

AJAX



JavaScript
Courses

www.courses.dp.ua

В чём проблема?

Ситуация №1 http://courses.dp.ua/demo/ajax_1/

Ситуация №2 http://courses.dp.ua/demo/ajax_2/

Ситуация №3 http://courses.dp.ua/demo/ajax_3/

Есть необходимость обновлять только часть страницы. При этом данных которые необходимо разместить на момент загрузки не было. Возникает необходимость сделать дополнительные сетевые запросы, чтобы получить нужные данные. Причём всё это будет происходить когда пользователь уже взаимодействует со страницей в своём браузере.

Что делать?

The diagram shows a web interface for an 'AJAX DEMO (v.1.0)'. It features a form with a text input and a 'Отправить' button, and a message display area showing a timestamp and the text 'Hello world!'. Annotations include an orange box around the form, a purple box around the message display, and a blue box around the entire interface. To the right, two callout boxes are present: an orange one labeled 'Не обновлять' (Do not refresh) pointing to the form, and a purple one labeled 'Обновлять' (Refresh) pointing to the message display.

AJAX DEMO (v.1.0)

Сообщение:

02.02.2017 13:17:16 : Hello world!

Copyright (c) 2017

Не обновлять

Обновлять

Хорошо бы обновлять часть страницы с сообщениями, а часть с формой не обновлять пока посетитель сам не отправит новое сообщение.

AJAX

Asynchronous Javascript and XML

AJAX

(Asynchronous Javascript and XML — «асинхронный JavaScript и XML»)



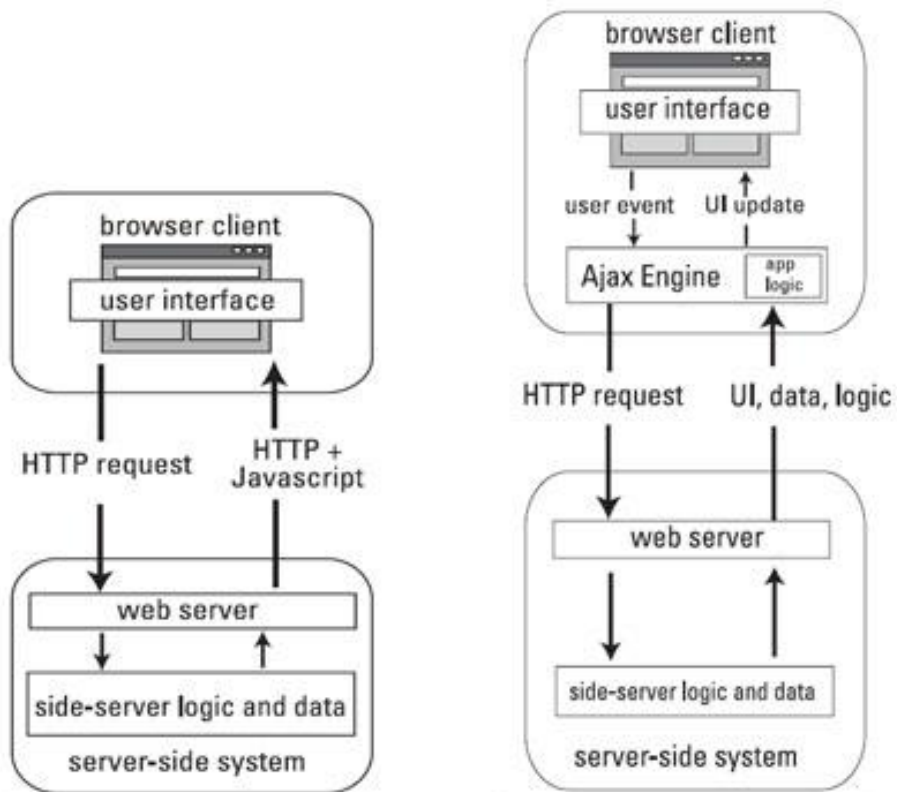
За изменение страницы в браузере пользователя отвечает JavaScript, но до этого момента JavaScript изменял страницу только на основе данных полученные еще при загрузке страницы в браузер и/или в зависимости от действий пользователя. Получить какие-то новые (дополнительные) данные JavaScript не мог.

AJAX

(Asynchronous Javascript and XML — «асинхронный JavaScript и XML»)



*С появлением в браузерах специального объекта **XMLHttpRequest** у **JavaScript** появилась возможность делать HTTP-запросы к сайтам, и изменять страницу уже на основе данных которых не было при загрузке страницы. Т.е. дозагружать HTML и/или другие данные и вставлять их на страницу.*



*Идея заложенная в **AJAX** - не перезагружая страницу полностью, запросить у сервера данные и вставить их в дерево документа.*

JSON

JavaScript Object Notation

JSON (JavaScript Object Notation)

