COOKIE в PHP. Основы

Введение

HTTP-cookie, или просто **cookie** (англ. *печенье*) — это небольшие **текстовые файлы с фрагментами данных о посещении сайта**. Они могут содержать логин, пароль, геолокацию, язык, страницы перехода и другую информацию о вас и ваших действиях.

Текстовые файлы **cookie** генерируются сервером интернет-ресурса, по мере веб-сёрфинга **cookie** пополняются новыми сведениями.

Принцип работы cookie: когда пользователь вводит адрес страницы в браузере, он ищет на устройстве cookie-файл с этого сайта. Если такой файл имеется, он **пересылается на сервер ресурса**, где и используется. Если файл **cookie** не найден, ресурс воспримет пользователя как **нового посетителя** и запросит согласие на обработку данных (например).

Таким образом, каждый раз, когда вы посещаете сайт, который использует **cookie**, устройство отправляет сохраненные данные на сервер для того, чтобы правильно опознать вас среди других пользователей.

> ***Важно***. **Cookies** (cookie) – это механизм хранения данных браузером

для отслеживания или идентификации возвращающихся посетителей.

Информация **cookies** хранится на жестком диске компьютера пользователя. Куки бывают временными и постоянными.

Важно.

- Cookie **передаются клиенту** вместе с другими HTTP-заголовками, поэтому **setcookie**() должна быть вызвана до вывода в браузер.
- Cookie **передаются от клиента серверному сценарию** вместе с другими HTTP-заголовками, при последующих запросах клиента.

При работе с cookie выполняется следующий жизненный цикл:

- **установка** cookies;
- **чтение** cookies;
- **удаление** cookies.

Механизм взаимодействия клиента и сервера через **cookies** важен еще и потому, что **cookie** играют важную роль в организации **сессий**.

Очень важно. При **тестировании** демонстрационных примеров не забывайте **очищать браузер от данных cookie**. Иначе получите неверный результат работы скрипта и как следствие, неверное представление о механизме работы **cookies**.

Установка cookies

Данные **cookie** устанавливаются с помощью функции **setcookie**().

setcookie()

setcookie () — отправляет cookie.

setcookie() задаёт cookie, которое будет передано клиенту вместе с другими HTTP-заголовками. Как и любой другой заголовок, **cookie** должны передаваться до того, как будут выведены какие-либо другие данные скрипта (это ограничение протокола). Это значит, что в скрипте вызовы этой функции должны располагаться до остального вывода, включая вывод тегов <**html**> и <**head**>, а также **пустые строки** и **пробельные символы**.

> ***Важно***. Вызовы **setcookie**() должны располагаться до вывода в браузер, включая вывод любых тегов, а также **пустых строк** и **пробельных символов**.

После передачи клиенту данных файлов cookie станут доступны через массив $_**COOKIE при следующей загрузке страницы**. Значения cookie также есть в $_**REQUEST**.

> ***Важно***. Данные **cookie** доступны при следующей загрузки страницы. Другими словами – при посещении любой страницы сайта после установки cookie. Почему так происходит, рассмотрим в одном из примеров.

Описание

```
setcookie($name, $value, $expires, $path, $domain, $secure, $httponly);
```

Параметры

- **name** – название;
- **value** – значение;
- **expires** – время жизни (метка времени Unix), если 0 или пропустить аргумент, cookie будут действовать до закрытия браузера.

- **path** – путь к директории, из которой будут доступны cookie. Если задать '/', cookie будут доступны во всем домене.
- **domain** – домен, которому доступны cookie.

- **secure** – при true значения cookie будут доступны только по HTTPS.
- **httponly** – при **true**, cookie будут доступны только через HTTP-протокол.

Возвращаемые значения

Если перед вызовом функции клиенту уже передавался какой-либо вывод (теги, пустые строки, пробелы, текст и т.п.), **setcookie**() потерпит неудачу и вернёт **false**.

Если **setcookie**() успешно отработает, то вернёт **true**. Это, однако, не означает, что клиентское приложение (браузер) правильно приняло и обработало cookie.

> ***Важно***. При работе с файлами cookie очень важно понимать следующее:
>
> - Успешно отправленные данные не гарантия того, что эти данные будут **приняты** и правильно **обработаны клиентом**.
> - Кроме того, нет гарантии, в **целостности** и **идентичности** данных, полученных сервером от клиентского приложения.

Пример установки файлов **cookie**:

```
// файлы cookie должны быть установлены до вывода в браузер
// устанавливаем cookie login
setcookie('login', 'admin');
```

Доступны значения **cookies** будут только на следующей странице приложения.

Альтернативное описание функции setcookie

Для функции **setcookie** доступно альтернативное описание.

```
setcookie($name, $value, $options);
```

Параметры

- **name** – название;
- **value** – значение;
- **options** – ассоциатиыный массив, который может содержать любой из ключей: **expires**, **path**, **domain**, **secure**, **httponly**.

```
// классическая установка файла cookie
setcookie("team[$key]", $value, 0, '/');;
// альтернативная установка файла cookie
setcookie("team[$key]", $value, array('expires'=>0,
'path'=>'/'))
```

Хранение cookie

Хранение файлов cookie зависит от браузера. Если вы используете браузер **Google Chrome**, файлы **cookie** хранятся в следующей директории:

C:\Users\Your_User_Name\AppData\Local\Google\Chrome\User Data\Default\Network

Информация, содержащаяся в файле **cookie** недоступна для чтения человеком. Для того, чтобы просматривать файлы **cookie** и управлять ими, следует использовать **интерфейс браузера**.

Максимальный размер cookies

Cookies передаются в виде **HTTP-заголовка**, это накладывает на них ограничения. Максимальный размер куки составляет **4096** байт. Длина каждого атрибута не может превышать **1024** байтов. В случае превышения размера файлы будут отклонены.

Атрибут SameSite

Атрибут **SameSite** используется для управления межсайтовыми запросами. Тем не менее, некоторые браузеры могут выдавать такое предупреждение при неправильном использовании **cookie**.

Чтение cookies

После передачи клиенту заголовков **cookie** станут доступны через глобальный массив $_**COOKIE** при следующей загрузке страницы. Значения **cookie** также есть в других глобальных массивах ($_SERVER, $_REQUEST, $GLOBALS), рассмотрим в демонстрационном примере **example_1**.

Например, вывести одно конкретное значение cookie:

```
// если cookie существует и равна admin
if (isset($_COOKIE['login'])) {
    // выводим cookie
    // данные cookie находятся в массиве $_COOKIE
    echo "<h2>Добро пожаловать на сайт!</h2>";
    echo "Вы зашли как: <b>" . $_COOKIE['login'] . "</b>";
}
```

В демонстрационном коде **example_1** приведен пример установки и несколько способов получения данных **cookie**.

example_1. index.php

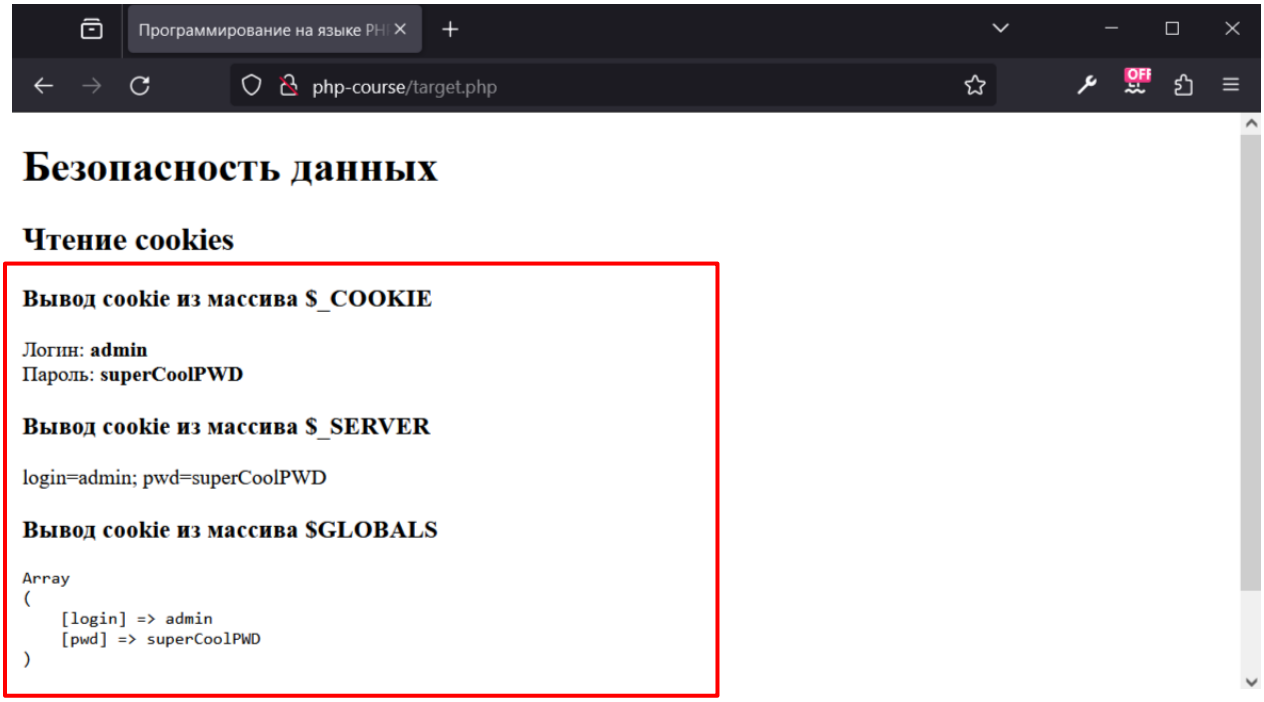
```php
<?php
    // файлы cookie должны быть установлены до вывода в браузер
    // устанавливаем cookie login
    setcookie('login', 'admin');
    // устанавливаем cookie pwd
    setcookie('pwd', 'superCoolPWD');
?>
```

