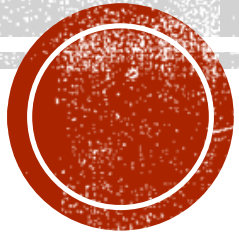
Yii2: Application Structure

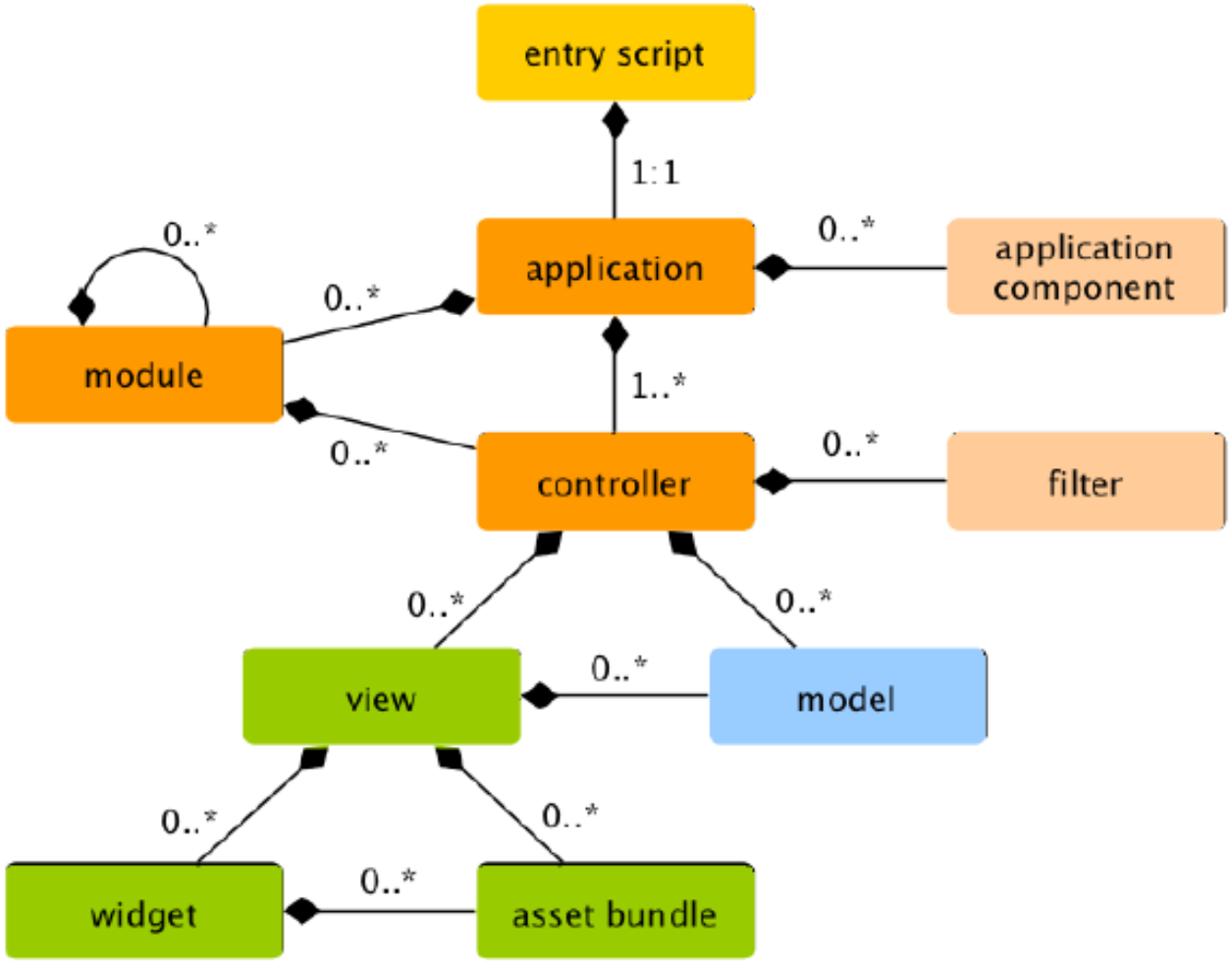
Oleh: Ahmad Syauqi Ahsan



Overview

- Aplikasi Yii2 menganut pola desain MVC (Model-View-Controller)
  - Model: merepresentasikan data, logika bisnis, serta aturan-aturan yang ada.
  - View: representasi keluaran dari model.
  - Controller: menerima masukan dan merubahnya menjadi perintah untuk mengakses model dan menampilkan view.
- Selain MVC, sebuah aplikasi Yii2 juga memiliki:
  - Entry scripts: PHP script yang bias diakses langsung oleh pengguna. Mereka merupakan "awal" dari semua request ke aplikasi.
  - ***Application***: merupakan objek yang dapat diakses secara global. Objek ini mengelola berbagai komponen dan mengkoordinasikan mereka untuk memenuhi *request*.
  - Komponen ***application***: merupakan objek-objek yang tergabung dengan ***application*** dan menyediakan berbagai layanan untuk memenuhi *request*.
  - ***Modules***: merupakan sebuah paket yang didalamnya terdapat MVC secara komplit. Sebuah aplikasi Yii2 dapat dibuat dan dikelola menjadi beberapa modules.
  - ***Filters***: merepresentasikan kode yang perlu untuk dijalankan sebelum dan/atau sesudah sebuah *request* ditangani oleh controller.
  - ***Widgets***: merupakan objek yang biasanya "dilekatkan" pada view untuk mempermudah pembentukan tampilan.