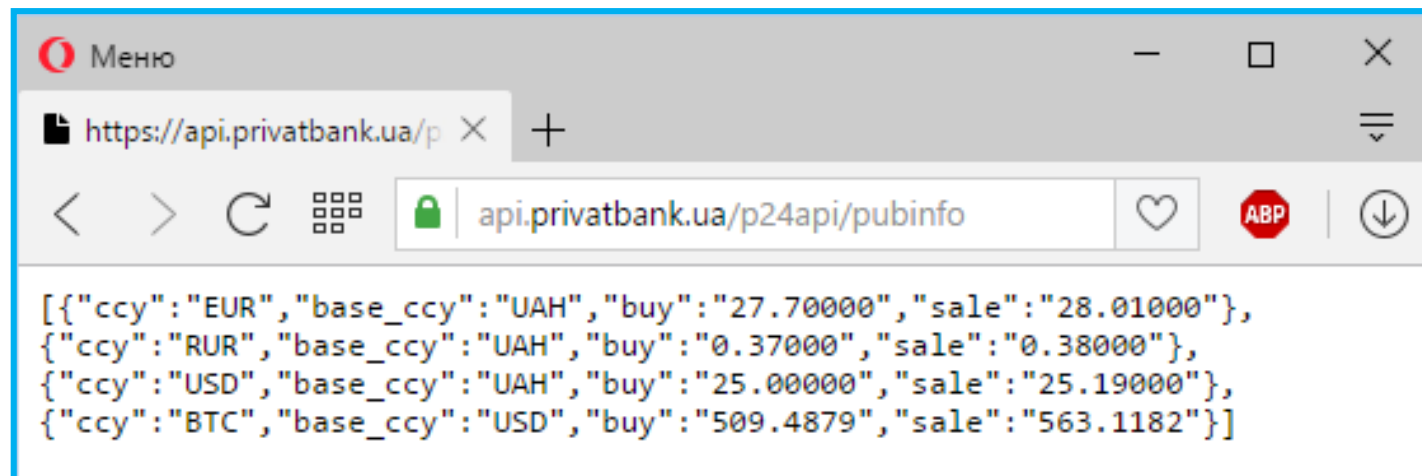
***JSON** - текстовый формат обмена данными, удобный для чтения и написания как человеком, так и компьютером. Основан на синтаксисе (правилах записи) массивов в **JavaScript (ECMAScript 3)**. Формат поддерживается практически во всех современных языках программирования.*

```
' { "name": "Вася", "age": 35, "isAdmin":  
  false, "friends": [0,1,78,99] } ' ;
```

*Сохранение объекта/массива в строковом виде и последующее восстановление объекта еще называют **сериализацией**.*

<http://www.json.org/json-ru.html>

JSON в интернете



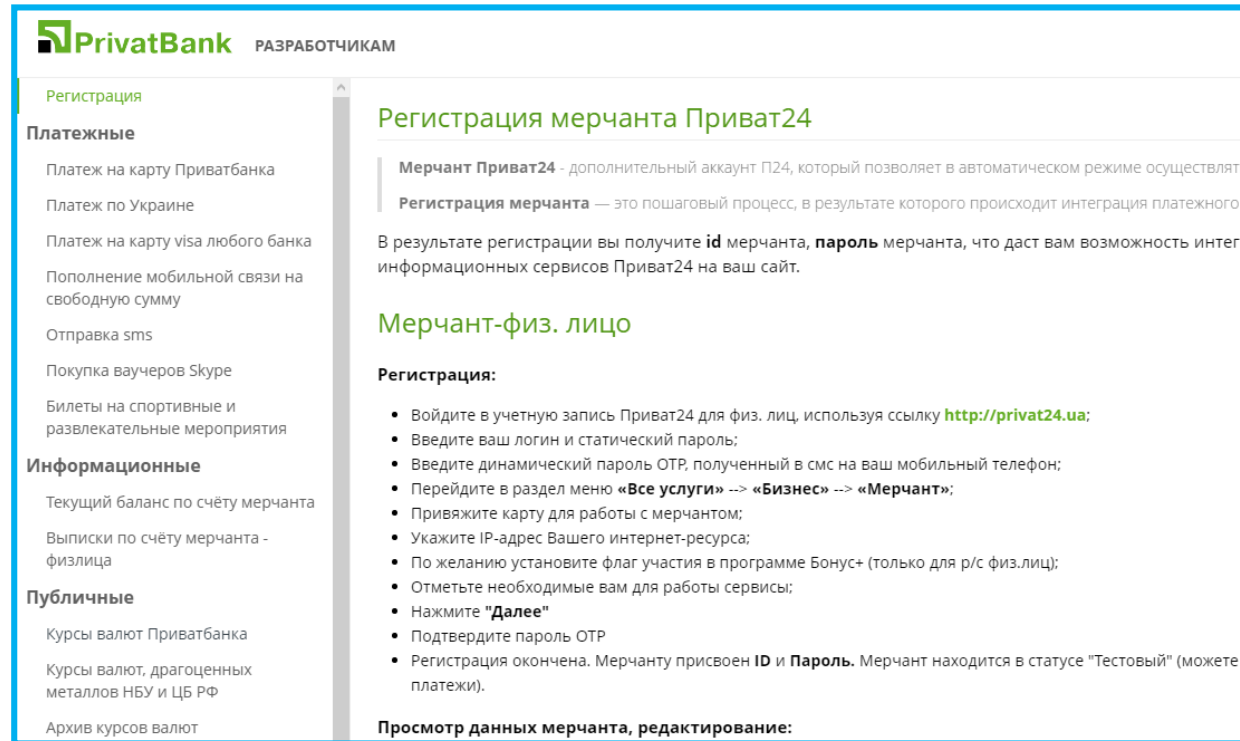
***JSON** является популярным форматом для обмена информацией в Интернете. Большое количество сервисов позволяют получить информацию в формате **JSON** для более удобной её обработки.*

API

(Application Programming
Interface)

API

API (интерфейс программирования приложений, интерфейс прикладного программирования) ([англ. *application programming interface*](#)) — Набор правил, которые определяют как необходимо общаться со сторонним сайтом/программой/системой если мы хотим запросить у него данные или передать ему данные.



