



Filters

Table of Contents

- Filters
- Applying Filters
- Filter Types

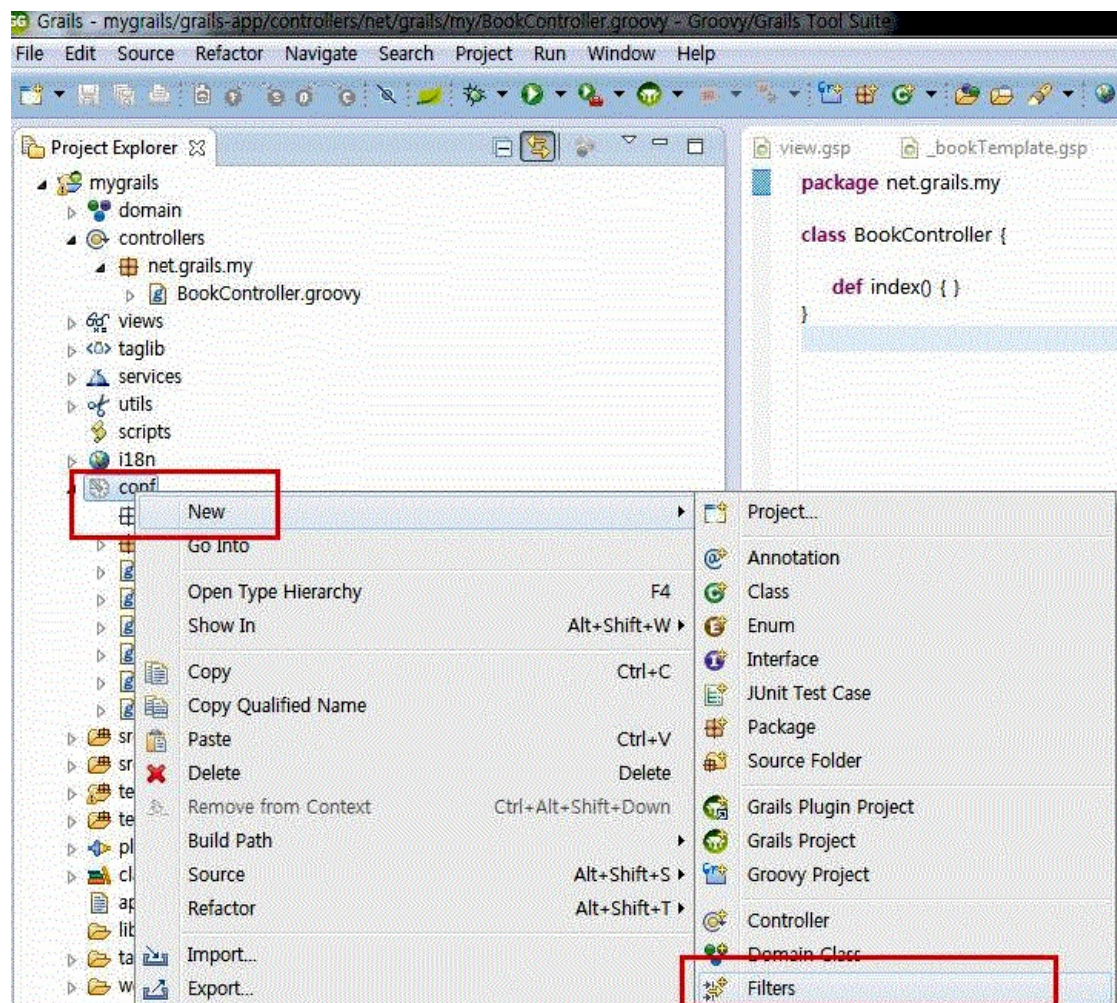
Grails 컨트롤러(controllers)는 잘 만들어진 인터셉터를 지원한다. 인터셉터는 적은 수의 컨트롤러에는 정말 유용하지만 대형 어플리케이션에서 관리하는 것은 매우 어렵다. 반면에 필터는 컨트롤러, URI 공간(space), 액션들의 그룹에 적용될 수 있다. 필터는 주 컨트롤러 로직과 분리하여 쉽게 끼워 넣고 plug-in 관리할 수 있고 보안, 로깅, 등에 필요한 크로스커팅(cross cutting)을 지원한다.

