Возможности HTML и CSS

1.

**Теги header и footer, хедер и футер**

В этой вводной части мы обзорно пройдём по теме HTML и CSS, освежим в памяти то, [что уже изучили раньше](https://htmlacademy.ru/courses/basic-html-css), а также познакомимся с новыми возможностями.

Для этого мы пошагово соберём блог известного вам веб-разработчика.

Начнём с «шапки» и «подвала» сайта. Разметим их тегами <header> и <footer>.

* <header> — хедер сайта или раздела;
* <footer> — футер сайта или раздела.

Да-да, это те самые хедер и футер, которые раньше верстались дивами с классами header или footer.

index.html

<!DOCTYPE html>

<html lang="ru">

<head>

<title>Теги header и footer, хедер и футер</title>

<meta charset="utf-8">

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

</head>

<body>

Хедер

Футер

</body>

</html>

style.css

html,

body {

margin: 0;

padding: 0;

color: white;

background-color: #2c3e50;

}

/\* Шапка \*/

.page-header {

position: relative;

z-index: 5;

min-height: 20px;

padding: 20px;

color: white;

background-image:

linear-gradient(45deg, #34495e 25%, transparent 25%),

linear-gradient(-45deg, #34495e 25%, transparent 25%),

linear-gradient(135deg, #34495e 25%, transparent 25%),

linear-gradient(-135deg, #34495e 25%, transparent 25%);

background-position: 10px 0, 10px 0, 0 0, 0 0;

background-size: 20px 20px;

box-shadow: 0 0 5px 0 #333333;

}

/\* Подвал \*/

.page-footer {

min-height: 50px;

padding: 20px;

font-size: 14px;

color: black;

background-color: #f1c40f;

background-image:

linear-gradient(45deg, transparent 50%, white 50%),

linear-gradient(-45deg, transparent 50%, white 50%),

linear-gradient(135deg, transparent 50%, #2c3e50 50%),

linear-gradient(-135deg, transparent 50%, #2c3e50 50%);

background-repeat: repeat-x;

background-position: 0 0, 0 0, 0 100%, 0 100%;

background-size: 12px 12px;

}

Задачи

1. Добавьте в body элемент header с классом page-header и текстом Хедер,
2. а сразу после него элемент footer с классом page-footer и текстом Футер

2.

# Тег main, основное содержание

Обратите внимание, что мы задали классы для хедера и футера. Сделано это потому, что этих элементов на странице может быть несколько.

Хедер — это не только привычная шапка сайта с логотипом и меню, он может использоваться и как «шапка» какой-нибудь статьи или раздела сайта. Конечно, в этом случае хедер называют не «шапкой», а вводной частью, в которой могут содержаться заголовки, оглавление и так далее.

С футером ситуация аналогичная. В привычном нам понимании это «подвал» сайта, с копирайтами, контактной информацией и так далее. Но футер может использоваться и в других разделах сайта. Например, в футере статьи можно разместить дополнительную информацию: данные об авторе, дополнительные ссылки и так далее.

А раз теги не уникальные, то и стилизовать их лучше с помощью классов, как мы и сделали.

Если вы не хотите использовать классы для шапки и подвала сайта, то можете использовать селекторы body > header и body > footer. Эти селекторы не повлияют на хедеры и футеры, вложенные более глубоко.

Для разметки основного содержимого используем тег <main>. Для его стилизации класс использовать не будем, так как на сайте будет только одно единственное основное содержимое.

Кстати, загляните в CSS и посмотрите, как сделаны фоны для основных блоков. В хедере и футере мы использовали линейные градиенты, которые детально разберём в более позднем тренажёре. А фоновую картинку для основного содержания мы задали с помощью так называемого data:URI. Это не опечатка, аббревиатура URI расшифровывается как **U**niform **R**esource **I**dentifier. Кодирование изображения прямо в CSS-коде — это одна из продвинутых техник оптимизации вёрстки.

Задачи

1. Добавьте между хедером и футером элемент main c текстом Основное содержание
2. Добавить в стили

main {

min-height: 200px;

padding-top: 50px;

color: black;

background-color: white;

background-image: url("data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAACMAAAAjCAMAAAApB0NrAAAABlBMVEX///+pqalTpKl9AAAAAnRSTlMAGovxNEIAAAAdSURBVDjLY2AAAUaG4UYxDDsfjfp91O+jfqfE7wCzpQCw/dq+qQAAAABJRU5ErkJggg==");

}



# 3. Теги article и section, разделы страницы

С шапками, подвалами и основным содержанием разобрались. Теперь перейдём к остальным разделам сайта.

Разметим разделы «Обо мне» и «Последние посты». Используем для этого теги <section> и <article>:

* <section> — смысловой или логический раздел документа;
* <article> — самостоятельный и независимый раздел документа.

Чтобы не было путаницы, разберём где и когда использовать разные контейнеры:

* <div> — контейнер общего назначения, не обязательно смысловой. Дивы используются для разметки мелких блоков, создания сетки и декоративных эффектов.
* <section> — более крупный логический контейнер, объединяющий содержание по смыслу. Например, блок «О компании», список товаров, раздел личной информации в профиле и так далее.
* <article> — самостоятельный, цельный и независимый раздел документа. Этот раздел можно в неизменном виде использовать в различных местах, в том числе и на других сайтах. Примеры: статья, пост в блоге, сообщение на форуме и так далее.

В этом задании мы сверстаем блок «Обо мне» как независимый раздел, который можно будет использовать в разных местах сайта. А список постов — это логический раздел, группирующий последние посты, поэтому для него используем <section>.

Задачи

1. Сначала удалите текст из main.
2. Затем добавьте внутрь main тег article с классом about-me и текстом Обо мне.
3. А сразу после article добавьте тег section с классом posts и текстом Последние посты
4. Добавить стили

main::after {

content: "";

display: table;

clear: both;

}

.about-me {

min-height: 100px;

margin: 0 20px;

padding: 10px;

background-color: rgba(241, 196, 15, 0.1);

border-radius: 5px;

box-shadow: 0 0 3px #cccccc;

}

.posts {

min-height: 50px;

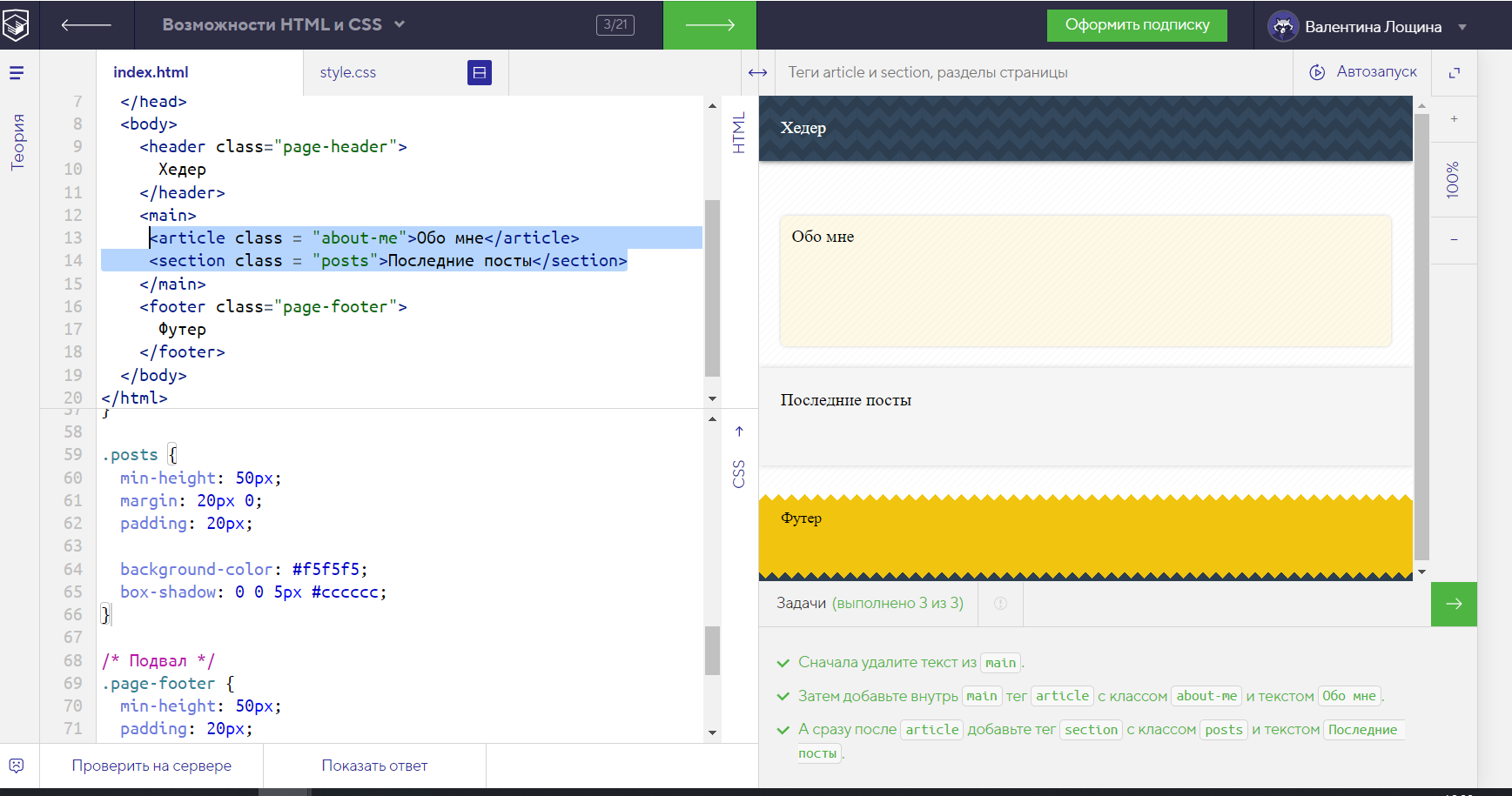
margin: 20px 0;

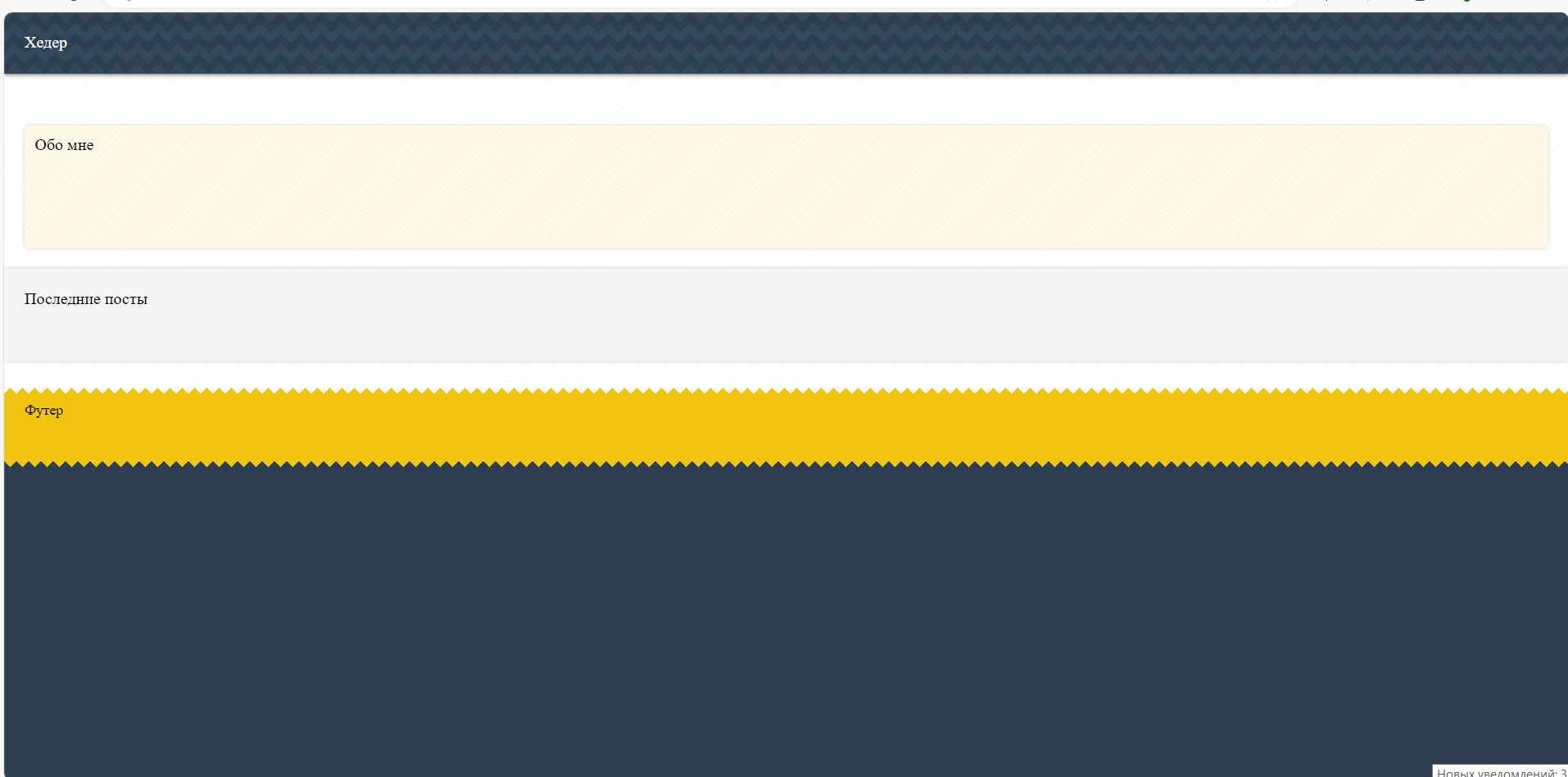
padding: 20px;

background-color: #f5f5f5;

box-shadow: 0 0 5px #cccccc;

}





# 4. Теги h1-h6, заголовки разделов страницы

Как мы [уже разбирали ранее](https://htmlacademy.ru/courses/299/run/11), для структурирования текстового содержания важно правильно озаглавить основные разделы документа.

Как правило, в разделы <section> и <article> включают заголовки, резюмирующие содержимое разделов.

Всего может быть шесть уровней заголовков. Размечаются они тегами <h1>, <h2> … <h6>. А как правильно определить нужный уровень заголовков?

При озаглавливании разделов учитывается общая иерархия текстового содержимого страницы: сначала определяется, о чём эта страница целиком, какие разделы документа более крупные, какие — дополнительные. В итоге заголовки должны из себя представлять явное дерево: от главного <h1> на странице до структурно следующих за ним <h2>, <h3> и так далее.

