



YII FRAMEWORK 2.0

The Fast, Secure & Professional Framework

#3

Gii, Pjax & Layout

Let's start it!



Outline

- Gii: Code Generator
 - konfigurasi
 - cara penggunaan
 - meng-generate operasi CRUD
- Ajax & Pjax
 - Asset
 - Ajax
 - Pjax
- Layout



Gii: Code Generator

Yii mempunyai *tools* yang berfungsi untuk meng-*generate* kode program bernama Gii

Gii: Code Generator

Gii sangat membantu para *developer* untuk mempercepat pekerjaannya terutama untuk hal-hal yang umum dan sering dilakukan, salah satunya adalah aplikasi CRUD.

CRUD mempunyai pola kode yang sama sehingga bisa dibuat *tools* untuk meng-*generate* kodenya.

Konfigurasi Gii

Gii merupakan *extension* dalam bentuk *module* yang secara *default* telah terinstal dan siap digunakan ketika menginstal Yii.

Letak konfigurasi Gii:

 @app/config/web.php

```
if (YII_ENV_DEV) {  
    // configuration adjustments for 'dev' environment  
    $config['bootstrap'][] = 'debug';  
    $config['modules']['debug'] = [  
        'class' => 'yii\debug\Module',  
    ];  
  
    $config['bootstrap'][] = 'gii';  
    $config['modules']['gii'] = [  
        'class' => 'yii\gii\Module',  
    ];  
}
```

Konfigurasi Gii (cont)

Mengubah mode dari *development* menjadi *production*, melalui file *entry script*

 @app/web/index.php

```
<?php

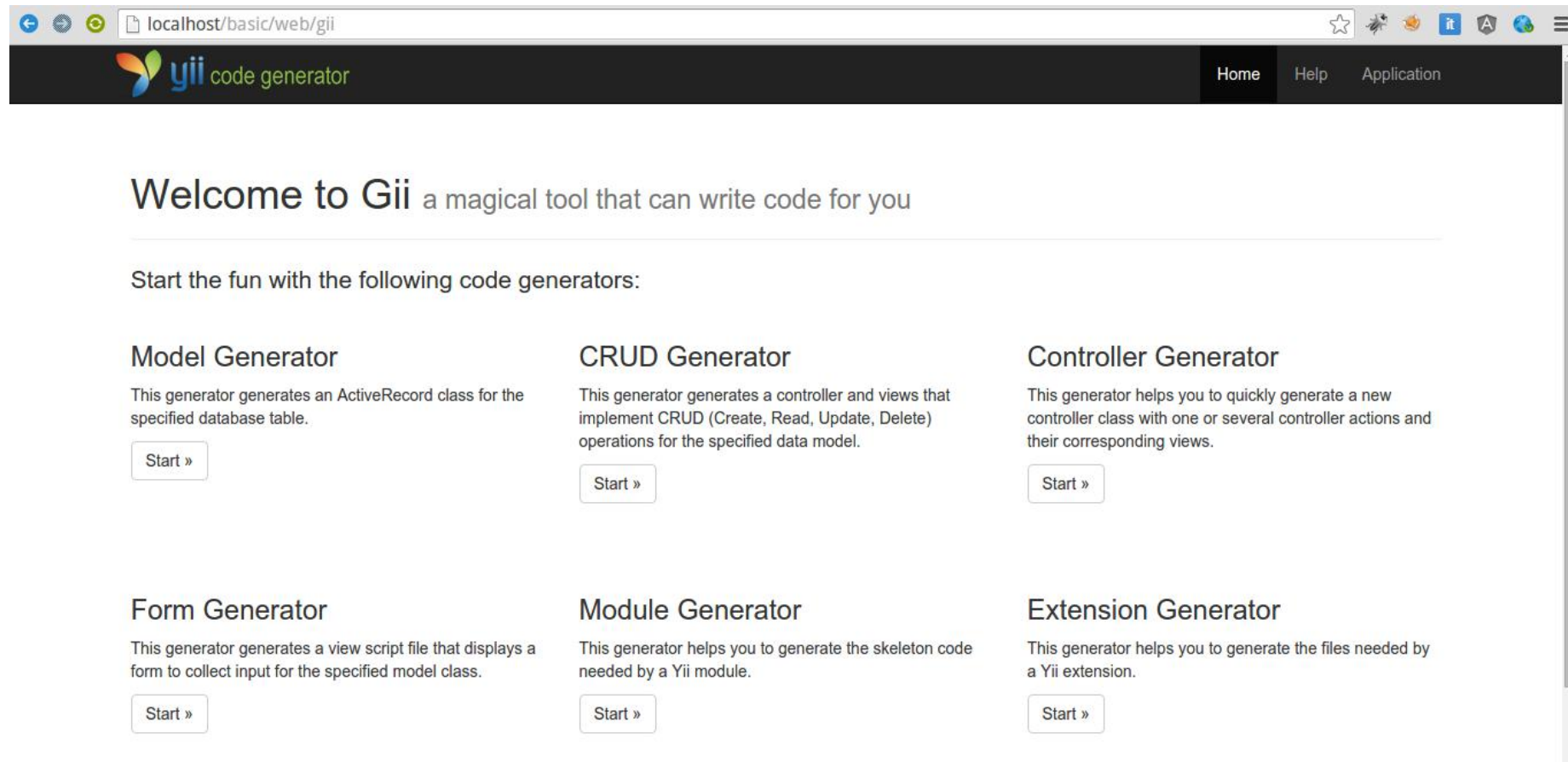
// comment out the following two lines when deployed to production
defined('YII_DEBUG') or define('YII_DEBUG', true);
defined('YII_ENV') or define('YII_ENV', 'dev');
```

