Объединение и минимизация asset

Если Веб-страница содержит много файлов CSS и/или JavaScript, страница будет открываться очень медленно, так как браузер отправляет большое количество HTTP-запросов на загрузку каждого файла в отдельных потоках. Чтобы уменьшить количество запросов и подключений, мы можем объединить и сжать несколько файлов CSS / JavaScript в один или очень мало файлов в рабочем режиме, а затем включить эти сжатые файлы на странице вместо исходных.

Подготовка

* Создайте новое yii2-app-basic приложение с помощью диспетчера пакетов Composer, как описано в официальном руководстве по адресу  
  <http://www.yiiframework.com/doc-2.0/guide-start-installation.html>.   
  По русски <http://yiiframework.domain-na.me/doc/guide/2.0/ru/start-installation>
* Скачать компилятор jar файлов из <https://developers.google.com/closure/compiler/>
* Скачать yuicompressor.файл jar из [https://githiih.rom/yni/ynirompressor/releases](https://github.com/yui/yuicompressor/releases)
* Загрузите и установите среду выполнения Java (JRE) из <http://www.java.com>

Как это сделать...

Выполните следующие действия, чтобы объединить и минимизировать активы:

1. Откройте исходный HTML-код страницы индекса приложения. Проверьте, похож ли он на следующую структуру:

< ! DOCTYPE html>

<html lang="en-US">

<head>

<title>My Yii Application</title>

<link href="/assets/9b3b2888/css/bootstrap.css" rel="stylesheet">

<link href="/css/site.css" rel="stylesheet">

</head>

<body>

<script src="/assets/25f82b8a/jquery.js"></script>

<script src="/assets/f4307424/yii.js"></script>

<script src="/assets/9b3b2888/js/bootstrap.js"></script>

</body>

</html>

Страница содержит три файла JavaScript.

1. Откройте config/console.php файл и добавить @webroot и @web определения псевдонимов:

<?php

Yii::setAlias('@webroot', DIR . '/../web');

Yii::setAlias('@web', '/');

1. Откройте консоль и выполните следующую команду:  
   yii asset/template assets.php
2. Откройте сгенерированные assets.php файл и настроить его следующим образом:

<?php  
return [

'jsCompressor' => 'java -jar compiler.jar --js {from} --js\_output\_file {to}',  
'cssCompressor' => 'java -jar yuicompressor.jar --type css {from} -o {to}',  
'bundles' => [

'app\assets\AppAsset',

'yii\bootstrap\BootstrapPluginAsset',

],

'targets' => [

'all' => [

'class' => 'yii\web\AssetBundle',

'basePath' => '@webroot/assets',

'baseUrl' => '@web/assets',

'js' => 'all-{hash}.js',

'css' => 'all-{hash}.css',

],

],

'assetManager' => [

'basePath' => '@webroot/assets',

'baseUrl' => '@web/assets',

],

];

5. Выполните команду объединения

yii asset assets.php config/assets-prod.php.

Если это успешно, вы должны получить config/assets-prod.php файл со следующей конфигурацией:

<?php  
return [

'all' => [

'class' => 'yii\\web\\AssetBundle',

'basePath' => '@webroot/assets',

'baseUrl' => '@web/assets',

'js' => [

' all-fe792d4766bead53e7a9d851adfc6ec2.js',

],

'css' => [

'all-37cfb42649f74eb0a4bfe0d0e715c420.css',

],

],

'yii\\web\\JqueryAsset' => [

'sourcePath' => null,

'js' => [],

' css ' => [],

'depends' => [

'all',

],

],

'yii\\web\\YiiAsset' => [

'sourcePath' => null,

'js' => [],

'css' => [],

'depends' => [

'yii\\web\\JqueryAsset',

'all',

],

],

'yii\\bootstrap\\BootstrapAsset' => [

'sourcePath' => null,

'js' => [],

' css ' => [],

'depends' => [

'all',

],

],

'app\\assets\\AppAsset' => [

'sourcePath' => null,

'js' => [],

'css' => [],

'depends' => [

'yii\\web\\YiiAsset',

'yii\\bootstrap\\BootstrapAsset',

'all',

],

],

'yii\\bootstrap\\BootstrapPluginAsset' => [

'sourcePath' => null,

'js' => [],

' css ' => [],

'depends' => [

'yii\\web\\JqueryAsset',

'yii\\bootstrap\\BootstrapAsset',

'all',

],

],

];

1. Добавьте конфигурацию для компонента assetManager в config/web.php-файл:

'components' => [

// ...

