

ПРОГРАММИРОВАНИЕ В INTERNET

ОСНОВЫ ПОСТРОЕНИЯ WEB-ПРИЛОЖЕНИЙ

Общие принципы построения web-приложений

- web-ресурсы приложения;
- запросы и ответы;
- соединение (сессия);
- конфигурационный файл приложения;
- контекст приложения;
- фильтры;
- кэш (данных и вывода);
- слушатели событий;
- принципы безопасности.

Web-ресурс =

сущность, **расположенная на стороне сервера и имеющая URL/URI**, к которой можно сделать http-запрос и получить http-ответ. Одно web-приложение представлено одним или более ресурсов.

Виды web-ресурсов

- 1) **статические** – **отправляются** клиенту **без изменения** (html-страницы, рисунки, видео-файлы, ...),
- 2) **динамические** – **динамически** (программно) **формируются** на сервере и отправляются клиенту (сервлеты, JSP, http-обработчики, aspx-страницы,...).

Ресурс может быть статическим относительно сервера и динамическим относительно клиента (html-страницы с JS).

Запрос (Request)

=

серверный объект, который образуется в результате обработки сервером **http-запроса**, поступающего от клиента и передается серверному программному коду для обработки.

Содержит всю информацию из http-запроса: метод, коллекция заголовков, коллекция параметров, поток данных ...

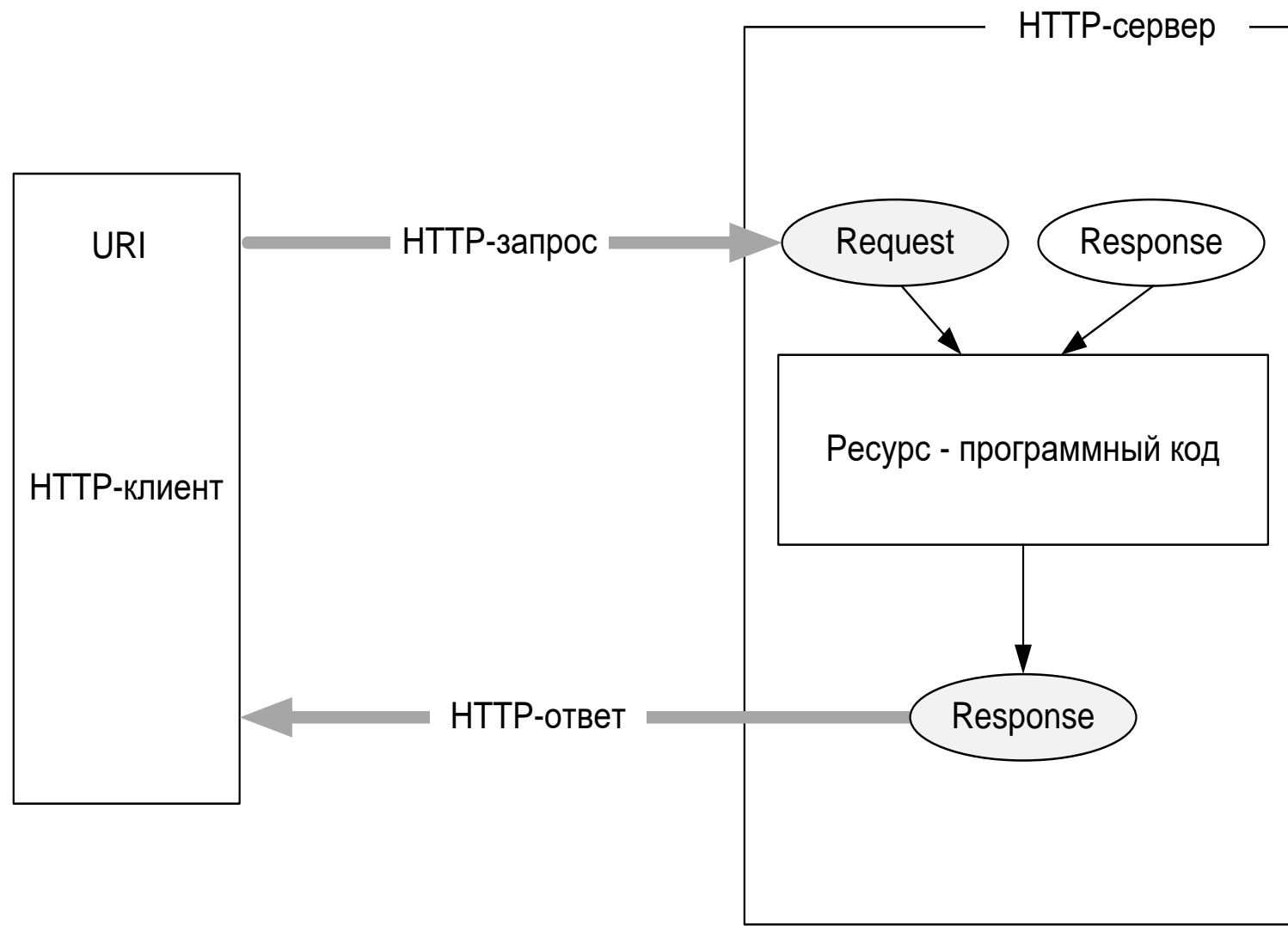
Обычно объект Request предоставляет возможность хранить данные в формате ключ/значение.

Ответ (Response)

=

серверный объект, который автоматически формируется сервером при получении **http-запроса** (одновременно с объектом Request), **заполняется данными** серверным программным кодом, **преобразуется в http-ответ** и **отправляется клиенту**.

Содержит всю информацию, которая должна быть помещена в **http-ответ**: статус, коллекция заголовков, поток данных, ...



Сессия (Session)

