

Intercepting Filter

Pola Desain Perangkat Lunak
Tjatur Kandaga G.
Fakultas IT - UK Maranatha



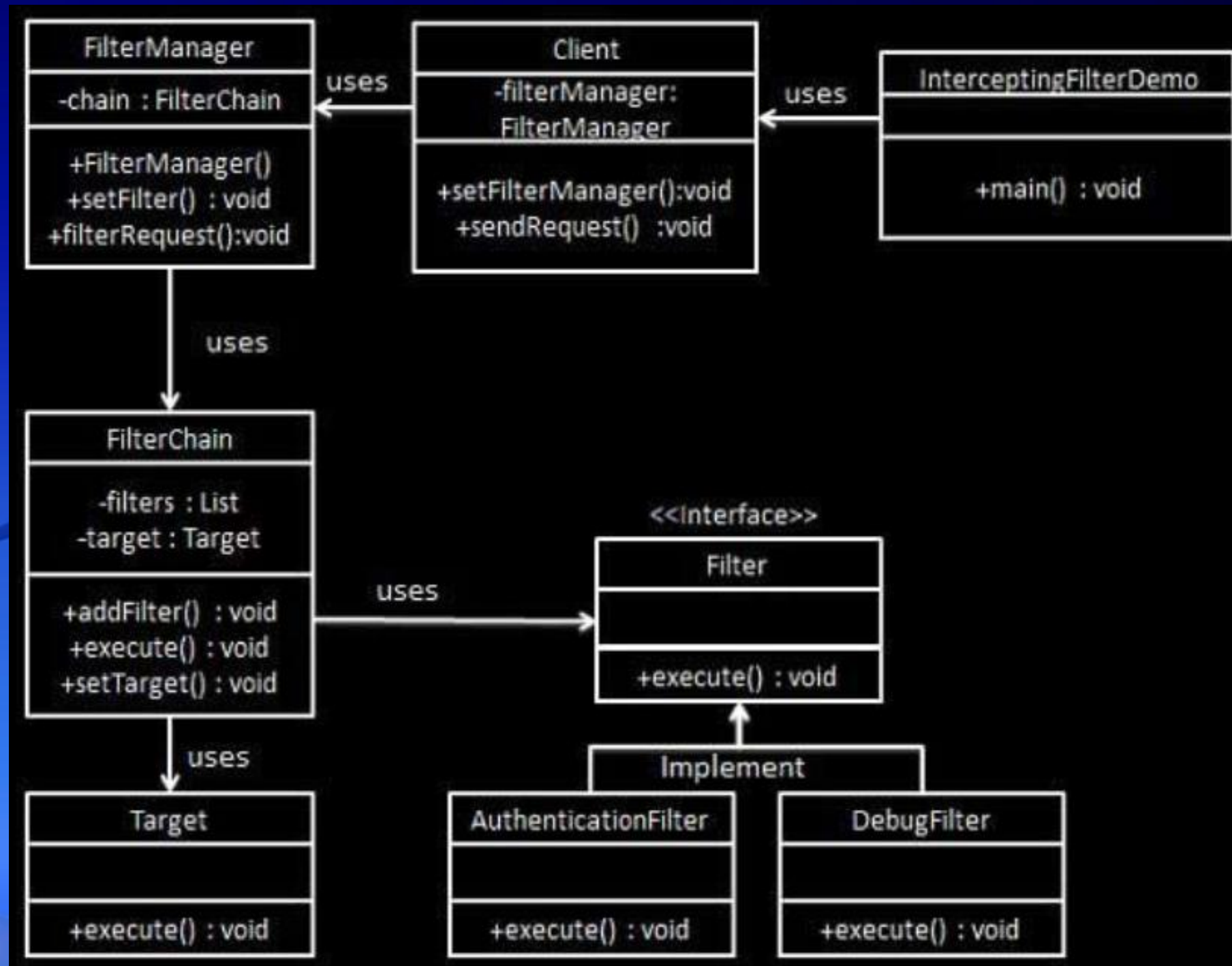
Mengenal Intercepting Filter

- Intercepting filter pattern digunakan ketika kita membutuhkan pre-processing / post-processing bersamaan dengan request atau response dari aplikasi.
- Filter-filter didefinisikan dan diterapkan pada sebuah request sebelum request tersebut diteruskan ke aplikasi targetnya.
- Filter-filter dapat melakukan authentication / authorization / logging atau tracking request kemudian baru mengirimkan request tersebut ke handler-nya.

Mengenal Intercepting Filter

- Entitas-entitas yang terlibat dalam design pattern ini:
 - Filter: melakukan proses sebelum atau setelah request diproses oleh handler.
 - Filter Chain: berisi banyak filter yang akan dijalankan pada target sesuai urutan tertentu.
 - Target: request handler.
 - Filter Manager: mengatur filter-filter dan Filter Chain.
 - Client: yang meminta request ke target.