Пример:

h1 — Интернет-магазин зоотоваров «Усы, лапы и хвост»

h2 — Список товаров

h2 — Похожие предложения

h3 — Когтеточка «Росомаха»

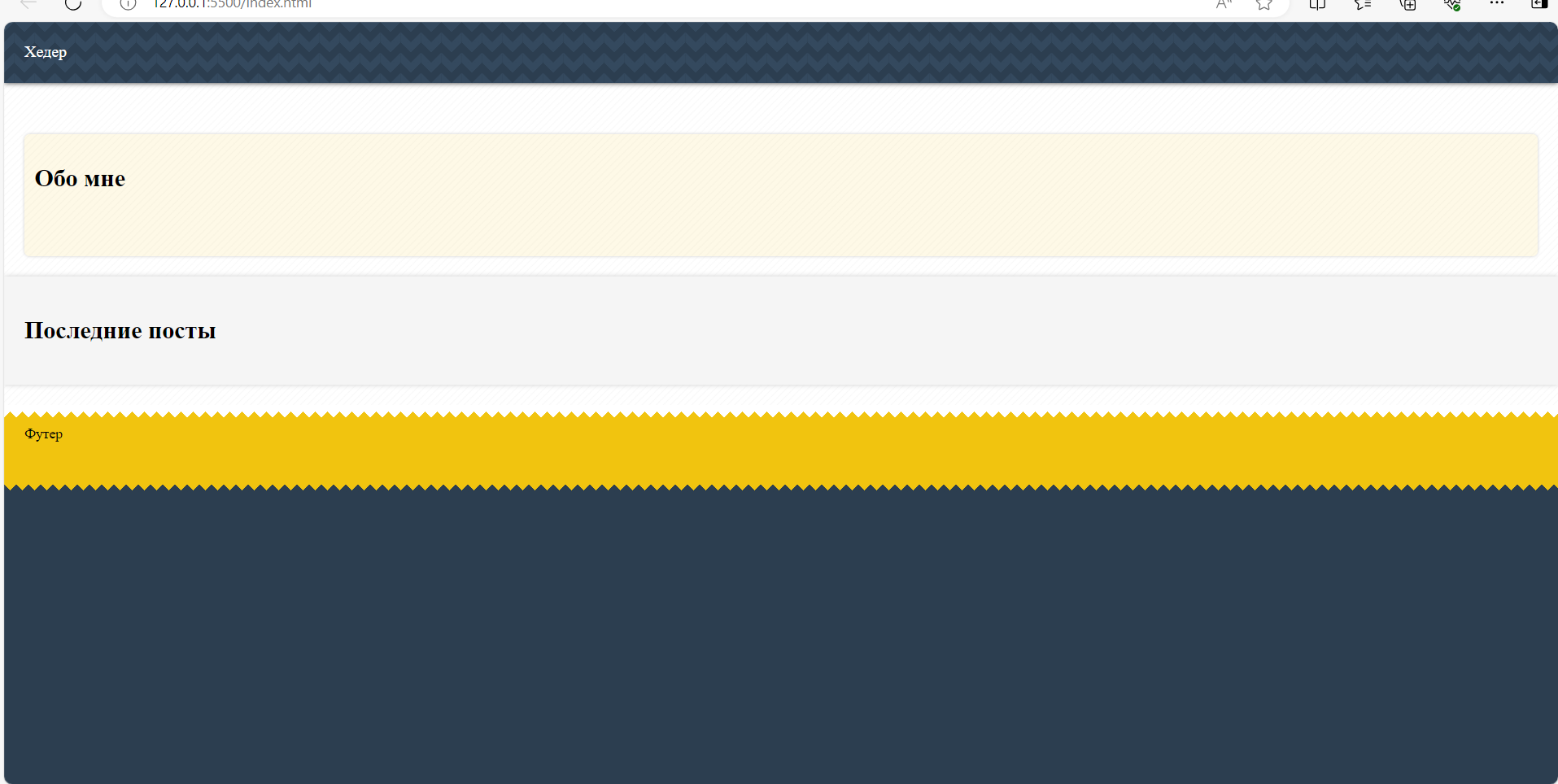
h3 — Палка с пером «Шерстяной охотник»

…

Если в дизайне страницы у раздела не предусмотрен заголовок, его всё равно следует добавлять в разметку. При этом с помощью стилей заголовки можно «визуально спрятать», но оставить доступным их содержимое. Подробнее об этом рассказывается в [Шорте «Как прятать»](https://htmlacademy.ru/blog/boost/frontend/short-12).

Задачи

1. В блоке .about-me оберните текст Обо мне в тег <h1>,
2. В блоке .posts оберните текст Последние посты в тег <h2>.



# 5. Изображения в формате SVG

Мы разметили базовые разделы нашей страницы. Теперь поработаем над каждым из них и начнём с шапки сайта.

Давайте добавим нашему сайту логотип. Для этого добавим в хедер <div>, а внутри него разместим привычное изображение. Используем картинку в векторном формате SVG. Векторные изображения можно уменьшать и увеличивать без потери качества, в этом вы убедитесь в задании.

Кстати, обратите внимание на полупрозрачный фон в блоке «Обо мне». Он сделан с помощью rgba-формата записи цветов в CSS.

Задачи

1. Добавьте в хедер div с классом logo.
2. Внутрь дива вставьте изображение с адресом logo.svg, шириной и высотой 90 пикселей.
3. Увеличьте ширину и высоту этого изображения до 300 пикселей.

Размеры изображения задавайте с помощью атрибутов тега <img>, как в [этом задании](https://htmlacademy.ru/courses/305/run/11).



<body>

<header class="page-header">

<div class="logo"><img src = "logo.svg" width = "300" height = "300"></div>

</header>

<main>

<article class="about-me">

<h1>Обо мне</h1>

</article>

<section class="posts">

<h2>Последние посты</h2>

</section>

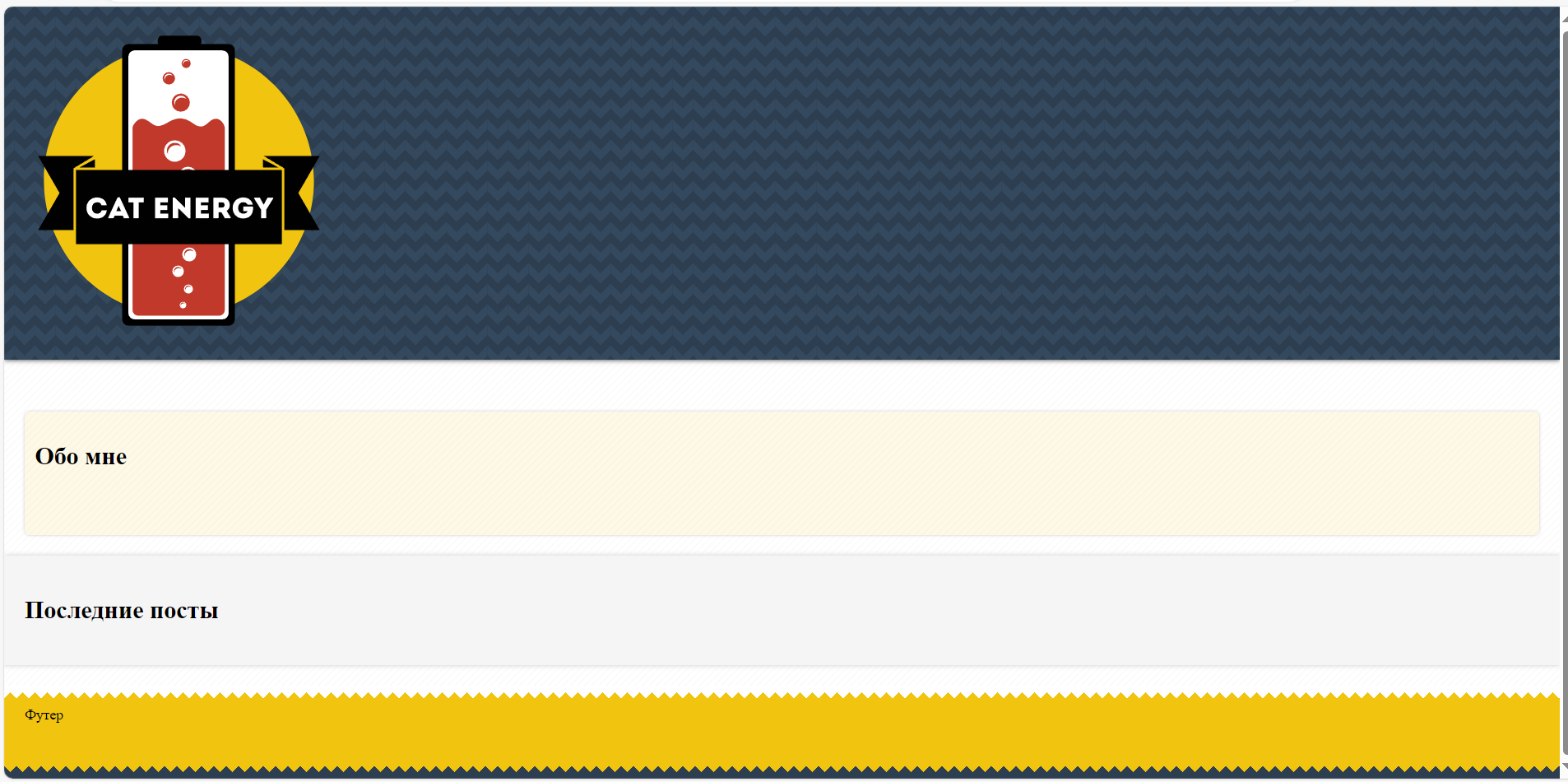
</main>

<footer class="page-footer">

Футер

</footer>

</body>



# 6. Тег nav, основная навигация

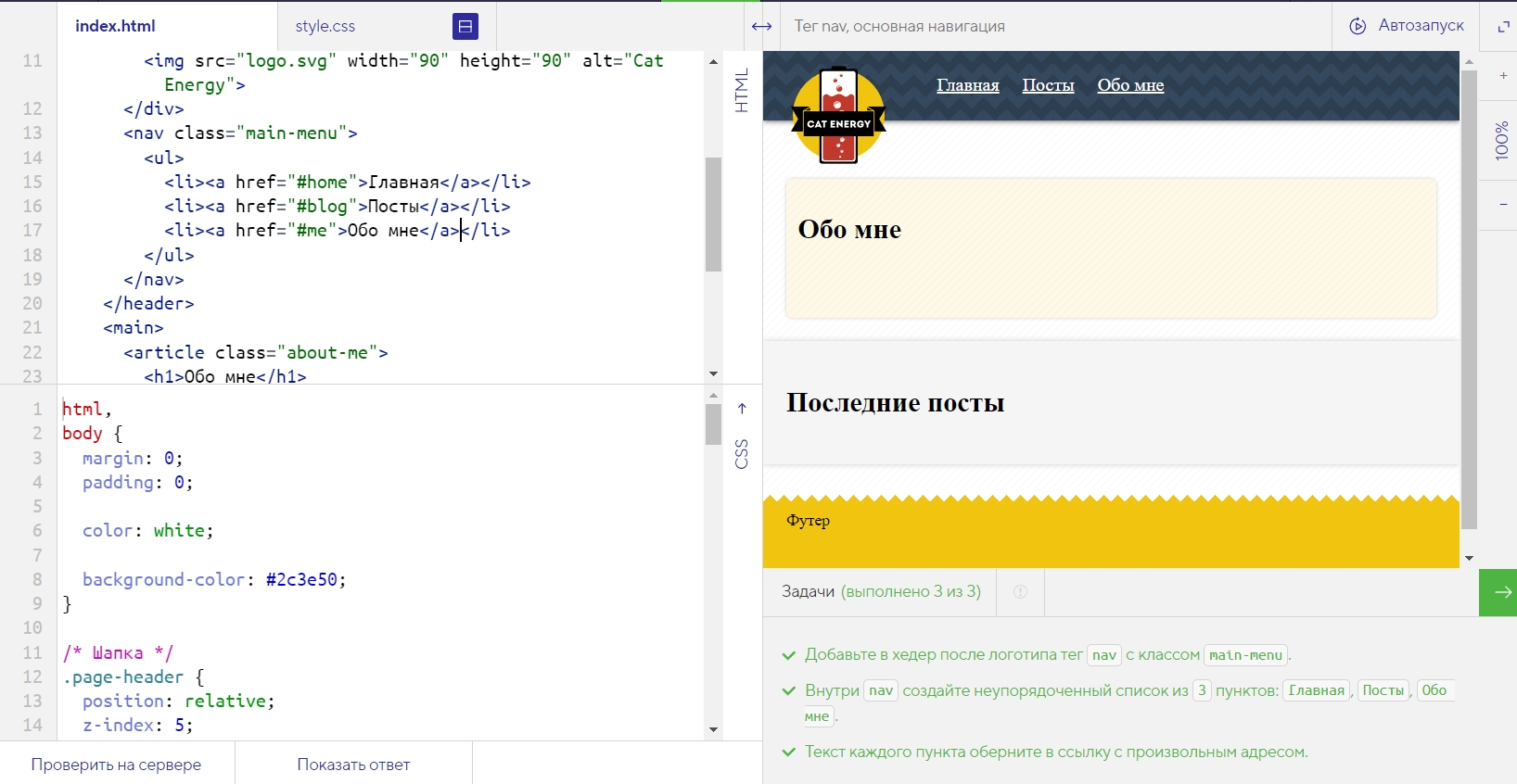
Теперь давайте добавим в шапку главное меню.

