

COOKIE и SESSION в PHP.

Важное

- Введение
- Настройки времени выполнения
- Особенности механизма сессий
- Практика применения директив
- Междоменные сессии
- Время жизни сессии
- Заключение

Введение

В предыдущ мы изучили поведение массивов `$_COOKIE` и `$_SESSION` по умолчанию (в тепличных условиях). В реальных приложениях приходится кодить с учетом возможного поведения пользователя.

Протестируем выполнение серверного сценария при следующих начальных условиях:

- Поддержка cookie **разрешена** пользователем.
- Поддержка cookie **запрещена** пользователем.

Поддержка cookie разрешена пользователем

По умолчанию **cookie** разрешены. Выполните и проанализируйте код демонстрационного примера **example_1**.

example_1. index.php.

```
<?php

// выполнить пример с:
```

```
// - включенными $_COOKIE

// - отключенными $_COOKIE

// сессия создает cookie PHPSESSID

session_start();

// сохраняем в cookie логин

setcookie('login', 'admin');

// сохраняем в сессию сообщение

$_SESSION['msg'] = 'It\'s Work?!';

echo '<h3>Массив $_COOKIE</h3>';

echo '<pre>';

print_r($_COOKIE);

echo '</pre>';

echo '<h3>Массив $_SESSION</h3>';

echo '<pre>';

print_r($_SESSION);

echo '</pre>';

// echo '<h3>Массив $GLOBALS</h3>';

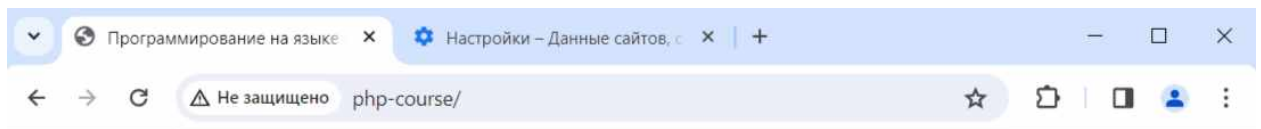
// echo '<pre>';

// print_r($GLOBALS);

// echo '</pre>';

?>
```

Скриншот вывода сценария демонстрационного примера при включенных файлах **cookie** приведен на следующем рисунке.



Массив `$_COOKIE`

```
Array
(
    [PHPSESSID] => 9ip6fuucqumbu272tfoth76k6446dgjd
    [login] => admin
)
```

Массив `$_SESSION`

```
Array
(
    [msg] => It's Work?!
)
```

Безопасность приложений

При анализе результата выполнения кода сценария можно утверждать следующее:

- Массив `$_COOKIE` – содержит значение для ключа **login**, добавленное функцией **setcookie()** и значение для **PHPSESSID**, добавленное функцией **session_start()**.
- Массив `$_SESSION` – содержит некоторое строковое значение. Сессия стартует, данные в массив `$_SESSION` добавляются.

Поддержка cookie запрещена пользователем

Теперь выполним демонстрационный пример **example_1**, но при отключенных пользователем файлах **cookie**. Алгоритм действий для отключения **cookie** зависит от браузера, в котором вы работаете.

Приведу цепочку действий для отключения **cookie** браузера **Google Chrome**:

1. **Настройки.**
2. **Конфиденциальность и безопасность.**
3. **Настройки сайтов.**

4. Данные сайтов, сохраненные на устройстве.

5. Настройки по умолчанию.

- Разрешить сайтам сохранять данные на устройстве.
- Запретить сайтам сохранять данные на устройстве (не рекомендуется).

