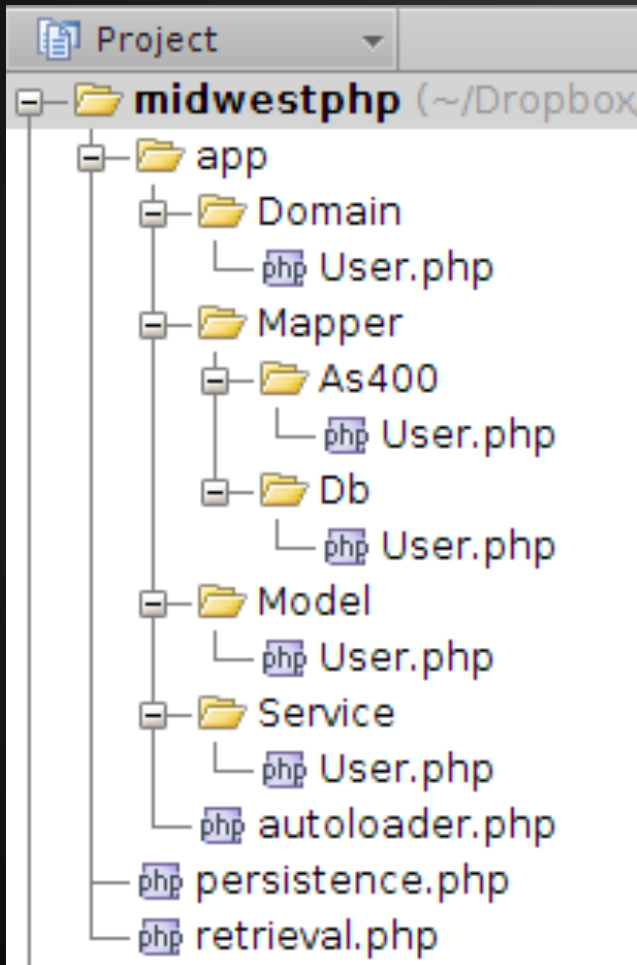


Example Code

- Let's see this in action. Things to remember:
- This is not a whole app - don't treat it as one
- It is example code
- This is not a whole app
- Don't forget, this is not a completed app

**Just trying to demonstrate the concepts
with the least amount of distraction!**

Layout - Overview



Main layout of files

We will cover each code section

Retrieval Options (A Controller)

```
php retrieval.php x
1  <?php
2  require 'app/autoloader.php';
3
4  $service = new Service_User();
5
6  //one id 3
7  $user = $service->findOneById(3);
8
9  // find only 5 first users
10 $users = $service->findAll(array('limit'=>5));
11
12 //users created before today's date
13 $legacyUsers = $service->findAllCreatedBeforeToday();
```

Retrieve one by ID

```
php User.php x
\Service User
1 <?php
2 class Service_User
3 {
4     public function findOneById($id)
5     {
6         $mapper = $this->_getMapper();
7         $params = array('id'=>$id);
8         return $mapper->findOne($params);
9     }
10
11     protected function _getMapper()
12     {
13         return new Mapper_Db_User();
14     }
15
16     //snippy
```

Other retrieval code

```
php User.php ×
\Service User
1  <?php
2  class Service_User
3  {
4      public function findAll($params = array())
5      {
6          $mapper = $this->_getMapper();
7          return $mapper->findAll($params);
8      }
9
10     public function findAllCreatedBeforeToday()
11     {
12         $mapper = $this->_getMapper();
13         $params = array('createdBefore'=>date('Y-m-d'));
14         return $mapper->findAll($params);
15     }
```

The Mapper

```
php User.php x
\Mapper Db User
1  <?php
2  class Mapper_Db_User
3  {
4      protected static $pdo = null;
5
6      protected function _getDao()
7      {
8          if (self::$pdo == null) {
9              // obviously remove credentials and don't use root
10             self::$pdo = new PDO('mysql:host=localhost;dbname=test', 'root', 'password');
11         }
12         return self::$pdo;
13     }
14
15     // and then more code ...
16
```

The Mapper (con't)

```
6 public function findAll($params = array())
7 {
8     $whereStrings = $whereParams = array();
9
10    if (isset($params['id'])) {
11        $whereStrings[] = 'id = ?';
12        $whereParams[] = $params['id'];
13    }
14    if (isset($params['firstname'])) {
15        $whereStrings[] = 'firstname = ?';
16        $whereParams[] = $params['firstname'];
17    }
18    if (isset($params['createdBefore'])) {
19        $whereStrings[] = 'datecreated < ?';
20        $whereParams[] = $params['createdBefore'];
21    }
22
23    $sql = "select * from user";
24    if (!empty($whereStrings)) {
25        $sql .= " where " . implode(' AND ', $whereStrings);
26    }
27    if (isset($params['limit'])) {
28        $sql .= " limit " . intval($params['limit']);
29    }
30
31    $statement = $this->_getDao()->prepare($sql);
32    $statement->execute($whereParams);
33
34    $results = $statement->fetchAll();
35
36    return $this->_populateCollection($results);
37 }
```


