

GridView Widget

GridView Widget

Make item clickable link

```
[
  'attribute'=>'naam',
  'label' => 'Studentnaam',
  'contentOptions' => ['style' => 'width:600px; white-space: normal;'],
  'format' => 'raw',
  'value' => function ($data) {
    return Html::a($data->naam, ['/examen/update?id='.$data->id],['title'=> 'Edit',]);
  },
],

// for related table values, use
'value' => 'student.locatie',
```

Title of column name

Change the model, example:

```
public function attributeLabels() {
    return [
        'id' => 'ID',
        'naam' => 'examennaam',
        'datum_van' => 'van',
        'datum_tot' => 'tot',
        'actief' => 'actief',
    ];
}
```

The database column '*datum_van*' will be showed as '*van*'.

View column conditionally

```
[
    'class' => 'yii\grid\ActionColumn',
    'contentOptions' => ['style' => 'width:400px; white-space: normal;'],
    'header'=>'Actions',
    'template' => '{view} {update} {delete}',
    'visibleButtons'=>[
        'view'=> function($model){
            return $model->status!=1;
        },
    ]
],
```

Add own controller (copy)

```
[
    'class' => 'yii\grid\ActionColumn',
    'template' => '{download} {view} {update} {delete}',
    'buttons' => [
        'download' => function ($url) {
            return Html::a(
                '<span class="glyphicon glyphicon-arrow-down"></span>',
                ['another-controller/another-action', 'id' => $model->id], // mag ook alleen url als je geen parameter
                wilt meegeven, dan hoef je geen array te maken.
            [
                'title' => 'Download',
                'data-pjax' => '0',
            ]
        );
    },
],
```

// Alternatief, copy button, niet vergeten in template te te zetten

```
'buttons'=>[
    'copy' => function ($url, $model, $key) {
        return Html::a('<span class="glyphicon glyphicon-copy"></span>', ['copy', 'id'=>$model->id,['title'=>'Copy']]);
    },
],
```

Boolean (toggle icon)

Show boolean, if clicked, change status.

```
[
    'attribute'=>'actief',
    'contentOptions' => ['style' => 'width:10px;'],
    'format' => 'raw',
    'value' => function ($data) {
        // $status = icon 'ok' or icon 'minus' depending on $data->ctief
        $status = $data->actief ? '<span class="glyphicon glyphicon-ok"></span>' : '<span class="glyphicon glyphicon-minus"></span>';
        return Html::a($status, 'rolspeler/toggle-actief?id='.$data->id);
    }
],

// Controller
public function actionToggleActief($id) {
    // function toggles boolean actief, in this example only one row can be active, so we execute two queries
    $sql="update examen set actief=1 where id = :id; update examen set actief=0 where id != :id;";
    $params = array(':id'=> $id);
    Yii::$app->db->createCommand($sql)->bindValues($params)->execute();
    return $this->redirect(['index']); // go back to view
}
```

In this example only one row can be active. If you want a simple toggle use SQL statement:

```
update <table> set abss(active = active-1)
```

Drop down in grid select

<https://stackoverflow.com/questions/29231013/how-can-i-use-a-simple-dropdown-list-in-the-search-box-of-gridviewwidget-yii2>

```
[
    'attribute'=>'attribute name',
    'filter'=>array("ID1"=>"Name1","ID2"=>"Name2"),
],

// of
```

```
[
    'attribute'=>'attribute name',
    'filter'=>ArrayHelper::map(Model::find()->asArray()->all(), 'ID', 'Name'),
],

// of

[
    'attribute' => 'attribute_name',
    'value' => 'attribute_value',
    'filter' => Html::activeDropDownList(
        $searchModel,
        'attribute_name',
        ArrayHelper::map(ModelName::find()->asArray()->all(), 'ID', 'Name'),
        ['class'=>'form-control','prompt' => 'Select Category']
    ),
],
```

