Ereditarietà

Il concetto di Ereditarietà è fondamentale nella programmazione ad oggetti. Tramite questa proprietà è possibile fare in modo che una classe-figlio ottenga in maniera automatica (quindi ereditando) sia i metodi che le proprietà da un'altra classe-padre.

Diciamo che una scuola è formata da docenti, amministrativi, addetti mensa e ovviamente alunni. Sia alunni che docenti sono persone, giusto? Sia alunni che docenti hanno nome, cognome ed età.

```
<?php
class Person
{

  public $firstname;
  public $lastname;
  public $age;



  public function __construct($name, $surname, $age)
  {
    $this→firstname = $name;
    $this→lastname = $surname;
    $this→age = $age;
  }
}

$person1 = new Person('Francesco', 'Mansi', '31');
print_r($person1);
```

Uno studente avrà le materie che segue e una media.

Quindi Studenti estende la classe Persona.

```
<?php
class Person
{

  public $firstname;
  public $lastname;
  public $age;



  public function __construct($name, $surname, $age)
  {
    $this->firstname = $name;
    $this->lastname = $surname;
    $this->age = $age;
  }
}


class Student extends Person
{
  public $average;
  public $subjects = [];

  public function __construct($name, $surname, $age, $average, $subject
s)
  {
    //$this->firstname = $name;
    //$this->lastname = $surname;
    //$this->age = $age;
    parent::__construct($name, $surname, $age);
    $this->average = $average;
    $this->subjects = $subjects;
  }
}
```

```
$student1 = new Student('Carlo', 'Rossi', 18, 9, ['Italiano']);
print_r($student1);
```

Metodi ereditati & Sovrascrittura

```
<?php
class Person
{

  public $firstname;
  public $lastname;
  public $age;




  public function __construct($name, $surname, $age)
  {
    $this->firstname = $name;
    $this->lastname = $surname;
    $this->age = $age;
  }

    public function speak(){
     echo "Piacere, $this->firstname $this->lastname";
    }
}


class Student extends Person
{
  public $average;
  public $subjects = [];

  public function __construct($name, $surname, $age, $average, $subject
s)
  {
    parent::__construct($name, $surname, $age);
    $this->average = $average;
    $this->subjects = $subjects;
  }
}

$student1 = new Student('Carlo', 'Rossi', 18, 9, ['Italiano']);
print_r($student1);
```

Estendere infinite volte la classe Person

```
<?php
class Person
{

  public $firstname;
  public $lastname;
  public $age;




  public function __construct($name, $surname, $age)
  {
    $this->firstname = $name;
    $this->lastname = $surname;
    $this->age = $age;
  }

    public function speak(){
     echo "Piacere, $this->firstname $this->lastname";
    }
}



class Student extends Person
{
  public $average;
  public $subjects = [];

  public function __construct($name, $surname, $age, $average, $subject
s)
  {
    parent::__construct($name, $surname, $age);
    $this→average = $average;
    $this→subjects = $subjects;
  }
}

class Teacher extends Person
{
  public $salary;
  public $subjects = [];

  public function __construct($name, $surname, $age, $salary, $subjects)
  {
    parent::__construct($name, $surname, $age);
    $this→salary = $salary;
    $this→subjects = $subjects;
  }
}
$teacher1 = new Student('Carlo', 'Rossi', 18, 9000, ['Italiano']);
$student1 = new Student('Carlo', 'Bianchi', 18, 10, ['Italiano']);
print_r($student1);
print_r($teacher1);
```

Matrioska Class

```
<?php
class Continent
{

  public $name_of_continent;

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

  public function geoLoaction()
  {
    echo "Mi trovo in $this->name_of_continent \n";
  }

  public function sayHello()
  {
    echo "Ciao \n";
  }
}

class Country extends Continent
{
  public $name_of_country;

  public function __construct($continente, $nazione)
  {
    parent::__construct($continente);
    $this->name_of_country = $nazione;
  }
  public function geoLoaction()
  {
    echo "Mi trovo in $this->name_of_continent, $this->name_of_country \n";
  }
}

class Region extends Country
{
  public $name_of_region;

  public function __construct($continente, $nazione, $regione)
  {
    parent::__construct($continente, $nazione);
    $this->name_of_region = $regione;
  }
  public function geoLoaction()
  {
    echo "Mi trovo in $this→name_of_continent, $this→name_of_country, $th
is→name_of_region \n";
  }

  public function sayGoodBye()
  {
    echo "Addio \n";
  }
}



$output = new Region('Europa', 'Italia', 'Puglia');

$output→geoLoaction();

$output→sayHello();
```

Visibilità - Access modifiers

- Public: accessibile sia all'interno che all'esterno della classe;
- Protected: accessibile all'interno della classe e da tutte le classi che estendono la classe(ereditarietà)
- Private: accessibile soltanto all'interno della classe

PUBLIC

```


class Person
{


  public $firstname;
  public function __construct($firstname)
  {
    $this→firstname = $firstname;
  }
}

class Student extends Person
{

  public $lastname;



  public function __construct($firstname, $lastname)
  {
    parent::__construct($firstname);
    $this→lastname = $lastname;
  }
}

$francesco = new Student('Francesco', 'Mansi');

$francesco→firstname = 'Ciccio';//Scrittura
//Oppure
print_r($francesco→firstname);//Lettura
```

Protected

Non posso accedere all'esterno della classe e delle estensioni, ne in lettura e nemmeno in scrittura. Come ovviare? Creando dei metodi dentro la classe Student

```
class Person
{

  protected $firstname;



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

class Student extends Person
{

  public $lastname;



  public function __construct($firstname, $lastname)
  {
    parent::__construct($firstname);
    $this->lastname = $lastname;
  }
  public function getFirstName()
  {
    echo "$this->firstname\n";
  }

  public function setFirstName($newFirstName)
  {

    $this->firstname = $newFirstName;
    var_dump($this);
    echo "$this->firstname\n";
  }
}

$francesco = new Student('Francesco', 'Mansi');

$francesco->getFirstName();
$francesco->setFirstName('Carlo');
```

Private

La visibilità Private rende possibile la modifica solo nella classe stessa.

```
<?php
class Person
{

  private $firstname;



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

  public function setFirstName($newFirstName)
  {

    $this->firstname = $newFirstName;
    var_dump($this);
    echo "$this->firstname\n";
  }
}

class Student extends Person
{

  public $lastname;



  public function __construct($firstname, $lastname)
  {
    parent::__construct($firstname);
    $this->lastname = $lastname;
  }
  public function getFirstName()
  {
    echo "$this->firstname\n";
  }
}
```

```
$francesco = new Student('Francesco', 'Mansi');

$francesco→getFirstName();
$francesco→setFirstName('Carlo');
```