Struktur Aplikasi Yii2

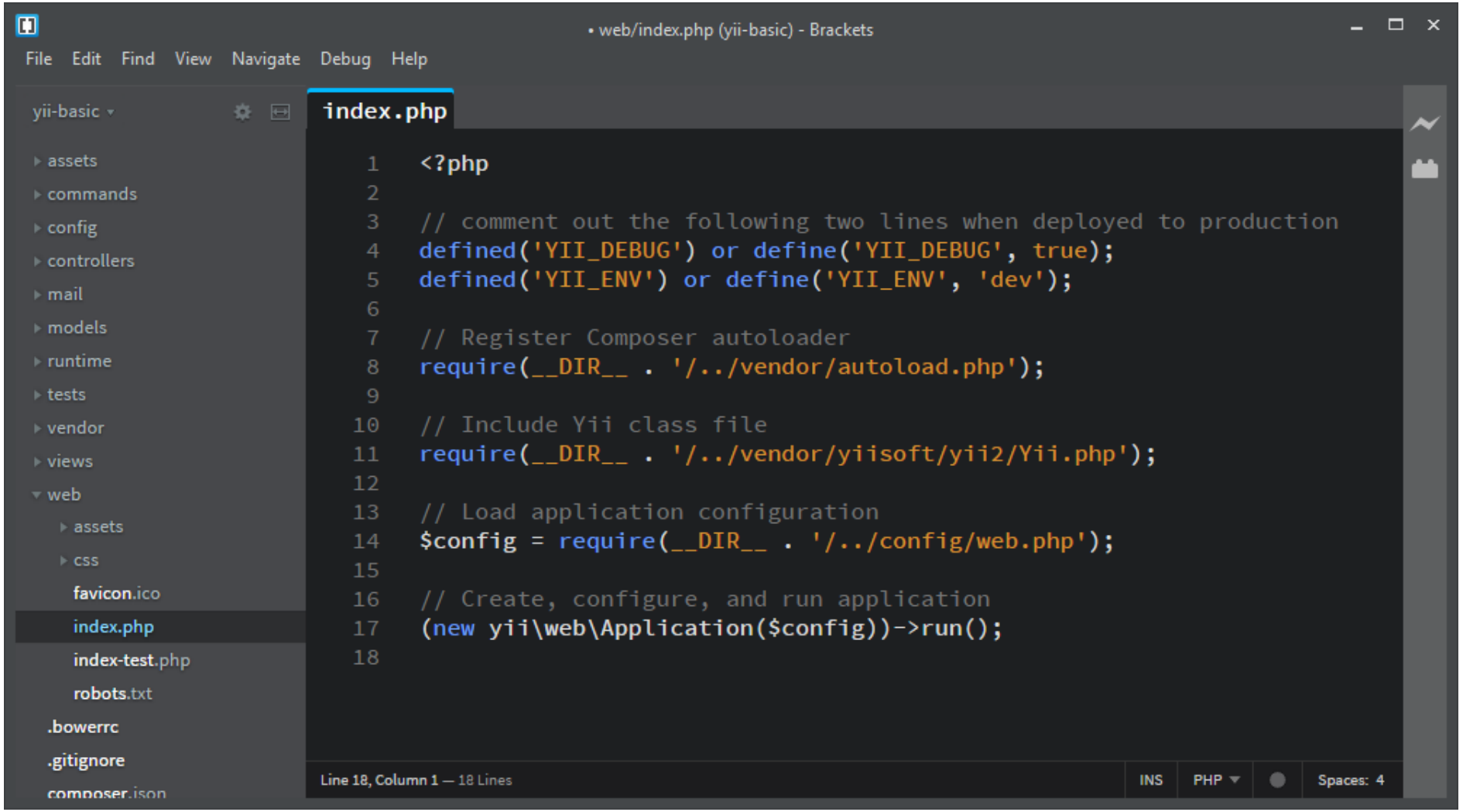


Entry Script

- Entry script merupakan mata rantai pertama dari aplikasi Yii2. Setiap aplikasi Yii (baik aplikasi web maupun aplikasi console) harus memiliki 1 entry script.
- Entry script untuk aplikasi web biasanya bernama **index.php**. File ini harus diletakkan pada direktori yang dapat diakses melalui web.
- Entry script biasanya digunakan untuk hal-hal berikut ini:
  - Mendefinisikan konstanta global
  - Meregister Composer Autoloader
  - Mengikutsertakaan class Yii
  - Memuat komfigurasi aplikasi
  - Membuat dan mengkonfigurasi Instance dari aplikasi
  - Memanggil fungsi yii\base\Application::run() untuk memproses permintaan

- Note: selain dapat digunakan untuk membuat aplikasi web, Yii2 juga dapat digunakan untuk membuat aplikasi console.

Entry Script untuk Aplikasi Web



Application

- **Application** merupakan objek yang mengelola keseluruhan struktur dan siklus hidup dari aplikasi Yii2.
- Setiap aplikasi Yii2 mempunyai sebuah objek aplikasi yang dibuat di *entry script*.
- Objek aplikasi ini dapat diakses secara global melalui **\Yii::$app**.
- Konfigurasi dari **application** juga akan dimuat bersamaan dengan pembuatan **application** di *entry script*. File konfigurasi untuk basic template biasanya bernama **web.php**.
- File konfigurasi berisi property, component, dan module yang digunakan oleh aplikasi.

web.php

```php
<?php

$params = require(__DIR__ . '/params.php');

$config = [
    'name' => 'SI Jurusan',
    'id' => 'sijur',
