OOP

Object-oriented programming

Was ist OOP in PHP?

klar aus Java…

Klassen und Objekte

gleich wie in Java

Klasse erstellen

ähnlich wie in Java

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Objekte erstellen

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

$this

$this referenziert das aktuelle Objekt -> wie in Java

instanceof

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

\_\_construct Funktion

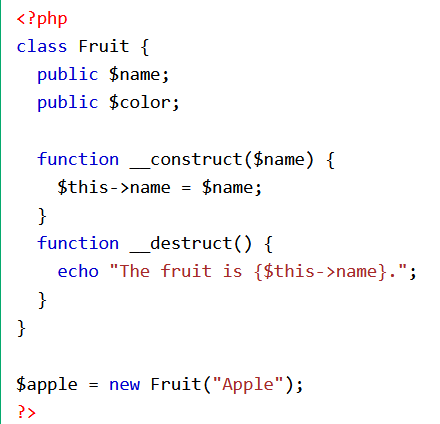
Konstruktor

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

\_\_destruct Funktion

wenn eine \_\_destruct Funktion erstellt wurde, wird sie automatisch am Ende des Skripts ausgeführt.



Berechtigungs-Modifikatoren

* public
* protected
* private

wie in Java…

Vererbung

wie in Java

extends

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Override

einfach Methodenkopf genau gleich schreiben, kein Keyword wird benötigt

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

final

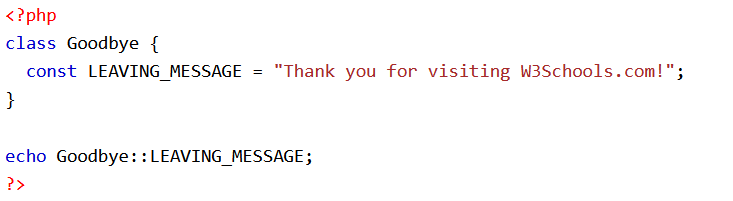
Klasse kann nicht extended werden

Klassen-Konstanten

können nicht mehr verändert werden.

const Keyword

Konstanten sollten komplett in Upper-Case geschrieben werden



eine Konstante innerhalb der Klasse kann mit self Keyword aufgerufen werden

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abstrakte Klassen und Methoden

wieder wie in Java

Kinder-Klassen müssen Methoden implementieren.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Wenn Klassen von abstrakten Klassen erben, müssen Methoden genau gleich deklariert werden und den gleichen oder einen nicht so restriktiven Berechtigungs-Modifikator verwenden. Wenn Methode in abstrakter Klasse protected ist, muss in Kind-Klasse entweder protected oder public verwendet werden.

Interfaces

wie in Java

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Interfaces benutzen

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Traits

PHP unterstützt nur einfache Vererbung. Eine Klasse kann nur von einer anderen erben.

Traits lösen dieses Problem.

Traits werden benutzt, um Methoden zu deklarieren die in mehreren Klassen verwendet werden können. Traits können Methoden und abstrakte Methoden haben, die in mehreren Klassen implementiert werden.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Mehrere Traits

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Statische Methoden

können direkt aufgerufen werden, ohne eine Instanz der Klasse erzeugen zu müssen

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Statische Eigenschaften

können auch direkt aufgerufen werden

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Namespaces

Namespaces lösen 2 Probleme:

1. Erlauben bessere Organisation, indem Klassen gruppiert werden können

2. Erlauben, dass derselbe Name in mehreren Klassen verwendet werden kann.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Namespaces benutzen

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung