1. Напишите JSON к вашим ответам на вопрос 2 из прошлой недели.

1) [{

"city": "Москва",

"place": ["Северный речной вокзал", "Водный стадион"],

"company": 5,

"doing": "Прогулка",

"restaurant": "Corner Burger"

},{

city": "Выборг",

"place": "Башня Святого Олафа",

"company": 2,

"doing": "Посещение музея",

"restaurant": null

},{

city": "Энколово",

"place": "Конюшня",

"company": 3,

"doing": "Конная прогулка",

"restaurant": "Столовая при конюшне"

}]

2) [{

"dough": "Пшеничная",

"pasta": "Спагетти",

"sauce": "Сливочный",

"addition": ["Бекон", "Лук", "Зелень", "Яйцо", "Чеснок", "Свежемолотый перец"],

"cheese": true

},{

"dough": "Гречневая",

"pasta": "Фузилли",

"sauce": "Паназиатский",

"addition": ["Перец сладкий", "Перец чили", "Зелень", "Фасоль стручковая", "Чеснок", "Кунжут"],

"cheese": false

},{

"dough": "Рисовая",

"pasta": "Тальятелле",

"sauce": "Болоньезе",

"addition": ["Свеемолотый перец", "Помидоры черри свежие", "Веточка базилика"],

"cheese": true

}]

3) [{

"top": "Белая футболка",

"bottom": "Джинсы светлые",

"shoes": "Лоферы",

"outerwear": false,

"accessories": "Часы"

},{

"top": "Рубашка свободного кроя",

"bottom": "Брюки чёрные",

"shoes": "Ботинки челси",

"outerwear": true,

"accessories": "Сумка небольшая"

},{

"top": "Свитер из смесовой шерсти",

"bottom": "Джинсы",

"shoes": "Ботинки на тракторной подошве",

"outerwear": true,

"accessories": "Шляпа с ровными полями"

}]

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?

XML (расширенный язык разметки) специализируется на бизнес-транзакциях в вебе (структурированные данные), синтаксис схож с HTML. Хранит данные к текстовом формте и интегрирует их в HTML. Поддерживает много типов данных, таких как текст, массив, изображение, графики и т.д.

JSON - легкий формат открытого стандарта для обмена данными (иерархические данные), является подмножестовм синтаксиса языка JavaScript. Поддерживает текстовые, числовые, булевые типы данных, а также позволяет использование массивов и объектов.

<? xml version-"1.0" encoding-"UTF-8" ?>

<root>

<Dog>

<Breed>Beagle</Breed>

<Size>large</Size>

<Colour>orange</Colour>

<Age>6 years</Age>

</Dog>

</root>

{

"Breed": "Beagle",

"Size": "large",

"Colour": "orange",

"Age": "6 years"

}

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

Сериализация - преобразование данных в какой-то формат, чтобы потом можно было эти данные восстановить (в нашем случае в текстовый JSON формат).

Десериализация - обратный процесс.

Сериализованный формат лего сохраняется на диск, передаётся по сети или передаётся другим процессам, в отличии от "чистых" данных. JSON часто используется для передачи информации от веб-сервера пользователю.

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

Нет, данные можно обработать только один раз.

5. В чем особенность асинхронных запросов?

Работают, не изменяя URL (не перегружая страницу) и независимо от других процессов на странице (не останавливают их и не мешают им).

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

Не перезагружает страницу, так как работает асинхронно и, тем самым, увеличивает быстродействие сайта.

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

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

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

.then(commits => console.log(commits))

8. Самостоятельно разберитесь, что такое SPA?

SPA - Single Page Application, одностраничное приложение. То есть сайт или приложение, которое загрузось один раз и далее работает с помощью асинхронных скриптов (запрос-ответ), не презагружая страницу.