    'basePath' => dirname(__DIR__),
    'bootstrap' => ['log'],
    'components' => [
        'request' => [
            // !!! insert a secret key in the follow
            empty) - this is required by cookie val
            'cookieValidationKey' => 'sdafsfasdfd',
        ],
        'cache' => [
            'class' => 'yii\caching\FileCache',
        ],
        'user' => [
            'identityClass' => 'app\models\User',
            'enableAutoLogin' => true,
```

Application Property

- Property dari aplikasi Yii2 dapat diakses melalui **\Yii::$app->*nama_property***.
- Berikut ini adalah ***required property*** dari aplikasi Yii.
  - Yii\base\Application::id : adalah property yang menyimpan id dari aplikasi. ID ini bersifat unik dan berguna untuk membedakan aplikasi satu dengan lainnya. Gunakan hanya alphanumeric untuk id.
  - Yii\base\Application::basePath : adalah property yang digunakan untuk menentukan **root directory** dari aplikasi.
- Berikut ini adalah beberapa property lainnya.
  - Yii\base\Application::name : untuk menyimpan nama dari aplikasi.
  - Yii\base\Application::components : merupakan property yang digunakan untuk menentukan komponen-komponen yang dapat digunakan oleh aplikasi.
  - Yii\base\Application::modules : merupakan property yang digunakan untuk menentukan modul-modul yang dapat digunakan oleh aplikasi.
  - Yii\base\Application::defaultRoute: untuk menentukan **route** yang akan digunakan jika ada request yang tidak menyebutkan **route** yang diminta. Nilai default dari property ini adalah **site/index**.

