

How To Generate JSON With PHP and Parse It With JQuery

FEB. 12, 2010

JSON has become one of, if not the best, format for transmitting data across web applications. The thing I like about JSON over XML is that it's light weight and that it can easily be parsed with my favorite Javascript library, JQuery.

I've used JSON in many projects which have involved generating user data, this can give your app enormous flexibility because it's just like having your own little API. In this tutorial we will generate JSON with PHP and parse it with JQuery.

The JSON-PHP File (posts.php)

The JSON data we will make with PHP will be nothing more than a series of posts which we will then read onto a page.

Our posts will have the fields headline, body, posted_on, and posted_by.

```
<?php
$json='
{"posts":[
    {
        "headline":"The Headline for post 1",
        "body":"Lorem ipsum dolor sit amet <br />consectetur adipiscing elit.<br /> adipiscing
        "posted_on":"Feb 11, 2010",
        "posted_by":"username"
    },
    {
        "headline":"The Headline for post 2",
        "body":"Lorem ipsum dolor sit amet <br />consectetur adipiscing elit.",
        "posted_on":"Feb 12, 2010",
        "posted_by":"anotheruser"
    }
]}
';
```

```
echo $json;
```

The JSON Parser File (index.html)


The file we will use to parse will be an HTML file since we won't need any PHP, but you can use the PHP extension if your script requires it.

This file is also a simple one, we will parse our posts to a div with the id content and we will style our posts as we parse using only HTML tags.

Here is the file before we include the jquery which will do all the heavy work.

```
<html>
<body>
    <div id="content">

    </div>
    <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.4.1/jque
</body>
</html>
```



We will parse our posts on load, to do this we will make use of JQuery's \$.getJSON and \$.each.

The first function, getJSON, needs a URL (posts.php) where our data resides, it will also need a callback function whose parameter is the data returned from the url. I've named this parameter accordingly but you can change the name if you prefer. Our \$.each function will be put inside getJSON's callback and it will loop through and add tags to the results and append them to the div content.

The \$.getJSON Function

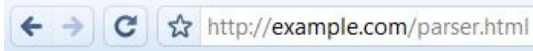
```
<script type="text/javascript">
    $(document).ready(function() {
        var url="posts.php";
        $.getJSON(url,function(json) {
                                // loop through the posts here
        });
    });
</script>
```

The \$.each loop

Just like getJSON, \$.each has two parameters, first you have to give it the data then the callback function which is used to parse the data. The letter "i" in its callback function keeps count of the results while the post variable represents the current results.

```
$.each(json.posts,function(i,post) {
    $("#content").append(
        '<div class="post">' +
        '<h1>' + post.headline + '</h1>' +
        '<p>' + post.body + '</p>' +
        '<p>added: <em>' + post.posted_on + '</em></p>' +
        '<p>posted by: <strong>' + post.posted_by + '</strong></p>' +
        '</div>'
    );
});
```

The Result: Beautifully Styled PHP+JSON Generated Posts



The Headline for post 1

Lorem ipsum dolor sit amet
consectetur adipiscing elit.
adipiscing elit.

added: *Feb 11, 2010*

posted by: **username**

The Headline for post 2

Lorem ipsum dolor sit amet
consectetur adipiscing elit.

added: *Feb 12, 2010*

posted by: **anotheruser**

I hoped you enjoyed this tutorial. It's not hard to see how you can generate JSON from a MySQL table, but drop a comment if you have any questions.