Использование cookies

Понятие cookie

- Cookies- это небольшие блоки текстовой информации, которые Web-сервер посылает браузеру, а браузер возвращает неизмененными, когда позже посещает тот же самый Web-сайт или домен.
- Cookies представляют достаточно простой механизм обмена информацией между сервером и браузером. Во время соединения сервер просто создает или открывает уже существующий файл в системе удаленного пользователя, в котором сохраняет некую информацию, которая потом будет применяться для идентификации пользователя и для оптимизации работы с ним.

Стандартная HTTP-транзакция

- Сначала браузер обращается по URL и входит в контакт с сервером. После этого сервер пересылает браузеру некоторую затребованную информацию, или сообщение об ошибке. Затем соединение разрывается, и сервер более ничего не помнит о произошедшей транзакции.
- Таким образом, последующие сеансы связи никак не скореллированы с результатами предыдущих транзакций. Именно этот перекос в некоторой степени выправляют cookies.

Стандарт cookie

- Согласно стандарту, cookies представляют собой обычную строку, не превосходящую по размеру 4000 символов, которая отсылается сервером браузеру.
- Браузер анализирует полученный cookie, проверяет длину, дату истечения срока годности, после чего сохраняет в отдельном файле.
- Cookies не могут содержать исполняемый или интерпретируемый код, а также его фрагменты.

Структура cookies

- Cookie содержит обязательные поля, опциональные поля, а также любую другую информацию в текстовой формате, обработку которой берет на себя сервер.
- Стандартный вид заголовка cookie выглядит следующим образом:

Set-Cookie: name=<значение>; expires=<дата>; path=<путь>; domain=<имя_домена>; secure.

Поля заголовка cookie

- name=<значение> -определение имени и содержания cookie;
- expires=<дата> - это срок годности cookie;
- path=<путь> cookie будет выдан только при затребовании документов, лежащих в указанном каталоге или в его подкаталогах. При помощи этого параметра можно создавать отдельные cookies для каждой Web-страницы, входящей в состав сайта.

Поля заголовка cookie

- domain=<имя_домена> - в этом параметре определяется имя домена, куда будут возвращаться cookies. По умолчанию это имя домена есть имя сервера, приславшего cookie.
- Secure – если в cookie есть это поле, то он будет возвращаться только на сервер, обеспечивающий сертифицированный метод безопасности, обычно SSL (Secure Socket Level).

Свойства cookie

- Domain –свойство содержит доменное имя сайта, с которого устанавливается данный cookie.
- Expris – свойство содержит срок действия cookie.
- HasKeys – если cookie помимо стандартных полей содержит дополнительную информацию, свойство имеет значение true.
- Name – наименование cookie.
- Path – содержит путь к некоему виртуальному каталогу.
- Secure – свойство имеет значение true, если cookie отправляется на систему удаленного пользователя или получается из нее только по защищенному протоколу, такому как HTTPS.
- Value – содержимое cookie.
- Values коллекция значений всех параметров искомого cookie.

Методы коллекции HttpCookieCollection.

- Add –метод добавляет в коллекцию еще один cookie.
- Clear - метод очищает содержимое всей коллекции.
- CopyTo – метод позволяет копировать в некий массив содержимое всех cookies, входящих в коллекцию.
- GetKey – метод возвращает наименование cookie по его порядковому номеру в коллекции.
- Remove – метод позволяет удалить cookie из коллекции по его наименованию.
- Set – метод позволяет изменить содержимое cookie.

