JSON: PHP und JavaScript

JSON, kurz für JavaScript Object Notation und gesprochen wie der Name Jason, ist ein kompaktes Computer-Format in für Mensch und Maschine einfach lesbarer Textform, zum Zweck des Datenaustauschs zwischen Anwendungen.

== Wann benötigt man ein solches Format? ==
Wollen wir zum Beispiel ein Personen-Objekt aus unserer C# Desktop Anwendung an unseren PHP Internet Server senden, so benötigen wir neben den Internet Protokollen auch einen Standard um Daten zwischen C# und PHP auszutauschen.

Mit einer Semikolon-getrennten Liste kann man rudmentäre Daten super übertragen und in beiden Programmiersprachen ist es ein leichtes den Datensatz zu verwenden.

Quellcode

```
1. Max;Mustermann;Musterstraße 11;11111 Musterstadt;
```

Verwenden wir stattdessen eine komplizierte, mehrdimensionale Struktur so sind diese s.g. CSV Datensätze nicht zu gebrauchen.
Eine Lösung muss her: In diesem Beitrag wird die Lösung JSON anhand einer häufig verbreiteten Kombination "JavaScript und PHP" erläutert.

Vergleichen wir zuerst wie wir eine Liste in beiden Sprachen ausgeben.

== Liste in PHP verarbeiten ==

Quellcode

```
1. <?php
2. // PHP Datenerhebung
3. $data = array(
4. array(
5. 'firstname' => 'Max',
6. 'lastname' => 'Mustermann',
7. 'websites' =>
8. array(
9. 'http://www.easy-coding.de',
10. 'http://www.coder-suche.de'
11. )
12. ),
13. array(
14. 'firstname' => 'Larry',
15. 'lastname' => 'Page',
16. 'websites' =>
17. array(
18. 'http://www.google.de',
19. 'http://www.google.com',
20. 'http://www.google.co.uk'
21. )
22. )
23. );
24. // PHP Daten ausgeben
26. foreach($data as $row) {
27. echo $row['firstname'];
28. echo "<ul>";
29. foreach($row['websites'] as $url) {
30. echo "<li>{$url}</li>";
```

```
31. }
32. echo "</ul>";
33. }
34. ?>
```

Alles anzeigen

== Liste in JavaScript verarbeiten ==

Quellcode

```
1. <script type="text/javascript">
2. // JavaScript Datenerhebung
3. var data = new Object(
4. {
5. firstname : 'Max',
6. lastname : 'Mustermann',
7. websites : new Array(
8. 'http://www.easy-coding.de',
9. 'http://www.coder-suche.de'
10. )
11. },
12. {
13. firstname : 'Larry',
14. lastname : 'Page',
15. websites : new Array(
16. 'http://www.google.de',
17. 'http://www.google.com',
18. 'http://www.google.co.uk'
19. )
20. }
21. );
22. // JavaScript Daten ausgeben
24. for(var row in data) {
25. var row = data[row];
26. document.writeln(row['firstname']);
27. document.writeln("<ul>");
28. for(var url in row['websites']) {
29. url = row['websites'][url];
30. document.writeln("<li>"+url+"</li>");
31. }
32. document.writeln("</ul>");
33. }
34. </script>
```

Alles anzeigen

== Brücke von PHP nach JavaScript schlagen ==

Sieht man von den syntaktischen Unterschieden der Sprachen ab, haben wir das selbe Beispiel mit zwei verschiedenen Programmiersprachen umgesetzt.
In diesem Schritt wird die Brücke von PHP nach JavaScript geschlagen.
Als Eingabestream werden wir nicht mehr das JavaScript Array "data" - sondern serialisieren stattdessen das PHP Array zu einem JSON String.

Quellcode

```
<?php
// PHP Datenerhebung
$data = array(
array(
'firstname' => 'Max',
'lastname' => 'Mustermann',
'websites' =>
array(
'http://www.easy-coding.de',
'http://www.coder-suche.de'
)
),
array(
'firstname' => 'Larry',
'lastname' => 'Page',
'websites' =>
array(
'http://www.google.de',
'http://www.google.com',
'http://www.google.co.uk'
)
)
);
// Brücke zwischen PHP und JavaScript (serverseitig)
$json = json_encode($data);
?>
<script type="text/javascript">
// JavaScript Datenerhebung
var json = '<?= $json ?>';
// Brücke zwischen PHP und JavaScript (clientseitig)
var data = eval('(' + json + ')');
// JavaScript Daten ausgeben
for(var row in data) {
var row = data[row];
document.writeln(row['firstname']);
document.writeln("<ul>");
for(var url in row['websites']) {
url = row['websites'][url];
document.writeln("<li>"+url+"</li>");
}
document.writeln("</ul>");
}
</script>
```

Alles anzeigen

Im Klartext sieht der JSON String folgendermaßen aus:

Quellcode

```
var json = '[{"firstname":"Max","lastname":"Mustermann","websites":["http:\/\/www.easy-coding.de","http:\/\/www.coder-suche.de"]},
{"firstname":"Larry","lastname":"Page","websites":["http:\/\/www.google.de","http:\/\/www.google.com","http:\/\/www.google.co
```

== Ältere PHP Versionen ==

PHP 5 bringt ein Modul mit, dass, weil es in C geschrieben ist, sehr performant arbeitet. Leider ist es nicht in allen Installationen vorhanden. Daher würde ich euch empfehlen zweigleisig zu fahren und eine Schnittstelle einzuführen, die bei Existenz der C-Funktion diese bevorzugt, bei Nichtexistenz aber eine andere verwendet.
Diese Schnittstelle sieht folgendermaßen aus:

Quellcode

```
require_once 'Services_JSON.php';
class JSON {
/**
* @param objekt -> PHP Objekt/Array/Variable
* @param force -> Benutzung von Service_JSON erzwingen
* @return -> JSON String
*/
static public function encode($objekt, $force=false) {
if(!function_exists('json_encode') || $force) {
$tmp = new Services_JSON();
return $tmp->encode($objekt)
} else {
return json_encode($objekt);
}
}
/**
* @param str -> JSON String
* @param force -> Benutzung von Service_JSON erzwingen
* @return -> PHP Objekt/Array/Variable
*/
static public function decode($str, $force=false) {
if(!function_exists('json_decode') || $force) {
$tmp = new Services_JSON();
return $tmp->decode($str)
} else {
return json_decode($str);
}
}
}
```

Alles anzeigen

Es verwendet Services_JSON, das unter der BSD Lizenz steht und hier erhältlich ist: pear.php.net/pepr/pepr-proposal-show.php?id=198
Nun zu einer Beispielimplementierung:

Quellcode

```
<?php
require_once 'JSON.class.php';
class Ob {
public $arr = array("hans"=>"123", "peter"=>"456");
public $x = 5;
}
$ob = new Ob();
echo JSON::encode($ob);
// Ausgabe = {"arr":{"hans":"123","peter":"456"},"x":5}
```