**Retrieving a cell by coordinate**

To retrieve the value of a cell, the cell should first be retrieved from the worksheet using the getCell method. A cell’s value can be read again using the following line of code:

$objPHPExcel->getActiveSheet()->getCell('B8')->getValue();

Section 4.5.4 of the developer documentation

**Retrieving a cell by column and row**

To retrieve the value of a cell, the cell should first be retrieved from the worksheet using the getCellByColumnAndRow method. A cell’s value can be read again using the following line of code:

// Get cell B8

$objPHPExcel->getActiveSheet()->getCellByColumnAndRow(1, 8)->getValue();

If you need the calculated value of a cell, use the following code. This is further explained in 4.4.35

// Get cell B8

$objPHPExcel->getActiveSheet()->getCellByColumnAndRow(1, 8)->getCalculatedValue();

[**Getting Cell as String in PHPExcel by column and row**](http://stackoverflow.com/questions/27708459/getting-cell-as-string-in-phpexcel-by-column-and-row)**:**

$keyCell = $sheet->getCellByColumnAndRow(1,5)->getValue();

Will read the cell data in the format that it's actually stored by Excel, you can't arbitrarily change that or tell PHPExcel to read it as a different datatype.

However, if the cell has formatting applied, then you can use

$keyCell = $sheet->getCellByColumnAndRow(1,5)->getFormattedValue();

instead, and this will return the data as a string, with whatever format mask was defined in the Excel spreadsheet

**CellCollections or Array of cell or cell integrator**

<?php

$objReader =PHPExcel\_IOFactory::createReader('Excel2007');

$objReader->setReadDataOnly(true);

$objPHPExcel = $objReader->load("test.xlsx");

$objWorksheet = $objPHPExcel->getActiveSheet();

echo '<table>'."\n";

foreach($objWorksheet->getRowIterator()as $row){

echo '<tr>'."\n";

$cellIterator = $row->getCellIterator();

$cellIterator->setIterateOnlyExistingCells(false);// This loops all cells,

// even if it is not set.

// By default, only cells

// that are set will be

// iterated.

foreach($cellIterator as $cell){

echo '<td>'. $cell->getValue().'</td>'."\n";

}

echo '</tr>'."\n";

}

echo '</table>'."\n";

?>

**Another Example:**

foreach($cell\_collection as $cell)

{

$column = $objPHPExcel->getActiveSheet()->getCell($cell)->getColumn();

$row = $objPHPExcel->getActiveSheet()->getCell($cell)->getRow();

$data\_value = $objPHPExcel->getActiveSheet()->getCell($cell)->getValue();

if($row ==1)

{

$header[$row][$column]= $data\_value;

}

else

{

$arr\_data[$row][$column]= $data\_value;

}

}

**Last or Highest Row/Column**

There is no ColumnIterator, so you'll have to do this by hand.

To loop rows for a column:

$column = 'A';

$lastRow = $worksheet->getHighestRow();

for ($row = 1; $row <= $lastRow; $row++) {

$cell = $worksheet->getCell($column.$row);

// Do what you want with the cell

}

To loop columns in a row, you can take advantage of PHP's Perls-style ability to increment characters:

$row = 1;

$lastColumn = $worksheet->getHighestColumn();

$lastColumn++;

for ($column = 'A'; $column != $lastColumn; $column++) {

$cell = $worksheet->getCell($column.$row);

//$worksheet->cellExists($column.$row);

// Do what you want with the cell

}

Note that when comparing column letters to test for the last column in the loop, we can't simply use < or <= because we're comparing strings, and "B" > "AZ" in standard string comparison, so we use a != comparison, having incremented the highest column value to give the first column ID past the end point.

You can also use

$worksheet->cellExists($column.$row);

in the loop to test for the existence of a cell before accessing it using getCell() (or not) to emulate the iterator getIterateOnlyExistingCells() behaviour

The iterators are actually fairly slow, so you may well find these simple loops are faster than using the iterators.

# [How do I get a range of cells in PHPExcel?](http://stackoverflow.com/questions/18203595/how-do-i-get-a-range-of-cells-in-phpexcel)

There is, the [rangeToArray()](http://hitautodestruct.github.io/PHPExcelAPIDocs/classes/PHPExcel_Worksheet.html#method_rangeToArray) method:

$objPHPExcel->setActiveSheetIndex(0)->rangeToArray('A1:C3');

Wondering why I bother documenting these methods in the first place, but here's the argument list as well.

Push Cell Data in an Array

$price = UPLOAD\_DIR . $price;

$file\_type =PHPExcel\_IOFactory::identify($price);

$filter =newShipperReadFilter();

$filter->getShipperSettings($shipper);

$reader =PHPExcel\_IOFactory::createReader($file\_type)->setReadDataOnly(true)->setReadFilter($filter)->load($price);

$worksheet = $reader->getActiveSheet();

$col\_number =PHPExcel\_Cell::columnIndexFromString($worksheet->getHighestDataColumn());

$data = array();

foreach($worksheet->getRowIterator($filter->getStartRow())as $row){

$cellIterator = $row->getCellIterator();

$item = array();

foreach($cellIterator as $cell){

// May be PROBLEM is Here?

array\_push($item, $cell->getCalculatedValue());

}

array\_push($data, $item);

}

var\_dump($data);

**Answer:**

By default, the cell iterator will only iterate over cells that actually exist

$cellIterator = $row->getCellIterator();

$cellIterator->setIterateOnlyExistingCells(false);

$item = array();

foreach($cellIterator as $cell){

array\_push($item, $cell->getCalculatedValue());

}

However, if all you're doing is building an array of calculated values for each row, then you might find that using the worksheet's rangeToArray() method is more efficient.