Applying Filters

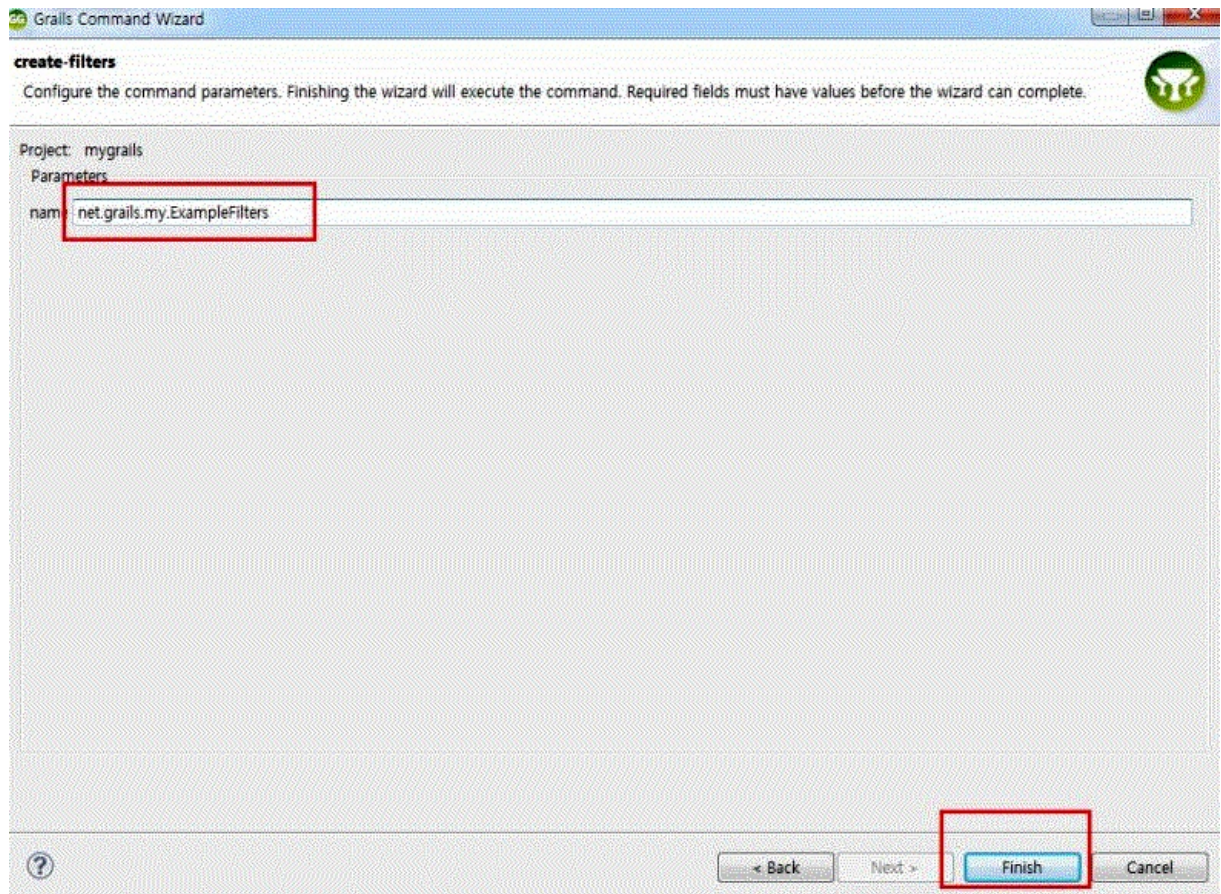
필터를 만드는 방법은 Convention에 따라 grails-app/conf 디렉토리에 Filter로 끝나는 클래스를 만들면된다. 이 클래스 안에 filters 라고 불리는 코드 블록을 만들고 이 블록에 필터의 내용을 정의한다.