Cukup dengan meng-*comment* dua baris kode di atas maka Yii berubah menjadi mode *production*

Cara Penggunaan

Akses *tools* Gii melalui *browser* melalui URL berikut

```
http://localhost/basic/web/gii
```



Persiapan



Sebelum menggunakan *generator*, kita buat dulu satu tabel untuk latihan dengan nama *category*

```
CREATE TABLE category (  
  id int primary key auto_increment,  
  title varchar(255) NOT NULL,  
  description varchar(255) NOT NULL,  
  created_at int,  
  updated_at int,  
  created_by int,  
  updated_by int  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Meng-*generate Model Active Record*

Generator ini akan meng-*generate class model Active Record* berdasarkan tabel tertentu pada *database*

Welcome to Gii a magical

Start the fun with the following code ge

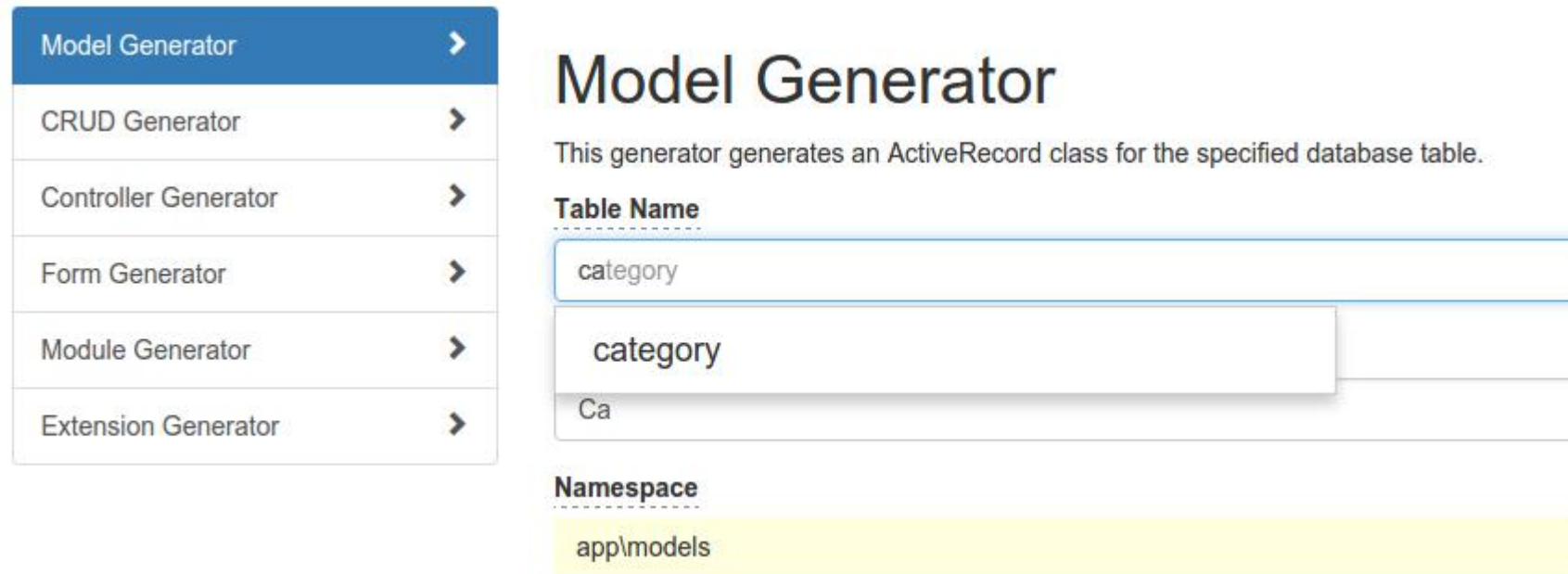
Model Generator

This generator generates an ActiveRecord class for the specified database table.

Start »

Meng-*generate* Model Active Record

Isi *form* dengan memasukkan nama tabel pada kolom *Table Name*



The screenshot shows a web interface for a 'Model Generator'. On the left is a sidebar menu with the following items: 'Model Generator' (highlighted in blue), 'CRUD Generator', 'Controller Generator', 'Form Generator', 'Module Generator', and 'Extension Generator'. Each item has a right-pointing chevron. The main content area is titled 'Model Generator' and contains the text: 'This generator generates an ActiveRecord class for the specified database table.' Below this text are two input fields. The first field is labeled 'Table Name' and contains the text 'category'. A dropdown menu is open below this field, showing three options: 'category', 'Ca', and 'Ca'. The second field is labeled 'Namespace' and contains the text 'app\models', which is highlighted with a yellow background.

Model Generator	>
CRUD Generator	>
Controller Generator	>
Form Generator	>
Module Generator	>
Extension Generator	>

Model Generator

This generator generates an ActiveRecord class for the specified database table.

Table Name

category

Ca

Namespace

Kalau sudah diisi form nya klik tombol *Preview*.
Setelah itu klik tombol *Generate*

Meng-*generate Model Active Record*

Kalau kita lihat hasil *generate file model* Category, khususnya pada fungsi *rules*, maka kita dapati bahwa tipe data yang kita set pada tabel dibaca sebagai fungsi *rules*.

