PHP – Deprecated Features

As some new features are added with each new version, some features are also removed as they are deemed to be obsolete. In this chapter, we have a look at deprecated features after PHP version 5.

Deprecated in PHP Ver 7

PHP 4 Style Constructors

PHP 4 style Constructors are methods having same name as the class they are defined in, are now deprecated, and will be removed in the future. PHP 7 will emit E_DEPRECATED if a PHP 4 constructor is the only constructor defined within a class. Classes implementing a __construct() method are unaffected.

Example

Take a look at the following example –

```
<?php
   class A {
      function A() {
         print('Style Constructor');
      }
   }
?>
```

It produces the following **output** on the browser –

```
Deprecated: Methods with the same name as their class will not be
constructors in a future version of PHP; A has a deprecated constructor in...
```

Static Calls to Non-static Methods

Static calls to non-static methods are deprecated, and may be removed in the future.

Example

Take a look at the following example –

```php
<?php
   class A {
      function b() {
         print('Non-static call');
      }
   }
   A::b();
?>
```

It produces the following **output** on the browser –

```
Deprecated: Non-static method A::b() should not be called statically in...
Non-static call
```

password_hash() salt option

The salt option for the password_hash() function has been deprecated so that the developers do not generate their own (usually insecure) salts. The function itself generates a cryptographically secure salt, when no salt is provided by the developer - thus custom salt generation is not required any more.

capture_session_meta SSL context option

The capture_session_meta SSL context option has been deprecated. SSL metadata is now used through the stream_get_meta_data() function.

ext/mcrypt

The mcrypt extension has been deprecated in favour of OpenSSL.

Unquoted Strings

Unquoted strings that are non-existent global constants are taken to be strings of themselves. This behaviour used to emit an E_NOTICE, but will now emit an E_WARNING. In the next major version of PHP, an Error exception will be thrown instead.

The __autoload() Method

The __autoload() method has been deprecated because it is inferior to spl_autoload_register() (due to it not being able to chain autoloaders), and there is no interoperability between the two autoloading styles.

The create_function() Function

Given the security issues of this function has now been deprecated. The preferred alternative is to use anonymous functions.

The each() Function

This function causes implementation issues for some language changes. It has therefore been deprecated.

Case-Insensitive Constants

The declaration of case-insensitive constants has been deprecated. Passing true as the third argument to define() will now generate a deprecation warning.

The (real) and is-real() Function

The (real) cast is deprecated, use (float) instead. The is_real() function is also deprecated, use is_float() instead.

The "parent" Leyword

Using parent inside a class without a parent is deprecated, and will throw a compile-time error in the future. Currently an error will only be generated if/when the parent is accessed at run-time.

Deprecated in PHP Ver 8

If a parameter with a default value is followed by a required parameter, the default value has no effect. This is deprecated as of PHP 8.0.0 and can generally be resolved by dropping the default value, without a change in functionality –

```
<?php
   function test($a = [], $b) {}  // Before
   function test($a, $b) {}       // After
?>
```

One exception to this rule are parameters of the form Type $param = null, where the null default makes the type implicitly nullable. This usage remains allowed, but it is recommended to use an explicit nullable type instead –

```
<?php
   function test(A $a = null, $b) {}    // Still allowed
```

```
    function test(?A $a, $b) {}         // Recommended
?>
```

Calling get_defined_functions() with exclude_disabled explicitly set to false is deprecated and no longer has an effect. get_defined_functions() will never include disabled functions.

Sort comparison functions that return true or false will now throw a deprecation warning, and should be replaced with an implementation that returns an integer less than, equal to, or greater than zero.

```
<?php
    // Replace
    usort($array, fn($a, $b) => $a > $b);
    // With
    usort($array, fn($a, $b) => $a <=> $b);
?>
```

Implicit Incompatible float to int Conversions

The implicit conversion of float to int which leads to a loss in precision is now deprecated. This affects array keys, int type declarations in coercive mode, and operators working on ints.

Calling a Static Element on a Trait

Calling a static method, or accessing a static property directly on a trait is deprecated. Static methods and properties should only be accessed on a class using the trait.

Date Functions

date_sunrise() and date_sunset() have been deprecated. Use date_sun_info() instead.

strptime() has been deprecated. Use date_parse_from_format() instead (for locale-independent parsing), or IntlDateFormatter::parse() (for locale-dependent parsing).

strftime() and gmstrftime() have been deprecated. You can use date() instead (for locale-independent formatting), or IntlDateFormatter::format() (for locale-dependent formatting).

Dynamic Properties

The creation of dynamic properties is deprecated. Instead, use stdClass that allows dynamic properties.