Application Components

- Sebuah aplikasi Yii2 dibentuk dari beberapa komponen yang menyediakan layanan yang berbeda-beda.
- Sebagai contoh, komponen **urlManager** bertanggung jawab untuk menentukan rute dari sebuah request. Komponen **db** menyediakan layanan terkait dengan penggunaan database. Dan lain sebagainya.
- Komponen dari aplikasi akan dibuat ketika komponen tersebut digunakan untuk pertama kali.
- Berikut ini beberapa komponen inti dari aplikasi Yii2:
  - "db" : mengelola hubungan dengan database.
  - "errorHandler" : menangani **error** dan **exception** pada PHP.
  - "user" : menangani otentikasi user
  - Dan lain-lain

Controllers

- **Controller** adalah bagian dari arsitektur MVC yang bertugas untuk menerima permintaan (request) dan memberikan tangapan (response).
- **Controller** akan menerima permintaan, meneruskan permintaan ke model, memasukkan hasil dari pemrosesan dari model ke dalam view, dan kemudian membentuk tanggapan.
- Nama file php dari sebuah controller mengikuti format ***Controllerid*Controller.php.** Sebagai contoh, file **JurusanController.php** merupakan file untuk controller dengan id **jurusan**.

Controller - Action

- Sebuah **controller** dibentuk dari satu atau beberapa **actions**.
- ActionID merupakan kata unik yang digunakan untuk membedakan **action** satu dengan **action** lainnya dalam sebuah **controller**.
- ActionID ditulis dalam format ***"actionActionID"***. Contoh: **actionCreate()**.
- Pengguna dapat mengakses **action** dari **controller** menggunakan **route** dengan format ***"controllerID/actionID"***.
  Sebagai contoh, **action** "create" dari **controller** "Jurusan" dapat diakses melalui **route** "jurusan/create"
- Secara default, action "**index**" akan dipanggil jika dalam request hanya menyebutkan nama controller saja (tanpa nama action).
- Action default dapat di ganti melalui property defaultAction.

JurusanController.php

```php
<?php

namespace app\controllers;

use Yii;
use app\models\Jurusan;
use app\models\JurusanSearch;
use yii\web\Controller;

class JurusanController extends Controller
{
    public $defaultAction = 'create';

    public function actionIndex()
    {
        $searchModel = new JurusanSearch();
        $dataProvider = $searchModel->search(Yii::$app->request->queryParams);

        return $this->render('index', [
            'searchModel' => $searchModel,
            'dataProvider' => $dataProvider,
        ]);
    }

    public function actionCreate()
    {
        $model = new Jurusan();

        if ($model->load(Yii::$app->request->post()) && $model->save()) {
            return $this->redirect(['view', 'id' => $model->id]);
        } else {
            return $this->render('create', [
                'model' => $model,
            ]);
        }
    }

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

    /**
     * Updates an existing Jurusan model.
```

Controller - Best Practices

- Di dalam aplikasi yang didesain dengan baik, sebuah controller biasanya berukuran sangat kecil, dan untuk setiap *action* hanya berisi beberapa baris kode saja.
- Secara garis besar dapat disimpulkan bahwa controller:
  - Dapat mengakses data permintaan (request) seperti HTTP headers, cookies, request parameter, dan lain-lain.
  - Dapat memanggil **methods** dari **models** atau dari **components** lain.
  - Dapat menggunakan **views** untuk membentuk **response** (keluaran).
  - Seharusnya tidak memproses data permintaan (request). Pemrosesan ini harusnya dilakukan dalam **models**.
  - Seharusnya tidak mengikutsertakan kode HTML. Kode HTML lebih baik diletakkan pada **views**.

