

# Persistence Done

Yay

Now - let's introduce the value!

# Users must be sync'd to AS400

- Previously was from mysql db
- Need to change as little code as possible
- The only thing changing is how we map data
  - to / from the data source
  - "map"... mapper
- Lets substitute in a different mapper

# Modify User Service

```
64     protected function _getMapper()  
65     {  
66         //return new Mapper_Db_User();  
67         return new Mapper_As400_User();  
68     }
```

This is the **ONLY** change to your code.

All else is **NEW** code.

# What does this mapper look like?

```
php User.php x
\Mapper As400 User
1 <?php
2 class Mapper_As400_User
3 {
4     protected static $pdo = null;
5
6     public function findAll($params = array())
7     {
8         $in = array();
9
10        $id = isset($params['id']) ? $params['id'] : 0;
11        $in['id'] = str_pad($id, 10, '0', STR_PAD_LEFT);
12
13        $firstname = isset($params['firstname']) ? $params['firstname'] : '';
14        $in['firstname'] = $firstname;
15
16        if (isset($params['createdBefore'])) {
17            $createdBefore = date('Ymd', strtotime($params['createdBefore']));
18        }
19        else {
20            $createdBefore = '00000000';
21        }
22        $in['createdBefore'] = $createdBefore;
23
24        $out = array(
25            'id'=>'',
26            'firstname'=>'',
27            'lastname'=>'',
28            'email'=>'',
29            'id'=>'',
30            'datecreated'=>''
31        );

```

# AS400 Mapper Continued

```
33     $query = "CALL MYLIB.GETUSER(?, ?, ?, ?, ?, ?, ?, ?, ?)";
34     $stmt = $this->_getDao()->prepare($query);
35     $stmt->bindParam(1, $in['id']);
36     $stmt->bindParam(2, $in['firstname']);
37     $stmt->bindParam(3, $in['createdBefore']);
38     $stmt->bindParam(4, $out['id'], PDO::PARAM_STR|PDO::PARAM_INPUT_OUTPUT);
39     $stmt->bindParam(5, $out['firstname'], PDO::PARAM_STR|PDO::PARAM_INPUT_OUTPUT);
40     $stmt->bindParam(6, $out['lastname'], PDO::PARAM_STR|PDO::PARAM_INPUT_OUTPUT);
41     $stmt->bindParam(7, $out['email'], PDO::PARAM_STR|PDO::PARAM_INPUT_OUTPUT);
42     $stmt->bindParam(8, $out['id'], PDO::PARAM_STR|PDO::PARAM_INPUT_OUTPUT);
43     $stmt->bindParam(9, $out['datecreated'], PDO::PARAM_STR|PDO::PARAM_INPUT_OUTPUT);
44
45     $results = array();
46
47     do {
48         $stmt->execute();
49         $results[] = $out;
50     } while ($stmt->nextRowset());
51
52     return $this->_populateCollection($results);
53 }
54
55 public function findOne($params = array())
56 {
57     $collection = $this->findAll($params);
58
59     if (count($collection) > 1) {
60         throw new Exception("More than one result found");
61     }
62
63     $returnOne = null;
64     if (!empty($collection)) {
65         $returnOne = array_shift($collection);
66     }
67
68     return $returnOne;
69 }
```

# AS400 Mapper Continued

```
71 public function mapFromArray($array, Model_User $user = null)
72 {
73     if (is_null($user)) $user = new Model_User();
74     $user->firstname = trim($array['firstname']);
75     $user->lastname = trim($array['lastname']);
76     $user->email = trim($array['email']);
77     $user->id = intval($array['id']);
78     $user->datecreated = date('Y-m-d', strtotime($array['datecreated']));
79     $return[] = $user;
80
81     return $user;
82 }
```

# AS400 Mapper Continued

```
84     protected function _populateCollection($results)
85     {
86         $return = array();
87
88         foreach ($results as $result) {
89             $return[] = $this->mapFromArray($result);
90         }
91
92         return $return;
93     }
94
95     protected function _getDao()
96     {
97         if (self::$pdo == null) {
98             self::$pdo = new PDO('odbc:iSeriesDSN', 'MYUSER', 'MYPASS');
99         }
100        return self::$pdo;
101    }
102 }
```

**Whew! Done!**

Check it out at

<http://aaronsaray.com/midwestphp>

Tweet at @aaronsaray