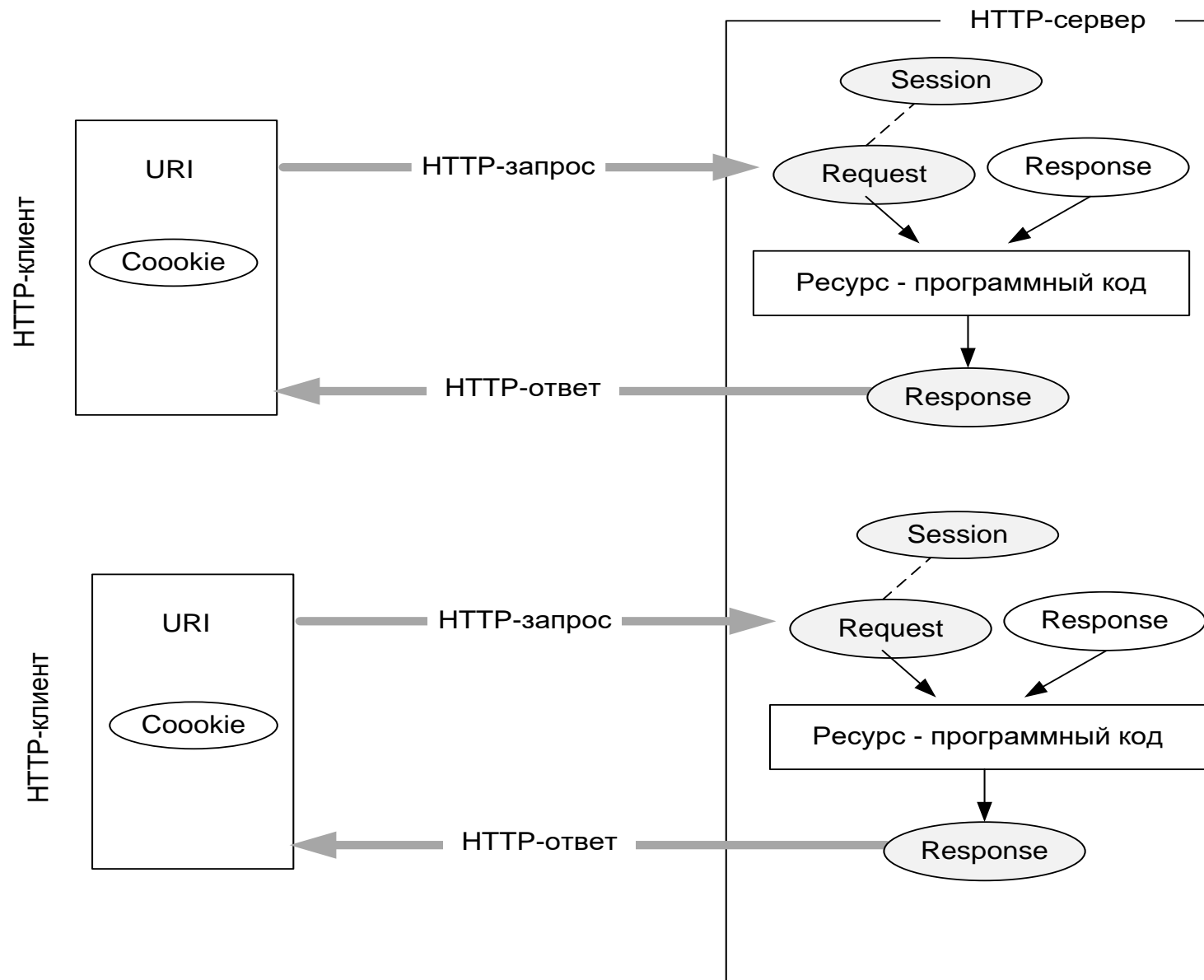
=

серверный объект, хранящий информацию о соединении с клиентом, создается при первом обращении.

Время жизни: **timeout** (системный параметр, обычно равен 10–30 минутам) – **максимальное время между запросами клиента**. Если timeout превышен, то Session разрушается и при следующем запросе создается новый экземпляр.

Каждая сессия имеет собственный идентификатор (Session ID, 16 или более байт). **Каждый Request принадлежит какой-то сессии** (имеет ссылку на объект Session или содержит Session ID).

Обычно объект Session предоставляет приложению возможность хранить данные в формате ключ/значение.



Cookie

=

фрагмент данных,
отправленный web-сервером и
хранимый web-клиентом.

Используется для
аутентификации, хранения
пользовательских
предпочтений, статистики,
информации о сеансе (обычно
Session ID). Обычно имеет имя,
содержащее URL, может иметь
срок действия.

Для создания и пересылки
Cookie применяются заголовки.

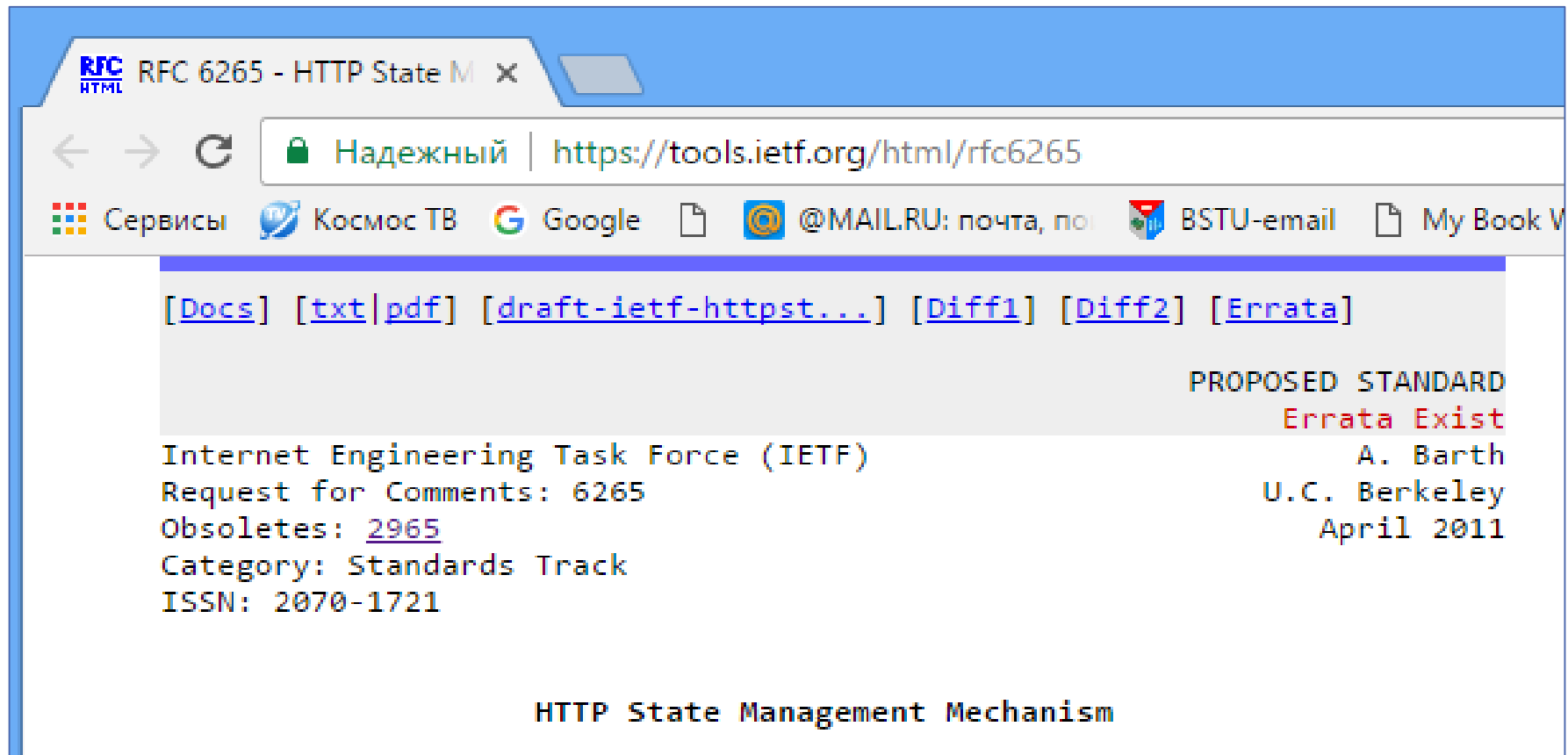
```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html; charset=utf-8
Vary: Accept-Encoding
Server: Microsoft-IIS/8.5
X-AspNetMvc-Version: 5.2
X-AspNet-Version: 4.0.30319
Set-Cookie: ASP.NET_SessionId=imdp40w4hcs55g3jto0aa04b; path=/; HttpOnly
X-Powered-By: ASP.NET
Date: Tue, 14 Feb 2017 21:06:16 GMT
Transfer-Encoding: chunked
```