아래와 같이 net.grails.my.ExampleFilters 를 생성한다.

conf - 우측 마우스 - New - Filters 클릭한다.



Filter 이름을 입력후 Finish 클릭한다



filters 블록안의 필터들에는 이름과 스코프를 지정할 수 있다.

ExampleFilters 클래스를 보면

all 이라는 메소드가 이름이고, 스코프는 네임드 파라미터인 controller:"*", action:"*" 를 의미한다.

all 이라는 이름의 필터는 모든 클래스와 모든 액션에서 동작한다는 의미이다.

ExampleFilters의 내용을 아래와 같이 변경한다.

```
1 package net.grails.my
2
3 class ExampleFilters {
4
5     def filters = {
6         all(controller: '*', action: '*') {
7             before = {
8                 println "before Controller : " + "아이디는 " + params.id + "입니다. "
9             }
10        }
11    }
12 }
```

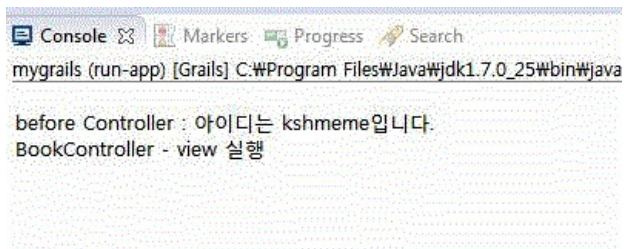
BookController의 내용을 아래와 같이 변경한다.

```
1 package net.grails.my
2
3 class BookController {
4
5     def index() { }
6
7     def view = {
8
9         println "BookController - view 실행"
10        render "Book Controller : view 입니다."
11    }
12 }
```

확인주소 : <http://localhost:8080/mygrails/book/view?id=kshmememe>

실행후 Console 을 확인해보면.

Filter의 내용이 먼저 출력되고 그 다음에 Controller의 내용이 출력되는 것을 확인할 수 있다



all filter는 모든 컨트롤러와 액션에 적용되는 필터이다.

몇개의 필터 예제를 보자

모든 컨트롤러와 액션에 적용되는 필터

```
1 | all(controller:'*', action:'*') {  
2 | }
```



BookController에만 적용되는 필터

```
1 | justBook(controller:'book', action:'*') {  
2 | }
```



URI에 적용되는 필터

```
1 | someURIs(uri:'/book/*') {  
2 | }
```



모든 URI에 적용되는 필터

```
1 | allURIs(uri:'/*') {  
2 | }
```



Filter Types

필터의 바디에 정의할 수 있는 인터셉터의 타입은 다음과 같다

- before - 액션이 실행되기 전에 실행된다. false를 반환하면 뒤따르는 모든 필터와 액션이 실행되지 않음
- after - 액션이 실행된 후에 실행된다. 첫번째 파라미터로 뷰 모델을 넘겨 받음
- afterView - 뷰가 렌더링된 후에 실행

ExampleFilters의 내용을 아래와 같이 변경한다.

```
1 | package net.grails.my  
2 |  
3 | class ExampleFilters {  
4 |  
5 |     def filters = {  
6 |         loginCheck(controller:'*', action:'*') {  
7 |             before = {  
8 |  
9 |                 if(!session.user && !actionName.equals('login') && !actionName.equals("logged")) {  
10 |                     println "login page로 이동합니다."  
11 |                     redirect(action:'login')  
12 |                 }  
13 |             }  
14 |         }  
15 |     }  
16 | }
```



```

12         return session.login?
13     }
14     }
15     }
16     }
17 }

```

BookController의 내용을 아래와 같이 변경한다.

```

1 package net.grails.my
2
3 class BookController {
4
5
6     def view = {
7
8         println "BookController - view 실행"
9         render "Book Controller : view 입니다."
10    }
11
12    def login = {
13
14        def html = "id : <input type='text' /> <br /> pw : <input type='password' /> "
15        html += " <br /> <input type='button' value='로그인' />"
16
17        render html
18    }
19
20    def logged = {
21        session.user = params.id
22
23        render "사용자는 " + session.user + "입니다."
24    }
25
26    def logout = {
27        session.user = null
28        render "사용자는 " + session.user + "입니다."
29    }
30 }
31
32
33
34

