Datové Typy v PHP

PHP podporuje několik základních datových typů:

Integer (celočíselný datový typ): Tento datový typ reprezentuje celá čísla. Může být kladné, záporné nebo nulové. Příklad: $number = 42;

Float (datový typ s plovoucí desetinnou čárkou): Float reprezentuje čísla s desetinnou čárkou. Příklad: $pi = 3.14;

String (řetězcový datový typ): String reprezentuje text. Text v PHP musí být v jednoduchých nebo dvojitých uvozovkách. Příklad: $text = "Hello, World!";

Boolean (logický datový typ): Boolean reprezentuje buď pravda (true) nebo nepravda (false). Používá se pro podmínky a logické operace. Příklad: $isTrue = true;

Array (pole): Pole umožňuje ukládat více hodnot do jedné proměnné. Příklad: $colors = ["red", "green", "blue"];

Array v php je podobné idicitionary v c#

Object (objektový datový typ): Objekt reprezentuje instanci třídy. Objektově orientované programování v PHP je pokročilejší téma, a na něj se podíváme níže.

Objektově Orientované Programování (OOP) v PHP

V PHP můžete vytvářet objekty a třídy, což je hlavní koncept objektově orientovaného programování. Zde jsou základní prvky objektového programování v PHP:

Třídy (Classes): Třída je šablona pro vytváření objektů. Obsahuje metody a vlastnosti, které objekty budou sdílet.

php

class Car {

public $brand;

public $model;

public function startEngine() {

echo "Engine started!";

}

}

Objekty (Objects): Objekty jsou instance tříd. Můžete vytvořit objekt z třídy a pracovat s ním.

php

Copy code

$myCar = new Car();

$myCar->brand = "Toyota";

$myCar->model = "Camry";

$myCar->startEngine();

Zapouzdření (Encapsulation): Toto je princip, který umožňuje skrýt vlastnosti třídy před vnějším světem a nastavit metody pro přístup k nim.

Dědičnost (Inheritance): Dědičnost umožňuje vytvářet nové třídy na základě existujících tříd. Nová třída může zdědit vlastnosti a metody rodičovské třídy. Implements se používá pro implementaci interface , extends je místo : pro dědičnost

Polymorfizmus (Polymorphism): Polymorfizmus umožňuje jednomu rozhraní nebo třídě mít více různých implementací, např: class person má metodu hello , class child , která dědí má také metodu hello , pokud vytvoříme objekt child , metoda v class child přepíše metodu v class person