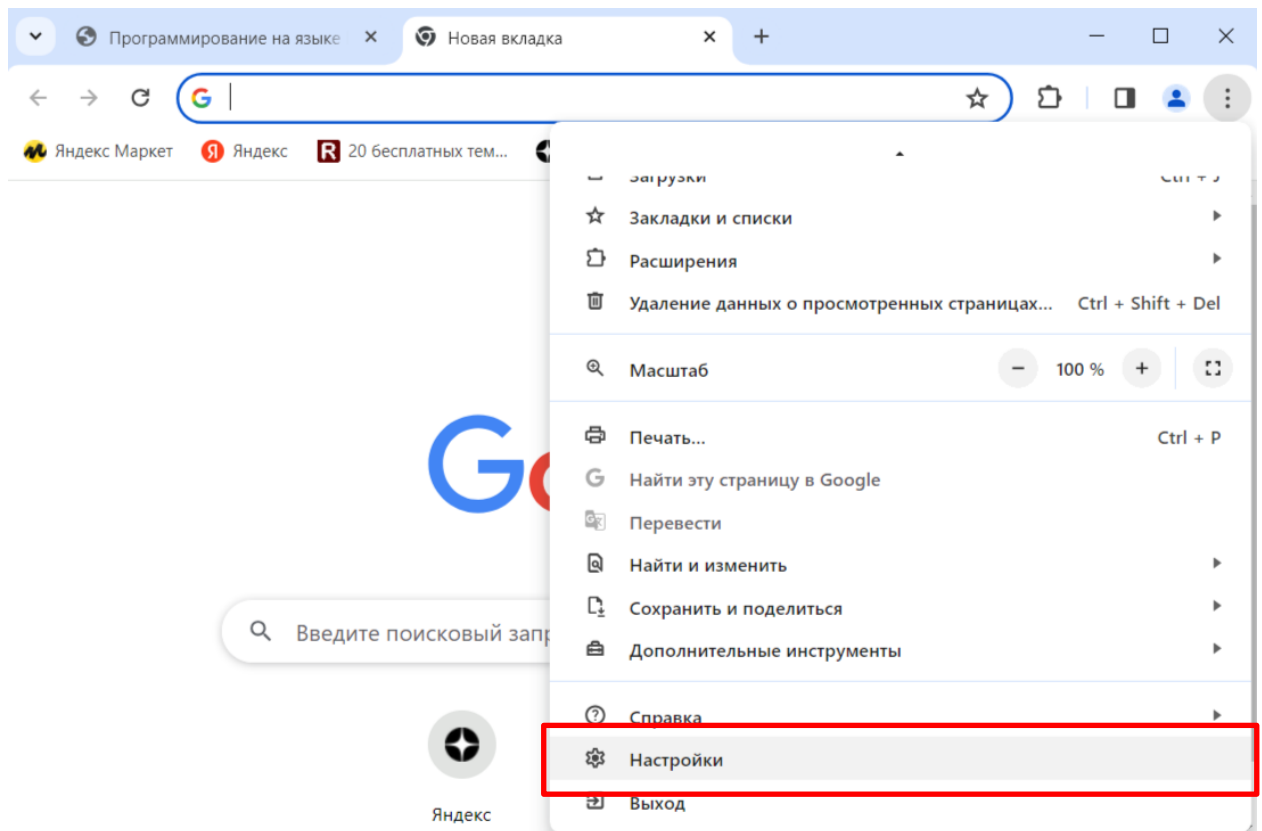


Рис. 1. Настройки

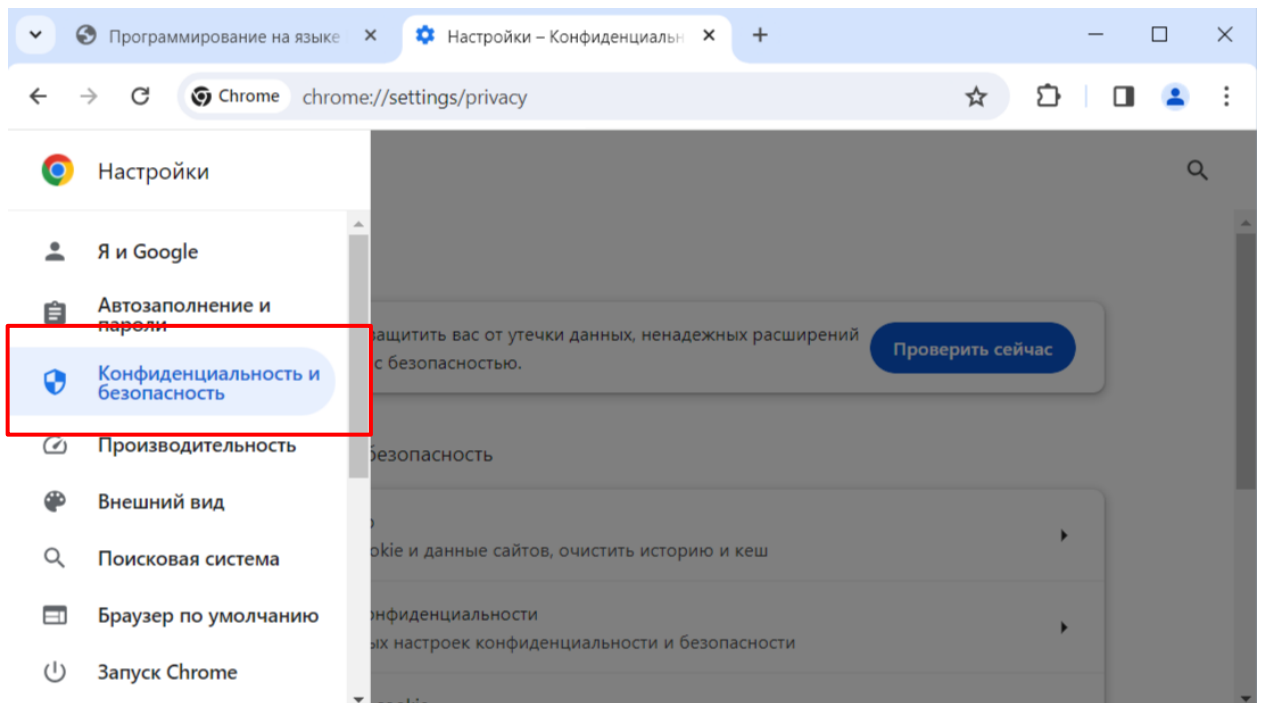


Рис. 2. Конфиденциальность и безопасность

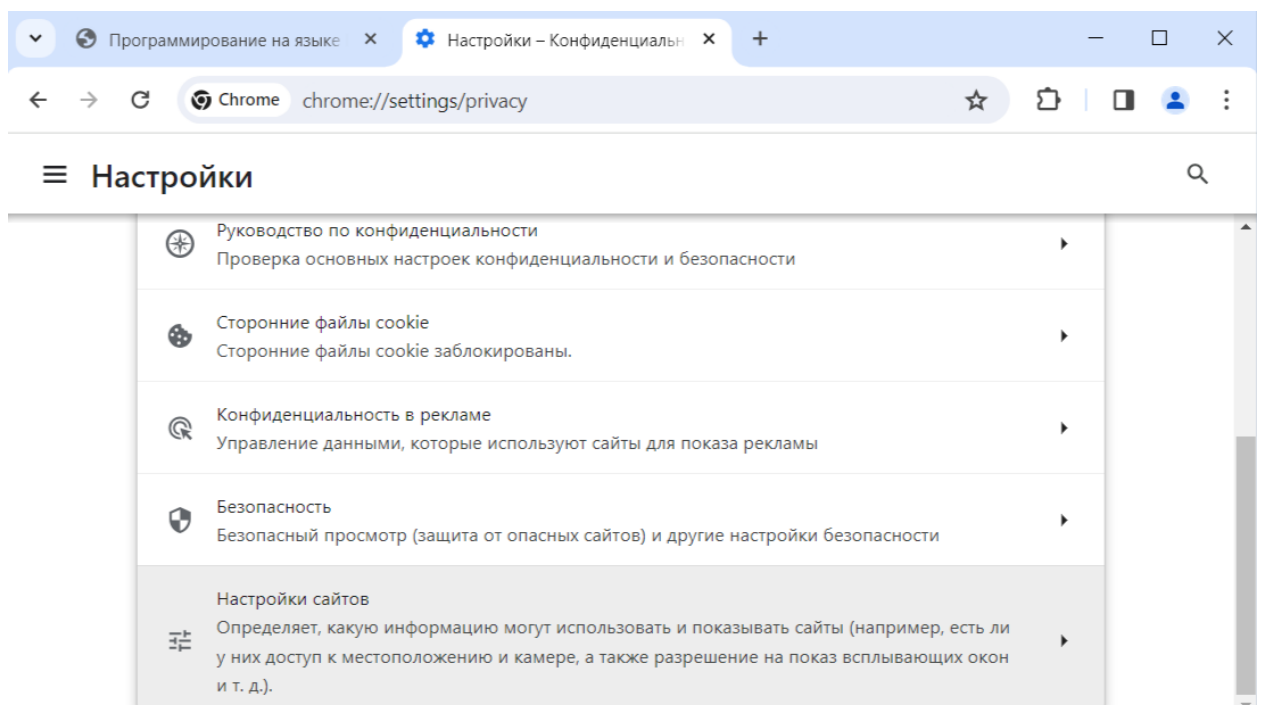