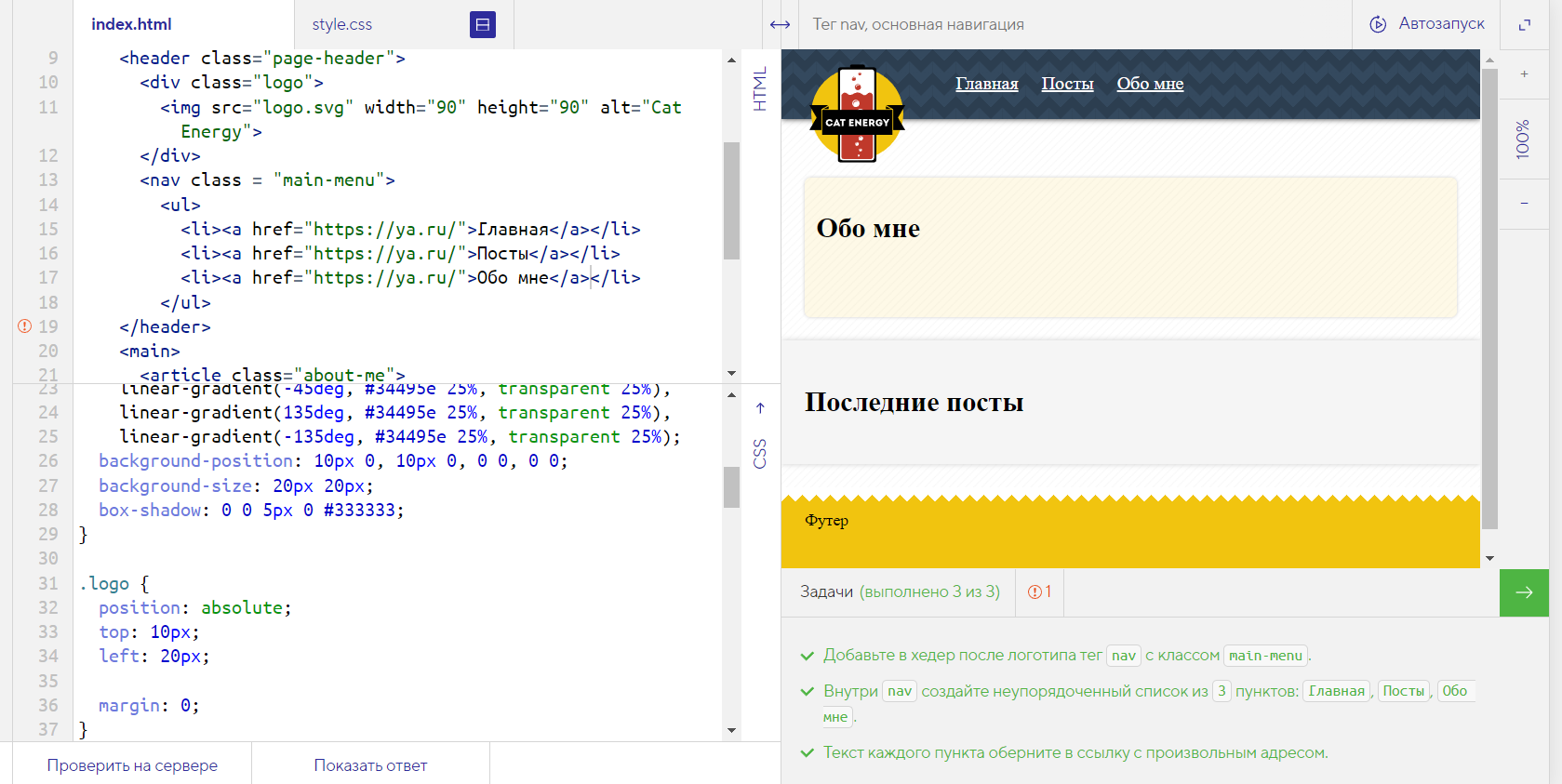
Как вы помните из [прошлой части](https://htmlacademy.ru/courses/299/run/8), в HTML есть специальный элемент для навигационных блоков — тег <nav>. Так что меню и блоки ссылок лучше оборачивать не в дивы, а в навы.

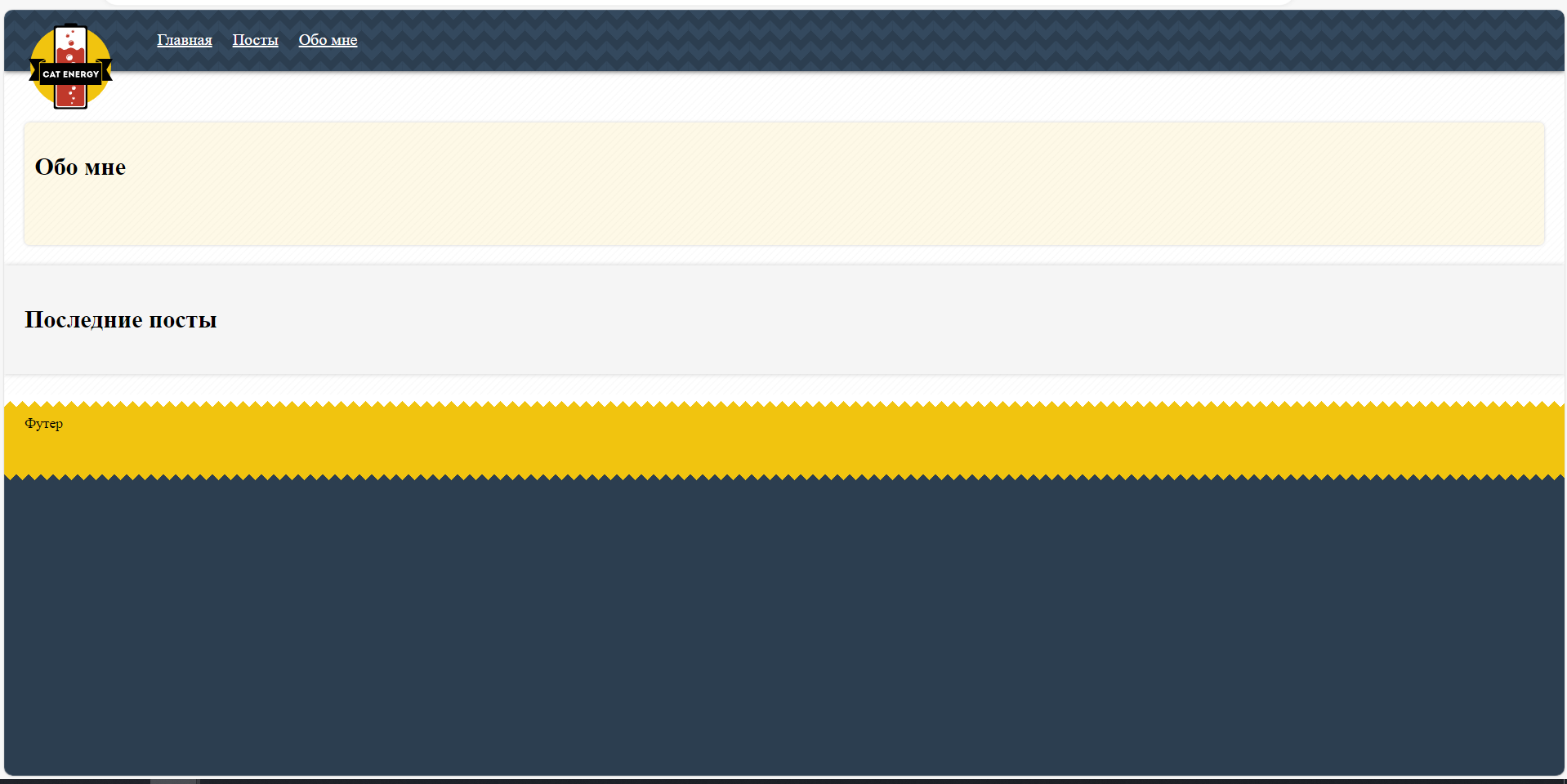
В случае, когда навигация шапки дублируется в подвале сайта, не рекомендуется использовать тег <nav> повторно. Достаточно будет использовать список со ссылками.

Задачи

1. Добавьте в хедер после логотипа тег nav с классом main-menu.
2. Внутри nav создайте неупорядоченный список из 3 пунктов: Главная, Посты, Обо мне.
3. Текст каждого пункта оберните в ссылку с произвольным адресом.
4. <img src="logo.svg" width="90" height="90" alt="Cat Energy">







<header class="page-header">

      <div class="logo"><img src = "logo.svg" width = "90" height = "90"></div>

      <nav class = "main-menu">

        <ul>

          <li><a href="https://ya.ru/">Главная</a></li>

          <li><a href="https://ya.ru/">Посты</a></li>

          <li><a href="https://ya.ru/">Обо мне</a></li>

        </ul>

        </nav>

    </header>

# 7. Завершаем футер

Теперь перейдём к доработке подвала сайта. Там нет ничего сложного: для стилизации нужно добавить пару контейнеров-обёрток. Использовать <section> в этом случае избыточно, дивы нужны только для создания сетки и оформления.

Обратите внимание на то, как повернётся изображение автора блога. Это сделано с помощью CSS-трансформаций, которые мы детально разберём в одной из последующих частей.

# Завершаем футер

Теперь перейдём к доработке подвала сайта. Там нет ничего сложного: для стилизации нужно добавить пару контейнеров-обёрток. Использовать <section> в этом случае избыточно, дивы нужны только для создания сетки и оформления.

Обратите внимание на то, как повернётся изображение автора блога. Это сделано с помощью CSS-трансформаций, которые мы детально разберём в одной из последующих частей.

Задачи

Внесите следующие правки в футер:

1. Оберните фото кота, его имя и адрес почты в div с классом info,
2. а информацию о копирайте в div с классом copyright

3. <footer class="page-footer">

<img src="keks.jpg" width="50" alt="Портрет Кекса">

Кекс, keks@htmlacademy.ru

© 2008—2014 Cat Energy

</footer>

4. Сохранить картинку с правильным названием, чтобы она появилась на странице.

5.

.page-footer::after {

content: "";

display: table;

clear: both;

}

.page-footer .info {

float: left;

}

.page-footer .info img {

margin-right: 0.5em;

vertical-align: middle;

border: 2px solid white;

box-shadow: 0 0 4px #c0392b;

transform: rotate(-5deg);

}

.page-footer .copyright {

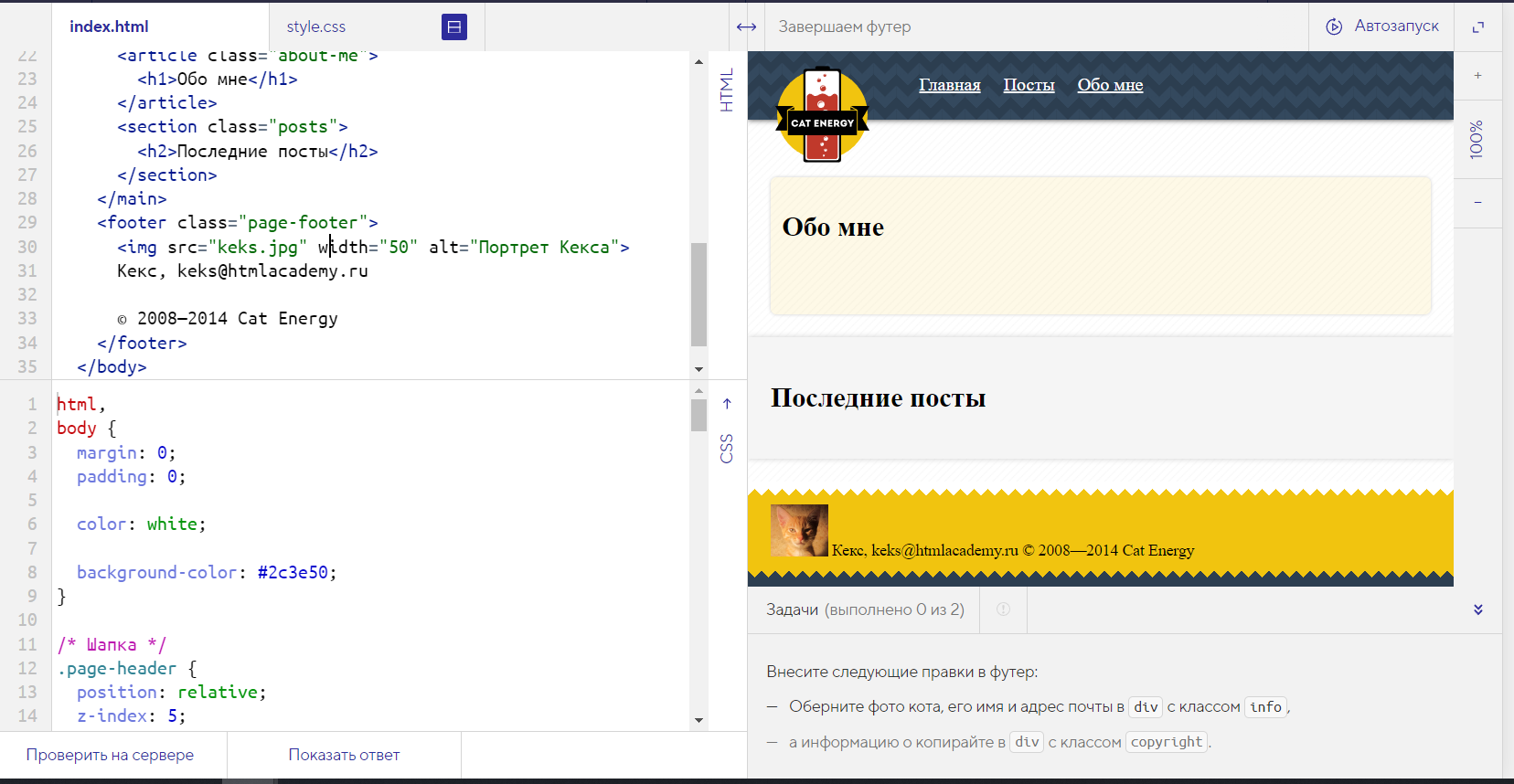
float: right;

padding-top: 20px;

font-size: 12px;