<https://api.privatbank.ua/>

Валютные API есть также у НБУ

API с курсами валют есть и у НБУ:

[https://bank.gov.ua/control/uk/publish/article
?art_id=38441973](https://bank.gov.ua/control/uk/publish/article?art_id=38441973)



**НАЦІОНАЛЬНИЙ
БАНК УКРАЇНИ**

XMLHttpRequest

или

AJAX на практике

Объект XMLHttpRequest

Объект ***XMLHttpRequest*** позволяет использовать функциональность HTTP-клиента, а по простому – делать HTTP-запросы когда страница уже в браузере.

Несмотря на название ***XML*** в названии объекта, с его помощью можно передавать всё, что можно передать по протоколу ***HTTP***.

<http://xmlhttprequest.ru>

Объект XMLHttpRequest

```
1 <script>
2
3     var xhr = new XMLHttpRequest();
4
5     xhr.open("GET", "http://ip-api.com/json");
6
7     xhr.send();
8
9     console.dir(xhr);
10
11 </script>
```

Объект **XMLHttpRequest** позволяет использовать функциональность HTTP-клиента, а по простому – делать HTTP-запросы когда страница уже в браузере.

<http://xmlhttprequest.ru>

```
xmlhttprequest.html:8
▼ XMLHttpRequest
  onabort: null
  onerror: null
  onload: null
  onloadend: null
  onloadstart: null
  onprogress: null
  onreadystatechange: null
  ontimeout: null
  readyState: 4
  response: '{"as":"AS15377 ISP Fregat Ltd.,"city":"Dnipropetro
  responseText: '{"as":"AS15377 ISP Fregat Ltd.,"city":"Dniprop
  responseType: ""
  responseURL: "http://ip-api.com/json"
  responseXML: null
  status: 200
  statusText: "OK"
  timeout: 0
  ▶ upload: XMLHttpRequestUpload
    withCredentials: false
  ▶ __proto__: XMLHttpRequest
>
```

Объект XMLHttpRequest

***XMLHttpRequest** – поддерживает событийную модель, и в зависимости от развития ситуации генерирует те или иные события.*

***Синхронный запрос** – при котором браузер ждёт ответа, скрипт при этом «замирает» до прихода ответа. **Асинхронный** – скрипт продолжает выполняться, при поступлении ответа будет вызвана функция зарегистрированная как обработчик события **onload**.*

АJAX на практике

```
1 <script>
2   var usd_sum = prompt("Введите сумму в долларах: ");
3
4   var xhr = new XMLHttpRequest();
5   var url = "https://api.privatbank.ua/p24api/pubinfo?json&exchange&coursid=5";
6
7   xhr.open("GET", url, true);
8
9   xhr.onload = function(){
10      var result = xhr.responseText;
11      result = JSON.parse(result);
12      var uah_sum = parseFloat(result[2]["sale"]) * usd_sum;
13      console.log("AJAX Finish", (new Date).toISOString());
14      alert(usd_sum + "$ => " + uah_sum + " грн.");
15   }
16
17   xhr.onerror = function(){
18      alert("Ошибка загрузки данных.");
19   }
20
21   console.log("AJAX Start", (new Date).toISOString());
22   xhr.send();
23 </script>
```



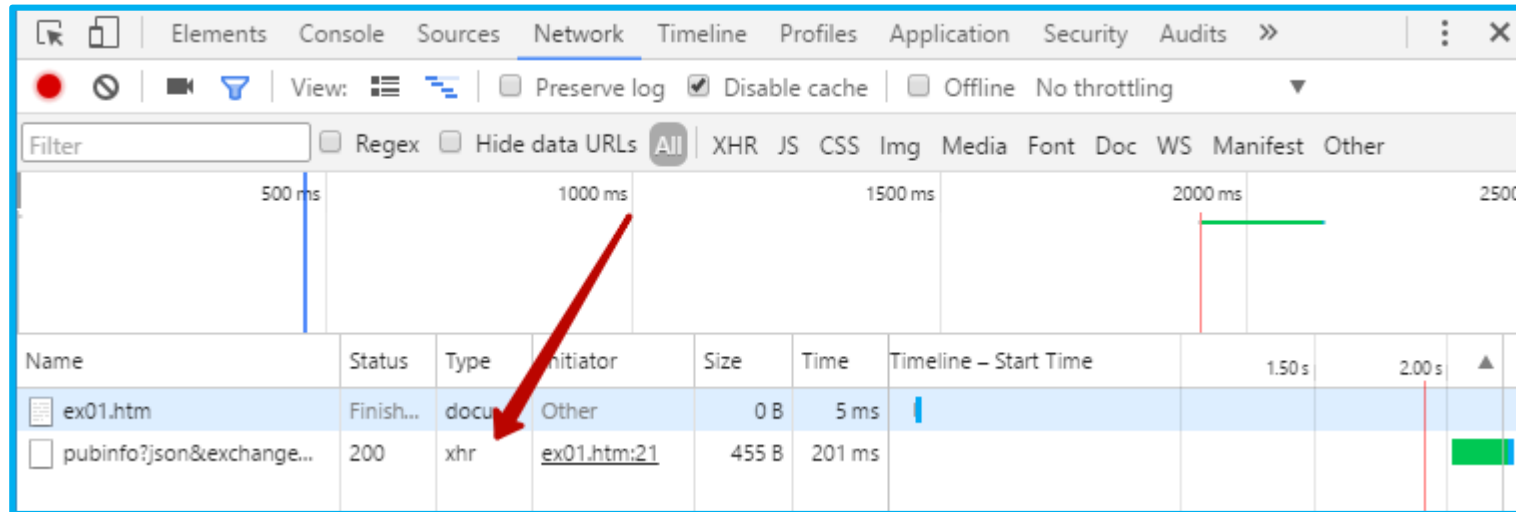
Скрипт для своей работы использует актуальные курсы валют Приватбанка.