```

확인주소 : <http://localhost:8080/mygrails/book/view?id=kshmememe>
<http://localhost:8080/mygrails/book/login>

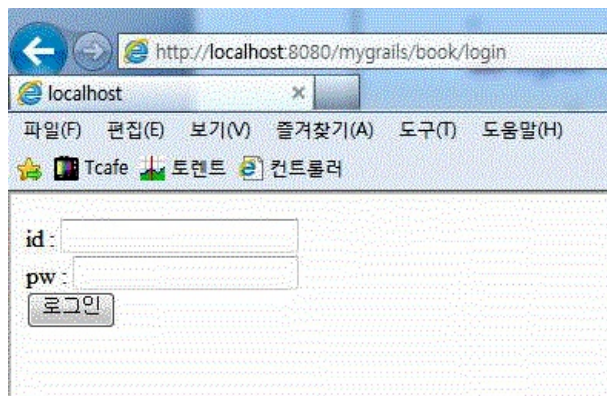
<http://localhost:8080/mygrails/book/logined?id=kshmememe>
<http://localhost:8080/mygrails/book/logout>

실행 후

<http://localhost:8080/mygrails/book/view?id=kshmememe>

접속 해보면

session 에 user 정보가 없으므로 login 페이지로 이동하는 걸 확인할 수 있다.



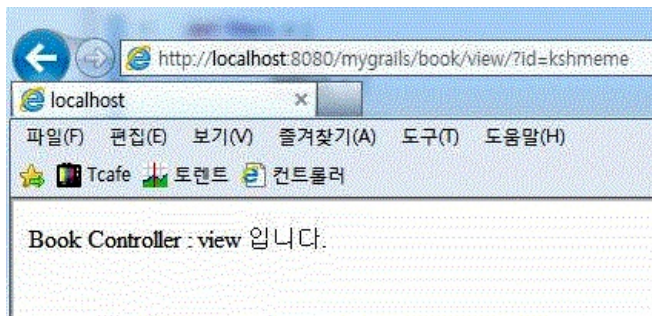
테스트로 session 에 user정보를 넣는 페이지인

<http://localhost:8080/mygrails/book/logined?id=kshmememe>

로 이동을 한다.(로그인 프로세스를 실행 했다는 가정하에 session 정보를 저장)

그 이후에

http://localhost:8080/mygrails/book/view/?id=kshmemem 로 다시 접근을 해보면 session에 user 정보가 있기 때문에 아래처럼 정상적으로 페이지가 뜨는 걸 확인할 수 있다.



http://localhost:8080/mygrails/book/logout 에 접속해서 session을 지우고(로그아웃) 다시 http://localhost:8080/mygrails/book/view/?id=kshmemem 에 접속하면 http://localhost:8080/mygrails/book/login 페이지로 이동 하는 걸 확인할 수 있다.

보통의 인증 케이스를 이런식으로 구현 할 수 있다.
filter에서 리턴이 false 이면 액션이 실행되지 않는 다는 점도 기억한다.

Filter Capabilities

필터는 controllers, 태그 라이브러리(tag libraries), 어플리케이션 컨텍스트에서 사용되는 모든 프로퍼티들을 지원한다:

request - HttpServletRequest 객체
response - HttpServletResponse 객체
session - HttpSession 객체
servletContext - ServletContext 객체
flash - flash 객체
params - 요청 파라미터 객체
actionName - 가로챈 액션의 이름
controllerName - 가로챈 컨트롤러의 이름
grailsApplication - 현재 실행중인 Grails 어플리케이션
applicationContext - ApplicationContext 객체

하지만 필터는 컨트롤러나 태그 라이브러리들에서 사용 가능한 모든 메소드들이 아니라 일부분만 지원한다:

redirect - 컨트롤러와 다른 액션으로 리다이렉트시키기 위해 사용
render - 사용자 정의 응답을 렌더링하기 위해 사용한다

관련 키워드

[Filters](#)