Сценарий файла **target.php** выводит в браузер данные **cookie**, установленные в файле **index.php**.

example_1. target.php

```php
<?php
    // если cookie существуют
    if (
        isset($_COOKIE['login']) &&
        isset($_COOKIE['pwd'])
    )
    {
        // данные cookie находятся в массиве $_COOKIE
        echo '<h3>Вывод cookie из массива $_COOKIE</h3>';
        echo 'Логин: <b>' . $_COOKIE['login'] . '</b><br>';
        echo 'Пароль: <b>' . $_COOKIE['pwd'] . '</b>';
        // cookie также хранятся в массиве $_SERVER
        echo '<h3>Вывод cookie из массива $_SERVER</h3>';
        echo $_SERVER['HTTP_COOKIE'];
        // или в массиве $GLOBALS
```

```php
        echo '<h3>Вывод cookie из массива $GLOBALS</h3>';
        echo '<pre>';
        print_r ($GLOBALS['_COOKIE']);
        echo '</pre>';
        // или в массиве $_REQUEST (опционально)
        echo '<h3>Вывод cookie из массива $_REQUEST</h3>';
        echo '<pre>';
        print_r ($_REQUEST);
        echo '</pre>';
    }
?>
```

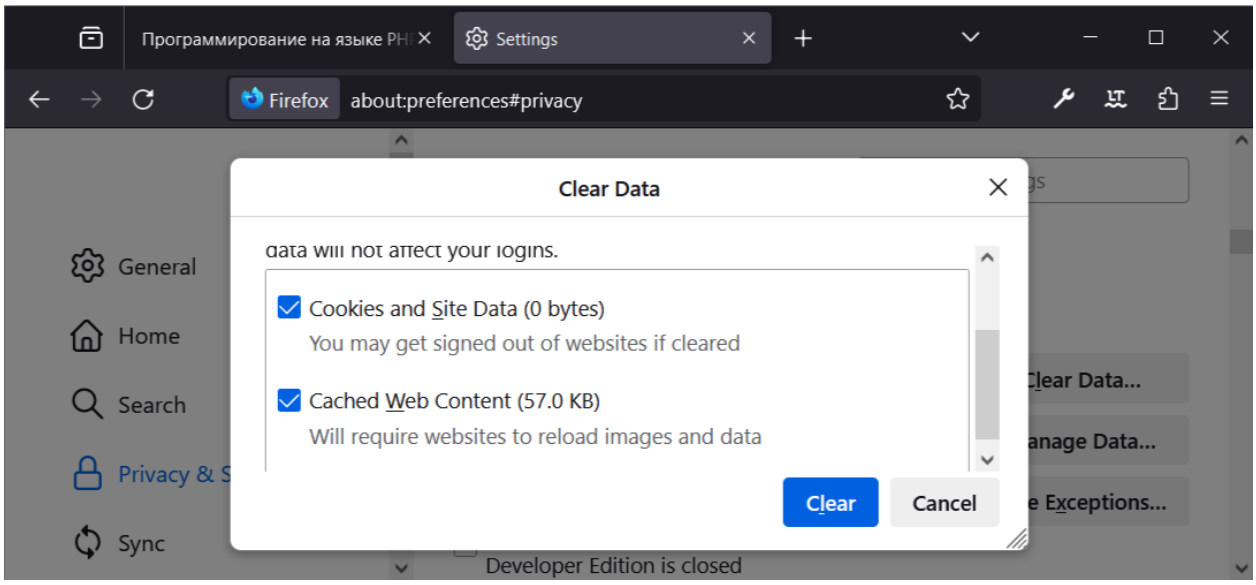


Глобальный массив $_**REQUEST** может не содержать переменные из $_**COOKIE**, если параметр `request_order` в файле **php.ini** имеет значение "GP".

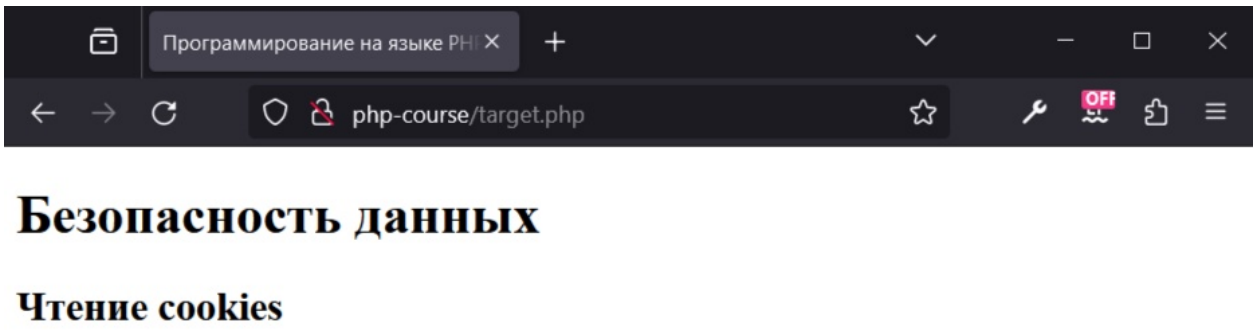
Это означает, что в массив $_REQUEST будут заноситься только $_**GET** или $_**POST**. Если переменная передаётся через $_COOKIE, то в массиве она не будет доступна.

Чтобы добавить в $_**REQUEST** переменные из $_**COOKIE**, нужно изменить параметр `request_order` на "GPC".

Проведем небольшой эксперимент, почистим **cookies** браузера.



После **обновления** страницы cookies исчезли, вывод сценария поменялся.



Алгоритм обмена данными cookie

В демонстрационном примере **example_1** мы рассмотрели пример установки и чтения файла **cookie**. Сейчас продемонстрирую алгоритм обмена файлами между сервером и клиентом. Это важно.

Данные **cookie** доступны из **заголовков запроса браузера** при повторном посещении ресурса. Поэтому, при установке **новых cookie текущего сценария**, данные **cookie** не будут доступны.

> ***Важно***. Массив $_**COOKIE** формируется из заголовков запроса браузера, а не при вызове функции **setcookie**().

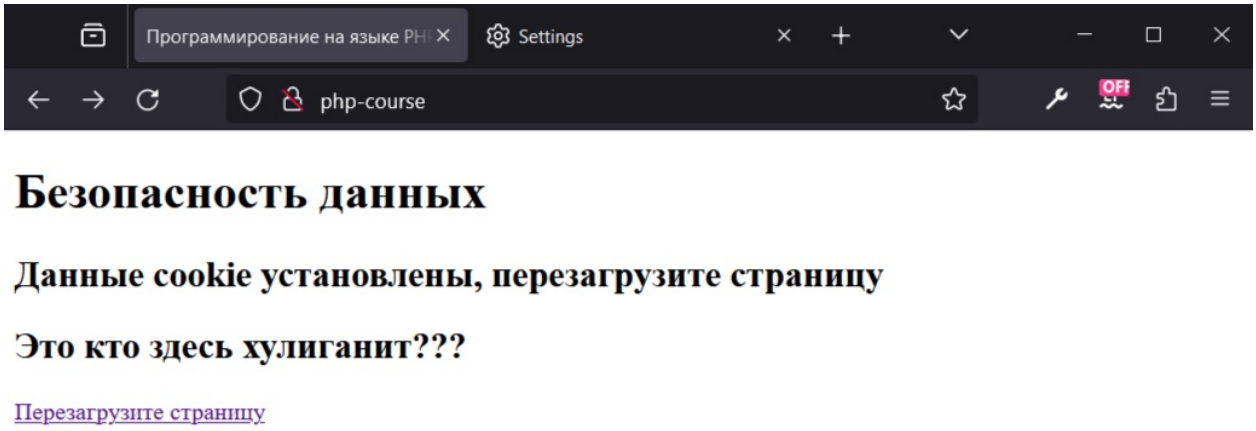
В демонстрационном примере **example_2** данные cookie доступны **только при перезагрузке** страницы. Не забудьте почистить данные **cookie** из предыдущего примера.

