中山大學
SUN YAT-SEN UNIVERSITY

Lecture 11
Advanced PHP and Web Frameworks

SE-805 Web 2.0 Programming (supported by Google)

http://my.ss.sysu.edu.cn/courses/web2.0/

School of Software, Sun Yat-sen University

Outline

- **Object-Oriented PHP**
- Web Frameworks

Why OOP?

- PHP is a primarily procedural language
- Small programs are easily written without adding any classes or objects
- Larger programs, however, become cluttered with so many disorganized functions
- Grouping *related data and behavior* into objects helps manage size and complexity
- And the most important of all, if you are not able to code in OOP, then you might be like this poor guy …….

Constructing and Using Objects

```
# construct an object
$name = new ClassName(parameters);

# access an object's field (if the field is public)
$name->fieldName

# call an object's method
$name->methodName(parameters);
                                                              PHP
```

```
$zip = new ZipArchive();
$zip->open("moviefiles.zip");
$zip->extractTo("images/");
$zip->close();
                                                              PHP
```

- The above code unzips a file
- Test whether a class is installed with class_exists

Object Example: Fetch File from Web

```php
# create an HTTP request to fetch student.php
$req = new HttpRequest("student.php", HttpRequest::METH_GET);
$params = array("first_name" => $fname, "last_name" => $lname);
$req->addPostFields($params);

# send request and examine result
$req->send();
$http_result_code = $req->getResponseCode();   # 200 means OK
print "$http_result_code\n";
print $req->getResponseBody();
```
PHP

- PHP's HttpRequest object can fetch a document from the web

Class Declaration Syntax

```php
class ClassName {
  # fields - data inside each object
  public $name;     # public field
  private $name;    # private field

  # constructor - initializes each object's state
  public function __construct(parameters) {
    statement(s);
  }

  # method - behavior of each object
  public function name(parameters) {
    statements;
  }
}
```
PHP

- Inside a constructor or method, refer to the current object as **$this**

Class Example

```php
<?php
class Point {
  public $x;
  public $y;

  # equivalent of a Java constructor
  public function __construct($x, $y) {
    $this->x = $x;
    $this->y = $y;
  }

  public function distance($p) {
    $dx = $this->x - $p->x;
    $dy = $this->y - $p->y;
    return sqrt($dx * $dx + $dy * $dy);
  }

  # equivalent of Java's toString method
  public function __toString() {
    return "(" . $this->x . ", " . $this->y . ")";
  }
}
?>
```

PHP

Class Usage Example

```
<?php
# this code could go into a file named use_point.php
include("Point.php");

$p1 = new Point(0, 0);
$p2 = new Point(4, 3);
print "Distance between $p1 and $p2 is " . $p1->distance($p2) . "\n\n";

var_dump($p2);    # var_dump prints detailed state of an object
?>
```

PHP

```
Distance between (0, 0) and (4, 3) is 5

object(Point)[2]
  public 'x' => int 4
  public 'y' => int 3
```

PHP

- $p1 and $p2 are references to Point objects

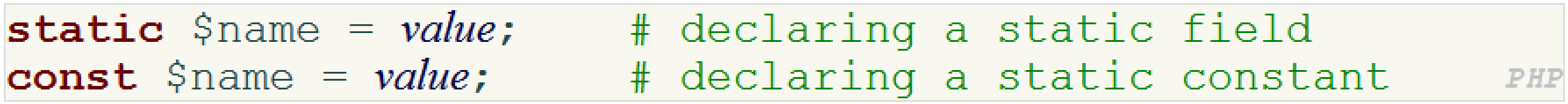
Basic inheritance

```
class ClassName extends ClassName {
  ...
}                                                    PHP
```