Заголовок ответа Set-Cookie используется для отправки файла cookie с сервера клиенту (браузеру), чтобы клиент мог отправить его обратно на сервер позже.

Заголовок запроса Cookie содержит ранее сохраненные файлы cookie, связанные с сервером.

```
GET http://80.94.166.212:40001/EGHRGE/Forecast HTTP/1.1
Host: 80.94.166.212:40001
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://80.94.166.212:40001/EGHRGE
Accept-Encoding: gzip, deflate, sdch
Accept-Language: ru-RU,ru;q=0.8,en-US;q=0.6,en;q=0.4
Cookie: ASP.NET_SessionId=imdp40w4hcs55g3jto0aa04b
```

RFC 6265 (стандарт, описывающий Cookie)



Конфигурационны
й файл web-
приложения =

файл содержащий **системные
параметры приложения**,
обычно в XML-формате, служит
**основой для создания
контекста** web-приложения.

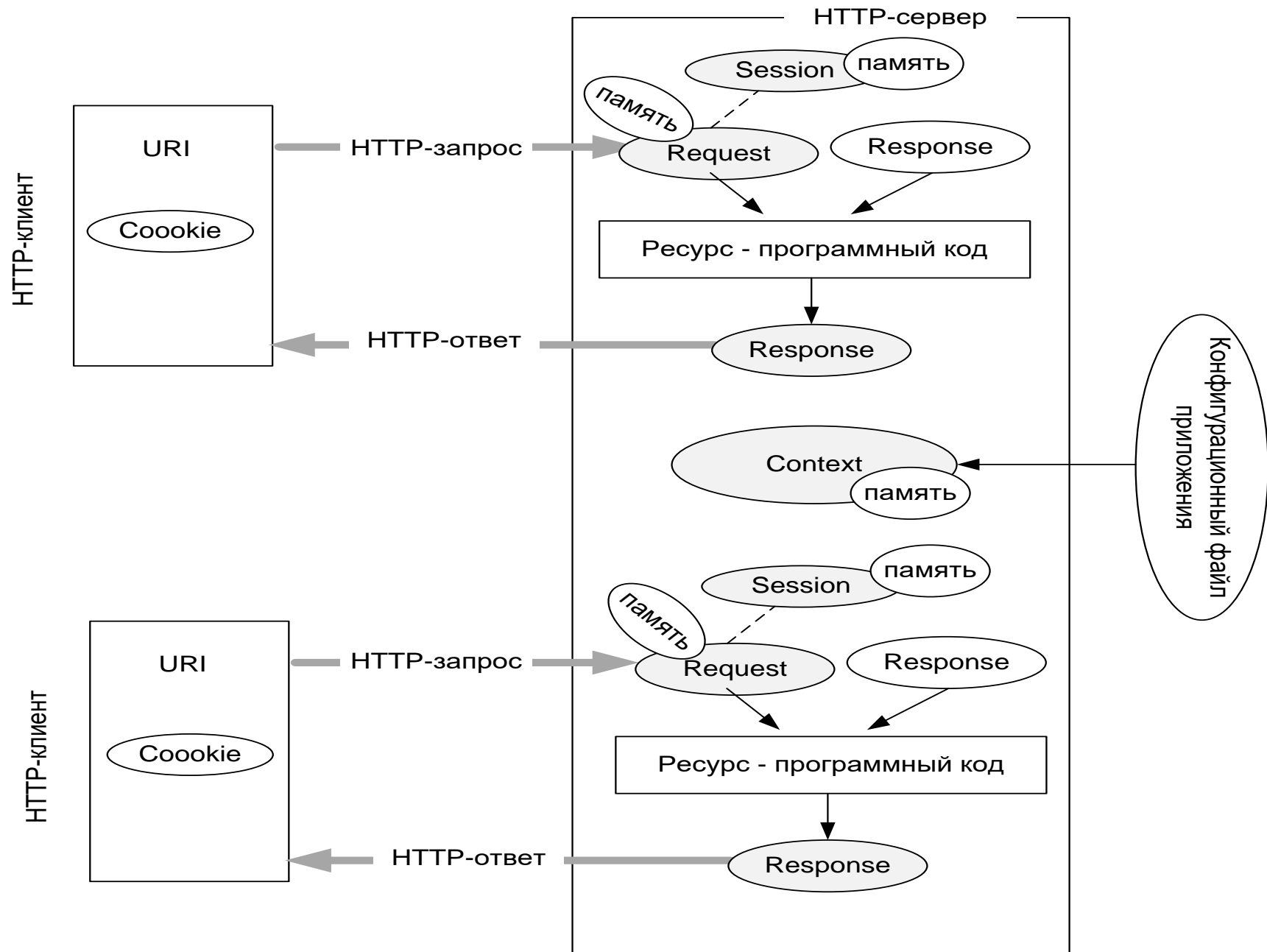
Контекст web-
приложения

=

серверный объект,
предназначенный для хранения
информации об одном web-
приложении.

Как правило, формируется сразу
при загрузке web-сервера,
основные данные (параметры
приложения) копируются из
конфигурационного файла
приложения, общий для всех
сессий приложения.

Обычно контекст предоставляет
возможность хранить данные в
формате ключ/значение.