example_2. index.php

```
<?php
    // файлы cookie должны быть установлены до вывода в браузер
    // устанавливаем cookie с именем login и значением admin
    setcookie('login', 'admin');
    // устанавливаем cookie с именем pwd и значением superCoolPWD
    setcookie('pwd', 'superCoolPWD');
?>
<!-- ... -->
<?php
    // если cookie login и pwd существуют
    // выведем их значения в браузер
    if (
```

```
        isset($_COOKIE['login']) &&
        isset($_COOKIE['pwd'])
    )
    {
        echo 'login = <b>' . $_COOKIE['login'] . '</b>';
        echo "<br>";
        echo 'password = <b>' . $_COOKIE['pwd'] . '</b>';
    }
    else {
        // иначе выведем сообщение
        echo "<h2>Это кто здесь хулиганит???</h2>";
    }
?>
```

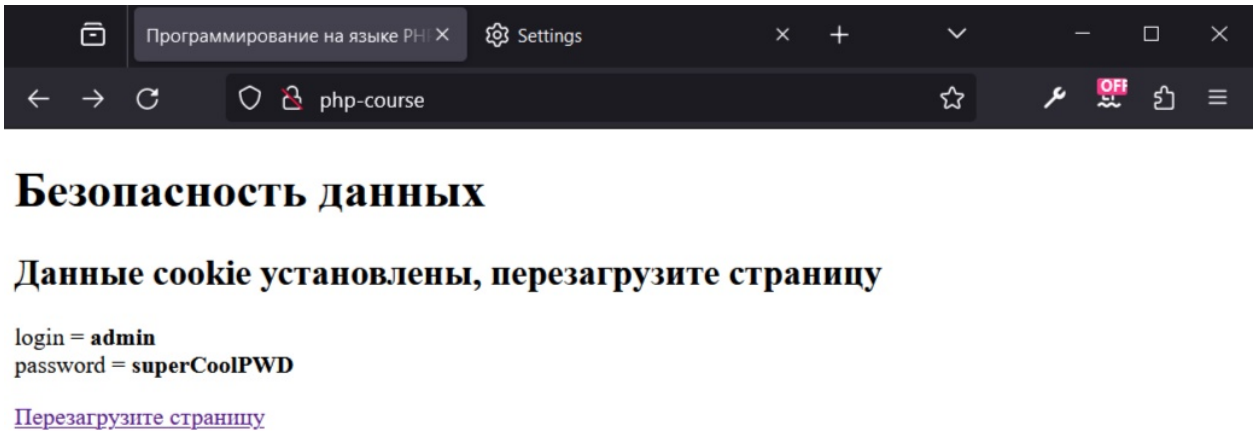
Логику работы приведенного сценария подробно разберу в материале ниже.



При перезагрузке страницы файлы **cookie** уже установлены, поэтому сработает следующее условие:

```
// если cookie login и pwd существуют
```

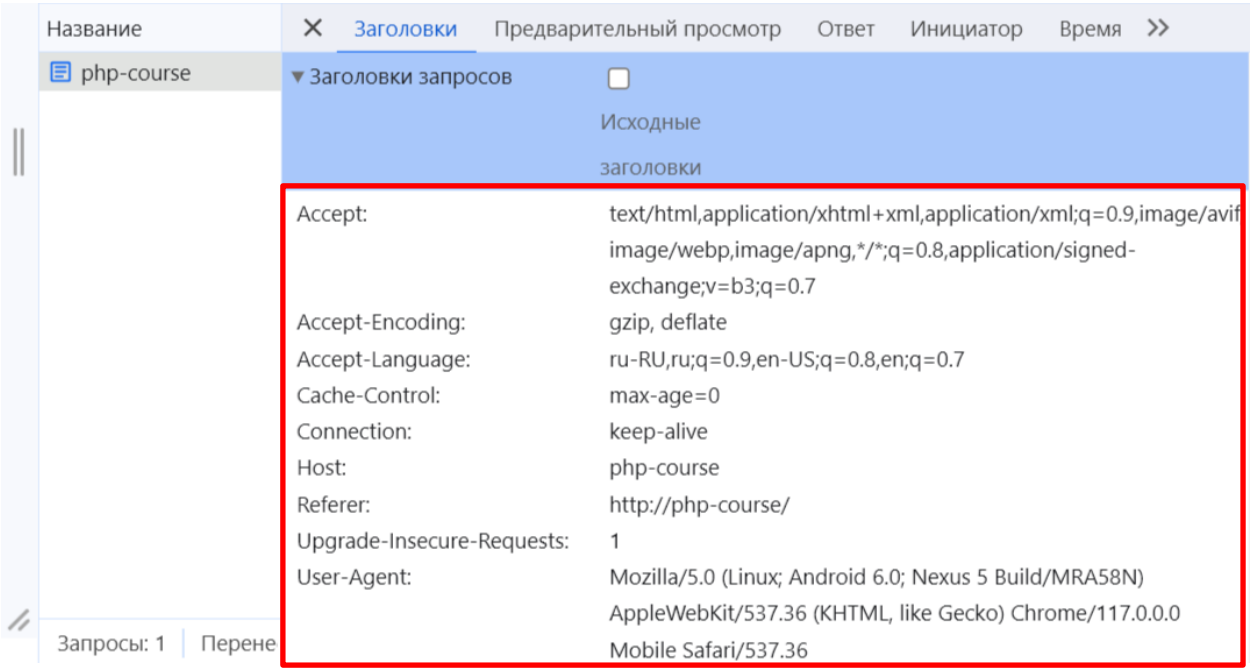
```php
// выведем их значения в браузер
if (
    isset($_COOKIE['login']) &&
    isset($_COOKIE['pwd'])
) {
    // ...
}
```



На простом примере важно понять **алгоритм обмена данными cookie** между клиентом и сервером. **Проанализируем обмен данными в режиме разработчика** (клавиша F12).

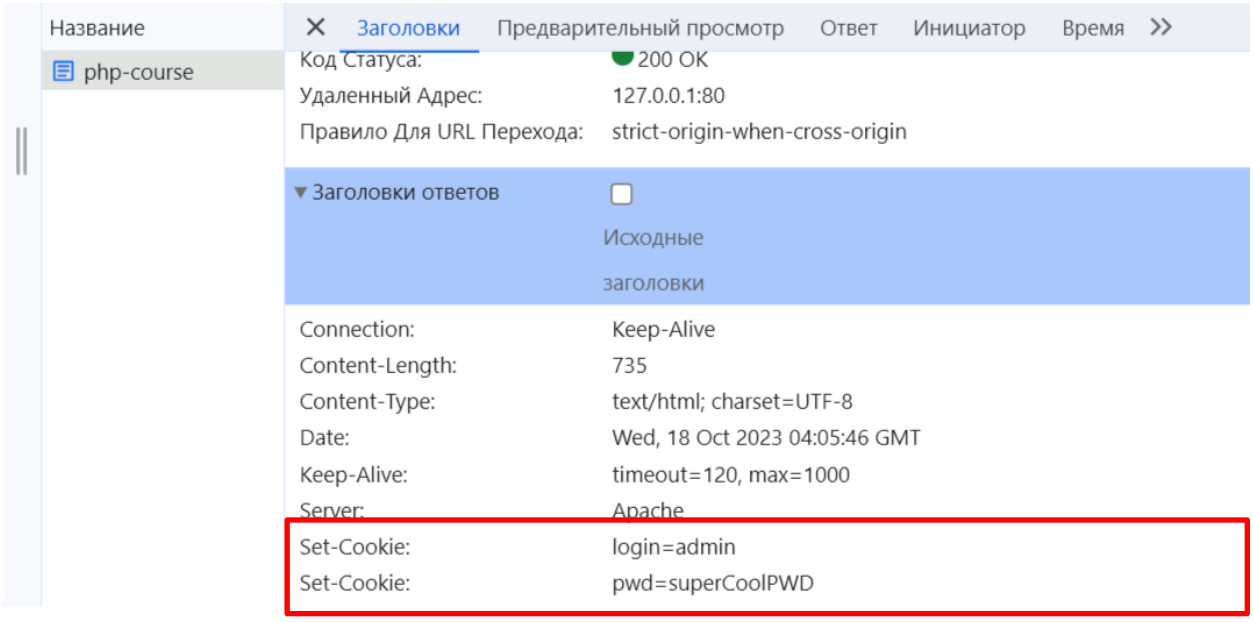
> ***Важно***. Мой проект расположен на локальном ресурсе **php-course**. Названия ваших домена или файлов могут отличаться, логика обмена данными при этом не меняется.

