



Scripting and Web Applications EE1081

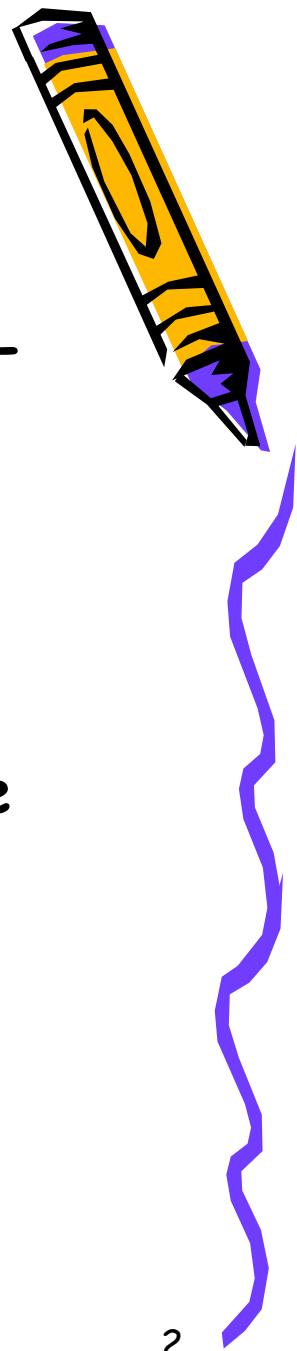
Lecture 7 JavaScript

By: A. Mousavi

SERG, School of Engineering Design, Brunel University, UK



JavaScript Security and Cookies



1. About JavaScript and security issues (client-side)
2. Cookies
3. Short introduction to JavaScript server-side
4. Short introduction to JavaScript object notation (JSON) (source: <http://json.org/>)

Security issues

Amongst a few issues:

1. Spoofing password authentication

- Normally using pop-up windows to collect data
- Change the status of the browser
- ✱ Modern Browser restrict and control access to such information

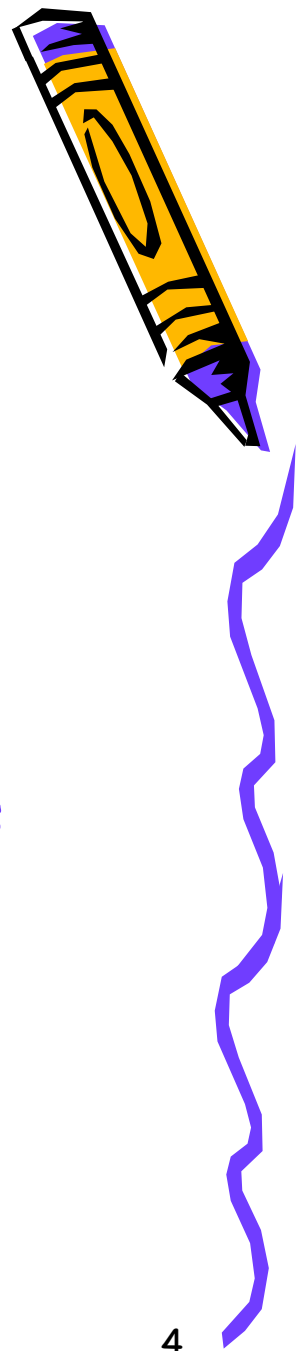
2. Denial-of-Service attacks

- Creating zombie applications that overwhelm the server
- Modal pop-up windows that can not be closed

