GeekBrains

Основы программирования

Разработка веб-сайта с играми

Разработка интерактивных игр на сайте.

Оглавление

HTML + CSS = Сайт

Мы освоили основы JavaScript, но все наши программы выглядели не очень красиво. Чтобы сделать веб-страницу более насыщенной, познакомимся с языком разметки HTML и языком описания внешнего вида документа — CSS.

HTML

Все программы, которые мы писали прежде, были не совсем правильными веб-страницами. Но браузеры так устроены, что если даже страница написана неверно, он старается отобразить информацию так, как считает нужным. Посмотрим, как мы можем указать браузеру, что и как выводить.

Наберите в текстовом редакторе текст, как в примере. Сохраните его под названием **index.html** и откройте в браузере, щелкнув на файле **index.html** два раза.

```
<!DOCTYPE html>
<html>
    <head></head>
    <body>
        Тело веб-страницы
    </body>
</html>
```

Возможно, у вас выведется текст или непонятные символы. Узнаем, от чего это зависит.

Шаблон веб-страницы состоит из тегов — ключевых слов, заключенных в угловые скобки. Теги бывают *парные* и *одинарные*.

<!DOCTYPE html> — это указание браузеру, что тип документа, с которым ему придется работать, **html**. Он одинарный и не требует закрывающего тега.

<html> </html> — это парные теги, внутри которых лежит веб-страница

Она состоит из двух частей: заголовка и тела.

Заголовок помещается между тегами **<head>** и **</head>**. Здесь находится служебная информация, которая влияет на отображение веб-страницы.

Тело, то есть сама веб-страница, располагается между тегами **<body>** и **</body>**. Большинство текста по описанию веб-страницы помещается сюда.

Добавим в заголовок веб-страницы:

- указание, в какой кодировке (тег **<meta charset=”utf-8”>**) вы создаете веб-страницу;
- заголовок, который будет отображаться на вкладке браузера;
- и произвольный текст.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Иванов Иван</title>
</head>

<body>
    <strong>Папа у Васи силен в
математике,</strong><br>
    <em>Учится папа за Васю весь год,</em><br>
    <i>Где это видано, где это слыхано,</i><br>
    <b>Папа решает, а Вася сдает?</b>
</body>
</html>
```

Папа у Васи силен в математике,
Учится папа за Васю весь год,
Где это видано, где это слыхано,
Папа решает, а Вася сдает?

Теги **strong** и **b** делают шрифт жирным, а **em** и **i** — наклонным. Отличие тега **strong** от **b** в том, что он не просто делает текст жирным, но и придает ему больший вес для поисковых систем.

Поначалу количество тегов в HTML может наводить тоску, но они легко запоминаются, а специальные редакторы, такие как **Sublime Text** или **Brackets**, имеют встроенную контекстную подсказку по тегам.

CSS

Добавим простой CSS прямо в заголовок страницы. Для этого между тегами **head** вставим парный тег **style**, а внутри него — код CSS.

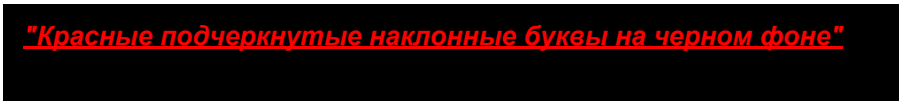
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Иванов Иван</title>
        <style>
            strong {
                color: red;
            }
            .my_style {
                font-size: 30px;
                color: blue;
            }
        </style>
    </head>

    <body>
        <strong>Папа у Васи силен в математике,</strong><br>
        <em class="my_style">Учится папа за Васю весь год,</em><br>
        <i>Где это видано, где это слыхано,</i><br>
        <b>Папа решает, а Вася сдает?</b>
    </body>
</html>
```

CSS задает стиль отображения информации. Здесь мы продемонстрировали два способа задать стиль. Первый, **strong**, мы описали без точки в начале, что означает, что этот стиль применяется для всех тегов **strong** на этой странице. Второй, **.my_style** (с точкой в начале) — класс стиля, который потом можно назначать на веб-странице. Так мы и сделали, назначив этот стиль тексту, заключенному между тегами **em**. Сами стили простые: в первом случаем мы указали, что нужно выводить текст в красном цвете, а во втором — что требуется задать размер шрифта в 30 пикселей и выводить информацию синим цветом.