}





<div class="info">

<img src="keks.jpg" width="50" alt="Портрет Кекса">

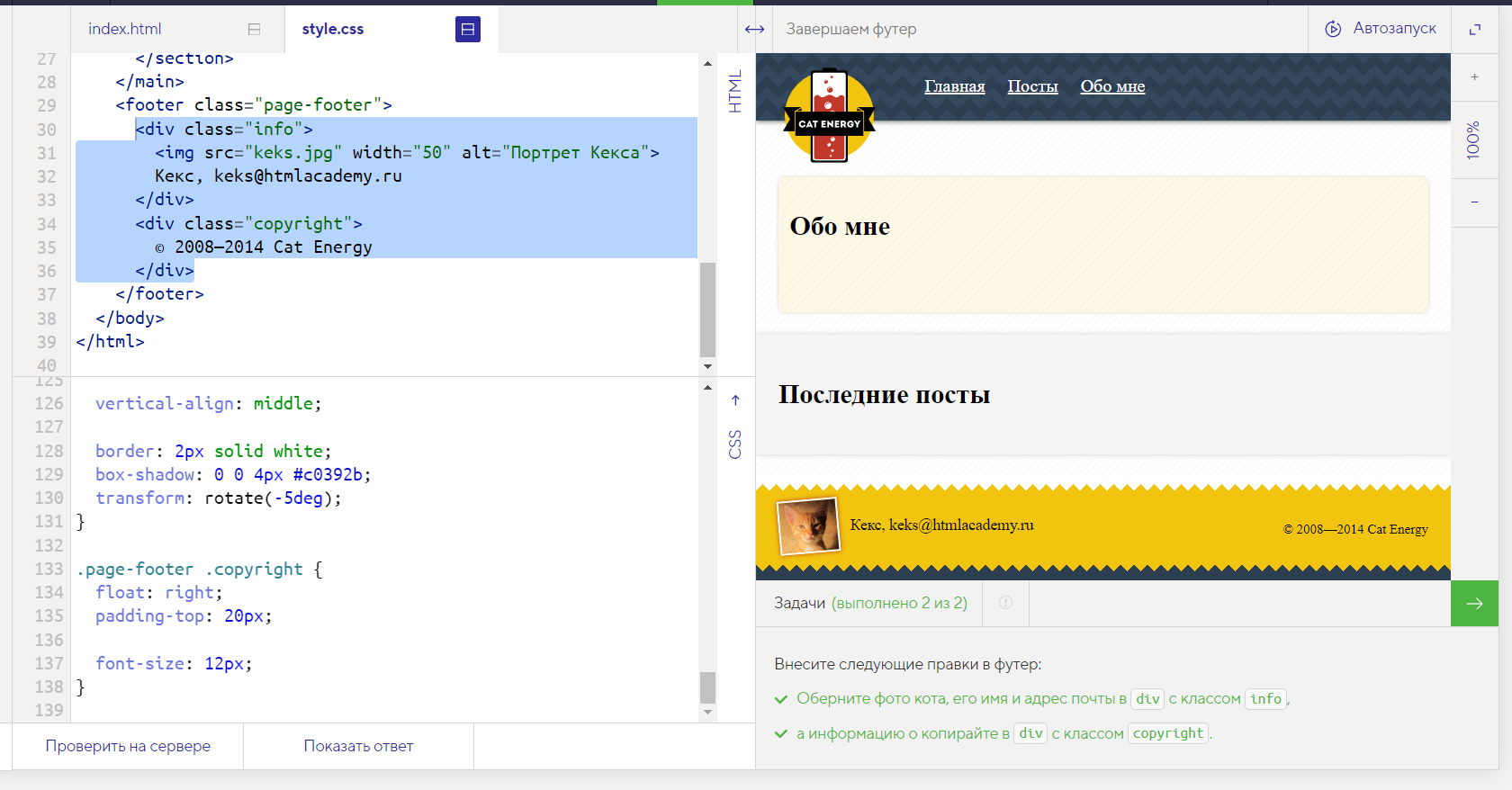
Кекс, keks@htmlacademy.ru

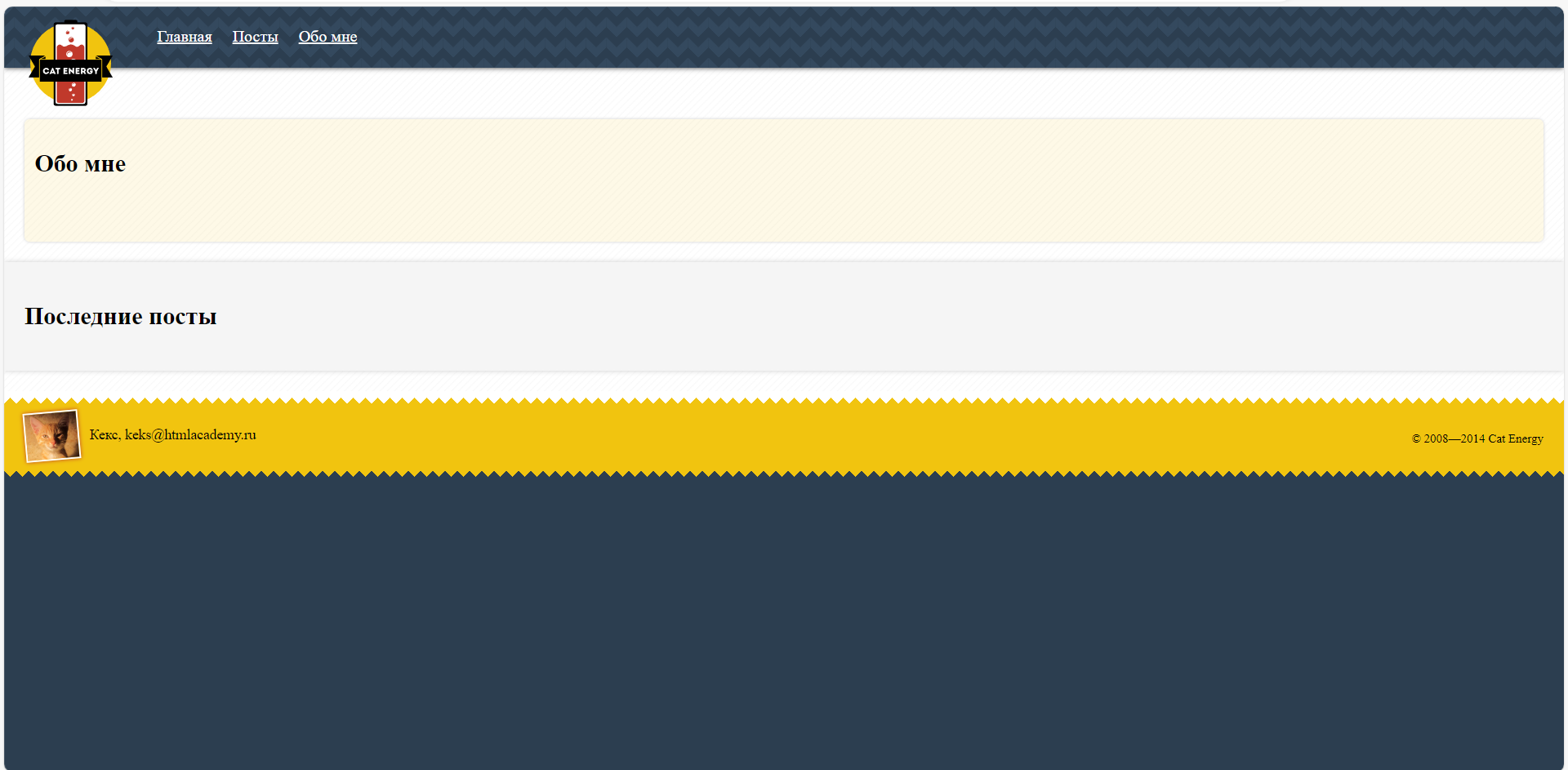
</div>

<div class="copyright">

© 2008—2014 Cat Energy

</div>





# 8. Использование нестандартных шрифтов

Перед тем как продолжить работу над сайтом, давайте разберёмся со шрифтами.

В HTML и CSS есть возможность подключать и использовать на странице нестандартные шрифты. Простейший способ использовать такой шрифт — найти его в специальном сервисе по названию, получить там код подключения шрифта, вставить этот код в свою вёрстку и использовать шрифт, как обычно, с помощью свойства font-family.

Подобных сервисов с бесплатными шрифтами достаточно много. Один из самых известных — это [Google Fonts](https://www.google.com/fonts). А вот похожий сервис: [fontstorage](https://fontstorage.com/ru/" \t "_blank).

При использовании веб-шрифтов не забывайте указывать так называемые «фоллбэк»-шрифты — стандартные шрифты, которые будут отображаться, если веб-шрифт либо недоступен, либо не поддерживается старым браузером пользователя. Для этого нужно всего лишь перечислить их через запятую после нестандартного шрифта:

font-family: "PT Sans", "Arial", sans-serif;

Потренируемся подключать и использовать веб-шрифты.

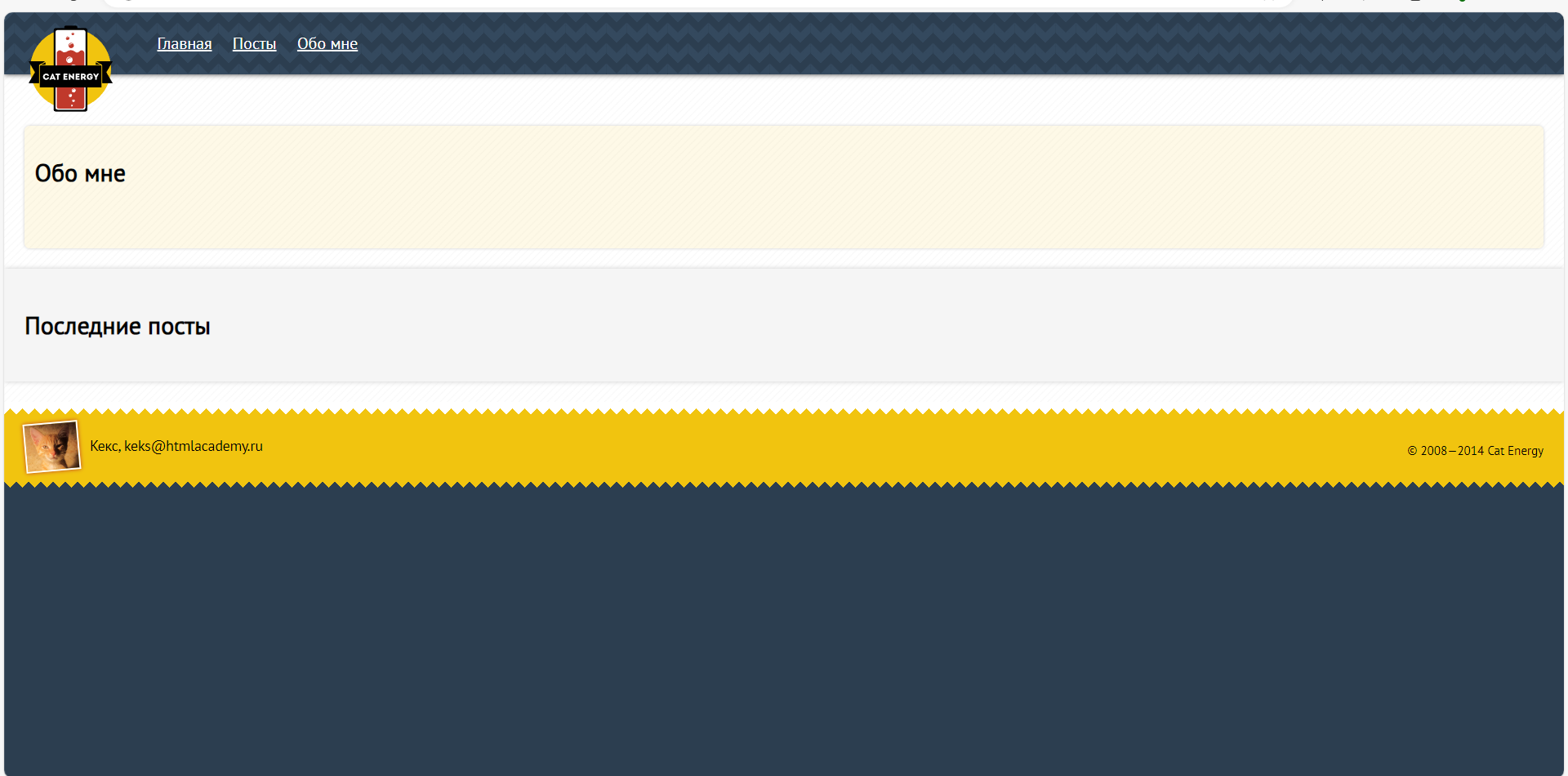
Задачи

1. Вставьте тег link внутри head.

<link href="//fonts.googleapis.com/css?family=PT+Sans:400&subset=cyrillic" rel="stylesheet" type="text/css">

1. Задайте для body шрифт "PT Sans", "Arial", sans-serif

font-family: "PT Sans", "Arial", sans-serif;



# 9. Подробнее о шрифтах: правило @font-face

Технически подключение веб-шрифтов производится с помощью CSS-правила @font-face. Читается как «эт-правило font-face». Вот пример:

@font-face {

font-family: "Roboto";

src:

local("Roboto Regular"),

url("roboto.woff") format("woff");

}

В этом правиле вы указываете название шрифта, которое будете использовать в font-family и источники, из которых браузер сможет загрузить шрифт. Обычно сначала указывают название шрифта в системе, чтобы браузер попытался найти его локально на компьютере пользователя, а затем указывают адрес файла шрифта в интернете.

Более подробно о технических тонкостях подключения шрифтов можно прочитать в серии статей на fontstorage: [первая часть](https://fontstorage.com/blog/about-font-face-part-one/) и [вторая часть](https://fontstorage.com/blog/about-font-face-part-two/).

Получается, что можно хранить шрифты и подключать их со своего сервера. Это особенно полезно, когда шрифт очень редкий и его нет ни в одном из шрифтовых сервисов. В этом случае поступают так:

1. Берут файл шрифта (например, .ttf) и конвертируют в веб-формат в сервисе наподобие [Font Squirrel](http://www.fontsquirrel.com/tools/webfont-generator) (для кириллических шрифтов надо указать дополнительные параметры конвертации).
2. Затем сконвертированные файлы шрифта размещают у себя на сервере.
3. И подключают шрифт с помощью @font-face.

Сейчас мы подключим шрифт, который расположен на сервере htmlacademy. Мы уже сконвертировали шрифт в веб-формат.

@font-face {

font-family: "Roboto";

font-style: normal;

font-weight: 400;

src:

local("Roboto-Regular"),

url("roboto.woff") format("woff");

}

Задачи

1. Удалите тег link внутри head, у которого атрибут href имеет значение //fonts.googleapis.com/css?family=PT+Sans:400&subset=cyrillic.
2. Вставьте правило @font-face в CSS.
3. Задайте для body шрифт "Roboto", sans-serif

font-family: "Roboto", sans-serif;

# 10. Тег aside, дополнительное содержание

Теперь продолжим работу над страницей блога. Детальнее разметим раздел «Обо мне».

Этот раздел состоит из двух частей. В первой части будет содержаться основная информация, а во второй вспомогательная. Поэтому основную часть мы разметим с помощью уже знакомого <section>. А вторую часть разметим с помощью тега <aside>.

<aside> — это дополнительное содержание, не связанное напрямую с основным. Ещё такие блоки часто называют «сайдбарами» или боковыми панелями.

Задачи

1. Внутрь .about-me после заголовка добавьте тег section с классом profile и текстом Личная информация.
2. А сразу после блока .profile добавьте тег aside с классом last-actions и текстом Последняя активность
3. Добавить в стили

.about-me::after {

content: "";

display: table;

clear: both;

}

.profile {

float: left;

width: 51%;

min-height: 100px;

padding: 2%;

background-color: rgba(243, 156, 18, 0.1);

border-radius: 5px;

}

.last-actions {

float: right;

width: 40%;

min-height: 100px;

background-color: #f1c40f;

border: 5px solid white;

border-bottom-width: 15px;

box-shadow: 0 0 8px -1px #f39c12;

}

В index.html должно быть так.