Diagram Kelas Intercepting Filter



Contoh Intercepting Filter

```
1 package edu.maranatha.pdpl;  
2  
3 public interface Filter {  
4     public void execute(String request);  
5 }
```

```
3 public class AuthenticationFilter implements Filter {  
4     public void execute(String request) {  
5         System.out.println("Authenticating request: " + request);  
6     }  
7 }
```

```
3 public class DebugFilter implements Filter {  
4     public void execute(String request){  
5         System.out.println("request log: " + request);  
6     }  
7 }
```

Contoh Intercepting Filter

```
3 public class Target {  
4     public void execute(String request){  
5         System.out.println("Executing request: " + request);  
6     }  
7 }
```

```
6 public class FilterChain {  
7     private List<Filter> filters = new ArrayList<Filter>();  
8     private Target target;  
9     public void addFilter(Filter filter){  
10         filters.add(filter);  
11     }  
12     public void execute(String request){  
13         for (Filter filter : filters) {  
14             filter.execute(request);  
15         }  
16         target.execute(request);  
17     }  
18     public void setTarget(Target target){  
19         this.target = target;  
20     }  
21 }
```


Contoh Intercepting Filter

```
3 public class FilterManager {  
4     FilterChain filterChain;  
5     public FilterManager(Target target){  
6         filterChain = new FilterChain();  
7         filterChain.setTarget(target);  
8     }  
9     public void setFilter(Filter filter){  
10         filterChain.addFilter(filter);  
11     }  
12     public void filterRequest(String request){  
13         filterChain.execute(request);  
14     }  
15 }
```

Contoh Intercepting Filter

```
3 public class Client {  
4     FilterManager filterManager;  
5     public void setFilterManager(FilterManager filterManager){  
6         this.filterManager = filterManager;  
7     }  
8     public void sendRequest(String request){  
9         filterManager.filterRequest(request);  
10    }  
11 }
```


Contoh Intercepting Filter

```
3 public class InterceptingFilterDemo {  
4     public static void main(String[] args) {  
5         FilterManager filterManager =  
6             new FilterManager(new Target());  
7         filterManager.setFilter(new AuthenticationFilter());  
8         filterManager.setFilter(new DebugFilter());  
9         Client client = new Client();  
10        client.setFilterManager(filterManager);  
11        client.sendRequest("HOME");  
12    }  
13 }
```

Authenticating request: HOME
request log: HOME
Executing request: HOME

Contoh Intercepting Filter (C#)

```
7 namespace InterceptingFilterCS
8 {
9     interface Filter
10     {
11         void execute(String request);
12     }
13 }
14
```

```
7 namespace InterceptingFilterCS
8 {
9     class DebugFilter : Filter
10     {
11         public void execute(String request)
12         {
13             Console.WriteLine("Request log: " + request);
14         }
15     }
16 }
```

Contoh Intercepting Filter (C#)

```
7 namespace InterceptingFilterCS
8 {
9     public class AuthenticationFilter : Filter
10     {
11
12         public void execute(String request)
13         {
14             Console.WriteLine("Authenticating request: " + request);
15         }
16     }
17 }
```

```
7 namespace InterceptingFilterCS
8 {
9     class Target
10     {
11         public void execute(String request)
12         {
13             Console.WriteLine("Executing request: " + request);
14         }
15     }
16 }
```


Contoh Intercepting Filter (C#)

```
7 namespace InterceptingFilterCS
8 {
9     class FilterChain
10    {
11        private List<Filter> filters = new List<Filter>();
12        private Target target;
13
14        public void addFilter(Filter filter)
15        {
16            filters.Add(filter);
17        }
18
19        public void execute(String request)
20        {
21            foreach (Filter filter in filters)
22            {
23                filter.execute(request);
24            }
25            target.execute(request);
26        }
27
28        public void setTarget(Target target)
29        {
30            this.target = target;
31        }
32    }
33 }
```

Contoh Intercepting Filter (C#)

```
7 namespace InterceptingFilterCS
8 {
9     class FilterManager
10    {
11        FilterChain filterChain;
12
13        public FilterManager(Target target)
14        {
15            filterChain = new FilterChain();
16            filterChain.setTarget(target);
17        }
18
19        public void setFilter(Filter filter)
20        {
21            filterChain.addFilter(filter);
22        }
23
24        public void filterRequest(String request)
25        {
26            filterChain.execute(request);
27        }
28    }
29 }
```

Contoh Intercepting Filter (C#)

```
7 namespace InterceptingFilterCS
8 {
9     class Client
10    {
11        FilterManager filterManager;
12
13        public void setFilterManager(FilterManager filterManager)
14        {
15            this.filterManager = filterManager;
16        }
17
18        public void sendRequest(String request)
19        {
20            filterManager.filterRequest(request);
21        }
22    }
23 }
```


Contoh Intercepting Filter (C#)

```
7 namespace InterceptingFilterCS
8 {
9     class Program
10     {
11         static void Main(string[] args)
12         {
13             FilterManager filterManager = new FilterManager(new Target());
14             filterManager.setFilter(new AuthenticationFilter());
15             filterManager.setFilter(new DebugFilter());
16             Client client = new Client();
17             client.setFilterManager(filterManager);
18             client.sendRequest("HOME");
19         }
20     }
21 }
```

C:\Windows\system32\cmd.exe

```
Authenticating request: HOME
Request log: HOME
Executing request: HOME
Press any key to continue . . .
```

The background of the slide is a photograph of a wind farm, showing several wind turbines with three blades each, mounted on tall metal towers. The image is covered with a semi-transparent blue filter. Overlaid on this background is the text 'Intercepting Filter' in a large, bold, yellow font. Below it, the text 'add pre/post-' is in a smaller yellow font, followed by 'processings to' and 'client request.' in the same large, bold, yellow font as the first line.

Intercepting Filter
add pre/post-
processings to
client request.