Models

- **Model** adalah bagian dari arsitektur MVC.
- **Model** adalah objek yang digunakan untuk merepresentasikan data, aturan-aturan, dan logika bisnis.
- **Model** dapat dihubungkan dengan table di sebuah database ataupun berdiri sendiri.
- Sebuah **model** dapat memiliki **attributes**, **attribute labels**, **validation rules**, dll.
- **Attributes** merepresentasikan data dalam sebuah **model**.
  - **Model** yang dihubungkan dengan table di database secara otomatis menjadikan seluruh field dalam table tersebut sebagai atributnya.
  - Atribut dapat ditambahkan dengan cara seperti membuat variable.
  - Atributes dapat diakses secara normal atau seperti mengakses array.

```
$model->test = 'contoh'; //atau
$model['test'] = 'contoh';
```

```php
<?php

namespace app\models;

use Yii;

class Jurusan extends \yii\db\ActiveRecord
{
    public $test;
    public $coba;
```

Models (2)

- **Attribute Labels** adalah label yang ditampilkan pada form untuk setiap attribute.
- Jika model dibuat menggunakan **Gii**, maka semua field dalam table akan dibuatkan **attribute labels**-nya secara otomatis.
- **Validasi**: setiap data untuk sebuah model yang diterima dari pengguna harus divalidasi terlebih dahulu.
  Contoh validasi:
  - Jumlah huruf maksimal untuk no telepon tidak boleh melebihi 15.
  - Email harus sesuai dengan bentuk penulisan email yang valid.
  - Dan lain-lain.
- Proses validasi ini dapat diatur melalui method **rules**.

