Examples

This distribution of the Mono PHP Compiler includes two examples:
example (1) demonstrates the compilation and execution of a PHP script on its own,
example (2) additionally demonstrates how a compiled PHP script can be used from a class written in another .NET language.

Example (1)

In the file samples\Geometry.php you'll find a PHP script defining some geometric classes to show the usage of some basic features that PHP offers:

```
<?php

class Shape {
	const pi = 3.14159265;
}

class Circle extends Shape {
	public $radius;
	public function __construct($radius) {
		$this->radius = $radius;
	}
	public function GetArea() {
		return 0.5 * Shape::pi * $this->radius * $this->radius;
	}
	public function GetRound() {
		return 2 * Shape::pi * $this->radius;
	}
}

class Rectangle extends Shape {
	public $x;
	public $y;
	public function __construct($x, $y) {
		$this->x = $x;
		$this->y = $y;
	}
	public function GetArea() {
		return $this->x * $this->y;
	}
	public function GetRound() {
		return 2 * $this->x + 2 * $this->y;
	}
}
```

You can for example see some of PHP's object orientated features, so how classes, functions and constructors are defined, how classes inherit other classes and how class constants are defined.

In the next few statements we're playing a little with some objects. Like that you can see how objects are created, how their methods are invoked and how their members are accessed.

```
$c = new Circle(2);
echo '$c is a circle with radius 2\n';
echo 'area of $c = ' . $c->GetArea() . '\n';
echo 'round of $c = ' . $c->GetRound() . '\n\n';

$r = new Rectangle(2, 4);
echo '$r is a rectangle with lateral leghts ' . $r->x . ' and ' . $r->y . '\n';
echo 'area of $r = ' . $r->GetArea() . '\n';
echo 'round of $r = ' . $r->GetRound() . '\n\n';

echo '$c is ' . (!($c instanceof Shape) ? 'not ' : '') . 'a Shape' . '\n';
echo '$c is ' . (!($c instanceof Circle) ? 'not ' : '') . 'a Circle' . '\n';
echo '$c is ' . (!($c instanceof Rectangle) ? 'not ' : '') . 'a Rectangle' . '\n';
echo '$r is ' . (!($r instanceof Shape) ? 'not ' : '') . 'a Shape' . '\n';
echo '$r is ' . (!($r instanceof Circle) ? 'not ' : '') . 'a Circle' . '\n';
echo '$r is ' . (!($r instanceof Rectangle) ? 'not ' : '') . 'a Rectangle' . '\n';

?>
```

To compile the script, call

```
mono mPHP.exe samples\Geometry.php
```

which will produce an executable file called `Geometry.exe`.
To run it, call

```
mono Geometry.exe
```

which will output the result as expected:

```
$c is a circle with radius 2
area of $c = 6.2831853
round of $c = 12.5663706

$r is a rectangle with lateral leghts 2 and 4
area of $r = 8
round of $r = 12

$c is a Shape
$c is a Circle
$c is not a Rectangle
$r is a Shape
$r is not a Circle
$r is a Rectangle
```

Example (2)

In the file samples\MathPHP.php you'll find a PHP script defining two functions, one for calculating faculty numbers, the other one for fibonacci numbers:

```php
<?php
class MathPHP {
	public static function Fib($a) {
		if ($a == 0)
			return 0;
		else if ($a == 1)
			return 1;
		else
			return fib($a - 1) + fib($a - 2);
	}
	public static function Fac($a) {
		if ($a == 0 || $a == 1)
			return 1;
		else
			return $a * fac($a - 1);
	}
}
?>
```

Now we don't want to invoke them from inside the PHP script as in (1), but from a class written in another .NET language. For this purpose we'll compile the class with the /target:library option:

```
mono mPHP.exe /t:library samples\MathPHP.php
```

which will produce an assembly file called `MathPHP.dll`.
Now have a look into the file samples\MathCS.cs. You'll find a C# class in the file MathCS.cs using the functionality of the compiled PHP script:

```
public class MathCS {
	public static void Main(string[] args) {
		System.Console.WriteLine(
		"The following calculations are performed by the PHP class:");
		// define the number 5 as an object of type PHP.Integer
		PHP.Integer i = new PHP.Integer(5);
		// calculating faculty of 5
		PHP.Mixed fac5 = MathPHP.Fac(i);
		System.Console.WriteLine("Faculty of 5 = " + fac5.ToString());
		// calculating fibonacci of 5
		PHP.Mixed fib5 = MathPHP.Fib(i);
		System.Console.WriteLine("Fibonacci of 5 = " + fib5.ToString());
	}
}
```

We'll now compile this class. As it uses features of the compiled PHP script, it's important to refer to the assembly we just created and to the mPHPRuntime:

```
mcs /r:MathPHP.dll,mPHPRuntime.dll /out:MathCS.exe samples\MathCS.cs
```

This will produce the executable file `MathCS.exe`. Calling `mono MathCS.exe` will output the result as expected:

```
The following calculations are performed by the PHP class:
Faculty of 5 = 120
Fibonaccy of 5 = 5
```