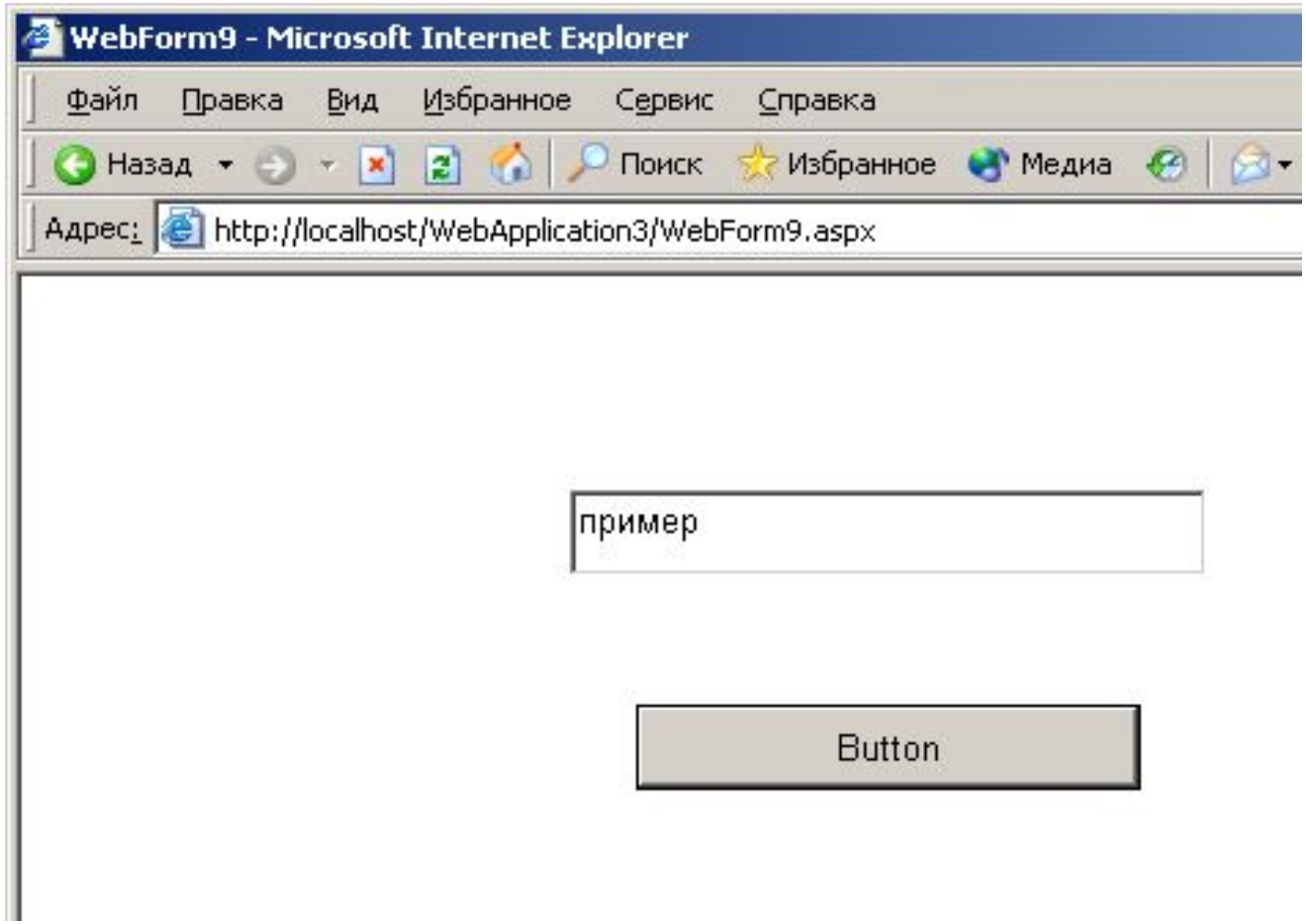
Пример использования механизмов cookies

- Начнем с установки cookies в систему удаленного пользователя. При входе на сайт предложим ему указать свое имя. Для того, чтобы в дальнейшем сайт мог «узнать» и поприветствовать этого пользователя при его новых заходах, можно сохранить в cookies имя, указанное пользователем, и дату его последнего визита на сайт. Для того, чтобы записать информацию в локальную систему пользователя, воспользуемся объектом HttpResponse и его свойством Cookies.

Пример использования механизмов cookies

- В составе объекта HttpResponse нет метода, который явно записывал бы cookie в систему удаленного пользователя. Достаточно лишь создать элементы коллекции Cookies, а они будут переданы браузеру пользователя без каких- либо дополнительных директив, так как объект HttpResponse как раз и создан для передачи информации удаленному пользователю. Стоит также отметить, что все cookies будут переданы браузеру при помощи заголовков HTTP с наименованием Set-Cookie.