1. Запрос **клиентом** ресурса **http://php-course/index.php**. Пока нет никаких **cookie**.



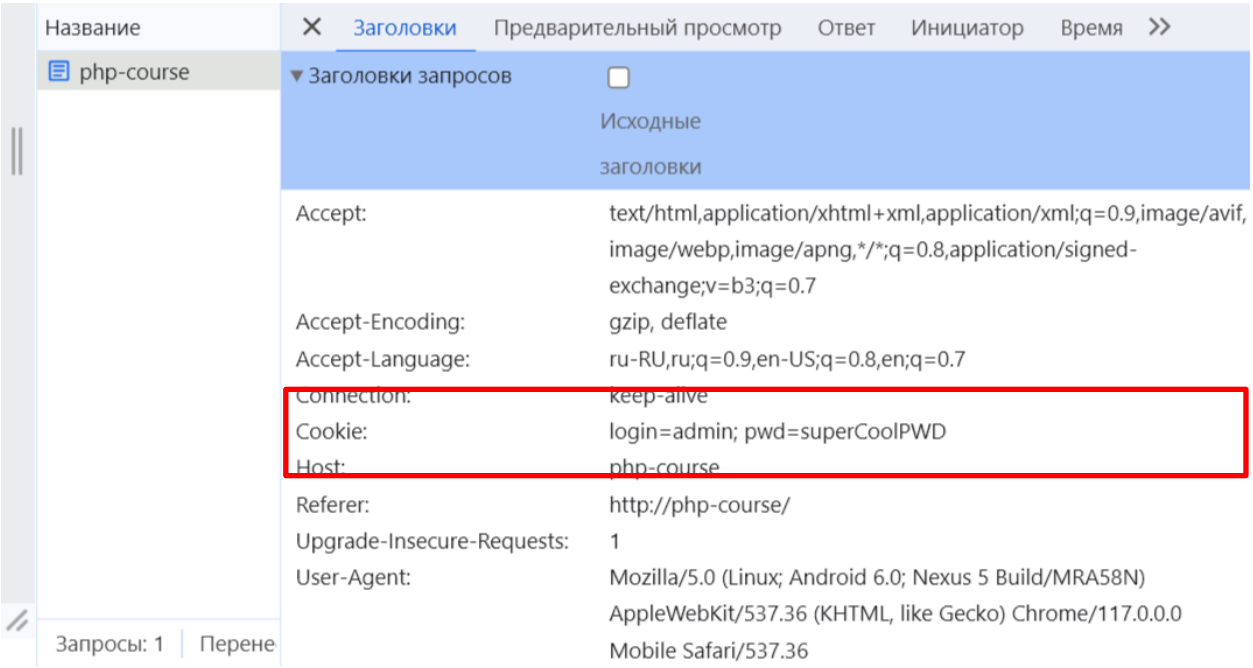
2. **Сценарий** файла **index.php** устанавливает файлы **cookie** и с помощью функции **setcookie**() отправляет заголовки клиенту с данными этих файлов.

```php
<?php
    // файлы cookie должны быть установлены до вывода в браузер
    // устанавливаем cookie с именем login и значением admin
    setcookie('login', 'admin');
    // устанавливаем cookie с именем pwd и значением superCoolPWD
    setcookie('pwd', 'superCoolPWD');
?>
```



3. При перезагрузке страницы клиент повторно запрашивает страницу **http://php-course/index.php.** Но теперь в запросе присутствуют данные файлов **cookie**.

```
<!-- перезагрузка страницы -->
<p><a href="/">Перезагрузите страницу</a></p>
```

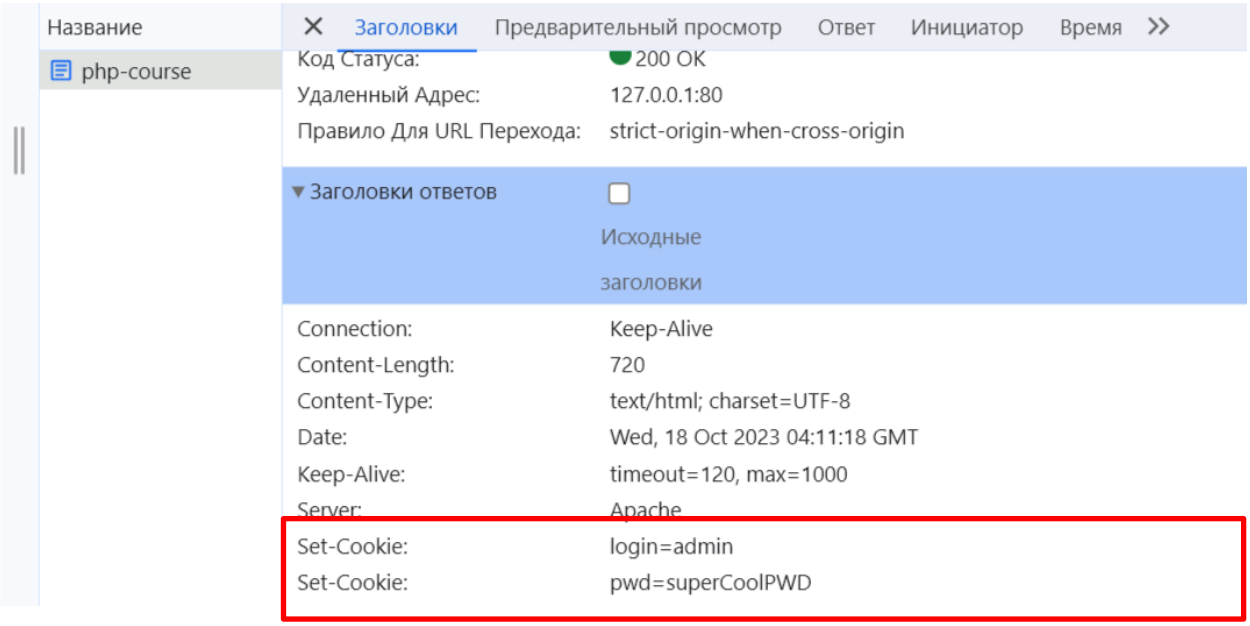


4. Сценарий файла **index.php при повторной загрузке** получает **доступ** к данным через глобальный массив $_**COOKIE**.

```
<?php
```

```php
    // если cookie login и pwd существуют
    // выведем их значения в браузер
    if (
        isset($_COOKIE['login']) &&
        isset($_COOKIE['pwd'])
    ) {
        echo 'login = <b>' . $_COOKIE['login'] . '</b>';
        echo "<br>";
        echo 'password = <b>' . $_COOKIE['pwd'] . '</b>';
    } else {
        // иначе выведем сообщение
        echo "<h2>Это кто здесь хулиганит???</h2>";
    }
?>
```

Обратите внимание, файлы **cookie** опять отправляются в браузер **клиента**.



Следующий фрагмент кода никуда не делся:

```php
<?php
```

```
    // файлы cookie должны быть установлены до вывода в браузер
    // устанавливаем cookie с именем login и значением admin
    setcookie('login', 'admin');
    // устанавливаем cookie с именем pwd и значением superCoolPWD
    setcookie('pwd', 'superCoolPWD');
?>
```

В следующем демонстрационном примере исправим код **example_2** таким образом, чтобы данные **cookie** отправлялись клиенту один раз при их инициализации.

Включите режим разработчика, протестируйте код. Отследите обмен данными между клиентом и серверным сценарием.

example_3. index.php

```
<?php
    // файлы cookie должны быть установлены до вывода в браузер
    // установим cookie, только если они не существуют
    if (
        empty($_COOKIE['login']) &&
        empty($_COOKIE['pwd'])
    ) {
        // устанавливаем cookie с именем login и значением admin
        setcookie('login', 'admin');
        // устанавливаем cookie с именем pwd и значением
        superCoolPWD
        setcookie('pwd', 'superCoolPWD');
    }
?>
<!-- ... -->
```

```php
<?php
    // если cookie login и pwd существуют
    // выведем их значения в браузер
    if (
        isset($_COOKIE['login']) &&
        isset($_COOKIE['pwd'])
    )
    {
        echo 'login = <b>' . $_COOKIE['login'] . '</b>';
        echo "<br>";
        echo 'password = <b>' . $_COOKIE['pwd'] . '</b>';
    }
    else {
        // иначе выведем сообщение
        echo "<h2>Это кто здесь хулиганит???</h2>";
    }
?>
```