Повідомлення з цієї сторінки:
1000\$ => 27300 грн.
☐ Заборонити створення додаткових діалогових вікон цією сторінкою.
OK

AJAX Start 2017-02-25T15:19:10.987Z
AJAX Finish 2017-02-25T15:19:11.173Z
>

Документация: <https://api.privatbank.ua/#p24/exchange>

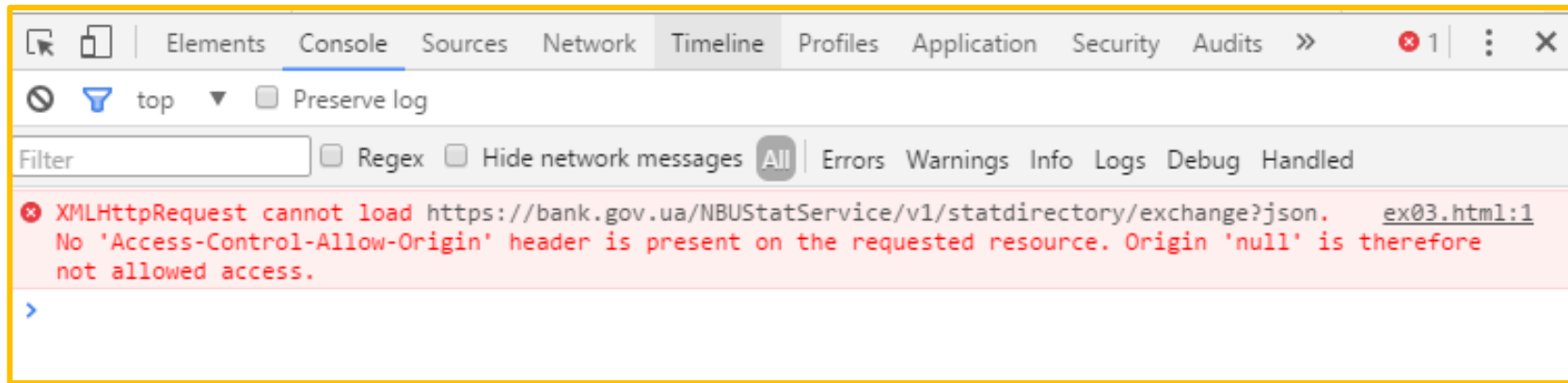
АЈАХ на практике



В консоли разработчика хорошо заметны запросы которые делались через **XMLHttpRequest** – по характерной метке **type** равной **xhr**.

Кросс-доменные запросы

Кросс-доменные запросы



*Не все AJAX запросы безопасны, браузер
бдит 😊*

Сравните результаты AJAX-запросов по адресам:

http://www.courses.dp.ua/demo/ajax_json_1/

http://www.courses.dp.ua/demo/ajax_json_2/

Кросс-доменные запросы

Кросс-доменные запросы (т.е. запросы к другому домену, не к тому с которого загружен скрипт) проходят контроль безопасности (который осуществляет браузер).

*Чтобы страница могла быть доступна через кросс-доменные запросы (читай **AJAX** запросы к страницам других сайтов), страница должна сама сказать об этом, а именно установить в **HTTP** ответе заголовок **Access-Control-Allow-Origin**.*

<https://learn.javascript.ru/xhr-crossdomain>

Что-то много кода...

Функции асинхронной загрузка данных jQuery

.load()

.get()

.post()

.getJSON()

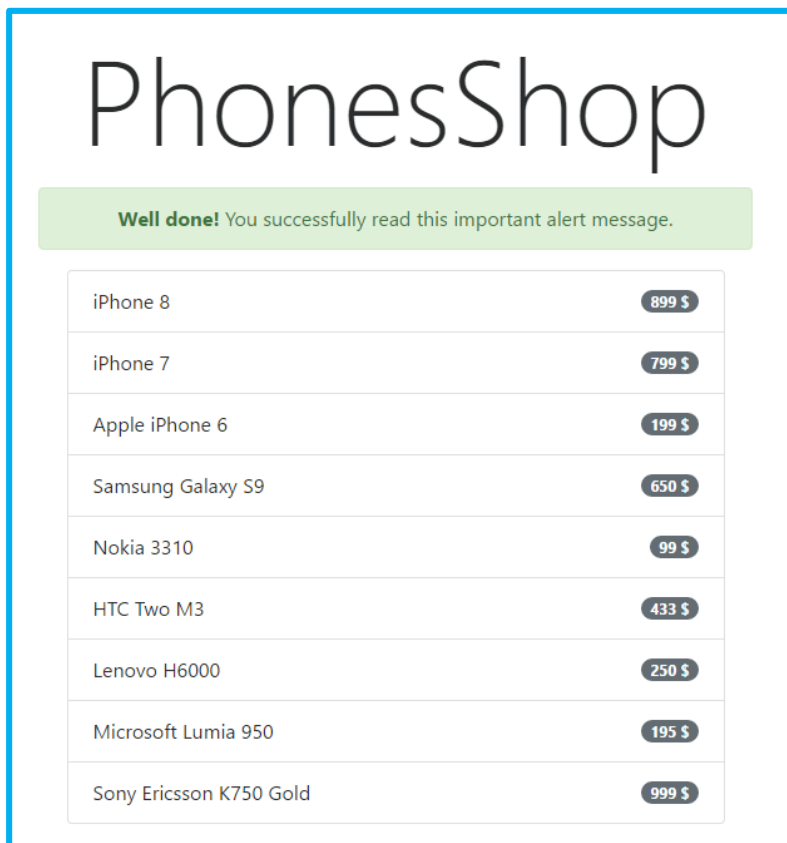
.ajax()