Рис. 3. Настройки сайтов

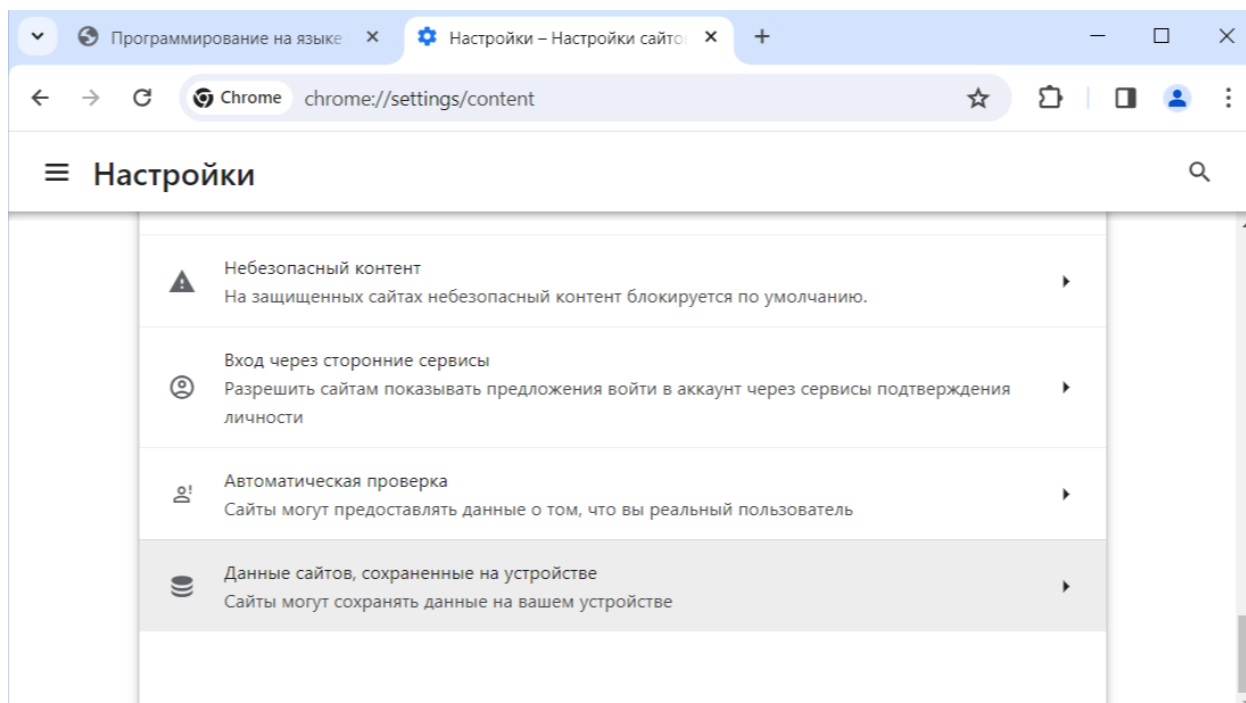


Рис. 4. Данные сайтов, сохраненные на устройстве

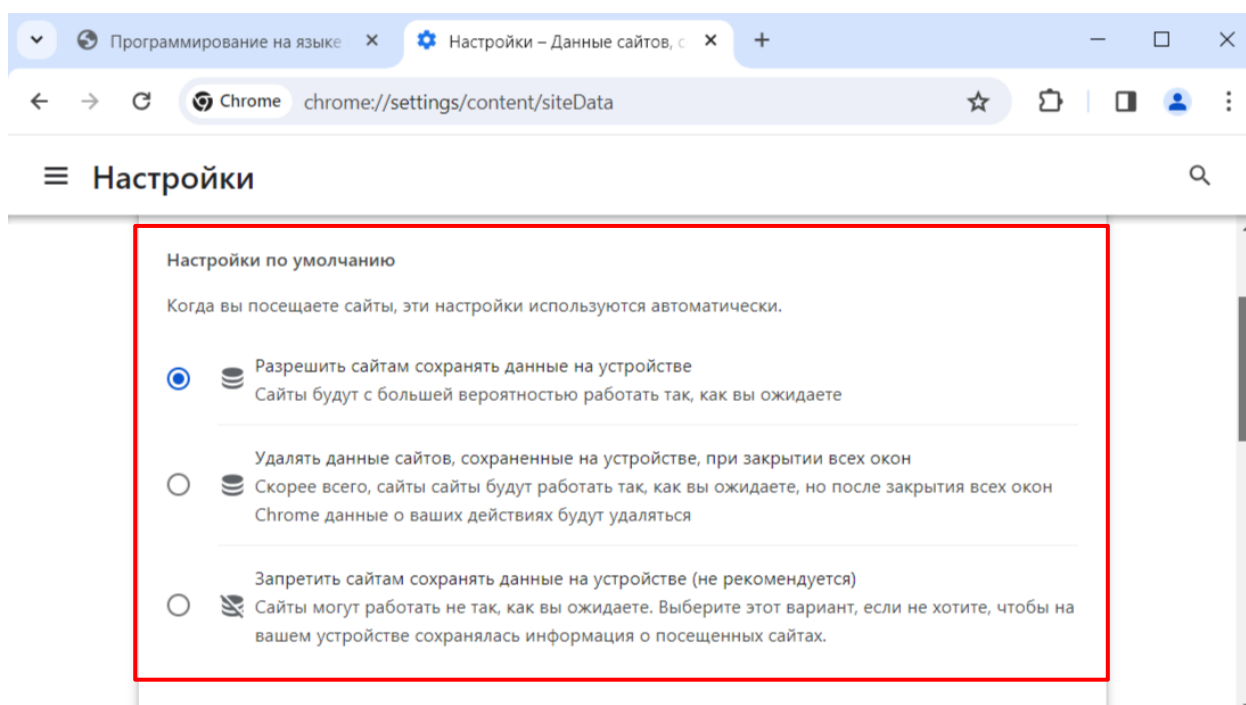
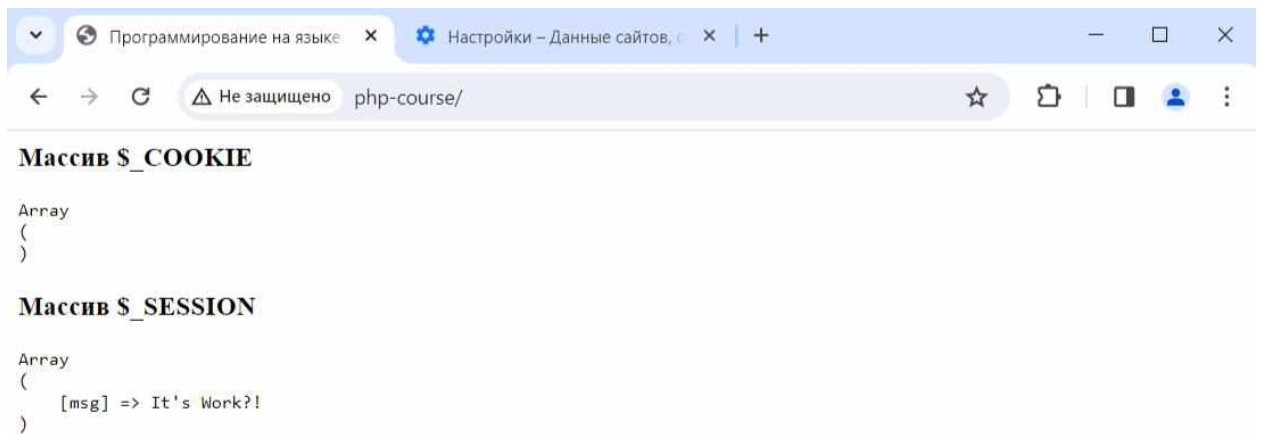


