Filters

What are Filters

Similar to Controller interceptors

Can be applied across a whole group of controllers, a URI space or to a specific action.

Creating Filters

Create filters in the `grails-app/conf` directory

```
> grails create-filters
```

Class name ends in Filters.

In the class, define a code block called `filters`

Creating Filters

```
Class QuestionFilters {
    def filters ={
        all(controller:'*', action:'*') {
        // interceptor definitions
        }
    }
}
```

All is the name of the filter, can be anything

Filters are applied in order, top down

Scope

The scope of the filter can be one of the following things:

- A controller and/or action name pairing with optional wildcards
- A URI, with Ant path matching syntax

Rule Attributes

controller - `bookController(controller: 'book')`

controllerExclude - `action: 'b*', actionExclude: 'bad*',`

action - `saveAction(action: 'save', find: true)`

actionExclude - `notSaveAction(actionExclude: 'Save')`

regex (true/false)

uri - `/book/**`

uriExclude - `/animals/**`

find (true/false) - `saveInActionName(action:'*save*', find:true)`

invert (true/false) - `notBook(controller: 'book', invert: true)`

Filter Types

`before:` before the action is called

`after:` after the action is called

`afterView:` after the view is rendered

There is no equivalent for afterView in Controller interceptors

Variables & Scope

Filters have access to all the same variables as controllers and tag libraries

Only have support redirect and render

Workshop

Create a filters class named `com.opi.DownVote` in `grails-app/conf` directory

The filter should only apply to the `VoteController` and the action `voteDownAnswer`

The filter should add a flash.message of `"your a down voter"`

Chaining Filters

You can control the order in which your Filter Classes are executed

Chaining Filters

```
Class ImportantFilters {
    dependsOn = [SuperImportant]
    filters = {
        filter1 (uri: "/*") {...}
        filter2 (uri: "/*") {...}
    }
}
Class SuperImportant {
    filters = {
        filter3 (uri: "/*") {...}
    }
}
```

Order will be filter3, filter1, filter2