How it works...

At first, we added a piece of code to our controller, which implements our custom filter:

public function behaviors()

{

return [

'access' => [

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

],

];

}

By default, the filter applies to all actions of the controller, but we can specify actions for which it will be applied, or even exclude actions from our filter.

You have two actions inside it—beforeAction and afterActions. The first one runs before the controller’s actions and the next one after.

In our simple example, we defined a condition which doesn’t allow access to website if the time is earlier than 10 AM, and in the after method we just run a trace method if the current path is test/index.

You can see the result in the debugger, in the log section:



In real applications, filters are more complex and also, Yii2 provides a lot of built-in filters, such as core, authentication, content negotiator, HTTP cache end, and so on.

See also

For further information, refer to [http://www.yiiframework.com/doc-2.0/guidestrurtnre-filters.html](http://www.yiiframework.com/doc-2.0/guidestructure-filters.html).