Автоматическая настройка автора

Поведение Blameable позволяет автоматически обновлять одно или несколько полей авторов. Это в основном используется для заполнения данных в поля created\_by и updated\_by. Подобно поведению Метки времени, можно легко задать некоторые специальные параметры и важные события для этого поведения.

Вернемся к примеру из предыдущего раздела. У нас также есть посты нашего блога. Например, предположим, что наша модель блога называется Blogpost. Модель имеет author\_id, поле, которое указывает на то, кто создал этот пост, и updater\_id, поле, которое указывает на то, кто обновил. Мы хотели бы заполнить эти атрибуты автоматически во время событий создания / обновления модели. Сейчас вы можете узнать, как это сделать.

Getting ready

1. Создайте новое приложение с помощью composer, как описано в официальном руководстве <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> )
2. Настройте подключение к базе данных и создайте таблицу с именем blog\_post следующим образом:

DROP TABLE IF EXISTS 'blog\_post';

CREATE TABLE IF NOT EXISTS 'blog\_post' (

'id' INT(10) UNSIGNED NOT NULL AUTO\_INCREMENT,

'author\_id' INT(10) UNSIGNED DEFAULT NULL,

'updater\_id' INT(10) UNSIGNED DEFAULT NULL,

'title' VARCHAR(255) NOT NULL,

'text' TEXT NOT NULL,

PRIMARY KEY ('id')

);

1. Используйте Gii для создания модели BlogPost для таблицы blost\_post.

Как это сделать...

1. Добавьте следующий метод поведения в models/BlogPost.php:

<?php

namespace app\models;

use Yii;

use yii\db\BaseActiveRecord;

/\*\*

* This is the model class for table "blog\_post".

\*

* @property integer $id
* @property integer $author\_id
* @property integer $updater\_id
* @property string $title
* @property string $text \*/

class BlogPost extends \yii\db\ActiveRecord {

/\*\*

\* @return array \*/

public function behaviors()

{

return [

[

'class' => 'yii\behaviors\BlameableBehavior',

'attributes' => [

BaseActiveRecord::EVENT\_BEFORE\_INSERT => 'author\_id', BaseActiveRecord::EVENT\_BEFORE\_UPDATE => 'updater\_id'

]

]

];

}

}

2. Создание controllers/TestController.php следующим образом:

<?php

namespace app\controllers;

use app\models\BlogPost;

use app\models\User;

use Yii;

use yii\helpers\Html;

use yii\helpers\VarDumper;

use yii\web\Controller;

/\*\*

* Class TestController.
* @package app\controllers \*/

class TestController extends Controller {

public function actionIndex()

{

$users = new User();

$identity = $users->findIdentity(100);

Yii::$app->user->setIdentity($identity);

$blogPost = new BlogPost();

$blogPost->title = 'Very pretty title';

$blogPost->text = 'Success is not final, failure is not fatal...';

$blogPost->save();

return $this->renderContent(Html::tag('pre', VarDumper::dumpAsString( $blogPost->attributes

)));

}

}

3. Вот и все. Теперь запустите test/index. Вы получите следующий результат



Как это работает...

По умолчанию поведение Blameable заполняет атрибуты created\_by и updated\_by, но мы внесем изменения и настроим наше поведение в соответствии с нашими собственными полями.

Мы также указали события и поля модели в модели, поэтому при создании модели будет заполнен author\_id. Аналогично, при обновлении модели, мы заполним updater\_id.

Что Blameable не вставить текущее значение идентификатора пользователя в created\_by и поля updated\_by в создать/обновить модель событий. Это супер-удобный способ делать вещи. Каждый раз, когда модель создается или обновляется, мы автоматически заполняем необходимые поля.

Это очень хорошо работает для небольших проектов, таких как для больших систем, где несколько пользователей являются администраторами, и вам нужно отслеживать, кто что делает. Вы также можете использовать это для реализации веб-интерфейса, например, если у вас была Таблица blog\_comment и вы хотели использовать этот метод для отслеживания автора комментария. Кроме того, можно задать поля автора в контроллере, но поведение помогает избежать написания ненужного и дополнительного кода. Это очень эффективный и простой способ реализовать эту вещь.

Есть еще.

Иногда нам нужно заполнить author\_id и updater\_id идентификатором, отличным от идентификатора текущего пользователя. В таком случае, мы можем отделить наше поведение следующим образом:

$model->detachBehavior('blammable ' );

Мы можем отделить любое поведение которое понравится в этом случае.

Смотрите так же

Для получения дополнительной информации обратитесь к <http://www.yiiframework.com/doc-2.0/yii-behaviors-blameablebehavior.html>