In dit voorbeeld wordt er data uit een andere table/model gehaald. De foreign key moet worden ge-update aan de hand van de rolspeleers table.

```
<?php

// *****

// * in controller *
// *****

// pass rolspeeler

$rolspeeler = Rolspeeler::find()->where(['actief' => '1'])->all();
return $this->render('index', [
    'searchModel' => $searchModel,
    'dataProvider' => $dataProvider,
    'rolspeeler' => $rolspeeler,
]);

// function for update (ajax) call from form

public function actionUpdateStatus($id, $status, $rolspeelerid) {
    $model = $this->findModel($id);
```

```
$model->status=$status;
$model->rolspelerid=$rolspelerid;
$model->save();
}
```

```
// *****
```

```
// * in view (index) *
```

```
// *****
```

```
// ajax call to update record
```

```
<script>
```

```
function changeStatus(id, status, rolspelerid) {
    // console.log(val, id);
    $.ajax({
        url: "<?= Url::to(['update-status']) ?>",
        data: {id: id, 'status': status, 'rolspelerid': rolspelerid },
        cache: false
    }).done(function (html) {
        location.reload();
    });
}
```

```
</script>
```

```
// prepare ass. array for the drop down.
```

```
<?php $rolspelerList = ArrayHelper::map($rolspeler,'id','naam'); ?>
```

```
...
```

```
<?= GridView::widget([
```

```
    'dataProvider' => $dataProvider,
    'filterModel' => $searchModel,
    'columns' => [
```

```
        // show dropdown and call ajax script that calls the methos/function in teh controller
```

```
        'format' => 'raw',
        'value' => function ($model) {
```

```
return Html::dropDownList('status', $model->status, ['0'=>'Wachten','1'=>'Loopt','2'=>'Klaar'],  
    ['onchange' => "changeStatus('$model->id', $(this).val(), '$model->rolspelerid')"]);  
    }],  
    ...
```

From: <https://www.trickyhints.com/how-to-create-dropdown-in-yii2-gridview/>

Column customization

In Yii2's GridView, each column in the grid can be customized using a variety of properties. These properties allow you to define how each column behaves, how it's rendered, and how it interacts with the data provided. Here's a list of common properties that you can specify in a column in Yii2 GridView:

- **attribute****: The attribute name of the data model. The value of the attribute will be displayed in this column.
- **label****: The label of the column in the header.
- **value****: A PHP callable that returns the content of the column. It should be used if the column content is not directly tied to a model attribute.
- **format****: How the data of this column will be formatted. Common formats include 'text', 'html', 'date', 'number', etc.
- **header****: The header cell content. This will override the automatically generated header cell content.
- **footer****: The footer cell content. This will override the automatically generated footer cell content.
- **headerOptions****: HTML attributes for the header cell tag.
- **contentOptions****: HTML attributes for the data cell tag.
- **footerOptions****: HTML attributes for the footer cell tag.
- **visible****: Whether the column is visible. If set to false, the column will not be displayed.
- **filter****: The filter input content. This can be a string or an array to specify the type of filter input.
- **filterOptions****: HTML attributes for the filter cell tag.
- **headerHtmlOptions****: Deprecated, use `headerOptions` instead.
- **footerHtmlOptions****: Deprecated, use `footerOptions` instead.
- **class****: The class name of the column. Yii2 provides several built-in column types like `yii\grid\SerialColumn`, `yii\grid\DataColumn`, `yii\grid\CheckboxColumn`, etc.
- **enableSorting****: Whether sorting is enabled for this column.
- **sortLinkOptions****: HTML attributes for the sorting link.
- **filterInputOptions****: HTML attributes for the filter input.
- **filterWidgetOptions****: Configuration options for the filter widget.
- **noWrap****: Whether to disable text wrapping for this column.

Revision #18

Created 22 July 2020 21:08:45 by Max

Updated 2 February 2024 19:36:28 by Max