*jQuery предоставляет несколько функций для совершения AJAX запросов. Но можно обойтись одной **.ajax()**, все остальные лишь её обёртки.*

<http://api.jquery.com/category/ajax/>

Немного практики #1

jQuery + AJAX на практике



Воспользуйтесь заготовкой: [./source/ex01.html](http://source/ex01.html)

Пересчитаем цены в гривневые, за основу курса
возьмём данные Приватбанка.

jQuery + AJAX на практике

```
22 <script src="https://code.jquery.com/jquery-3.1.1.min.js" integrity=
    "sha256-hVVnYaiADRT02PzUGmuLJr8BLUSjGIZsDYGmIJLv2b8=" crossorigin="anonymous"></script>
23
24 <script>
25     $(function(){
26         var url = "https://api.privatbank.ua/p24api/pubinfo?json&exchange&coursid=5";
27
28         $(function(){
29             $.getJSON(url, function(data){
30                 var kurs = data[2]["sale"];
31                 $("li > span").each(function(index, value){
32                     var uah_price = Math.floor(parseInt(value.innerHTML) * kurs);
33                     value.innerHTML = uah_price + " грн.";
34                 });
35             });
36         });
37     });
38 </script>
```

Samsung Galaxy S9	17745 грн.
Nokia 3310	2702 грн.
HTC Two M3	11820 грн.
Lenovo H6000	6825 грн.

Пересчитаем цены в гривневые, за основу курса возьмём данные Приватбанка.

Немного практики #2

Определяем страну по IP-адресу

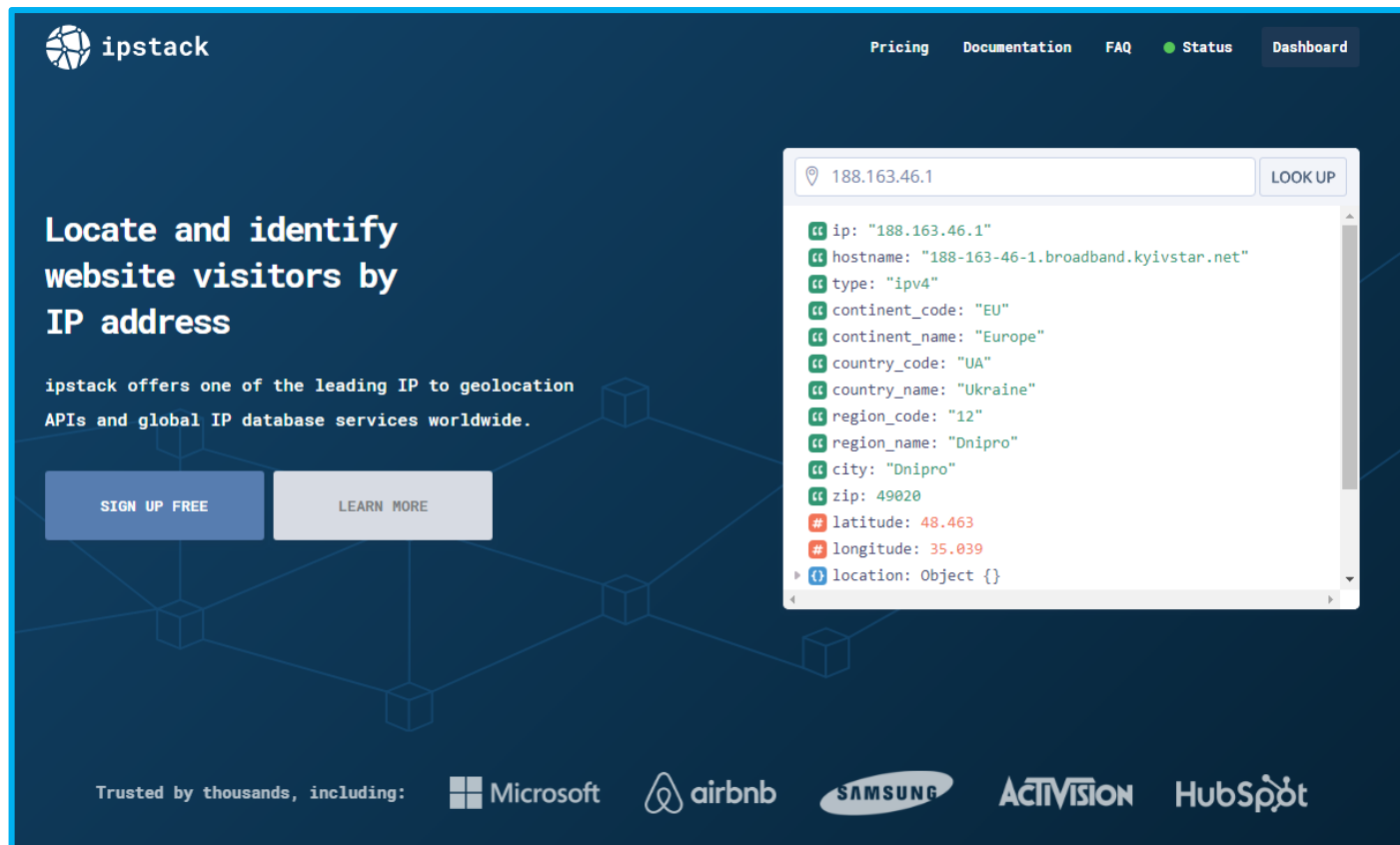
Users list						
#	First Name	Last Name	Username	IP	Country	Country Flag
1	Mark	Otto	@mdo	253.217.28.3		
2	Jacob	Thornton	@fat	56.204.132.206		
3	Larry	the Bird	@twitter	45.45.204.197		
4	Jhon H.	Smith	@colonel	230.138.69.125		
5	B.A.	Baracus	@driver	16.26.168.28		