Фильтр (Filter)

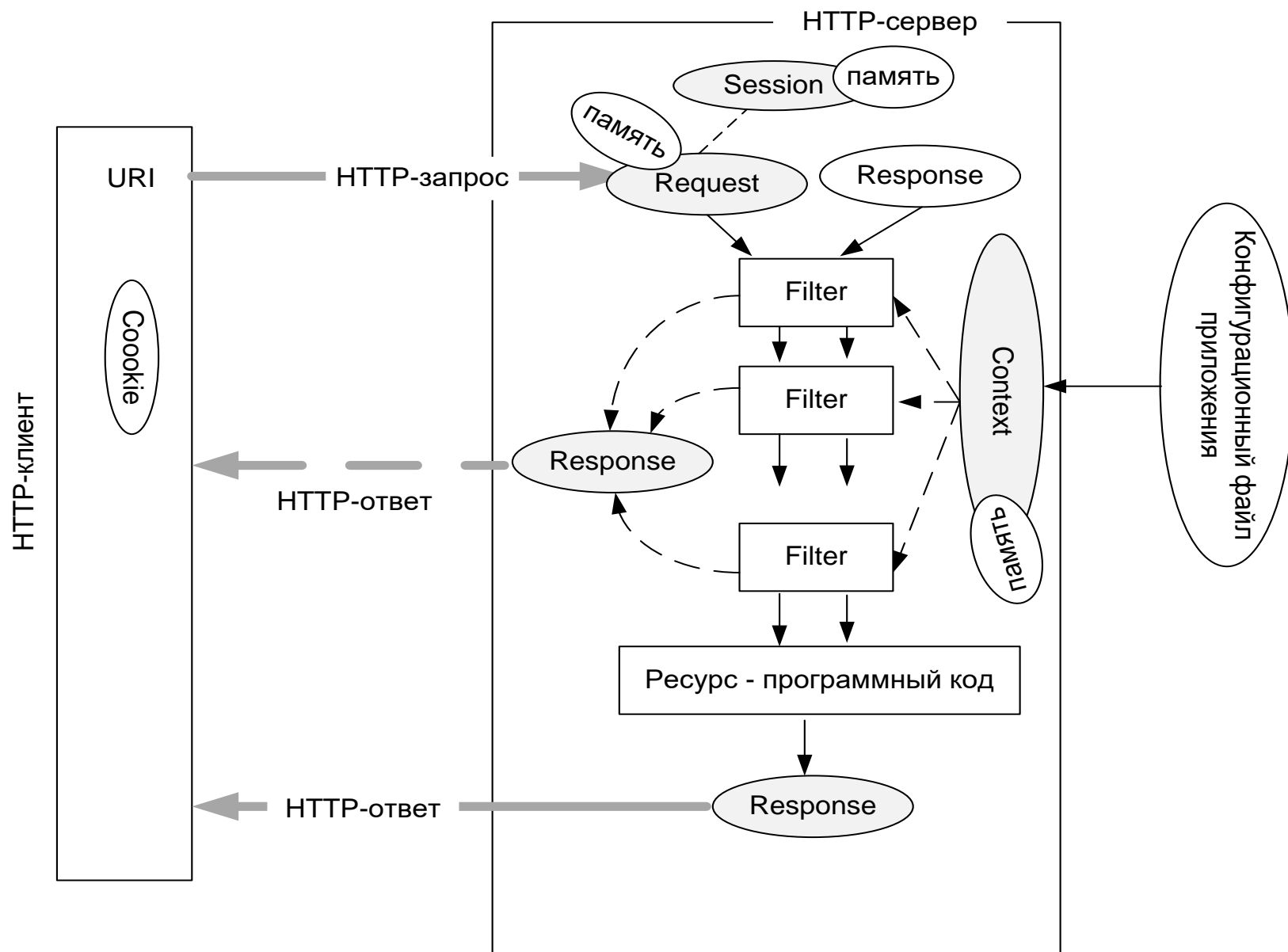
=

серверный объект – препроцессор запроса, предназначен **для предварительной обработки** объекта **Request**.

К одному ресурсу может быть построена цепочка фильтров, последний в цепочке – ресурс. Фильтр может прервать цепочку и сам сформировать ответ клиенту.

Один и тот же фильтр может быть применен к нескольким ресурсам.

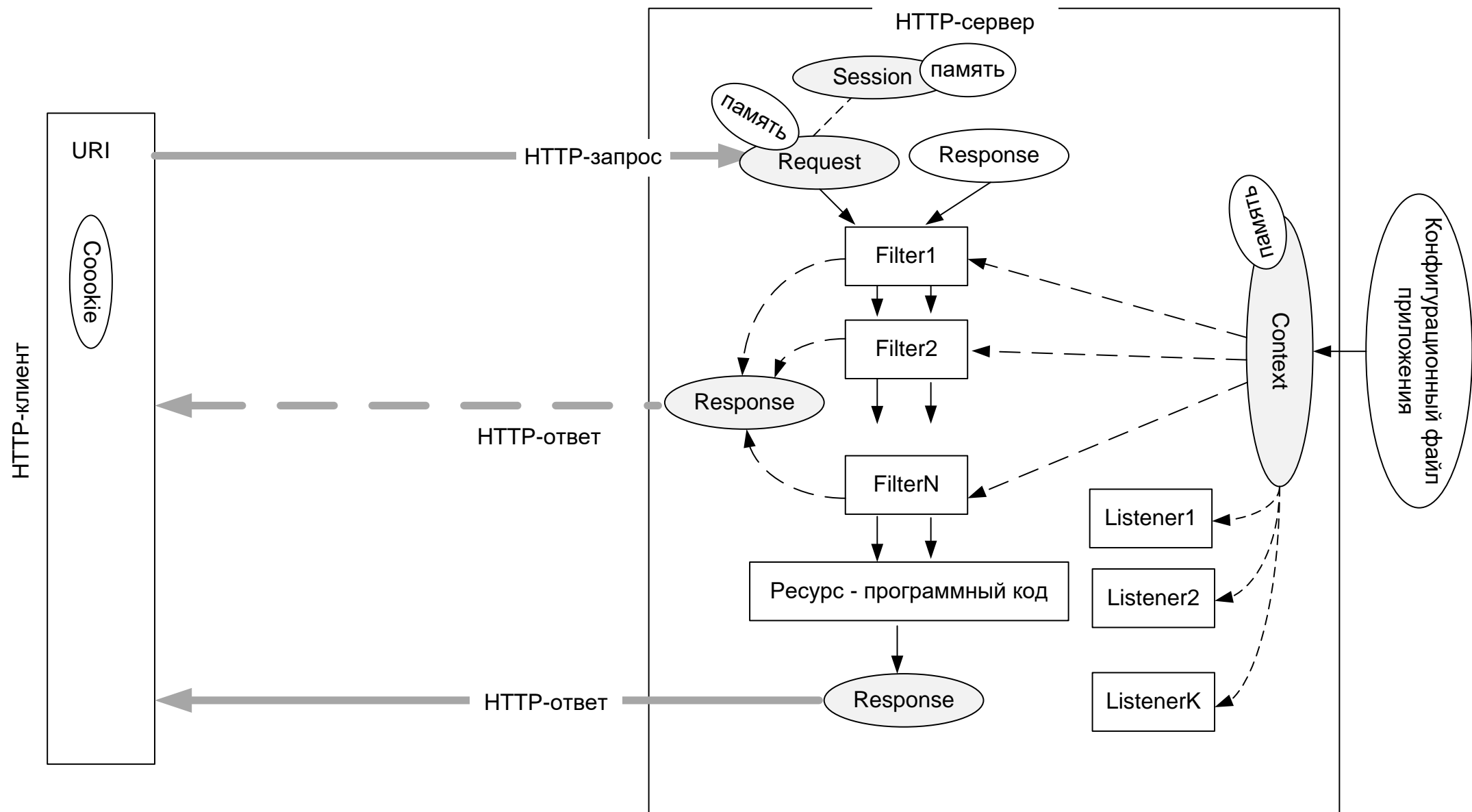
В качестве параметров фильтр получает объекты Request и Response, которые он **передает дальше по цепочке или обрывает цепочку и заполняет** объект **Response**.