Рис. 5. Настройки по умолчанию

Скриншот вывода сценария при отключенных файлах **cookie** приведен на рисунке.



Безопасность приложений

При анализе результата выполнения кода сценария можно утверждать следующее:

- Массив **\$_COOKIE** – пуст, что логично. Файлы **cookie** запрещены браузером.
- Массив **\$_SESSION** – содержит значение ключа **msg**, что тоже логично. Сессия стартует, данные в массив **\$_SESSION** добавляются.

Таким образом, работа приложения может сильно отличаться от действий пользователя. Задача разработчика – предусмотреть возможный ход развития событий и принять соответствующие меры.

Например, использовать **директивы** времени выполнения функции **ini_set()**.

Настройки времени выполнения

Работа механизма сессий зависит от установок **конфигурационного** файла **php.ini**, **конфигурационного** файла **.htaccess** или от настроек времени выполнения функции **ini_set()**.

Ini_set() — это функция в PHP, которая динамически изменяет настройки **php.ini** (настройки конфигурации) во время выполнения скрипта.

Функция позволяет временно настраивать определённые значения конфигурации для **конкретного скрипта** без необходимости прямого редактирования конфигурационных файлов.

В этом уроке рассмотрим **директивы** функции **ini_set()**:

- session.**use_cookies**
- session.**use_only_cookies**
- session.**use_trans_sid**
- session.**gc_maxlifetime**
- session.**cookie_lifetime**

session.use_cookies

session.use_cookies определяет, **будет ли модуль PHP использовать cookies** для хранения идентификатора сессии на стороне клиента. По умолчанию 1 (включено).

Пример настройки:

```
// использовать cookie  
  
ini_set('session.use_cookies', 1); // да  
  
ini_set('session.use_cookies', 0); // нет
```

session.use_only_cookies

session.use_only_cookies определяет, **будет ли модуль PHP использовать только cookies** для хранения идентификатора сессии на стороне клиента. По умолчанию 1 (включено).

Пример настройки:


```
// использовать только(!) cookie  
  
ini_set("session.use_only_cookies", 1); // да  
ini_set("session.use_only_cookies", 0); // нет
```

session.use_trans_sid

session.use_trans_sid указывает, используется ли прозрачная поддержка **SID** или нет. По умолчанию 0 (отключено). **SID** – предопределённая константа модуля.

Пример настройки:

```
// использовать SID  
  
ini_set("session.use_trans_sid", 1); // да  
ini_set("session.use_trans_sid", 0); // нет
```

SID

Константа, содержащая либо имя сессии и идентификатор в виде "**name=ID**" либо **пустую строку**, если идентификатор сессии был установлен в соответствующие **cookies** сессии. Это тот же идентификатор, что возвращает функция **session_id()**.

Пример константы:

```
SID = "PHPSESSID=u8h2auaua4a8f6vbg6lms00f8ml70nte"
```

Важно. Константа **SID** будет непустой только при:

- отказе от cookie,
- разрешении использовать не только cookie,

т.е. при следующих настройках:

- `ini_set('session.use_cookies', 0);`

- `ini_set("session.use_only_cookies", 0);`

session.gc_maxlifetime

session.gc_maxlifetime задаёт отсрочку времени в секундах, после которой данные будут рассматриваться как "мусор" и потенциально будут удалены. Сбор мусора может произойти в течение старта сессии (в зависимости от значений `session.gc_probability` и `session.gc_divisor`). По умолчанию значение **1440** (24 минуты).

Пример настройки:

```
// увеличим время жизни сессии
ini_set('session.gc_maxlifetime', 86400);
```

session.cookie_lifetime

session.cookie_lifetime указывает время жизни **cookies**, отправляемого в браузер клиента, в секундах. Значение 0 означает, что **cookies** будут валидны до закрытия браузера. По умолчанию равно 0.

Пример настройки:

```
// время жизни сессионного cookie
ini_set('session.cookie_lifetime', 0);
```

Примечание. Директив настроек времени выполнения очень много. Их изучение может занять достаточно много времени, но для первого погружения в тему сессий указанных директив вполне достаточно.

Примечание. Для лучшего запоминания, каждая из настроек будет описана в комментариях к коду демонстрационных примеров.

Особенности механизма сессий

Работа сессий определяется тем, включена ли поддержка **cookie** на стороне клиента.

Поддержка cookie включена

В этом случае глобальной переменной **PHPSESSID** присваивается значение идентификатора сессии, а **cookie** используются для передачи значения **PHPSESSID** между страницами (сценариями) сайта.

Поддержка cookie отключена

Интерпретатор PHP создает константу **SID**, значением которой является значение "PHPSESSID=идентификатор_сессии".

Использование константы определяется значением директивы **session.use_trans_sid**. По умолчанию значение директивы равно 0 (директива отключена). Чтобы включить директиву, необходимо в файле конфигурации **php.ini** изменить соответствующую строку:

session.use_trans_sid = 1. При включенной директиве можно создавать сценарии, не зависящие от настройки браузера по применению **cookie**.