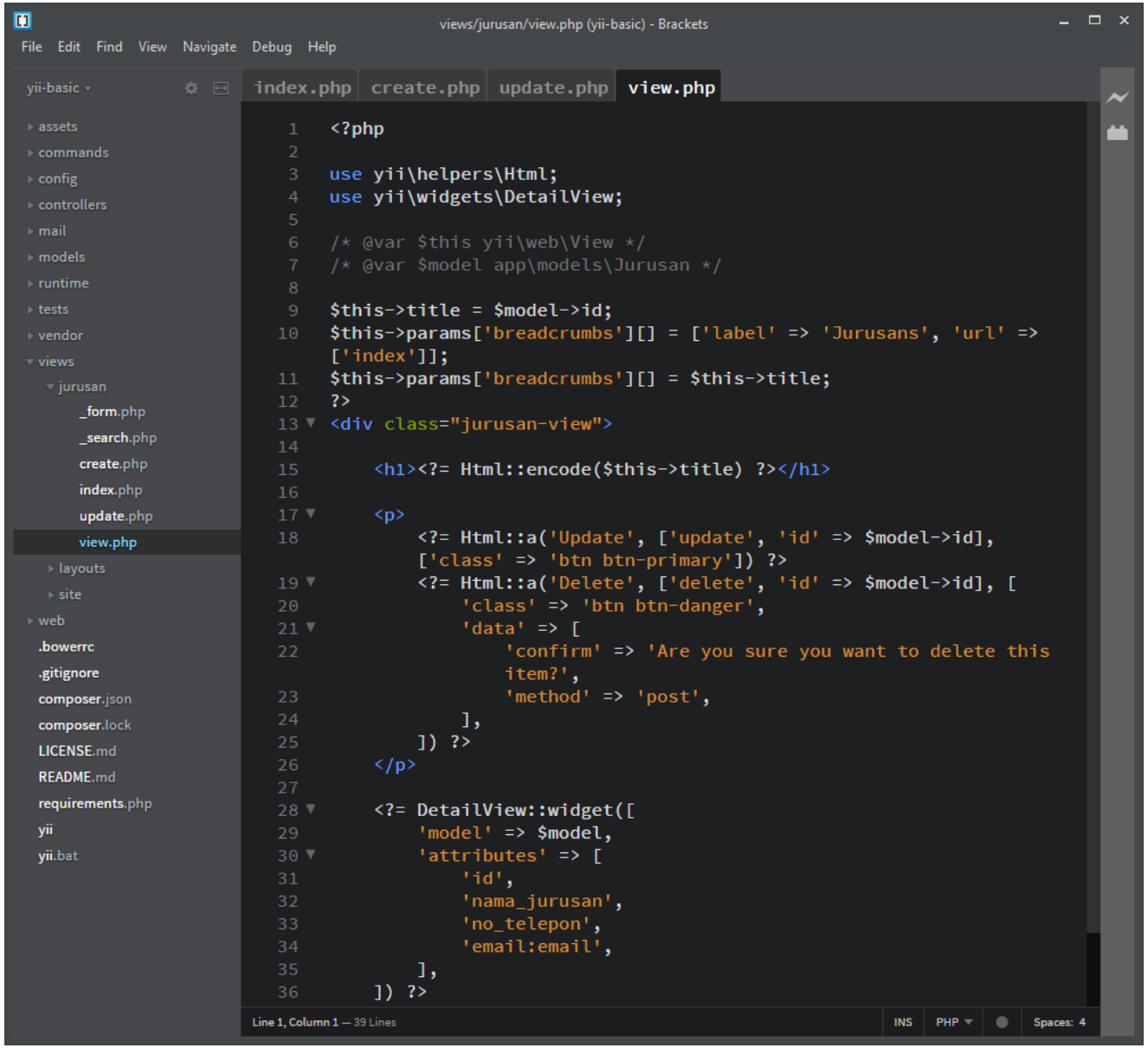
```php
<?php

namespace app\models;

use Yii;

class Jurusan extends \yii\db\ActiveRecord
{
    public $test;
    public $coba;

    public static function tableName()
    {
        return 'jurusan';
    }

    public function attributeLabels()
    {
        return [
            'id' => 'ID',
            'nama_jurusan' => 'Nama Jurusan',
            'no_telepon' => 'No Telepon',
            'email' => 'Email',
        ];
    }

    public function rules()
    {
        return [
            [['nama_jurusan', 'no_telepon', 'email'], 'required'],
            [['nama_jurusan'], 'string', 'max' => 255],
            [['no_telepon'], 'string', 'max' => 15],
            [['email'], 'string', 'max' => 50],
            [['email'], 'email'],
        ];
    }


}
```

Models - Best Practices

- Secara garis besar, sebuah model:
  - Dapat berisi atribut untuk merepresentasikan data bisnis.
  - Dapat berisi validation rules (aturan validasi) untuk memastikan validitas dan integritas data
  - Dapat berisi metode-metode untuk mengimplementasikan logika bisnis.
  - Seharusnya tidak mengakses request, session, atau environtment data yang lain secara langsung. Data-data ini harus dimasukkan ke dalam **model** oleh **controller**.
  - Seharusnya tidak mengikutsertakan kode HTML. Kode HTML lebih baik diletakkan pada **views**.
  - Seharusnya tidak memiliki scenario yang terlalu banyak.

Views

- **View** adalah bagian dari arsitektur MVC yang digunakan untuk menampilkan data ke pengguna.
- **View** dibentuk menggunakan gabungan dari kode PHP dan HTML.
- Dengan pertimbangan keamanan (untuk menghindari ***cross-site-scripting***), terapkan metode pemeriksaan untuk menampilkan data yang dimasukkan oleh pengguna.
  - Gunakan metode ***`yii\helpers\Html::encode()`*** untuk menampilkan data teks
  - Gunakan metode **`yii\helpers\HtmlPurifier::process()`** untuk menampilkan data html

