<http://www.json.org/>

JavaScript Object Notation

**JSON** (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the [JavaScript Programming Language](http://javascript.crockford.com/), [Standard ECMA-262 3rd Edition - December 1999](http://www.ecma-international.org/publications/files/ecma-st/ECMA-262.pdf). JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON is built on two structures:

* A collection of name/value pairs. In various languages, this is realized as an *object*, record, struct, dictionary, hash table, keyed list, or associative array.
* An ordered list of values. In most languages, this is realized as an *array*, vector, list, or sequence.

These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is interchangeable with programming languages also be based on these structures.

In JSON, they take on these forms:

An *object* is an unordered set of name/value pairs. An object begins with { (left brace) and ends with } (right brace). Each name is followed by: (colon) and the name/value pairs are separated by , (comma).



An *array* is an ordered collection of values. An array begins with [ (left bracket) and ends with ] (right bracket). Values are separated by , (comma).



A *value* can be a *string* in double quotes, or a *number*, or true or false or null, or an *object* or an *array*. These structures can be nested.



A *string* is a sequence of zero or more Unicode characters, wrapped in double quotes, using backslash escapes. A character is represented as a single character string. A string is very much like a C or Java string.



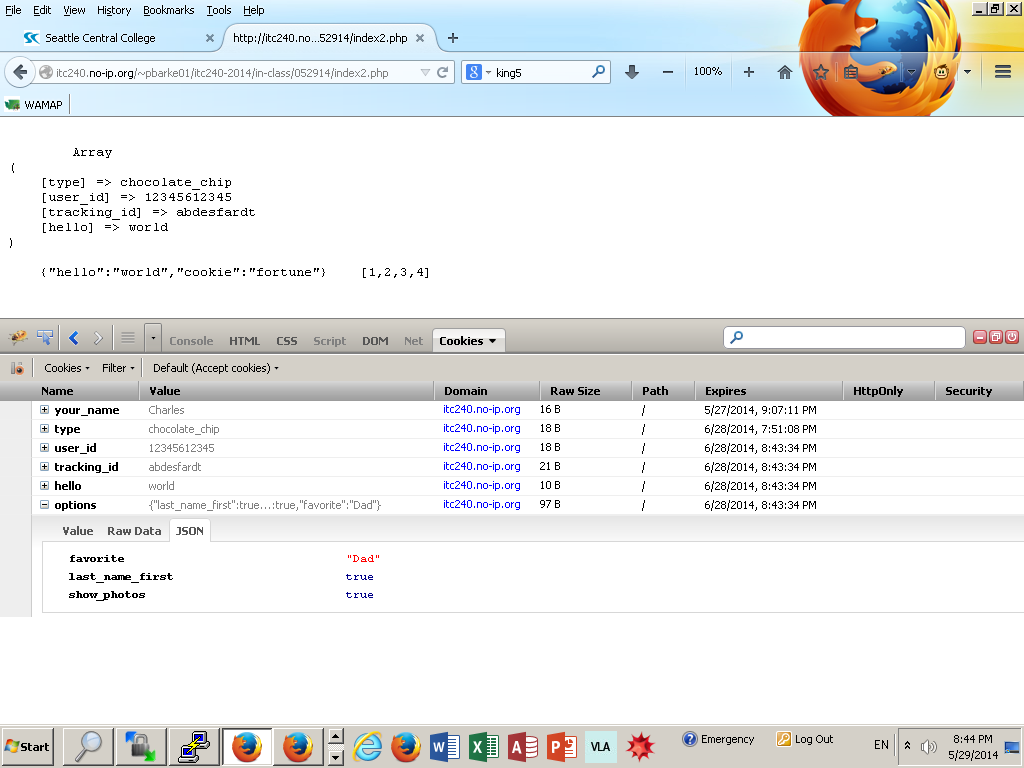
A *number* is very much like a C or Java number, except that the octal and hexadecimal formats are not used.



Whitespace can be inserted between any pair of tokens. Excepting a few encoding details, that completely describes the language.

