Code review (maestro)

Coder found 1 projects, 67 files, 4 *critical* warnings, 515 *normal* warnings, 538 *minor* warnings, 0 warnings were flagged to be ignored

Use the Selection form to select options for this code review, or change the Default Settings and use the Default tab above.

Selection form

sites/all/modules/contrib/maestro/maestro.module

maestro.module

- Line 12: Missing period

```
* Implements hook_menu()
```

- Line 160: Missing period

```
* Implements hook_permission()
```

- Line 183: Missing period

```
* Implements hook_help()
```

- Line 190: String concatenation should be formatted with a space separating the operators (dot .) and the surrounding terms

```
$output = '<p>' . t("Define and administer workflows.") .'</p>';
```

- Line 210: Missing period

```
* Implements hook_theme()
```

- Line 233: missing space after comma

```
'variables' => array('task' => NULL,'source' => NULL),
```

- Line 237: missing space after comma

```
'variables' => array('rowid' => NULL,'tracking_id' => NULL),
```

- Line 375: Missing period

```
* Implements hook_action_info()
```

- Line 383: Functions should be called with no spaces between the function name and opening parentheses

```
'triggers' => array ('any'),
```

- Line 419: Control statements should have one space between the control keyword and opening parenthesis

```
if($nid>0) {
```

- Line 463: There should be no trailing spaces

```
*
```

- Line 492: Missing period

```
* Implements hook_trigger_info()
```

- Line 530: Control statements should have one space between the control keyword and opening parenthesis

```
if(variable_get('maestro_run_orchestrator_in_task_console', 1)){
```

- Line 530: use a space between the closing parenthesis and the open bracket

```
if(variable_get('maestro_run_orchestrator_in_task_console', 1)){
```

- Line 549: Functions should be called with no spaces between the function name and opening parentheses

```
$template = intval ($template);
```

- Line 555: Potential problem: drupal_set_message() only accepts filtered text, be sure to use check_plain(), filter_xss() or similar to ensure your $variable is fully sanit

```
drupal_set_message("New Process Code Success! - Process ID: $newprocess");
```

- Line 555: The $message argument to drupal_set_message() should be enclosed within t() so that it is translatable.

```
drupal_set_message("New Process Code Success! - Process ID: $newprocess");
```

- Line 556: else statements should begin on a new line

```
} else {
```

- Line 557: Potential problem: drupal_set_message() only accepts filtered text, be sure to use check_plain(), filter_xss() or similar to ensure your $variable is fully sanit

```
drupal_set_message("New Process Code FAIL! - Template: $template not defined");
```

- Line 557: The $message argument to drupal_set_message() should be enclosed within t() so that it is translatable.

```
drupal_set_message("New Process Code FAIL! - Template: $template not defined");
```

- Line 561: The $message argument to drupal_set_message() should be enclosed within t() so that it is translatable.

```
drupal_set_message("New Process Code FAIL! - No Template ID Given");
```

- Line 579: Functions should be called with no spaces between the function name and opening parentheses

```
$template = intval ($template);
```

- Line 585: Potential problem: drupal_set_message() only accepts filtered text, be sure to use check_plain(), filter_xss() or similar to ensure your $variable is fully sanit

```
drupal_set_message("New Process Code Success! - Process ID: $newprocess");
```

- Line 585: The $message argument to drupal_set_message() should be enclosed within t() so that it is translatable.

```
drupal_set_message("New Process Code Success! - Process ID: $newprocess");
```

- Line 586: else statements should begin on a new line

```
} else {
```

- Line 587: Potential problem: drupal_set_message() only accepts filtered text, be sure to use check_plain(), filter_xss() or similar to ensure your $variable is fully sanit

```
drupal_set_message("New Process Code FAIL! - Template: $template not defined");
```

- Line 587: The $message argument to drupal_set_message() should be enclosed within t() so that it is translatable.

```
drupal_set_message("New Process Code FAIL! - Template: $template not defined");
```

- Line 591: The $message argument to drupal_set_message() should be enclosed within t() so that it is translatable.

```
drupal_set_message("New Process Code FAIL! - No Template ID Given");
```

- Line 632: missing space after comma

```
$formatted_task->task_started = ', Started: ' . format_date($task->dates['started'],'short');
```

- Line 634: else statements should begin on a new line

```
} else {
```

- Line 638: missing space after comma

```
$formatted_task->assigned_shortdate = format_date($task->dates['created'],'custom','m/d/y');
```

- Line 639: missing space after comma

```
$formatted_task->assigned_longdate = format_date($task->dates['created'],'medium');
```

- Line 650: missing space after comma

```
$action_record = $maestro->engine()->showInteractiveTask($current_task,$task->queue_id);
```

- Line 653: else statements should begin on a new line

```
} else {
```

- Line 656: else statements should begin on a new line

```
} else {
```

- Line 678: Use an indent of 2 spaces, with no tabs

```
$variables['process_dropdown']=$options;
```

- Line 686: missing space after comma

```
global $base_url,$user;
```

- Line 701: else statements should begin on a new line

```
} else {
```

- Line 706: missing space after comma

```
$projectRec = db_select('maestro_projects','a')
```

- Line 707: missing space after comma

```
->fields('a',array('project_num','originator_uid','description','status','prev_status','related_processes'))
```

- Line 722: else statements should begin on a new line

```
} else {
```

- Line 730: missing space after comma

```
$query = db_select('maestro_process','process');
```

- Line 736: missing space after comma

```
$query->fields('queue', array('process_id','created_date','started_date','completed_date','status'));
```

- Line 737: missing space after comma

```
$query->fields('template', array('taskname','is_dynamic_taskname','dynamic_taskname_variable_id'));
```

- Line 738: missing space after comma

```
$query->fields('assignment', array('assign_id','assign_type','process_variable'));
```

- Line 739: missing space after comma

```
$query->addField('assignment','id','assign_recid');
```

- Line 740: missing space after comma

```
$query->addField('queue','id','queue_id');
```

- Line 741: missing space after comma

```
$query->addField('user','name','username');
```

- Line 742: missing space after comma

```
$query->addField('role','name','rolename');
```

- Line 743: missing space after comma

```
$query->addField('process','pid','parent_process_id');
```

- Line 744: missing space after comma

```
$query->addField('process','tracking_id','tracking_id');
```

- Line 746: missing space after comma

```
$query->condition('queue.status',0,'=');
```

- Line 747: missing space after comma

```
$query->condition('queue.show_in_detail',1,'=');
```

- Line 751: missing space after comma

```
  $otask->assigned_date = strftime("%b %d/%Y %H:%M",$record->created_date);
```

- Line 764: Control statements should have one space between the control keyword and opening parenthesis

```
      if($record->assign_id == 0 AND $record->process_variable > 0) {
```

- Line 769: else statements should begin on a new line

```
      } else {
```

- Line 777: Control statements should have one space between the control keyword and opening parenthesis

```
        if($record->assign_id == 0 AND $record->process_variable > 0) {
```

- Line 782: else statements should begin on a new line

```
        } else {
```

- Line 786: else statements should begin on a new line

```
      } else {
```

- Line 795: else statements should begin on a new line

```
  } else {
```

- Line 801: missing space after comma

```
$res = db_select('users','users')
```

- Line 802: missing space after comma

```
->fields('users', array('uid','name'))
```

- Line 811: missing space after comma

```
$query = db_select('maestro_process','process');
```

- Line 813: missing space after comma

```
$query->fields('queue', array('id','process_id','template_data_id','task_class_name'));
```

- Line 814: missing space after comma

```
$query->addField('queue','id','queue_id');
```

- Line 820: Control statements should have one space between the control keyword and opening parenthesis

```
  if(!in_array($taskrec->template_data_id,$uniqueTasks)) {
```

- Line 820: missing space after comma

```
  if(!in_array($taskrec->template_data_id,$uniqueTasks)) {
```

- Line 823: missing space after comma

```
    $taskContent = $current_task->showContentDetail($task->tracking_id,$taskrec->id);
```

- Line 843: missing space after comma

```
$query = db_select('maestro_project_comments','a');
```

- Line 844: missing space after comma

```
$query->fields('a',array('id','uid','timestamp','comment'));
```

- Line 845: missing space after comma

```
$query->join('users','b','b.uid = a.uid');
```

- Line 846: missing space after comma

```
$query->addField('b','name','username');
```

- Line 847: missing space after comma

```
$query->condition('a.tracking_id',$variables['tracking_id'],'=');
```

- Line 851: missing space after comma
```
    $query = db_select('maestro_queue','a');
```
- Line 852: missing space after comma
```
    $query->join('maestro_template_data','b','b.id = a.template_data_id');
```
- Line 853: missing space after comma
```
    $query->addField('b','taskname','taskname');
```
- Line 854: missing space after comma
```
    $query->condition('a.id',$record->task_id,'=');
```
- Line 856: else statements should begin on a new line
```
    } else {
```
- Line 859: missing space after comma
```
    $record->date = strftime("%b %d/%Y %H:%M",$record->timestamp);
```
- Line 862: else statements should begin on a new line
```
    } else {
```
- Line 886: missing space after comma
```
  $query->addExpression('COUNT(a.id)','templateCount');
```
- Line 924: missing space after comma
```
    $query->fields('a',array('template_name','app_group'));
```
- Line 925: missing space after comma
```
    $query->condition('a.id',$variables['tid'],'=');
```
- Line 967: Control statements should have one space between the control keyword and opening parenthesis
```
  if($variables['var_id'] == $variables['edit_var']) {
```
- Line 985: use a space between the closing parenthesis and the open bracket
```
function maestro_createAppGroupDropDown($name, $selected = 0){
```
- Line 991: Control statements should have one space between the control keyword and opening parenthesis
```
    if($selected == $rec->id) $sel = " selected ";
```
- Line 1000: There should be no trailing spaces
```
 *
```
- Line 1048: There should be no trailing spaces
```
 *   Keyed array of keyed arrays defining handler function and the function description
```
- Line 1061: indent secondary line of comment one space
```
* @desc Implements HOOK_maestro_set_process_variable_methods.
```
- Line 1061: Missing parenthesis after function name
```
* @desc Implements HOOK_maestro_set_process_variable_methods.
```
- Line 1062: indent secondary line of comment one space
```
*       The return must be an array structure as follows:
```
- Line 1063: indent secondary line of comment one space
```
*       array (
```
- Line 1064: indent secondary line of comment one space
```
*         'set_method_name' => array (    //a unique name for your set method
```
- Line 1065: indent secondary line of comment one space
```
*           'title' => t('Title'),        //the title which will show up when you edit the set process variable task
```
- Line 1066: indent secondary line of comment one space
```
*           'engine_handler' => 'maestro_set_process_variable_handler'    //the function that gets called which returns
```
- Line 1067: indent secondary line of comment one space
```
*         )                                                               //the value to set the process variable to.
```
- Line 1068: indent secondary line of comment one space
```
*       );
```
- Line 1069: indent secondary line of comment one space
```
*/
```
- Line 1072: Functions should be called with no spaces between the function name and opening parentheses
```
    'hardcoded_value' => array (
```
- Line 1076: Functions should be called with no spaces between the function name and opening parentheses

```
    'increment_value' => array (
```

- Line 1077: use
 instead of

```
      'title' => t('Add or Subtract a Value') . '<br>' . t('(negative number for subtraction)'),
```

- Line 1080: Functions should be called with no spaces between the function name and opening parentheses

```
    'maestro_content_type_task' => array (
```

- Line 1089: put a space between the asterisk and the comment text

```
 *@desc the following are functions that set_process_variable() calls to
```

- Line 1118: missing space after comma

```
  $query = db_select('maestro_project_content','content');
```

- Line 1119: missing space after comma

```
  $query->addField('content','nid');
```

- Line 1120: missing space after comma

```
  $query->condition('content.tracking_id',$tracking_id,'=');
```

- Line 1121: missing space after comma

```
  $query->condition('content.instance',1,'=');
```

- Line 1122: missing space after comma

```
  $query->condition('content.content_type',$content_type,'=');
```

- Line 1141: missing space after comma

```
  $node_entity = entity_load('node',array($nid));
```

- Line 1142: missing space after comma

```
  $data = field_view_field('node',$node_entity[$nid],$field_name);
```

- Line 1167: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
      ->condition('id', $_POST['taskid'], '=')
```

- Line 1173: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
    $tid = intval($_POST['templateid']);
```

- Line 1175: Functions should be called with no spaces between the function name and opening parentheses

```
    $retdata = array ('status' => "1", 'processid' => $pid);
```

- Line 1180: Functions should be called with no spaces between the function name and opening parentheses

```
    $retdata = array ('status' => FALSE);
```

- Line 1183: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
      $taskid = intval($_POST['taskid']);
```

- Line 1195: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
      if ($assigned_value > 0 OR $_POST['variable_id'] > 0) {
```

- Line 1197: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
        switch ($_POST['assignment_type']) {
```

- Line 1200: missing space after comma

```
          $maestro->engine()->reassignTask($taskid,$_POST['task_reassign_uid'],$_POST['taskuser'],$_POST['variable_id'],1);
```

- Line 1200: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
          $maestro->engine()->reassignTask($taskid,$_POST['task_reassign_uid'],$_POST['taskuser'],$_POST['variable_id'],1);
```

- Line 1202: else statements should begin on a new line

```
        } else {
```

- Line 1205: Functions should be called with no spaces between the function name and opening parentheses

```
        $retdata = array ('status' => TRUE, 'username' => $username);
```

- Line 1209: missing space after comma

```
        $maestro->engine()->reassignTask($taskid,$_POST['task_reassign_uid'],$_POST['taskuser'],$_POST['variable_id'],2);
```

- Line 1209: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
        $maestro->engine()->reassignTask($taskid,$_POST['task_reassign_uid'],$_POST['taskuser'],$_POST['variable_id'],2);
```

- Line 1211: Functions should be called with no spaces between the function name and opening parentheses

```
        $retdata = array ('status' => TRUE, 'username' => $username);
```

- Line 1215: missing space after comma

