Filters/Interceptors
==========
   It can be used for PreProcessing and PostProcessing of the request before they reach the target resource in the
   web application.

Areas of Application
==========
      Logging, Security,Altering Request information, Compressing response,encryption of response,authenication etc...

Filters Concpets introduced in Servlet 2.3V

FilterApi contains 3 interfaces
1. Filter(I)
2. FilterConfig(I)
3. FilterChain(I)

To create a filter we need to implement Filter(I)
==========
```
public interface Filter {
  public void init(FilterConfig config) throws ServletException;
  public abstract void doFilter(ServletRequest request, ServletResponse response,
FilterChain chain) throws IOException,
                                                                    ServletException;
  public void destroy();
}
```
refer: FilterApp

Behind the Scenes
==========
1. Whenever we are sending the request to TargetServlet, web container will check if any filter is configured for this servlet
     or not.
2. If any Filter is configured, web container forwards the request to Filter instead of Servlet.
3. After completing the Filter logic, Filter forwards the request to TargetServlet.
4. After processing the TargertServlet, the response will be forwarded to Filter instead of browser.
5. After executing the Filtering logic, filter forwards the total response to the browser.

eg: http://localhost:9999/Filterapp/test

output
```
This line is added by DemoFilter before processing the request...
This is Target Servlet...
This line is added by DemoFilter after processing the request...
```

FilterMapping to particular url-pattern
==========
```
<filter-mapping>
    <filter-name>DemoFilter</filter-name>
    <url-pattern>/test</url-pattern>
</filter-mapping>
```

FilterMapping to Total WebApplication
==========
```
<filter-mapping>
 <filter-mapping>
    <filter-name>DemoFilter</filter-name>
```

```
    <url-pattern>*</url-pattern>
</filter-mapping>
```

FilterChaining
==========
Dividing the Request Pre-Processing tasks into mulitple filters

Webcontainers rule for ordering the filter in FilterChain
==========
It is depended on the container can't be predicted by the user.

What is the difference b/w Filter doFilter() and FilterChain doFilter()?
Filter===> doFilter()
doFilter(ServletRequest, ServletResponse, FilterChain)throws SE,IOE
Total Filtering logic(pre+post) processing logic.
It is a call back method as it is called by the container automatically.

FilterChain===>doFilter()
doFilter(ServletRequest,ServletResponse) throws IOException, ServletException;
Forwarding the request to another filter/servlet.
It is not a callback method,because we have to call explicitly.

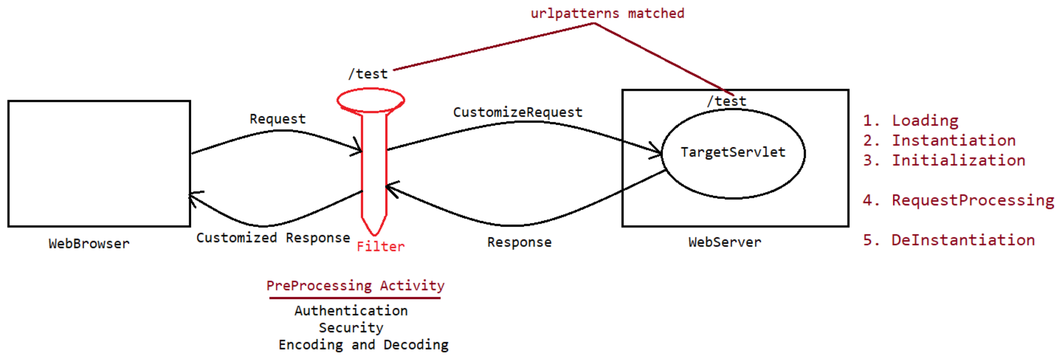
Wrappers and Listeners
==========
Sometimes we need to alter the request and response object inside filters, to do this we need to use "Wrappers" class.

eg1:
within the filter, we have to convert resume information from wordformat to pdfformat.(upload)

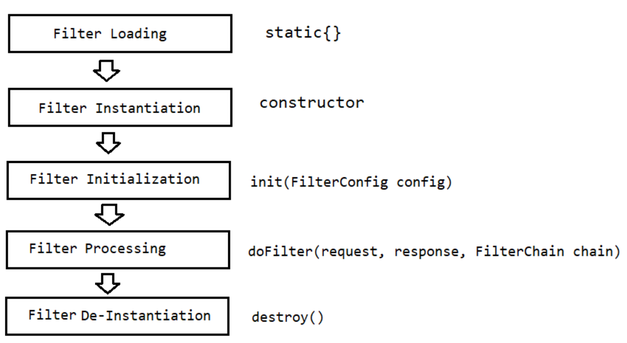
eg2:
within the filer, we have to compress the response and that compressed response we can send to browser so that
download time can be reduced.

