

Сервлет – это web-компонент, представляющий собой класс Java,  
предназначенный для динамического формирования содержимого ответов на запросы клиентского приложения (обычно web-браузера).  
Сервлеты размещаются на сервере и выполняются в специальной среде – контейнере сервлетов (Servlet engine), – на виртуальной Javaмашине (JVM) сервера. Контейнер сервлетов является составной частью web-контейнера и определяется спецификацией Java Servlet. В  
общем случае сервлеты не зависят от протокола связи, но наиболее  
часто они применяются для обработки http-запросов.  
Количество сервлетов на web-сервере не ограничено. Один сервлет может отвечать на один или несколько типов запросов клиента

**Контейнер** сервлетов (Веб-контейнер) — программа, представляющая собой сервер, который занимается системной поддержкой сервлетов и обеспечивает их жизненный цикл в соответствии с правилами, определёнными в спецификациях

**Java Servlet API** — стандартизированный API, предназначенный для реализации на сервере и работе с клиентом по схеме запрос-ответ.

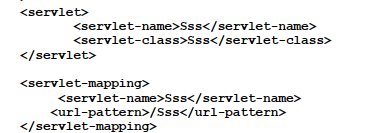
Java Servlet API организован в виде двух пакетов, содержащих два  
набора базовых классов: **javax.servlet** – общие для всех сервлетов интерфейсы и классы; **javax.servlet.http** – расширение пакета  
**javax.servlet**, обеспечивающее поддержку протокола HTTP

Жизненный цикл состоит из 3 методов:init,service,destroy.

Метод **init** вызывается сервером при инициализации сервлета.  
Этот метод позволяет программисту выполнить некоторые действия  
перед началом работы сервлета.

Метод **destroy** тоже вызывается сервером, но при его выгрузке.  
Этот метод используется разработчиком сервлета для выполнения  
действий, связанных с окончанием работы, – освобождение ресурсов,  
закрытие соединений с сервером базы данных и т. п.

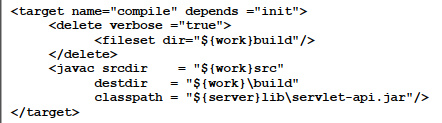
Метод **service** предназначен для обработки запроса клиента. Метод вызывается сервером при получении запроса клиента на вызов  
сервлета. Сервер формирует два параметра. Первый реализует интерфейс **HTTPServletRequest** и используется для того, чтобы получить  
информацию о http-запросе. Второй параметр, реализующий интерфейс **HTTPServletResponse**, дает возможность сервлету формировать  
http-ответ клиенту. В данном примере в функции **service** используется  
вызов метода **getMethod** интерфейса **HTTPServletRequest**. Функция  
**getMethod** позволяет определить тип http-запроса (**get**, **post**, **put**,  
**delete**, **options** и т. д.)



**Web.xml тут мы подключаем сервлеты**

В элементе **<servlet>** содержится информация об именах сервлета (вложенный тег **<servlet-name>**) и классе сервлета (вложенный тег  
**<servlet-class>**).

Элемент **<servlet-mapping>** указывает на суффикс URL (тег **<urlpattern>**), с помощью которого можно вызвать сервлет. Указанная в элементе **<url-pattern>** строка не обязательно должна совпадать с именами сервлета или класса

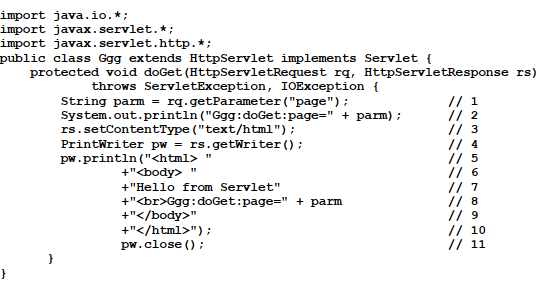


**Build.xml**

Прежде всего, в файле сборки появилась новая цель **compile**, которая использует новую задачу **javaс**, позволяющую откомпилировать все  
java-файлы, которые находятся в директории, указанной атрибутом  
**scrdir**, и поместить результаты компиляции (class-файлы) в директорию,  
указанную атрибутом **destdir**. Для компиляции сервлета необходима  
библиотека, поставляемая вместе с сервером Tomcat. Месторасположение этой библиотеки указывается атрибутом **classpath** задачи **javac**

Класс **HttpServlet** имеет собственную реализацию метода **service**,  
который в зависимости от типа http-запроса передает управление в  
один из следующих переопределяемых методов: **doGet**, **doPost**, **doPut**, **doDelete**, **doHead**, **doOptions**, **doTrace**. Перечисленные методы соответствуют различным типам http-запросов

**Обработка Get**



Для получения значения параметра с  
именем **page** применяется метод **getParameter** интерфейса **HttpServlet  
Request** (строка 1).  
Для вывода данных на web-браузер используется метод **println**класса **PrintWriter** (строки 5–10**)**. Объект **PrintWriter** создается webсервером, а ссылка на него может быть получена с помощью метода  
**getWriter** интерфейса **HttpServletResponse** (строка 4**)**. Для того чтобы  
web-браузер смог правильно распорядиться данными, с помощью метода **setContentType** устанавливается MIME-тип данных (строка 3) ответа.. Завершается вывод вызовом метода **close** объекта  
**PrintWriter** (строка 11).

**Обработка Post**

Разработка сервлета, обрабатывающего http-запрос **POST**, принципиально ничем не отличается от разработки сервлета, обрабатывающего запрос **GET**. Для генерации запроса **POST**, как правило,  
применяется html-форма, а обработка этого запроса осуществляется в  
методе **doPost** вызываемого сервлета.

В лаб работе в форме у нас есть методы ***action*** and ***method***

***Method=post***

Атрибут **action** определяет ссылку, куда должны быть пересланы  
для обработки данные из формы.Если action=”/”данные обрабатываются на этой же странице