1. **Сервлет** - это класс на языке Java, который расширяет возможности сервера, обрабатывающего запросы и формирующего ответы. Сервлеты обычно используются для обработки запросов HTTP.
2. **Web-сервер** нужен для обработки веб-запросов, хранения веб-страниц и предоставления ответов на запросы клиентов.
3. Сервлет создается на сервере, который обрабатывает запросы клиентов, выполняет некоторую работу и возвращает ответ. Клиентская сторона отправляет запросы и получает ответы от сервера.
4. Методы **doGet** и **doPost** отличаются тем, как они получают данные от клиента. **doGet** получает данные из строки запроса URL, в то время как **doPost** получает данные из тела запроса.
5. Методы жизненного цикла сервлета вызываются в следующем порядке: **init()**, **service()**, **destroy()**.
6. Имя и URL сервлета обычно определяются в файле конфигурации web.xml.
7. Значение параметра формы в сервлете можно получить с помощью метода **request.getParameter(“paramName”)**.
8. Сервлет возвращает результат своей работы на сторону клиента с помощью объекта **HttpServletResponse**.
9. Развертывание сервлета обычно включает в себя его компиляцию, упаковку в WAR-файл, развертывание WAR-файла на сервере и его регистрацию в файле конфигурации web.xml.
10. Интерфейс **ServletContext** предоставляет информацию о контексте веб-приложения, в котором работает сервлет.
11. **Атрибут контекста** - это объект, который сервлеты могут использовать для обмена информацией.
12. Из контекста сервлета можно получить информацию о сервере, веб-приложении и других сервлетах.
13. Интерфейс **ServletConfig** предоставляет конфигурационную информацию для сервлета.
14. Интерфейс **HttpServletRequest** содержит все информацию о HTTP-запросе, который был отправлен серверу.
15. Интерфейс **HttpServletResponse** используется для отправки HTTP-ответов клиенту.
16. Переадресация запросов и перенаправление откликов обычно выполняются с помощью методов **response.sendRedirect()** и **request.getRequestDispatcher().forward()** соответственно.
17. Данные приложения между запросами можно сохранить с помощью сессий или cookies.
18. Объект **HttpSession** можно получить с помощью метода **request.getSession()**. Он содержит информацию о сессии пользователя.
19. **Cookie** - это небольшой файл, который сервер веб-сайта отправляет в браузер пользователя. Он может быть создан с помощью нового объекта Cookie и добавлен в ответ с помощью **response.addCookie()**.
20. Интерфейс **Filter** используется для предварительной обработки запросов и/или постобработки ответов.
21. Методы жизненного цикла фильтра включают **init()**, **doFilter()** и **destroy()**.