Web-приложение для создания cookie



Создание cookies

```
// значение cookie
this.Response.Cookies["c1"].Value=
                                    this.TextBox1.Text;
// создание нового поля для записи даты создания
    cookie
this.Response.Cookies["c1"].Values["time"]=
                                DateTime.Now.ToString();
// указание срока годности cookie
this.Response.Cookies["c1"].Expires=
                                 Convert.ToDateTime("11:52:55");
// переход на другую страницу
this.Response.Redirect("Webform10.aspx");
```

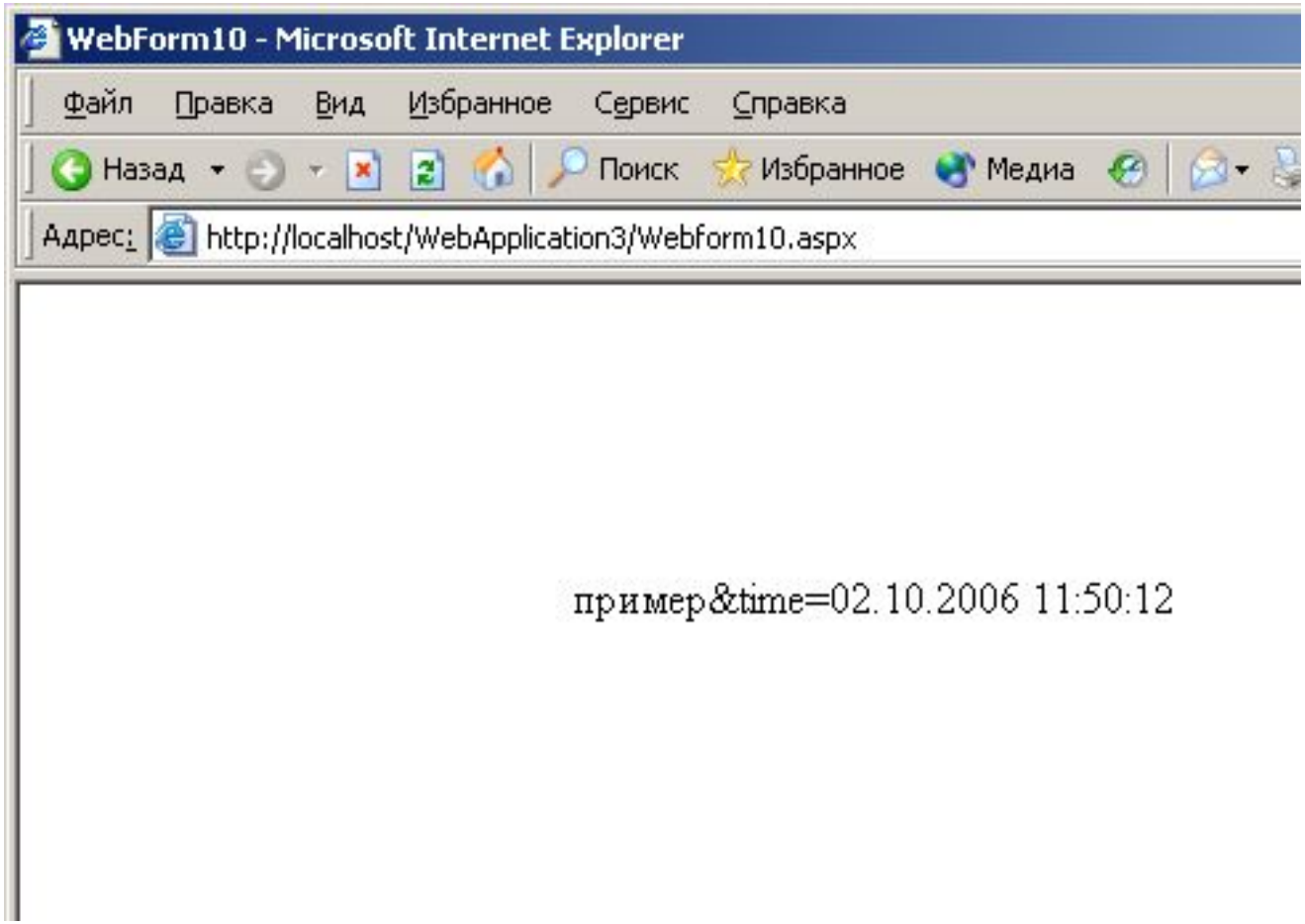
Передача cookie

- Переадресуем пользователя на другую Web-страницу.

```
this.Response.Redirect("Webform10.aspx");
```

- Именно в этот момент, вместе с содержимым этой Web-страницы браузер пользователя получает инструкцию записать в локальную систему содержимое cookie.