3. Stealing email addresses and browser history

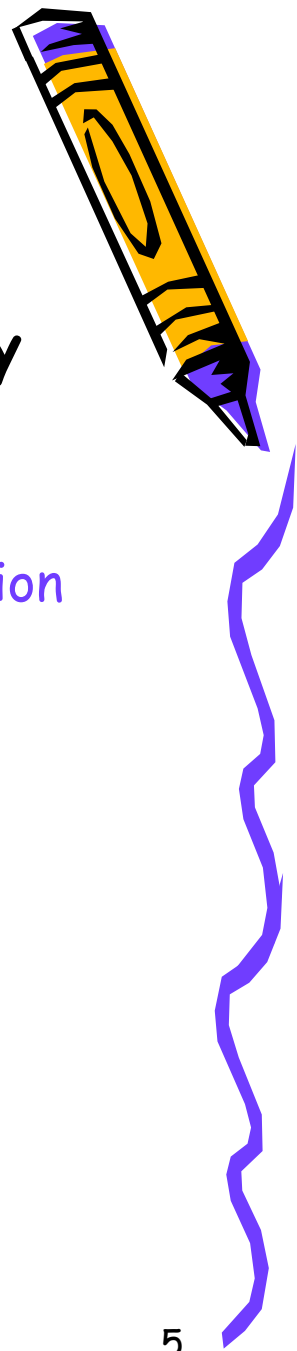


Client-Side JavaScript Security



- **Origin of the code is usually unknown**
 - Is the code trustworthy?
 - Does your browser provide any protection?
- **JavaScript containment**
 - JavaScript is not designed to directly invoke OS commands
 - Does not interfere and create local input/output files
 - Limitations in networking capabilities

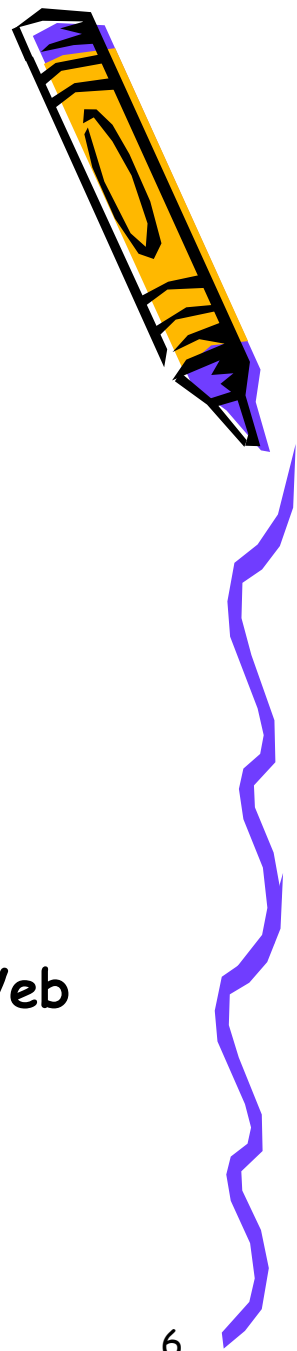
Good Practice digitally signed applications



- **Digital Signature: A digitally encrypted entity that secures the original code**
 - Stops unauthorised tampering of the original application
 - May be allowed more privileges and access to local resources subject to user's permission

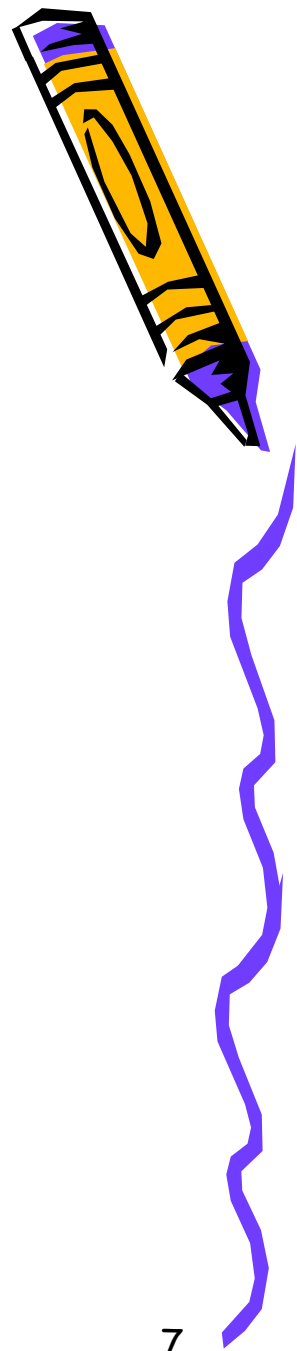
Cookies

- Cookies are a series of data stored by the user's browser (normally 4-5KB)
- There is a limit to the number of cookies that can be stored by the browser
- It is stored as a .txt file
- Web server receive and transmit cookies via http
- Cookies provide client-side state information to the Web application i.e. (shopping basket, remember user preferences, ...)