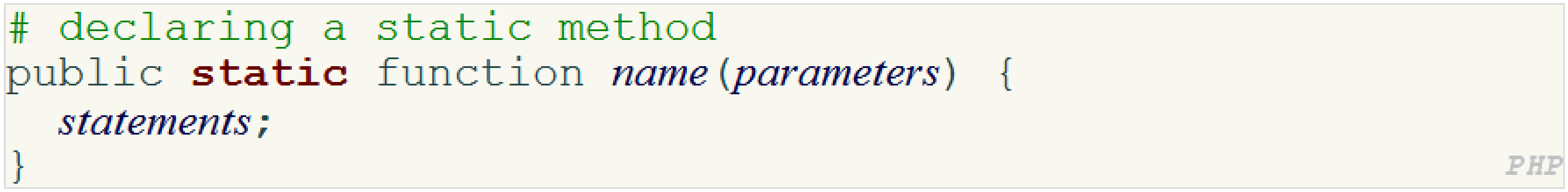
```
class Point3D extends Point {
  public $z;

  public function __construct($x, $y, $z) {
    parent::__construct($x, $y);
    $this->z = $z;
  }

  ...
}                                                    PHP
```

- The given class will inherit all data and behavior from *ClassName*

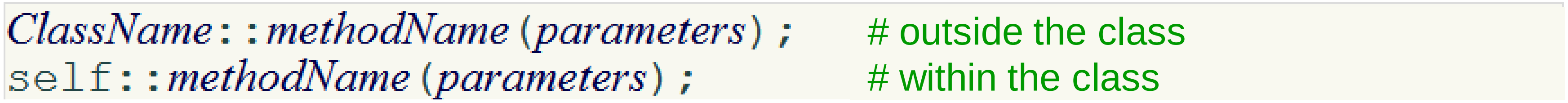
Static Methods, Fields, and Constants



```
static $name = value;      # declaring a static field
const $name = value;       # declaring a static constant      PHP
```



```
# declaring a static method
public static function name(parameters) {
  statements;
}                                                              PHP
```



```
ClassName::methodName(parameters);      # outside the class
self::methodName(parameters);           # within the class
```

- static fields/methods are **shared** throughout a class rather than replicated in every object

Abstract Classes and Interfaces

```
interface InterfaceName {
  public function name(parameters);
  public function name(parameters);
  ...
}

class ClassName implements InterfaceName { ...                PHP
```

```
abstract class ClassName {
  abstract public function name(parameters);
  ...
}                                                             PHP
```

- interfaces are supertypes that specify method headers without implementations
  - Cannot be instantiated; cannot contain function bodies or fields
  - Enables polymorphism between subtypes without sharing implementation code
- abstract classes are like interfaces, but you can specify fields, constructors, methods
  - Also cannot be instantiated; enables polymorphism with sharing of implementation code

Outline

- Object-Oriented PHP
- **Web Frameworks**

Framework vs. Library / Toolkit

- A **software framework**, in computer programming, is an abstraction in which common code providing generic functionality can be selectively overridden or specialized by user code providing specific functionality
- A Library is a collection of useful material for common use, and it is defined as a collection of related software constructs such as classes, functions and subroutines used to develop software.
- Toolkit is an alias of Library in software engineering.
- **A framework calls your code, but a library is called by your code.**

Common Problems in Dev. a Web App.

- Data validation and converting
- URL mapping
- Configurations (database connections, threads, timeout …)
- Session Management
- Page templates
- Database access and mapping
- User authentication
- Web services
- Ajax
- Caching

