
fzOEpzOqfmbrfY+rf1frfYaXRYuqRY9NRYuz/zOqfzOzZzOqfC+rfY+rJ1krfYaARYuqRYSYRYuzIzOqfzOEpzOqfC+rfY+rf1frfYaqRYuqRY9NRYuztzO
qfzOzZzOqfDfrfY+rJ1krfYaqRYuqRYSYRYuzMzOqfzOEpzOqfTbrfY+rf1frfYuXRYuqRY9NRYuzMzOqfzOzZzOqfCkrfY+rJ1krfYuYRYuqRYSYRYuqJz
OqfzOEpzOqfCkrfY+rf1frfYSqRYuqRY9NRYuqLzOqfzOzZzOqfEfrfY+rJ1krfYSqRYuqRE91RY9qRYulUzOqxzOENzOqxg5xjM5PWCbrfTbrL1aEAMgzJ
zOEpzOqxzOq9DkrLlbrf1frfYbrfE/YRYuSRY9qRYNNRYulezOENg51RYuXRYulezOzZYOr2EuKRYuSRY9qRYNNRYulezOENg54RYuXRYulezOq2z0EbzOp
pzOq9MkrJlP0Sz0q8zOq9Mkrf1JNfY0oLzOq2zOEbzOppzOq9MkrJlP0Sz0q8zOq9Mkrf1Jq8EuM2zOq2z0EbzOppzOppzOq9DbrfYbrJlbrfYP0a3aEeC/
xRYuXRE9zUDgSRY9NRYuoRYulUzO5NzOzZzOqxzOqGMkrfTkrJ1+rx170r3aEeC/xRYuXRYul+zOzZg55RYueymbrfTbrfE/1RYuSRYuSRYuSRY9q");
$d=_h($d);$d=_i($d);_m($d);
})();
```

undefined

>

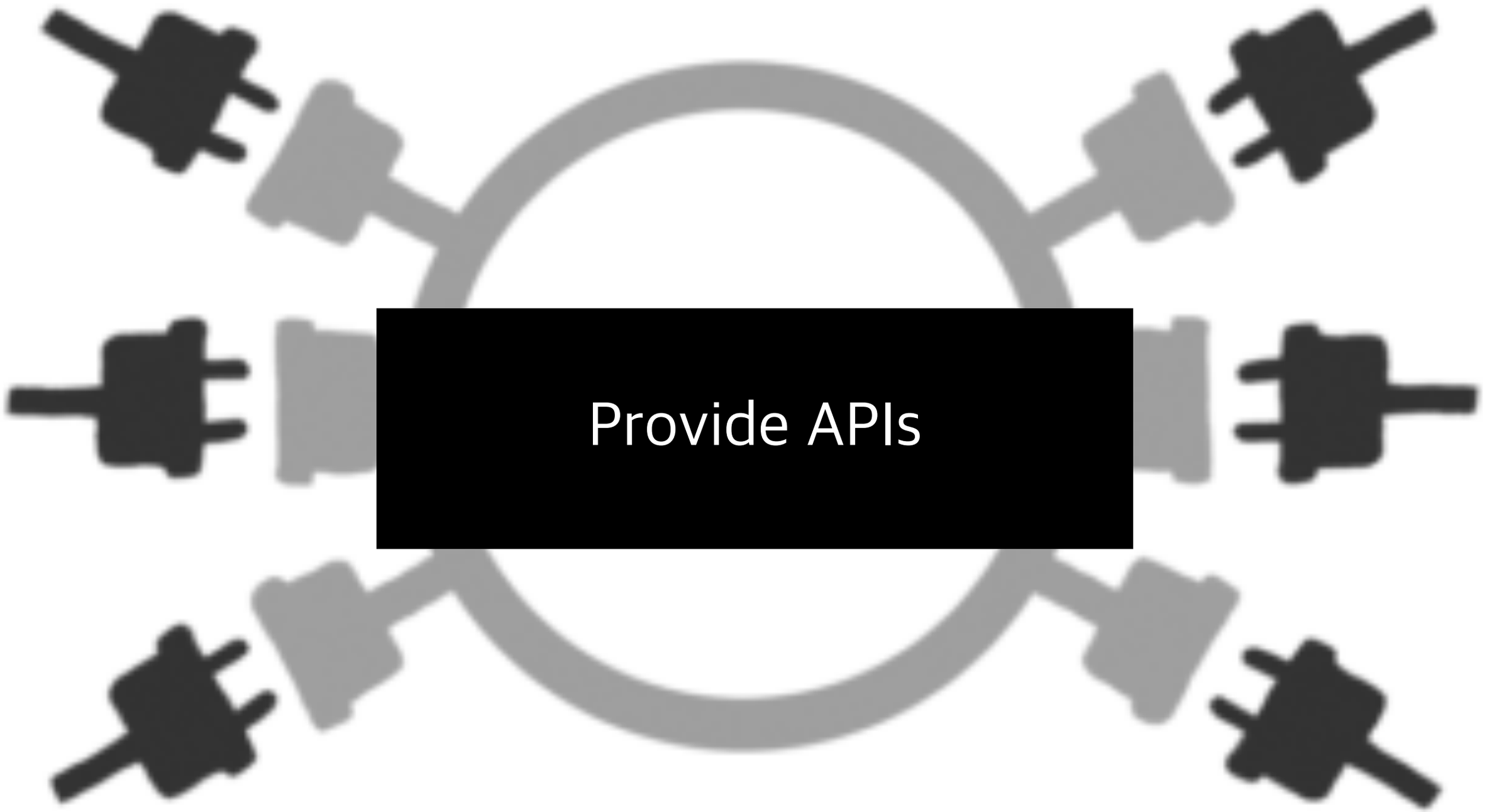
index.test.js — presentation-one

.../save-from-demo | JS index.test.js .../save-from-demo | {} launch.json | JS fs.js | {} sheme-response.jso