Cookies attributes

- It is optionally specified
- Used to specify
 - Domain (e.g. `www.mydomain.com`)
 - Path (`.../.../...`)
 - Expires
 - Secure (attribute that sets if the app uses https)



Creating Cookies *setCookie* and *getCookie* function



- The concept of creating cookies is to store and remember a specific information (*e.g. user's name*)
- The first step therefore is to create a function that stores the information of a user (*e.g. first name*)

- *setCookie(cookie_name, value, expiredate)*

```
function setCookie(cookie_name, expiredate)
```

```
{
```

Write the code that would hold the value and expiry date of a cookie

```
}
```

- *getCookie (cookie_name)*

```
function getCookie(cookie_name) {
```

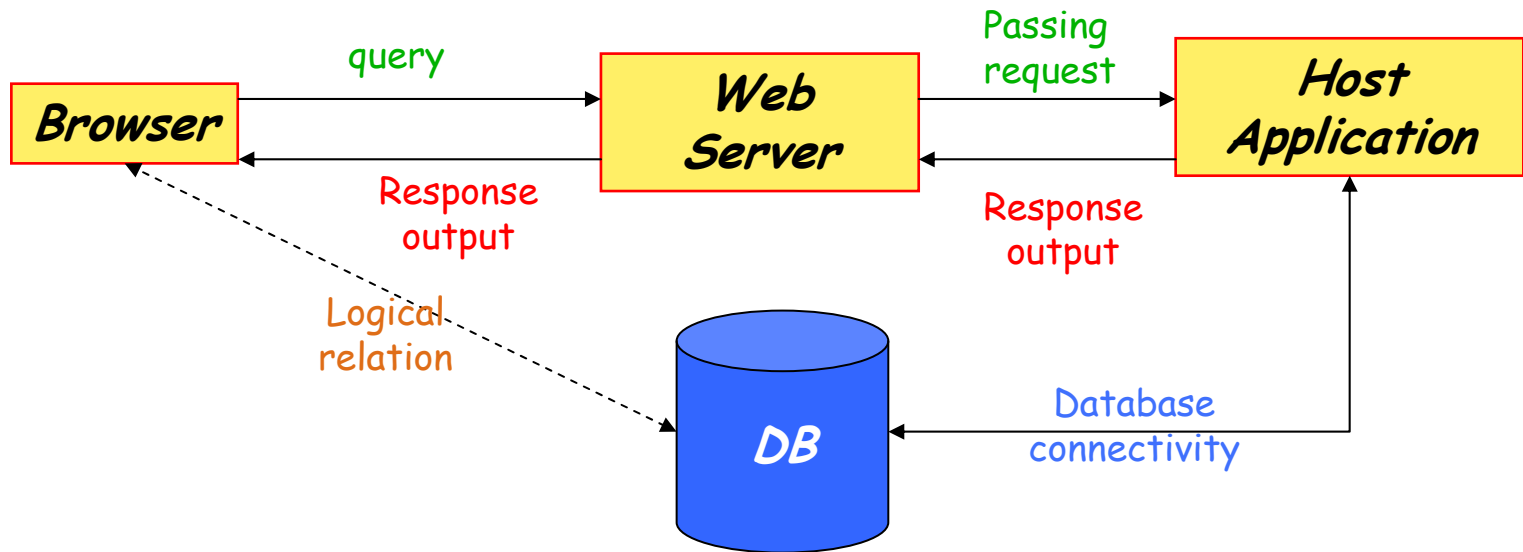
Check if a cookie is stored at all in the document.cookie object and if the cookie is stored return the value

```
}
```