```
        $maestro->engine()->reassignTask($taskid,$_POST['task_reassign_uid'],$_POST['taskuser'],$_POST['variable_id'],3);
```

- Line 1215: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
        $maestro->engine()->reassignTask($taskid,$_POST['task_reassign_uid'],$_POST['taskuser'],$_POST['variable_id'],3);
```

- Line 1217: Functions should be called with no spaces between the function name and opening parentheses

```
            $retdata = array ('status' => TRUE, 'username' => $username);
```

- Line 1227: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
      $queue_id = intval($_POST['queueid']);
```

- Line 1231: missing space after comma

```
        ->fields('maestro_queue', array('task_class_name','process_id'))
```

- Line 1240: missing space after comma

```
        $ret = $task->processInteractiveTask($queue_id,$taskop);
```

- Line 1245: missing space after comma

```
            $maestro->engine()->completeTask($queue_id,$ret->status);
```

- Line 1247: else statements should begin on a new line

```
          } else {
```

- Line 1250: else statements should begin on a new line

```
        } else{
```

- Line 1253: else statements should begin on a new line

```
      } else {
```

- Line 1264: Functions should be called with no spaces between the function name and opening parentheses

```
      $retdata = array ('status' => "1");
```

- Line 1269: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
      $tracking_id = intval($_POST['tracking_id']);
```

- Line 1271: Control statements should have one space between the control keyword and opening parenthesis

```
      if($retval) {
```

- Line 1272: Functions should be called with no spaces between the function name and opening parentheses

```
        $retdata = array ('status' => "1", 'message' => t('Successful Deletion of Project'));
```

- Line 1275: Functions should be called with no spaces between the function name and opening parentheses

```
        $retdata = array ('status' => "0", 'message' => t('You are not an admin.  Project not deleted.'));
```

- Line 1283: missing space after comma

```
        ->fields(array('tracking_id','uid','task_id','timestamp','comment'))
```

- Line 1292: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
        $rowid = intval($_POST['rowid']);
```

- Line 1293: missing space after comma

```
        $html = theme('maestro_project_comments',array('rowid' => $rowid, 'tracking_id' => $_POST['tracking_id']));
```

- Line 1294: Functions should be called with no spaces between the function name and opening parentheses

```
        $retdata = array ('html' => $html,'status' => "1");
```

- Line 1294: missing space after comma

```
        $retdata = array ('html' => $html,'status' => "1");
```

- Line 1295: else statements should begin on a new line

```
      } else {
```

- Line 1296: Functions should be called with no spaces between the function name and opening parentheses

```
        $retdata = array ('status' => "1");
```

- Line 1303: missing space after comma

```
        $query = db_select('maestro_project_comments','a');
```

- Line 1304: missing space after comma

```
        $query->addField('a','uid');
```

- Line 1305: missing space after comma

```
        $query->condition('a.id',$_POST['cid'],'=');
```

- Line 1305: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
        $query->condition('a.id',$_POST['cid'],'=');
```

- Line 1309: missing space after comma

```
            ->condition('id',$_POST['cid'],'=')
```

- Line 1309: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
            ->condition('id',$_POST['cid'],'=')
```

- Line 1311: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
          $rowid = intval($_POST['rowid']);
```

- Line 1312: missing space after comma

```
          $html = theme('maestro_project_comments',array('rowid' => $rowid, 'tracking_id' => $_POST['tracking_id']));
```

- Line 1313: Functions should be called with no spaces between the function name and opening parentheses

```
          $retdata = array ('html' => $html,'status' => "1");
```

- Line 1313: missing space after comma

```
          $retdata = array ('html' => $html,'status' => "1");
```

- Line 1314: else statements should begin on a new line

```
        } else {
```

- Line 1315: Functions should be called with no spaces between the function name and opening parentheses

```
          $retdata = array ('status' => "0");
```

- Line 1317: else statements should begin on a new line

```
      } else {
```

- Line 1318: Functions should be called with no spaces between the function name and opening parentheses

```
        $retdata = array ('status' => "0");
```

- Line 1325: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
      $task->tracking_id = maestro_getTaskTrackingId($_POST['taskid']);  // Retrieve the tracking_id from the taskid
```

- Line 1329: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
      $rowid = intval($_POST['rowid']);
```

- Line 1330: missing space after comma

```
      $html = theme('maestro_taskconsole_details',array('task' => $task, 'source' => 'mytasks', 'rowid' => $rowid ));
```

- Line 1332: Functions should be called with no spaces between the function name and opening parentheses

```
      $retdata = array ('html' => $html,'status' => "1");
```

- Line 1332: missing space after comma

```
      $retdata = array ('html' => $html,'status' => "1");
```

- Line 1337: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
      $myflowfilter =  array('myflows' => true);
```

- Line 1344: Control statements should have one space between the control keyword and opening parenthesis

```
      if(is_array($myflowfilter)) {  //this has to be an array.  otherwise, this is not a valid executable area for a non-regular
```

- Line 1348: Control statements should have one space between the control keyword and opening parenthesis

```
        if(!user_access('maestro admin')) {
```

- Line 1349: Functions should be called with no spaces between the function name and opening parentheses

```
          $retdata = array ('html' => t('Illegal attempt on viewing flow data detected.'),'status' => "0");
```

- Line 1349: missing space after comma

```
          $retdata = array ('html' => t('Illegal attempt on viewing flow data detected.'),'status' => "0");
```

- Line 1356: Functions should be called with no spaces between the function name and opening parentheses

```
      $retdata = array ('html' => theme('maestro_all_flows_display', array('ajax_url' => $ajax_url, 'database_result_set' => $row
```

- Line 1356: missing space after comma

```
      $retdata = array ('html' => theme('maestro_all_flows_display', array('ajax_url' => $ajax_url, 'database_result_set' => $row
```

- Line 1362: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
      $task->tracking_id = intval($_POST['projectID']);
```

- Line 1366: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
      $rowid = intval($_POST['projectID']);
```

- Line 1367: missing space after comma

```
      $html = theme('maestro_taskconsole_details',array('task' => $task, 'source' => 'mytasks', 'rowid' => $rowid ));
```

- Line 1368: Functions should be called with no spaces between the function name and opening parentheses

```
      $retdata = array ('html' => $html,'status' => "1");
```

- Line 1368: missing space after comma

```
      $retdata = array ('html' => $html,'status' => "1");
```

- Line 1376: Missing period

```
  * Implements hook_form_alter()
```

- Line 1383: Control statements should have one space between the control keyword and opening parenthesis

```
  if($maestro_content_types === FALSE) {  // if not set - scan templates for unique content types
```

- Line 1385: missing space after comma
```
    $query = db_select('maestro_template_data','template_data');
```
- Line 1386: missing space after comma
```
    $query->addField('template_data','task_data','task_data');
```
- Line 1391: missing space after comma
```
      if (!empty($data['content_type']) AND !in_array($data['content_type'],$types)) {
```
- Line 1402: missing space after comma
```
  if (in_array($form_id,$maestro_content_types)) {
```
- Line 1404: missing space after comma
```
    $requestParms = explode('/',$request);
```
- Line 1407: Control statements should have one space between the control keyword and opening parenthesis
```
    foreach($requestParms as $index => $val) {
```
- Line 1408: Control statements should have one space between the control keyword and opening parenthesis
```
      if($val == 'maestro') {
```
- Line 1414: Control statements should have one space between the control keyword and opening parenthesis
```
    if($maestroIndex >= 2 ) { //really only applies if your site is named maestro, ie localhost/maestro, but a valid index should
```
- Line 1446: Use "elseif" in place of "else if"
```
    else if (array_key_exists('complete form', $form_state) && array_key_exists('maestro_taskid', $form_state['complete form'])) {
```
- Line 1467: Missing period
```
 * Implements hook_node_insert()
```
- Line 1477: missing space after comma
```
    $task->processContent($node->maestro_taskid,'insert',$node);
```
- Line 1523: missing space after comma
```
function maestro_getNodeId($process_id,$content_type) {
```
- Line 1545: There should be no trailing spaces
```
 *
```
- Line 1550: missing space after comma
```
  $res = db_select('maestro_queue','a');
```
- Line 1552: missing space after comma
```
  $res->fields('b',array('tracking_id'));
```
- Line 1564: There should be no trailing spaces
```
 * Retrieve the task history for a maestro project (all related workflow instances)
```
- Line 1571: missing space after comma
```
  $query = db_select('maestro_process','process');
```
- Line 1575: missing space after comma
```
  $query->fields('queue', array('process_id','created_date','started_date','completed_date','status'));
```
- Line 1576: missing space after comma
```
  $query->fields('template', array('taskname','is_dynamic_taskname','dynamic_taskname_variable_id'));
```
- Line 1577: missing space after comma
```
  $query->addField('users','name','username');
```
- Line 1578: missing space after comma
```
  $query->addField('users','uid');
```
- Line 1579: missing space after comma
```
  $query->addField('queue','id','queue_id');
```
- Line 1580: missing space after comma
```
  $query->addField('process','pid','parent_process_id');
```
- Line 1582: missing space after comma
```
  $query->condition('queue.show_in_detail',1,'=');
```
- Line 1583: missing space after comma
```
  $query->condition(db_or()->condition('queue.completed_date',0,'>')->condition('queue.status',MaestroTaskStatusCodes::STATUS_ON_
```
- Line 1589: missing space after comma
```
    $ctask->assigned_date = strftime("%b %d/%Y %H:%M",$record->created_date);
```
- Line 1590: missing space after comma

```
    $ctask->started_date = strftime("%b %d/%Y %H:%M",$record->started_date);
```

- Line 1591: missing space after comma

```
    $ctask->completed_date = strftime("%b %d/%Y %H:%M",$record->completed_date);
```

- Line 1604: Comment should be read "Implements hook_foo()."

```
 * Implementation of hook_maestro_notification_observer() found in the notifications class for Maestro
```

- Line 1607: missing space after comma

```
  return array('MaestroEmailNotification','SAMPLEMaestroTwitterNotification', 'MaestroWatchDogNotification'); //MaestroEmailNotif
```

- Line 1611: Comment should be read "Implements hook_foo()."

```
 * Implementation of hook_mail()
```

- Line 1648: Control statements should have one space between the control keyword and opening parenthesis

```
    foreach($arr2 as $k=>$v) {
```

- Line 1648: Arrays should be formatted with a space separating each element and assignment operator

```
    foreach($arr2 as $k=>$v) {
```

- Line 1676: Arrays should be formatted with a space separating each element and assignment operator

```
    foreach ($arr2 as $k=>$v) {
```

- Line 1692: Functions should be called with no spaces between the function name and opening parentheses

```
function maestro_set_process_flow_name ($process_id, $new_flow_name) {
```

- Line 1693: Control statements should have one space between the control keyword and opening parenthesis

```
  if($process_id >0 && $new_flow_name != '') {
```

- Line 1704: Use an indent of 2 spaces, with no tabs

```
            ->fields(array('description' => $new_flow_name))
```

- Line 1705: Use an indent of 2 spaces, with no tabs

```
            ->condition('id', $proj_id, '=')
```

- Line 1706: Use an indent of 2 spaces, with no tabs

```
            ->execute();
```

- Line 1722: Control statements should have one space between the control keyword and opening parenthesis

```
  if(!user_access('maestro admin')) return false;
```

- Line 1722: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
  if(!user_access('maestro admin')) return false;
```

- Line 1725: missing space after comma

```
  $query->fields('a',array('id'));
```

- Line 1726: missing space after comma

```
  $query->condition('a.tracking_id',$tracking_id,'=');
```

- Line 1728: Control statements should have one space between the control keyword and opening parenthesis

```
  foreach($res as $rec) {
```

- Line 1736: missing space after comma

```
    $query->fields('a',array('id'));
```

- Line 1737: missing space after comma

```
    $query->condition('a.process_id',$rec->id,'=');
```

- Line 1739: Control statements should have one space between the control keyword and opening parenthesis

```
    foreach($queueres as $queuerec) {
```

- Line 1774: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
  return true;
```

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-and.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-batch-edit.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-batch-function-edit.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-batch-function.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-batch.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-content-type-edit.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-content-type.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-end.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-fire-trigger-edit.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-fire-trigger.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-if-edit.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-if.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-interactive-function-edit.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-interactive-function.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-manual-web-edit.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-manual-web.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-set-process-variable-edit.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-set-process-variable.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-start.tpl.php

sites/all/modules/contrib/maestro/theme/structure/tasks/maestro-task-unknown.tpl.php

sites/all/modules/contrib/maestro/theme/structure/maestro-workflow-assign-notify-select-boxes.tpl.php

sites/all/modules/contrib/maestro/theme/structure/maestro-workflow-edit-tasks-frame.tpl.php

sites/all/modules/contrib/maestro/theme/structure/maestro-workflow-edit-template-variables-list.tpl.php

sites/all/modules/contrib/maestro/theme/structure/maestro-workflow-edit-template-variables.tpl.php

sites/all/modules/contrib/maestro/theme/structure/maestro-workflow-edit.tpl.php

sites/all/modules/contrib/maestro/theme/structure/maestro-workflow-list-item.tpl.php

maestro-workflow-list-item.tpl.php

- Line 14: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l('<img src="' . $module_path . '/images/admin/edit_tasks.gif" border="0" title="' . t('Edit Tasks') .'">', 'a
```

- Line 15: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l('<img src="' . $module_path . '/images/admin/edit_properties.gif" border="0" title="' . t('Edit Template Pro
```

- Line 18: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l('<img src="' . $module_path . '/images/admin/export.gif" border="0" title="' . t('Export Template') .'">', '
```

- Line 33: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l('<input class="form-submit" type="button" value="' . t('Close') . '">', 'admin/structure/maestro', a
```

- Line 66: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l('<input class="form-submit" id="tcancel_<?php print $cntr; ?>" type="button" value="' . t('Close
```

sites/all/modules/contrib/maestro/theme/structure/maestro-workflow-list.tpl.php

maestro-workflow-list.tpl.php

- Line 36: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l('here','admin/structure/maestro'); ?>
```

- Line 50: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l('here','admin/structure/maestro'); ?>
```

sites/all/modules/contrib/maestro/theme/structure/maestro-workflow-task-frame.tpl.php

maestro-workflow-task-frame.tpl.php

- Line -1: @file block missing (Drupal Docs)

sites/all/modules/contrib/maestro/theme/maestro-all-flows-display.tpl.php

maestro-all-flows-display.tpl.php

- Line -1: @file block missing (Drupal Docs)

sites/all/modules/contrib/maestro/theme/maestro-all-flows.tpl.php

maestro-all-flows.tpl.php

- Line -1: @file block missing (Drupal Docs)

sites/all/modules/contrib/maestro/theme/maestro-outstanding-tasks.tpl.php

maestro-outstanding-tasks.tpl.php

- Line 37: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l("<img class=\"valigncenter\" src=\"{$maestro_url}/images/taskconsole/reassign.png\">", "maestro/outstand
```

- Line 38: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l("<img class=\"valigncenter\" src=\"{$maestro_url}/images/taskconsole/email.png\">", "maestro/outstanding
```

- Line 39: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l("<img class=\"valigncenter\" src=\"{$maestro_url}/images/taskconsole/trace.png\">", "maestro/trace/0/{$t
```

- Line 40: The $text argument to l() should be enclosed within t() so that it is translatable.