Слушатели
событий
(Listener)

=

серверные объекты – для
обработки **событий жизненного
цикла** web-приложения.



Кэш (Cache)

=

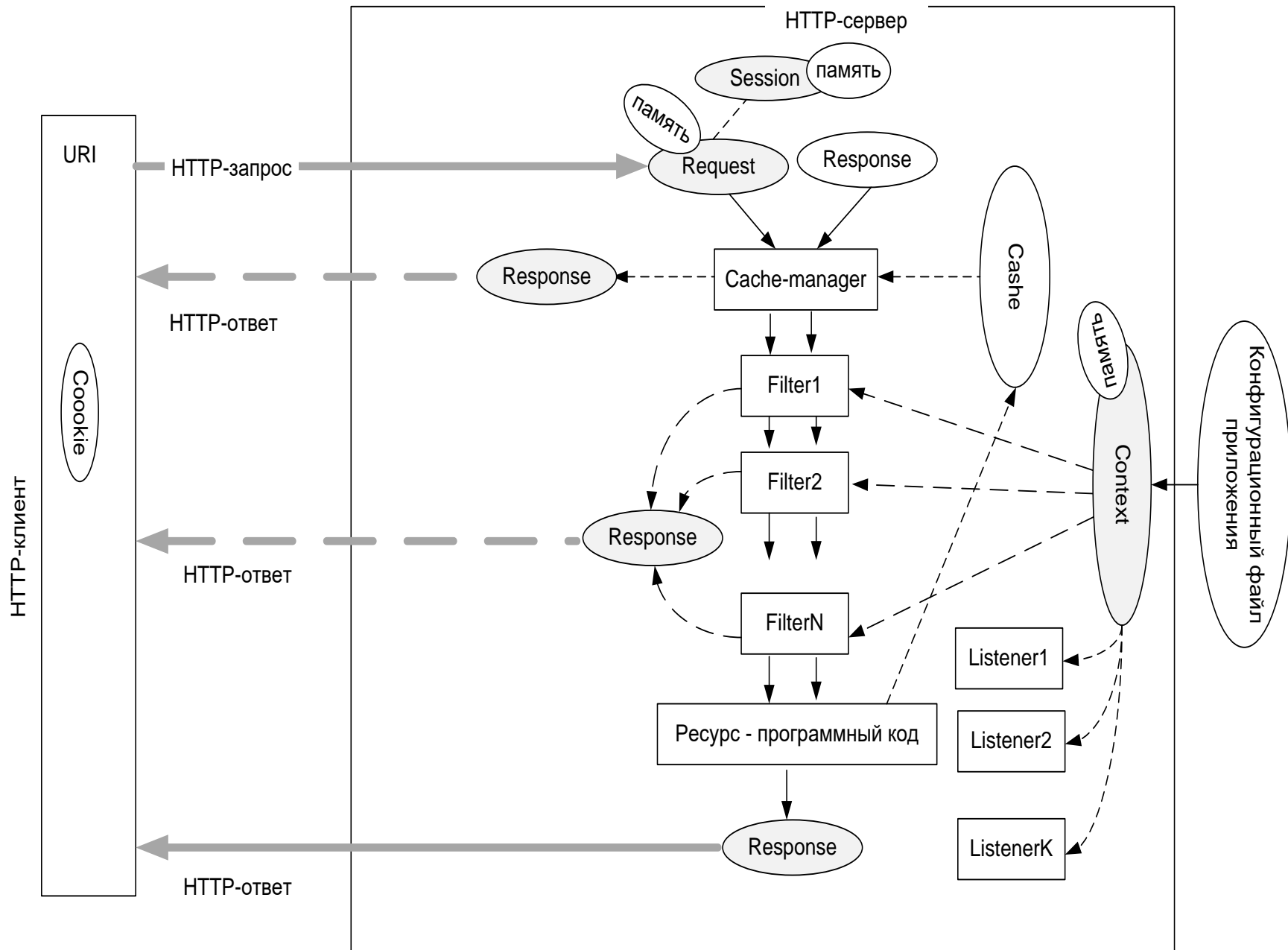
серверный объект, предназначенный для временного **хранения данных** с целью **ускорения выполнения запроса**.

Кэширование – процессы записи и извлечения данных в/из Cache.

Различают кэширование данных и кэширование вывода.

Кэширование данных – кэширование часто используемых данных.

