Вопросы 💎

1. Напишите JSON к вашим ответам на вопрос 2 из прошлой недели. Например, если вы ответили \*Книга (название, автор, год выпуска),\* то JSON может выглядеть так (минимум 3 примера):

*Device (name, purchase year, number of programs, target muscles)*

[{

"title": "Crossttrainer",

"year": 2010,

"programs": 5,

"muscles": upper-body,

},{

"title": "Stationary Bicyle",

"year": 2012,

"programs": 7,

"muscles": legs,

},{

"title": "Rowing Machine",

"year": 2015,

"programs": 3,

"muscles": core,

}]

*Visitor (name, age, paymenttype, begin of contract)*

[{

"name": "Anna Ivanova",

"age": 23,

" paymenttype ": card,

"contractBegin": 01.01.2020,

},{

"name": "Ivan Ivanov",

"age": 30,

" paymenttype ": cash,

"contractBegin": 01.01.2021,

}

]

*Employee (name, age, work years)*

[{

"name": "Max",

"age": 35,

"workYears": 2,

},{

"name": "Olga",

"age": 34,

"workYears": 10,

}

]

2. Самостоятельно разберитесь, что за формат данных XML и чем он отличается от JSON? Приведите пример, как один и тот же объект \*собачки с картинки ниже\* будет выглядеть в JSON и в XML?

The difference between XML and JSON is that XML is a meta-language / markup language and JSON is a lightweight data exchange. Thus, the XML syntax is designed specifically to have no intrinsic semantics. The specific element names do not mean anything until a specific processing application handles them in a specific way. In contrast, JSON syntax has special semantics built into things between {} is an object, things between [] are an array, and so on.

Therefore, the JSON parser knows exactly what each JSON document means. The XML parser only knows how to separate the markup from the data. To understand the meaning of an XML document, you must write additional code.

JSON variant:

{

“breed”: “Beagle”,

“size”: “large”,

“color”: “orange”,

“age”: 6 “years”

}

XML variant:

<Dog>

<Bread>Beagle</Bread>

<Size>Large</Size>

<Color>Orange</Color>

<Age>6 years</Age>

</Dog>

3. Что такое сериализация и десериализация (парсинг)? В каких ситуациях они нужны?

In the second step, the client receives a JSON string as response from the server and unpacks (deserialises, parses) it, i.e. turns it into a JavaScript object. After that further actions are performed on the client, e.g. displaying them on the page.

Also, if we need to send some data to server, we should first pack it into JSON-format string (serialize it) and only then send it.

stringify() : To serialize JavaScript objects into a JSON string.

parse() : To parse JSON into a native JavaScript value.

Serialisation (in programming) is the process of translating a data structure into a sequence of bytes. The inverse of serialisation is deserialisation (structuring) - the creation of a data structure from a bit sequence.

4. Можно ли обработать ответ от сервера одновременно и как текст, и как JSON?

No, it’s possible only to choose one option.

5. В чем особенность асинхронных запросов?

AJAX allows us to update parts of a web page without reloading the whole page (their URL remains the same). This means that our page will run much faster - after all, you're only reloading a piece of it, not the whole page. One of the hardest parts of understanding how JavaScript works is dealing with asynchrony, where things happen in no particular order.

In most programming languages, we're used to operations being carried out in order (sequentially). The first line must be executed before we move on to the next line.

But in JS we have operations that run in the background/active mode, so our web application doesn't freeze every time it waits for a user event.

However, sometimes things have to go in order, otherwise it will cause chaos and unexpected results. For this reason, we can use asynchronous calls to make things work as they should. An example would be checking user credentials before moving on to the next operation. We will talk more about this in the future.

6. В чем преимущество AJAX-запросов перед старым способом работы с сервером через `<form action="имя скрипта на сервере">`?

The point of AJAX is interactivity and fast response times. AJAX is useful for small elements related to elementary actions: add to cart, subscribe, etc.

In the synchronous model, the browser sends a request to the server and hangs around, waiting for the server to do all the necessary work. The server queries the database, wraps the response in the required format and outputs it. The browser receives the answer and calls the display function.

All processes run sequentially, one after the other.

Network delays are enabled during the idle time indicated in grey in the diagram.

The user cannot do anything else on the same page while synchronous data exchange is taking place.

In the asynchronous model, the request is sent out and you can do something else. When the request is fulfilled - a preliminarily prepared function is started to show the server message,function prepared in advance by the programmer is executed.

7. Напишите, как будет выглядеть `fetch` для получения данных вашего пользователя на github? Адрес URL для запроса должен выглядеть так:

`'https://api.github.com/users/сюда подставьте свой логин с github'`

let url = 'https://api.github.com/users/LiubovKostomarova';

let response = await fetch(url); //здесь мы ждем асинхронный ответ от сервера

let data = await response.json(); // читаем ответ в формате JSON

alert(LiubovKostomarova.author.login); //my login data

```

Another way:

fetch(' https://api.github.com/users/LiubovKostomarova')

.then(response => response.json())

.then(data => alert(LiubovKostomarova.author.login));

```

8. Самостоятельно разберитесь, что такое SPA?

SPA or Single Page Application is a single-page web application which is loaded onto a single HTML page. Thanks to dynamic updating using JavaScript, there is no need to reload or load additional pages during use. In practice, this means that the user sees all the main content in the browser, and when scrolling or navigating to other pages, instead of a full reload, the desired elements are simply reloaded.

As they work, the user does not feel as if they are running a web site, but rather a desktop application, as it reacts instantly to all of their actions, without lags and "freezes".

This effect is achieved by using advanced JavaScript frameworks such as Angular, React, Ember, Meteor, and Knockout.

SPA – это web-приложение, размещенное на одной странице, которая для обеспечения работы загружает все javascript-файлы (модули, виджиты, контролы и т.д.) , а также файлы CSS вместе с загрузкой самой страницы.