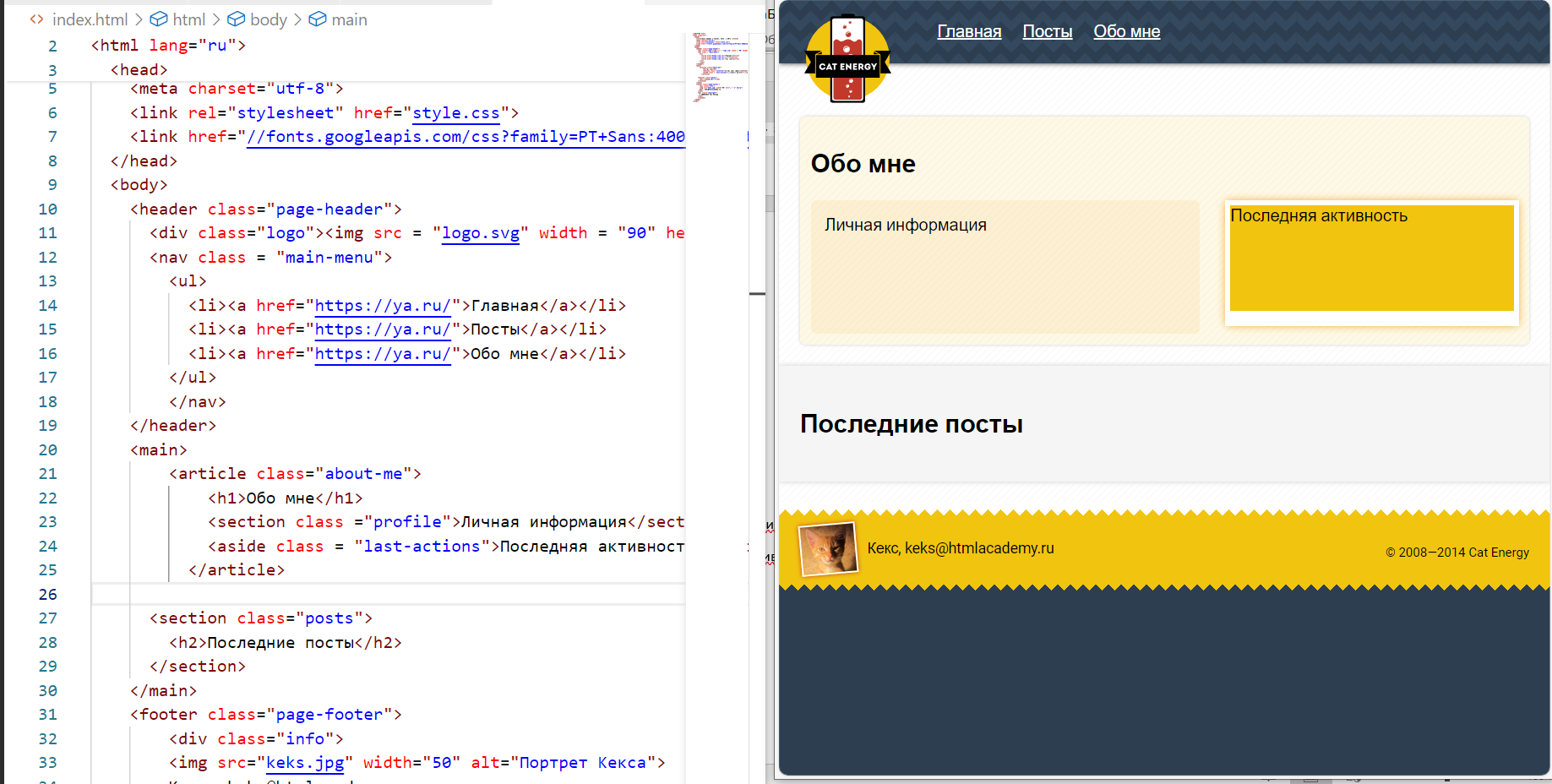
<article class="about-me">

<h1>Обо мне</h1>

<section class ="profile">Личная информация</section>

<aside class = "last-actions">Последняя активность</aside>

</article>



# 11. Ещё раз про article: анонс поста

Теперь добавим пост в блок с последними постами.

Пост — это цельный, законченный и самостоятельный фрагмент информации. Для его разметки идеально подойдёт тег <article>.

Так как <article> должен быть независимым, то при разметке удобно считать его отдельной и самостоятельной страницей сайта. При этом иерархия заголовков на всей нашей странице должна быть «сквозная», то есть внутри <article> заголовок будет «очередного» уровня. Заголовок родительского раздела <section> у нас второго уровня — h2, значит внутри <article> нужен заголовок третьего уровня — h3.

Задачи

1. Внутрь .posts после заголовка добавьте тег article с классом featured-post.
2. Внутрь .featured-post добавьте тег h3 с текстом Заголовок поста.
3. Сразу после заголовка h3 добавьте тег p c текстом Анонс поста

a:link,

a:visited,

a:hover,

a:active {

color: #c0392b;

}

<section class="posts">

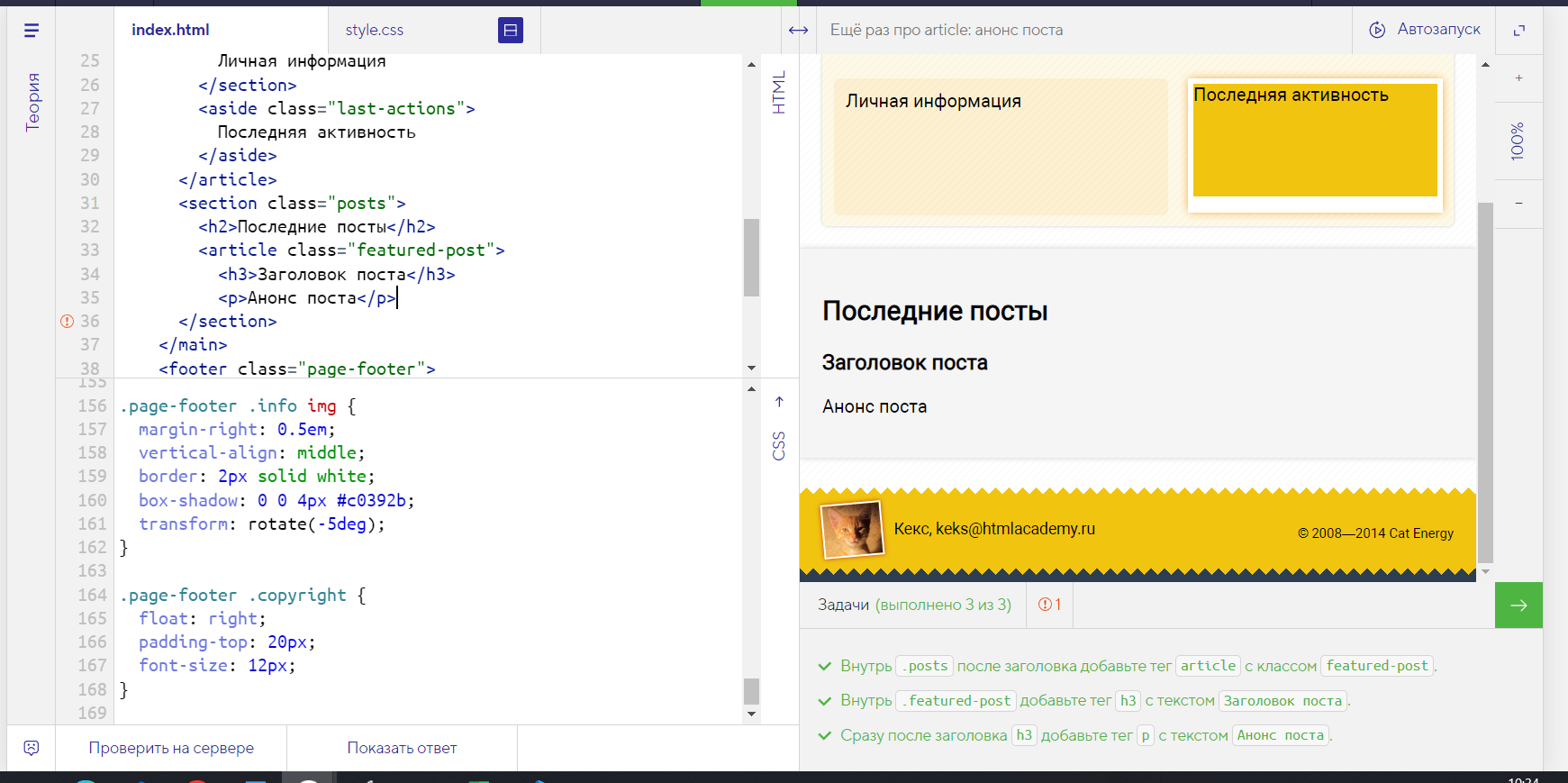
<h2>Последние посты</h2>

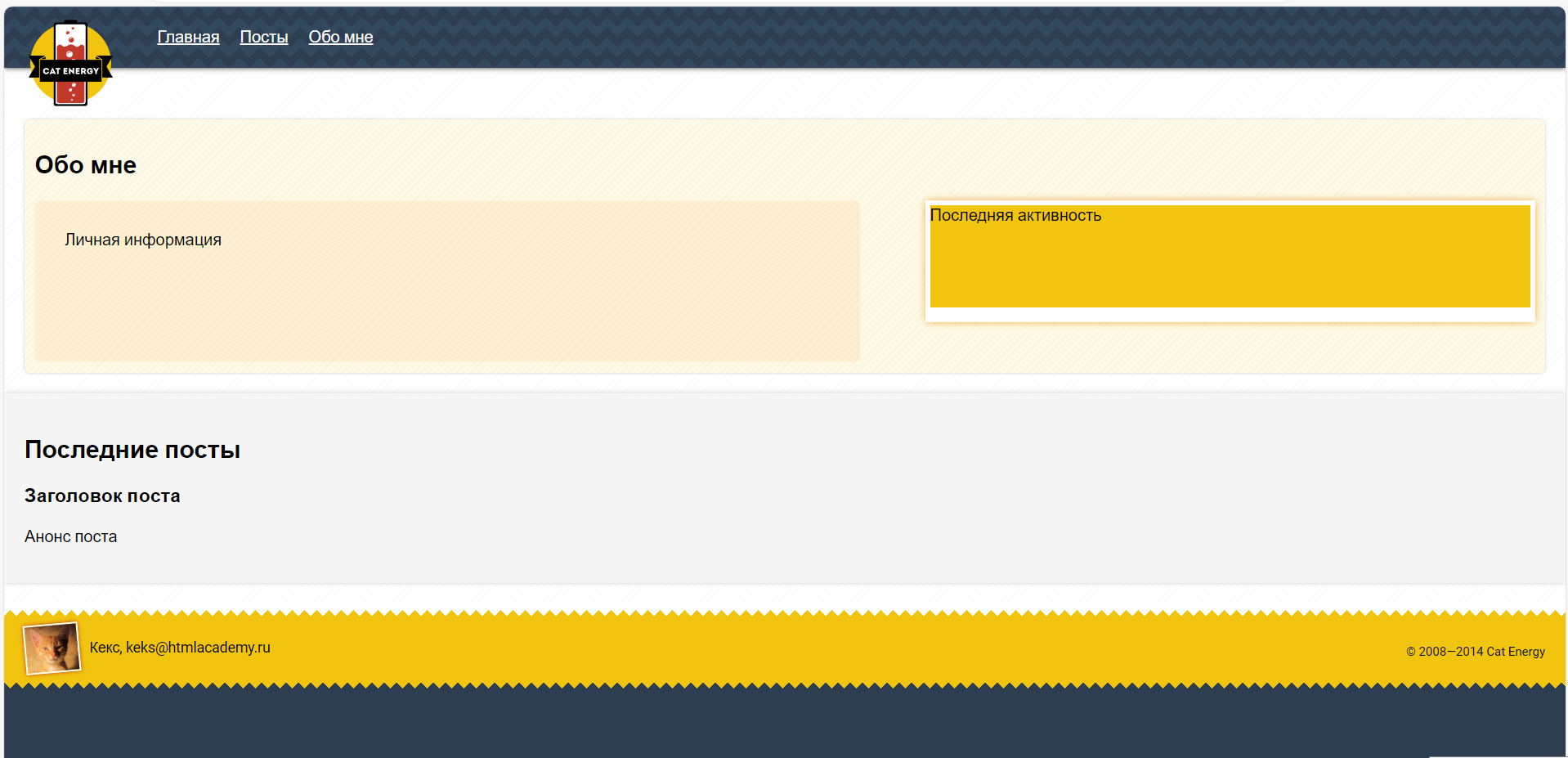
<article class="featured-post">

<h3>Заголовок поста</h3>

<p>Анонс поста</p>

</section>





# 12. Завершаем главную: наполнение

Это задание самое простое. Мы наполним подготовленную в предыдущих шагах разметку настоящим содержанием и посмотрим на готовую страницу.

На этом первая серия заданий завершается. Во второй серии заданий мы будем размечать внутренние страницы блога с самими постами и разберём ещё несколько новых элементов.

Задачи

1. Раскомментируйте код в разделе profile,

<section class="profile">

<!--

<p>Меня зовут Кекс. Я кот и веб-разработчик.</p>

<p><small>Живу в Санкт-Петербурге и работаю в HTML Academy консультантом по созданию учебных курсов.</small></p>

-->

</section>

1. а также в last-actions

<aside class="last-actions">

<!--

<ul>

<li>Зачекинился на кухне <time>час назад</time></li>

<li>Принесли немного еды <a href="#catfood">#catfood</a> <time>два часа назад</time></li>

<li>Зачекинился в ванной <time>вчера</time></li>

</ul>

-->

</aside>

1. и в featured-post

<article class="featured-post">

<!--

<h3><a href="post-1">Чешуя на градиентах</a></h3>

<p>В этом посте я расскажу как делать повторяющийся чешуйчатый фон на CSS-градиентах. Будем использовать градиенты в сочетании со свойством <code>background-size</code>.</p>

-->

</article>

.profile h1 {

margin: 0;

font-size: 1.7em;

}

.profile p {

margin: 0.5em 0;

}

5.

.last-actions ul {

margin: 0;

padding: 10px;

list-style: none;

}

.last-actions li {

margin-bottom: 0.5em;

font-size: 14px;

font-family: "Courier New", monospace;

}

.last-actions time {

display: block;

font-size: 11px;

color: #666666;

}

6.