Вторая Web-сторанница



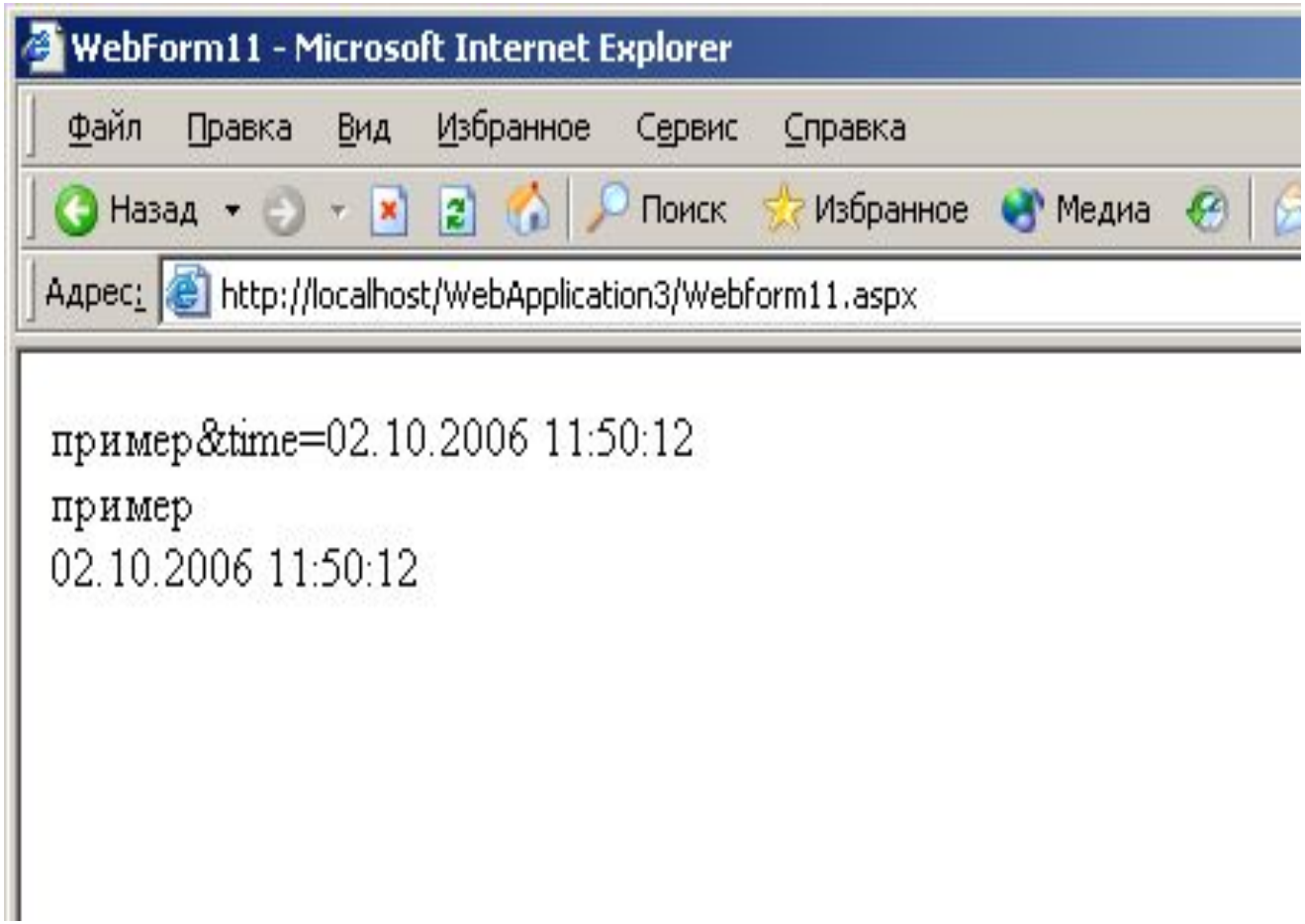
Чтение cookie

```
private void Page_Load(object sender,
    System.EventArgs e)
{
     this.Label1.Text=this.Request.Cookies
                                        ["c1"].Value;
}
```