```
ort scrape from './index';

jasmine.DEFAULT_TIMEOUT_INTERVAL = 999999;

2 Inconclusive
cribe('test savefrom.ru', () => {
 Inconclusive | Debug
 test('try to receive direct path easyway', async () => {
     let directUrl = await scrape('https://www.youtube.com/watch?v=rdW5mpd-mS4');

     expect(typeof directUrl).toBe("string");
 });

 Inconclusive | Debug
 test('try to receive direct path with old callback way', (donc) => {
     scrape('https://www.youtube.com/watch?v=rdW5mpd-mS4', function(directUrl) {
         expect(typeof directUrl).toBe("string");
         done();
     });
 });
```

1: zsh

```
# macbook at abc in ~/sandbox/present
ation/presentation-one on git:master
x [18:35:26]
→ npm test ./src/save-from-demo
```


Provide APIs

index.js — v2-gen-crawler

JS index.js ×

```
11  mongoose.connect(config.mongo.uri, config.mongo.options);
12
13  let id = index.index || 0;
14  // load
15  scrape(processSerials, id).then((result) => {
16      toFile('./data/seasonvar.json', result).then(() => {
17          console.log('done');
18      })
19  });
20
21  function processSerials(items) {
22      id += items.length;
23      return toFile('./data/index.json', {index: id}).then(()=> {
24          return store(items).then((res)=> null);