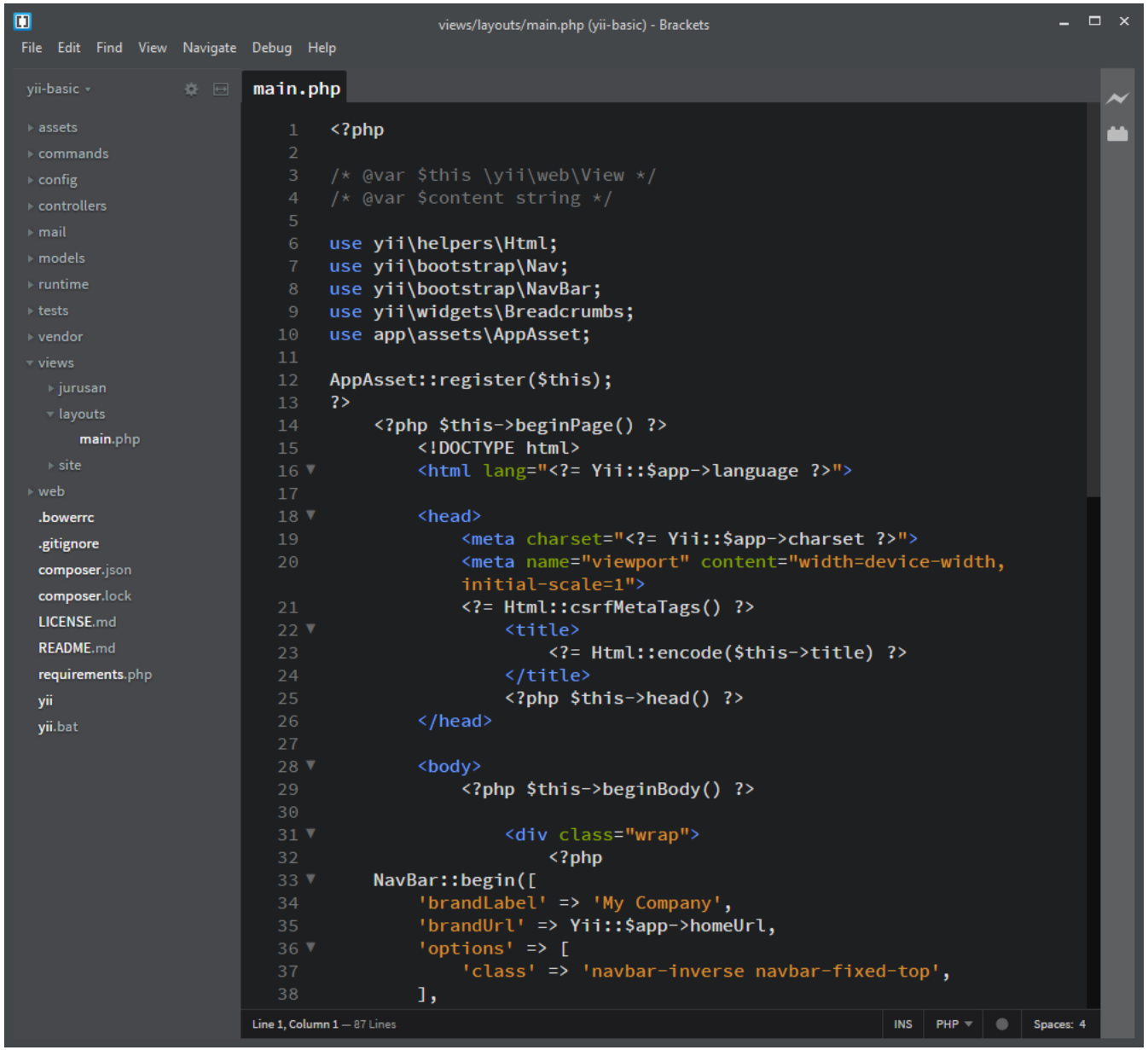


Views (2)

- Sebuah view dapat di-render (dipanggil) dari ***controller***.
- Dari ***controller***, sebuah view dapat dipanggil dengan salah satu metode berikut:
  - **`render()`**: memanggil view dan menerapkan layout
  - **`renderPartial()`**: memanggil view tanpa layout apapun
  - **`renderAjax()`**: memanggil view tanpa layout, dengan mengikutsertakan semua ***js*** dan ***css***. Biasa digunakan untuk merespon request Ajax.
  - **`renderFile()`** dan **`renderContent()`**: dua metode ini relative jarang digunakan.
- Beberapa views yang di-render dari sebuah controller yang sama harus diletakkan di dalam sebuah direktori yang sama pula.
  Contoh: semua view yang di-render dari ***JurusanController*** diletakkan pada directory "jurusan".