Воспользуйтесь заготовкой: [*./source/ex02.html*](#)

Задание:

- 1) определить страну по IP адресу в таблице;
- 2) Установить флаг страны в соответствующем столбце таблицы.

API получения информации об IP адресе



*Сервис позволяют получить информацию в формате **JSON**. Но необходимо зарегистрироваться и получить ключ*

<https://ipstack.com/>

Определяем страну по IP-адресу

```
▼ Object ⓘ  
  city: "Orlando"  
  continent_code: "NA"  
  continent_name: "North America"  
  country_code: "US"  
  country_name: "United States" ←  
  ip: "108.83.164.46"  
  latitude: 28.5225  
  ▼ location:  
    calling_code: "1"  
    capital: "Washington D.C."  
    country_flag: "http://assets.ipstack.com/flags/us.svg" ←  
    country_flag_emoji: "us"  
    country_flag_emoji_unicode: "U+1F1FA U+1F1F8"  
    geoname_id: 4167147  
    is_eu: false  
    ► languages: [{...}]  
    ► __proto__: Object  
    longitude: -81.4835  
    region_code: "FL"  
    region_name: "Florida"  
    type: "ipv4"  
    zip: "32835"  
    ► __proto__: Object
```






Сервис позволяет обратившись по URL с параметрами:
http://api.ipstack.com/6.44.69.250?access_key=a45662d4c84564e1f339afbc7541b60ce
получить JSON с данными об интересующем IP адресе.

Определяем страну по IP-адресу

```
19 <script>
20
21 var url = "http://api.ipstack.com/";
22 var key = "?access_key=a5362d4c844100e1f339afbc591b60ce";
23
24 $(function() {
25
26     $("td:nth-child(5)").each(function(i, tag) {
27         tag = $(tag);
28         var ip = tag.html();
29         $.getJSON(url + ip + key, function(answer) {
30             if(answer.country_name) {
31                 tag.next().html(answer.city + ", " + answer.country_name);
32                 tag.next().next().html("<img width='32' src='" + answer.location.country_flag + "'>");
33             }
34         });
35     });
36
37 });
38
39 </script>
```



Определяем страну по IP-адресу

Users list						
#	First Name	Last Name	Username	IP	Country	Country Flag
1	Mark	Otto	@mdo	178.75.144.107	Helsinki, Finland	
2	Jacob	Thornton	@fat	49.70.196.84	Nanjing, China	
3	Larry	the Bird	@twitter	168.169.13.29	Buffalo, United States	
4	Jhon H.	Smith	@colonel	191.8.111.45	Mairiporã, Brazil	
5	B.A.	Baracus	@driver	221.204.132.238	Taiyuan, China	

В результате

Немного практики #3

Кредитный калькулятор 3.0

Кредитный калькулятор

Сумма кредита (грн.)

Срок кредитования (мес.)

Ставка (% годовых)

7000

3

24

Рассчитать

#	Сумма платежа (грн.)	Часть тела кредита (грн.)	Проценты (грн.)	Остаток долга (грн.)
1	2473.33	2333.33	140.00	4666.67
2	2426.67	2333.33	93.33	2333.33
3	2380.00	2333.33	46.67	-0.00

Переплата: 280.00 грн.

Воспользуемся заготовкой: [./source/ex03.html](#)

Реализуем кредитный калькулятор, который будет получать данные для расчёта из элементов ввода, и выводить план погашения кредита в разметку. Используем алгоритм классической схемы погашения.