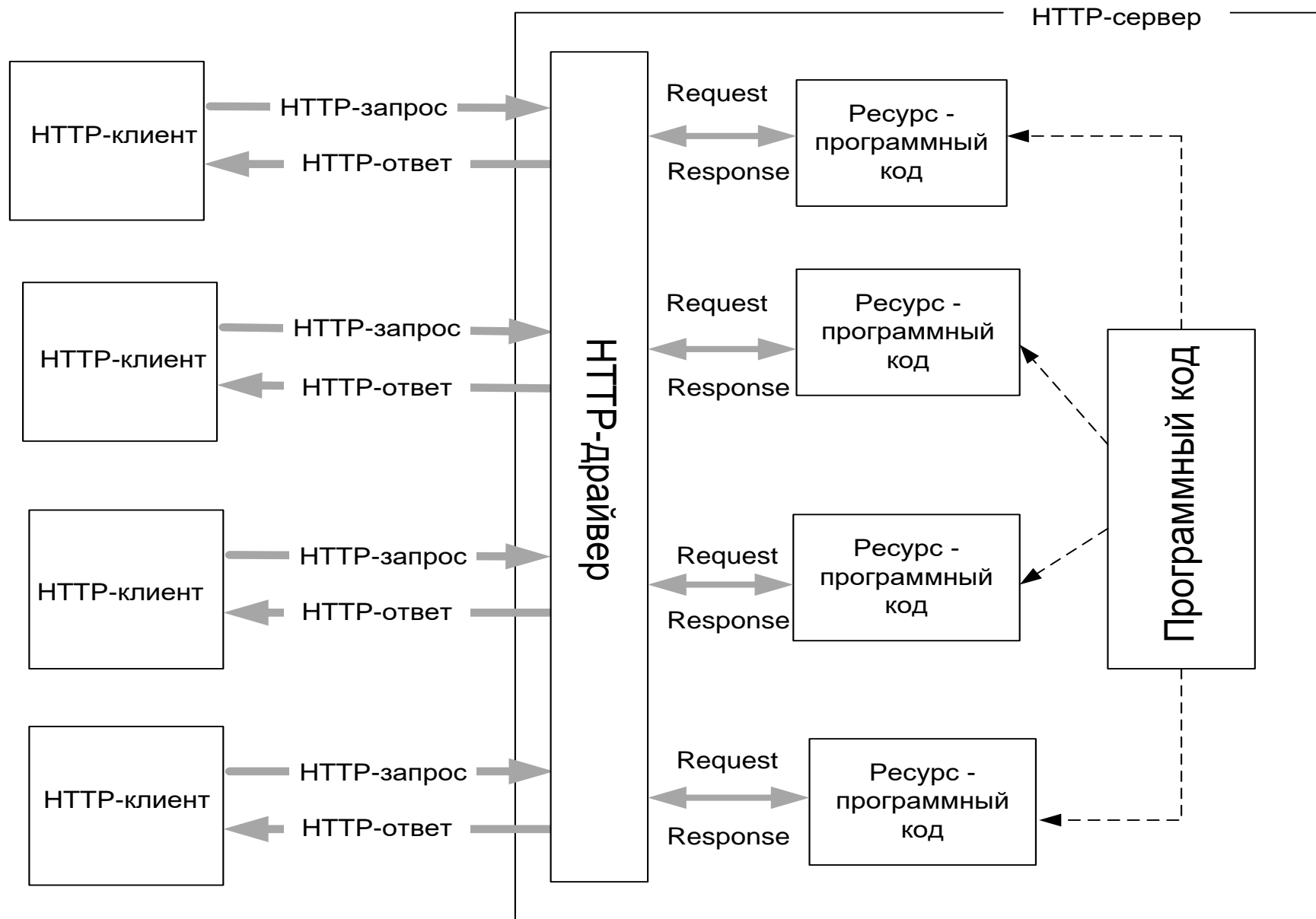
Кэширование вывода – кэширование объекта Response.



Ресурс –
программный =
код

экземпляр приложения, который создается для обработки каждого запроса.

HTTP-драйвер преобразует последовательность битов HTTP-запроса в объект Request.



Постоянное соединение =

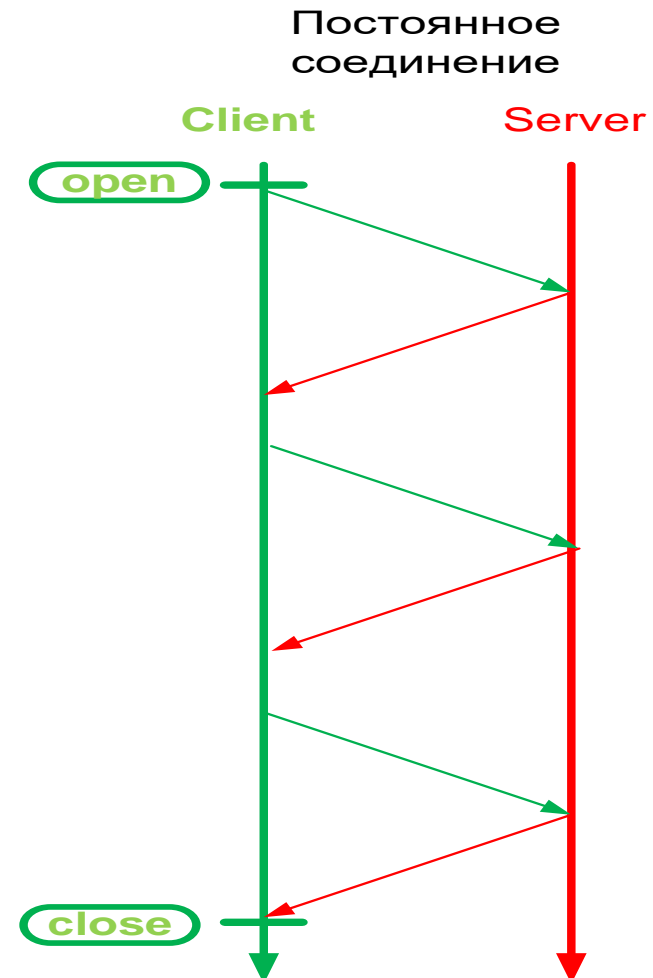
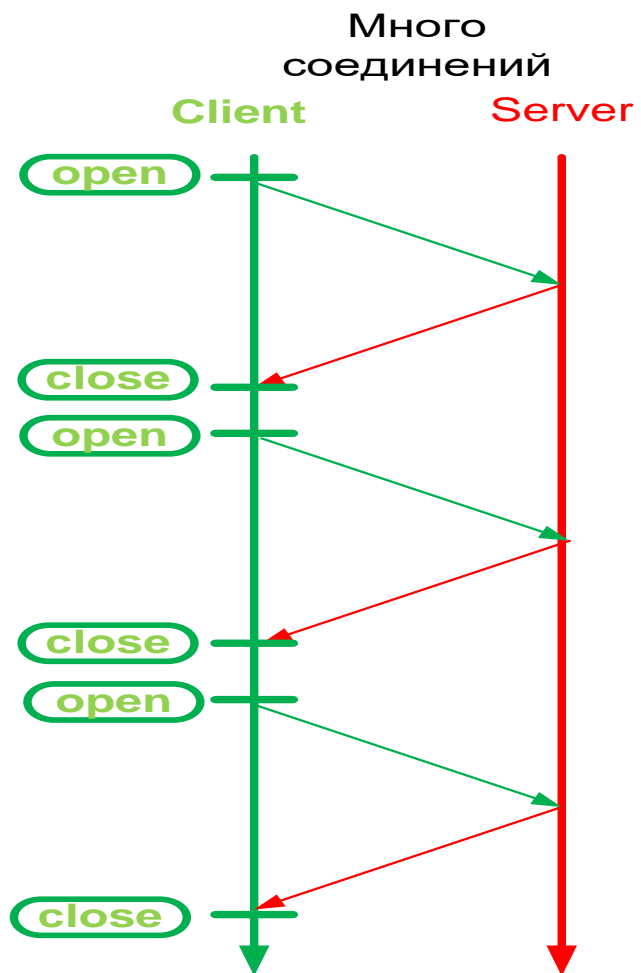
использование **одного ТСР-соединения** для многократных **пар запрос-ответ** вместо последовательного открытия новых соединений для каждой пары запрос-ответ.