More Mapper

```
39 public function findOne($params = array())
40 {
41     $collection = $this->findAll($params);
42
43     if (count($collection) > 1) {
44         throw new Exception("More than one result found");
45     }
46
47     $returnOne = null;
48     if (!empty($collection)) {
49         $returnOne = array_shift($collection);
50     }
51
52     return $returnOne;
53 }
```

Mapper (populating / mapping)

```
70 protected function _populateCollection($results)
71 {
72     $return = array();
73
74     foreach ($results as $result) {
75         $return[] = $this->mapFromArray($result);
76     }
77
78     return $return;
79 }
80
81 public function mapFromArray($array, Model_User $user = null)
82 {
83     if (is_null($user)) $user = new Model_User();
84     if (isset($array['firstname'])) $user->firstname = $array['firstname'];
85     if (isset($array['lastname'])) $user->lastname = $array['lastname'];
86     if (isset($array['email'])) $user->email = $array['email'];
87     if (isset($array['id'])) $user->id = $array['id'];
88     if (isset($array['datecreated'])) $user->datecreated = $array['datecreated'];
89     return $user;
90 }
```

Data Model

```
php User.php x
1  <?php
2  Class Model_User
3  {
4      public $firstname;
5      public $lastname;
6      public $email;
7      public $id;
8      public $datecreated;
9
10     public function getFullName()
11     {
12         return trim($this->firstname . ' ' . $this->lastname);
13     }
14 }
```

Next Up...

Retrieval is done...

Let's look at persistence

Persistence Controller (1 of 2)

```
php persistence.php x
1  <?php
2  require 'app/autoloader.php';
3
4  $service = new Service_User();
5
6  // save new user
7  $inputFromUser = array('firstname'=>'aaron', 'lastname'=>'saray', 'email'=>'me@me.com');
8  try {
9      $user = $service->createFromUserInput($inputFromUser);
10     $service->save($user);
11 }
12 catch (Exception $e) {
13     print $e->getMessage();
14 }
```

Persistence Controller (2 of 2)

```
16 // save new user, invalid data
17 $inputFromUser = array('firstname'=>'aaron', 'lastname'=>'', 'email'=>'me@me.com');
18 try {
19     $user = $service->createFromUserInput($inputFromUser);
20     $service->save($user);
21 }
22 catch (Exception $e) {
23     // would go to exception, deal with it how you'd like
24     print $e->getMessage();
25 }
26
27 //update existing user with data
28 $user = $service->findOneById(1);
29 $inputFromUser = array('firstname'=>'Joe');
30 try {
31     $service->applyUpdateFromUserInput($user, $inputFromUser);
32     $service->save($user);
33 }
34 catch (Exception $e) {
35     print $e->getMessage();
36 }
```

Creating a User

```
php User.php x
\Service User
1  <?php
2  class Service_User
3  {
4      public function createFromUserInput($params = array())
5      {
6          $domainValidation = $this->_getDomainValidation();
7
8          if (!$domainValidation->validateFirstname($params)) {
9              throw new Exception("First name did not validate");
10         }
11         if (!$domainValidation->validateLastname($params)) {
12             throw new Exception("Last name did not validate");
13         }
14
15         $mapper = $this->_getMapper();
16         $user = $mapper->mapFromArray($params);
17
18         return $user;
19     }
20
21     // more code below
22
```

Creating User (con't)

```
58     protected function _getDomainValidation()
59     {
60         return new Domain_User();
61     }
62
63     public function save(Model_User $user)
64     {
65         $this->_getMapper()->save($user);
66         $this->_audit("User has been saved");
67     }
68
69     protected function _audit($text)
70     {
71         // can log text here
72     }
73
74     protected function _getMapper()
75     {
76         return new Mapper_Db_User();
77     }
```


Domain User Model

```
php User.php x
1  <?php
2  class Domain_User
3  {
4      public function validateFirstname($values)
5      {
6          return !empty($values['firstname']) && strlen($values['firstname']) < 25;
7      }
8
9      public function validateLastname($values)
10     {
11         return !empty($values['lastname']) && strlen($values['lastname']) < 45;
12     }
13 }
```

Mapper DB - Save

```
1  <?php
2  class Mapper_Db_User
3  {
4      public function save(Model_User $user)
5      {
6          if ($user->id) {
7              $sql = "update user set firstname = ?, lastname = ?, email = ? where id = ?";
8              $params = array($user->firstname, $user->lastname, $user->email, $user->id);
9          }
10         else {
11             $sql = "insert into user (firstname, lastname, email) values (?, ?, ?)";
12             $params = array($user->firstname, $user->lastname, $user->email);
13         }
14
15         $statement = $this->_getDao()->prepare($sql);
16         $statement->execute($params);
17     }

```

Apply Update From User

```
41 public function applyUpdateFromUserInput(Model_User $user, $params = array())
42 {
43     $domainValidation = $this->_getDomainValidation();
44
45     if (isset($params['firstname']) && !$domainValidation->validateFirstname($params)) {
46         throw new Exception("First name did not validate");
47     }
48     if (isset($params['lastname']) && !$domainValidation->validateLastname($params)) {
49         throw new Exception("Last name did not validate");
50     }
51
52     $mapper = $this->_getMapper();
53     $user = $mapper->mapFromArray($params, $user);
54
55     return $user;
56 }
```