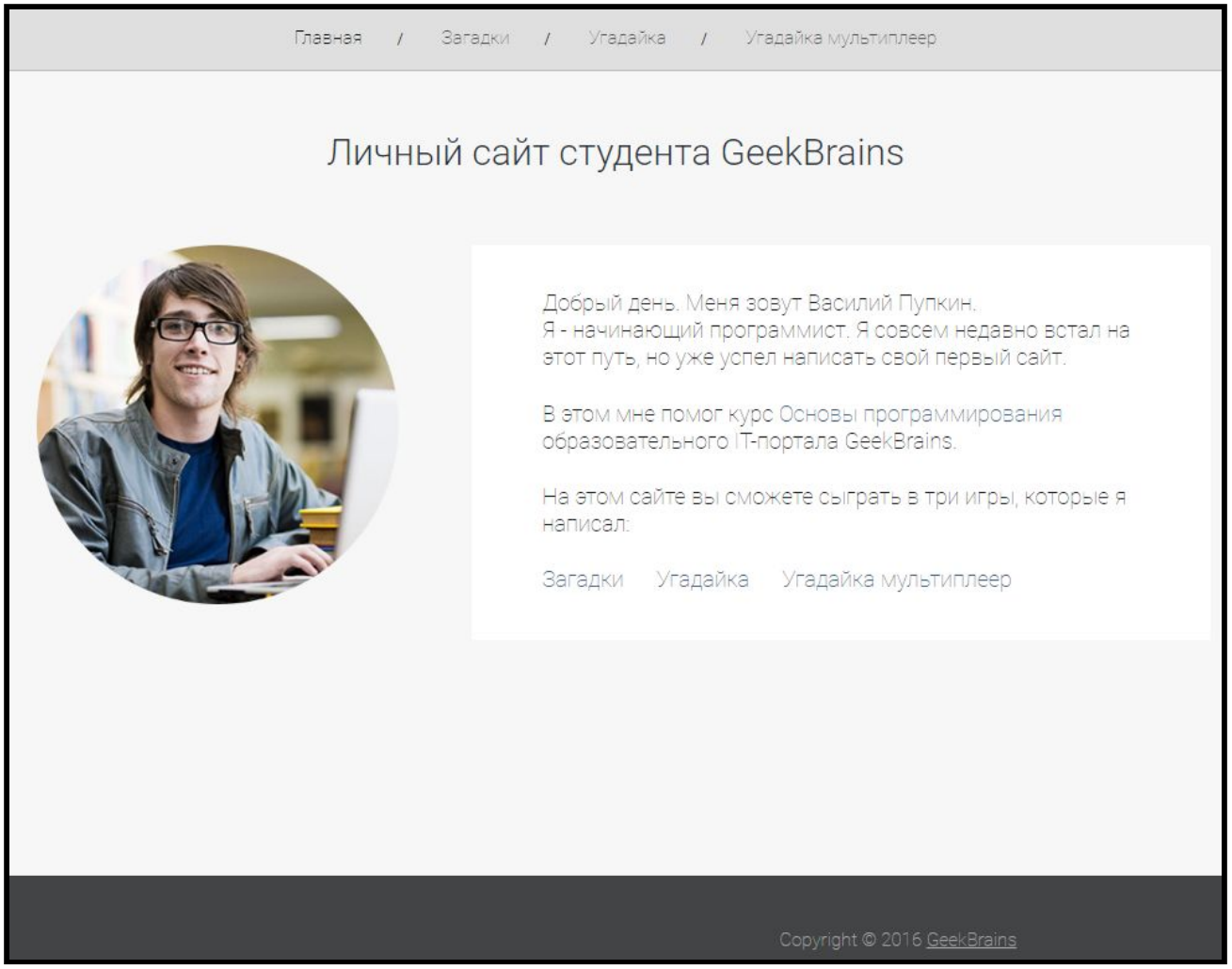
Личный сайт

Для начинающих изучать HTML один из самых сложных вопросов — «Какой сайт создать?». Зачастую люди заканчивают обучение, так и не ответив на него. В лучшем случае пишется страничка в стиле:



Не будем мучиться и создадим личный сайт, который будет вашим стартом.