```
public function rules()
{
    return [
        [['title', 'description'], 'required'],
        [['created_at', 'updated_at', 'created_by', 'updated_by'], 'integer'],
        [['title', 'description'], 'string', 'max' => 255],
    ];
}
```

Meng-*generate* Model Active Record

Disamping itu ada cara yang lebih mudah lagi dalam meng-*generate* model yaitu kita diizinkan untuk meng-*generate model* dari semua tabel sekaligus. Caranya isi *field table name* dengan karakter bintang *

Model Generator

This generator generates an ActiveRecord class for the specified database table.

Table Name

*

Namespace

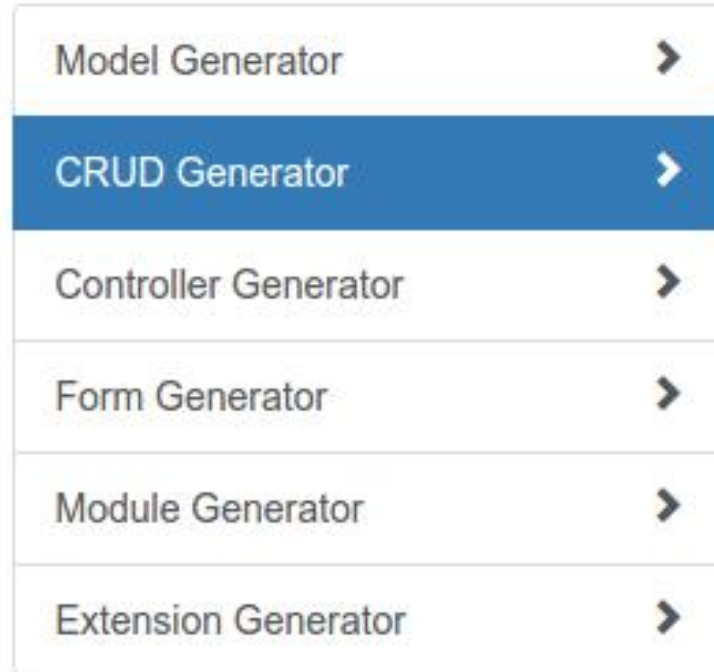
app\models

Base Class

yii\db\ActiveRecord

Meng-*generate* Operasi CRUD

Pada pertemuan sebelumnya, kita telah belajar tentang bagaimana caranya membuat aplikasi CRUD. Ternyata di Gii juga mempunyai fitur untuk meng-*generate* kode CRUD. Pilih menu *CRUD Generator*



Meng-*generate* Operasi CRUD

Pada *form* CRUD Generator isilah *field-field*nya. Setelah itu *Preview* & *Generete*

CRUD Generator

This generator generates a controller and views model.

Model Class

Search Model Class

Controller Class

View Path

Ujicoba Hasil Generator

Setelah meng-*generate* CRUD menggunakan Gii, maka sekarang saatnya mengujicoba hasilnya melalui *web browser*.

