database.

For example, we can create our own event. Just add a new actionTestNew with the following code:

public function actionTestNew()

{

$article = new Article();

$article->name = 'Valentine\'s Day\'s coming? Aw crap! I forgot to get a girlfriend again!';

$article->description = 'Bender is angry at Fry for dating a robot. Stay away from our women.

You've got metal fever, boy. Metal fever';

// $event is an object of yii\base\Event or a child class $article->on(Article::EVENT\_OUR\_CUSTOM\_EVENT, function($event) {

$followers = ['john2@teleworm.us', 'shivawhite@cuvox.de', 'kate@dayrep.com' ]; foreach($followers as $follower) {

Yii::$app->mailer->compose()

->setFrom('techblog@teleworm.us')

->setTo($follower)

->setSubject($event->sender->name)

->setTextBody($event->sender->description)

->send();

}

echo 'Emails have been sent';

});

if ($article->save()) {

$article->trigger(Article::EVENT\_OUR\_CUSTOM\_EVENT);

}

}

Also add the event\_our\_custom\_event constant to models/Article as:

class Article extends \yii\db\ActiveRecord {

CONST EVENT\_OUR\_CUSTOM\_EVENT = 'eventOurCustomEvent';

}

Run http: //localhost:8080/index.php?r=site/test-new.

You should see the same result and all notifications to followers will be sent again. The main difference is we used our custom event name.

After the save, we’ve triggered our event. Events may be triggered by calling the

yii\base\Component: : trigger() method. The method requires an event name, and optionally an event object that describes the parameters to be passed to the event handlers.

See also

[For more information about events refer to http://www.yiiframework.com/doc-2.0/guide-concept-](http://www.yiiframework.com/doc-2.0/guide-concept-events.html) events.html