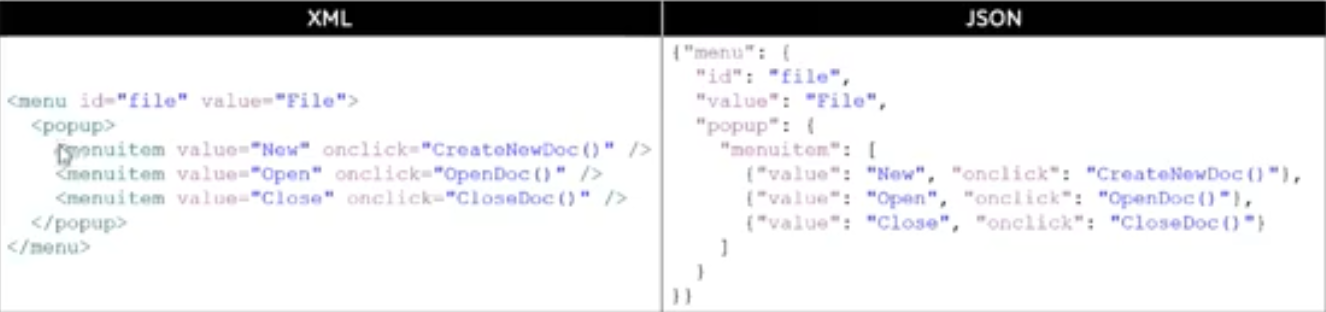
**Структуры данных**

XML построен на тэгах, а JSON на объектах



**XML**

**XML** – eXtensible вMarkup Language (расширяемый язык разметки). Служит для передачи данных между приложениями, хранения данных

**XPath** — язык запросов к элементам XML-документа.

**How to Access xPath**

1. Inside Notepad++ go to Plugins -> Plugin Manager -> Show Plugin Manager.
2. Install XML Tools.
3. Restart Notepad.
4. Load your XML Doc.
5. Place your cursor onto the node you are looking to generate the xpath for.
6. Go Plugins -> XML Tools -> Current XML Path (Default Hotkey : Ctrl + Alt + Shift + P)

****

**Абсолютный xPath**

**/bookstore/book[1]** Selects the first book element that is the child of the bookstore element

**/bookstore/book@category** Selects a tag

**/bookstore/book[last()]** Selects the last book element that is the child of the bookstore element

**/bookstore/book[last()-1]** Selects the last but one book element that is the child of the bookstore element

**/bookstore/book[position()<3]** Selects the first two book elements that are children of the bookstore element

**//title[@lang]** Selects all the title elements that have an attribute named lang

**//title[@lang='en']** Selects all the title elements that have a "lang" attribute with a value of "en"

**/bookstore/book[price>35.00]** Selects all the book elements of the bookstore element that have a price element with a value greater than 35.00

**/bookstore/book[price>35.00]/title** Selects all the title elements of the book elements of the bookstore element that have a price element with a value greater than 35.00

**/bookstore/book[1]/\* | /bookstore/book[2]/\*** Using “pipe” we can combine 2 xPaths. Suing \* sign we can select all children

**Относительный xPath**

**//book[last()]**

**…**

**XSD (XML Schema)**

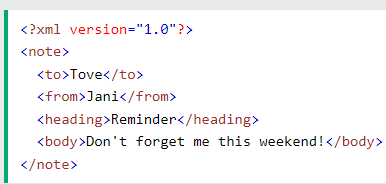
An XML Schema describes the structure of an XML document.

The purpose of an XML Schema is to define the legal building blocks of an XML document:

* the elements and attributes that can appear in a document
* the number of (and order of) child elements
* data types for elements and attributes
* default and fixed values for elements and attributes

**XML and it’s Schema**

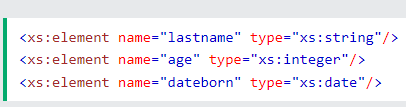
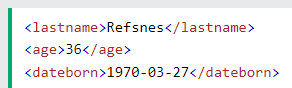
“note” element is a complexType element. It means it includes other elements



**Data types**

XML Schema has a lot of built-in data types. The most common types are:

* xs:string
* xs:decimal
* xs:integer
* xs:boolean
* xs:date
* xs:time



**JSON**

JSON - простой, основанный на использовании текста, способ хранить и передавать структурированные данные. С помощью простого синтаксиса вы можете легко хранить все, что угодно, начиная от одного числа до строк, массивов и объектов, в простом тексте. Также можно связывать между собой массивы и объекты, создавая сложные структуры данных.

После создания строки JSON, ее легко отправить другому приложению или в другое место сети, так как она представляет собой простой текст.

JSON имеет следующие преимущества:

* Он компактен.
* Его предложения легко читаются и составляются как человеком, так и компьютером.
* Его легко преобразовать в структуру данных для большинства языков программирования (числа, строки, логические переменные, массивы и так далее)
* Многие языки программирования имеют функции и библиотеки для чтения и создания структур JSON.

Название JSON означает JavaScript Object Notation (представление объектов JavaScript). Как и представляет имя, он основан на способе определения объектов (очень похоже на создание ассоциативных массивов в других языках) и массивов.