```
<?php print l("<img class=\"valigncenter\" src=\"{$maestro_url}/images/taskconsole/delete.png\">", "maestro/outstandin
```

sites/all/modules/contrib/maestro/theme/maestro-taskconsole.tpl.php

sites/all/modules/contrib/maestro/theme/maestro-trace.tpl.php

sites/all/modules/contrib/maestro/theme/project_detail_comments.tpl.php

project_detail_comments.tpl.php

- Line -1: @file block missing (Drupal Docs)

sites/all/modules/contrib/maestro/theme/project_detail_container.tpl.php

project_detail_container.tpl.php

- Line -1: @file block missing (Drupal Docs)

sites/all/modules/contrib/maestro/modules/maestro_test_flows/theme/maestro-manual-web-example.tpl.php

sites/all/modules/contrib/maestro/modules/maestro_test_flows/maestro_test_flows.install

maestro_test_flows.install

- Line 2: Commits to the Git repository do not require the CVS Id keyword in each file. (Drupal Docs)

```
// $Id:
```

- Line 10: missing space after comma

```
include(drupal_get_path('module','maestro') . '/maestro.admin.inc');
```

- Line 15: Use an indent of 2 spaces, with no tabs

```
        /* Basic Test Workflow Example */
```

- Line 22: do not use mixed case (camelCase), use lower case and _

```
$templateID = maestro_createNewTemplate("Test - Basic Workflow Example", TRUE, TRUE, 500);
```

- Line 23: do not use mixed case (camelCase), use lower case and _

```
$variable_xref_array[1] = maestro_createTemplateVariable($templateID, "initiator", "");
```

- Line 24: do not use mixed case (camelCase), use lower case and _

```
$variable_xref_array[2] = maestro_createTemplateVariable($templateID, "var1", "0");
```

- Line 25: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceStart(0, $templateID);
```

- Line 60: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceEnd(0, $templateID);
```

- Line 95: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceSetProcessVariable(0, $templateID);
```

- Line 130: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceIf(0, $templateID);
```

- Line 165: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 200: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 246: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 247: missing space after comma

```
$query->condition('a.id',$task_xref_array[5],'=');
```

- Line 249: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 251: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 254: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[5],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 258: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 259: missing space after comma

```
$query->condition('a.id',$task_xref_array[6],'=');
```

- Line 261: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 263: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 266: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[6],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 276: do not use mixed case (camelCase), use lower case and _

```
$templateID = maestro_createNewTemplate("Test Simple Parallel AND", TRUE, TRUE, 500);
```

- Line 277: do not use mixed case (camelCase), use lower case and _

```
maestro_joinAppGroup($templateID, $app_group_id);
```

- Line 278: do not use mixed case (camelCase), use lower case and _

```
$variable_xref_array[10] = maestro_createTemplateVariable($templateID, "initiator", "");
```

- Line 279: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceStart(0, $templateID);
```

- Line 313: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceEnd(0, $templateID);
```

- Line 347: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 381: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 415: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 449: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceAnd(0, $templateID);
```

- Line 492: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 493: missing space after comma

```
$query->condition('a.id',$task_xref_array[30],'=');
```

- Line 495: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 497: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 500: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[30],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 504: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 505: missing space after comma

```
$query->condition('a.id',$task_xref_array[31],'=');
```

- Line 507: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 509: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 512: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[31],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 516: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 517: missing space after comma

```
$query->condition('a.id',$task_xref_array[32],'=');
```

- Line 519: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 521: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 524: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[32],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 536: do not use mixed case (camelCase), use lower case and _

```
$templateID = maestro_createNewTemplate("Test Parallel and Sequential Branches", TRUE, TRUE, 600);
```

- Line 537: do not use mixed case (camelCase), use lower case and _

```
maestro_joinAppGroup($templateID, $app_group_id);
```

- Line 538: do not use mixed case (camelCase), use lower case and _

```
$variable_xref_array[11] = maestro_createTemplateVariable($templateID, "initiator", "");
```

- Line 539: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceStart(0, $templateID);
```

- Line 573: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceEnd(0, $templateID);
```

- Line 607: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 641: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 675: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 709: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 743: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 777: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceAnd(0, $templateID);
```

- Line 811: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceAnd(0, $templateID);
```

- Line 852: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 853: missing space after comma

```
$query->condition('a.id',$task_xref_array[36],'=');
```

- Line 855: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 857: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 860: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[36],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 864: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 865: missing space after comma

```
$query->condition('a.id',$task_xref_array[37],'=');
```

- Line 867: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 869: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 872: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[37],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 878: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 879: missing space after comma

```
$query->condition('a.id',$task_xref_array[38],'=');
```

- Line 881: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 883: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 886: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[38],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 890: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 891: missing space after comma

```
$query->condition('a.id',$task_xref_array[39],'=');
```

- Line 893: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 895: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 898: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[39],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 902: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 903: missing space after comma

```
$query->condition('a.id',$task_xref_array[40],'=');
```

- Line 905: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 907: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 910: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[40],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 923: do not use mixed case (camelCase), use lower case and _

  ```
  $templateID = maestro_createNewTemplate("Test Content Type Task - with regen", TRUE, TRUE, 500);
  ```

- Line 924: do not use mixed case (camelCase), use lower case and _

  ```
  $variable_xref_array[5] = maestro_createTemplateVariable($templateID, "initiator", "");
  ```

- Line 925: do not use mixed case (camelCase), use lower case and _

  ```
  $variable_xref_array[6] = maestro_createTemplateVariable($templateID, "content_type", "article");
  ```

- Line 926: do not use mixed case (camelCase), use lower case and _

  ```
  $ti = new MaestroTaskInterfaceStart(0, $templateID);
  ```

- Line 961: do not use mixed case (camelCase), use lower case and _

  ```
  $ti = new MaestroTaskInterfaceEnd(0, $templateID);
  ```

- Line 996: do not use mixed case (camelCase), use lower case and _

  ```
  $ti = new MaestroTaskInterfaceContentType(0, $templateID);
  ```

- Line 1031: do not use mixed case (camelCase), use lower case and _

  ```
  $ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
  ```

- Line 1066: do not use mixed case (camelCase), use lower case and _

  ```
  $ti = new MaestroTaskInterfaceIf(0, $templateID);
  ```

- Line 1101: do not use mixed case (camelCase), use lower case and _

  ```
  $ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
  ```

- Line 1136: do not use mixed case (camelCase), use lower case and _

  ```
  $ti = new MaestroTaskInterfaceIf(0, $templateID);
  ```

- Line 1171: do not use mixed case (camelCase), use lower case and _

  ```
  $ti = new MaestroTaskInterfaceBatchFunction(0, $templateID);
  ```

- Line 1211: missing space after comma

  ```
  $query->fields('a',array('task_class_name'));
  ```

- Line 1212: missing space after comma

  ```
  $query->condition('a.id',$task_xref_array[24],'=');
  ```

- Line 1214: in most cases, replace the string function with the drupal_ equivalent string functions

  ```
  $task_type = substr($name->task_class_name, 15);
  ```

- Line 1216: missing space after comma

  ```
  $ti = new $task_class(0,0);
  ```

- Line 1219: missing space after comma

  ```
  db_query($sql, array( 'a1' => $task_xref_array[24],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
  ```

- Line 1223: missing space after comma

  ```
  $query->fields('a',array('task_class_name'));
  ```

- Line 1224: missing space after comma

  ```
  $query->condition('a.id',$task_xref_array[25],'=');
  ```

- Line 1226: in most cases, replace the string function with the drupal_ equivalent string functions

  ```
  $task_type = substr($name->task_class_name, 15);
  ```

- Line 1228: missing space after comma

  ```
  $ti = new $task_class(0,0);
  ```

- Line 1231: missing space after comma

  ```
  db_query($sql, array( 'a1' => $task_xref_array[25],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
  ```

- Line 1239: missing space after comma

  ```
  $query->fields('a',array('task_class_name'));
  ```

- Line 1240: missing space after comma

  ```
  $query->condition('a.id',$task_xref_array[27],'=');
  ```

- Line 1242: in most cases, replace the string function with the drupal_ equivalent string functions

  ```
  $task_type = substr($name->task_class_name, 15);
  ```

- Line 1244: missing space after comma

  ```
  $ti = new $task_class(0,0);
  ```

- Line 1247: missing space after comma

  ```
  db_query($sql, array( 'a1' => $task_xref_array[27],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
  ```

- Line 1265: do not use mixed case (camelCase), use lower case and _

```
$templateID = maestro_createNewTemplate("Test - Basic Batch Task Example", TRUE, TRUE, 500);
```

- Line 1266: do not use mixed case (camelCase), use lower case and _

```
maestro_joinAppGroup($templateID, $app_group_id);
```

- Line 1267: do not use mixed case (camelCase), use lower case and _

```
$variable_xref_array[34] = maestro_createTemplateVariable($templateID, "initiator", "");
```

- Line 1268: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceStart(0, $templateID);
```

- Line 1302: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceEnd(0, $templateID);
```

- Line 1336: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceBatch(0, $templateID);
```

- Line 1370: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceIf(0, $templateID);
```

- Line 1404: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 1438: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 1483: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 1484: missing space after comma

```
$query->condition('a.id',$task_xref_array[140],'=');
```

- Line 1486: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 1488: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 1491: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[140],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 1495: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 1496: missing space after comma

```
$query->condition('a.id',$task_xref_array[141],'=');
```

- Line 1498: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 1500: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 1503: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[141],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 1512: do not use mixed case (camelCase), use lower case and _

```
$templateID = maestro_createNewTemplate("Test - Manual Web Example", TRUE, TRUE, 500);
```

- Line 1513: do not use mixed case (camelCase), use lower case and _

```
maestro_joinAppGroup($templateID, $app_group_id);
```

- Line 1514: do not use mixed case (camelCase), use lower case and _

```
$variable_xref_array[49] = maestro_createTemplateVariable($templateID, "initiator", "");
```

- Line 1515: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceStart(0, $templateID);
```

- Line 1549: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceEnd(0, $templateID);
```

- Line 1583: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceManualWeb(0, $templateID);
```

- Line 1622: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 1623: missing space after comma

```
$query->condition('a.id',$task_xref_array[214],'=');
```

- Line 1625: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 1627: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 1630: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[214],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

sites/all/modules/contrib/maestro/modules/maestro_technical_support_request_workflow/maestro_technical_support_request_workflow.install

maestro_technical_support_request_workflow.install

- Line -1: @file block missing (Drupal Docs)
- Line 11: missing space after comma

```
include(drupal_get_path('module','maestro') . '/maestro.admin.inc');
```

- Line 19: do not use mixed case (camelCase), use lower case and _

```
$templateID = maestro_createNewTemplate("Technical Support Request", TRUE, TRUE, 700);
```

- Line 20: do not use mixed case (camelCase), use lower case and _

```
$variable_xref_array[1] = maestro_createTemplateVariable($templateID, "initiator", "");
```

- Line 21: do not use mixed case (camelCase), use lower case and _

```
$variable_xref_array[2] = maestro_createTemplateVariable($templateID, "assignee", "0");
```

- Line 22: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceStart(0, $templateID);
```

- Line 57: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceEnd(0, $templateID);
```

- Line 92: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceContentType(0, $templateID);
```

- Line 127: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceSetProcessVariable(0, $templateID);
```

- Line 162: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceContentType(0, $templateID);
```

- Line 197: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInteractiveFunction(0, $templateID);
```

- Line 232: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceIf(0, $templateID);
```

- Line 267: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceInlineFormAPI(0, $templateID);
```

- Line 302: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceBatchFunction(0, $templateID);
```

- Line 337: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceContentType(0, $templateID);
```

- Line 372: do not use mixed case (camelCase), use lower case and _

```
$ti = new MaestroTaskInterfaceIf(0, $templateID);
```

- Line 412: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 413: missing space after comma

```
$query->condition('a.id',$task_xref_array[3],'=');
```

- Line 415: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 417: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 420: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[3],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 426: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 427: missing space after comma

```
$query->condition('a.id',$task_xref_array[5],'=');
```

- Line 429: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 431: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 434: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[5],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 438: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 439: missing space after comma

```
$query->condition('a.id',$task_xref_array[6],'=');
```

- Line 441: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 443: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 446: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[6],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 454: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 455: missing space after comma

```
$query->condition('a.id',$task_xref_array[8],'=');
```

- Line 457: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 459: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 462: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[8],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

- Line 468: missing space after comma

```
$query->fields('a',array('task_class_name'));
```

- Line 469: missing space after comma

```
$query->condition('a.id',$task_xref_array[10],'=');
```

- Line 471: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($name->task_class_name, 15);
```

