# Mouse Object Construction Options

The following examples take a Mouse class and produce a number of views reflecting standard Create, Read, Update, and Delete (CRUD) functionality.

Generally, Mouse objects can be constructed in one of two ways. The first way is to create a separate Mouse Object for each of the rendering types. The second way is with a single class representing the complete Mouse Object which uses “modify_” Reflection methods to alter the Object based on the type of rendering (eg. list, editor, or viewer) being performed. The first way is cleaner with regard to code clutter; the second, with regard to file clutter.

The first method is preferred as it is easier to read and manage, but the second method could provide a smaller memory footprint and faster memory access should the developer decide to keep the PHP files resident in the web server or compiled to C byte code.

- Log in [1] to post comments

# Multiple Mouse Classes (First Method)

The preferred method requires multiple files – one for each Page class – but the Mouse Object constructor contains only those Attributes and Attribute Options necessary for that Page class. As a result, the complete Message Mouse Object for only the Viewer Page file would look like:

```
class MessageDB extends DB {
  function __construct() {
  parent::__construct();
  $g_user_object = lms_get_user_object();

  $this->has(array(
  message_id => array(isa => 'int',
  init => $_REQUEST["message_id"],
  join => array(where => "lms_message.message_id = ? AND " .
  '(lms_message.from_user_id = ' . $g_user_object->user_id .
  ' OR lms_message.to_user_id = ' . $g_user_object->user_id . ')')),
  subject => array(isa => 'Str32'),
  message => array(isa => 'Text',
  name => "Message",
  row => 2,
  col => 1),
  from_user_id => array(name => 'From',
  isa => 'Username',
  join => array(table => 'lms_user lu_from',
  column => 'lu_from.username',
  where => 'lu_from.user_id = lms_message.from_user_id'),
  row => 1,
  col => 1),
  to_user_id => array(name => 'To',
  isa => 'Username',
  join => array(table => 'lms_user lu_to',
  column => 'lu_to.username',
  where => 'lu_to.user_id = lms_message.to_user_id'),
  row => 1,
```

```
  col => 2),
  sent_date_time => array(name => 'Time Sent',
  isa => 'DateTime',
  order => 'desc',
  row => 3,
  col => 1),
  read_date_time => array(isa => 'DateTime',
  name => 'Time Read',
  row => 3,
  col => 2,
  join => array(column => "IF(lms_message.read_date_time=” .
  “0,NOW(),lms_message.read_date_time)"))
  ));
  }

  function update_message_read_time() {
  try {
  $sql = "UPDATE lms_message SET read_date_time = NOW() where message_id = ? and
read_date_time in (0,null)";
  $bind_vars = array(array('value' => $this->get_message_id(), 'type' => 'i'));

  if ($stmt = $this->db_prepare($sql, $bind_vars)) {
  $stmt->execute();
  }
  } catch (Exception $e) {
  print($e->getMessage());
  }
  }
```

- Log in [2] to post comments

# Editor Mouse Class Example

```
class MessageDB extends DB {
  function __construct() {
  parent::__construct();
  global $g_user_object;

  $options = $this->get_usernames();

  $this->has(array(
  subject => array(isa => 'Str32',
  name => "Subject",
  row => 1,
  col => 1,
  required => 1,
  control => function($args){
  $args["size"] = "40";
  $args["maxlength"] = "32";
  render_input($args);}),
  message => array(isa => 'Text',
  name => "Message",
```

```
  row => 3,
  col => 1,
  required => 1,
  control => function($args) {
  $args["rows"] = "10";
  $args["cols"] = "60";
  render_textarea($args, $args['value']);}),
  from_user_id => array(isa => 'int',
  required => 1,
  init => $g_user_object->get_user_id()),
  to_user_id => array(name => 'To',
  isa => 'int',
  row => 2,
  col => 1,
  required => 1,
  control => function($args) use ($options) {
  render_multiselect("to_user_id[]", $_POST['to_user_id'], $options, "5");
  }));
  }

  function get_usernames() {
  $options = array();

  $sql = "SELECT user_id, username FROM lms_user ORDER BY username";
  if ($stmt = $this->db_prepare($sql, null)) {
  $stmt->execute();
  $result = $stmt->get_result();
  while ($row = $result->fetch_row()) {
  array_push($options, array('value' => $row[0], 'label' => $row[1]));
  }
  $stmt->close();
  }

  return($options);
  }
}
```

- Log in [3] to post comments

# Manager Mouse Class Example

```
class MessageDB extends DB {
  function __construct() {
  parent::__construct();
  $g_user_object = lms_get_user_object();

  $this->has(array(
  message_id => array(isa => 'int',
  join => array(where => '(lms_message.from_user_id = ' . $g_user_object->user_id . ' OR ' .
'lms_message.to_user_id = ' . $g_user_object->user_id . ')')),
  subject => array(name => 'Subject',
  isa => 'Str32',
```

```
  //read_date_time is in the 5th array position of this MOUSE object
  style => function($data) {
  if ($data[5] == "0000-00-00 00:00:00") { return("font-weight:bold;"); }},
  //message_id is in the 0th array position of this MOUSE object
  link => function($data) {
  return('default.php?appname=message-view&message_id=' . $data[0]);}),
  from_user_id => array(name => 'From',
  isa => 'Username',
  join => array(table => 'lms_user lu_from',
  column => 'lu_from.username',
  where => 'lu_from.user_id = lms_message.from_user_id')),
  to_user_id => array(name => 'To',
  isa => 'Username',
  join => array(table => 'lms_user lu_to',
  column => 'lu_to.username',
  where => 'lu_to.user_id = lms_message.to_user_id')),
  sent_date_time => array(name => 'Time Sent',
  isa => 'DateTime',
  order => 'desc'),
  read_date_time => array(isa => 'DateTime')
  ));
  }
}
```

- Log in [4] to post comments

# Single Mouse Class (Second Method)

As an example of the second method, the following Mouse Object occupies a single file, but has several methods which are called by the respective Page class in order to render the respective view. When the constructor is called, a basic Mouse Object is instantiated.

```
class MessageDB extends DB {
  function __construct() {
  parent::__construct();

  $this->has(array(
  message_id => array(isa => 'int'),
  subject => array(name => 'Subject',
  isa => 'Str32'),
  message => array(isa => 'Text'),
  from_user_id => array(name => 'From',
  isa => 'int'),
  to_user_id => array(name => 'To',
  isa => 'int'),
  sent_date_time => array(name => 'Time Sent',
  isa => 'DateTime',
  order => 'desc'),
  read_date_time => array(isa => 'DateTime')
  ));
  }
```

When the Viewer page class calls the set_view_options() method, the Mouse Object is modified to

reflect the needs of the Viewer page.

```
function set_view_options() {
  //modify the join option of the message_id attribute
  $message_id->join = array(where => "lms_message.message_id = " . $this->get_message_id());
  $this->modify_message_id($message_id);

  $subject->name = "";
  $this->modify_subject($subject);

  $message->row = 2;
  $message->col = 1;
  $message->name = "Message";
  $this->modify_message($message);

  $from_user_id->join = array(table => 'lms_user lu_from',
  column => 'lu_from.username',
  where => 'lu_from.user_id = lms_message.from_user_id');
  $from_user_id->row = 1;
  $from_user_id->col = 1;
  $this->modify_from_user_id($from_user_id);

  $to_user_id->join = array(table => 'lms_user lu_to',
  column => 'lu_to.username',
  where => 'lu_to.user_id = lms_message.to_user_id');
  $to_user_id->row = 1;
  $to_user_id->col = 2;
  $this->modify_to_user_id($to_user_id);

  $sent_date_time->row = 3;
  $sent_date_time->col = 1;
  $this->modify_sent_date_time($sent_date_time);

  $read_date_time->row = 3;
  $read_date_time->col = 2;
  $read_date_time->name = "Time Read";
  $this->modify_read_date_time($read_date_time);
}

function set_list_options() {
  // modify the Mouse Attribute "subject" to properly render the list
  //message_id is in the 0th array position of this MOUSE object
  $subject->link = function($data) {return('default.php?appname=message-view&message_id=' .
$data[0]);};
  $this->modify_subject($subject);

  // modify the Mouse user_id Attributes in order to render the user lists
  $from_user_id->join = array(table => 'lms_user lu_from',
  column => 'lu_from.username',
  where => 'lu_from.user_id = lms_message.from_user_id');
  $this->modify_from_user_id($from_user_id);

  $to_user_id->join = array(table => 'lms_user lu_to',
  column => 'lu_to.username',
  where => 'lu_to.user_id = lms_message.to_user_id');
  $this->modify_to_user_id($to_user_id);
}
```

- Log in [5] to post comments

**Source URL:** http://www.blackhillsystems.com/?q=node/24

**Links**
[1] http://www.blackhillsystems.com/?q=user/login&destination=node/24%23comment-form
[2] http://www.blackhillsystems.com/?q=user/login&destination=node/25%23comment-form
[3] http://www.blackhillsystems.com/?q=user/login&destination=node/26%23comment-form
[4] http://www.blackhillsystems.com/?q=user/login&destination=node/27%23comment-form
[5] http://www.blackhillsystems.com/?q=user/login&destination=node/28%23comment-form