Что примерно должно получиться:



Этот сайт состоит из нескольких частей: верхней, центральной и нижней. Центральная разделена еще на две. Интересно, что начинающие разработчики часто не могут построить сайт просто потому, что не очень понимают, из каких частей он будет состоять.

Центральная часть

Подготовка и вставка картинки

Сайт будет состоять из нескольких файлов, поэтому сначала создадим папку **mysite**, в которой они будут храниться. Для простоты можно разместить ее на рабочем столе. Далее создайте в ней папку **img**, в которой будут храниться картинки. Поместите туда свою фотографию или другое изображение, которое будет выводиться на месте фотографии. Назовите его **photo.png**. Если умеете обрабатывать фотографии, придайте картинке размер 390 x 390. Если нет — ничего страшного, при вставке мы заставим браузер задать нужные параметры.

Чтобы вставить картинку, используется одинарный тег **img**:

```
<img src="img/photo.png" alt="фотография">
```

Атрибут **src** указывает расположение картинки. Атрибут **alt** нужен для того, чтобы показать, какой текст будет выводиться, если у пользователя отключен вывод картинок или изображение будет недоступно.

Должно получиться:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <img src="img/photo.png" alt="фотография">
    </body>
</html>
```

Заголовок

Добавим заголовок. Для этого используется парный тег **<h1>**. Добавим перед картинкой текст:

```
<h1>Личный сайт студента GeekBrains</h1>
```

Получится:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <h1>Личный сайт студента GeekBrains</h1>
        <img src="img/photo.png" alt="фотография">
    </body>
</html>
```

Личный сайт студента GeekBrains

Информация о себе

Добавим информацию о себе. Для этого используем парный тег **<p>** и одинарный **
.** Под тегом **img** вставляем следующий текст:

```
<p>
    Добрый день. Меня зовут Василий Пупкин.
    <br>
    Я — начинающий программист. Я совсем недавно встал на этот путь, но уже
успел написать свой первый сайт.
</p>
<p>
    В этом мне помог курс Основы программирования образовательного IT-портала
GeekBrains.
</p>
<p>
    На этом сайте вы сможете сыграть в три игры, которые я написал:
</p>
```

Личный сайт студента GeekBrains

Добрый день. Меня зовут Василий Пупкин.
Я - начинающий программист. Я совсем недавно встал на этот путь, но уже успел написать свой первый сайт.

В этом мне помог курс Основы программирования образовательного IT-портала GeekBrains.

На этом сайте вы сможете сыграть в три игры, которые я написал:

Ссылки

Парный тег **<a>** указывает браузеру, что содержимое элемента (это может быть и текст, и картинка) является ссылкой. При щелчке по ней (то есть по содержимому) браузер попытается перейти по адресу, указанном в атрибуте **href**.

```
<p>
    В этом мне помог курс
    <a href="https://geekbrains.ru/courses/2">
        Основы программирования
    </a>
    образовательного IT-портала GeekBrains.
</p>
```

Тег **<a>** поддерживает атрибут **target**, который показывает, в какой вкладке открыть страницу, указанную в ссылке. Например, **target="_blank"** — указание загрузить страницу в новой вкладке.

```
<p>
    В этом мне помог курс
    <a href="https://geekbrains.ru/courses/2" target="_blank">
        Основы программирования
    </a>
    образовательного IT-портала GeekBrains.
</p>
```

Верхняя часть

Добавим перед тегом **<h1>**, сразу после **<body>**, следующий текст:

```
<a href="#">Главная</a>
<span>/</span>

<a href="puzzles.html">Загадки</a>
<span>/</span>

<a href="guess.html">Угадайка</a>
<span>/</span>

<a href="guess-2.html">Угадайка мультиплеер</a>
```

Атрибут **href="#"** — указание, что при щелчке по этой ссылке никуда переходить не нужно.

Парный тег **** — наверное, самый трудный для понимания. Он говорит, что внутри заключен фрагмент текста. На самом деле в HTML есть еще теги, которые позволяют задать логику страницы, и **span** — один из них. Он не влияет непосредственно на сам вывод, но позволяет в дальнейшем элементам, заключенным в этот тег, придать стиль посредством CSS. С помощью **** и CSS мы добавим промежутки между пунктами меню.

