5. В JSP предусмотрены следующие области действия переменных (объектов):  
  **request** - область действия запроса;  
  **session** - область действия сессии;  
  **application** - область действия приложения;  
  **page** - область действия страницы.  
   
При использовании области действия запроса **request** объект будет доступен на текущей JSP странице, странице пересылки (при использовании jsp:forward) или на включаемой странице (при использовании jsp:include).  
  
В случае области действия сессии **session**, объект будет помещен в сеанс пользователя. Бин будет доступен на всех JSP страницах и будет существовать пока существует сессия пользователя, или он не будет из нее принудительно удален.  
   
Бин с областью действия **application** доступен для всех пользователей на всех JSP страницах и существует на протяжении всей работы приложения или пока не будет удален принудительно и контекста приложения.  
  
При использовании области действия **page** объект доступен только на той странице, где он определен. На включаемых (jsp:include) и переадресуемых (jsp:forward)  страницах данный объект уже не доступен.

11 лаба

1. **Интерфейс ServletContext**

Определяет ряд методов, которые сервлет использует, чтобы связаться с его контейнером сервлета, например, получить тип MIME файла, диспетчеризировать запросы, или записать в файл журнала.

**2.контекст приложения**– это, прежде всего, информация, характеризующая само web-приложение и среду в которой оно работает. Эту информацию можно разделить на две части: статическую и динамическую.

Атрибуты являются динамически элементами контекста и могут быть установлены, изменены или удалены в методах сервлетов или в скриплетах JSP.

Атрибут - это переменная сервера, которая существует в пределах определенной области

**3.** Следующие методы позволяют получить из контекста сервлета базовую информацию:

String getRealPath(String filename) — определение истинного маршрута файла относительно каталога, в котором сервер хранит документы;

String getServerInfo() — возвращает имя и номер версии контейнера сер- влетов;

String getContextPath() — возвращает имя веб-приложения, находящегося в папке контейнера сервлетов, для Tomcat это будет папка webapps;

String getServletContextName() — возвращает имя веб-приложения, заданное в дескрипторе приложения в теге ;

void log(String msg) — запись лога с сообщением msg;

void log(String msg, Throwable t) — запись лога с текстовым сообщением msg и сообщением, извлекаемым из исключения.

4. **Интерфейс ServletConfig**

Параметры инициализации сервлета содержатся в экземпляре ServletConfig. Получить доступ к нему можно методом getServletConfig(), вызываемым на экземпляре сервлета.

5. Информация от компьютера клиента отправляется серверу в виде объекта запроса типа HttpServletRequest. Данный интерфейс является производным от интерфейса ServletRequest. Используя методы интерфейса ServletRequest, можно получить много дополнительной информации, в том числе и о сервлете и деталях протокола HTTP, закодированной и упакованной в запрос

6. Генерируемые сервлетами данные пересылаются серверу-контейнеру с помощью объектов, реализующих интерфейс ServletResponse, а сервер, в свою очередь, формирует и пересылает ответ клиенту, инициировавшему запрос.

# 7. Принципиальное отличие переадресации от переопределения запроса заключается в том, что передача управления от сервлета А к сервлету В осуществляется через посредника – клиента сервлета A.

# (Forward-переопределение, может выполнится, если другой сервлет находится в рамках этого приложения. Связь с первым сервлетом теряется, ответы поступают напрямую браузеру.)

# RequestDispatcher.forward()

метод делает редирект прямо на сервере. Т.е. запрос просто перенаправляется к другой странице. Браузер клиента об этом ничего не знает. Поэтому и название страницы, и относительные пути в браузере остаются старые.

# (переадресация)HttpServletResponse.sendRedirect()?

метод посылает клиентскому браузеру ответ, в котором говорит о необходимости форварда на другую страницу. Браузер, получая подобный ответ, запрашивает данные с новой страницы.

Т.е. первый вариант форварда проиисходит на сервере, а второй - на клиенте.

8.

9. **Что такое сессия?**

Сеанс есть сессия между клиентом и сервером, устанавливаемая на определенное время, за которое клиент может отправить на сервер сколько угодно запросов. Сеанс устанавливается непосредственно между клиентом и веб-сервером в момент получения первого запроса к веб-приложению. Каждый клиент устанавливает с сервером свой собственный сеанс, который сохраняется до окончания работы с приложением.

10. **Что такое Cookie?**

Cookie - это небольшие блоки текстовой информации, которые сервер посылает клиенту для сохранения в файлах cookies. Клиент может запретить браузеру прием файлов cookies. Браузер возвращает информацию обратно на сервер как часть заголовка HTTP, когда клиент повторно заходит на тот же веб-ресурс.

11. **Для чего нужны Фильтры?**

Реализация интерфейса Filter позволяет создать объект, который перехватывает запрос, может трансформировать заголовок и содержимое запроса клиента. Фильтры не создают запрос или ответ, а только модифицируют их. Фильтр выполняет предварительную обработку запроса, прежде чем тот попадает в сервлет, с последующей (если необходимо) обработкой ответа, исходящего из сервлета. Фильтр может взаимодействовать с разными типами ресурсов, в частности, и с сервлетами, и с JSP-страницами.

12. Жизненный цикл фильтра начинается с однократного вызова метода **init()**, затем контейнер вызывает метод **doFilter()** столько раз, сколько запросов будет сделано непосредственно к данному фильтру. При отключении фильтра вызывается метод **destroy()**.

\_\_\_\_\_\_\_\_\_\_4

Чтобы сохранить значения переменной в текущем сеансе, используется метод setAttribute(String name, Object value) класса HttpSession, прочесть — getAttribute(String name)

Чтобы открыть явный доступ к экземпляру сеанса/сессии пользователя веб-приложения, используются методы getSession(boolean create) или getSession() интерфейса HttpServletRequest. Экземпляр запроса возвращает ссылку на объект сессии. Метод не создает сессию, а только дает доступ с помощью запроса к экземпляру сессии HttpSession, соответствующему данному пользователю.

Вставка Cookie в заголовок HTTP-ответа происходит с помощью метода addCookie

Мы используем request.getParameter() для извлечения параметров запроса (т.е. данных, отправленных путем размещения html-формы). request.getParameter() всегда возвращает значение String, и данные поступают от клиента.