- Line 473: missing space after comma

```
$ti = new $task_class(0,0);
```

- Line 476: missing space after comma

```
db_query($sql, array( 'a1' => $task_xref_array[10],'b1' => 1,'c1' => 2,'d1' => $modulated_id,));
```

sites/all/modules/contrib/maestro/modules/maestro_inline_form_api_task/theme/maestro-task-inline-form-api-edit.tpl.php

sites/all/modules/contrib/maestro/modules/maestro_inline_form_api_task/theme/maestro-task-inline-form-api.tpl.php

sites/all/modules/contrib/maestro/modules/maestro_content_publish/content_review.tpl.php

content_review.tpl.php

- Line -1: @file block missing (Drupal Docs)

sites/all/modules/contrib/maestro/modules/maestro_common/maestro_show_message_accept_reject.tpl.php

maestro_show_message_accept_reject.tpl.php

- Line -1: @file block missing (Drupal Docs)

sites/all/modules/contrib/maestro/modules/maestro_common/show_message.tpl.php

show_message.tpl.php

- Line -1: @file block missing (Drupal Docs)

sites/all/modules/contrib/maestro/js/admin_template_editor.js

sites/all/modules/contrib/maestro/js/jquery.contextmenu.js

sites/all/modules/contrib/maestro/js/jquery.simplemodal.min.js

sites/all/modules/contrib/maestro/js/maestro_structure_admin.js

sites/all/modules/contrib/maestro/js/moderator.js

moderator.js

- Line 171: Javascript strings should be passed through Drupal.t().

```
alert('An error occurred updating assignment');
```

- Line 174: Javascript strings should be passed through Drupal.t().

```
error: function() { alert('there was a SERVER Error processing AJAX request'); }
```

sites/all/modules/contrib/maestro/js/taskconsole.js

taskconsole.js

- Line 52: Javascript strings should be passed through Drupal.t().

```
alert('An error occurred updating assignment');
```

- Line 55: Javascript strings should be passed through Drupal.t().

```
error: function() { alert('there was a SERVER Error processing AJAX request'); }
```

- Line 83: Javascript strings should be passed through Drupal.t().

```
alert('An error occurred updating assignment');
```

- Line 88: Javascript strings should be passed through Drupal.t().

```
error: function() { alert('there was a SERVER Error processing AJAX request'); }
```

- Line 112: Javascript strings should be passed through Drupal.t().

```
alert('there was a SERVER Error processing AJAX request');
```

- Line 140: Javascript strings should be passed through Drupal.t().

```
alert('An error occurred adding comment');
```

- Line 144: Javascript strings should be passed through Drupal.t().

```
alert('there was a SERVER Error processing AJAX request');
```

- Line 168: Javascript strings should be passed through Drupal.t().

```
alert('An error occurred deleting comment');
```

- Line 172: Javascript strings should be passed through Drupal.t().

```
alert('there was a SERVER Error processing AJAX request');
```

- Line 213: Javascript strings should be passed through Drupal.t().

```
alert('An error occurred processing this interactive task');
```

- Line 216: Javascript strings should be passed through Drupal.t().

```
error : function() { alert('there was a SERVER Error processing AJAX request'); },
```

sites/all/modules/contrib/maestro/js/wz_jsgraphics.js

sites/all/modules/contrib/maestro/batch/maestro_batch_script_sample.php

sites/all/modules/contrib/maestro/lib-test.php

lib-test.php

- Line -1: @file block missing (Drupal Docs)
- Line 16: missing space after comma

```
$notification = new MaestroNotification(1,'this is a test message','Sample Subject',0);
```

sites/all/modules/contrib/maestro/maestro.admin.inc

maestro.admin.inc

- Line 50: use a space between the closing parenthesis and the open bracket

```
function maestro_handle_structure_ajax_request(){
```

- Line 52: Control statements should have one space between the control keyword and opening parenthesis

```
if(!user_access('maestro admin')) return false;
```

- Line 52: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
if(!user_access('maestro admin')) return false;
```

- Line 53: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed by

```
$op=@check_plain($_POST['op']);
```

- Line 54: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed by

```
$cntr=@check_plain($_POST['cntr']);
```

- Line 55: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed by

```
$id=@check_plain($_POST['id']);
```

- Line 57: Control statements should have one space between the control keyword and opening parenthesis

```
switch($op){
```

- Line 57: use a space between the closing parenthesis and the open bracket

```
switch($op){
```

- Line 62: Control statements should have one space between the control keyword and opening parenthesis

```
    if(variable_get('maestro_enable_import_window',0) == 1) {
```

- Line 62: missing space after comma

```
    if(variable_get('maestro_enable_import_window',0) == 1) {
```

- Line 72: Control statements should have one space between the control keyword and opening parenthesis

```
    if(variable_get('maestro_enable_import_window',0) == 1) {
```

- Line 72: missing space after comma

```
    if(variable_get('maestro_enable_import_window',0) == 1) {
```

- Line 73: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed by

```
      $templatecode=($_POST['templatecode']);
```

- Line 75: Control statements should have one space between the control keyword and opening parenthesis

```
      if(strpos(strtolower($templatecode),'delete') !== FALSE ||
```

- Line 75: missing space after comma

```
      if(strpos(strtolower($templatecode),'delete') !== FALSE ||
```

- Line 75: in most cases, replace the string function with the drupal_ equivalent string functions

```
      if(strpos(strtolower($templatecode),'delete') !== FALSE ||
```

- Line 76: missing space after comma

```
      strpos(strtolower($templatecode),'drop') !== FALSE ||
```

- Line 76: in most cases, replace the string function with the drupal_ equivalent string functions

```
      strpos(strtolower($templatecode),'drop') !== FALSE ||
```

- Line 77: missing space after comma

```
      strpos(strtolower($templatecode),'truncate') !== FALSE ) {
```

- Line 77: in most cases, replace the string function with the drupal_ equivalent string functions

```
      strpos(strtolower($templatecode),'truncate') !== FALSE ) {
```

- Line 97: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed by

```
    $name=check_plain($_POST['templateName']);
```

- Line 107: Control statements should have one space between the control keyword and opening parenthesis

```
    if($update>=0){
```

- Line 107: use a space between the closing parenthesis and the open bracket

```
    if($update>=0){
```

- Line 115: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
    $name=check_plain($_POST['newVariableName']);
```

- Line 116: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
$value=check_plain($_POST['newVariableValue']);
```

- Line 117: do not use mixed case (camelCase), use lower case and _

```
$recID=maestro_createTemplateVariable($id, $name, $value);
```

- Line 119: Control statements should have one space between the control keyword and opening parenthesis

```
if($recID > 0){
```

- Line 119: use a space between the closing parenthesis and the open bracket

```
if($recID > 0){
```

- Line 119: do not use mixed case (camelCase), use lower case and _

```
if($recID > 0){
```

- Line 129: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$name=check_plain($_POST['name']);
```

- Line 130: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$value=check_plain($_POST['val']);
```

- Line 139: missing space after comma

```
$query->fields('a',array('template_id'));
```

- Line 140: missing space after comma

```
$query->condition('a.id',$id,'=');
```

- Line 144: Control statements should have one space between the control keyword and opening parenthesis

```
if($update>=0){
```

- Line 144: use a space between the closing parenthesis and the open bracket

```
if($update>=0){
```

- Line 153: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$tid=check_plain($_POST['tid']);
```

- Line 157: Control statements should have one space between the control keyword and opening parenthesis

```
if($ret){
```

- Line 157: use a space between the closing parenthesis and the open bracket

```
if($ret){
```

- Line 165: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$tid=check_plain($_POST['tid']);
```

- Line 178: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$name=check_plain($_POST['name']);
```

- Line 179: do not use mixed case (camelCase), use lower case and _

```
$recID=maestro_createNewTemplate($name);
```

- Line 181: Control statements should have one space between the control keyword and opening parenthesis

```
if($recID > 0){
```

- Line 181: use a space between the closing parenthesis and the open bracket

```
if($recID > 0){
```

- Line 181: do not use mixed case (camelCase), use lower case and _

```
if($recID > 0){
```

- Line 191: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$name=check_plain($_POST['name']);
```

- Line 192: Control statements should have one space between the control keyword and opening parenthesis

```
if($name != '')  $recID=maestro_createAppGroup($name);
```

- Line 192: do not use mixed case (camelCase), use lower case and _

```
if($name != '')  $recID=maestro_createAppGroup($name);
```

- Line 194: Control statements should have one space between the control keyword and opening parenthesis

```
if($recID > 0){
```

- Line 194: use a space between the closing parenthesis and the open bracket

```
if($recID > 0){
```

- Line 194: do not use mixed case (camelCase), use lower case and _

```
if($recID > 0){
```

- Line 209: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$which=check_plain($_POST['which']);
```

- Line 217: Control statements should have one space between the control keyword and opening parenthesis

```
foreach($res as $rec){
```

- Line 217: use a space between the closing parenthesis and the open bracket

```
foreach($res as $rec){
```

- Line 219: missing space after comma

```
  $query->condition('template_data_id',$rec->id,'=');
```

- Line 223: missing space after comma

```
  $query->condition('template_data_id',$rec->id,'=');
```

- Line 227: missing space after comma

```
  $query->condition('template_data_from',$rec->id,'=');
```

- Line 231: missing space after comma

```
  $query->condition('id',$rec->id,'=');
```

- Line 236: missing space after comma

```
$query->condition('template_id',$id,'=');
```

- Line 240: missing space after comma

```
$query->condition('id',$id,'=');
```

- Line 250: missing space after comma

```
$query->fields('a',array('template_name', 'app_group','canvas_height'));
```

- Line 251: missing space after comma

```
$query->condition('a.id',$id,'=');
```

- Line 258: do not use mixed case (camelCase), use lower case and _

```
$newTID=$record->id;
```

- Line 263: Control statements should have one space between the control keyword and opening parenthesis

```
foreach($res as $rec) {
```

- Line 265: do not use mixed case (camelCase), use lower case and _

```
  $newrecord->template_id = $newTID;
```

- Line 272: do not use mixed case (camelCase), use lower case and _

```
$taskDataArray=array();
```

- Line 274: Control statements should have one space between the control keyword and opening parenthesis

```
foreach($res as $rec) {
```

- Line 276: do not use mixed case (camelCase), use lower case and _

```
  $newrecord->template_id = $newTID;
```

- Line 281: in most cases, replace the string function with the drupal_ equivalent string functions

```
  $task_type = substr($rec->task_class_name, 15);
```

- Line 325: do not use mixed case (camelCase), use lower case and _

```
  $taskDataArray[$rec->id]= $newrecord->id;
```

- Line 331: Control statements should have one space between the control keyword and opening parenthesis

```
foreach($res as $rec) {
```

- Line 333: Control statements should have one space between the control keyword and opening parenthesis

```
  foreach($nextstepres as $nextstep) {
```

- Line 336: do not use mixed case (camelCase), use lower case and _

```
    db_query($sql, array('a1' => $taskDataArray[$nextstep->template_data_from], 'b1' => $taskDataArray[$nextstep->template
```

- Line 340: Control statements should have one space between the control keyword and opening parenthesis

```
  foreach($assignmentres as $ares) {
```

- Line 343: Control statements should have one space between the control keyword and opening parenthesis

```
    switch($ares->assign_type) {
```

- Line 345: Control statements should have one space between the control keyword and opening parenthesis

```
        if($assign_by == MaestroAssignmentBy::VARIABLE) {
```

- Line 361: do not use mixed case (camelCase), use lower case and _

```
  db_query($sql, array( 'a1' => $taskDataArray[$ares->template_data_id],
```

- Line 386: missing space after comma

```
'#default_value' => variable_get('maestro_batch_script_location', drupal_get_path('module','maestro') . "/batch/"),
```

- Line 420: do not use mixed case (camelCase), use lower case and _

```
    if ($declaredObserver = $function()) {
```

- Line 421: Control statements should have one space between the control keyword and opening parenthesis

```
      foreach($declaredObserver as $observer) {
```

- Line 421: do not use mixed case (camelCase), use lower case and _

```
      foreach($declaredObserver as $observer) {
```

- Line 428: Control statements should have one space between the control keyword and opening parenthesis

```
  if(variable_get('maestro_enabled_notifiers') == NULL || (is_array(variable_get('maestro_enabled_notifiers')) && count(variable
```

- Line 431: Control statements should have one space between the control keyword and opening parenthesis

```
    foreach($observers as $key => $val) {
```

- Line 432: Control statements should have one space between the control keyword and opening parenthesis

```
      if($key != '0') {
```

- Line 437: missing space after comma

```
    variable_set('maestro_enabled_notifiers',$x);
```

- Line 555: use a space between the closing parenthesis and the open bracket

```
function maestro_edit_properties($tid, $var=0){
```

- Line 565: use a space between the closing parenthesis and the open bracket

```
function maestro_createNewTemplate($name, $skip_task_creation = FALSE, $skip_variable_creation = FALSE, $canvas_height=500){
```

- Line 566: Control statements should have one space between the control keyword and opening parenthesis

```
  if(!user_access('maestro admin')) return false;
```

- Line 566: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
  if(!user_access('maestro admin')) return false;
```

- Line 567: Control statements should have one space between the control keyword and opening parenthesis

```
  if(trim($name) != '') {
```

- Line 572: do not use mixed case (camelCase), use lower case and _

```
    $newTemplateRecId = $record->id;
```

- Line 573: Control statements should have one space between the control keyword and opening parenthesis

```
    if(!$skip_task_creation) {
```

- Line 574: do not use mixed case (camelCase), use lower case and _

```
      $task=new MaestroTaskInterfaceStart(0, $newTemplateRecId);
```

- Line 576: do not use mixed case (camelCase), use lower case and _

```
      $task=new MaestroTaskInterfaceEnd(0, $newTemplateRecId);
```

- Line 579: Control statements should have one space between the control keyword and opening parenthesis

```
    if(!$skip_variable_creation) {
```

- Line 580: do not use mixed case (camelCase), use lower case and _

```
      maestro_createTemplateVariable($newTemplateRecId, 'initiator', '');
```

- Line 582: do not use mixed case (camelCase), use lower case and _

```
    return $newTemplateRecId;
```

- Line 589: use a space between the closing parenthesis and the open bracket

```
function maestro_createTemplateVariable($tid, $name, $value){
```

- Line 590: Control statements should have one space between the control keyword and opening parenthesis