.featured-post header h1 {

margin: 0;

margin-bottom: 0.8em;

}

Вопрос

Почему ссылки на странице разного цвета

Создаем новый документ post-1.html

<!DOCTYPE html>

<html lang="ru">

<head>

<title>Внутренняя страница: структура поста</title>

<meta charset="utf-8">

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

</head>

<body>

<header class="page-header">

<div class="logo">

<img src="logo.svg" width="90" height="90" alt="Cat Energy">

</div>

<nav class="main-menu">

<ul>

<li><a href="#home">Главная</a></li>

<li><a href="#blog">Посты</a></li>

<li><a href="#me">Обо мне</a></li>

</ul>

</nav>

</header>

<main>

<!-- Основное содержание -->

</main>

<footer class="page-footer">

<div class="info">

<img src="keks.jpg" width="50" alt="Портрет Кекса">

Кекс, keks@htmlacademy.ru

</div>

<div class="copyright">

© 2008—2014 Cat Energy

</div>

</footer>

</body>

</html>

# 13. Внутренняя страница: структура поста

В этом задании мы начинаем работу над внутренними страницами блога, то есть над самими постами.

Первым делом нам нужно разметить базовую структуру поста. Она будет похожа на ту, которую мы делали в [прошлом задании](https://htmlacademy.ru/courses/73/run/11). Но будут и небольшие различия, ведь на главной странице был анонс поста, а здесь пост целиком.

Как вы уже догадались, в качестве контейнера поста мы используем <article>. У каждого поста будет заголовок и дата публикации, а также блок введения и блок с основным содержанием. Не будем переусложнять структуру разметки — дополнительно добавим только два <div> для оформления.

Задачи

1. Внутрь основного содержания добавьте <article> с классом post.
2. Внутрь статьи добавьте заголовок первого уровня с текстом Заголовок поста.
3. Сразу после заголовка добавьте <div> с классом intro и текстом Введение.
4. После блока .intro добавьте <div> с классом content и текстом Содержание поста.

code {

padding: 0.2em 0.4em;

font-weight: bold;

color: #c0392b;

background-color: white;

box-shadow: 0 0 2px #cccccc;

}

6.

.post {

margin: 0 20px;

}

.post h1 {

margin: 0;

}

.post time {

margin-bottom: 1em;

font-size: 11px;

color: #999999;

}

.post .intro {

margin: 1em 0;

padding: 10px;

background-color: rgba(241, 196, 15, 0.1);

border-radius: 5px;

}

.post .intro p {

margin: 0;

}

.post .content {

min-height: 200px;

margin-bottom: 20px;

font-size: 14px;

}

.post .content h2 {

margin-top: 1em;

margin-bottom: 0.5em;

font-size: 18px;

}

Результат

<main>

<article class="post">

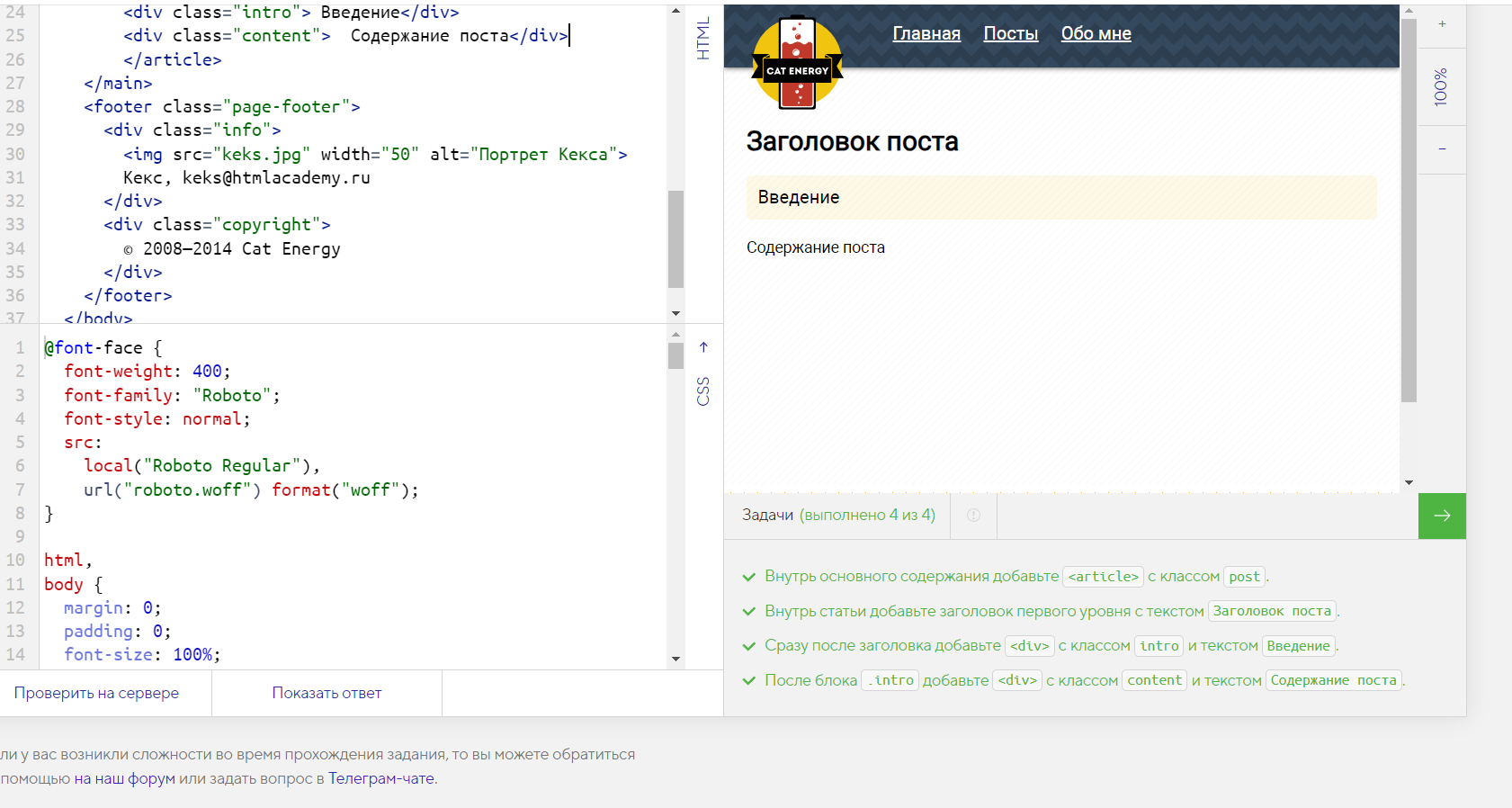
<h1> Заголовок поста</h1>

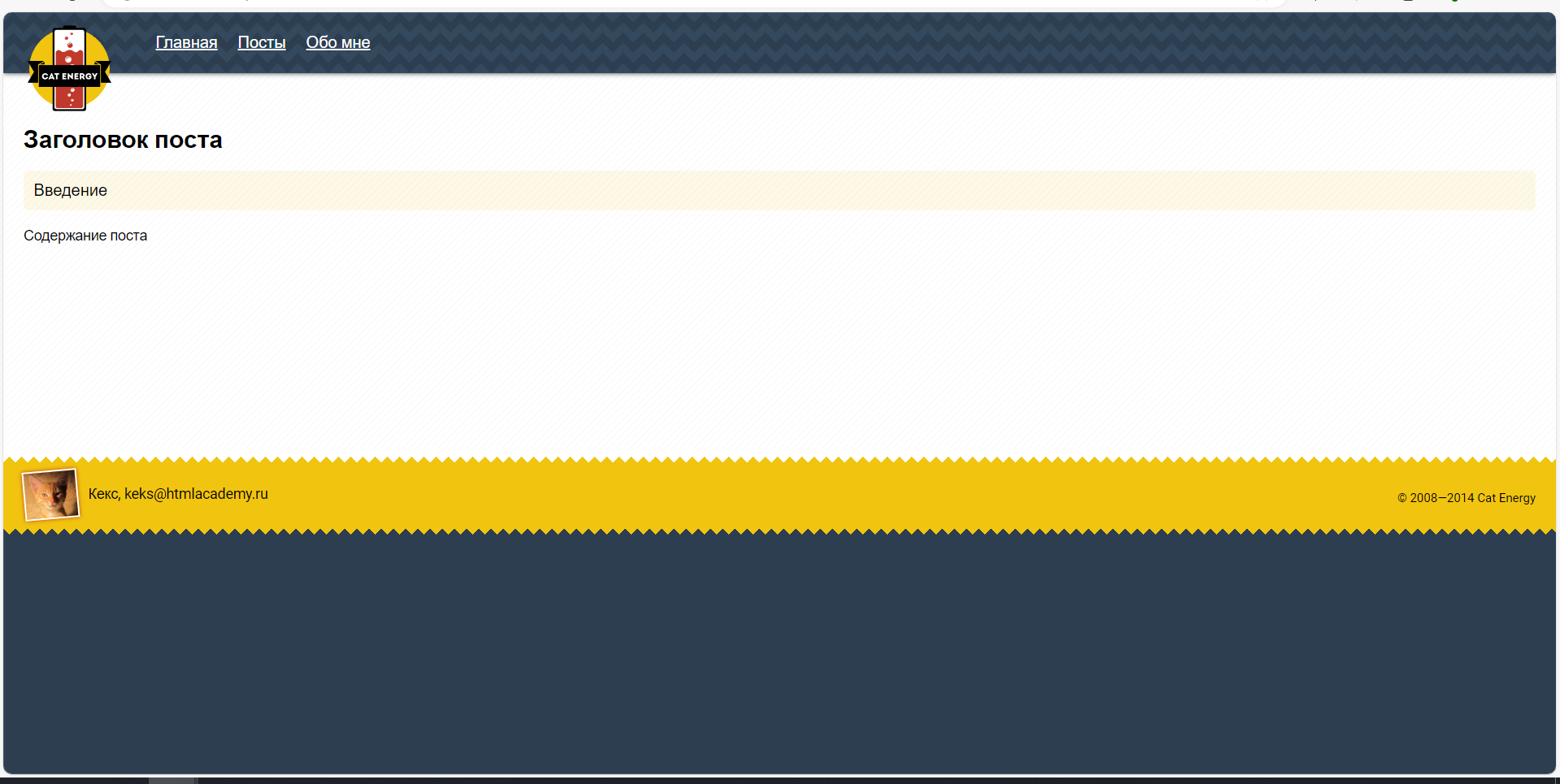
<div class="intro"> Введение</div>

<div class="content"> Содержание поста</div>

</article>

</main>





# 14. Тег time, дата и время для людей и машин

В этом задании мы добавим в пост дату и наполним его настоящим содержанием.

В HTML есть специальный тег для обозначения даты и времени — <time>. С помощью <time> можно описывать даты одновременно и для человека, и для машины, например:

<time datetime="2014-04-20">Вчера</time> мы готовили тренажёр к публикации.

Браузер отображает только содержимое тега, а содержимое datetime не отображается. Человек увидит только слово «вчера», а машина прочитает атрибут и получит дату в нужном ей формате. И все довольны.

Подробно про <time> уже рассказывалось в [прошлой части](https://htmlacademy.ru/courses/301/run/15).

Задачи

1. Добавьте дату публикации в пост после заголовка. Значение даты для людей 21 апреля 2014, а для машин 2014-04-21T11:00:00.

РЕЗУЛЬТАТ

<time datetime="2014-04-21T11:00:00">21 апреля 2014</time>

1. ДОБАВИТЬ ТЕКСТ

<main>

<article class="post">

<h1>Чешуя на градиентах</h1>

<!-- Добавьте time тут -->

<div class="intro">

<!--

<p>В этом посте я расскажу как делать повторяющийся чешуйчатый фон на CSS-градиентах. Будем использовать градиенты в сочетании со свойством <code>background-size</code>.</p>

-->

</div>

<div class="content">

<!--

<h2>Подготовим контейнер</h2>

<p>Чтобы было нагляднее, зададим контейнеру исходные размеры и сплошной цвет фона. В этом контейнере в фоне будет располагаться наш чешуйчатый орнамент. Временно запретим повторение фона, чтобы видеть только наш фрагмент.</p>

<h2>Создадим фрагмент фона</h2>

<p>Фрагмент орнамента, или одну чешуйку, создадим с помощью множественных радиальных градиентов. Первый градиент создаст чешуйки одного цвета:</p>

<p><code>radial-gradient(circle at 50% 100%, white 71%, transparent 71%, transparent)</code></p>

<p>Но это ещё не полноценная чешуя, нужно добавить две чешуйки другого цвета. Для этого добавим два градиента выше первого:</p>

<p><code>radial-gradient(circle at 100% 150%, silver 40%, transparent 40%, transparent)</code></p>

<p><code>radial-gradient(circle at 0 150%, silver 40%, transparent 40%, transparent)</code></p>

<p>И ниже первого:</p>

<p><code>radial-gradient(circle at 100% 50%, silver 49%, transparent 50%, transparent)</code></p>

<p><code>radial-gradient(circle at 0 50%, silver 49%, transparent 50%, transparent)</code></p>

<h2>Размножим фон</h2>

<p>После этого зададим размер <code>50px 25px</code> фрагменту чешуи и размножим его на весь контейнер. Вот, что получилось:</p>

<div class="example"></div>

<p>Готовый код можете посмотреть в CSS-редакторе у класса <code>.example</code>.</p>

-->

</div>

</article>

</main>

1. Раскомментируйте код введения поста.
2. Раскомментируйте код содержания поста.