Example: http://www.w3schools.com/js/js_cookies.asp

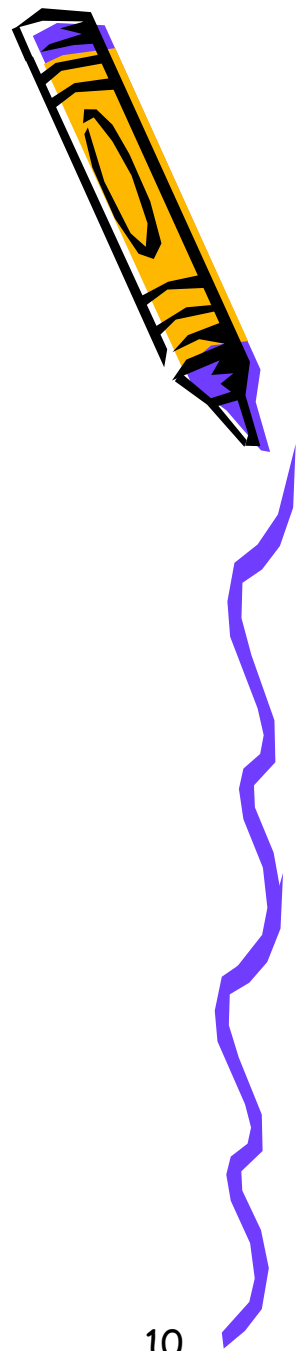
Server-side JavaScript

- Can invoke other programmes on the server
- Output from the program can create dynamic web pages



Server-side programming

- Java server pages (JSP)
- Java servlet APIs
- Active server pages (ASP)
- Interface with databases



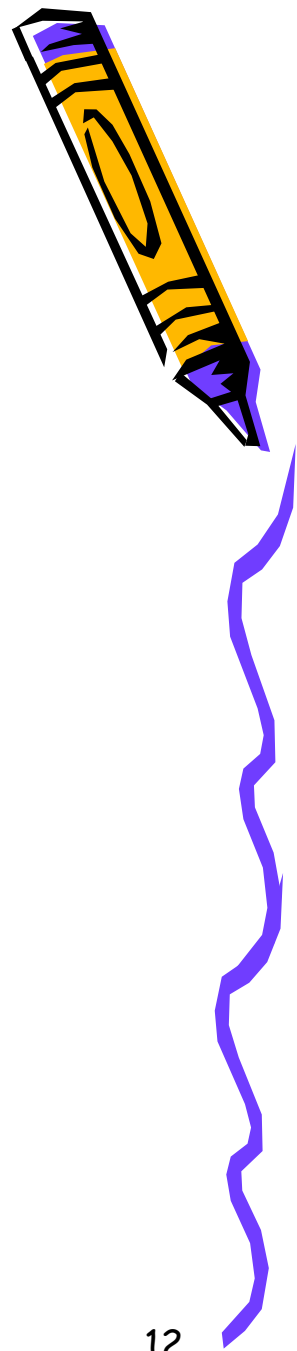
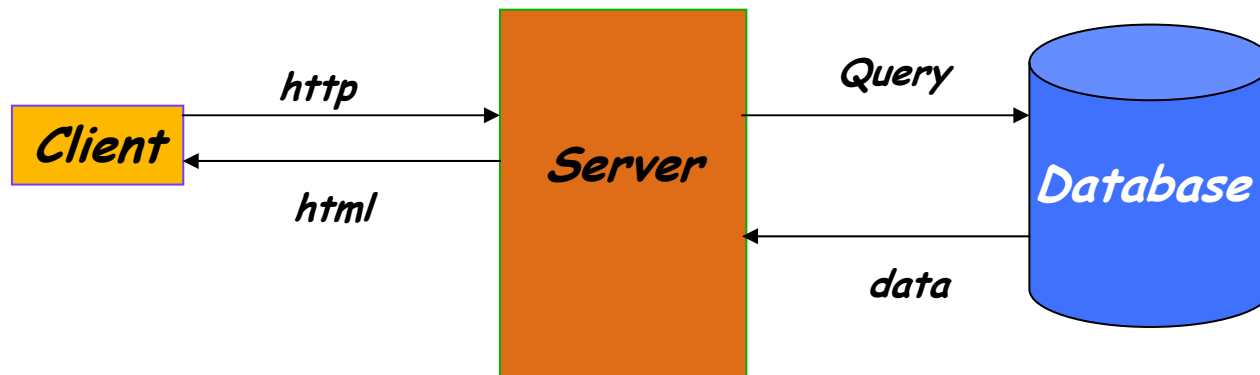
Databases and web applications



