**Объект ФИЛЬТР:**  
\* похож на middleware  
\* это серверный компонент, предн для предварит обработки запроса, созд. при старте сервера  
\* обычный класс, реализ интерфейс Filter (init, destroy, doFilter)  
\* они записаны в web.xml => вся инфа о фильтре хранится в контексте прил.   
(там же мэппинг фильтров. Один и тот же класс фильтра мб исп в разных цепочках)

**Методы:**

- init() – при иниц фильтра, чтобы фильтр мог восст нек данные, кот он где-то сохранил  
- destroy() – чтобы мог данные записать, чтобы потом использ при восст  
- doFilter() – метод, вызывается контейнером сервлетов при поступлении запросов позв вызвать doFilter и передать парам: запрос, ответ, filterChain (Для образования цепочки фильтров)  
**Цепочка фи** – фильтры, привяз к 1 ресурсу (ресурс: сервлет, jsp) и они будут посл-но выполняться. Обычно фильтр завершается вызовом doFilter след. фильтра в цепочке. Последний фильтр вызывает service() cервлета, к-му передает req, res.

И этот стек разворачивается в другую сторону и обрабатывает уже res.

**Слушатели событий** – спец механизм обработки событий на стороне С. М создать классы, кот рализуют этот ин-с и записать инфу об этих классах в web.xml. Тогда С при старте загрузит эти листенеры и мы можем обраб различ событий, связ с этими фильтрами.

Есть события, связ с созданием/разруш контекста, измен атрибутов… Те же события можем подловить и обработать у сессии (создание, измен ее атрибутов). Чтобы создать обработчик таких событий, м создать класс , кот реализует 1/неск интерфейсов (смотря на какие события срабатывает). Чтобы С эти объекты мог поднять и зарегистрировать в контексте. Для этого пропишу их в web.xml C сразу поднимает фильтры и листенеры.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**JDBC** – java db community – прогр ин-с, обесп взаимод прогр кода на Java с реляц БД.

Ин-с делится на 2 компонента:  
\* стандартная, общая для всего ПО, кот использ ин-с (.NET, Java пофиг)  
\* учит особености каждой СУБД (Драйвер бд – составная часть какой-то субд чаще, ее разрабатывают разработчики субд)

JDBC – jar-файл. Он кроссплатформенный, написан на джаву, и работает на всех платформах, где работает jvm. Можно смело переностиь на уровне байт-кода.

JDBC API – набор программных ин-сов (поименованный набор сигнатур). А драйвер – классы, кот реализуют этот объект. **Connection** – Объект, предост нам драйвер, но этот объект реализует ин-с Connection, кот есть в jsbc. **Драйвер** – реализация этих ин-сов. ПОграничным явл объекь Connection, кот мы извлекаем из этого драйвера и начинаем с ним работать.

**Чтобы соединиться с сервером СУБД:**  
\* ip, где сервер или символический адрес (dns, netBIOS, hosts, gethostbyname помогут)  
\* порты 1521 oracle, 1433 mssql  
\* user name, password  
\* имя инстанса, к к-му мы коннектимся  
\* тип драйвера: тонкий или толстый (для oracle)

Все эти параметры в объекте Connection

Дальше м делать статические, динамические запросы и вызывать удал.процедуры, кот есть на стороне Сервера