**EDIT**

To get only cells matching your read filter column list:

foreach($cellIterator as $cell){

$item[$cell->getColumn()]= $cell->getCalculatedValue();

}

$item = array\_intersect\_key($item, array\_flip($filter->valid\_columns));

or

foreach($cellIterator as $cell){

if(in\_array($cell->getColumn(), $filter->valid\_columns)){

array\_push($item, $cell->getCalculatedValue());

}

}

# [Get Cell Value with column name in Php Excel](http://stackoverflow.com/questions/32526534/get-cell-value-with-column-name-in-php-excel)

There's nothing magic in PHPExcel, but it's pretty easy to do in standard PHP

Retrieve the headings/keys that you want from the first row

$headings = $sheet->rangeToArray('A1:'. $highestColumn .1,

NULL,

TRUE,

FALSE);

Then reset the standard numeric keys to be the headings values inside your reading loop for each data row

for($row =2; $row <= $highestRow; $row++){

// Read a row of data into an array

$rowData = $sheet->rangeToArray('A'. $row .':'. $highestColumn . $row,

NULL,

TRUE,

FALSE);

$rowData[0]= array\_combine($headings[0], $rowData[0]);

}

# [Just get one row from PHPExcel](http://stackoverflow.com/questions/17951067/just-get-one-row-from-phpexcel)

$row = $this->objPHPExcel->getActiveSheet()->getRowIterator($rownumber)->current();

$cellIterator = $row->getCellIterator();

$cellIterator->setIterateOnlyExistingCells(false);

foreach ($cellIterator as $cell) {

echo $cell->getValue();

}

I don't remember if you are supposed to use next or current. If you are getting the wrong row, use current() instead of next().

Another Example:

$row = $objPHPExcel->getActiveSheet()

->getRowIterator($searchValue)->current();

$cellIterator = $row->getCellIterator();

$cellIterator->setIterateOnlyExistingCells(false);

foreach ($cellIterator as $cell) {

echo 'ROW: ', $cell->getRow(), PHP\_EOL;

echo 'COLUMN: ', $cell->getColumn(), PHP\_EOL;

echo 'COORDINATE: ', $cell->getCoordinate(), PHP\_EOL;

echo 'RAW VALUE: ', $cell->getValue(), PHP\_EOL;

echo $cell->getCalculatedValue(),PHP\_EOL; // Value here

}

Getting Row information

foreach($worksheet->getRowIterator() as $row){

$rowIndex = $row->getRowIndex();

$cell = $worksheet->getCell('A' . $rowIndex);

echo $cell->getCalculatedValue();

$cell = $worksheet->getCell('B' . $rowIndex);

echo $cell->getCalculatedValue();

}

Character-based column address starting with "A"

// Get worksheet dimensions

$sheet = $objPHPExcel->getSheet(0);

$highestRow = $sheet->getHighestRow();

$highestColumn = $sheet->getHighestColumn();

$highestColumn++;

for ($row = 1; $row <= $highestRow; $row++){

for($col = 'A'; $col != $highestColumn; $col++){

$cell = $sheet->getCellByColumnAndRow($col, $row);

$colIndex = PHPExcel\_Cell::columnIndexFromString($cell->getColumn());

echo $colIndex;

}

}

# [How to get row by cell value in PHPExcel](http://stackoverflow.com/questions/38611108/how-to-get-row-by-cell-value-in-phpexcel)

for ($row = 1; $cell = $list->getCellByColumnAndRow($desired\_column, $row) != ""; $row++)

{

if ($user\_input == $cell->getValue());

{

echo $row;

}

}

**How to style fields**

$header = 'a1:h1';

$ews->getStyle($header)->getFill()->setFillType(\PHPExcel\_Style\_Fill::FILL\_SOLID)->getStartColor()->setARGB('00ffff00');

$style = array(

'font' => array('bold' => true,),

'alignment' => array('horizontal' => \PHPExcel\_Style\_Alignment::HORIZONTAL\_CENTER,),

);

$ews->getStyle($header)->applyFromArray($style);

# [Set Font Color, Font Face and Font Size in PHPExcel](http://stackoverflow.com/questions/17317301/set-font-color-font-face-and-font-size-in-phpexcel)

$phpExcel = newPHPExcel();

$styleArray = array(

'font' => array(

'bold' =>true,

'color' => array('rgb' =>'FF0000'),

'size' =>15,

'name' =>'Verdana'

));

$phpExcel->getActiveSheet()->getCell('A1')->setValue('Some text');