```
http://localhost/basic/web/category
```


Modifikasi Kode Hasil Generator

Hasil *generate* CRUD masih berupa kode standar, untuk itu kita perlu melakukan beberapa modifikasi sesuai dengan keinginan kita.

Misalnya:

- *Field* yang ditampilkan pada *Gridview* hanya *Title*, *Description*, *created_at*, dan *created_by*
- *Field* *created_at*, *updated_at*, *created_by*, *updated_by* tidak ditampilkan pada form. Namun diisi oleh sistem karena nilainya sudah bisa dipastikan.
- *Field* *created_at* ditampilkan dalam format yang mudah dibaca oleh *user* yaitu "tgl-bulan-tahun jam:menit"



Modifikasi Kode Hasil Generator

Untuk melakukan modifikasi berikut langkah-langkahnya:

1. Modifikasi file  @app/views/category/index.php
Uncomment field yang ingin ditampilkan


```
<?= GridView::widget([
    'dataProvider' => $dataProvider,
    'filterModel' => $searchModel,
    'columns' => [
        ['class' => 'yii\grid\SerialColumn'],

        'id',
        'title',
        'description',
        'created_at',
        //'updated_at',
        'created_by',
        //'updated_by',

        ['class' => 'yii\grid\ActionColumn'],
    ],
]); ?>
```

Modifikasi Kode Hasil Generator



2. Modifikasi file  @app/views/category/_form.php
Hapus *field* created_at, updated_at, created_by, updated_by

```
<?php $form = ActiveForm::begin(); ?>

<?= $form->field($model, 'title')->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'description')->textInput(['maxlength' => true]) ?>

<div class="form-group">
    <?= Html::submitButton($model->isNewRecord ? 'Create' : 'Update', ['class'
=> $model->isNewRecord ? 'btn btn-success' : 'btn btn-primary']) ?>
</div>

<?php ActiveForm::end(); ?>
```

Modifikasi Kode Hasil Generator

3. Supaya keempat *field* yang dihapus bisa diisi secara otomatis oleh sistem maka kita perlu memodifikasi *model Category* yaitu dengan menambahkan fungsi untuk mengisi keempat *field* tersebut sebelum atau setelah proses *create* atau *update* data.

Yii menyebut fungsi yang seperti ini sebagai *Behavior*.
Ada dua *behavior* yang berperan untuk kasus ini yaitu:

- *Class yii\behaviors\TimestampBehavior*, dan
- *Class yii\behaviors\BlameableBehavior*

Timestamp & Blameable Behavior

Timestamp behavior berfungsi menginput data (waktu saat proses create/update) secara otomatis

Blameable behavior berfungsi menginput data *current user* secara otomatis ketika proses *create* atau *update* data

Modifikasi Kode Hasil Generator



3. Modifikasi file @app/models/Category.php

```
public function behaviors() {  
    return [  
        TimestampBehavior::className(),  
        BlameableBehavior::className()  
    ];  
}
```

Jangan lupa tambahkan *use* dua Class tersebut di awal *file*

```
use Yii;  
use yii\behaviors\TimestampBehavior;  
use yii\behaviors\BlameableBehavior;
```

Testing dengan login terlebih dahulu!



Ajax & Pjax

Asynchronous Javascript and XML - Push State + Ajax

Asset

Pemahaman tentang *Asset* ini penting sebelum kita belajar tentang *Ajax* dan *Pjax*.

Pada Yii, *Asset* adalah file yang dapat dirujuk dalam halaman web, contoh: file CSS, Javascript, gambar, atau video dll.

Kita bisa menggunakan *asset* yang berbeda untuk suatu *view* yang berbeda. Kita juga bisa mengelompokkan antara *asset* yang utama atau menjadi *base* dari aplikasi kita.

Manajemen *asset* di Yii menggunakan *class Asset Bundles*.

Asset Bundles

Berikut ini contoh kode dasar dari penerapan *class Asset Bundle* pada template basic



@app/assets/AppAsset.php