json\_encode = freeze

json\_decode = thaw



if (isset($\_COOKIE["options"])) {

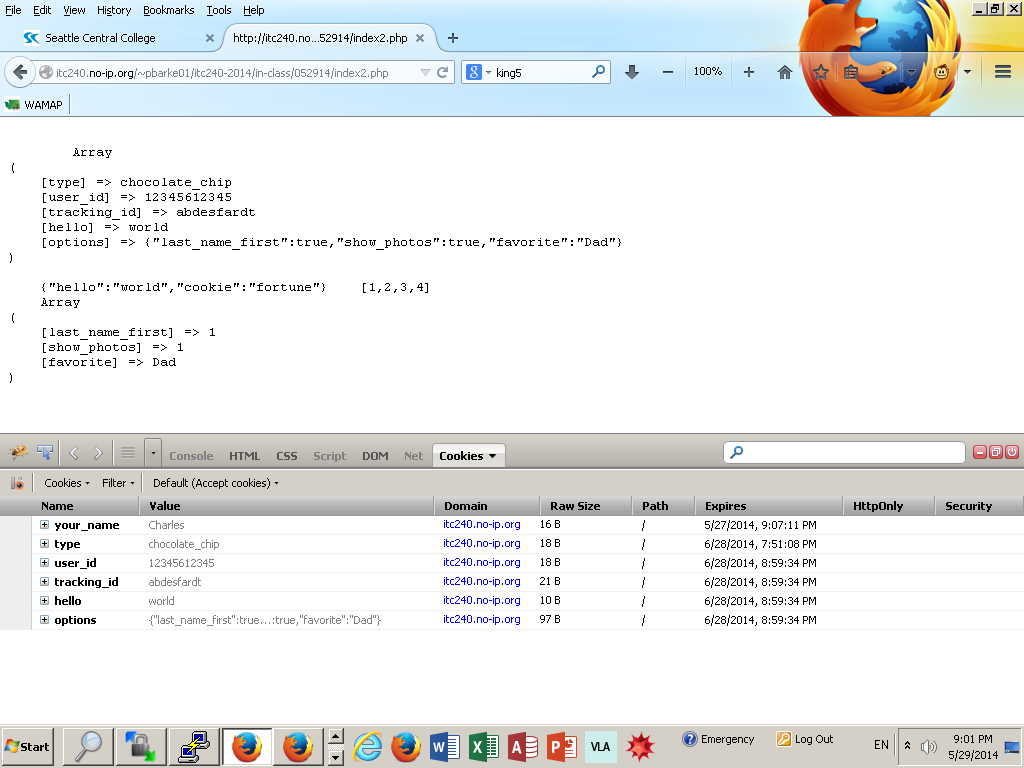
$from \_cookie = json\_decode($\_COOKIE["options"], true); //echo this out below - it turns back into an array when we decode it

}

1. *If there is a cookie named* ***“options”***
2. *Turn that string BACK into an array*
3. *Store that array in* ***$from\_cookie***

Use CSS to have a dark and light theme or small and large version…

The cookies remember and perform accordingly



**Assignment #7 - No Eating in the Library**

This week, we've been learning how to store and retrieve data from the user's browser using cookies. For your assignment, you're going to use these to remember the settings for a simple web app — in this case, a library listings page. This will require you to combine what you've learned in several past assignments.

Your page will show a list of books available for checkout, with the cover, title, and author name for each (see the genre listings for the [Seattle Public Library](http://spl.lib.overdrive.com) for examples). The page will have controls to choose between three sets of options, which should be remembered even if I close my browser and re-open it:

* A choice between a "cover" view with the book art, title, author name, and checkout button, or a "listing" view with art, title, author name, checkout button *and* a short description.
* Preferred sorting, either by title or by author name.
* One of two page themes (such as dark and light, or small and large print)

I'd recommend breaking this assignment into component pieces and addressing them one at a time, instead of trying to address all of them at once.

#cookies.php  
/\*This code will demonstrate use of cookies with PHP  
It is very easy to understand and is better for beginner to  
understand and get idea about power of cookies when used  
with PHP.

Here we give user a form to choose colors he/she  
likes for website and when he/she visits site again within one  
hour his/her settings are saved and read from cookie   
and he/she doesn't have to set the page color and page  
text color again.You can change time from 3600  
seconds to whatever you deem appropriate in your case.  
if you don't understand anything please email me\*/

<?php  
#checking if form has been submitted  
if (isset($\_POST['submitted'])){  
#if yes (form is submitted) assign values from POST array to variables  
$newbgColor=$\_POST['bgColor'];  
$newtxtColor=$\_POST['txtColor'];  
#set cookies  
setcookie("bgColor",$newbgColor,time()+3600);  
setcookie("txtColor",$newtxtColor,time()+3600);  
  
}  
#in case user has come for first time and cookies are not set then  
if ((!isset($\_COOKIE['bgColor']) ) && (!isset($\_COOKIE['txtColor']))){  
$bgColor = "Black";  
$txtColor="White";  
}  
#if cookies are set then use them  
else{  
$bgColor = $\_COOKIE['bgColor'];  
$txtColor = $\_COOKIE['txtColor'];  
}  
?>

<!-- HTML Page-->  
<html>  
<body bgcolor="<?php echo $bgColor ?>" text="<?php echo $txtColor ?>">   
<form action= "<?php echo $\_SERVER['PHP\_SELF']; ?>" method ="POST">

<p>Body Color:</p>  
<select name=bgColor>  
<option value ="Red">Red</option>  
<option value ="Green" selected>Green</option>  
<option value ="Blue">Blue</option>  
<option value ="Yellow">Yellow</option>  
<option value ="Black">Black</option>  
<option value ="Brown">Brown</option>  
<option value ="White">White</option>  
</select>  
<p>Text Color:</p>  
<select name=txtColor>  
<option value ="Red">Red</option>  
<option value ="Green" selected>Green</option>  
<option value ="Blue">Blue</option>  
<option value ="Yellow">Yellow</option>  
<option value ="Black">Black</option>  
<option value ="Brown">Brown</option>  
<option value ="White">White</option>  
</select>  
<input type ="hidden" name="submitted" value="true"></br>  
<input type="submit" value="remind">  
</form>  
</body>  
</html>