Третья Web-страница

- Теперь необходимо получить cookies при последующем посещении сайта тем же пользователем и правильно его обработать. В рамках существующего проекта можно создать третью Web-страницу, которая при обращении пользователя к ней будет получать cookie и обрабатывать его.
- Основные действия производятся в обработчике Page_Load, который инициируется при загрузке Web-страницы. По умолчанию, при загрузке страницы в объект Request заранее помещаются все cookies, связанные с данным сайтом.

Третья Web-страница



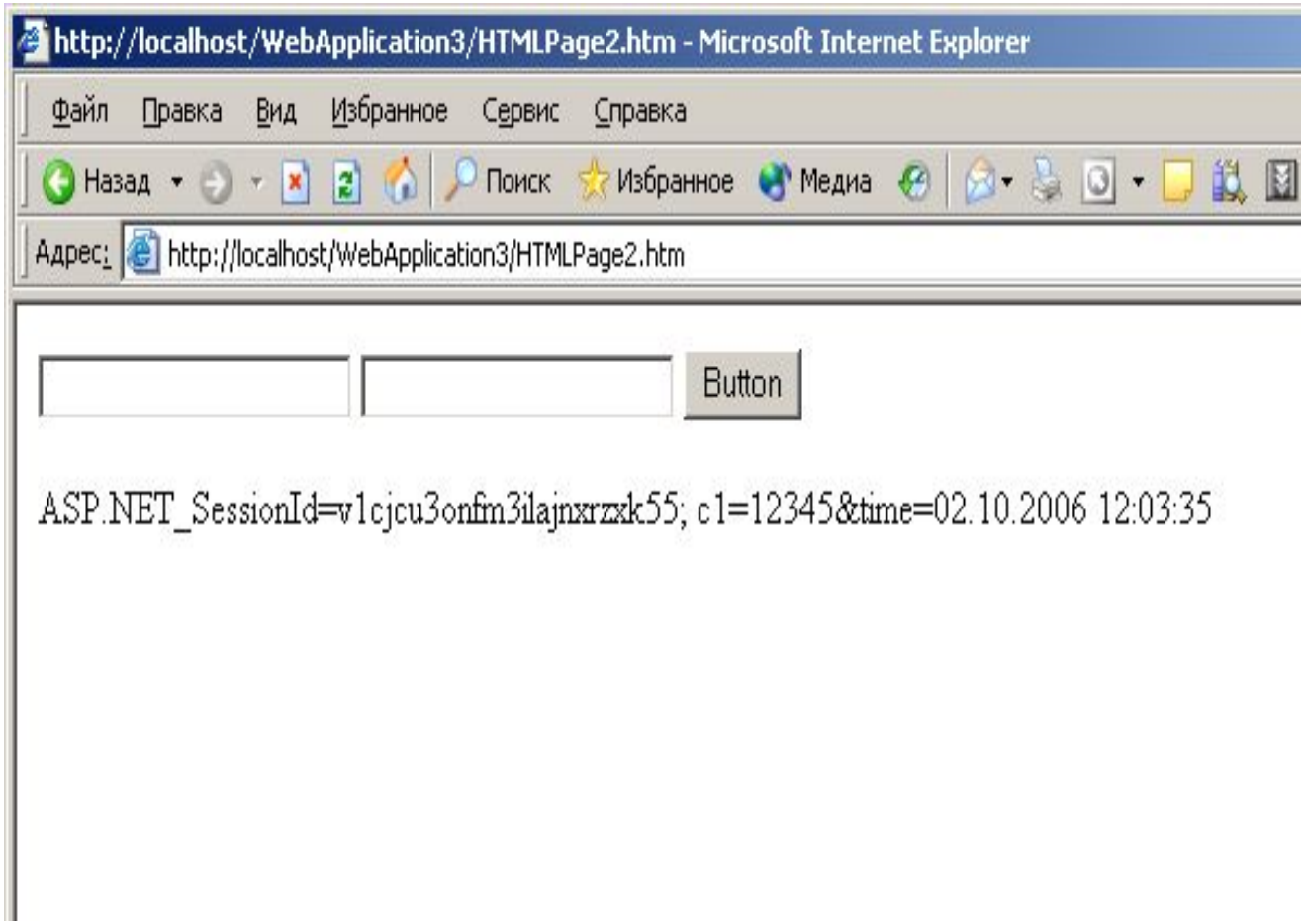
Третья Web-страница

```
try
{
    this.Response.Write(this.Request.Cookies["c1"].
                                Value+"<br>");
    this.Response.Write(this.Request.Cookies["c1"].
                                        Values[0]+"<br>");
    this.Response.Write(this.Request.Cookies["c1"].
                                       Values[1]+"<br>");
}
catch(Exception ex)
{
    this.Response.Write("Cookie перестал
   существовать");
}
```