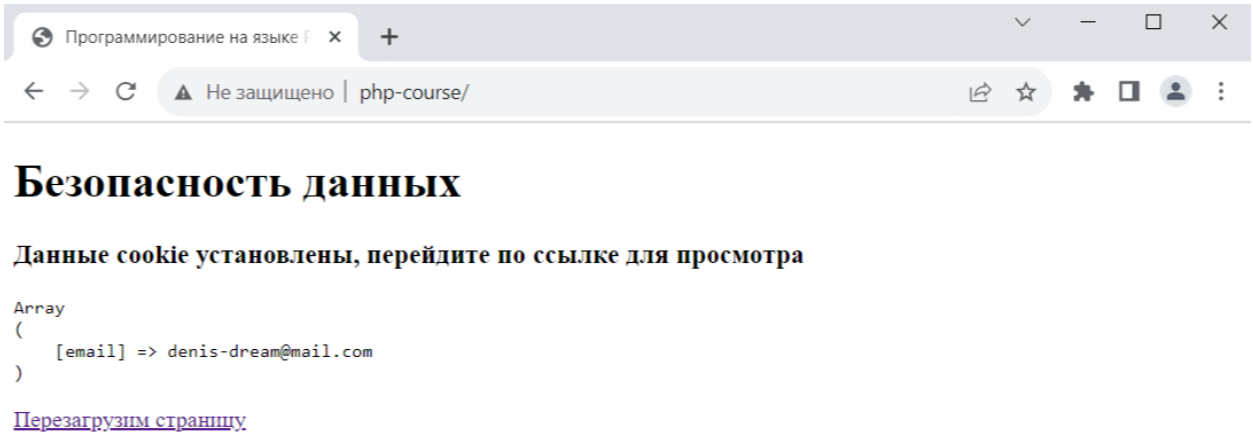
Идем дальше.

В демонстрационном примере **example_4** есть следующий фрагмент кода:

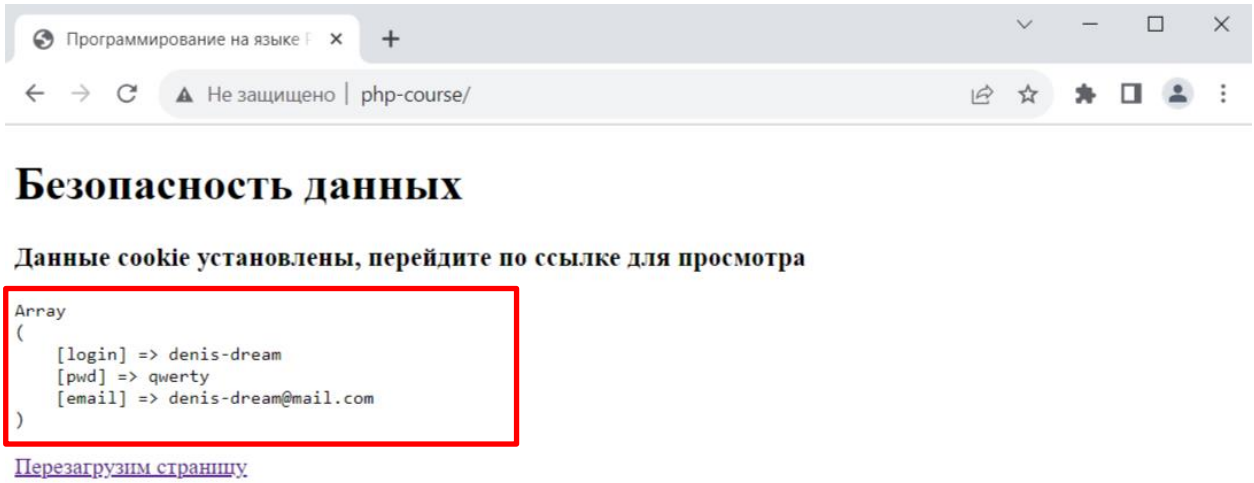
```php
<?php
    // файлы cookie должны быть установлены до вывода в браузер
    // устанавливаем cookies
    setcookie('login', 'denis-dream');
    setcookie('pwd', 'qwerty');
    // делать таким образом нет никакого смысла
    $_COOKIE['email'] = 'denis-dream@mail.com';
```

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

// вывод данных cookies

echo "<pre>";

print_r($_COOKIE);

echo "</pre>"
```

При первой загрузке файла **index.php** вывод массива $_**COOKIE** покажет следующий результат.



Перезагрузка страницы выведет полное содержимое массива $_**COOKIE**.

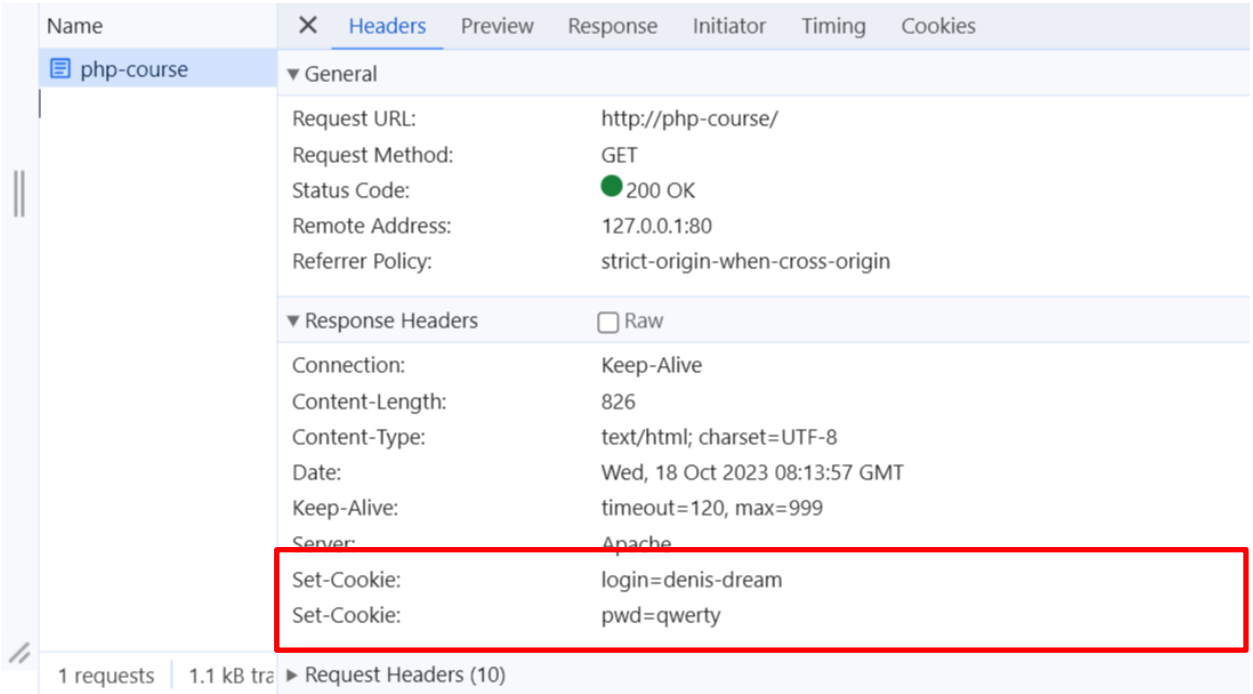


example_4. index.php