Кредитный калькулятор 3.0

```
14  $(() =>{
15      $("button").on("click", ()=>{
16          $('#results').empty();
17
18          var sum      = +sum_input.value;
19          var term      = +term_input.value;
20          var rate      = +rate_input.value;
21          var debt      = sum;
22
23          var rate_month = rate / 12 / 100;
24          var payment_body = +(sum / term).toFixed(2);
25
26          for(var i = 1; i <= term; i++){
27              var payment_percents = +(sum * rate_month).toFixed(2);
28              var payment = +(payment_body + payment_percents).toFixed(2);
29              if(i == term) {
30                  payment_body = sum;
31              }
32
33              sum -= payment_body;
34              sum = +sum.toFixed(2);
35
36              $('#results').append($(`
37                  <tr>
38                      <td>#${i}</td>
39                      <td>${payment}</td>
40                      <td>${payment_body}</td>
41                      <td>${payment_percents}</td>
42                      <td>${sum}</td>
43                  </tr>
44              `));
45          });
46      });
47  });
48  }
```

Математика в JS...

Математика в JavaScript

$$\begin{array}{c} 2 > -3 \\ 0.999... = 1 \\ \pi \approx 3.14 \\ \sqrt{2} \\ 5(2 + 2) \\ 101_2 = 5_{10} \end{array} \quad \begin{array}{c} + \\ - \\ \times \\ \div \\ 5^2 \\ (1 - 2) + 3 \end{array}$$

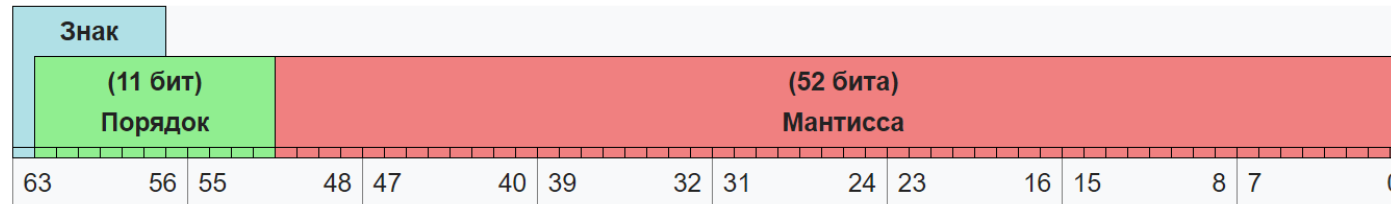
<https://habr.com/post/312880/>

<https://medium.com/nuances-of-programming/javascript-essentials-числа-и-математика-ccf6b582a79>

Числа в JavaScript

Как хранятся числа в переменных типа **Number**:

https://ru.wikipedia.org/wiki/Число_двойной_точности



Окончательное значение числа равняется $\pm \text{знак} \cdot (1 + \text{мантисса} / 2^{52}) \times 2^{\text{порядок} - 1023}$. Знак 0 соответствует положительным числам, знак 1 отрицательным. Старший бит мантиссы, который всегда равен единице, опускается. Порядок 0 ($2^0 = 1$) записывается как 1023.

Чисел в JavaScript

Специальные константы (и методы) в JavaScript'е для определение «безопасных» чисел

`Number.MIN_VALUE`

`Number.MAX_VALUE`

`Number.MIN_SAFE_INTEGER`

`Number.MAX_SAFE_INTEGER`

`Number.isSafeInteger()`

`Number.isInteger()`

Округление числе в JavaScript

Специальные Методы для округления:

`Number.toFixed()`

`Number.toPrecision()`

`Math.round()`

`Math.floor()`

`Math.ceil()`

`Math.abs()`

`Math.sign()`

Проблема сравнения чисел в JavaScript

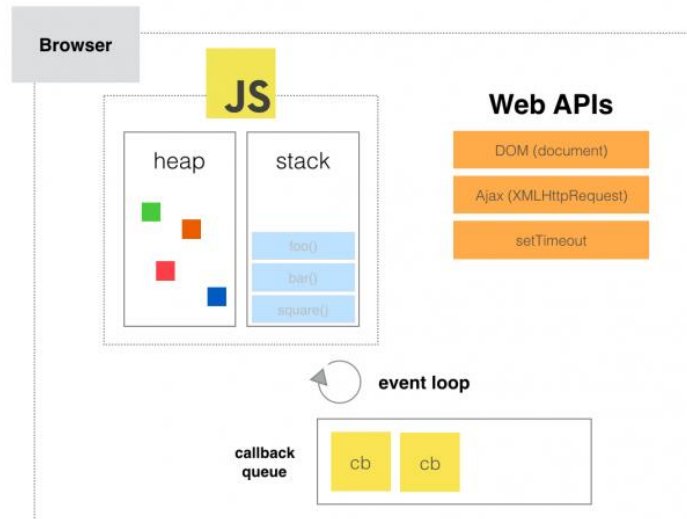
Сравнение чисел с поправкой на погрешность:

```
function epsilon_equality(x, y) {  
    return Math.abs(x) - Math.abs(y) < Number.EPSILON;  
}  
  
var a = 0.1 + 0.2;  
var b = 0.3;  
  
console.log( a == b );  
console.log( epsilon_equality(a, b) );
```

Number.EPSILON – отражает разницу между 1 и наименьшим числом с плавающей точкой, что более 1.

