1. Also, you can attach the filter to other controllers or modules.

How it works...

A filter should extend the yii\base\ActionFilter class, which extends yii\base\Behavior. We can override the beforeAction or afterAction method if we want to do post- and pre-filtering.

For example, we can check user access and throw corresponding HTTP-exceptions in a fail case. In this recipe, we redirect the user to the agreement page if the specific cookie value does not exist:

class AgreementFilter extends ActionFilter {

public function beforeAction($action)

{

$checker = new AgreementChecker(); if (!$checker->isAllowed()) {

Yii::$app->response->redirect(['/content/agreement'])->send(); return false;

}

return true;

}

}

You can attach filters to any controller or module. To specify the list of necessary routes, just use the only or except options. For example, we apply our filter only for the index action of the controller:

public function behaviors()

{

return [

[

'class' => AgreementFilter::className(),

' only' => ['index'],

],

];

}

Note

Do not forget to return a true value in the success case from the beforeAction method. Otherwise, the controller action will not be executed.

See also

[For more information about filters, refer to http://www.yiiframework. c om/doc -2.0/guide-structure -](http://www.yiiframework.com/doc-2.0/guide-structure-filters.html) filters.html.

For build-in cache and access control filters, refer to:

* <http://www.yiiframework.com/doc-2.0/guide-caching-http.html>
* [http://www.yiiframework.com/doc-2.0/guide-securityauthorizat.ion.ht.ml](http://www.yiiframework.com/doc-2.0/guide-securityauthorization.html)
* The Creating model behaviors recipe