```
<?php

    // файлы cookie должны быть установлены до вывода в браузер
```

```php
    // устанавливаем cookies
    setcookie('login', 'denis-dream');
    setcookie('pwd', 'qwerty');
    // делать таким образом нет никакого смысла
    $_COOKIE['email'] = 'denis-dream@mail.com';
?>
<!-- ... -->
<?php
    // вывод данных cookies
    echo "<pre>";
    print_r($_COOKIE);
    echo "</pre>"
?>
```

Но **анализ процесса обмена данными** показывает, что поле **email** не участвует в обмене данными между клиентом и сервером. Каждый раз в сценарии мы просто берем и присваиваем массиву $_**COOKIE** новое значение в поле ключа **email.**



Если цель была отправить клиенту заголовки с дополнительным значением **cookie**, то необходимо использовать функцию **setcookie**().

Если цель – сохранить значение переменной **email**, зачем использовать массив $_**COOKIE**? Массив существует не для записи, а для **считывания** данных.

> ***Впрочем***. Если разработчик понимает, что он делает и к каким последствиям может привести программа – почему бы и нет.

Продолжим эксперименты, почистим кэш, запустим код и **проанализируем обмен данными в более сложном коде** демонстрационного примера **example_5**.

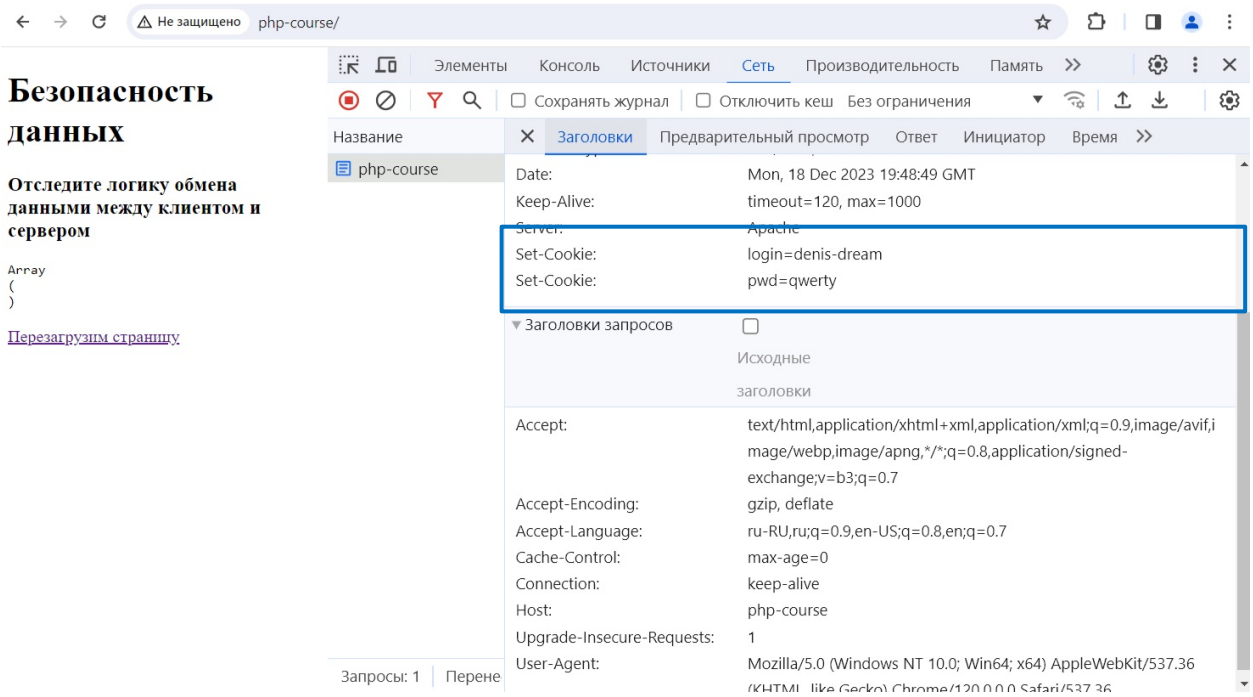
example_5. index.php

```
<?php
    // файлы cookie должны быть установлены до вывода в браузер
    //установим cookie login и pwd при первом посещении страницы
    if (empty ($_COOKIE['login']) || empty($_COOKIE['pwd'])) {
        setcookie('login', 'denis-dream');
        setcookie('pwd', 'qwerty');
    } else {
        // cookie login и pwd не пустые, проверяем cookie name и
        email
        if (
            empty($_COOKIE['name']) ||
            empty($_COOKIE['email'])
```

```
        ) {
            setcookie('login', $_COOKIE['login']);
            setcookie('pwd', $_COOKIE['pwd']);
            setcookie('name', 'Денис');
            setcookie('email', 'denis-dream@mail.com');
        }
        /*
        иначе все cookie есть, далее будем отправлять клиенту
        cookie
        из имеющихся значений
        */
        else {
            setcookie('login', $_COOKIE['login']);
            setcookie('pwd', $_COOKIE['pwd']);
            setcookie('name', $_COOKIE['name']);
            setcookie('email', $_COOKIE['email']);
        }
    }
?>
<!-- ... -->
<?php
    // вывод массива $_COOKIE
    echo "<pre>";
    print_r($_COOKIE);
    echo "</pre>"
?>
```

1. При **первом запросе** страницы **index.php** никаких файлов **cookie** на **клиенте** нет (надеюсь кэш почищен), и соответственно, на сервер ничего не отправляется. Но при ответе сервера, файлы **cookie** уже присутствуют:

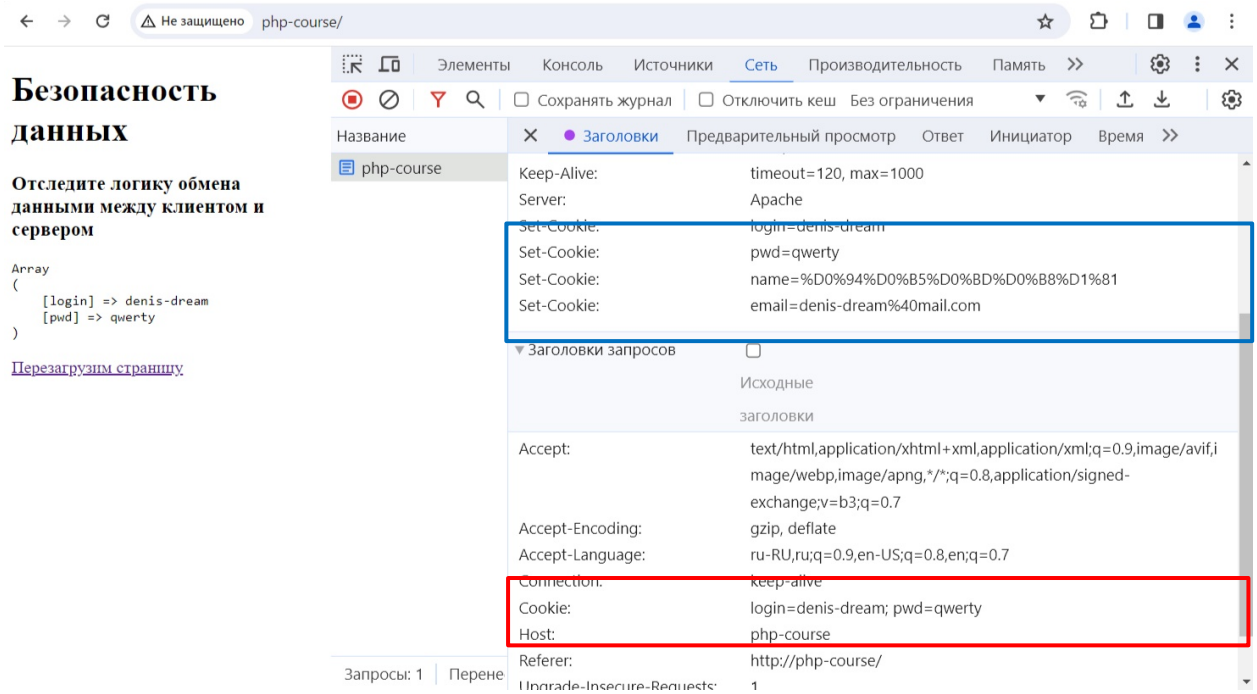
```
// установим cookie login и pwd при первом посещении страницы
if(empty($_COOKIE['login']) || empty($_COOKIE['pwd']) ) {
    setcookie('login', 'denis-dream');
    setcookie('pwd', 'qwerty');
} else {
    // ...
}
```



2. Перезагрузим страницу. Теперь запрос браузера содержит информацию о имеющихся на **клиенте cookie**. Ответ сервера содержит набор из уже имеющихся и вновь установленных файлов **cookie**.

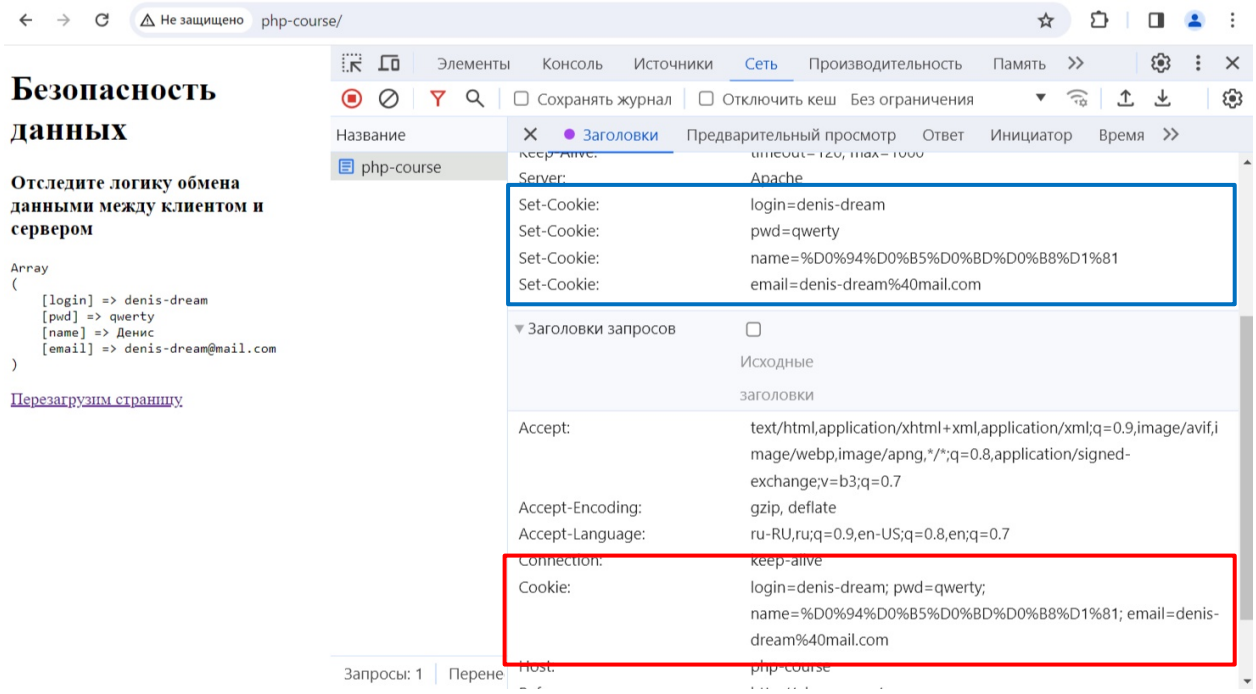
```
// установим cookie login и pwd при первом посещении страницы
if (empty ($_COOKIE['login']) || empty($_COOKIE['pwd'])) {
```