Домашнее задание
/узнать

Узнать/ JavaScript Event Loop



<https://www.youtube.com/watch?v=8cV4ZvHXQL4>

https://www.youtube.com/watch?v=j4_9BZezSUA

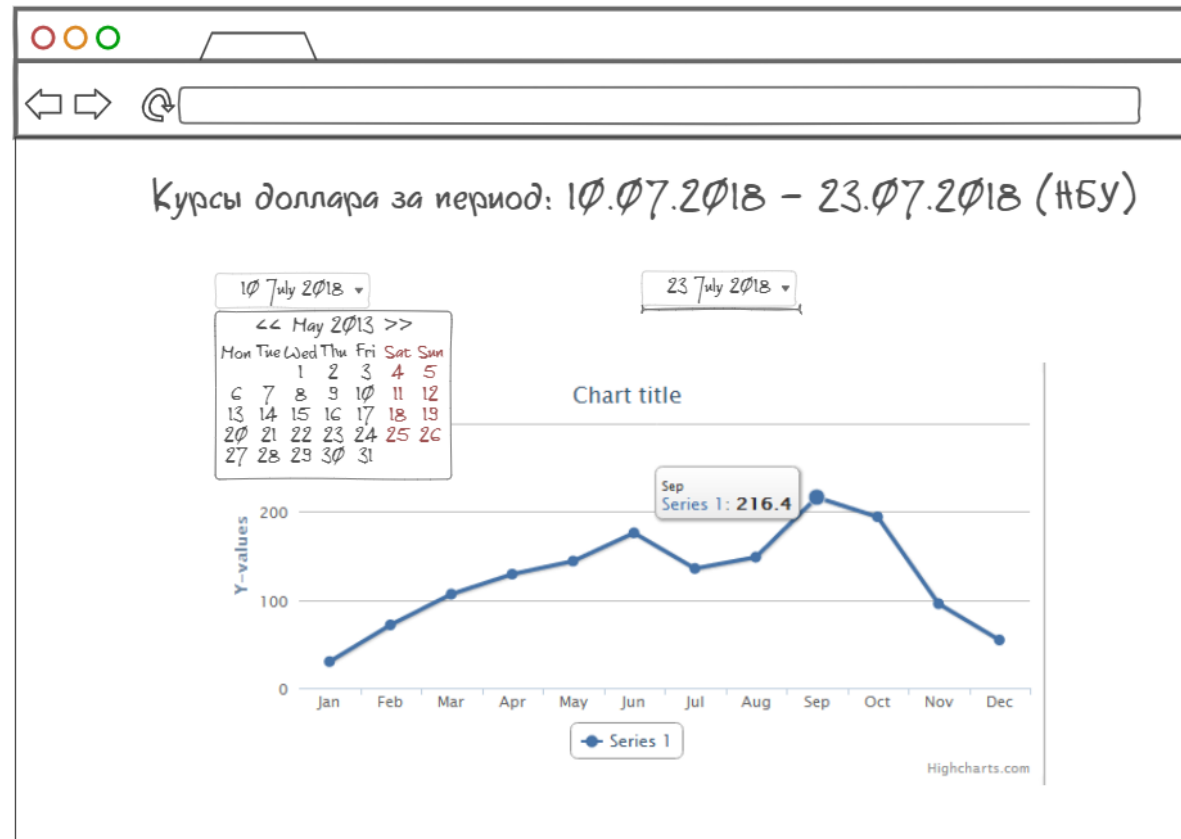
BigInt - потенциальное решение проблемы



<https://developers.google.com/web/updates/2018/05/bigint>

Домашнее задание
/сделать

Домашнее задание #J.1 График курсов валют



Задача: на основании валютных **API Национального банка Украины** и библиотеки **HighCharts** реализовать страницу на которой пользователь может выбирать диапазон из дат, а скрипт построит график курса доллара за выбранный период.

Домашнее задание #J.1+

График курса Биткоина


<https://apiv2.bitcoinaverage.com>

Суть задачи – разобраться в стороннем API.

*Есть API предоставляющее данные по курсам крипто-валют. Ваша задача вывести курс Биткоина **к гривне**, за выбранный период.*

Большое командное домашнее задание

Домашнее задание #J.2. Страница интернет магазина




Perfect Mobile Shop

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla quam velit, vulputate eu pharetra nec, mattis ac neque. Duis vulputate commodo lectus, ac blandit elit tincidunt id. Sed rhoncus, tortor sed eleifend tristique, tortor mauris

Сортировать по ▼

iPhone 7	★ 4.7	699 \$	Купить
iPhone 5	★ 4.1	399 \$	Купить
iPhone 2	★ 3.3	449 \$	Купить
iPhone 10	★ 3.9	99\$	Купить

Фильтр по цене:
От До


Perfect Mobile Shop (c) 2017

Домашнее задание #J.2. Страница интернет магазина

Необходимо реализовать страницу интернет магазина по предложенному макету

1. Список товаров должен загружаться асинхронно с адреса <http://files.courses.dp.ua/web/json/data01.php> (данные отдаются в формате JSON).
2. При помощи элемента ввода «Сортировать по» необходимо дать возможность пользователю выбрать вариант сортировки по одному из следующих направлений: **по возрастанию цены, по убыванию цены, по возрастанию рейтинга, по убыванию рейтинга.**
3. При помощи ползунков необходимо реализовать фильтрацию по цене (левый ползунок определяет нижнюю границу цены, правый ползунок – верхнюю границу). Фильтрация должна происходить динамически, без перезагрузки страницы, и в процессе перемещения того или иного ползунка.
 - 3а. Не допускать ситуации при которой левый ползунок может выбрать цену больше чем установлена в правом (и наоборот).
 - 3б. Ползунок можно заменить на компонент jQuery UI – «двойной ползунок» <https://jqueryui.com/slider/#range>