Classi, metodi e proprietà

Iniziamo subito a stampare Hello World con la programmazione ad oggetti.

```php
<?php

 class HelloWorld
{
        public $saluto;

                public function __construct()
                {
                        $this->saluto = "Hello World";
                }

                public function Saluta()
                {
                        echo $this->saluto;
                }
}

 $obj = new HelloWorld();

 $obj->Saluta();

 ?>
```

Avrei potuto fare in diversi altri modi, ho scelto questo in quanto è il più esplicativo visto che contiene tutti gli elementi principali. Vediamo quanto fatto passo dopo passo.

Le classi si dichiarano con la keyword “class” seguita dal nome della classe. In seguito andremo a dichiarare le proprietà. Nel nostro caso abbiamo un’unica proprietà ($saluto) con visibilità “public”. Esistono tre livelli di visibilità: public, protected e private. Tratterò approfonditamente il significato di questi attributi nel capitolo relativo all’incapsulamento. Per il momento dichiareremo tutte le proprietà e tutti i metodi come public.

I metodi si dichiarano con la keyword “function” e si comportano proprio come una funzione. Possiamo passare degli argomenti, degli argomenti facoltativi e possiamo farci ritornare un valore con “return”.

Il primo metodo è un metodo speciale e lo dichiariamo con il nome riservato __construct. Si tratta del metodo costruttore la sua particolarità sta nel fatto che viene eseguito automaticamente quando la classe viene istanziata (cosa significa istanziare la classe lo vedremo tra poco). Potremmo quindi dire che il costruttore è una sorta di inizializzazione. Nel nostro caso valorizziamo semplicemente la proprietà $saluto.
Per accedere ai metodi e alle proprietà della classe, dobbiamo utilizzare la variabile magica $this seguita da ->.
In seguito dichiariamo il metodo Saluta() che semplicemente ci stamperà la proprietà che abbiamo valorizzato.

A questo punto la nostra classe va attivata o come si dice in gergo istanziata; per essere utilizzata infatti, dobbiamo creare un'istanza, nel nostro caso $obj. La sintassi come vedete è $obj = new NomeClasse.
Ora grazie alla nostra istanza possiamo accedere ai metodi ed alle proprietà della classe.
Potremmo stampare Hello World anche eliminando

$obj->Saluta();

e sostituendolo con

echo $obj->saluto;

In questo caso invece che eseguire il metodo Saluta(), abbiamo stampato la proprietà saluto.

Possiamo anche valorizzare le proprietà da fuori la classe. Se ad esempio sostituite le ultime righe del codice in questo modo

```
$obj = new HelloWorld();
$obj->saluto = "ciao mondo";
$obj->Saluta();
```

In questo caso il costruttore valorizza saluto con Hello World, ma subito dopo rivalorizziamo saluto con ciao mondo. Al momento di eseguire il metodo Saluta() la proprietà saluto è valorizzata con ciao mondo ed è quello che stamperà.

Facciamo un altro esempio. La nostra classe ora deve stampare Hello World o Ciao Mondo a dipendenza del parametro che passiamo (it o en).

<?php

```php
class HelloWorld
{
	public $saluto;

		public function __construct($lang)
		{
			switch($lang)
			{
				case "en":
				$this->saluto = "Hello World";
				break;

				case "it":
				$this->saluto = "Ciao mondo";
				break;
			}

			$this->Saluta();
		}

		public function Saluta()
		{
			echo $this->saluto;
		}
}

$obj = new HelloWorld("en");

?>
```

Come vedete passo al costruttore l'argomento $lang. Con switch verifico: se è "en" valorizzo la proprietà con Hello World mentre se è "it" con Ciao Mondo. Infine, sempre nel costruttore, eseguo il metodo Saluta() direttamente. Questo vuole dire che semplicemente istanziano la classe (e naturalmente passando l'argomento en o it) mi verrà stampato il saluto.