Страница HTMLPage2.htm

```
<body>
    <form name=f1 method=get action=WebForm7.aspx>
        <INPUT id="Text1" type="text" name="Text1">
        <INPUT id="Text2" type="text" name="Text2">
        <INPUT id="Button1" type="submit"
  value="Button" name="Button1">
        </form>
        <script>
        document.write(document.cookie);
        </script>
    </body>
</html>
```

Страница HTMLPage2.htm



Преимущества использования cookies

- Идентификация пользователя во время сеанса.
- Отсутствие имени и пароля.
- Настройка сайта.
- Направленная реклама.

Идентификация пользователя во время сеанса

- Обычно HTTP-соединение закрывается после передачи каждой страницы. Сохраняемые (keep-alive) HTTP-соединения не решают эту проблему, потому что обычно они используются только для запросов, которые близки по времени, например, когда браузер запрашивает изображения, связанные с Web-страницей. Кроме того, во многих серверах и браузерах отсутствует поддержка сохраняемых соединений. Однако cookies могут решить эту проблему.

Отсутствие имени и пароля

- Многие большие сайты требуют, чтобы пользователь зарегистрировался, прежде чем воспользовался услугами сайта. Но помнить и вводить имя пользователя и пароль каждый раз, когда посещается такой сайт, неудобно. Хорошей альтернативой этому для сайтов с низким уровнем безопасности являются cookies.
- Когда пользователь регистрируются, ему посылается cookie, содержащий уникальный пользовательский идентификатор ID. Сервер просматривает его, определяет, что он принадлежит зарегистрированному пользователю, и разрешает доступ без явного задания имени пользователя и пароля. Сайт также может помнить адрес пользователя и т.п., что упрощает будущие транзакции.

Настройка сайта

- Многие сайты позволяют настраивать внешний вид главной страницы. Настраивать страницу при каждом посещении сайта неудобно, поэтому, чтобы запомнить настройку, используются cookies. Однако для более сложной настройки сайт посылает клиенту уникальный идентификатор и сохраняет базу данных на стороне сервера, которая связывает идентификатор с параметрами страницы.

Направленная реклама

- Cookies позволяют помнить, что пользователь искал раньше, и выводить соответствующее объявление вместо случайного.

Проблемы работы с cookies

- Cookies не представляют серьезной угрозы для безопасности сайтов. Cookies никогда не интерпретируются и не выполняются и, следовательно, не могут использоваться для внесения вирусов или организации атаки на систему пользователя.
- Более того, поскольку браузеры обычно принимают только 20 cookies на сайт и 300 cookies всего и поскольку каждый cookie может быть ограничен 4 килобайтами, они не могут использоваться для заполнения диска или инициализации другого способа атаки на компьютер пользователя.

Проблемы работы с cookies

- Однако, хотя cookies не представляют серьезной угрозы безопасности, они могут представлять значительную угрозу конфиденциальности. Вследствие наличия реальных и ощутимых проблем защиты конфиденциальности многие пользователи отключают cookies.