.post .example {

height: 150px;

background-color: silver;

background-image:

radial-gradient(circle at 100% 150%, silver 40%, transparent 40%, transparent),

radial-gradient(circle at 0 150%, silver 40%, transparent 40%, transparent),

radial-gradient(circle at 50% 100%, white 71%, transparent 71%, transparent),

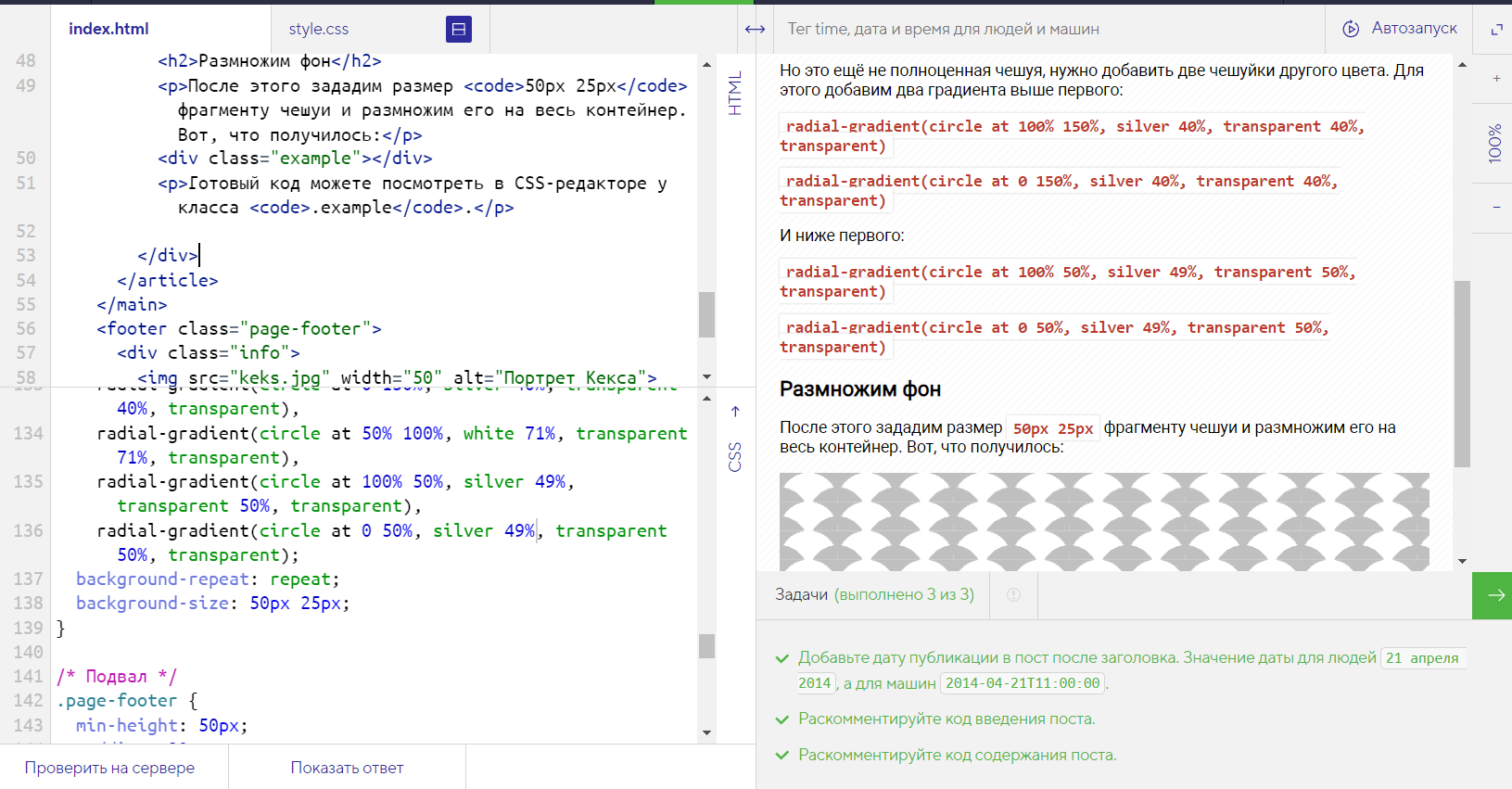
radial-gradient(circle at 100% 50%, silver 49%, transparent 50%, transparent),

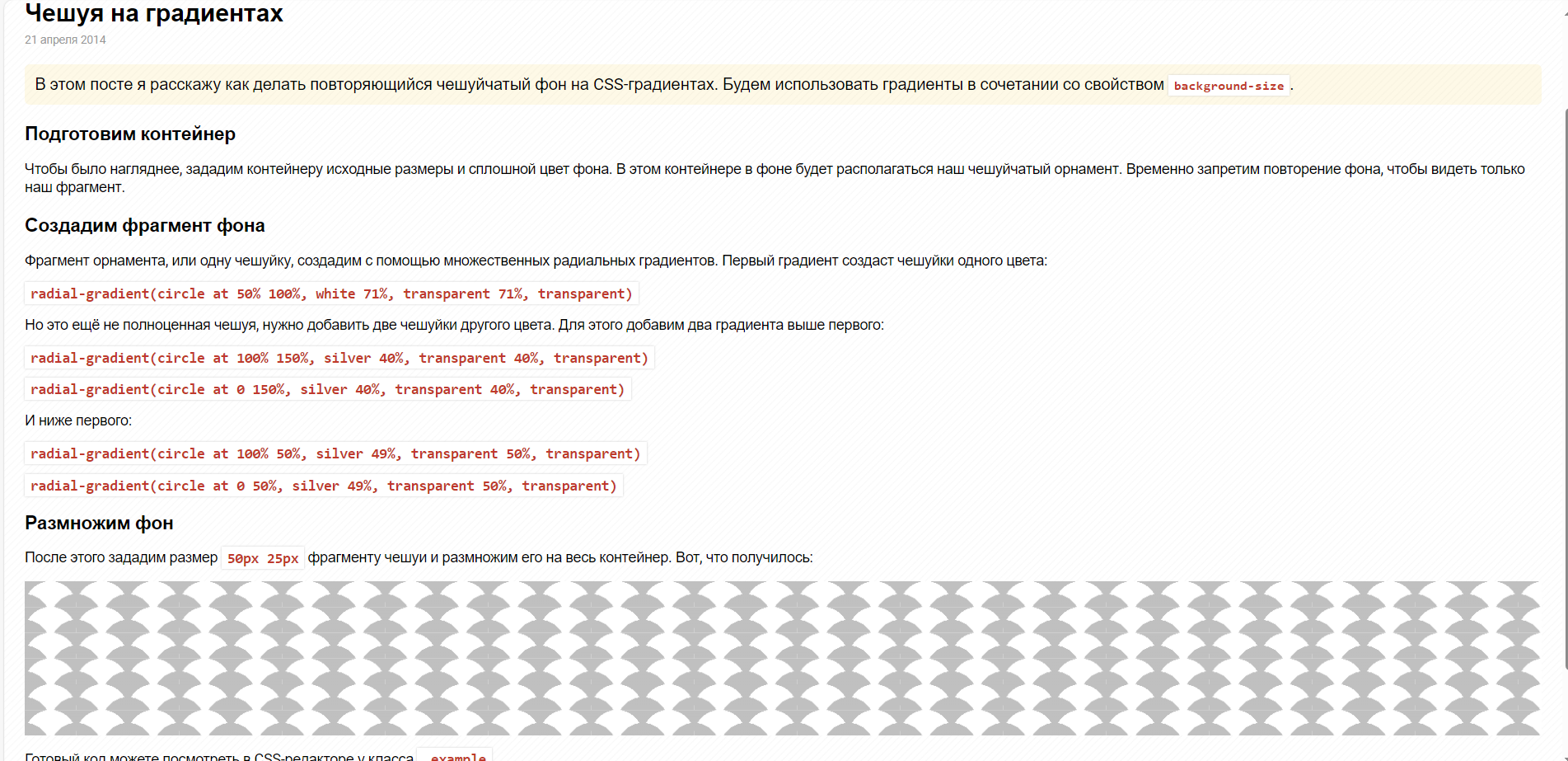
radial-gradient(circle at 0 50%, silver 49%, transparent 50%, transparent);

background-repeat: repeat;

background-size: 50px 25px;

}





# 15. Теги figure и figcaption, картинки с подписями

Теперь, когда шаблон поста готов, можно добавить в блог ещё несколько коротких записей. А начнём с фотопоста.

Для его разметки используем тег <figure> — он обозначает цельный и независимый блок содержания. Внутри этого тега размещают демонстрационный материал: изображения, схемы, куски кода и так далее.

Обычно каждый такой материал сопровождает разъясняющий комментарий или «легенда». Для обозначения этого комментария и предназначен ещё один новый тег — <figcaption>, который размещается первым или последним элементом внутри <figure>. Пример:

<figure>

схема,

график,

диаграмма

и так далее

<figcaption>Легенда</figcaption>

</figure>

Подробно про <figure> уже рассказывалось в [прошлой части](https://htmlacademy.ru/courses/305/run/14).

Мы используем этот тег для более прозаичных целей.

1. Добавим в index.html, для ссылки на создаваемую страницу.

<section class="profile">

                <p>Меня зовут Кекс. Я кот и веб-разработчик.</p>

                <p>

                    <small>

                        Живу в Санкт-Петербурге и работаю в HTML Academy консультантом по созданию учебных курсов.

                    </small>

                </p>

                <p><a href="figure.html">Теги figure и figcaption, картинки с подписям</a></p>

              </section>

2. Создадим новую страницу figure.html

<!DOCTYPE html>

<html lang="ru">

<head>

<title>Теги figure и figcaption, картинки с подписями</title>

<meta charset="utf-8">

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

</head>

<body>

<header class="page-header">

<div class="logo">

<img src="logo.svg" width="90" height="90" alt="Cat Energy">

</div>

<nav class="main-menu">

<ul>

<li><a href="#home">Главная</a></li>

<li><a href="#blog">Посты</a></li>

<li><a href="#me">Обо мне</a></li>

</ul>

</nav>

</header>

<main>

<article class="post">

<h1>Котфи на подоконнике</h1>

<time datetime="2014-04-15T11:00:00">

15 апреля 2014

</time>

<div class="intro">

<p>Все делают селфи, а я делаю котфи</p>

</div>

<div class="content">

<!-- содержание поста -->

</div>

</article>

</main>

<footer class="page-footer">

<div class="info">

<img src="keks.jpg" width="50" alt="Портрет Кекса">

Кекс, keks@htmlacademy.ru

</div>

<div class="copyright">

© 2008—2014 Cat Energy

</div>

</footer>

</body>

</html>

Задачи

1. Добавьте в содержание поста тег figure,
2. а внутрь него вставьте изображение с адресом catfy.jpg
3. и подпись с текстом Шерстяной наблюдатель
4. Посмотреть результат и добавить стили

.post figure {

position: relative;

width: 400px;

margin: 20px auto;

overflow: hidden;

border-radius: 5px;

box-shadow: 0 0 5px #999999;

}

.post figcaption {

position: absolute;

top: 0;

left: 0;

width: 90%;

padding: 15px 5%;

text-align: center;

background-color: rgba(255, 255, 255, 0.8);

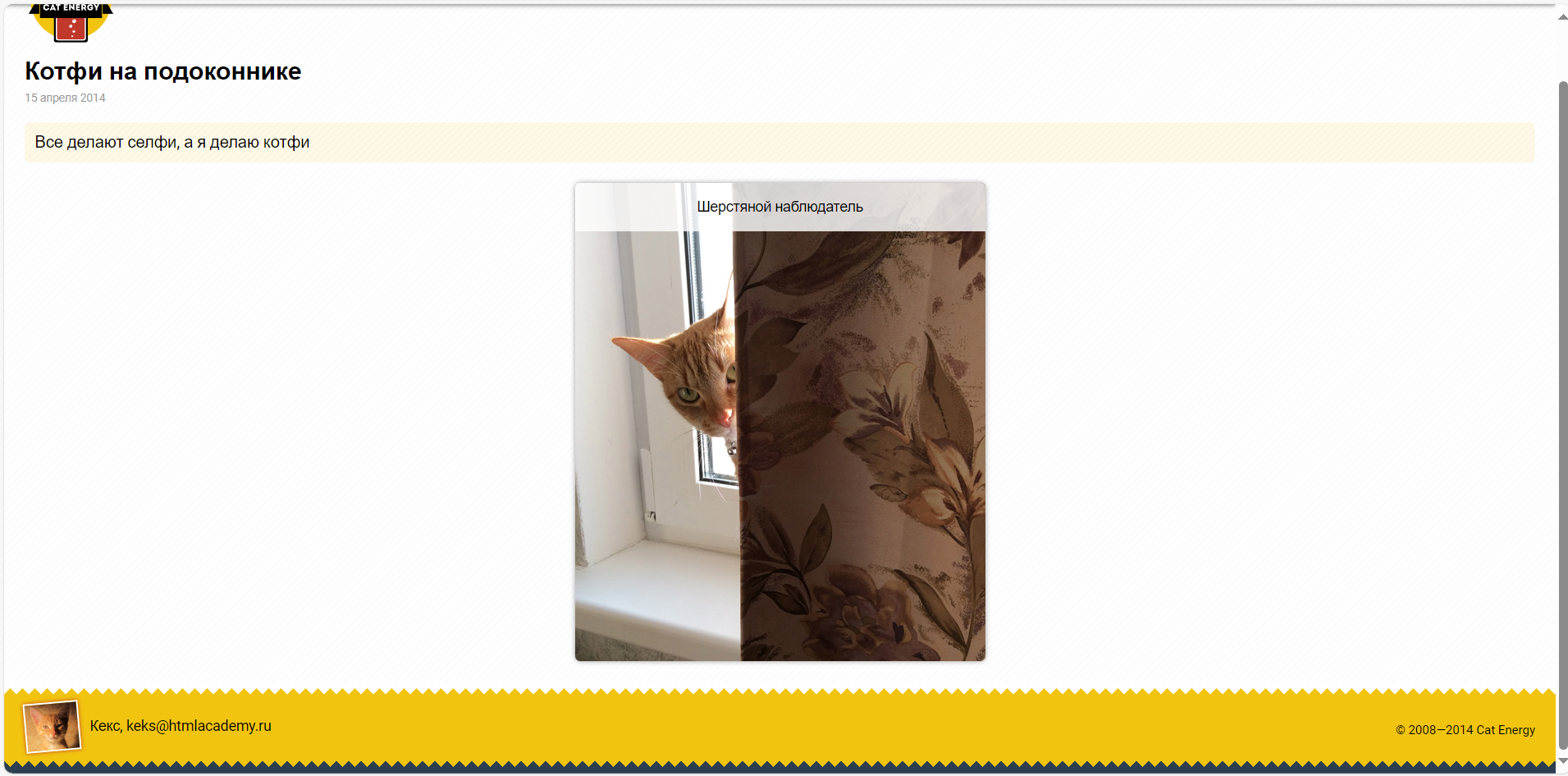
}

.post img {

display: block;

max-width: 100%;

}



# 16. Тег video, видео

В HTML с самого начала было очень просто вставлять изображения, но очень сложно видео и аудио. Сейчас эта проблема решена.

Для вставки видео предназначен специальный парный тег <video>. Его основные атрибуты:

|  |  |
| --- | --- |
| width и height | задают ширину и высоту видео |
| controls | пустой атрибут, при наличии которого отображается панель управления видео |
| preload | задаёт режим предзагрузки видео, имеет 3 возможных значения:   * none — не загружать ничего; * metadata — загрузить служебную мета-информацию (длительность, первый кадр и так далее); * auto — можно загрузить всё видео.   значение по умолчанию зависит от браузера |
| src | задаёт адрес видеофайла |
| autoplay | пустой атрибут, при наличии которого воспроизведение видео начинается автоматически |
| poster | задаёт адрес картинки-обложки, которая отображается, когда видео ещё не загрузилось или не воспроизводится |

1. В index.html добавим еще один пост со ссылкой на новую страницу.

