Cookies & Session

Cookies

PHP прозрачно поддерживает HTTP cookies. Cookies - это механизм хранения данных браузером удаленной машины для отслеживания или идентификации возвращающихся посетителей. Вы можете установить cookies при помощи функций setcookie() или setrawcookie(). Cookies являются частью HTTP-заголовка, поэтому setcookie() должна вызываться до любого вывода данных в броузер.

Мы можем использовать Cookie для хранения небольшой по объему информации у клиента (посетителя) сайта, например: настройки сайта (цвет фона страниц, язык, оформление таблиц и.т.д.), а также другой информации. Файлы Cookies представляют собой обыкновенные текстовые файлы, которые хранятся на диске у посетителей сайтов. Файлы Cookies и содержат ту информацию, которая была в них записана сервером.

Для установки Cookies используется функция setcookie(). Для этой функции можно указать шесть параметров, один из которых является обязательным:

- name - задает имя (строк), закрепленное за Cookie;
- value - определяет значение переменной (строка);
- expire - время "жизни" переменной (целое число). Если данный параметр не указать, то Cookie будут "жить" до конца сессии, то есть до закрытия браузера. Если время указано, то, когда оно наступит, Cookie самоуничтожится;
- path - путь к Cookie (строка);
- domain - домен (строка). В качестве значения устанавливается имя хоста, с которого Cookie был установлен;
- secure - передача Cookie через защищенное HTTPS-соединение.

Обычно используются только три первые параметра.

Пример установки Cookies:

```
setcookie("Test","Value");   // Устанавливаем Cookie до конца сессии
setcookie("My_Cookie","Value",time()+3600);   // Устанавливаем Cookie на один час после установки
```

При использовании Cookies необходимо иметь в виду, что Cookies должны устанавливаться до первого вывода информации в браузер (например, оперетором echo или выводом какой-либо функции). Поэтому желательно устанавливать Cookies в самом начале скрипта. Cookies устанавливаются с помощью определенного заголовка сервера, а если скрипт выводит что-либо, то это означает, что начинается тело документа. В результате Cookies не будут установлены и может быть выведено предупреждение.

Функция setcookie() возвращает TRUE в случае успешной установки Cookie. В случае, если Cookie установить не удается setcookie() возвратит FALSE и возможно, предупреждение (зависит от настроек PHP).

Получить доступ к Cookies и их значениям достаточно просто. Они хранятся в суперглобальных массивах и $_COOKIE и $HTTP_COOKIE_VARS.

Доступ к значениям осуществляется по имени установленных Cookies, например:

```
echo $_COOKIE['my_cookie'];
// Выводит значения установленной Cookie 'my_cookie'
```

А вот пример, как построить счетчик числа загрузок страницы с помощью Cookies:

```
if (isset($_COOKIE['count'])) {
	$count = $_COOKIE['count'] + 1;
} else {
	$count = 0;
}

setcookie("count", $count);

echo "<p>Вы посещали эту страницу <b>".@$_COOKIE['count']."</b> раз</p>";
```

Иногда возникает необходимость удаления Cookies. Сделать это несложно, необходимо лишь вновь установить Cookie с идентичным именем и пустым параметром. Например:

```
setcookie("Test", "");
```

Мы можем установить массив Cookies, используя квадратные скобки в именах Cookies [], а затем прочитать массив Cookies и значения этого массива:

```
setcookie("cookie[1]", "Первый");
setcookie("cookie[2]", "Второй");
setcookie("cookie[3]", "Третий");

if (isset($_COOKIE['cookie'])) {
    foreach ($_COOKIE['cookie'] as $name => $value) {
        echo "$name : $value <br>";
    }
}

// Result:
1 : Первый
2 : Второй
3 : Третий
```

Session

Поддержка сессий в PHP заключается в способе сохранения некоторых данных между несколькими последовательными доступами веб-сайта. Каждому посетителю сайта присваивается уникальный идентификатор, называемый идентификатором сессии (session id). Он хранится либо в cookie на стороне пользователя, либо передается через URL.

Поддержка сессий позволяет сохранять данные между запросами в суперглобальном массиве $_SESSION. В тот момент, когда посетитель получает доступ к сайту, PHP проверяет (автоматически, если session.auto_start установлено в 1, или по запросу явным образом через вызов session_start()), если определенный идентификатор сессии послан вместе с запросом. Если это так, восстанавливается сохраненное ранее окружение.

Сессии являются простым способом хранения информации для отдельных пользователей с уникальным идентификатором сессии. Это может использоваться для сохранения состояния между запросами страниц. Идентификаторы сессий обычно отправляются браузеру через сессионный cookie и используются для получения имеющихся данных сессии. Отсутствие идентификатора сессии или сессионного cookie сообщает PHP о том, что необходимо создать новую сессию и сгенерировать новый идентификатор сессии.

```
session_start();

if (isset($_SESSION['count'])) {
    $_SESSION['count']++;
} else {
    $_SESSION['count'] = 0;
}

echo $_SESSION['count'];

if ($_SESSION['count'] > 20) {
    unset($_SESSION['count']);
}
```

Cookies — файлы, хранящиеся на клиентской стороне.
Session — файлы, хранящиеся на сервере.

Как PHP связывает сессию с предыдущего запроса с текущей? - куки. Когда пользователю присваивается сессия, автоматически устанавливается http-only (чтобы люди не могли из js увести нашу сессию) кука, в которую записан идентификатор сессии.

Принципиальная разница между cookie и сессиями состоит в том, что cookie полностью хранятся в браузере пользователя (то есть на компьютере клиента), а при сессиях в cookie хранится только идентификатор сессии, а вся информация лежит на сервере в специальном уникальном файле. Именно из этого базового различия вытекают все остальные.

Если Вы решили хранить логин и пароль в cookie, то должны понимать, что cookie можно украсть. То есть минус cookie - низкая безопасность. Второй минус cookie - это то, что они живут ровно столько, сколько хранит их браузер. Чем это может обернуться? Простой пример.

У Вас серьёзный сайт, на котором у людей лежат крупные суммы денег. Ваш посетитель пришёл в какое-нибудь Интернет-кафе, авторизовался, но выйти забыл. Дальше приходит злоумышленник заходит на его аккаунт и забирает cookie. Дальше процесс очевиден. А если бы использовались сессии, то по умолчанию через 15 минут бездействия пользователя происходил бы автоматический выход (файл с данными сессиями бы стирался). Более того, ввиду того, что каждая сессия имеет уникальный идентификатор, то смысл воровства идентификатора сессии (а больше ничего взять не получится) уже отсутствует. Когда злоумшыленник придёт домой, этот идентификатор ему уже не поможет.

Но именно по причине временного хранения сессии, появляется большой минус сессий - неудобно. Допустим, человек авторизовался, чтобы следить за какими-нибудь данными (например, за входящей почтой), но делает это он примерно раз в полчаса. И каждый раз он вынужден авторизовываться, что, разумеется, неудобно.

Именно по этой причине cookie так популярны при работе с механизмом авторизации.