```
namespace app\assets;

use yii\web\AssetBundle;

class AppAsset extends AssetBundle
{
    public $basePath = '@webroot';
    public $baseUrl = '@web';
    public $css = [
        'css/site.css',
    ];
    public $js = [
    ];
    public $depends = [
        'yii\web\YiiAsset',
        'yii\bootstrap\BootstrapAsset',
    ];
}
```

Javascript & CSS tanpa *Asset Bundles*

Disamping menggunakan *Asset Bundles*, kita juga bisa menggunakan kode javascript/JS dan CSS tanpa melalui *Asset Bundles*, melainkan secara langsung pada *view*-nya.

- registerJs()
- registerCss()
- registerJsFile()
- registerCssFile()

Ajax

Ajax adalah singkatan dari *asynchronous Javascript and XML*, merupakan istilah yang populer dalam pemrograman web. Umumnya, penggunaan Ajax adalah untuk *me-load* suatu bagian halaman tanpa *me-refresh* keseluruhan halaman.

Ajax bekerja dengan menggunakan *engine* XMLHttpRequest, merupakan *engine* yang disematkan secara *default* pada browser dan memiliki kemampuan *me-load* data dari *server* secara *asynchronous*.

by default, Yii telah menyertakan librari JQuery yaitu librari Javascript yang memudahkan kita dalam mengimplementasikan Ajax pada aplikasi kita.



Implementasi

Mari kita buat *controller* baru untuk latihan *dependent dropdown* pada direktori @app/controllers/

 @app/controllers/AjaxController.php

```
<?php
namespace app\controllers;
class AjaxController extends \yii\web\Controller
{

}
```



Dropdown ke Textbox

Pada bagian ini, kita akan mensimulasikan sebuah *dropdown* yang berhubungan dengan beberapa *textbox*



@app/controllers/AjaxController.php

```
public function getBooks()
{
    $books = [
        ['id'=>'1', 'title'=>'Pemrograman PHP', 'author'=>'Hafid', 'year'=>'2015'],
        ['id'=>'2', 'title'=>'Pemrograman JS', 'author'=>'Juned', 'year'=>'2014'],
        ['id'=>'3', 'title'=>'Database MySQL', 'author'=>'Lily', 'year'=>'2013'],
    ];
    // $books = Book::find()->asArray()->all();
    return $books;
}
```



Dropdown ke Textbox



@app/controllers/AjaxController.php

```
public function actionBook()
{
    $model = new \yii\base\DynamicModel([
        'title', 'author', 'year'
    ]);
    $model->addRule(['title'], 'string');
    $model->addRule(['description'], 'string');
    $model->addRule(['year'], 'integer');

    return $this->render('book', [
        'model' => $model,
        'books' => $this->getBooks(),
    ]);
}
```



Dropdown ke Textbox



@app/views/ajax/book.php

```
<?php
use yii\widgets\ActiveForm;
use yii\helpers\ArrayHelper;
use yii\helpers\Url;

$form = ActiveForm::begin();
$data = ArrayHelper::map($books, 'id', 'title');
echo $form->field($model, 'title')->dropDownList($data, [
    'prompt'=>'-Choose a title-',
]);
echo $form->field($model, 'author')->textInput();
echo $form->field($model, 'year')->textInput();
ActiveForm::end();
```

Dropdown ke Textbox



@app/controllers/AjaxController.php

```
public function actionGetBook($id)
{
    $books = $this->getBooks();
    $bookSelected = [];
    foreach($books as $book){
        if($book['id']==$id){
            $bookSelected = $book;
        }
    }
    // $bookSelected = Book::findOne($id);
    \Yii::$app->response->format = \yii\web\Response::FORMAT_JSON;
    return [
        'book' => $bookSelected,
    ];
}
```




Dropdown ke Textbox



@app/views/ajax/book.php

```
$this->registerJs('

    $("#dynamicmodel-title").change(function() {
        $.get("'.Url::to(['get-book','id'=>'']) .'" + $(this).val(), function(data)
        {
            $("#dynamicmodel-author").val(data.book.author);
            $("#dynamicmodel-year").val(data.book.year);
        });
    });

');
```

Test:

```
http://localhost/basic/web/ajax/book
```



Dropdown ke Dropdown

Dropdown ke Dropdown atau yang terkenal disebut sebagai *dependent dropdown*. Kali ini kita akan menggunakan data dari tabel *database*, namun tanpa menggunakan *model Active Record*. Query dilakukan dengan menggunakan DAO.

Kita akan menggunakan dua tabel yaitu *province* dan *city*.