Layouts

- **Layouts** adalah **views** special yang digunakan untuk menampilkan bagian yang sama untuk semua (atau beberapa) views yang ada pada sebuah aplikasi Yii.
- Secara default, terdapat sebuah layout dengan nama ***main.php***.
- Tampilan umum (yang dimunculkan pada setiap halaman) dari sebuah aplikasi Yii, seperti menu, diletakkan dalam file ***main.php*** ini.



Filters

- Filters merupakan sebuah objek yang dieksekusi sebelum dan/atau sesudah ***action*** pada ***controller***.
- Filter dapat digunakan dengan meletakkannya di dalam metode ***behaviors()*** pada sebuah ***controller***.
- Contoh built-in filter yang sering digunakan adalah ***AccessControl***. Filter ini berfungsi untuk memastikan pengguna yang mengakses sebuah action memenuhi syarat tertentu. (lihat kode program disamping)
- Built-in filter lain yang cukup berguna adalah:
  - HttpCache: untuk mengimplementasikan caching di sisi client.
  - PageCache: untuk mengimplementasikan caching di sisi server.

```
// in Controller

use yii\filters\AccessControl;

public function behaviors()
{
  return [
    'access' => [
      'class' => AccessControl::className(),
        'only' => ['create', 'update'],
        'rules' => [
          // allow authenticated users
          [
            'allow' => true,
            'roles' => ['@'],
          ],
          // everything else is denied by default
        ],
      ],
  ];
}
```

Assets

- Asset pada aplikasi Yii merupakan file yang menjadi referensi dari sebuah halaman web (contoh: file css atau js)
- Untuk dapat digunakan, sebuah asset harus diletakkan pada directory yang dapat diakses dari internet.
- Pada aplikasi Yii2, direkomendasikan untuk mengelola asset menggunakan Asset Bundles
  - File .css atau .js tidak secara langsung di-include-kan dalam view.
  - Asset Bundles di-include-kan pada view dengan memanggil metode `register()`.
  - Dengan cara ini, file asset yang asli tidak harus diletakkan pada directory yang dapat diakses dari internet. Yii yang akan meng-copy-kannya secara otomatis.
  - Jika terdapat perubahan pada file asset, misalkan setelah melakukan upgrade, file asset otomatis akan diperbaharui.
- ***Note***: Widget akan secara otomatis meng-include-kan file asset yang dibutuhkannya.

```php
// file assets/AppAsset.php
<?php
namespace app\assets;
use yii\web\AssetBundle;

class FontAwesomeAsset extends AssetBundle
{
  public $baseUrl = '@web';
  public $css = [
    'css/site.css',
  ];

  public $js = [

  ];

  public $depends = [
    'yii\web\YiiAsset',
    'yii\bootstrap\BootstrapAsset',
  ];
}
```

```php
// file views/layouts/main.php
use app\assets\AppAsset;

AppAsset::register($this);
```

Latihan

- Untuk aplikasi frontend:
  - Buat CRUD untuk item (jika belum dibuat pada latihan sebelumnya). Kemudian hapus semua action kecuali action "index" dan "view".
  - Pada layouts/main, tambahkan menu "Produk" yang mengarah ke route "item/index".
  - Atur konfigurasi aplikasi sehingga route "item/index" menjadi default route.
- Beri nama aplikasi anda!
  - "Toko Baru" untuk aplikasi frontend dan "Admin Toko Baru" untuk aplikasi backend.
  - ***Note***: Gunakan property "name" pada file konfigurasi dari aplikasi anda)
- Tampilkan property "name" tersebut sebagai awalan untuk title dari semua halaman pada aplikasi anda!
  (***Note***: title diatur melalui *view*)

Terima Kasih