serializable

ENTITY

Interface for customized serializing.

Classes that implement this interface no longer support __sleep() and __wakeup(). The serialize method is called whenever an instance needs to be serialized. This does not invoke __destruct or have any other side effect unless programmed inside the method. When the data is unserialized the class is known and the appropriate unserialize method is called as a constructor instead of calling __construct. The constructor can be called from within the unserialize method if desired.

> As of PHP 8.1.0, a class which implements Serializable without also implementing __serialize() and __unserialize() will generate a deprecation warning.

New code should use the __serialize() and __unserialize() magic methods instead, available as of PHP 7.4.0. When a class declares both them and this interface, `serialize` always uses __serialize() and never reaches Serializable::serialize. Unserializing is decided by the format of the data instead: a payload written by PHP 7.4.0 or later is read by __unserialize(), one written before it is still read by Serializable::unserialize. Implementing Serializable therefore remains of use for reading such older payloads, and for satisfying a Serializable type declaration; declaring all four methods covers every case, and avoids the deprecation notice.

> The magic methods have no interface of their own, so a class which declares them without implementing Serializable is not an instance of it. `method_exists` is the way to detect them.

ENTITY

Serializable

ENTITY

ENTITY

Supporting PHP 7.1.0 through 7.3.0

The magic methods carry the serialized form, and the interface methods delegate to them, so that a single representation serves both.

```
<?php
class Task implements Serializable
{
    private $label;

    public function __construct($label)
    {
        $this->label = $label;
    }

    public function __serialize(): array
    {
        return ['label' => $this->label];
    }

    public function __unserialize(array $data): void
    {
        $this->label = $data['label'];
    }

    // Never called as of PHP 7.4.0
    public function serialize()
    {
        return serialize($this->__serialize());
    }

    // Still called as of PHP 7.4.0, for data written before it
    public function unserialize($data)
    {
        $this->__unserialize(unserialize($data));
    }
}
```

```
var_dump(serialize(new Task('deploy')));
?>
```

ENTITY

```
string(40) "O:4:"Task":1:{s:5:"label";s:6:"deploy";}"
```

Prior to PHP 7.4.0, the same code serializes through the interface instead, and outputs `string(47) "C:4:"Task":31:{a:1:{s:5:"label";s:6:"deploy";}}"`.

ENTITY ENTITY

□□□ #2
□ □□□ □□□ 2026-10-31 17:45:35 UTC
□ □□□ □□□ 2026-10-31 17:45:37 UTC