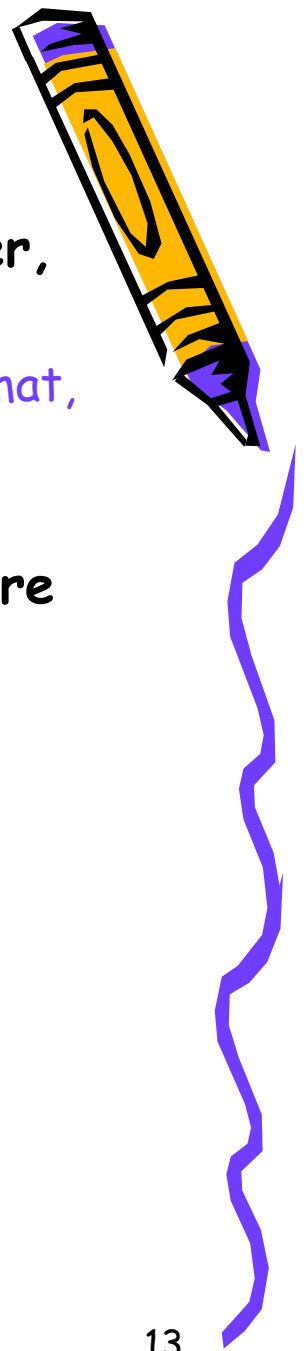
The process:

- Browser submits the data on a form
- Server-side application JavaScript queries the database (SQL)
- Database returns the queried data to the application
- JavaScript application returns the data with the proper html format to the browser

Schematic view



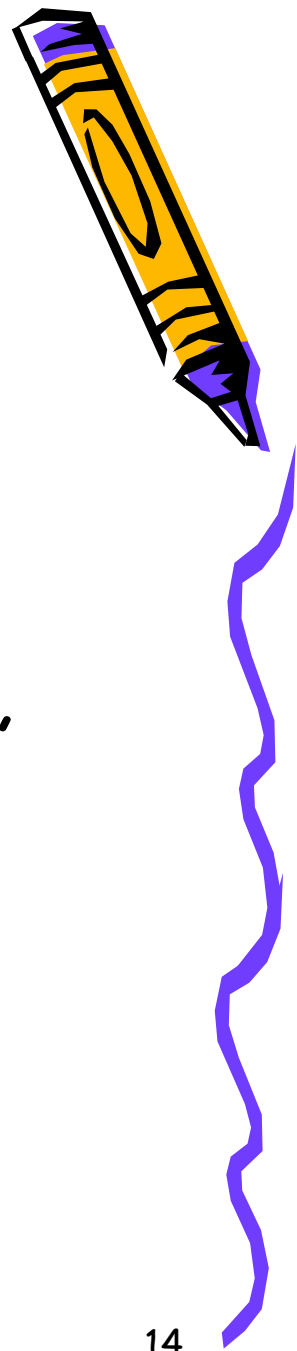
Database design for Web applications



1. **Define and database system (e.g. Access, SQL server, etc.)**
 - Design the tables, relationships, access controls, data format, content,...
 - Configure connections (JDBC, ODBC, native interface)
2. **Design the HTML forms that mirror the data structure in the DB**
 - The forms should be able to query, submit, update, and in general handle the data
 - Have validation capabilities
3. **Develop the JavaScript application to conduct the require operations**
 - Check for errors (validate data)
 - Parse information (get and post methods)
 - Build the SQL commands (embed in functionalities)