```
CREATE TABLE province (  
  id int primary key auto_increment,  
  name varchar(255) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
CREATE TABLE city (  
  id int primary key auto_increment,  
  province_id int,  
  name varchar(255) NOT NULL,  
  type enum('kabupaten','kota'),  
  postal_code varchar(10),  
  foreign key city(province_id) references province(id)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```



Dropdown ke Dropdown



@app/controllers/AjaxController.php

```
public function getProvinces()  
{  
    return (new \yii\db\Query())  
        ->select('*')  
        ->from('province')  
        ->orderBy(['name' => SORT_DESC])  
        ->all(\yii::$app->db);  
}
```



Dropdown ke Dropdown



@app/controllers/AjaxController.php

```
public function actionDepdrop()
{
    $model = new \yii\base\DynamicModel([
        'province_id', 'city_id',
    ]);
    $model->addRule(['province_id'], 'integer');
    $model->addRule(['city_id'], 'integer');

    return $this->render('depdrop', [
        'model' => $model,
        'provinces' => $this->getProvinces(),
    ]);
}
```



Dropdown ke Dropdown



@app/views/ajax/depdrop.php

```
<?php
use yii\widgets\ActiveForm;
use yii\helpers\ArrayHelper;
use yii\helpers\Url;
$form = ActiveForm::begin();
$data = ArrayHelper::map($provinces, 'id', 'name');
echo $form->field($model, 'province_id')->dropDownList($data, [
    'prompt'=>'-Choose a province-',
]);
echo $form->field($model, 'city_id')->dropDownList([], [
    'prompt'=>'-Choose a city-',
]);
ActiveForm::end();
```



Dropdown ke Dropdown



@app/controllers/AjaxController.php

```
public function actionGetCities($province_id)
{
    $cities = (new \yii\db\Query())
        ->select('*')
        ->from('city')
        ->where([
            'province_id'=>$province_id,
        ])
        ->all(\yii::$app->db);
    \Yii::$app->response->format = \yii\web\Response::FORMAT_JSON;
    return [
        'cities' => $cities,
    ];
}
```



Dropdown ke Dropdown



@app/views/ajax/depdrop.php

```
$this->registerJs('

$("#dynamicmodel-city_id").attr("disabled",true);
$("#dynamicmodel-province_id").change(function() {
    $.get("' . Url::to(['get-cities','province_id'=>'']) . '" + $(this).val(), function(data)
    {
        select = $("#dynamicmodel-city_id")
        select.empty();
        var options = "<option value=\'\'>-Choose a city-</option>";
        $.each(data.cities, function(key, value) {
            options += "<option value=\'"+value.id+"\'>" + value.name + "</option>";
        });
        select.append(options);
        $("#dynamicmodel-city_id").attr("disabled",false);
    });
});

');
```

Pjax

Pjax singkatan dari *push state* + *Ajax*, merupakan plugin JQuery yang menggunakan Ajax dan HTML5 *pushState* untuk memberikan pengalaman *browsing* yang cepat.

Cara kerja Pjax adalah mengambil konten halaman web dari server melalui Ajax dan mengganti konten dari suatu halaman yang sedang tampil. Kemudian meng-*update* URL Address dan *state browser* menggunakan HTML5 *pushState* tanpa me-*refresh* halaman.



Pjax pada *GridView*



@app/views/category/index.php

```
<?php \yii\widgets\Pjax::begin(['timeout'=>5000, 'id'=>'pjax-  
gridview']); ?>  
<?= GridView::widget([  
...  
]); ?>  
<?php \yii\widgets\Pjax::end() ?>
```



Pjax pada *GridView*



@app/views/category/index.php

```
<?= GridView::widget([
    'dataProvider' => $dataProvider,
    'filterModel' => $searchModel,
    'columns' => [
        ...
        [
            'class' => 'yii\grid\ActionColumn',
            'buttons' => [
                'view' => function ($url, $model) {
                    $icon='<span class="glyphicon glyphicon-eye-open"></span>';
                    return Html::a($icon,$url);
                },
                'update' => function ($url, $model) {
                    $icon='<span class="glyphicon glyphicon-pencil"></span>';
                    return Html::a($icon,$url);
                },
                'delete' => function ($url, $model) {
                    $icon='<span class="glyphicon glyphicon-trash"></span>';
                    return Html::a($icon,$url,[
                        'data-confirm'=>"Are you sure you want to delete this item?",
                        'data-method'=>'post',
                    ]);
                },
            ],
        ],
    ],
]); ?>
```



Pjax pada *Gridview*



@app/controllers/CategoryController.php

```
public function actionView($id)
{
    return $this->renderAjax('view', [
        'model' => $this->findModel($id),
    ]);
}
```