Нижняя часть

В нижнюю часть добавим ссылки на будущие веб-страницы и информацию об авторском праве.

```
<a href="puzzles.html">Загадки</a>
<a href="guess.html">Угадайка</a>
<a href="guess-2.html">Угадайка мультиплеер</a>
Copyright © 2016 <a href="https://geekbrains.ru/"
target="_blank">GeekBrains</a>
```

Итоговый код должен получиться примерно таким:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <a href="#">Главная</a>
        <span>/</span>

        <a href="puzzles.html">Загадки</a>
        <span>/</span>

        <a href="guess.html">Угадайка</a>
        <span>/</span>

        <a href="guess-2.html">Угадайка мультиплеер</a>

        <h1>Личный сайт студента GeekBrains</h1>
        <img src="img/photo.png" alt="фотография">
        <p>
            Добрый день. Меня зовут Василий Пупкин.
            <br>
            Я - начинающий программист. Я совсем недавно встал на этот путь, но
уже успел написать свой первый сайт.
        </p>
        <p>
            В этом мне помог курс
            <a href="https://geekbrains.ru/courses/2" target="_blank">
                Основы программирования
            </a>
            образовательного IT-портала GeekBrains.
        </p>
        <p>
            На этом сайте вы сможете сыграть в три игры, которые я написал:
        </p>

        <a href="puzzles.html">Загадки</a>
        <a href="guess.html">Угадайка</a>
        <a href="guess-2.html">Угадайка мультиплеер</a>
        Copyright © 2016 <a href="https://geekbrains.ru/"
target="_blank">GeekBrains</a>

    </body>
</html>
```

Главная/ Загадки/ Угадайка/ Угадайка мультиплеер

Личный сайт студента GeekBrains

Добрый день. Меня зовут Василий Пупкин.
Я - начинающий программист. Я совсем недавно встал на этот путь, но уже успел написать свой первый сайт.

В этом мне помог курс Основы программирования образовательного IT-портала GeekBrains.

На этом сайте вы сможете сыграть в три игры, которые я написал:

Загадки Угадайка Угадайка мультиплеер Copyright © 2016 GeekBrains

DIV — разделяй и властвуй

Пока сайт выглядит невзрачно. Чтобы он стал более красивым, нужно задать стили оформления, но прежде — продумать и реализовать логику сайта с помощью тегов ** ** и атрибутов **id**.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Личный сайт студента GeekBrains</title>
</head>
<body>
        <div id="header">
            <a href="#">Главная</a>
            <span>/</span>

            <a href="#">Загадки</a>
            <span>/</span>

            <a href="#">Угадайка</a>
            <span>/</span>

            <a href="#">Угадайка мультиплеер</a>
        </div>

        <div id="content">
            <h1>Личный сайт студента GeekBrains</h1>
            <div id="center">
                <img src="img/photo.png" alt="фотография">

                <div id="box_text">
                    <p>
                        Добрый день. Меня зовут <b>Василий Пупкин</b>.
                        Я - <i>начинающий программист</i>. Я совсем недавно
встал на этот путь,
                        но уже успел написать свой первый сайт.
                    </p>

                    <p>
                        В этом мне помог курс <a
href="https://geekbrains.ru/courses/2">Основы программирования</a>
                    </p>

                    <p>
                        На этом сайте вы сможете сыграть в три игры, которые я
написал:
                    </p>
                    <a href="#">Главная</a>
                    <a href="#">Загадки</a>
                    <a href="#">Угадайка</a>
                    <a href="#">Угадайка мультиплеер</a>
                    <br><br>

                </div>
            </div>
        </div>

    <div id="footer">
```

```
        Copyright © 2016 <a href="https://geekbrains.ru/">GeekBrains</a>
    </div>
</body>
</html>
```

Сам тег **div**, как и ****, не влияет на вывод информации на экран. Но он позволяет выделить на веб-странице разделы, на которые можно будет потом влиять с помощью CSS-кода. Атрибут ID дает названия разделам, чтобы по ним можно было обратиться к нужному и изменить его стиль оформления.