```
  if(!user_access('maestro admin')) return false;
```

- Line 590: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
  if(!user_access('maestro admin')) return false;
```

- Line 599: use a space between the closing parenthesis and the open bracket

```
function maestro_deleteTemplateVariable($id){
```

- Line 600: Control statements should have one space between the control keyword and opening parenthesis

```
  if(!user_access('maestro admin')) return false;
```

- Line 600: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
  if(!user_access('maestro admin')) return false;
```

- Line 602: missing space after comma

```
  $query->fields('a',array('variable_name'));
```

- Line 605: Control statements should have one space between the control keyword and opening parenthesis

```
  if($name->variable_name != 'initiator') {
```

- Line 609: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
    return true;
```

- Line 612: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
    return false;
```

- Line 617: use a space between the closing parenthesis and the open bracket

```
function maestro_createAppGroup($name){
```

- Line 618: Control statements should have one space between the control keyword and opening parenthesis

```
  if(!user_access('maestro admin')) return false;
```

- Line 618: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
  if(!user_access('maestro admin')) return false;
```

- Line 625: use a space between the closing parenthesis and the open bracket

```
function maestro_deleteAppGroup($id){
```

- Line 626: Control statements should have one space between the control keyword and opening parenthesis

```
  if(!user_access('maestro admin')) return false;
```

- Line 626: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
  if(!user_access('maestro admin')) return false;
```

- Line 633: Control statements should have one space between the control keyword and opening parenthesis

```
  if(!user_access('maestro admin')) return false;
```

- Line 633: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
  if(!user_access('maestro admin')) return false;
```

- Line 644: Control statements should have one space between the control keyword and opening parenthesis

```
  if($id > 0) {
```

- Line 653: missing space after comma

```
    $query->fields('a',array('template_name', 'canvas_height'));
```

- Line 654: missing space after comma

```
    $query->condition('a.id',$id,'=');
```

- Line 662: Control statements should have one space between the control keyword and opening parenthesis

```
    foreach($template_variables_record_set as $rec) {
```

- Line 665: Control statements should have one space between the control keyword and opening parenthesis

```
      if($rec->variable_name == 'initiator') {
```

- Line 672: Control statements should have one space between the control keyword and opening parenthesis

```
    foreach($template_data_record_set as $rec) {
```

- Line 673: in most cases, replace the string function with the drupal_ equivalent string functions

```
      $task_type = substr($rec->task_class_name, 15);
```

- Line 682: Control statements should have one space between the control keyword and opening parenthesis

```
      foreach($rec as $key => $data){
```

- Line 682: use a space between the closing parenthesis and the open bracket

```
      foreach($rec as $key => $data){
```

- Line 683: Control statements should have one space between the control keyword and opening parenthesis

```
        if($key!='id' && $key!='task_class_name' && $key!='template_id') {
```

- Line 684: Control statements should have one space between the control keyword and opening parenthesis

```
          if($key == 'task_data') {
```

- Line 688: Control statements should have one space between the control keyword and opening parenthesis

```
          elseif(is_numeric($data)) {
```

- Line 700: Control statements should have one space between the control keyword and opening parenthesis

```
    foreach($template_data_record_set as $rec) {
```

- Line 702: Control statements should have one space between the control keyword and opening parenthesis

```
      foreach($nextstepres as $nextstep) {
```

- Line 707: Control statements should have one space between the control keyword and opening parenthesis

```
      foreach($assignmentres as $ares) {
```

- Line 711: String concatenation should be formatted with a space separating the operators (dot .) and the surrounding terms

```
        $output_php .='$name=current($query->execute()->fetchAll());'. "\n";
```

- Line 724: Control statements should have one space between the control keyword and opening parenthesis

```
        switch($ares->assign_type) {
```

- Line 726: Control statements should have one space between the control keyword and opening parenthesis

```
            if($assign_by == MaestroAssignmentBy::FIXED) {
```

- Line 733: Control statements should have one space between the control keyword and opening parenthesis

```
            if($assign_by == MaestroAssignmentBy::FIXED) {
```

- Line 734: Control statements should have one space between the control keyword and opening parenthesis

```
              if(module_exists('og')) {
```

- Line 736: missing space after comma

```
                $query->addField('a','label');
```

- Line 737: missing space after comma

```
                $query->condition('a.gid', intval($ares->assign_id),'=');
```

- Line 740: String concatenation should be formatted with a space separating the operators (dot .) and the surrounding terms

```
                $modulated_string .='if($modulated_id == FALSE) {'. "\n";
```

- Line 750: The $message argument to watchdog() should NOT be enclosed within t(), so that it can be properly translated at display time.

```
                watchdog('maestro import', t('You do not have OG installed.  Task assignment changed to assigned by INITIATOR va
```

- Line 759: Control statements should have one space between the control keyword and opening parenthesis

```
            if($assign_by == MaestroAssignmentBy::FIXED) {
```

- Line 761: missing space after comma

```
            $query->addField('a','name');
```

- Line 762: missing space after comma

```
            $query->condition('a.rid', intval($ares->assign_id),'=');
```

- Line 784: use a space between the closing parenthesis and the open bracket

```
function maestro_export_template($tid, $var=0){
```

sites/all/modules/contrib/maestro/maestro.class.php

maestro.class.php

- Line -1: @file block missing (Drupal Docs)
- Line 8: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
    private $_engine = null;
```

- Line 10: Functions should be called with no spaces between the function name and opening parentheses

```
    static function createMaestroObject ($version, $options = FALSE){
```

- Line 10: use a space between the closing parenthesis and the open bracket

```
    static function createMaestroObject ($version, $options = FALSE){
```

- Line 14: else statements should begin on a new line

```
        } else {
```

- Line 23: String concatenation should be formatted with a space separating the operators (dot .) and the surrounding terms

```
      $classfile = drupal_get_path('module','maestro')."/maestro_engine_version{$version}.class.php";
```

- Line 23: missing space after comma

```
      $classfile = drupal_get_path('module','maestro')."/maestro_engine_version{$version}.class.php";
```

- Line 28: else statements should begin on a new line

```
        } else {
```

- Line 31: else statements should begin on a new line

```
      } else {
```

- Line 47: use a space between the closing parenthesis and the open bracket

```
    public function engine(){
```

sites/all/modules/contrib/maestro/maestro.install

maestro.install

- Line 935: Arrays should be formatted with a space separating each element and assignment operator

```
      'instance'=> array(
```

- Line 988: missing space after comma

```
db_drop_unique_key('maestro_production_assignments','assign_back_uid');
```

sites/all/modules/contrib/maestro/maestro.moderator.inc

maestro.moderator.inc

- Line 4: indent secondary line of comment one space

```
* @file
```

- Line 5: indent secondary line of comment one space

```
* maestro.moderator.inc
```

- Line 6: indent secondary line of comment one space

```
*/
```

- Line 20: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed by

```
$show_system_tasks = (intval(@($_POST['show_system_tasks'])) == 1) ? TRUE:FALSE;
```

- Line 24: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed by

```
$m->reassignTask($qid, $_POST['reassign_id'], $_POST['current_uid']);
```

- Line 25: The $message argument to drupal_set_message() should be enclosed within t() so that it is translatable.

```
drupal_set_message("Task re-assignment completed and notification sent.");
```

- Line 30: The $message argument to drupal_set_message() should be enclosed within t() so that it is translatable.

```
drupal_set_message("Task assignment reminder notification sent.");
```

- Line 85: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed by

```
$op = $_POST['op'];
```

- Line 174: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed b

```
$rec->archived = $_POST['archived'][$key];
```

- Line 225: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
'myflows' => true,
```

- Line 251: Control statements should have one space between the control keyword and opening parenthesis

```
if(is_array($filters)) {
```

- Line 252: Control statements should have one space between the control keyword and opening parenthesis

```
foreach($filters as $key => $filter) {
```

- Line 253: Control statements should have one space between the control keyword and opening parenthesis

```
if(isset($filter) && $filter !=''){
```

- Line 253: use a space between the closing parenthesis and the open bracket

```
if(isset($filter) && $filter !=''){
```

- Line 255: Control statements should have one space between the control keyword and opening parenthesis

```
switch($key) {
```

sites/all/modules/contrib/maestro/maestro.test

maestro.test

- Line 39: missing space after comma

```
$exp=var_export($v1,true);
```

- Line 40: missing space after comma

```
$exp2=var_export($v2,true);
```

- Line 48: missing space after comma

```
$this->assertResponse(403,t('Non admin user should not be able to access the Maestro Structure menu.'));
```

- Line 59: missing space after comma

```
$this->assertTrue($recID == 2,t('First new template must have an ID of 2.'),'Maestro');
```

- Line 66: missing space after comma

```
$query->addExpression('count(a.id)','cnt');
```

- Line 67: missing space after comma

```
$query->condition('a.template_id',1,'=');
```

- Line 70: missing space after comma

```
$this->varID=maestro_createTemplateVariable(1,'test variable','5');
```

- Line 71: missing space after comma

```
$this->assertTrue($this->varID > 0,t('New Variable must have an ID greater than 0. New ID:' . $this->varID),'Maestro');
```

- Line 75: use a space between the closing parenthesis and the open bracket

```
function testDeleteMaestroTemplateVariable(){
```

- Line 79: missing space after comma

```
$query->addExpression('count(a.id)','cnt');
```

- Line 80: missing space after comma

```
$query->condition('a.template_id',1,'=');
```

- Line 82: missing space after comma

```
$this->assertTrue($cnt->cnt == $this->varcount,t('Deleting variable successful'),'Maestro');
```

- Line 89: missing space after comma

```
$query->addExpression('count(a.id)','cnt');
```

- Line 93: missing space after comma

```
$this->assertTrue($this->appGroupID > 0,t('New App Group must have an ID greater than 0. New ID:' . $this->appGroupID),'Maes
```

- Line 97: use a space between the closing parenthesis and the open bracket

```
function testDeleteMaestroAppGroup(){
```

- Line 101: missing space after comma

```
$query->addExpression('count(a.id)','cnt');
```

- Line 103: missing space after comma

```
$this->assertTrue($cnt->cnt == $this->appGroupCount,t('Deleting App Group successful'),'Maestro');
```

sites/all/modules/contrib/maestro/maestro.views.inc

maestro.views.inc

- Line 3: Commits to the Git repository do not require the CVS Id keyword in each file. (Drupal Docs)

```
// $Id:
```

- Line 12: Comment should be read "Implements hook_foo()."

```
 * Implementation of hook_views_api().
```

- Line 13: indent secondary line of comment one space

```
*/
```

- Line 116: Functions should be called with no spaces between the function name and opening parentheses

```
'field' => array (
```

- Line 125: Functions should be called with no spaces between the function name and opening parentheses

```
'field' => array (
```

- Line 132: Functions should be called with no spaces between the function name and opening parentheses

```
$data['maestro_queue']['process_id'] = array (
```

- Line 135: Functions should be called with no spaces between the function name and opening parentheses

```
'field' => array (
```

- Line 147: Functions should be called with no spaces between the function name and opening parentheses

```
$data['maestro_queue']['engine_version'] = array (
```

- Line 150: Functions should be called with no spaces between the function name and opening parentheses

```
'field' => array (
```

- Line 162: Functions should be called with no spaces between the function name and opening parentheses

```
$data['maestro_queue']['is_interactive'] = array (
```

- Line 165: Functions should be called with no spaces between the function name and opening parentheses

```
'field' => array (
```

- Line 177: Functions should be called with no spaces between the function name and opening parentheses

```
$data['maestro_queue']['status'] = array (
```

- Line 180: Functions should be called with no spaces between the function name and opening parentheses

```
'field' => array (
```

- Line 192: Functions should be called with no spaces between the function name and opening parentheses

```
$data['maestro_queue']['uid'] = array (
```

- Line 195: Functions should be called with no spaces between the function name and opening parentheses

```
'field' => array (
```

- Line 222: missing space after comma

```
@include_once($base_path . drupal_get_path('module','maestro') . '/maestro_contatnts.class.php');
```

- Line 257: missing space after comma

```
$q->addField('b','uid');
```

- Line 258: missing space after comma

```
$q->addField('b','mail');
```

- Line 259: missing space after comma

```
$q->addField('b','name');
```

- Line 260: missing space after comma

```
$q->condition('a.task_id',intval($value));
```

- Line 263: Control statements should have one space between the control keyword and opening parenthesis

```
foreach($res as $record) {
```

- Line 264: Control statements should have one space between the control keyword and opening parenthesis

```
  if($output != '') $output .= "<br>";
```

- Line 264: use
 instead of

```
  if($output != '') $output .= "<br>";
```

- Line 267: Control statements should have one space between the control keyword and opening parenthesis

```
if($output == '') {
```

sites/all/modules/contrib/maestro/maestro_base_engine.class.php

maestro_base_engine.class.php

- Line -1: @file block missing (Drupal Docs)
- Line 32: missing space after comma

```
      watchdog('maestro',"Set debug mode on");
```

- Line 78: else statements should begin on a new line

```
  } elseif ($pid > 0) {
```

- Line 93: Use an indent of 2 spaces, with no tabs

```
   return $task->execute();
```

- Line 97: Use an indent of 2 spaces, with no tabs

```
   return $task->prepareTask();
```

- Line 100: missing space after comma

```
public function showInteractiveTask(MaestroTask $task,$taskid) {
```

- Line 108: Use an indent of 2 spaces, with no tabs

```
    return '';
```

- Line 110: Use "elseif" in place of "else if"

```
  else if (empty($retval)) {
```

- Line 132: Use "elseif" in place of "else if"

```
  else if ($process_id == 0) {
```

- Line 134: missing space after comma

```
     watchdog('maestro',"get_ProcessVariable: The Process ID has not been set.");
```

- Line 142: missing space after comma

```
    $query->addField('a','variable_value');
```

- Line 143: missing space after comma

```
    $query->condition('a.process_id',$process_id,'=');
```

- Line 144: missing space after comma

```
    $query->condition('a.template_variable_id',$variable,'=');
```