$phpExcel->getActiveSheet()->getStyle('A1')->applyFromArray($styleArray);

Another Example:

function cellColor($cells,$color){

global $objPHPExcel;

$objPHPExcel->getActiveSheet()->getStyle($cells)->getFill()->applyFromArray(array(

'type' => PHPExcel\_Style\_Fill::FILL\_SOLID,

'startcolor' => array(

'rgb' => $color

)

));

}

cellColor('B5', 'F28A8C');

cellColor('G5', 'F28A8C');

cellColor('A7:I7', 'F28A8C');

cellColor('A17:I17', 'F28A8C');

cellColor('A30:Z30', 'F28A8C');

**Associative Array**

$cars = array  
  (  
  array("Volvo",22,18),  
  array("BMW",15,13),  
  array("Saab",5,2),  
  array("Land Rover",17,15)  
  );

**Another example:**

<?php  
$cars = array  
  (  
  array("Volvo",22,18),  
  array("BMW",15,13),  
  array("Saab",5,2),  
  array("Land Rover",17,15)  
  );  
    
echo $cars[0][0].": In stock: ".$cars[0][1].", sold: ".$cars[0][2].".<br>";  
echo $cars[1][0].": In stock: ".$cars[1][1].", sold: ".$cars[1][2].".<br>";  
echo $cars[2][0].": In stock: ".$cars[2][1].", sold: ".$cars[2][2].".<br>";  
echo $cars[3][0].": In stock: ".$cars[3][1].", sold: ".$cars[3][2].".<br>";  
?>

**Another example:**

<?php  
$cars = array  
  (  
  array("Volvo",22,18),  
  array("BMW",15,13),  
  array("Saab",5,2),  
  array("Land Rover",17,15)  
  );  
      
for ($row = 0; $row < 4; $row++) {  
  echo "<p><b>Row number $row</b></p>";  
  echo "<ul>";  
  for ($col = 0; $col < 3; $col++) {  
    echo "<li>".$cars[$row][$col]."</li>";  
  }  
  echo "</ul>";  
}  
?>

Anothe example:

foreach ($info as $name => $locations) {

foreach ($locations as $location) {

echo "Area {$name} and person located {$location}<br />";

}

}

**Another Example:**

publicstaticvoid main(String[] args) {

int twoDm[][]= newint[7][5];

int i,j,k=1;

for(i=0;i<7;i++) {

for(j=0;j<5;j++) {

twoDm[i][j]=k;

k++;

}

}

for(int[] row : twoDm) {

printRow(row);

}

}

**Row and column Select:**

$objWorksheet->getActiveSheet()->getRowDimension('1')->setRowHeight(40);

$objWorksheet->getActiveSheet()->getColumnDimension('A')->setWidth(100);

or define auto-size:

$objWorksheet->getRowDimension('1')->setRowHeight(-1);

**Get Style by Column and Row**

->getStyle('A1:E1')

Remove A Row:

$objPHPExcel->getActiveSheet()->removeRow($row);

$objPHPExcel->getActiveSheet()->removeRow($row,$row);

Remove a Column:

$objPHPExcel->getActiveSheet()->removeColumn('F');

You use the Worksheet object's removeColumn() or removeColumnByIndex() methods.

removeColumn() accepts a column id by name (e.g. D, F, L).

removeColumnByIndex() accepts a column id by its index number (e.g. 3, 6, 12)

**Column Index:**

$colIndex = PHPExcel\_Cell::columnIndexFromString($cell->getColumn());

**Save file in specific folder**

$name = '/path/to/folder/xyz.xlsx';

$objWriter->save($name);

**Save as pdf**

**\\**DomPDF, TcPDF or mPDF

<?php

set\_include\_path('Classes/');

require\_once 'PHPExcel/IOFactory.php';

require\_once 'PHPExcel.php';

$rendererName = PHPExcel\_Settings::PDF\_RENDERER\_DOMPDF;

$rendererLibrary = 'domPDF0.6.0beta3';

$rendererLibraryPath = dirname(\_\_FILE\_\_). 'libs/classes/dompdf' . $rendererLibrary;

// Here's the magic: you \_\_tell\_\_ PHPExcel what rendering engine to use

// and where the library is located in your filesystem

if (!PHPExcel\_Settings::setPdfRenderer(

$rendererName,

$rendererLibraryPath

)) {

die('NOTICE: Please set the $rendererName and $rendererLibraryPath values' .

'<br />' .

'at the top of this script as appropriate for your directory structure'

);

}

$objPHPExcel = new PHPExcel();

$objReader = new PHPExcel\_Reader\_Excel5();

$objPHPExcel = $objReader->load('document.xls');

$rowIterator = $objPHPExcel->getActiveSheet()->getRowIterator();

$sheet = $objPHPExcel->getActiveSheet();

$sheet->setCellValue('A1', 'řčřč');

$sheet->setCellValue('B1', 'B');

$sheet->setCellValue('C1', 'C');

$objWriter = PHPExcel\_IOFactory::createWriter($objPHPExcel, 'PDF');

$objWriter->save('document.pdf');

?>