Подключим CSS

Добавьте в **<head>** тег загрузки стиля:

```
<link rel="stylesheet" href="style.css">
```

Должно получиться примерно следующее:

```
<head>
    <meta charset="utf-8">
    <title>Личный сайт студента GeekBrains</title>
    <link rel="stylesheet" href="style.css">
</head>
```

Теперь перейдем к созданию файла, который будет влиять на стиль отображения информации на экране.

Подробнее о CSS

Трудность этой части урока — в большом количестве новой информации. Но на первых порах не обязательно понимать все, что увидите. CSS не так сложен, как может показаться на первый взгляд. Для начала поймите, что он будет влиять на отображение страницы.

CSS (Cascading Style Sheets) — каскадные таблицы стилей. Первое время создатели сайтов использовали теги и их атрибуты, чтобы указать браузеру, как выводить информацию на экран. Например, только тег **<h1>** позволял создать заголовок, а **<font>** — задать размер шрифта. CSS подменяет такую логику. Теперь с помощью тега **<h1>** указывается браузеру, что текст внутри будет заголовком. А как он будет выглядеть, мы можем задать с помощью таблицы стилей. Сейчас основное предназначение тегов — создать логику сайта, а уже CSS управляет тем, как этот текст будет выглядеть в браузере.

Если же для каких-то тегов мы не задали стиль, то браузер использует свою внутреннюю таблицу стилей.

Структура CSS

CSS — это уже не HTML, хотя и используются они совместно. При описании CSS чаще всего применяют эту структуру:

```
Название_стиля
{
    описание1;
    описание2;
...
}
```

Название_стиля — это *селектор*, а описание — *свойство.*

body и #header

Перед названием стиля могут стоять знаки **.** или **#**, которые указывают, к чему надо применить этот стиль.

Например, для тегов можно задать стиль таким образом:

```
body {
    background-color: #555;
    font-family: sans-serif;
}
```

Это указание, что для тега **body** сделать фон цветом **#555**, а шрифт — **sans-serif**.

Таким образом можно указать стиль для тегов, у которых атрибут **id=”header”**:

```
#header {
    width: 1300px;
    margin: 0 auto;
    height: 50px;
    text-align: center;
    padding-top: 18px;
}
```

*. padding и margin

* — звездочка — это указание применить стиль для всех тегов:

```
* {
    margin: 0;
    padding: 0;
}
```

padding и **margin** — это отступы. Первое создает отступы вокруг содержимого, второе расширяет поле до границы элемента.

Стилизация вложенных тегов

```
#header a {
    color: #4d4d4d;
    font-size: 20px;
    font-weight: 100;
    text-align: left;
    text-decoration: none;
    height: 30px;
    line-height: 30px;
    display: inline-block;
}
```

Здесь мы как бы говорим: «Эй! Все теги **<a>**, которые имеют **id=”header”**, будьте со своим стилем».

@import

Позволяет импортировать содержимое CSS-файла в текущую стилевую таблицу. В нашем примере с помощью этой команды мы подключим набор шрифтов.