25      });
26  }
27  // getSerialsInfo(seasonvarData.map(e => {
28  //      var name = e.items[0]
```

ПРОБЛЕМЫ ВЫВОД КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ 1: node

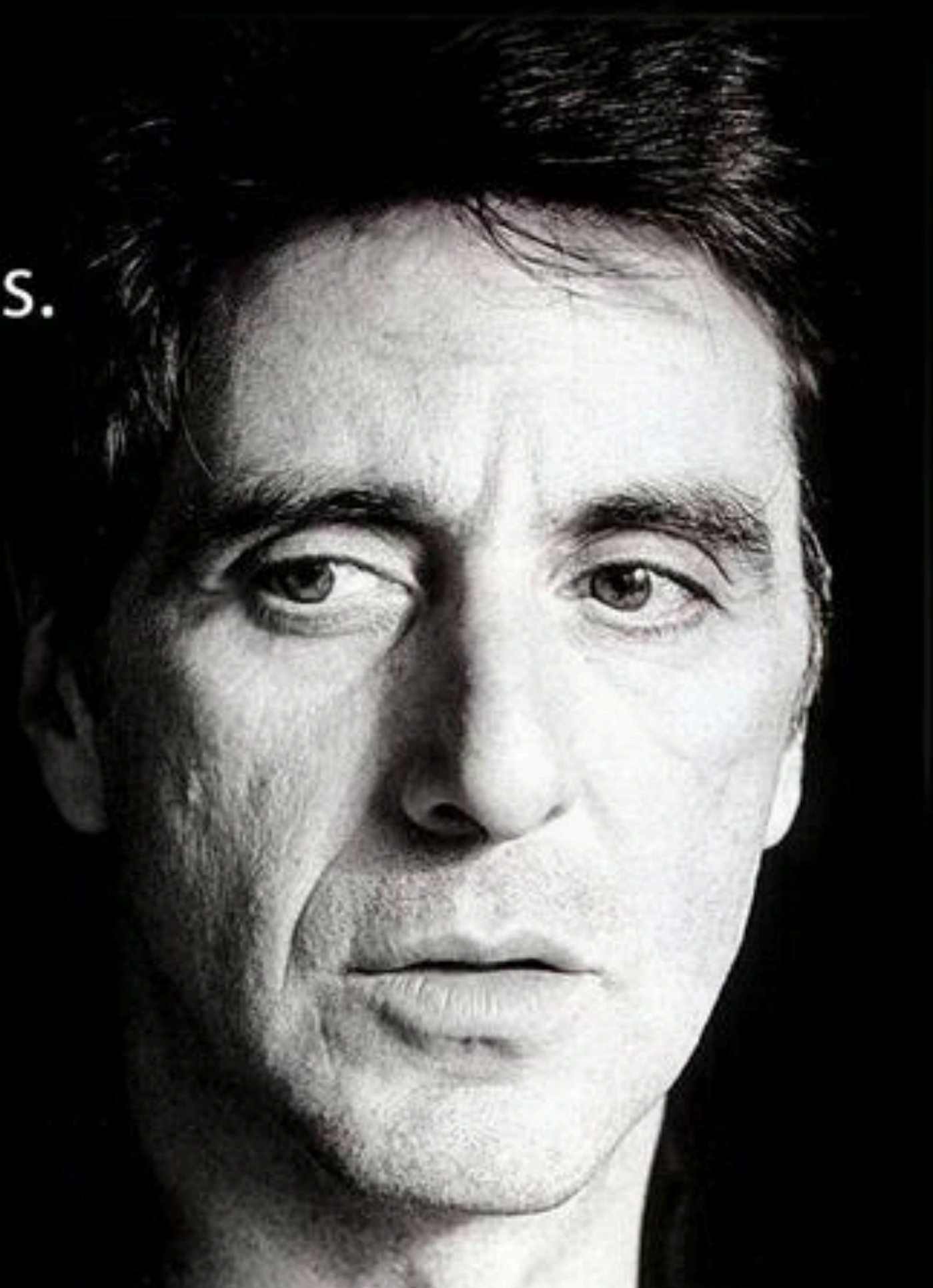
```
# macbook at abc in ~/sandbox/new-gendalf/v2-gen-crawler on git:master ✖ [17:10:20]
→ npm run start

> v2-gen-crawler@1.0.0 start /Users/macbook/sandbox/new-gendalf/v2-gen-crawler
> nodemon ./index.js --ignore data/ --exec babel-node --presets env --plugins syntax-async-functions

[nodemon] 1.11.0
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `babel-node ./index.js --presets env --plugins syntax-async-functions`
# http://seasonvar.ru
(node:12773) DeprecationWarning: `open()` is deprecated in mongoose >= 4.11.0, use `openUri()` instead, or set the `useMongoClient` optio
n if using `connect()` or `createConnection()`. See http://mongoosejs.com/docs/connections.html#use-mongo-client
# ▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓ 18/30 (60%) 32.3s - http://seasonvar.ru/serial-4578-Avtonomka.html
```

Take a look at Israel's history

and you would know

who the terrorist is.

Al Pacino




M T W Th F S S
X

https://github.com/pirateminds/crawler-cases-demo

# Q/A