```
    setcookie('login', 'denis-dream');
    setcookie('pwd', 'qwerty');
} else {
    // cookie login и pwd не пустые, проверяем cookie name и
    email
    if (
        empty($_COOKIE['name']) ||
        empty($_COOKIE['email'])
    ) {
        setcookie('login', $_COOKIE['login']);
        setcookie('pwd', $_COOKIE['pwd']);
        setcookie('name', 'Денис');
        setcookie('email', 'denis-dream@mail.com');
    }
    else {
        // ...
    }
}
```



3. Теперь каждый новый запрос браузера к файлу **index.php** будет генерировать набор хранящихся на клиенте файлов **cookie**.

```php
setcookie('login', $_COOKIE['login']);
setcookie('pwd', $_COOKIE['pwd']);
setcookie('name', $_COOKIE['name']);
setcookie('email', $_COOKIE['email']);
```



В демонстрационном примере **example_6** рассмотрим применение **setcookie**() с применением следующих параметров:

- **expires** – время жизни, если 0, **cookie** будут действовать до закрытия браузера.
- **path** – путь к директории, из которой будут доступны **cookie**.

example_6. index.php

```
<?php
    // файлы cookie должны быть установлены до вывода в браузер
    // отправляем файлы cookie
    /*
    время жизни - до закрытия браузера
    актуальная директория - /cookie
    */
    setcookie('login', 'admin', 0,'/cookie');
    setcookie('pwd', 'superCoolPWD', 0,'/cookie');
?>
<!-- ... -->
<a href="cookie/">Здесь нас ждут!</a><p>
<a href="no-cookie">А здесь нам не рады :(</a>
```

Файлы-обработчики **cookie/index.php** и **no-cookie/index.php** абсолютно одинаковы.

example_6. cookie/index.php

```
<?php
    // если cookie login/pwd существуют
    if (
        isset($_COOKIE['login']) &&
```

```
        isset($_COOKIE['pwd'])
    )
    {
        echo "<h2>Здравствуйте, мы где-то уже встречались?</h2>";
        echo "Логин: <b>" . $_COOKIE['login'] . "</b><br>";
        echo "Пароль: <b>" . $_COOKIE['pwd'] . "</b><br>";
        echo "<img src='../images/welcome.jpg' width='300px'>" .
        "<hr>";
    } else {
        echo "<h2>Go Home!!!</h2>";
        echo "<img src='../images/go-home.jpg' width='300px'>" .
        "<hr>";
    }
?>
```

Удаление cookies

Если файлы **cookie** больше не нужны, их можно сделать неактуальными или удалить.

> ***Важно***. Сделать данные **неактуальными** и **удалить** данные массива $_**COOKIE** – очень разные понятия. Если понятен раздел "Алгоритм обмена данными cookie", то следующие примеры дополнят представление о механизме передачи данных **cookies**.

Деактуализация cookie через setcookie()

Чтобы параметр **cookie** сделать неактуальным, достаточно в **setcookie**() аргумент **expires** установить в какое-либо **прошедшее время**. Например, -1 час:

```
setcookie('test', '', time() - 3600);

// сделаем файлы cookie неактуальными

setcookie("login", time()-3600);

setcookie("pwd", time()-3600);
```

В демонстрационном примере **example_7** выполним цепочку действий по **созданию** и **удалению** файлов cookie.

example_7. index.php

```
<?php

    // отправка (установка) cookie

    setcookie('login', 'admin');

    setcookie('pwd', 'superCoolPWD');

    echo "<pre>";

    print_r($_COOKIE);

    echo "</pre>";

?>

<!-- -->

<a href="/target.php">Просмотр файлов cookie</a><p>
```

example_7. target.php

```
<?php

    // если cookie существуют

    if (

        isset($_COOKIE['login']) &&
```

```
        isset($_COOKIE['pwd'])
    )
    {
        echo "<h3>Получены следующие данные:</h3>";
        echo "Логин: <b>" . $_COOKIE['login'] . "</b><br>";
        echo "Пароль: <b>" . $_COOKIE['pwd'] . "</b><br>";;
        echo "<p>Текущее время: <b>" . time() . "</b>";
    }
?>
<p>Мы удалили cookie:
<a href="/target.php">Перегрузите страницу для проверки</a></p>
<p>Для установки cookie вернитесь на главную:
<a href="/">Установить cookie</a></p>
```

Удаление данных cookie через unset()

Можно физически удалить данные по ключу из массива $_**COOKIE**.

```
// деактулизируем и удаляем данные cookie
setcookie('login', '', -1);
unset($_COOKIE['login']);
setcookie('pwd', '', -1);
unset($_COOKIE['pwd']);
```

example_8. index.php

```
<?php
    // установка cookie login
    setcookie('login', 'admin');
    // установка cookie pwd
```

```
    setcookie('pwd', 'superCoolPWD');
?>
<!-- ... -->
<?php
    // cookies текущего сценария
    echo "<pre>";
    print_r($_COOKIE);
    echo "</pre>";
?>
<a href="/target.php">Просмотр файлов cookie</a><p>
```

example_8. target.php

```
<?php
    // если cookie login/pwd существуют
    if (isset($_COOKIE['login']) && isset($_COOKIE['pwd'])) {
        // деактулизируем и удаляем данные cookie
        setcookie('login', '', -1);
        unset($_COOKIE['login']);
        setcookie('pwd', '', -1);
        unset($_COOKIE['pwd']);
    }
?>
<!-- ... -->
<?php
    // если cookie login/pwd существуют
    if (
        isset($_COOKIE['login']) &&
```

```
        isset($_COOKIE['pwd'])
    )
    {
        echo "<h3>Получены следующие данные:</h3>";
        echo "Логин: <b>" . $_COOKIE['login'] . "</b><br>";
        echo "Пароль: <b>" . $_COOKIE['pwd'] . "</b>";
    } else {
        echo '<h3>Данные cookie безвозвратно удалены</h3>';
    }
?>
<a href="/">На главную</a></p>
```

Cookie в массивах

Чтобы рассмотреть более продвинутый способ передачи и хранения данных, разберем работу **cookies** с массивами данных.

example_9. index.php

```
<?php
    // определяем массив
    $team = array (
        'name' => "The Rolling Stones",
        'alias' => "the-rolling-stones",
        'country' => "Великобритания",
        'content' => "The Rolling Stones  — британская рок-
        группа, образовавшаяся 12 июля 1962 года и многие годы
        соперничавшая по популярности с The Beatles. The Rolling
        Stones, ставшие важной частью Британского вторжения,
        считаются одной из самых влиятельных и успешных групп в
```

```
        истории рока… Музыкальный стиль The Rolling Stones,
        формировавшийся под влиянием Роберта Джонсона, Бадди
        Холли, Элвиса Пресли, Чака Берри, Бо Диддли и Мадди
        Уотерса, с течением времени обретал индивидуальные черты;
        авторский дуэт Джаггер-Ричардс получил в конечном итоге
        всемирное признание."
    );
    // сохраняем данные в массив cookies
    setcookie('team[0]', $team["name"]);
    setcookie('team[1]', $team["alias"]);
    setcookie('team[2]', $team["country"]);
    setcookie('team[3]', $team["content"]);
?>
```

Переходим в файл **form.php** для вывода данных **cookies** в элементы формы для условного редактирования.

example_9. form.php

```
<?php
    // echo "<pre>";
    // print_r($_COOKIE);
    // echo "</pre>";
    echo <<<HERE
        <form action="" method="post">
            Название: <input type="text" name="name"
            value="{$_COOKIE['team'][0]}"><p>
            Алиас: <input type="text" name="alias"
            value="{$_COOKIE['team'][1]}"><p>
            Страна: <input type="text" name="country"
            value="{$_COOKIE['team'][2]}"><p>
```

```
            <textarea
            name="description">{$_COOKIE['team'][3]}</textarea><p
            >
            <input type="submit">
        </form>
    HERE;
?>
```

Естественно, все действия с **массивами** удобнее выполнять используя **циклы** ;)

example_10. index.php