Pjax pada *Gridview*



@app/controllers/CategoryController.php

```
public function actionView($id)
{
    if(Yii::$app->request->isAjax){
        return $this->renderAjax('view', [
            'model' => $this->findModel($id),
        ]);
    }
    else{
        return $this->render('view', [
            'model' => $this->findModel($id),
        ]);
    }
}
```



Pjax pada *Gridview*

Lakukan hal yang sama untuk actionCreate dan actionUpdate



@app/controllers/CategoryController.php

```
public function actionCreate()
{
    $model = new Category();
    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        return $this->redirect(['view', 'id' => $model->id]);
    } else {
        if (Yii::$app->request->isAjax) {
            return $this->renderAjax('create', [
                'model' => $model,
            ]);
        }
        else{
            return $this->render('create', [
                'model' => $model,
            ]);
        }
    }
}
```



Pjax pada Fungsi Hapus



@app/views/category/index.php

```
'delete' => function ($url, $model) {  
    $icon='<span class="glyphicon glyphicon-trash"></span>';  
    return Html::a($icon,$url,[  
        'class'=>'pjaxDelete'  
    ]);  
},
```



Pjax pada Fungsi Hapus



@app/views/category/index.php

```
$this->registerJs('
    /* fungsi ini akan dijalankan ketika class pjaxDelete di klik */
    $(".pjaxDelete").on("click", function (e) {
        /* cegah link menjalankan default action */
        e.preventDefault();
        if(confirm("Are you sure you want to delete this item?")){
            /* request actionDelete dengan method post */
            $.post($(this).attr("href"), function(data) {
                /* reload gridview */
                $.pjax.reload("#pjax-gridview",{ "timeout":false});
            });
        }
    });
');
```



Pjax pada Fungsi Hapus



@app/controllers/CategoryController.php

```
public function actionDelete($id)
{
    $this->findModel($id)->delete();
    //return $this->redirect(['index']);
}
```




Pjax pada Submit Form



@app/views/category/_form.php

```
<?php
use yii\helpers\Html;
use yii\widgets\ActiveForm;
use yii\widgets\Pjax;
Pjax::begin([
    'id'=>'pjax-form','timeout'=>false,
]);
?>

<?php $form = ActiveForm::begin([
    'options' => ['data-pjax' => true ]
]); ?>
<?= $form->field($model, 'title')->textInput(['maxlength' => true]) ?>
<?= $form->field($model, 'description')->textInput(['maxlength' => true]) ?>
<div class="form-group">
    <?= Html::submitButton($model->isNewRecord ? 'Create' : 'Update', ['class' => $model->isNewRecord ? 'btn btn-success' : 'btn btn-primary']) ?>
    <?= Html::a('Back', ['index'], ['class' => 'btn btn-success']) ?>
</div>

<?php ActiveForm::end(); ?>
<?php Pjax::end(); ?>
```



Pjax pada Submit Form



@app/controllers/CategoryController.php

```
public function actionCreate()
{
    $model = new Category();
    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        Yii::$app->session->setFlash('success', 'Data berhasil disimpan');
        if (Yii::$app->request->isAjax) {
            $model = new Category();
            return $this->renderAjax('create', [
                'model' => $model,
            ]);
        }
        else{
            return $this->redirect(['view', 'id' => $model->id]);
        }
    } else {
        ...
    }
}
```



Pjax pada Submit Form



@app/views/category/_form.php

```
Pjax::begin([
    'id'=>'pjax-form','timeout'=>false,
]);
?>

<?php
if(Yii::$app->request->isAjax)
    echo \app\widgets\Alert::widget();
?>

<?php $form = ActiveForm::begin([
    'options' => ['data-pjax' => true ]
]); ?>
```

```
<?php
$this->registerJs('
    $("#pjax-form").on("pjax:end", function() {
        $.pjax.reload("#pjax-gridview",{
            "timeout": false,
            "url": "'.\yii\helpers\Url::to(['index']).'",
            "replace": false,
        });
    });
');
```



Pjax pada Modal



@app/views/category/index.php

```
<?php
use yii\bootstrap\Modal;
Modal::begin([
    'header' => '<h2>Category Modal</h2>',
    'id' => 'categoryModal',
]);
Pjax::begin([
    'id'=>'pjax-modal','timeout'=>false,
    'enablePushState'=>false,
    'enableReplaceState'=>false,
]);

Pjax::end();
Modal::end();
?>
```