Клиент может запросить постоянное соединение с помощью **общего заголовка Connection: Keep-Alive**, сервер подтверждает заголовком Connection: Keep-Alive.

Некоторые серверы открывают постоянное соединение по умолчанию и закрывают через заданное время (timeout) бездействия браузера. Некоторые браузеры по умолчанию открывают постоянные соединения.

Серверы, как правило, поддерживают ограниченное количество постоянных соединений. Обычно по умолчанию используется в HTTPS.

Постоянное соединение



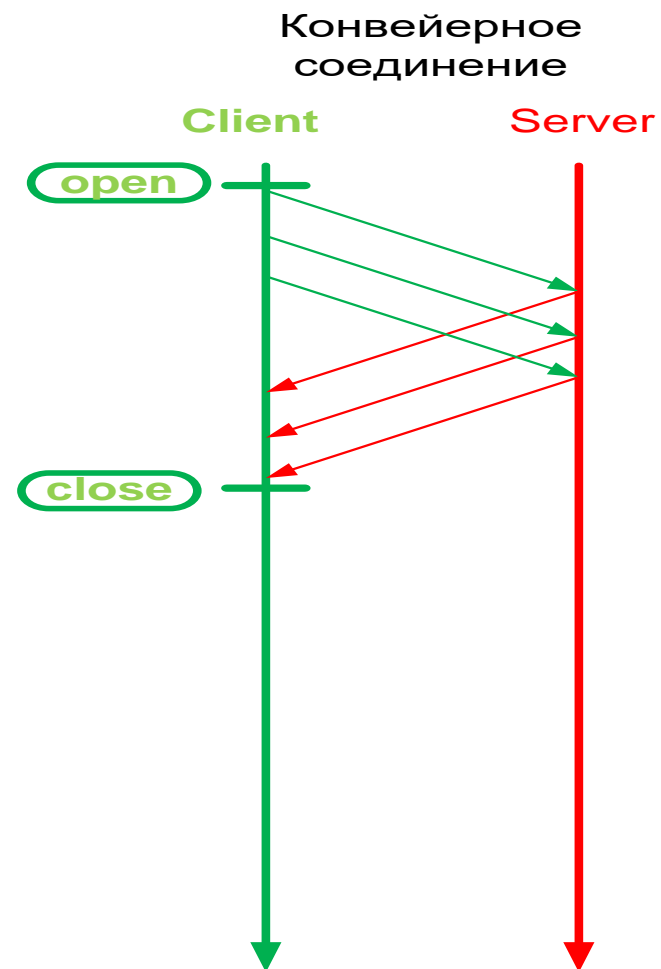
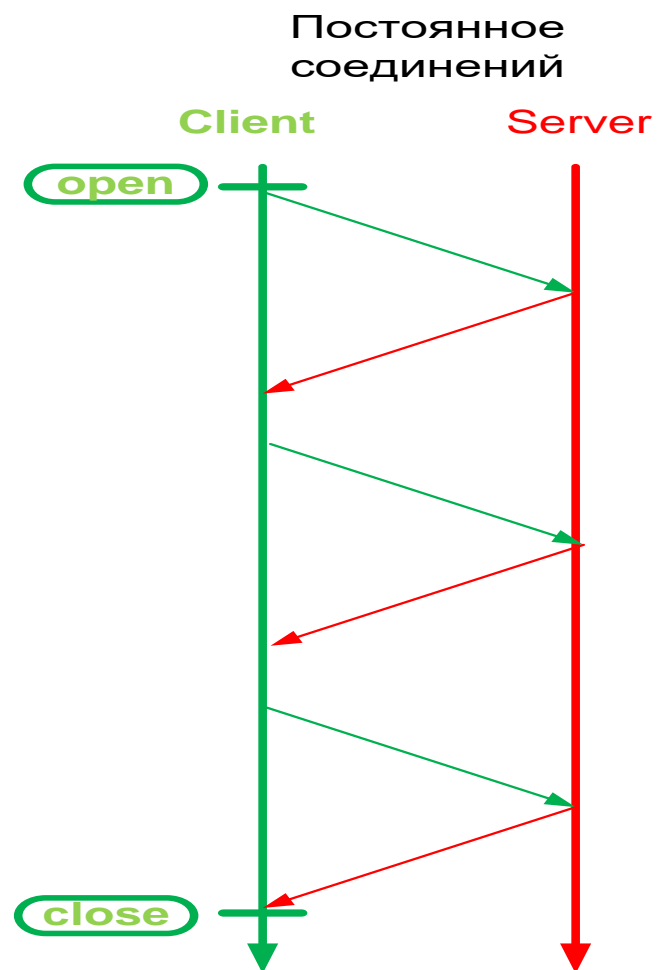
Конвейерная обработка = (HTTP pipelining)

технология, которая позволяет **в одном соединении передавать несколько различных пар запрос–ответ**.