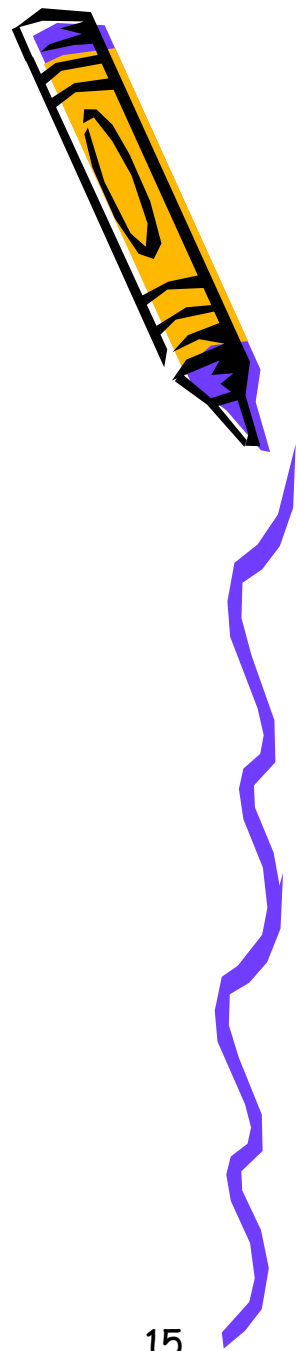
JavaScript Object Notation (JSON)

- Is an ideal data interchange language
- JSON is a text format that is completely language independent
- It works with for example C, C++, C#, Java, JavaScript, Perl, Python, and many others.



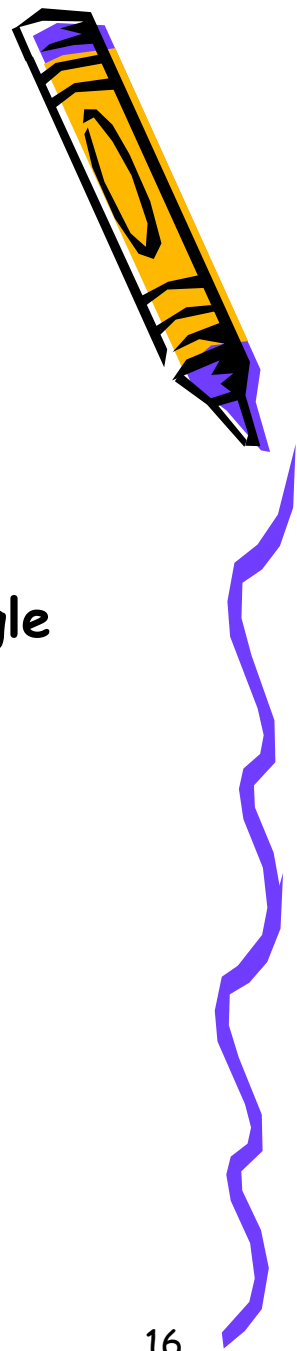
JSON Structure

- JSON is built on two structures:
 - A collection of name/value pairs (*Objects*).
 - Ordered list of values i.e. *array*, *vector*, or *list*

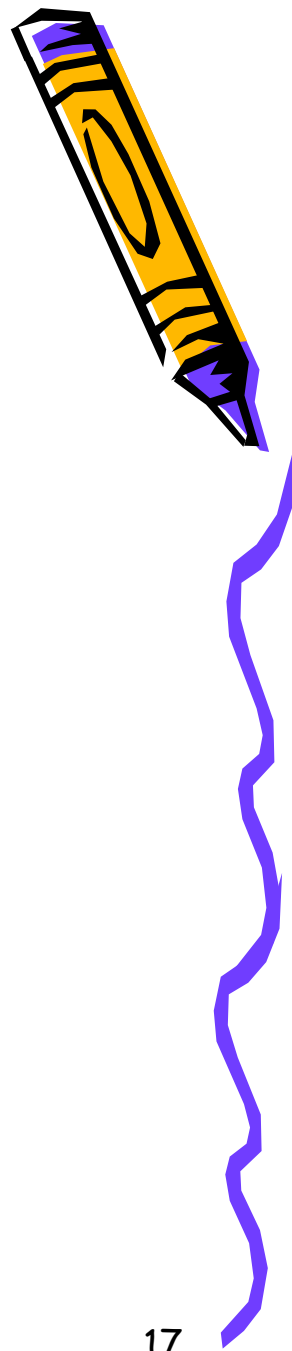


JSON implementation

1. Setup an object access rule using a name/value pair
2. The rule is normally an expression for accessing an object member
3. The rule is normally a function or a string with a single argument which is evaluated at transformation time