There are 2 types of Wrappers
a. Request Wrappers
b. Response Wrappers

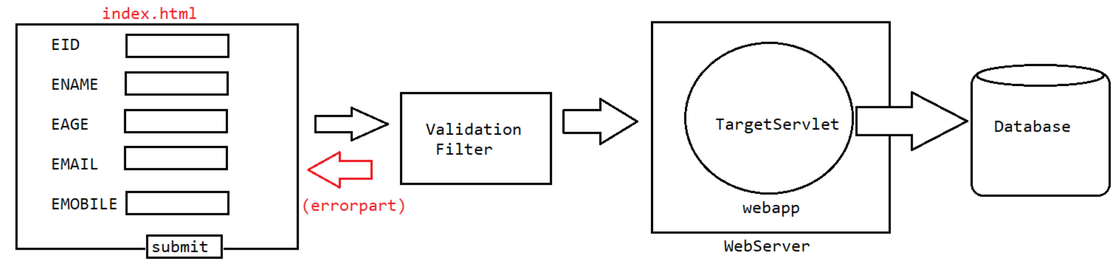
note:
1. No configuration is required for wrappers.
2. Annotation used to represent filter is :@WebFilter(urlPatterns={})



Filter Life Cycle Actions



For filters we don't need to give <load-on-startup> value, It is autoloaded, instantiated and initalized...




```
eid is empty     => EmployeeId is required
ename is empty   => EmployeeName is required
eage  is empty   => EmployeeAge is required
email is empty   => EmployeeEmail is requried
mobile is empty  => EmployeeMobile is required
```

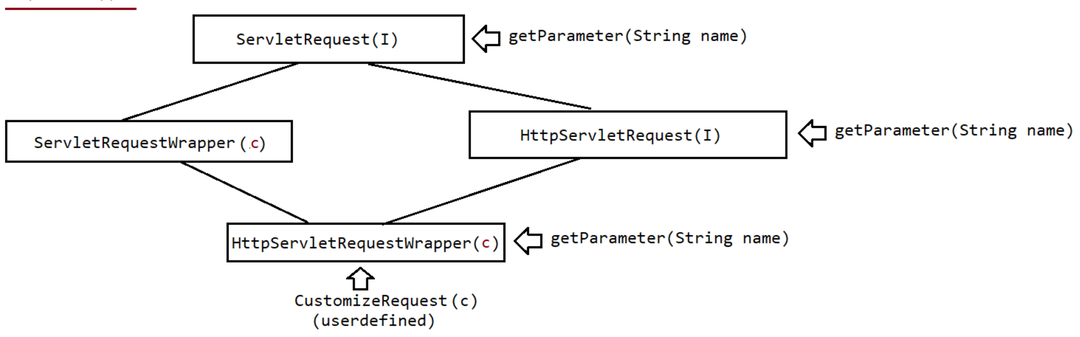
First level checking

```
    id     => starts with iNeuron-
    eage   => with in 20 to 30
    email  => end with @ineuron.ai
    mobile => 91-XXXXXXXX
```

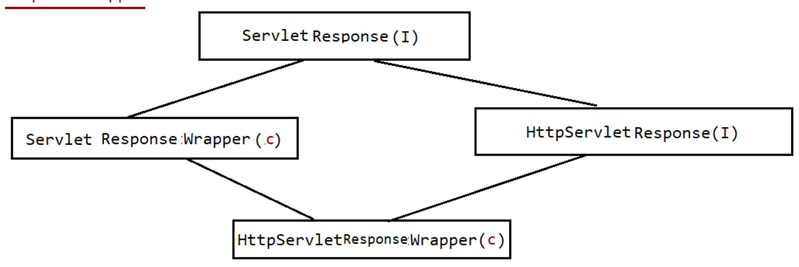
Second level checking

In which order filter gets executed in FilterChaining?

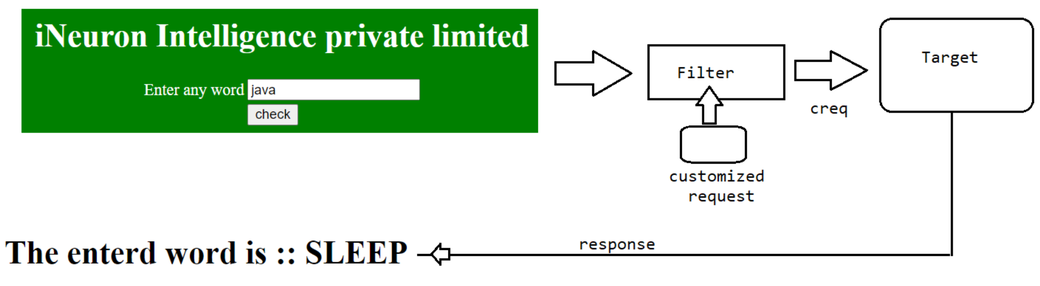
RequestWrapper



ResponseWrapper



HttpServletRequestWrapper



HttpServletResponseWrapper