Рассмотрим случаи различных значений директивы.

session.use_trans_sid = 1. Идентификатор сессии при отключенной поддержке **cookie** пересылается посредством элементов, в которых указан URL-адрес. При переходе на другую страницу с помощью

- **ссылки**,
- **формы** с методом **GET**,
- **формы** с методом **POST**

идентификатор сессии добавляется к URL-адресу и виден в адресной строке браузера, что не всегда желательно. Следует иметь в виду, что идентификатор сессии добавляется к URL-адресам, соответствующим одному и тому же сайту. Если в URL-адресе содержится имя сервера, то к такому URL-адресу идентификатор сессии добавлен не будет. Например, при использовании ссылки:

`` – идентификатор будет добавлен к адресу, а при использовании ссылки:

`` – нет.

Кроме того, можно использовать передачу идентификатора сессии в скрытых полях формы с методом **POST**, при котором идентификатор сессии не добавляется к URL-адресу:

```
<input type="hidden" name="PHPSESSID" value="$PHPSESSID">
```

Важно. Идентификатор сессии **не будет** передаваться вместе с заголовком **Location**, даже если включена настройка **session.use_trans_sid**. Его нужно передавать вручную, используя константу **SID**.

session.use_trans_sid = 0. Идентификатор сессии при отключенной поддержке **cookie** не пересылается. При необходимости разработчик приложения может сделать это **самостоятельно**, используя константу **SID**. Поскольку эта константа имеет значение вида:

"PHPSESSID=идентификатор_сессии", то можно просто добавить ее к URL-адресу, например:

```
<a href="page.php?<?php SID ?>">Главная страница</a>
```

Все сказанное закрепим практикой применения директив. Во время тестирования регулярно очищайте браузер от файлов **cookie** предыдущих примеров, отслеживайте передаваемые заголовки с использованием режима разработчика (F12).

Практика применения директив

Следующие демонстрационные примеры могут отличаться всего несколькими символами. Но, тем не менее, понимание сути происходящих процессов настолько важно, что будут приведены полные варианты исходного кода.

Важно. При отсутствии особой отметки демо-примеры выполняются с настройкой браузера – **Разрешить сайтам сохранять данные на устройстве.**

В вашем браузере данная настройка может называться как-нибудь иначе (смысл от этого не меняется).

В демонстрационных примерах example_2 – example_3 обратите внимание на следующие моменты:

- **Идентификатор** сессии. При запрете использовать cookie обновляется каждый раз при старте сессии.
- Константа **SID**. Будет иметь значение только при следующих настройках:
 - **ini_set('session.use_cookies', 0)** (использовать cookie – нет);
 - **ini_set("session.use_only_cookies", 0)** (использовать только cookie – нет);

В демонстрационном примере **example_2 по умолчанию** выполнены следующие настройки текущего скрипта:

```
// использовать cookie  
  
ini_set('session.use_cookies', 0); // нет  
  
// использовать только cookie  
  
ini_set("session.use_only_cookies", 1); // да
```

example_2. index.php

```
<?php

    // настройки времени выполнения

    /*

        использовать cookie:

        1 - да

        0 - нет

    */

    ini_set('session.use_cookies', 0);

    /*

        использовать только cookie:

        1 - да

        0 - нет (можно использовать SID)

    */

    ini_set("session.use_only_cookies", 1);

    // старт сессии

    session_start();

    // вывод информации о сессии

    echo 'Идентификатор сессии: ' . session_id() . '<br>';

    echo 'Имя сессии: ' . session_name() . '<br>';

    // константа SID

    // будет непустой только при:

    // -- отказе от cookie и

    // -- разрешении использовать не только cookie

    echo 'Константа SID: <b>' . SID . '</b>';

?>
```

Директива **session.use_only_cookies** указывает на то, что необходимо использовать **только cookie**, а следовательно константа **SID** генерироваться не будет.

Идем дальше.

В демонстрационном примере **example_3 по умолчанию** выполнены следующие настройки текущего скрипта:

```
// использовать cookie
ini_set('session.use_cookies', 0); // нет
// использовать только cookie
ini_set("session.use_only_cookies", 0); // нет
```

example_3. index.php

```
<?php

    // настройки времени выполнения

    /*

        использовать cookie:

        1 - да

        0 - нет

    */

    ini_set('session.use_cookies', 0);

    /*

        использовать только cookie:

        1 - да

        0 - нет (можно использовать SID)

    */

    ini_set("session.use_only_cookies", 0);

    // старт сессии
```

```
session_start();

// вывод информации о сессии

echo 'Идентификатор сессии: ' . session_id() . '<br>';

echo 'Имя сессии: ' . session_name() . '<br>';

// константа SID актуальна

echo 'Константа SID: <b>' . SID . '</b>';

?>
```

Директива session.use_trans_sid включена

В демонстрационном примере **example_4** по умолчанию выполнены следующие настройки текущего скрипта:

```
// использовать cookie

ini_set('session.use_cookies', 0); // нет

// использовать только cookie

ini_set("session.use_only_cookies", 0); // нет

// использовать SID

ini_set("session.use_trans_sid", 1); // да
```

Обратите внимание на **автоматическое добавление к URL** ссылки константы **SID**.

example_4. index.php

```
<?php

// настройки времени выполнения

/*

    использовать cookie:

    1 - да

    0 - нет
```



```

*/

ini_set('session.use_cookies', 0);

/*

    использовать только cookie:

    1 - да

    0 - нет (можно использовать SID)

*/

ini_set("session.use_only_cookies", 0);

/*

    использовать SID:

    1 - да

    0 - нет

*/

ini_set("session.use_trans_sid", 1);

// старт сессии

session_start();

// сохраняем в сессию данные

$_SESSION['login'] = 'admin';

$_SESSION['msg'] = 'It\'s Work!';

// вывод информации о сессии

echo 'Идентификатор сессии: ' . session_id() . '<br>';

echo 'Имя сессии: ' . session_name() . '<br>';

echo 'Константа SID: <b>' . SID . '</b>';

// URL для перехода между страницами

echo '<a href="target.php">Переход к target.php</a>';

```

?>

В целевом файле убедимся, несмотря на то что **имя** и **значение сессии**, переданные через константу **SID** доступны, в сценарии файла **target.php** стартует новая сессия.

example_4. target-1.php

```
<?php

// старт сессии

session_start();

// вывод информации о сессии

echo 'Идентификатор сессии: ' . session_id() . '<br>';

echo 'Имя сессии: ' . session_name() . '<br>';

echo 'Константа SID: <b>' . SID . '</b>';

echo '<h3>Массив $_GET</h3>';

echo '<pre>';

print_r ($_GET);

echo '</pre>';

echo '<h3>Массив $_REQUEST</h3>';

echo '<pre>';

print_r ($_REQUEST);

echo '</pre>';

?>
```

Все просто. В файле **index.php** мы запретили файлы **cookie**, следовательно, не обнаружив **cookie** в заголовке запроса, функция **session_start()** сценария **target.php** стартует **новую сессию**.