JSON Syntax

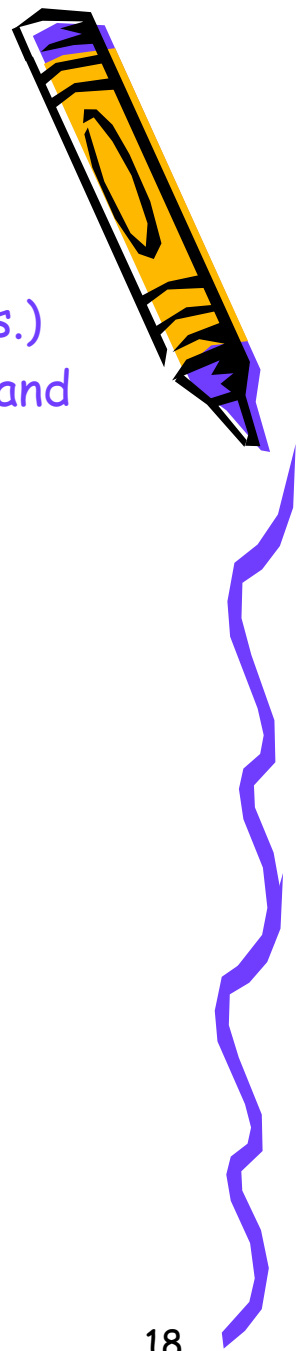


- JSON is designed to pass objects containing name, values, arrays, ...

```
{"Exam": {  
  "Maths": [  
    {"name": "Calculus", "percentage": "30"}, // object access rule  
    {"name": "Algebra", "percentage": "30"}  
    {"name": "Geometry", "percentage": "40"} ]  
  }  
}
```

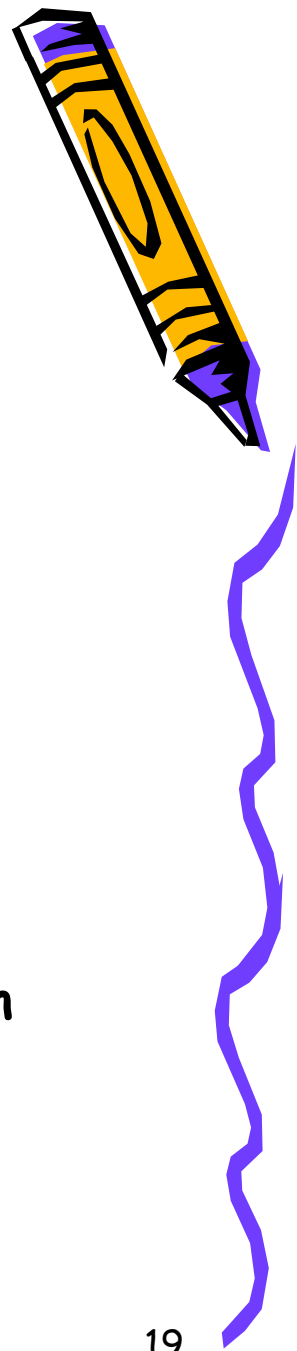
- *{ }* are containers
- *[]* contain arrays
- Names and values are separated by *" : "*
- Array elements are separated by *" , "*

JSON and XML



- **Similarities:**
 - Both are hierarchical. (i.e. you can have values within values.)
 - Both can be parsed and used (most programmes can use it and pass variables)
- **Differences:**
 - In syntax
 - JSON can be parsed by simply using the `eval()` method in JavaScript
 - JSON elements in the array object do not have names of their own
- **If you are using Ajax JSON is quicker to develop.**

Asynchronous JavaScript and XML (AJAX)



- Next week discussion
- Not new probably around since 1997
- A combination of technologies and techniques
 - XML data interchange only
 - Passing JavaScript methods to client
 - DHTML widgets
 - XML & XSLT
- The main philosophy is the asynchronous communication with the server without a page refresh
- *Also see www.adaptivepath.com*

What does AJAX bring

- Advent of broadband allows for more sophisticated applications with complex interfaces
- Recent developments in areas of browser development, search applications, locations based services (e.g. Google Maps)
- People are now designing web applications and not merely web sites and pages
- Next week more on AJAX