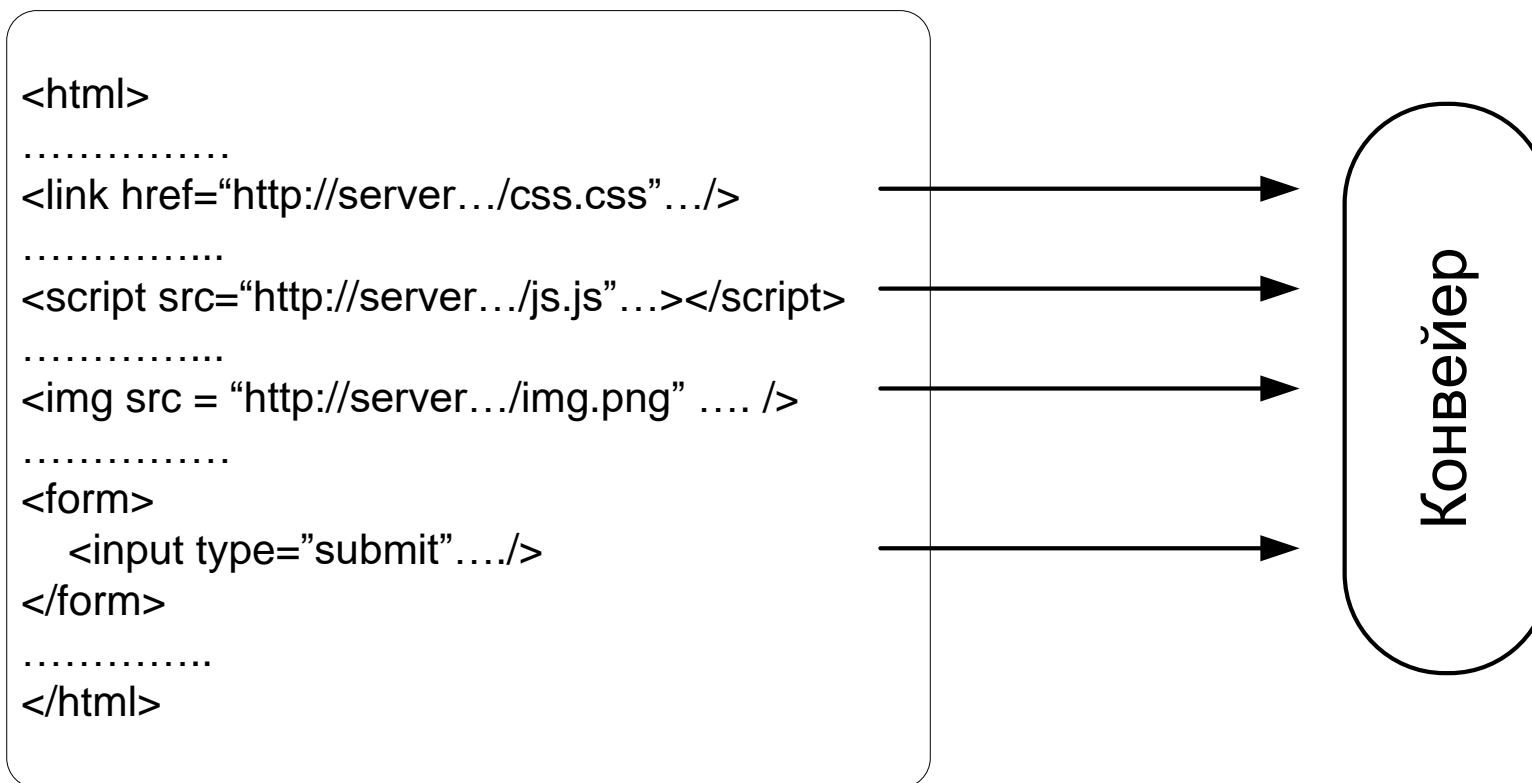
Эффективно, если на одной html-странице запрашивается несколько ресурсов с одного сервера.

В основном поддерживается браузерами мобильных систем.

Конвейерная обработка



Конвейерная обработка



Пул соединений с базой данных =

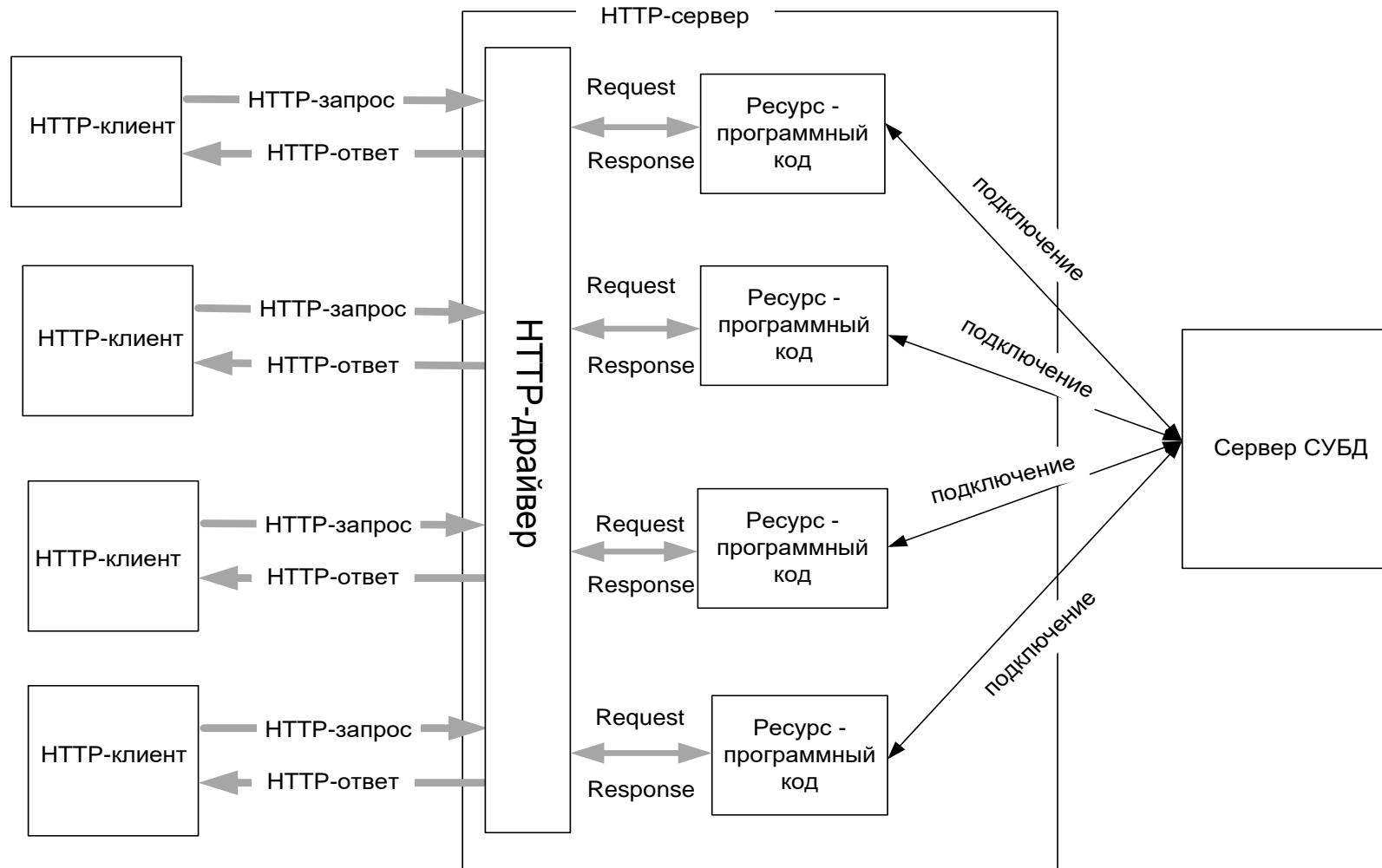
несколько предварительно и **постоянно открытых соединений** с сервером СУБД, которые используют приложения.

Если все подключения пула заняты, запрос на соединение ставится в очередь.

Применение пула позволяет увеличить производительность за счет **отсутствия процесса подключения к серверу**.

Альтернатива: держать постоянное открытое соединение для каждого подключения (в общем случае для web-приложения неприемлемо) или открывать/закрывать соединение при каждом запросе (большие накладные расходы на установку соединения).

Схема работы с базой данных без пула



Пул соединения с базой данных