Проблема решается передачей в качестве параметра функции **session_id()** идентификатора сессии, доступного в **GET**-параметре адресной строки.

```
// идентификатор сессии
```

```
session_id($_GET['PHPSESSID']);  
  
// старт сессии  
  
session_start();
```

example_4. target-2.php

```
<?php  
  
    // передача идентификатора сессии  
  
    session_id($_GET['PHPSESSID']);  
  
    // старт сессии  
  
    session_start();  
  
    // вывод информации о сессии  
  
    echo 'Идентификатор сессии: <b>' . session_id() . '</b><br>';  
  
    echo 'Имя сессии: ' . session_name() . '<br>';  
  
    echo 'Константа SID: <b>' . SID . '</b>';  
  
    echo '<h3>Массив $_GET</h3>';  
  
    echo '<pre>';  
  
    print_r ($_GET);  
  
    echo '</pre>';  
  
    echo '<h3>Массив $_REQUEST</h3>';  
  
    echo '<pre>';  
  
    print_r ($_REQUEST);  
  
    echo '</pre>';  
  
?>
```

В демонстрационном примере **example_5** для передачи константы **SID** задействуем:

- форму с методом **GET**;
- ссылку.

example_5. index.php

```
<?php

    // настройки времени выполнения

    // использовать cookie

    ini_set('session.use_cookies', 0); // нет

    // использовать только cookie

    ini_set("session.use_only_cookies", 0); // нет

    // использовать SID

    ini_set("session.use_trans_sid", 1); // да

    // старт сессии

    session_start();

    // сохраняем в сессию данные

    $_SESSION['login'] = 'admin';

    $_SESSION['msg'] = 'It\'s Work!';

    // вывод информации о сессии

    echo 'Идентификатор сессии: ' . session_id() . '<br>';

    echo 'Имя сессии: ' . session_name() . '<br>';

    echo 'Константа SID: <b>' . SID . '</b><p>';

?>

<!-- ... -->

<!--

    и форма и ссылка ссылаются на один целевой файл

-->

<form action='target.php' method='get'>

    <input type='text' value='Hello, User!' name='msg'>

    <button>Отправить</button>

</form>
```

```
<a href='target.php'>Переход к target.php</a><p>
```

Внимательно анализируйте взаимодействие сценариев передающего и принимающего файлов демонстрационного примера.

example_5. target.php

```
<?php

    // идентификатор сессии

    session_id($_GET['PHPSESSID']);

    // старт сессии

    session_start();

    // вывод информации о сессии

    echo 'Идентификатор сессии: ' . session_id() . '<br>';

    echo 'Имя сессии: ' . session_name() . '<br>';

    echo '<h3>Массив $_GET</h3>';

    echo '<pre>';

    print_r ($_GET);

    echo '</pre>';

    echo '<h3>Массив $_POST</h3>';

    echo '<pre>';

    print_r ($_POST);

    echo '</pre>';

?>
```

В демонстрационном примере **example_6** для передачи константы **SID** задействуем:

- форму с методом **POST**;
- ссылку.

Кроме того, старт существующей сессии в принимающем файле сделаем не передачей параметра для **session_id()**, а использованием **директив** функции **ini_set()**.

Важно. Принципиальная разница между алгоритмами двух примеров при одинаковом видимом результате.

example_6. index.php

```
<?php

// настройки времени выполнения

// использовать cookie

ini_set('session.use_cookies', 0); // нет

// использовать только cookie

ini_set("session.use_only_cookies", 0); // нет

// использовать SID

ini_set("session.use_trans_sid", 1); // да

// старт сессии

session_start();

// сохраняем в сессию данные

$_SESSION['login'] = 'admin';

$_SESSION['msg'] = 'It\'s Work!';

// вывод информации о сессии

echo 'Идентификатор сессии: ' . session_id() . '<br>';

echo 'Имя сессии: ' . session_name() . '<br>';

echo 'Константа SID: <b>' . SID . '</b>';

?>

<!-- ... -->
```

```
<!--  
    и форма и ссылка ссылаются на один целевой файл  
-->  
<form action='target.php' method='post'>  
    <input type='text' value='Hello, User!' name='msg'>  
    <button>Отправить</button>  
</form>  
<a href='target.php'>Переход к target.php</a><p>
```

example_6. target.php

```
<?php  
    // использовать cookie  
    ini_set('session.use_cookies', 0); // нет  
    // использовать только cookie  
    ini_set("session.use_only_cookies", 0); // нет  
    // использовать SID  
    ini_set("session.use_trans_sid", 1); // да  
    // старт сессии  
    session_start();  
    // вывод информации о сессии  
    echo 'Идентификатор сессии: ' . session_id() . '<br>';  
    echo 'Имя сессии: ' . session_name() . '<br>';  
    echo '<h3>Массив $_GET</h3>';  
    echo '<pre>';  
    print_r ($_GET);  
    echo '</pre>';  
    echo '<h3>Массив $_POST</h3>';
```

```
echo '<pre>';

print_r ($_POST);

echo '</pre>';

?>
```

Директива `session.use_trans_sid` выключена

Немного сложнее реализовать процесс обмена идентификатором сессии при выключенной директиве `session.use_trans_sid`.

```
// настройки времени выполнения

ini_set('session.use_cookies', 0);

ini_set("session.use_only_cookies", 0);

ini_set("session.use_trans_sid", 0);
```

В примере **example_7** выполняем переход к целевому файлу через:

- ссылку **с параметром** SID;
- ссылку **без параметра** SID.

example_7. index.php

```
<?php

// настройки времени выполнения

ini_set('session.use_cookies', 0);

ini_set("session.use_only_cookies", 0);

ini_set("session.use_trans_sid", 0);

// старт сессии

session_start();

// вывод информации о сессии

echo 'Идентификатор сессии: ' . session_id() . '<br>';
```



```

echo 'Имя сессии: ' . session_name() . '<br>';

echo 'Константа SID: <b>' . SID . '</b>';

// сохраняем в сессию данные

// роль

$_SESSION['role'] = 'admin';

// реквизиты банковской карты

$_SESSION['card_owner'] = 'DAUGAVA EVGENIY';

$_SESSION['card_number'] = '1254 4568 4687 2135';

$_SESSION['card_date'] = '12/25';

$_SESSION['card_cvc'] = '126';

?>

<!-- ... -->

<!-- ссылка с параметром SID -->

<a href='target.php?<?=SID; ?>'>Ключ к благосостоянию кроется за
этой ссылкой!</a><p>

<!-- ссылка без параметра SID -->

<a href='target.php'>Или за этой...</a><p>