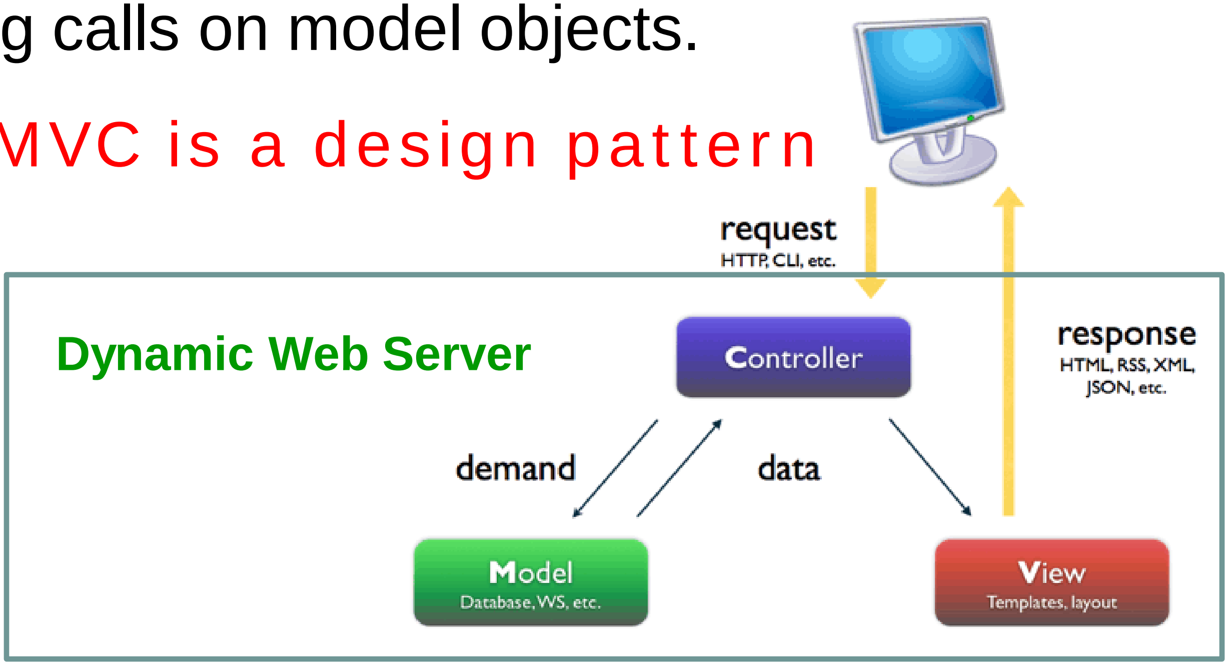
Web Frameworks

- A **web application framework** is a software framework that is designed to support the development of dynamic websites, Web applications and Web services. The framework aims to alleviate the overhead associated with common activities performed in Web development. For example, many frameworks provide libraries for database access, templating frameworks and session management, and they often promote code reuse.
- Almost all server-side languages have their other web frameworks
    - Spring Framework, Apache Struts, ASP.NET AJAX, Symfony, CakePHP, Zend Framework, Catalyst, Django, Ruby on Rails, ……

MVC

- The **Model** is the domain-specific representation of the data upon which the application operates.
- The **View** renders the model into a form suitable for interaction, typically a user interface element.
- The **Controller** receives input and initiates a response by making calls on model objects.

The MVC is a design pattern



Summary

- Object-Oriented PHP
  - Object, class
  - Inheritance
  - Static method, field, and constant
  - Abstract class, interface
- Web Frameworks
  - Framework vs. library / toolkit
  - Common problems in development of a Web application
  - Frameworks
  - MVC

Exercises

- Write a php script defining classes of Employee, Manager, and Secretary
  - Each Employee has a name and a salary
  - Each Manager is an Employee, and manages a group of other Employees
  - Each Secretary is an Employee, and works for a Manager
- add methods to these classes
  - Each Employee has a show() method returns her name and salary as a string
  - Each Manager has a getInferiors() method returns her inferiors
  - Each Secretary has a getSuperior() method returns her boss

Further Readings

- OO PHP for Beginners http://www.killerphp.com/tutorials/object-oriented-php/
- Object-Oriented PHP http://objectorientedphp.com/
- Zend http://www.zend.com/en/
- PHP MVC tutorial http://www.phpro.org/tutorials/Model-View-Controller-MVC.html
- TOP 10 PHP MVC frameworks http://www.mustap.com/phpzone_post_73_top-10-php-mvc-frameworks

Thank you!