JavaScript Object Notation

Introduction

This extension implements the » JavaScript Object Notation (JSON) data-interchange format. The decoding is handled by a parser based on the JSON_checker by Douglas Crockford.

Installing/Configuring

Requirements

As of PHP 5.2.0, the JSON extension is bundled and compiled into PHP by default.

Information for installing this PECL extension may be found in the manual chapter titled Installation of PECL extensions. Additional information such as new releases, downloads, source files, maintainer information, and a CHANGELOG, can be located here: » http://pecl.php.net/package/json

Installation

There is no installation needed to use these functions; they are part of the PHP core.

Runtime Configuration

This extension has no configuration directives defined in *php.ini*.

Resource Types

This extension has no resource types defined.

Predefined Constants

This extension has no constants defined.

JSON Functions

json_decode

json_decode -- Decodes a JSON string

Description

mixed **json_decode** (string `$json` [, bool `$assoc`])

Takes a JSON encoded string and converts it into a PHP variable.

Parameters

`json`
The *`json`* string being decoded.

`assoc`
When **TRUE**, returned object s will be converted into associative array s.

Return Values

Returns an object or if the optional *`assoc`* parameter is **TRUE**, an associative array is instead returned.

Examples

Example #1 - json_decode() examples

```
<?php
$json = '{"a":1,"b":2,"c":3,"d":4,"e":5}';

var_dump(json_decode($json));
var_dump(json_decode($json, true));

?>
```

The above example will output:

```
object(stdClass)#1 (5) {
   ["a"] => int(1)
   ["b"] => int(2)
   ["c"] => int(3)
   ["d"] => int(4)
   ["e"] => int(5)
}

array(5) {
   ["a"] => int(1)
```

```
   ["b"] => int(2)
   ["c"] => int(3)
   ["d"] => int(4)
   ["e"] => int(5)
}
```

Example #2 - Another example

```
<?php

$json = '{"foo-bar": 12345}';

$obj = json_decode($json);
print $obj->{'foo-bar'}; // 12345

?>
```

Notes

Caution

This function will return false if the JSON encoded data is deeper than 127 elements.

ChangeLog

Version	Description
5.2.3	The nesting limit was increased from 20 to 128

See Also

- json_encode()

json_encode

json_encode -- Returns the JSON representation of a value

Description

string **json_encode** (mixed `$value`)

Returns a string containing the JSON representation of *`value`*.

Parameters

`value`
: The *`value`* being encoded. Can be any type except a resource. This function only works with UTF-8 encoded data.

Return Values

Returns a JSON encoded string on success.

ChangeLog

Version	Description
5.2.1	Added support to JSON encode basic types

Examples

Example #3 - A json_encode() example

```
<?php
$arr = array ('a'=>1,'b'=>2,'c'=>3,'d'=>4,'e'=>5);

echo json_encode($arr);
?>
```

The above example will output:

```
{"a":1,"b":2,"c":3,"d":4,"e":5}
```

See Also

- json_decode()