```

example_7. target.php

```

<?php

// идентификатор сессии из адресной строки

session_id($_GET['PHPSESSID'] ?? null);

// старт сессии

session_start();

// вывод информации о сессии

echo 'Идентификатор сессии: ' . session_id() . '<br>';

echo 'Имя сессии: ' . session_name() . '<br>';

```

```

    echo 'Константа SID: <b>' . SID . '</b>';

?>

<!-- ... -->

<?php

    // если существует сессия role и role=admin

    if(

        isset($_SESSION['role']) &&

        $_SESSION['role'] == 'admin'

    ) {

        echo "<h2>Будьте здоровы, живите богато! ;)</h2>";

        echo "<pre>";

        print_r($_SESSION);

        echo "</pre>";

    } else {

        echo "<h2>У вас нет права на роскошную жизнь... :(</h2>";

    }

    // удаляем данные и саму сессию

    session_unset();

    session_destroy();

?>

```

В примере **example_8** передаем идентификатор сессии через:

- текстовое поле формы методом **POST** (в реальном проекте поле должно быть скрытым);
- ссылку (метод **GET**).

Обратите внимание на настройки времени выполнения.

example_8. index.php

```
<?php

    // настройки времени выполнения

    // cookie не используем

    // SID используем

    // SID передаем вручную

    ini_set('session.use_cookies', 0);

    ini_set("session.use_only_cookies", 0);

    ini_set("session.use_trans_sid", 0);

    // старт сессии

    session_start();

    // вывод информации о сессии

    echo 'Идентификатор сессии: ' . session_id() . '<br>';

    echo 'Имя сессии: ' . session_name() . '<br>';

    echo 'Константа SID: <b>' . SID . '</b>';

    // сохраняем в сессию данные

    // роль

    $_SESSION['role'] = 'admin';

    // реквизиты банковской карты

    $_SESSION['card_owner'] = 'DAUGAVA EVGENIY';

    $_SESSION['card_number'] = '1254 4568 4687 2135';

    $_SESSION['card_date'] = '12/25';

    $_SESSION['card_cvc'] = '126';

?>

<!-- ... -->

<!-- форма с полем для передачи идентификатора сессии -->

<form action='target.php' method='post'>
```

```

<input type='text' value='Hello, User!' name='msg'><p>

<input type='text' value='<?= session_id(); ?>'
name='PHPSESSID'><p>

<button>Измени свои мысли и ты изменишь мир!</button>

</form><p>

<!-- ссылка с параметром SID -->

<a href='target.php?<?= SID; ?>'>Follow your heart</a><p>

<!-- ссылка без параметра -->

<a href='target.php'>Мечтатели - это спасители мира</a>

```

example_8. target.php

```

<?php

    // идентификатор сессии из поля формы

    if (isset($_POST['PHPSESSID']))

        $sid = $_POST['PHPSESSID'];

    if (isset($_GET['PHPSESSID']))

        $sid = $_GET['PHPSESSID'];

    session_id($sid ?? null);

    // старт сессии

    session_start();

    // вывод информации о сессии

    echo 'Идентификатор сессии: ' . session_id() . '<br>';

    echo 'Имя сессии: ' . session_name() . '<br>';

    echo 'Константа SID: <b>' . SID . '</b>';

?>

<!-- ... -->

<?php

```

```

// если существует сессия role и role=admin

    if(

        isset($_SESSION['role']) &&

        $_SESSION['role'] == 'admin'

    ) {

        echo "<h2>Пользуйтесь на здоровье! ;)</h2>";

        echo "<pre>";

        print_r($_SESSION);

        echo "</pre>";

        // для следующих корректных переходов очистим массив

        session_unset();

    } else {

        echo "<h2>Деньги возьмете в тумбочке ... :(</h2>";

    }

?>

```

Немного усложним сценарий. В демонстрационном примере **example_9** **добавим капчу** и **промежуточную** страницу для добавления данных в сессию. Не забывайте, настройки времени выполнения актуальны только для сценария, в котором установлены.

Важно. Настройки времени выполнения **актуальны** только для сценария, в котором установлены.

example_9. index.php

```

<?php

// набор символов для генерации капчи

$chars = '0123456789abcdefghijklmnopqrstuvwxyz';

```

```

$captcha = substr(str_shuffle($chars), 0, 5);

// настройки времени выполнения

ini_set('session.use_cookies', 0);

ini_set("session.use_only_cookies", 0);

ini_set("session.use_trans_sid", 1);

// старт сессии

session_start();

// вывод информации о сессии

echo 'Идентификатор сессии: ' . session_id() . '<br>';

echo 'Имя сессии: ' . session_name() . '<br>';

echo 'Константа SID: <b>' . SID . '</b>';

// сохраняем в сессию капчу

$_SESSION['captcha'] = $captcha;

?>

<!-- ... -->

Повторите символы на картинке :)<p>

<?= "<span>$captcha</span>" ?><p>

<form action="page.php" method="post">

    <!--

        PHP использует форму для передачи SID

    -->

    <input type="text" name="captcha" value=""><p>

    <button>Отправить</button>

</form>

```

example_9. page.php

```
<?php
```

```
// настройки времени выполнения

ini_set('session.use_cookies', 0);

ini_set("session.use_only_cookies", 0);

ini_set("session.use_trans_sid", 1);

// старт сессии

session_start();

// вывод информации о сессии

echo 'Идентификатор сессии: ' . session_id() . '<br>';

echo 'Имя сессии: ' . session_name() . '<br>';

echo 'Константа SID: <b>' . SID . '</b>';

?>

<!-- ... -->

<?php

    if (empty($_POST['captcha'])) {

        echo "<h2>Вы капчу видели, товарищ??</h2>";

        echo "<a href='index.php'>Назад</a>";

    }

    elseif ($_POST['captcha'] !== $_SESSION['captcha']) {

        echo "<h2>Спасибо, конечно, но это не то ...(</h2>";

        echo "<a href='index.php'>Попробовать еще раз</a>";

    }

    elseif ($_POST['captcha'] === $_SESSION['captcha']) {

        // если капчи равны запишем в сессию данные

        echo "<h2>Данные о педагоге сохранены в сессию</h2>";

        echo "<a href='target.php'>Далее</a>";

        $_SESSION['user'] = array(

            'id_personnel' => 1,
```

```
'surname' => 'Бородина',  
  
'name' => 'Ксения',  
  
'patronymic' => 'Алексеевна',  
  
'post' => 'Преподаватель',  
  
'category' => 'Соответствие занимаемой должности',  
  
'level_edu' => 'Высшее профессиональное ',  
  
'experience_total' => 21.3,  
  
'experience_college' => 14.5,  
  
'email' => '',  
  
'site' => ''  
  
);  
  
}  
  
?>
```