**Для чего используется JSON?**

Наиболее частое распространенное использование JSON - пересылка данных от сервера к браузеру. Обычно данные JSON доставляются с помощью [AJAX](http://en.wikipedia.org/wiki/Ajax_%28programming%29), который позволяет обмениваться данными браузеру и серверу без необходимости перезагружать страницу.

Пример:

1. Пользователь нажимает миниатюру продукта в онлайн магазине.
2. JavaScript, выполняющийся на браузере, генерирует запрос AJAX к скрипту PHP, запущенному на сервере, передавая ID выбранного продукта.
3. Скрипт PHP получает название продукта, описание, цену и другую информацию из базы данных. Затем составляет из данных строку JSON и отсылает ее браузеру.
4. JavaScript, выполняющийся на браузере, получает строку JSON, декодирует ее и выводит информацию о продукте на странице для пользователя.

Также можно использовать JSON для отправки данных от браузера на сервер, передавая строку JSON в качестве параметра запросов GET или POST. Но данный метод имеет меньшее распространение, так как передача данных через запросы AJAX может быть упрощена. Например, ID продукта может быть включен в адрес URL как часть запроса GET.

**Как создать строку JSON?**

Есть несколько основных правил для создания строки JSON:

* Строка JSON содержит либо массив значений, либо объект (ассоциативный массив пар имя/значение).
* *Массив* заключается в квадратные скобки ([ и ]) и содержит разделенный запятой список значений.
* *Объект* заключается в фигурные скобки ({ и }) и содержит разделенный запятой список пар имя/значение.
* *Пара имя/значение* состоит из имени поля, заключенного в двойные кавычки, за которым следует двоеточие (:) и значение поля.
* *Значение* в массиве или объекте может быть:
  + Числом (целым или с плавающей точкой)
  + Строкой (в двойных кавычках)
  + Логическим значением (true или false)
  + Другим массивом (заключенным в квадратные скобки)
  + Другой объект (заключенный в фигурные скобки)
  + Значение null

Чтобы включить двойные кавычки в строку, нужно использовать обратную косую черту: \". Так же, как и во многих языках программирования, можно помещать управляющие символы и шестнадцатеричные коды в строку, предваряя их обратной косой чертой

**Сравнение JSON и XML**

Во многих отношениях вы можете рассматривать JSON как альтернативу XML, по крайней мере, в сфере веб приложений. Концепция AJAX оригинально основывалась на использовании XML для передачи данных между сервером и браузером. Но в последние годы JSON становится все более популярным для переноса данных AJAX.

Хотя XML является проверенной технологией, которая используется в достаточном количестве приложений, преимуществами JSON являются более компактный  и простой для распознавания формат данных.

Вот как будет выглядеть выше приведенный пример объекта на XML:



Версия XML имеет существенно больший размер. В действительности она имеет длину 1128 символов, а вариант JSON - только 323 символа. Версию XML также достаточно трудно воспринимать.

Конечно, это радикальный пример. И возможно создать более компактную запись XML. Но даже она будет существенно длиннее эквивалента на JSON.

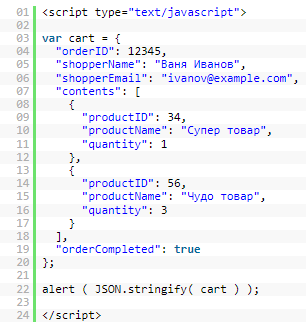
**Работаем со строкой JSON в JavaScript**

JSON имеет простой формат, но создавать строку JSON вручную достаточно утомительно. Кроме того, часто нужно взять строку JSON, конвертировать ее содержание в переменную, которую можно будет использовать в коде.

Большинство языков программирования имеют инструменты для простого преобразования переменных в строки JSON и наоборот.

**Создаем строку JSON из переменной**

JavaScript имеет встроенный метод JSON.stringify(), который берет переменную и возвращает строку JSON, представляющую ее содержание. Например, создадим объект JavaScript, который содержит сведения о заказе из нашего примера, а затем создадим из него строку JSON:



Данный код выдаст:



Обратите внимание, что метод JSON.stringify() возвращает строку JSON без пробелов. Ее сложнее читать, но зато она более компактна для передачи через сеть.

### Создаем переменную из строки JSON

Существует несколько способов разобрать строку JSON в JavaScript, но самый безопасный и надежный - использовать встроенный метод JSON.parse(). Он получает строку JSON и возвращает объект или массив JavaScript, который содержит данные. Например:

****

Мы создали переменную jsonString, которая содержит строку JSON нашего примера с заказом. Затем мы передаем данную строку методу JSON.parse(), который создает объект, содержащий данные JSON и сохраняет его в переменной cart. Остается только осуществить проверку, выведя свойства объекта shopperEmail и productName массива contents.

В результате мы получим следующий вывод:



В реальном приложении ваш JavaScript код будет получать заказ в виде строки JSON в AJAX ответе от скрипта сервера, передавать строку методу JSON.parse(), а затем использовать данные для отображения на странице пользователя.

JSON.stringify() и JSON.parse() имеют другие возможности, такие как  использование возвратных функций для пользовательской конвертации определённых данных. Такие опции очень удобны для конвертации различных данных в правильные объекты JavaScrip