'assetManager' => [

'bundles' => YII\_ENV\_PROD ? require( DIR . '/assets-prod.php') : [],

],

],

1. Включите производственный режим в web/index.php:  
   defined('YII\_ENV') or define('YII\_ENV', 'prod');
2. Перезагрузите страницу в браузере и снова увидите HTML-код. Теперь он должен содержать одиночные строки для включения наших сжатых файлов:

< ! DOCTYPE html>

<html lang="en-US">

<head>

<title>My Yii Application</title>

<link href="/assets/all-37cfb42649f74eb0a4bfe0d0e715c420.css"  
rel="stylesheet">

</head>

<body>

<script src="/assets/all-fe792d4766bead53e7a9d851adfc6ec2.js"></script>  
</body>

</html>

Как это работает...

Прежде всего, на нашей странице был набор включенных файлов:

<link href="/assets/9b3b2888/css/bootstrap.css" rel="stylesheet">

<link href="/css/site.css" rel="stylesheet">

<script src="/assets/25f82b8a/jquery.js"></script>

<script src="/assets/f4307424/yii.js"></script>

<script src="/assets/9b3b2888/js/bootstrap.js"></script>

Затем мы создали asset.php конфигурационный файл и указанные пакеты для сжатия:

'bundles' => [

'app\assets\AppAsset',

'yii\bootstrap\BootstrapPluginAsset',

],

Замечание:

Мы могли бы определить все промежуточные пакеты ассетов, таких как веб-фреймворка yii\\jqueryAsset и yii\web\YiiAsset, но эти активы уже указаны как зависимости от AppAsset и BootstrapPluginAsset, и сжатие команда автоматически решает все эти зависимости.

AssetManager публикует все активы в классические подкаталоги в web/assets и после публикации запускает компрессоры, чтобы объединить все CSS и JS файлы в all-{hash}.js and all-{hash}.css.

Проверьте, включает ли файл CSS другие ресурсы относительными путями, такими как bootstrap.css:  
@font-face {

font-family: 'Glyphicons Halflings';

src: url('../fonts/glyphicons-halflings-regular.eot');

}

Если это так, то в комбинированном файле наш компрессор изменяет все относительные пути для хранения всех отношений следующим образом:

@font-face{

font-family: 'Glyphicons Halflings';

src: url('9b3b2888/fonts/glyphicons-halflings-regular.eot');

}

После обработки мы получаем assets-prod.php файл с конфигурацией пакетов компонента assetManager. Он определяет новый виртуальный ресурс как зависимость чистых копий исходных пакетов:

return [

'all' => [

'class' => 'yii\\web\\AssetBundle',

'basePath' => '@webroot/assets',

'baseUrl' => '@web/assets',

'js' => [

'all-fe792d4766bead53e7a9d851adfc6ec2 .js',

],

'css' => [

' all-37cfb42649f74eb0a4bfe0d0e715c420.css',

],

],

'yii\\web\\JqueryAsset' => [

'sourcePath' => null,

'js' => [],

'css' => [],

'depends' => [

'all',

],

],

// ...

]

Теперь мы можем потребовать эту конфигурацию в config/web.php:

'components' => [

// ...

'assetManager' => [

'bundles' => require( DIR . '/assets-prod.php'),

],

],

Кроме того, файл может потребоваться только для рабочей среды:

'components' => [

// ...

'assetManager' => [

'bundles' => YII\_ENV\_PROD ? require( DIR . '/assets-prod.php') : [],

],

],

Замечание

Не забывайте восстанавливать все сжатые и комбинированные файлы после любых обновлений исходных ресурсов.

Смотрите так же

* Дополнительные сведения об активах см. по следующему URL-адресу <https://www.yiiframework.com/doc/guide/2.0/en/structure-assets>
  1. по русски https://www.yiiframework.com/doc/guide/2.0/ru/structure-assets
* Для компилятора обратитесь к следующему URL-адресу: <https://developers.google.com/closure/compiler/>
* Для компрессора YUI обратитесь к следующему URL: [https://github.com/vmyvnirompressor/](https://github.com/yui/yuicompressor/)