- Line 145: else statements should begin on a new line

```
  } else {
```

- Line 146: in most cases, replace the string function with the drupal_ equivalent string functions

```
    $thisvar = strtolower($variable);
```

- Line 148: missing space after comma

```
    $query->addField('a','variable_value');
```

- Line 150: missing space after comma

```
$query->condition('a.process_id',$process_id,'=');
```

- Line 151: missing space after comma

```
$query->condition('b.variable_name',$thisvar,'=');
```

- Line 159: missing space after comma

```
watchdog('maestro',"get_ProcessVariable: $variable -> $retval");
```

- Line 164: missing space after comma

```
watchdog('maestro',"get_processVariable -> Process:{$this->_processId}, variable:$variable - DOES NOT EXIST");
```

- Line 178: in most cases, replace the string function with the drupal_ equivalent string functions

```
$thisvar = strtolower($variableName);
```

- Line 183: Use "elseif" in place of "else if"

```
else if ($process_id == 0) {
```

- Line 185: missing space after comma

```
watchdog('maestro',"get_ProcessVariable: The Process ID has not been set.");
```

- Line 192: missing space after comma

```
$query->addField('a','id','process_variable_id');
```

- Line 193: missing space after comma

```
$query->addField('a','template_variable_id','variable_id');
```

- Line 196: missing space after comma

```
$query->condition('b.variable_name',$thisvar,'=');
```

- Line 207: missing space after comma

```
watchdog('maestro',"set_processVariable -> Process:{$process_id}, variable:$thisvar, value:$variableValue");
```

- Line 229: missing space after comma

```
watchdog('maestro',"set_processVariable -> Process:{$process_id}, variable:$thisvar - DOES NOT EXIST");
```

- Line 236: Functions should be called with no spaces between the function name and opening parentheses

```
function sendTaskAssignmentNotifications ($qid=0) {
```

- Line 248: Functions should be called with no spaces between the function name and opening parentheses

```
function sendTaskCompletionNotifications ($qid=0) {
```

- Line 260: Functions should be called with no spaces between the function name and opening parentheses

```
function sendTaskReminderNotifications ($qid=0, $user_id=0) {
```

- Line 270: Control statements should have one space between the control keyword and opening parenthesis

```
if($user_id != 0) {
```

- Line 351: missing space after comma

```
abstract function assignTask($queueId,$userObject);
```

- Line 355: missing space after comma

```
abstract function completeTask($queueId,$status = 1);
```

sites/all/modules/contrib/maestro/maestro_constants.class.php

maestro_constants.class.php

- Line 2: Commits to the Git repository do not require the CVS Id keyword in each file. (Drupal Docs)

```
// $Id:
```

- Line 130: Control statements should have one space between the control keyword and opening parenthesis

```
if(module_exists('og')) {
```

- Line 132: else statements should begin on a new line

```
} else {
```

sites/all/modules/contrib/maestro/maestro_engine_version1.class.php

maestro_engine_version1.class.php

- Line -1: @file block missing (Drupal Docs)
- Line 41: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
function newProcess($template, $startoffset = null, $pid = null , $useExistingGroupingId = FALSE) {
```

- Line 46: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
if ($startoffset == null ) {
```

- Line 48: missing space after comma

```
$query->fields('a',array('template_data_from'));
```

- Line 49: missing space after comma

```
$query->fields('b',array('regen_all_live_tasks','show_in_detail','reminder_interval','task_class_name','is_interactive','t
```

- Line 50: missing space after comma

```
$query->addField('b','id','taskid');
```

- Line 51: missing space after comma

```
$query->fields('c',array('template_name'));
```

- Line 54: missing space after comma

```
$query->condition('b.first_task',1,'=');
```

- Line 55: missing space after comma

```
$query->condition('c.id',$template,'=');
```

- Line 56: missing space after comma

```
$query->orderBy('template_data_from','ASC');
```

- Line 57: missing space after comma

```
$query->range(0,1);
```

- Line 62: missing space after comma

```
$query = db_select('maestro_template_data','a');
```

- Line 63: missing space after comma

```
$query->addField('a','id','taskid');
```

- Line 64: missing space after comma

```
$query->fields('b',array('template_name'));
```

- Line 65: missing space after comma

```
$query->fields('a',array('regen_all_live_tasks','show_in_detail','reminder_interval','task_class_name','is_interactive','t
```

- Line 67: missing space after comma

```
$query->condition('a.id',$startoffset);
```

- Line 70: missing space after comma

```
watchdog('maestro','New process code executing');
```

- Line 79: In SQL strings, Use db_query() placeholders in place of variables. This is a potential source of SQL injection attacks when the variable can come from u

```
$flowname = db_query("SELECT flow_name FROM {maestro_process} WHERE id=$pid")->fetchField();
```

- Line 82: In SQL strings, Use db_query() placeholders in place of variables. This is a potential source of SQL injection attacks when the variable can come from u

```
$flowname = db_query("SELECT template_name FROM {maestro_template} WHERE id=$template")->fetchField();
```

- Line 93: missing space after comma

```
drupal_write_record('maestro_process',$process_record);
```

- Line 132: missing space after comma

```
drupal_write_record('maestro_queue',$queue_record);
```

- Line 143: missing space after comma

```
->condition('id',$pid,'=')
```

- Line 149: Control statements should have one space between the control keyword and opening parenthesis

```
if($templaterec->regen_all_live_tasks == 1) {
```

- Line 150: missing space after comma

```
$q2 = db_select('maestro_queue','a');
```

- Line 151: missing space after comma

```
$q2->addField('a','id','id');
```

- Line 153: missing space after comma

```
$q2->condition('b.task_class_name','MaestroTaskTypeAnd');
```

- Line 154: missing space after comma

```
$q2->condition('a.process_id',$pid);
```

- Line 155: missing space after comma

```
$q2->condition(db_or()->condition('a.archived',0)->condition('a.archived',NULL));
```

- Line 161: missing space after comma

- Line 161: missing space after comma

```
            $q3 = db_select('maesto_queue_from','a');
```

- Line 162: missing space after comma

```
            $q3->addField('a','from_queue_id');
```

- Line 169: missing space after comma

```
              drupal_write_record('maestro_queue',$record);
```

- Line 174: missing space after comma

```
            ->condition(db_or()->condition('archived',0)->condition('archived',NULL))
```

- Line 179: missing space after comma

```
        $pvquery = db_select('maestro_process_variables','a');
```

- Line 180: missing space after comma

```
        $pvquery->addExpression($new_processid,'process_id');
```

- Line 181: missing space after comma

```
        $pvquery->fields('a',array('variable_value','template_variable_id'));
```

- Line 182: missing space after comma

```
        $pvquery->condition('a.process_id',$pid);
```

- Line 184: missing space after comma

```
        ->fields(array('variable_value','template_variable_id','process_id'))
```

- Line 189: else statements should begin on a new line

```
      } else {
```

- Line 191: missing space after comma

```
        $pvquery = db_select('maestro_template_variables','a');
```

- Line 192: missing space after comma

```
        $pvquery->addExpression($new_processid,'process_id');
```

- Line 193: missing space after comma

```
        $pvquery->fields('a',array('variable_value','id'));
```

- Line 194: missing space after comma

```
        $pvquery->condition('a.template_id',$template,'=');
```

- Line 196: missing space after comma

```
        ->fields(array('variable_value','template_variable_id','process_id'))
```

- Line 201: missing space after comma

```
        watchdog('maestro',"New queue id (1) : {$queue_record->id} - Template Taskid: {$templaterec->taskid}");
```

- Line 213: Control statements should have one space between the control keyword and opening parenthesis

```
      if($useExistingGroupingId === FALSE) {
```

- Line 215: Control statements should have one space between the control keyword and opening parenthesis

```
        if(empty($pid)) {
```

- Line 222: missing space after comma

```
          drupal_write_record('maestro_projects',$project_record);
```

- Line 225: missing space after comma

```
            watchdog('maestro',"new process: created new project_id: {$project_record->id}");
```

- Line 231: missing space after comma

```
          $parent_tracking_id = db_select('maestro_process','a')
```

- Line 232: missing space after comma

```
          ->fields('a',array('tracking_id'))
```

- Line 235: missing space after comma

```
          $related_processes = db_select('maestro_projects','a')
```

- Line 236: missing space after comma

```
          ->fields('a',array('related_processes'))
```

- Line 250: missing space after comma

```
            watchdog('maestro',"updated existing project record process ({$new_processid}), set related_processes set to: $relat
```

- Line 259: Control statements should have one space between the control keyword and opening parenthesis

```
        if(!empty($pid)) {
```

- Line 263: missing space after comma

```
            $related_processes = db_select('maestro_projects','a')
```

```
            $related_processes = db_select('maestro_projects', 'a')
```

- Line 264: missing space after comma

```
            ->fields('a',array('related_processes'))
```

- Line 267: Control statements should have one space between the control keyword and opening parenthesis

```
            if(!empty($related_processes)) {
```

- Line 280: Control statements should have one space between the control keyword and opening parenthesis

```
      if($this->getTrackingId() == NULL) {
```

- Line 315: missing space after comma

```
    $query->fields('a',array('id','status','archived','template_data_id','task_class_name','engine_version','is_interactive'));
```

- Line 316: missing space after comma

```
    $query->addField('b','id','process_id');
```

- Line 317: missing space after comma

```
    $query->addField('c','task_class_name','step_type');
```

- Line 318: missing space after comma

```
    $query->addField('a','handler');
```

- Line 319: missing space after comma

```
    $query->addField('d','template_name');
```

- Line 320: missing space after comma

```
    $query->condition(db_and()->condition('a.archived',0)->condition('b.complete',0)->condition('a.run_once',0));
```

- Line 323: missing space after comma

```
      watchdog('maestro',"CleanQueue: Number of entries in the queue:" . count($res));
```

- Line 329: missing space after comma

```
        watchdog('maestro',"CleanQueue: processing task of type: {$queueRecord->step_type}");
```

- Line 347: else statements should begin on a new line

```
      } else {
```

- Line 362: missing space after comma

```
          watchdog('maestro',"Failed Task: {$this->_queueId}, Process: {$this->_processId} , Step Type: $this->_taskType");
```

- Line 363: String concatenation should be formatted with a space separating the operators (dot .) and the surrounding terms

```
          watchdog('maestro',"Task Information: ". $task->getMessage());
```

- Line 363: missing space after comma

```
          watchdog('maestro',"Task Information: ". $task->getMessage());
```

- Line 367: Use "elseif" in place of "else if"

```
        else if ($task->completionStatus > 0) {
```

- Line 378: missing space after comma

```
      watchdog('maestro','cleanQueue - 0 rows returned.  Nothing in queue.');
```

- Line 393: Control statements should have one space between the control keyword and opening parenthesis

```
    if($this->_lastTestStatus == MaestroTaskStatusCodes::STATUS_IF_CONDITION_FALSE) {
```

- Line 394: missing space after comma

```
      $query->addField('b','template_data_to_false','taskid');
```

- Line 397: missing space after comma

```
      $query->addField('b','template_data_to','taskid');
```

- Line 399: missing space after comma

```
    $query->fields('c',array('task_class_name','is_interactive','show_in_detail','reminder_interval'));
```

- Line 401: Control statements should have one space between the control keyword and opening parenthesis

```
    if($this->_lastTestStatus == MaestroTaskStatusCodes::STATUS_IF_CONDITION_FALSE) {
```

- Line 407: missing space after comma

```
    $query->condition('a.process_id',$this->_processId,'=');
```

- Line 408: missing space after comma

```
    $query->condition('a.id',$this->_queueId,'=');
```

- Line 421: else statements should begin on a new line

```
    } else { // we've got tasks
```

- Line 424: missing space after comma

```
          watchdog('maestro',"Got tasks  qid: {$this->_queueId}, pid: {$this->_processId} and Next taskid: {$nextTaskRec->taskid
```

- Line 427: Use uppercase for PHP constants e.g. NULL, TRUE, FALSE

- Line 427: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
        if ($nextTaskRec->taskid == null or $nextTaskRec->taskid == '' ) {
```

- Line 440: missing space after comma

```
          $query->addField('a','id');
```

- Line 441: missing space after comma

```
          $query->addExpression('COUNT(a.id)','rec_count');
```

- Line 443: missing space after comma

```
          $query->condition('a.process_id', $this->_processId,'=');
```

- Line 444: missing space after comma

```
          $query->condition('a.template_data_id', $nextTaskRec->taskid,'=');
```

- Line 474: missing space after comma

```
              drupal_write_record('maestro_queue',$queue_record);
```

- Line 479: missing space after comma

```
              drupal_write_record('maestro_queue_from',$next_record);
```

- Line 484: missing space after comma

```
                  $logmsg .= "Assigned to " . $this->getTaskOwner($nextTaskRec->taskid,$this->_processId);
```

- Line 494: missing space after comma

```
                $this->assignTask($queue_record->id,$newTaskAssignedUsers);
```

- Line 502: Potential problem: drupal_set_message() only accepts filtered text, be sure to use check_plain(), filter_xss() or similar to ensure your $variable is fully sa

```
              drupal_set_message("Invalid Task Type: {$nextTaskRec->task_class_name}", 'error');
```

- Line 502: The $message argument to drupal_set_message() should be enclosed within t() so that it is translatable.

```
              drupal_set_message("Invalid Task Type: {$nextTaskRec->task_class_name}", 'error');
```

- Line 512: missing space after comma

```
            $query->fields('a',array('id','regenerate','template_id'));
```

- Line 513: missing space after comma

```
            $query->addExpression('COUNT(id)','rec_count');
```

- Line 517: missing space after comma

```
            $query->condition('a.id', $nextTaskRec->taskid,'=');
```

- Line 533: missing space after comma

```
              drupal_write_record('maestro_queue_from',$next_record);
```

- Line 536: missing space after comma

```
              $query->addExpression('COUNT(id)','rec_count');
```

- Line 537: missing space after comma

```
              $query->condition('a.process_id', $this->_processId,'=');
```

- Line 538: missing space after comma

```
              $query->condition('a.template_data_id', $nextTaskRec->taskid,'=');
```

- Line 561: missing space after comma

```
  function assignTask($queueId,$assignment) {
```

- Line 569: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
          if (strpos($assignId, ':') !== false) {
```

- Line 581: Control statements should have one space between the control keyword and opening parenthesis

```
            if($processVariableId < 0) $processVariableId = 0;
```

- Line 584: missing space after comma

```
              $query->fields('a',array('away_start','away_return','is_active'));
```

- Line 585: missing space after comma

```
              $query->condition('a.uid',$assignId,'=');
```

- Line 618: missing space after comma

```
            $query->addField('a','assign_id');
```

- Line 619: missing space after comma

```
            $query->condition('a.task_id',$queueId,'=');
```

- Line 621: missing space after comma

```
              $query->condition('a.process_variable',$processVariableId,'=');
```

- Line 624: missing space after comma

```
              $query->condition('a.process_variable',0,'<');
```

```
            $query->condition('a.process_variable',0,'=');
```

- Line 625: missing space after comma

```
            $query->condition('a.assign_id',$assignId,'=');
```

- Line 627: missing space after comma

```
          $query->condition('a.assign_type',$assignType,'=');
```

- Line 632: missing space after comma

```
            ->fields(array('task_id','assign_id','assign_type','process_variable','assign_back_id','last_updated'))
```

- Line 647: missing space after comma

```
            ->condition('process_variable',$processVariableId,'=')
```

- Line 684: Use "elseif" in place of "else if"

```
      else if ($currentUid > 0) {
```

- Line 698: missing space after comma

```
        ->fields(array('task_id','assign_id','process_variable','assign_back_id','last_updated'))
```

- Line 709: else statements should begin on a new line

```
      } else {
```

- Line 723: missing space after comma

```
      $query->join('maestro_process','b','b.id = a.process_id');
```

- Line 724: missing space after comma

```
      $query->fields('a', array('id','template_data_id', 'process_id'));
```

- Line 754: missing space after comma

```
      ->fields(array('tracking_id','uid','task_id','timestamp','comment'))
```

- Line 822: put a space between the asterisk and the comment text

```
    /*TODO: hack for now to support current assignment. in beta we will need to return the $assigned array as it is right now, w
```

- Line 839: missing space after comma

```
      watchdog('maestro',"Task ID #$qid no longer exists in queue table.  It was potenially removed by an admin from outstanding
```

- Line 847: missing space after comma

```
      watchdog('maestro',"Complete_task - updating queue item: $qid, project (tracking id): $trackingId");
```

- Line 853: Control statements should have one space between the control keyword and opening parenthesis

```
    if($is_interactive == MaestroInteractiveFlag::IS_INTERACTIVE) {
```

- Line 854: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
      if ($this->_userId == '' or $this->_userId == null ) {
```

- Line 857: else statements should begin on a new line

```
      } else {
```

- Line 867: missing space after comma

```
    ->condition('id',$qid,'=')
```

- Line 875: missing space after comma

```
    ->condition('task_id',$qid,'=')
```

- Line 883: missing space after comma

```
    ->condition('id',$qid,'=')
```

- Line 888: missing space after comma

```
    ->condition('task_id',$qid,'=')
```

- Line 896: missing space after comma

```
    ->condition('id',$qid,'=')
```

- Line 912: missing space after comma

```
        watchdog('maestro',"Entering getQueue - {$this->mode} mode");
```

- Line 919: missing space after comma

```
      $query->fields('a',array('id','template_data_id','process_id','is_interactive','handler','task_data','created_date','start
```

- Line 920: missing space after comma

```
      $query->fields('b',array('task_class_name','template_id','taskname','is_dynamic_taskname','dynamic_taskname_variable_id'))
```

- Line 922: missing space after comma

```
        $query->fields('c',array('assign_id', 'assign_type'));
```

- Line 923: missing space after comma

```
        $query->fields('e',array('name'));
```

- Line 926: missing space after

- Line 926: missing space after comma

```
$query->addField('d','pid','parent_process_id');
```

- Line 927: missing space after comma

```
$query->fields('d',array('tracking_id','flow_name'));
```

- Line 929: missing space after comma

```
$query->condition('c.assign_id',$this->_userId,'=');
```

- Line 934: missing space after comma

```
$query->condition(db_or()->condition('a.archived',0)->condition('a.archived',NULL));
```

- Line 937: missing space after comma

```
$query->orderBy('a.id','DESC');
```

- Line 947: missing space after comma

```
$query->fields('a',array('id','template_data_id','process_id','is_interactive','handler','task_data','created_date','start
```

- Line 948: missing space after comma

```
$query->fields('b',array('task_class_name','template_id','taskname','is_dynamic_taskname','dynamic_taskname_variable_id'))
```

- Line 950: missing space after comma

```
$query->fields('c',array('assign_id', 'assign_type'));
```

- Line 951: missing space after comma

```
$query->fields('g',array('name'));
```

- Line 955: missing space after comma

```
$query->addField('d','pid','parent_process_id');
```

- Line 956: missing space after comma

```
$query->fields('d',array('tracking_id','flow_name'));
```

- Line 958: missing space after comma

```
$query->condition('f.uid',$this->_userId,'=');
```

- Line 963: missing space after comma

```
$query->condition(db_or()->condition('a.archived',0)->condition('a.archived',NULL));
```

- Line 966: missing space after comma

```
$query->orderBy('a.id','DESC');
```

- Line 977: missing space after comma

```
$query->fields('a',array('id','template_data_id','process_id','is_interactive','handler','task_data','created_date','sta
```

- Line 978: missing space after comma

```
$query->fields('b',array('task_class_name','template_id','taskname','is_dynamic_taskname','dynamic_taskname_variable_id'
```

- Line 979: missing space after comma

```
$query->fields('c',array('assign_id', 'assign_type', 'process_variable'));
```

- Line 984: missing space after comma

```
$query->addField('d','pid','parent_process_id');
```

- Line 985: missing space after comma

```
$query->fields('d',array('tracking_id','flow_name'));
```

- Line 989: missing space after comma

```
$query->condition(db_or()->condition('a.archived',0)->condition('a.archived',NULL));
```

- Line 992: missing space after comma

```
$query->orderBy('a.id','DESC');
```

- Line 1025: else statements should begin on a new line

```
} else {
```

- Line 1026: missing space after comma

```
watchdog('maestro',"maestro->getQueue could not resolve the assigned to user id for queue record {$userTaskRecor
```

- Line 1033: else statements should begin on a new line

```
} else {
```

- Line 1039: Control statements should have one space between the control keyword and opening parenthesis

```
for($flagcntr = 0;$flagcntr <= $this->_userTaskCount;$flagcntr++ ) {
```

- Line 1055: missing space after comma

```
$regenquery->addExpression('COUNT(id)','rec_count');
```

- Line 1056: missing space after comma

```
                $regenquery->condition('a.process_id', $userTaskRecord->parent_process_id,'=');
```

- Line 1057: missing space after comma

```
                $regenquery->condition(db_and()->condition('a.template_data_id', $userTaskRecord->template_data_id,'='));
```

- Line 1085: Control statements should have one space between the control keyword and opening parenthesis

```
            if($userTaskRecord->is_dynamic_taskname == 1) {
```

- Line 1087: missing space after comma

```
                $q2->addField('a','variable_value');
```

- Line 1088: missing space after comma

```
                $q2->condition('a.process_id',$userTaskRecord->process_id,'=');
```

- Line 1089: missing space after comma

```
                $q2->condition('a.template_variable_id',$userTaskRecord->dynamic_taskname_variable_id,'=');
```

- Line 1119: missing space after comma

```
      watchdog('maestro',"Exiting getQueue - user mode");
```

sites/all/modules/contrib/maestro/maestro_engine_version2.class.php

maestro_engine_version2.class.php

- Line -1: @file block missing (Drupal Docs)
- Line 2: There should be no trailing spaces

- Line 4: There should be no trailing spaces

- Line 6: There should be no trailing spaces

- Line 9: There should be no trailing spaces

- Line 11: use
 instead of

```
        echo "<br>Version 2 __constructor";
```

- Line 13: There should be no trailing spaces

```
        $this->_properties = $options;
```

- Line 15: There should be no trailing spaces

- Line 16: There should be no trailing spaces

- Line 18: There should be no trailing spaces

```
        return $this->_version;
```

- Line 19: There should be no trailing spaces

```
      }
```

- Line 20: There should be no trailing spaces

- Line 22: There should be no trailing spaces

- Line 23: missing space after comma

```
    function assignTask($queueId,$userObject) {}
```

- Line 24: There should be no trailing spaces

- Line 25: There should be no trailing spaces

```
    function completeTask($queueId) {}
```

- Line 26: There should be no trailing spaces

- Line 27: There should be no trailing spaces

```
    function archiveTask($queueId) {}
```

- Line 28: There should be no trailing spaces

- Line 29: There should be no trailing spaces

```
function cancelTask($queueId) {}
```

- Line 32: There should be no trailing spaces

- Line 33: missing space after comma

```
function setProcessVariable($variable,$value) {}
```

- Line 33: There should be no trailing spaces

```
function setProcessVariable($variable,$value) {}
```

- Line 36: There should be no trailing spaces

- Line 37: There should be no trailing spaces

sites/all/modules/contrib/maestro/maestro_interface.class.php

maestro_interface.class.php

- Line 17: Control statements should have one space between the control keyword and opening parenthesis

```
if($context_options === FALSE) {
```

- Line 32: Control statements should have one space between the control keyword and opening parenthesis

```
if($handler_options === FALSE) {
```

- Line 37: missing space after comma

```
        $handler_options = maestro_array_merge_keys($arr,$handler_options);
```

- Line 72: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($rec->task_class_name, 15);
```

- Line 100: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($option['class_name'], 20);
```

- Line 126: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($option['class_name'], 20);
```

- Line 148: the final ?> should be omitted from all code files
- Line 156: Arrays should be formatted with a space separating each element and assignment operator

```
$res2 = DB_query("SELECT template_data_to, template_data_to_false FROM {maestro_template_data_next_step} WHERE template_da
```

- Line 158: Functions should be called with no spaces between the function name and opening parentheses

```
$to = intval ($rec2->template_data_to);
```

- Line 159: Functions should be called with no spaces between the function name and opening parentheses

```
$to_false = intval ($rec2->template_data_to_false);
```

sites/all/modules/contrib/maestro/maestro_notification.class.php

maestro_notification.class.php

- Line 14: indent secondary line of comment one space

```
*/
```

- Line 66: Control statements should have one space between the control keyword and opening parenthesis

```
if($observers === FALSE) {  //build the observer cache
```

- Line 71: Control statements should have one space between the control keyword and opening parenthesis

```
    foreach($declaredObserver as $observerToCache) {
```

- Line 135: Control statements should have one space between the control keyword and opening parenthesis

```
if(intval($this->_queueID) > 0 && $this->_notificationType != '') {
```

- Line 208: Control statements should have one space between the control keyword and opening parenthesis

```
if(is_array($userIDs) && count($userIDs) > 0) {
```

- Line 217: use a space between the closing parenthesis and the open bracket

```
function getUserIDs(){
```

- Line 249: Control statements should have one space between the control keyword and opening parenthesis

```
if(variable_get('maestro_enable_notifications', 1) == 1) {
```

```
if(variable_get('maestro_enable_notifications',1) == 1) {
```

- Line 249: missing space after comma
- Line 252: Control statements should have one space between the control keyword and opening parenthesis

```
  if($enabled_notifiers == NULL) {
```

- Line 253: Control statements should have one space between the control keyword and opening parenthesis

```
    if(is_array($this->_observers) && count($this->_observers) > 0 ) {
```

- Line 254: Control statements should have one space between the control keyword and opening parenthesis

```
      foreach($this->_observers as $obj) {
```

- Line 255: Control statements should have one space between the control keyword and opening parenthesis

```
        if(class_exists($obj)) {
```

- Line 256: Use an indent of 2 spaces, with no tabs

```
         $notifyObject = new $obj();
```

- Line 263: Control statements should have one space between the control keyword and opening parenthesis

```
    foreach($enabled_notifiers as $obj) {
```

- Line 264: Control statements should have one space between the control keyword and opening parenthesis

```
      if(class_exists($obj)) {
```

- Line 290: Control statements should have one space between the control keyword and opening parenthesis

```
if(is_array($obj->getUserIDs())) {
```

- Line 291: Control statements should have one space between the control keyword and opening parenthesis

```
  foreach($obj->getUserIDs() as $userID) {
```

- Line 309: Control statements should have one space between the control keyword and opening parenthesis

```
if(is_array($obj->getUserIDs())) {
```

- Line 310: Control statements should have one space between the control keyword and opening parenthesis

```
  foreach($obj->getUserIDs() as $userID) {
```

- Line 311: String concatenation should be formatted with a space separating the operators (dot .) and the surrounding terms

```
    watchdog('Maestro', "Notification issued for UserID: ". $userID . " email address: " . $obj->getUserEmailAddresses($user
```

- Line 334: Control statements should have one space between the control keyword and opening parenthesis

```
if(is_array($obj->getUserIDs())) {
```

- Line 335: Control statements should have one space between the control keyword and opening parenthesis

```
  foreach($obj->getUserIDs() as $userID) {
```

sites/all/modules/contrib/maestro/maestro_task_interface.class.php

maestro_task_interface.class.php

- Line 45: use a space between the closing parenthesis and the open bracket

```
function get_task_id(){
```

- Line 72: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed by

```
$rec->offset_left = $_POST['offset_left'];
```

- Line 73: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed by

```
$rec->offset_top = $_POST['offset_top'];
```

- Line 93: In SQL strings, Use db_query() placeholders in place of variables. This is a potential source of SQL injection attacks when the variable can come from u

```
  db_query("DELETE FROM {maestro_template_data_next_step} WHERE template_data_to={$tdID} OR template_data_to_false={$tdID} O
```

- Line 101: use
 instead of

```
  $retval .= '<br>';
```

- Line 127: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($rec->task_class_name, 15);
```

- Line 142: in most cases, replace the string function with the drupal_ equivalent string functions

```
$task_type = substr($vars->task_class_name, 15);
```

- Line 169: Control statements should have one space between the control keyword and opening parenthesis

```
  if(module_exists('og')) {
```

- Line 187: Control statements should have one space between the control keyword and opening parenthesis

```
  foreach($types as $j => $opt) {
```

- Line 190: Control statements should have one space between the control keyword and opening parenthesis

```
foreach($bys as $k => $opt) {
```

- Line 193: Control statements should have one space between the control keyword and opening parenthesis

```
foreach($whens as $l => $opt) {
```

- Line 269: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->pre_notify_subject = $_POST['pre_notify_subject'];
```

- Line 270: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->pre_notify_message = $_POST['pre_notify_message'];
```

- Line 271: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->post_notify_subject = $_POST['post_notify_subject'];
```

- Line 272: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->post_notify_message = $_POST['post_notify_message'];
```

- Line 273: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->reminder_subject = $_POST['reminder_subject'];
```

- Line 274: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->reminder_message = $_POST['reminder_message'];
```

- Line 275: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->escalation_subject = $_POST['escalation_subject'];
```

- Line 276: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->escalation_message = $_POST['escalation_message'];
```

- Line 277: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->reminder_interval = $_POST['reminder_interval'];
```

- Line 278: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->escalation_interval = $_POST['escalation_interval'];
```

- Line 287: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$optional_parms[$var_name] = $_POST['op_var_values'][$key];
```

- Line 297: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->taskname = $_POST['taskname'];
```

- Line 299: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->regenerate = $_POST['regen'];
```

- Line 302: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->regen_all_live_tasks = $_POST['regenall'];
```

- Line 305: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->show_in_detail = $_POST['showindetail'];
```

- Line 315: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$offset_left = intval($_POST['offset_left']);
```

- Line 316: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$offset_top = intval($_POST['offset_top']);
```

- Line 329: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$cond1 = db_or()->condition('a.template_data_to', $_POST['line_to'], '=')->condition('a.template_data_to_false', $_POST['lin
```

- Line 333: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$cond2fin = db_and()->condition('a.template_data_from', $_POST['line_to'], '=')->condition($cond2);
```

- Line 343: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->template_data_to = $_POST['line_to'];
```

- Line 357: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$cond1 = db_or()->condition('a.template_data_to', $_POST['line_to'], '=')->condition('a.template_data_to_false', $_POST['lin
```

- Line 361: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$cond2fin = db_and()->condition('a.template_data_from', $_POST['line_to'], '=')->condition($cond2);
```

- Line 372: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->template_data_to_false = $_POST['line_to'];
```

- Line 385: In SQL strings, Use db_query() placeholders in place of variables. This is a potential source of SQL injection attacks when the variable can come from

```
db_query("DELETE FROM {maestro_template_data_next_step} WHERE template_data_from={$taskid} OR template_data_to={$taskid} OR
```

- Line 391: Functions should be called with no spaces between the function name and opening parentheses

```
$options = array (
```

- Line 535: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->canvas_height = $_POST['height'];
```

- Line 551: missing space after comma

```
$query->fields('a',array('uid'));
```

- Line 552: missing space after comma

```
$query->condition('a.name',$username,'=');
```

- Line 554: Control statements should have one space between the control keyword and opening parenthesis

```
if(is_object($uid)) {
```

- Line 567: missing space after comma

```
$query->fields('a',array('rid'));
```

- Line 570: Control statements should have one space between the control keyword and opening parenthesis

```
if(is_numeric($rid->rid)) {
```

- Line 582: Control statements should have one space between the control keyword and opening parenthesis

```
if(module_exists('og')) {
```

- Line 584: missing space after comma

```
  $query->fields('a',array('gid'));
```

- Line 587: Control statements should have one space between the control keyword and opening parenthesis

```
  if(is_numeric($gid->gid)) {
```

- Line 621: Functions should be called with no spaces between the function name and opening parentheses

```
$options = array (
```

- Line 678: Functions should be called with no spaces between the function name and opening parentheses

```
$options = array (
```

- Line 721: Functions should be called with no spaces between the function name and opening parentheses

```
$options = array (
```

- Line 742: Control statements should have one space between the control keyword and opening parenthesis

```
if($fixed_data !== FALSE) {
```

- Line 743: Control statements should have one space between the control keyword and opening parenthesis

```
  if($fixed_data['if_argument_variable'] != '') $fixed_data['if_argument_variable'] = $xref_array[$fixed_data['if_argument_v
```

- Line 762: Control statements should have one space between the control keyword and opening parenthesis

```
  if($this->_task_data->task_data['if_argument_variable'] == $rec->id) $selected = " selected ";
```

- Line 771: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->id = $_POST['template_data_id'];
```

- Line 773: Control statements should have one space between the control keyword and opening parenthesis

```
if(check_plain($_POST['ifTaskArguments']) == 'status'){
```

- Line 773: use a space between the closing parenthesis and the open bracket

```
if(check_plain($_POST['ifTaskArguments']) == 'status'){
```

- Line 798: Functions should be called with no spaces between the function name and opening parentheses

```
$options = array (
```

- Line 855: missing space after comma

```
$this->_task_data->task_data['handler_location'] = variable_get('maestro_batch_script_location', drupal_get_path('module','m
```

- Line 861: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->id = $_POST['template_data_id'];
```

- Line 885: missing space after comma

```
$batch_function = drupal_get_path('module','maestro') . "/batch/batch_functions.php";
```

- Line 895: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->id = $_POST['template_data_id'];
```

- Line 930: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->id = $_POST['template_data_id'];
```

- Line 953: Control statements should have one space between the control keyword and opening parenthesis

```
if($fixed_data !== FALSE) {
```

- Line 989: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$rec->id = $_POST['template_data_id'];
```

- Line 993: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$task_data[$key . '_value'] = $_POST[$key . '_value'];
```

- Line 995: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$task_data['var_to_set'] = $_POST['var_to_set'];
```

- Line 996: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed t

```
$task_data['set_type'] = $_POST['set_type'];
```

- Line 1006: Control statements should have one space between the control keyword and opening parenthesis

```
if($set_process_variable_methods === FALSE) {
```

- Line 1059: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed

```
$rec->id = $_POST['template_data_id'];
```

- Line 1093: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed

```
$rec->id = $_POST['template_data_id'];
```

- Line 1148: Potential problem: use the Form API to prevent against CSRF attacks. If you need to use $_POST variables, ensure they are fully sanitized if displayed

```
$actions = $_POST['actions'];
```

sites/all/modules/contrib/maestro/maestro_tasks.class.php

maestro_tasks.class.php

- Line -1: @file block missing (Drupal Docs)
- Line 24: Functions should be called with no spaces between the function name and opening parentheses

```
abstract function execute ();
```

- Line 31: Functions should be called with no spaces between the function name and opening parentheses

```
abstract function prepareTask ();
```

- Line 37: use a space between the closing parenthesis and the open bracket

```
function getTaskConsoleURL(){
```

- Line 192: missing space after comma

```
$current_path=variable_get('maestro_batch_script_location',drupal_get_path('module','maestro') . "/batch/");
```

- Line 196: else statements should begin on a new line

```
} elseif (file_exists($this->_properties->handler)) {  // Check in current directory
```

- Line 215: missing space after comma

```
return array('handler' => $taskdata['handler'],'serialized_data' => $serializedData);
```

- Line 227: missing space after comma

```
$success = $function($this->_properties->id,$this->_properties->process_id);
```

- Line 228: else statements should begin on a new line

```
} else {
```

- Line 229: missing space after comma

```
watchdog('maestro',"MaestroTaskTypeBatchFunction - unable to find the function: {$this->_properties->handler}");
```

- Line 246: missing space after comma

```
return array('handler' => $taskdata['handler'],'serialized_data' => $serializedData);
```

- Line 261: missing space after comma

```
$query->addExpression('COUNT(a.id)','templatecount');
```

- Line 262: missing space after comma

```
$query->condition("a.id",$this->_properties->id,"=");
```

- Line 267: missing space after comma

```
$query->addExpression('COUNT(a.id)','processcount');
```

- Line 268: missing space after comma

```
$query->condition(db_and()->condition("a.queue_id",$this->_properties->id,"=")->condition("b.process_id",$this->_properties-
```

- Line 276: else statements should begin on a new line

```
} else {
```

- Line 303: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
if ($templateVariableID == null or $templateVariableID == '' ) { // logical entry it is
```

- Line 306: missing space after comma

```
$query->join('maestro_queue','b','a.from_queue_id=b.id');
```

- Line 308: missing space after comma

- Line 308: missing space after comma

```
$query->condition("a.queue_id", $this->_properties->id,"=");
```

- Line 312: Use uppercase for PHP constants, e.g. NULL, TRUE, FALSE

```
$useTrueBranch = null;
```

- Line 313: in most cases, replace the string function with the drupal_ equivalent string functions

```
switch (strtolower($ifArgumentProcess) ) {
```

- Line 355: missing space after comma

```
$query->fields('a',array('variable_value'));
```

- Line 356: missing space after comma

```
$query->condition(db_and()->condition("a.process_id",$this->_properties->process_id)->condition('a.template_variable_id',$
```

- Line 366: else statements should begin on a new line

```
        } else {
```

- Line 373: else statements should begin on a new line

```
        } else {
```

- Line 380: else statements should begin on a new line

```
        } else {
```

- Line 387: else statements should begin on a new line

```
        } else {
```

- Line 407: Use "elseif" in place of "else if"

```
else if ($useTrueBranch === FALSE) { // point to the false branch
```

- Line 413: else statements should begin on a new line

```
} else {   // We have an unexpected situation - so flag a task error
```

- Line 424: missing space after comma

```
return array('handler' => '' ,'serialized_data' => $serializedData);
```

- Line 449: missing space after comma

```
return array('handler' => $taskdata['handler'],'serialized_data' => $serializedData);
```

- Line 458: missing space after comma

```
  $ret = $taskdata['handler']('display',$this,$taskdata['optional_parm']);
```

- Line 462: else statements should begin on a new line

```
} else {
```

- Line 464: missing space after comma

```
  $retval .= t('Interactive Function "@taskname" was  not found.',array('@taskname' => $taskdata['handler']));
```

- Line 470: missing space after comma

```
function processInteractiveTask($taskid,$taskop) {
```

- Line 478: missing space after comma

```
  $ret = $taskdata['handler']($taskop,$this,$taskdata['optional_parm']);
```

- Line 493: missing space after comma

```
$query->fields('a',array('task_data'));
```

- Line 494: missing space after comma

```
$query->condition('a.id', $this->_properties->template_data_id,'=');
```

- Line 501: missing space after comma

```
    $query->addField('a','variable_value');
```

- Line 502: missing space after comma

```
    $query->condition('a.process_id', $this->_properties->process_id,'=');
```

- Line 503: missing space after comma

```
    $query->condition('a.template_variable_id', $taskDefinitionRec->task_data['var_to_set'],'=');
```

- Line 511: else statements should begin on a new line

```
      } else {
```

- Line 521: missing space after comma

```
    $query->addField('a','variable_value');
```

- Line 522: missing space after comma

```
    $query->condition('a.process_id', $this->_properties->process_id,'=');
```

- Line 523: missing space after comma

```
    $query->condition(db_and()->condition('a.template_variable_id', $taskDefinitionRec->task_data['var_to_set'],'='));
```

- Line 534: Control statements should have one space between the control keyword and opening parenthesis

```
if($set_process_variable_methods === FALSE) {
```

- Line 571: use a space between the closing parenthesis and the open bracket

```
function getTaskConsoleURL(){
```

- Line 576: There should be no trailing spaces

- Line 578: There should be no trailing spaces

- Line 580: There should be no trailing spaces

```
// http://drupal.org/node/1097152
```

- Line 581: missing space after comma

```
$url=str_replace('[site_url]', $base_url,$url);
```

- Line 582: missing space after comma

```
$url=str_replace('[queue_id]', $this->_properties->queue_id,$url);
```

- Line 583: missing space after comma

```
$url=str_replace('[process_id]', $this->_properties->process_id,$url);
```

- Line 592: missing space after comma

```
return array('handler' => $taskdata['handler'],'serialized_data' => $serializedData);
```

- Line 602: Control statements should have one space between the control keyword and opening parenthesis

```
if($this->_properties->status == 1) {
```

- Line 615: use a space between the closing parenthesis and the open bracket

```
function getTaskConsoleURL(){
```

- Line 621: missing space after comma

```
$content_type = str_replace('_','-',$taskdata['content_type']);
```

- Line 629: missing space after comma

```
$query->addField('a','nid');
```

- Line 630: missing space after comma

```
$query->condition('a.instance', 1,'=');
```

- Line 631: missing space after comma

```
$query->condition('a.tracking_id', $tracking_id,'=');
```

- Line 632: missing space after comma

```
$query->condition(db_and()->condition('a.content_type', $taskdata['content_type'],'='));
```

- Line 637: missing space after comma

```
  $query->addField('a','nid');
```

- Line 638: missing space after comma

```
  $query->condition('a.instance', 1,'=');
```

- Line 639: missing space after comma

```
  $query->condition('a.tracking_id', $tracking_id,'=');
```

- Line 640: missing space after comma

```
  $query->condition(db_and()->condition('a.content_type', $taskdata['content_type'],'='));
```

- Line 648: else statements should begin on a new line

```
} else {
```

- Line 667: missing space after comma

```
function processContent($taskid,$op,$object) {
```

- Line 668: missing space after comma

```
watchdog('maestro',"processContent function");
```

- Line 671: missing space after comma

```
->fields('maestro_queue', array('process_id','template_data_id'))
```

- Line 677: else statements should begin on a new line

```
} else {
```

- Line 705: missing space after comma

```
function showContentDetail($tracking_id,$task_id) {
```

- Line 709: missing space after comma

```
$query = db_select('maestro_project_content','content');
```

- Line 710: missing space after comma

```
$query->addField('content','nid');
```

- Line 711: missing space after comma

```
$query->addField('content','status');
```

- Line 712: missing space after comma

```
$query->condition('content.tracking_id',$tracking_id,'=');
```

- Line 713: missing space after comma

```
$query->condition('content.task_id',$task_id,'=');
```