Pjax pada Modal



@app/views/category/index.php

```
[
    'class' => 'yii\grid\ActionColumn',
    'buttons' => [
        'view' => function ($url, $model) {
            $icon='<span class="glyphicon glyphicon-eye-open"></span>';
            return Html::a($icon,$url,[
                'data-toggle'=>"modal",
                'data-target'=>"#categoryModal",
            ]);
        },
        'update' => function ($url, $model) {
            $icon='<span class="glyphicon glyphicon-pencil"></span>';
            return Html::a($icon,$url,[
                'data-toggle'=>"modal",
                'data-target'=>"#categoryModal",
            ]);
        },
    ],
]
```



Pjax pada Modal

 @app/views/category/index.php

```
<?php
$this->registerJs('
    ...

    $("#categoryModal").on("shown.bs.modal", function (event) {
        var button = $(event.relatedTarget)
        var href = button.attr("href")
        $.pjax.reload("#pjax-modal",{
            "timeout":false,
            "url": href,
            "replace": false,
        });
    })
');
```



Pjax pada Modal



@app/views/category/_form.php

```
<?= Html::a('Close', ['index'], [  
    'class' => 'btn btn-success',  
    'onclick'=>  
        $("#categoryModal").modal("hide");  
        return false;  
    ],  
    true)  
>
```



Layout

Bekerja dengan layout

Bekerja dengan Layout

Layout merupakan bagian yang bertanggung jawab atas tampilan aplikasi Yii.

Kode layout ada di @app/views/layout

Contoh pengaturan layout

```
public $layout = 'login';
```

atau pada action

```
$this->layout = 'login';
```



Layout satu kolom



@app/views/layout/login.php

```
<?php
use yii\helpers\Html;
use app\assets\AppAsset;
AppAsset::register($this);
?>
<?php $this->beginPage() ?>
<!DOCTYPE html>
<html lang="<?= Yii::$app->language ?>">
<head>
    <meta charset="<?= Yii::$app->charset ?>">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <?= Html::csrfMetaTags() ?>
    <title><?= Html::encode($this->title) ?></title>
    <?php $this->head() ?>
</head>
<body>
<?php $this->beginBody() ?>

<div class="wrap">
    <div class="container">
        <?= $content ?>
    </div>
</div>

<?php $this->endBody() ?>
</body>
</html>
<?php $this->endPage() ?>
```



Layout satu kolom



@app/controllers/TestController.php

```
<?php
namespace app\controllers;
use Yii;
use yii\web\Controller;
class TestController extends Controller
{
    public $layout = 'login';

    public function actionLogin()
    {
        // render view login
        return $this->render('form-login');
    }
}
```



Layout satu kolom



@app/views/test/form-login.php

```
<h1> Form Login </h1>
<p>
Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet.
Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet.
</p>
```



Layout dua kolom



@app/views/layout/blog.php

```
<?php
use yii\helpers\Html;
use app\assets\AppAsset;
AppAsset::register($this);
?>
<?php $this->beginPage() ?>
<!DOCTYPE html>
<html lang="<?= Yii::$app->language ?>">
<head>
    <meta charset="<?= Yii::$app->charset ?>">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <?= Html::csrfMetaTags() ?>
    <title><?= Html::encode($this->title) ?></title>
    <?php $this->head() ?>
</head>
<body>
<?php $this->beginBody() ?>
```

```
<div class="wrap">
    <div class="container">
<div class="col-md-3">
<h1> Menu </h1>
<ul>
    <li>Home</li>
    <li>About</li>
    <li>Contact</li>
</ul>
</div>
        <div class="col-md-9">
            <?= $content ?>
        </div>
    </div>

<?php $this->endBody() ?>
</body>
</html>
<?php $this->endPage() ?>
```



Layout dua kolom



@app/controllers/TestController.php

```
public function actionBlog()
{
    // select layouts
    $this->layout = 'blog';
    // render view blog
    return $this->render('blog');
}
```



Layout dua kolom



@app/views/test/blog.php

```
<h1> Article </h1>
<p> Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet. Lorem
ipsum sit dolor amet. Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet. Lorem ipsum sit
dolor amet. Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet.
Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet. Lorem ipsum sit dolor amet. Lorem
ipsum sit dolor amet. Lorem ipsum sit dolor amet. </p>
```

Thanks!

Any questions?

You can find me at:
@edoriansyah
edo@nurulfikri.co.id

Credits

Special thanks to :

- Presentation template by [SlidesCarnival](#)
- "Membangun Aplikasi Profesional Berbasis Web Menggunakan Yii Framework" Book by Hafid Mukhlisin