Магия CSS

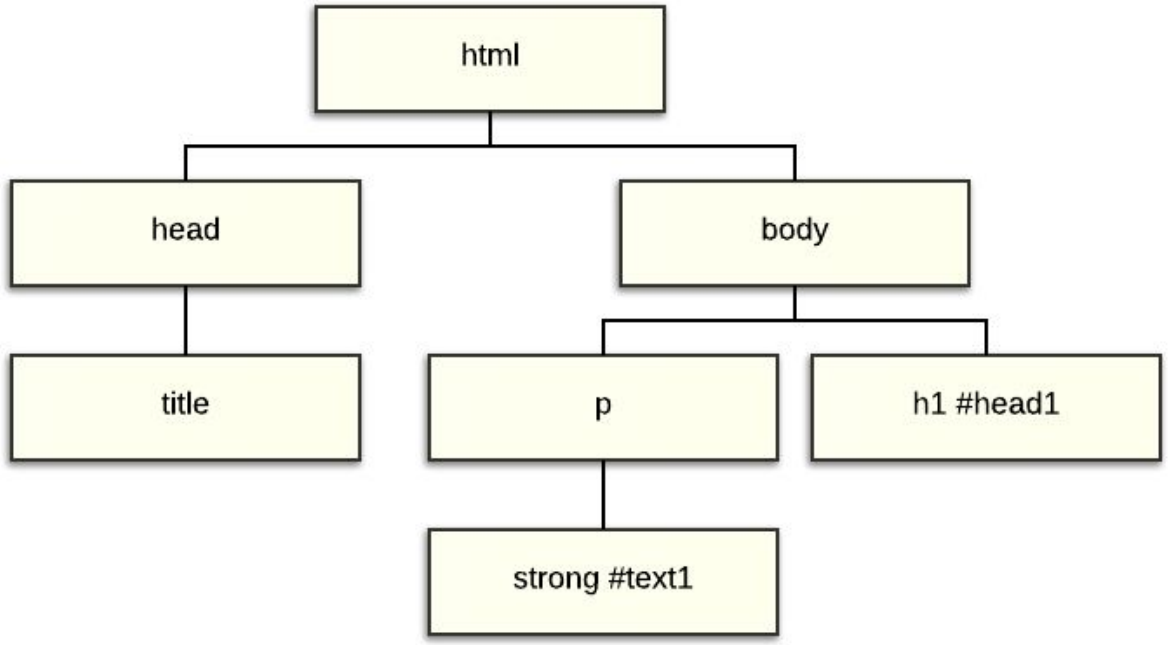
На этом теоретическое введение закончим и рассмотрим CSS-файл, который нам предоставил преподаватель курса HTML & CSS. Скопируйте текст с этой страницы и сохраните его в файл с именем **style.css** в папку **site**. После этого обновите в браузере страницу с html-файлом, и она магическим образом преобразится.

DOM, который построим мы

Рассмотрим небольшой html-документ:

```
<!DOCTYPE HTML>
<html>
<head>
    <meta charset="utf-8">
    <title>Пример документа</title>
</head>
<body>
    <h1 id="head1">Заголовок</h1>
    <p>
        Параграф с <strong id="text1">очень важным </strong>текстом
    </p>
</body>
</html>
```

Когда браузер считывает веб-страницу, в памяти строится схема. Для нашей страницы схема будет выглядеть так:



(Если у элемента есть идентификатор, то он начинается с решетки.)

Эта структура называется DOM (Document Object Model — объектная модель документа). Мы можем обращаться к этим элементам с помощью JavaScript и считывать или изменять содержимое веб-страницы.