```
<?php
    // определяем массив
    $team = array (
        'name' => "The Rolling Stones",
        'alias' => "the-rolling-stones",
        'country' => "Великобритания",
        'content' => "The Rolling Stones  — британская рок-
        группа, образовавшаяся 12 июля 1962 года и многие годы
        соперничавшая по популярности с The Beatles..."
    );
    // сохраняем данные в массив cookies
    foreach ($team as $key => $value) {
        setcookie("team[$key]", $value, 0, '/');
    }
?>
```

Переходим в файл **form.php**.

example_10. form.php

```php
<?php

    // echo "<pre>";

    // print_r($_COOKIE);

    // echo "</pre>";

    echo <<<HERE

        <form action="" method="post">

            Название: <input type="text" name="name"
            value="{$_COOKIE['team']['name']}"><p>

            Алиас: <input type="text" name="alias"
            value="{$_COOKIE['team']['alias']}"><p>

            Страна: <input type="text" name="country"
            value="{$_COOKIE['team']['country']}"><p>

            <textarea
            name="description">{$_COOKIE['team']['content']}</tex
            tarea><p>

            <input type="submit">

        </form>

    HERE;

?>
```

Используем JSON в cookies

В демо примере **example_11** поэкспериментируем с использованием в файлах **cookie** формата **JSON**.

example_11. index.php

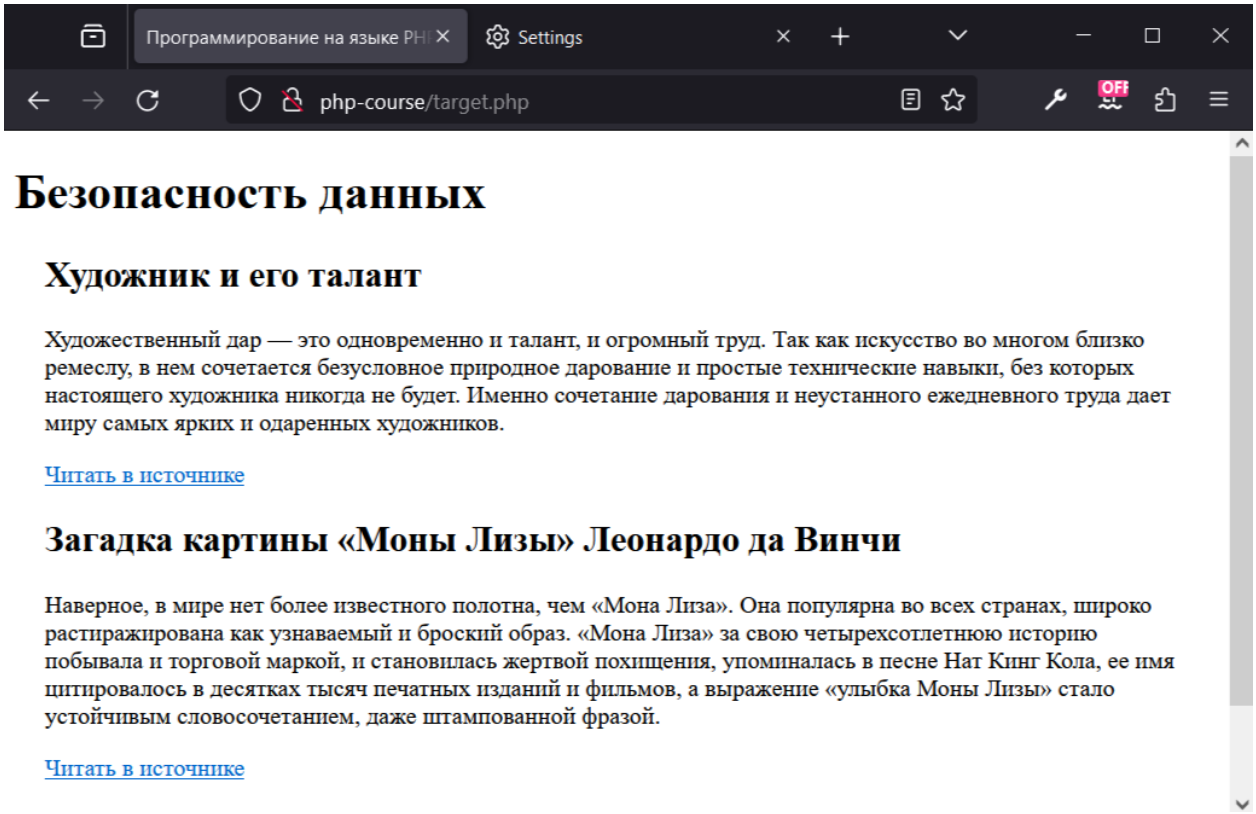
```php
<?php

    $articles = [
```

```
    [
        "title" => "Художник и его талант",
        "teaser" => "Художественный дар — это одновременно и талант, и огромный труд...",
        "link" => "https://muzei-mira.com/article/2148-hudozhnik-i-ego-talant.html"
    ],
    [
        "title" => "Загадка картины «Моны Лизы» Леонардо да Винчи",
        "teaser" => "Наверное, в мире нет более известного полотна, чем «Мона Лиза». Она популярна во всех странах, широко растиражирована как узнаваемый и броский образ...",
        "link" => "https://muzei-mira.com/article/2147-zagadka-kartiny-mony-lizy-leonardo-da-vinchi.html"
    ]
    ];
    // перебираем массив для установки файлов cookie
    foreach ($articles as $key => $article) {
        $json = json_encode($article, JSON_UNESCAPED_UNICODE);
        setcookie("articles[$key]", $json);
    }
?>
```

example_11. target.php

```
<?php
    // перебираем массив cookies
```

```php
    foreach ($_COOKIE['articles'] as $item) {

        // декодируем JSON

        $article = json_decode($item, 1);

        // выводим содержимое в браузер

        echo <<<HERE

            <div style='margin:20px'>

            <h2>{$article['title']}</h2>

            <p>{$article['teaser']}</p>

            <a href='{$article['link']}' target='_blank'>Читать в
            источнике</a>

            </div>

        HERE;

    }
?>
```



Междоменные cookies

Из соображений безопасности читать/писать **cookie** из другого домена нельзя. Возможность использовать **междоменные cookie** в современных браузерах обычно отключены. Браузеры принимают файлы **cookie** только с сайта, адрес которого указан в **адресной строке**.

Для следующего демонстрационного примера я размести файлы сценариев следующим образом:

- домен **php-course**:
  - **index.php**
  - **target.php**
- домен **site**:
  - **target-cross.php**

example_12. index.php

```
<?php

    setcookie('name', 'Денис');

    setcookie('login', 'denis-dream');

    setcookie('email', 'denis-dream@mail.com');

    setcookie('pwd', 'qwerty');

?>

<!-- ... -->

<p><a href='target.php'>Просмотр данных cookie</a>

<p><a href='http://site/target-cross.php'>Междоменный переход</a>
```

Файлы **target.php** и **target-cross.php** одинаковы.

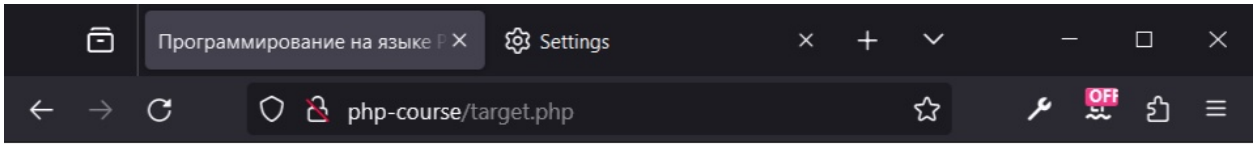
example_12. target.php

```
<?php
```

```
    echo "<pre>";
    print_r($_COOKIE);
    echo "</pre>"
?>
```

> ***Важно***. Сказанное не означает, что между доменами нельзя передать информацию. Это означает, что для передачи информации нужно предпринимать дополнительные (иногда непростые) усилия.

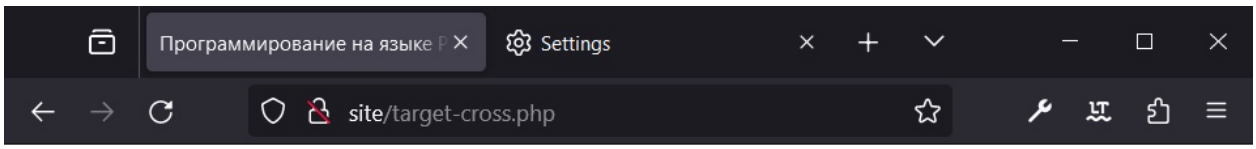
Домен **php-course**.



Безопасность данных

```
Array
(
    [name] => Денис
    [login] => denis-dream
    [email] => denis-dream@mail.com
    [pwd] => qwerty
)
```

Домен **site**.



Безопасность данных

```
Array
(
)
```