example_9. target.php

```
<?php  
  
    // идентификатор сессии из GET-параметра  
  
    session_id($_GET['PHPSESSID']);  
  
    // старт сессии  
  
    session_start();  
  
    // вывод информации о сессии  
  
    echo 'Идентификатор сессии: ' . session_id() . '<br>';  
  
    echo 'Имя сессии: ' . session_name() . '<br>';  
  
    echo 'Константа SID: <b>' . SID . '</b>';  
  
?>  
  
<!-- ... -->  
  
<?php
```



```
if (isset($_SESSION['user'])) {  
    echo "<h2>Персональные данные педагога</h2>";  
    echo "<pre>";  
    print_r($_SESSION['user']);  
    echo "</pre>";  
} else {  
    echo "<h2>Ой, что-то пошло не так...</h2>";  
}  
?>
```

Междоменные сессии

Как мы рассмотрели в одно из предыдущ файлов **cookie** при междоменных запросах не передаются (это не значит, что этот вопрос в принципе не решаем). А значит идентификатор сессии тоже передать нельзя.

В демонстрационном примере **example_10** решим задачу передачи идентификатора сессии посредством самостоятельного прикрепления константы **SID** к **URL** ссылки.

Обратите внимание, целевые файлы **target-1.php** и **target-2.php** должны находиться на разных доменах.

example_10. index.php

```
<?php  
// настройки времени выполнения  
ini_set('session.use_cookies', 0);  
ini_set("session.use_only_cookies", 0);  
ini_set("session.use_trans_sid", 0);
```

```

// старт сессии

session_start();

// записываем в сессию данные

$_SESSION['name'] = "Авиценна";

$_SESSION['cite'] = array(

    'Нет безнадежных больных. Есть только безнадежные
врачи.',

    'Камень не может не достичь своей цели.',

    'Кто счастья не ценил, тот близится к несчастью.'

)

?>

<!-- ... -->

<a href='/target-1.php?= SID ?'>Посмотреть информацию о сессии
на ТЕКУЩЕМ сайте</a><p>

<a href='http://site/target-2.php?= SID ?'>Просмотреть
информацию о сессии на СТОРОННЕМ сайте</a>

```

example_10. target-1.php

```

<?php

// настройки времени выполнения

ini_set('session.use_cookies', 0);

ini_set("session.use_only_cookies", 0);

ini_set("session.use_trans_sid", 1);

// старт сессии

session_start();

?>

<!-- ... -->

```

```
<?php

    // вывод сессионных данных

    echo "<h2>Философ и врач - {$_SESSION['name']}
```

example_10. target-2.php

```
<?php

    // идентификатор сессии

    session_id($_GET['PHPSESSID']);

    // старт сессии

    session_start();

?>

<!-- ... -->

<?php

    // вывод сессионных данных

    echo "<h2>Философ и врач - {$_SESSION['name']}
```

Время жизни сессии

От времени действия сессии зависит период работоспособности PHP-переменных `$_SESSION`, и как следствие, активность веб-приложения. Например, если пользователь авторизовался в системе, время в течении которого он может бездействовать без необходимости повторного ввода логина и пароля зависит от времени жизни сессии.

По умолчанию время жизни сессий всего 24 минуты, а **cookies** сохраняются до закрытия браузера, такие параметры установлены на многих хостингах. Это не приемлемо если у вас интернет-магазин, покупатель может положить товары в корзину и вернуться на сайт через час или больше, а корзина уже пуста.

Конфигурационный файл `php.ini`

```
session.cookie_lifetime = 172800
```

Конфигурационный файл в `.htaccess`

```
php_value session.gc_maxlifetime 172800  
php_value session.cookie_lifetime 172800
```

Функция `ini_set()`

Установка директив

```
ini_set('session.gc_maxlifetime', 172800);  
ini_set('session.cookie_lifetime', 0);
```

Чтение директив

```
$maxlifetime = ini_get('session.gc_maxlifetime');  
$lifetime = ini_get('session.cookie_lifetime');
```

В демонстрационном примере **example_11** увеличим время жизни сессии.

example_11. index.php

```
<?php
    // установка значений директив

    ini_set('session.gc_maxlifetime', 86401);

    ini_set('session.cookie_lifetime', 0);

    // session_set_cookie_params(0);

    // получение значений директив

    $maxlifetime = ini_get("session.gc_maxlifetime");

    $lifetime = ini_get("session.cookie_lifetime");

    session_start();

    // вывод значений директив

    echo 'session.gc_maxlifetime = ' . $maxlifetime . '<p>';

    echo 'session.cookie_lifetime = ' . $lifetime;

?>
```

Заключение

В качестве некоторого подведения итогов урокам, посвященных базовым знаниям о куках и сессиях несколько утверждений:

Куки (cookies):

- создаются браузером по *просьбе* сервера. Браузер не обязан выполнять *просьбы* сервера;
- хранятся на клиенте;
- создаются функцией **setcookie()**, доступны в сценарии в массиве **\$_COOKIE**;
- передаются в **HTTP**-заголовках;
- имеют определенную область видимости;

- имеют определенное время жизни;
- однажды установленная кука будет передаваться браузером при каждом запросе к ресурсу, который ее установил;
- с помощью **cookie** передается идентификатор сессии;
- удаляются браузером по *просьбе* сервера. Браузер не обязан выполнять *просьбы* сервера.

Сессии (session):

- создаются и хранятся на сервере;
- данные сессии автоматически сериализуются / десериализуются;
- прежде чем создать сессию, ее нужно стартовать, если при старте окажется, что такая сессия существует – сеанс сессии будет продлен;
- доступны в массиве \$_**SESSION**;
- имя и идентификатор сессии устанавливаются автоматически, но могут быть переопределены;
- идентификатор сессии — это имя серверного файла (часть имени), в котором хранятся сессионные данные;
- если браузер *отказывается* принимать **cookie** с идентификатором сессии, идентификатор можно передать и другим способом.

Сессии и куки очень важный инструмент веб-разработчика и имеют множество дополнительных настроек и директив. Вопрос использования дополнительных настроек – уровень квалификации разработчика, рано или поздно с этим придется столкнуться.