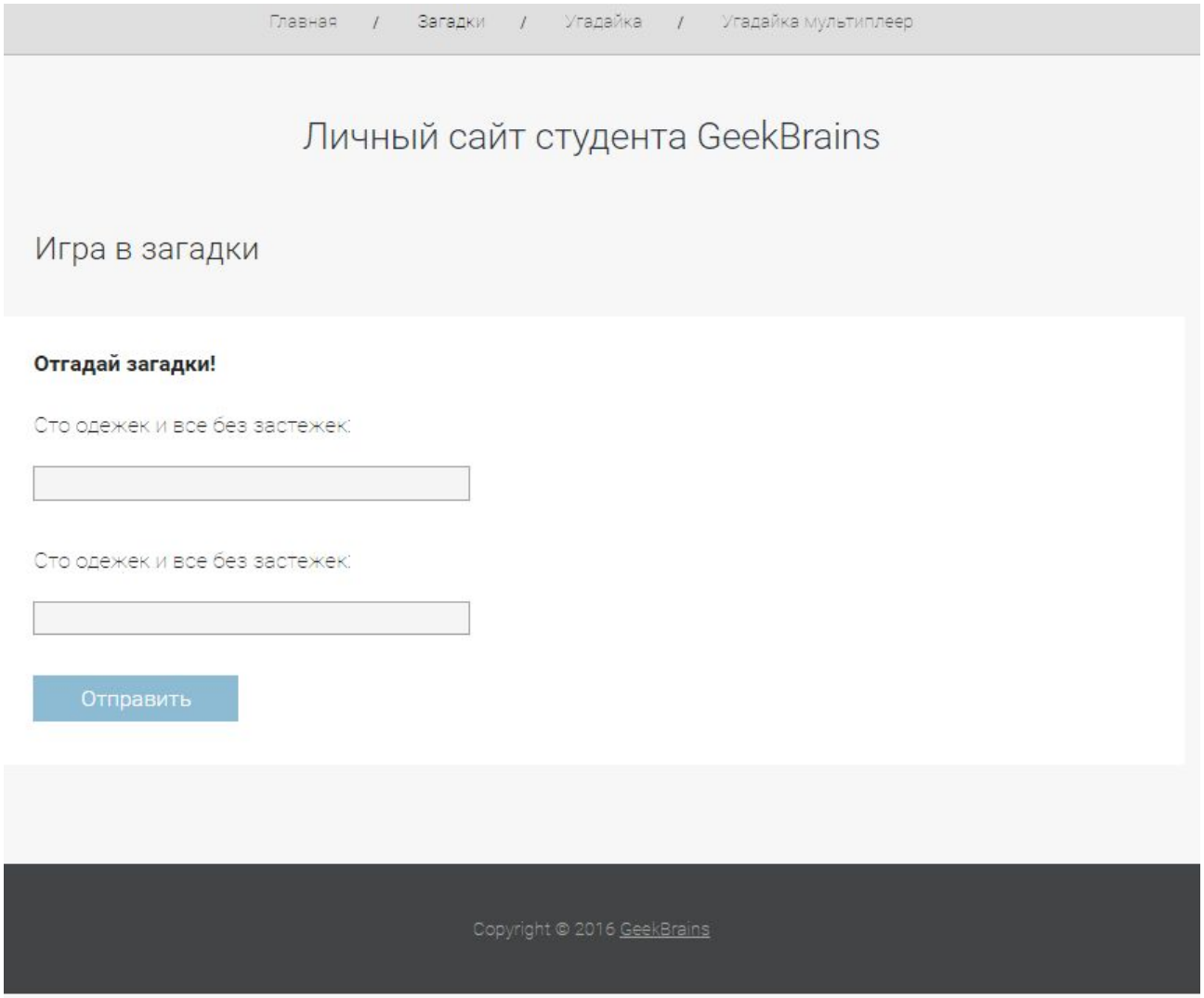
Например:

```
<html>
<head>
    <meta charset="utf-8">
    <title>Пример документа</title>
</head>
<body>
    <h1 id="head1">Заголовок до изменения</h1>
    <p>
        Параграф с <strong id="text1">очень важным </strong>текстом
    </p>
    <script>
        var head1 = document.getElementById("head1");  // Находим элемент в DOM
        head1.innerHTML = "Измененный заголовок";      // Обращаемся к свойству
элемента
    </script>
</body>
</html>
```

Здесь мы находим с помощью специальной функции **getElementById** элемент с идентификатором **head1**. У большинства есть свойства (переменные, которые описывают состояние элемента), и мы используем свойство **innerHTML**, чтобы изменить внутренний текст элемента.

Страница с загадками

Чтобы закрепить знания, создадим интерактивную страницу с загадками.


Главная / Загадки / Угадайка / Угадайка мультиплеер
Личный сайт студента GeekBrains
Игра в загадки
Отгадай загадки!
Сто одежек и все без застежек:
Сто одежек и все без застежек:
Отправить
Copyright © 2016 GeekBrains

input

Познакомимся с новым тегом **<input>**. Этот тег позволяет выводить на страницу различные элементы ввода информации.

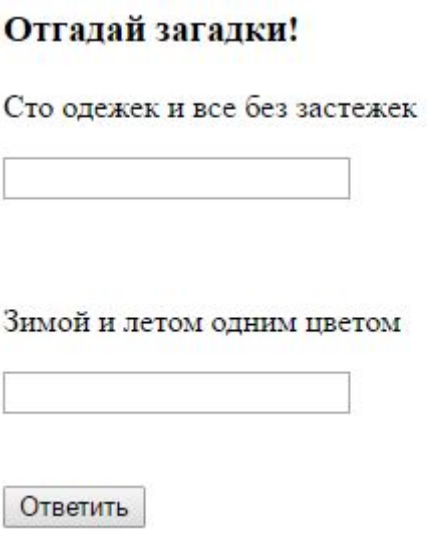
Например:

```
<html>
<head>
    <title>Пример документа</title>
    <meta charset="utf-8">
</head>
<body>
    <h3>Отгадай загадки!</h3>
    <p>
        Сто одежек и все без застежек
    </p>
    <input type="text" id="puzzle1">
    <br><br><br>

    <p>
        Зимой и летом одним цветом
    </p>
    <input type="text" id="puzzle2">
    <br><br><br>

    <button>Ответить</button>
</body>
</html>
```

Программа выведет на экран текст загадки, поля, куда пользователь может вводить ответы, и кнопку с надписью «Ответить».



Мы реализовали возможность вводить ответы, теперь требуется их проверять. Для этого познакомимся с событиями.

События, или events

В JavaScript функции играют еще большую роль, чем в других языках, — они позволяют реализовать механизм событий.

События — это действия, происходящие в браузере: щелчки мышью, нажатия клавиш, перемещения указателя мыши, загрузка рисунка и другие. Самые различные события могут влиять на дальнейшее поведение браузера.

Программы, которые позволяют определять события и выполнять соответствующие им действия, называются **обработчиками событий**.

Заставим кнопку реагировать на нажатие:

```
<button onclick="alert('Событие!');">Ответить</button>
```

Обратите внимание, что нужно использовать разные кавычки в тексте:

```
onclick="alert('Событие!');"
```

Теперь при нажатии на кнопку (возникновении события **click**) будет выполняться команда **alert**.

Но чаще бывает нужно выполнить несколько команд. В этом случае лучше использовать функции:

```
<button onclick="click1();">Ответить</button>

<script>
    function click1()
    {
        alert("События удобнее обрабатывать");
        alert("с помощью функций!");
    }
</script>
```

Скрипт с функцией обработки события может быть описан как до, так и после элемента, в котором назначается обработчик события.

Итоговая программа

Функция checkAnswer

Функция **checkAnswer** упрощает проверку значений. В нее мы передаем **id** элемента и правильный ответ. Функция находит элемент по **id** и запоминает его значение в переменной **userAnswer**. Далее мы сравниваем **userAnswer** с правильным ответом. Если значения переменных совпадают, функция возвращает true, иначе — false.

```
function checkAnswer(id, trueAnswer) {
    var userAnswer = document.getElementById(id).value;
    return userAnswer == trueAnswer;
}
```

Функция checkAnswers

В этой функции мы множество раз обращаемся к функции **checkAnswer** и передаем ей **id** элемента и правильный ответ. Каждый раз, когда **checkAnswer** возвращает истину, переменная **correctAnswers** увеличивается на единицу. В конце мы выводим значение **correctAnswers**.

```
function checkAnswers()
{
    var correctAnswers = 0;

    if (checkAnswer('puzzle1', 'капуста'))
        correctAnswers++;

    if (checkAnswer('puzzle2', 'елка'))
        correctAnswers++;

    alert('Количество правильных ответов: ' + correctAnswers);
}
```

Текст всей программы веб-страницы:

```
<html>
<head>
    <title>Пример документа</title>
    <meta charset="utf-8">
    <script>
        function checkAnswer(id, trueAnswer) {
            var userAnswer = document.getElementById(id).value;
            return userAnswer == trueAnswer;
        }

        function checkAnswers()
        {
            var correctAnswers = 0;

            if (checkAnswer('puzzle1', 'капуста'))
                correctAnswers++;

            if (checkAnswer('puzzle2', 'елка'))
                correctAnswers++;

            alert('Количество правильных ответов: ' + correctAnswers);
        }
    </script>
</head>
<body>
<h3>Отгадай загадки!</h3>
<p>
    Сто одежек и все без застежек
</p>
<input type="text" id="puzzle1">
<br><br><br>

<p>
    Зимой и летом одним цветом
</p>
<input type="text" id="puzzle2">
<br><br><br>

<button onclick="checkAnswers();">Ответить</button>

</body>
</html>
```

Практическое задание

1. Разместить на сайте свою фотографию и написать о себе.
2. Разместить на сайте свои загадки.
3. Разместить на сайте игру «Угадайка» для двух игроков.

4. * Разместить на сайте генератор случайных паролей. Пользователь указывает в текстовом поле нужную длину пароля и получает случайную комбинацию, состоящую из цифр и латинских букв.