<section class="posts">

            <h2>Последние посты</h2>

            <article class="featured-post">

                <h3><a href="post-1.html">Чешуя на градиентах</a></h3>

                <p>В этом посте я расскажу как делать повторяющийся чешуйчатый фон на CSS-градиентах. Будем использовать градиенты в сочетании со свойством <code>background-size</code>.</p>

                <article class="featured-post">

                    <h3><a href="post-2.html">UX-тестирование</a></h3>

                    <p>В этом посте я расскажу про UX-тестирование.</p>

            </section>

2. Создадим эту страницу post-2.html

<!DOCTYPE html>

<html lang="ru">

<head>

<title>Тег video, видео</title>

<meta charset="utf-8">

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

</head>

<body>

<header class="page-header">

<div class="logo">

<img src="logo.svg" width="90" height="90" alt="Cat Energy">

</div>

<nav class="main-menu">

<ul>

<li><a href="#home">Главная</a></li>

<li><a href="#blog">Посты</a></li>

<li><a href="#me">Обо мне</a></li>

</ul>

</nav>

</header>

<main>

<article class="post">

<h1>UX-тестирование</h1>

<time datetime="2014-04-10T15:25:00">

10 апреля 2014

</time>

<div class="intro">

<p>Записали процесс тестирования пользовательских интерфейсов. Очень рекомендую к просмотру.</p>

</div>

<div class="content">

<!-- содержание поста -->

</div>

</article>

</main>

<footer class="page-footer">

<div class="info">

<img src="keks.jpg" width="50" alt="Портрет Кекса">

Кекс, keks@htmlacademy.ru

</div>

<div class="copyright">

© 2008—2014 Cat Energy

</div>

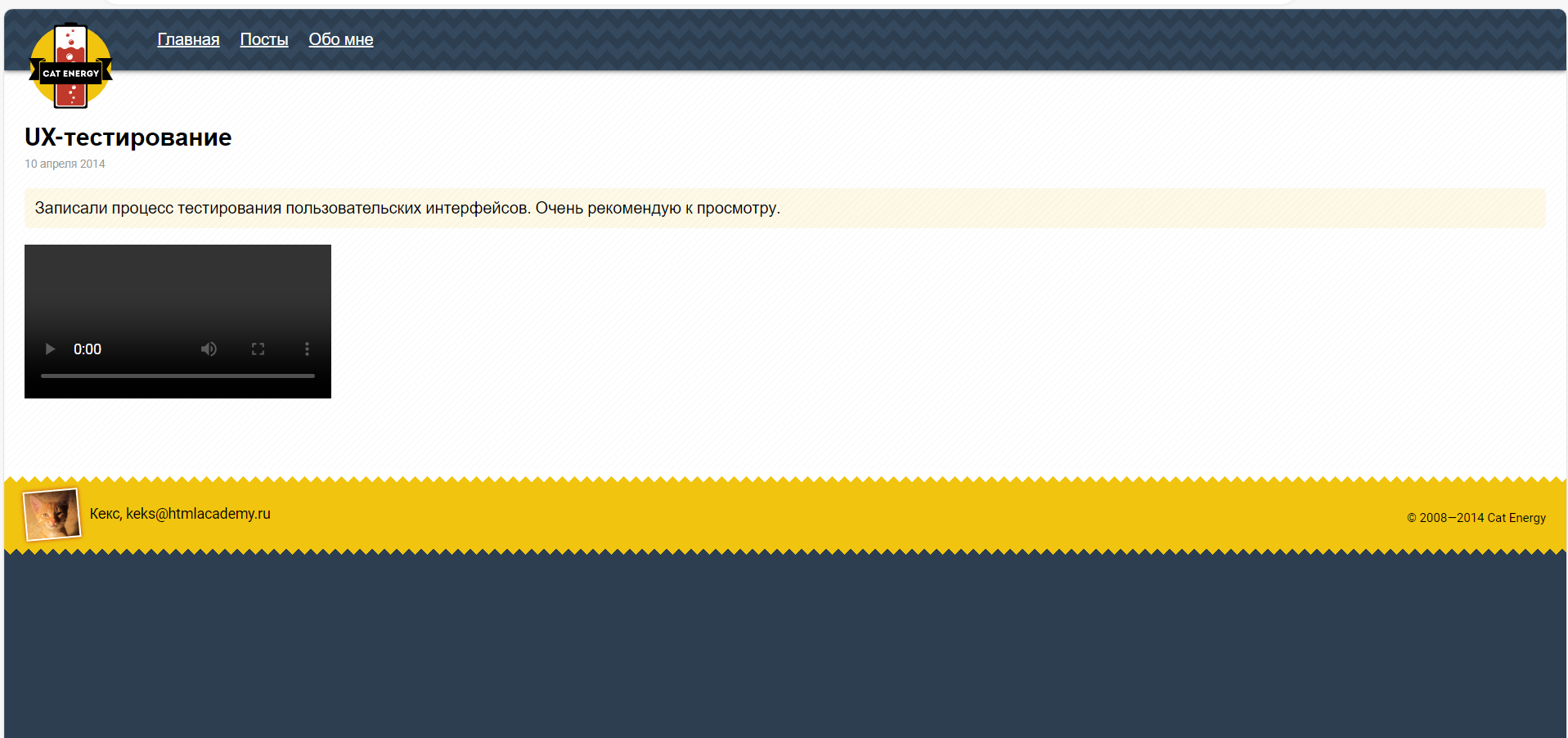
</footer>

</body>

</html>

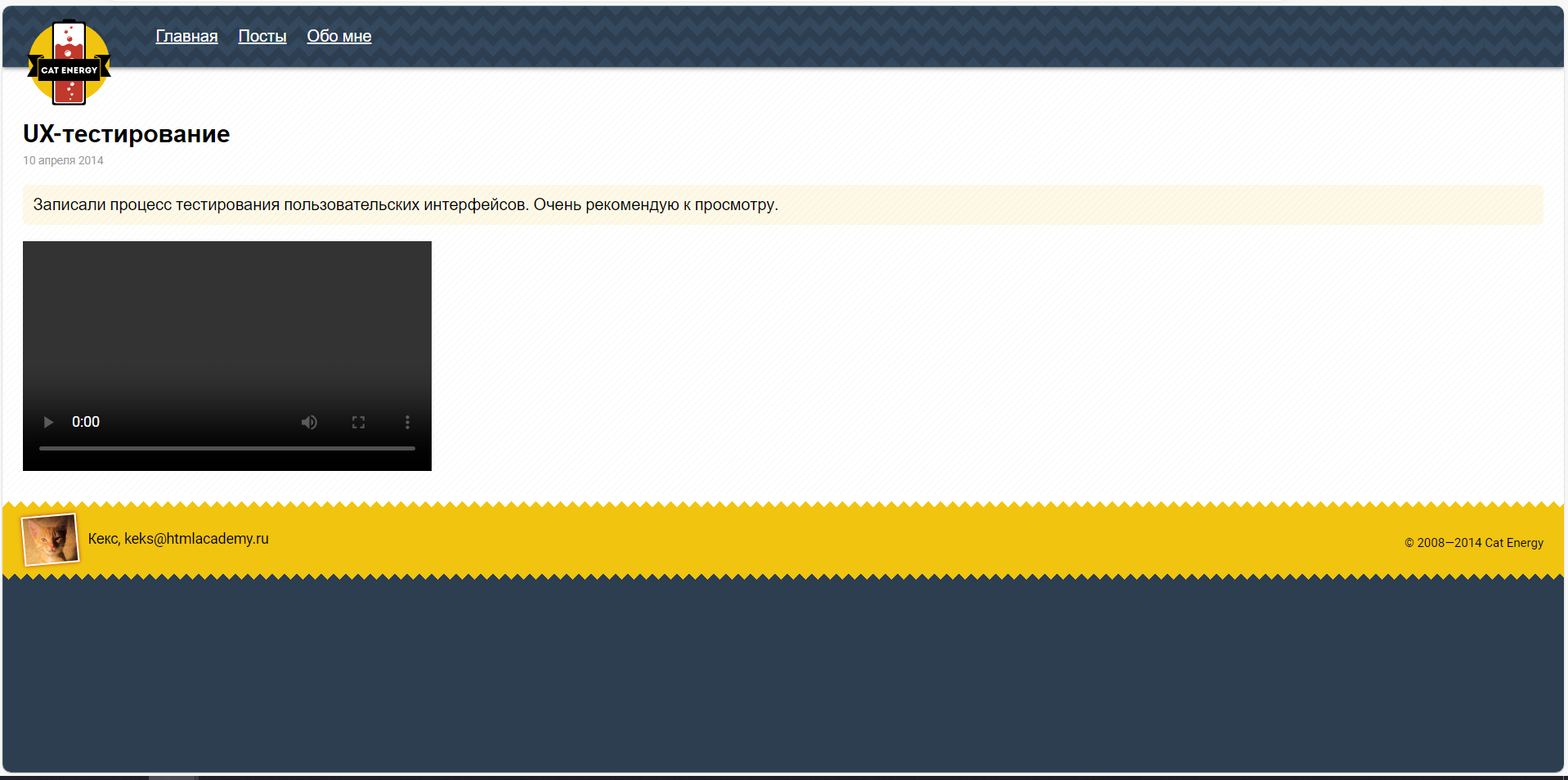
Задачи

1. Добавьте в содержание поста тег video.
2. Включите у видео управляющую панель,



1. включите предзагрузку мета-информации,
2. и задайте видеоролику ширину 400 и высоту 225 пикселей.

<video controls preload="metadata" width="400" height="225"></video>



# 17. Форматы и источники видео

У тега <video> есть атрибут src, в котором можно указать адрес видеофайла, но мы почему-то его не использовали.

Всё дело в том, что в текущий момент существует несколько форматов видео, каждый из которых хорошо поддерживается лишь некоторыми браузерами. Вот три самых распространённых формата и их поддержка:

* [MPEG-4/H.264](http://caniuse.com/mpeg4)
* [OGG/Theora](http://caniuse.com/ogv)
* [WebM](http://caniuse.com/webm)

Поэтому мы должны в видео указывать адреса файлов во всех этих форматах (и конвертировать исходное видео в эти форматы, конечно). Делается это с помощью тегов <source>:

<video controls>

<source src="video.mp4" type="video/mp4">

<source src="video.ogv" type="video/ogg">

<source src="video.webm" type="video/webm">

</video>

В атрибуте src указывается адрес видеофайла, а в атрибуте type его тип (также там могут указываться и кодеки). Браузер из списка видеофайлов выбирает первый, который может проиграть и загружает его.

Атрибут type не является обязательным, так как браузер умеет сам определять тип и кодеки, но указывая тип явно, мы помогаем ему не ошибиться.

Задачи

1. Добавьте в видео первый источник с типом video/mp4 и адресом uxtesting.mp4
2. второй с типом video/ogg и адресом uxtesting.ogv
3. третий с типом video/webm и адресом uxtesting.webm

<!-- источники видео -->

<source src="uxtesting.mp4" type="video/mp4">

<source src="uxtesting.ogv" type="video/ogg">

<source src="uxtesting.webm" type="video/webm">

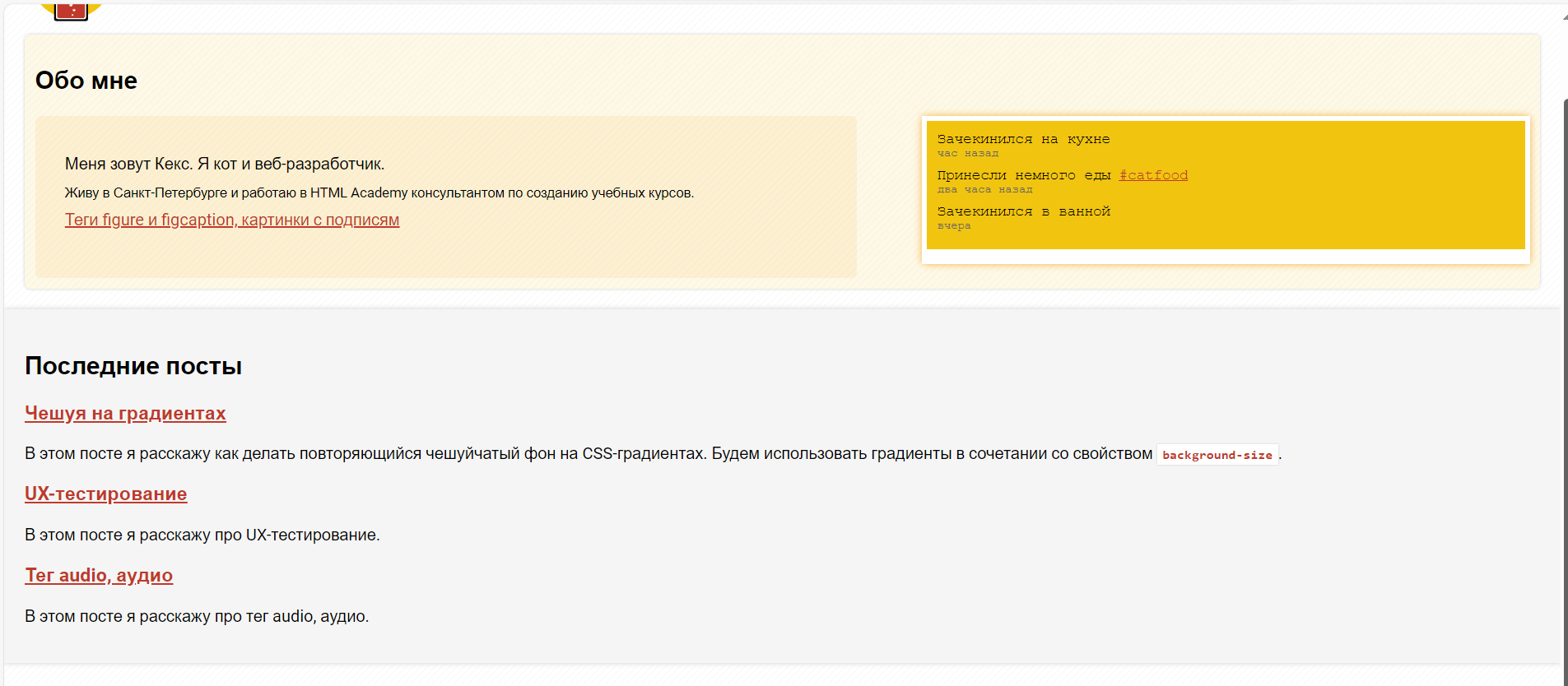
Видео должно быть сохранено

# 18. Тег audio, аудио

Работа с аудио в HTML очень похожа на работу с видео, только у тега для аудио чуть меньше параметров.

Для вставки звука предназначен тег <audio>. Его основные атрибуты:

|  |  |
| --- | --- |
| controls | пустой атрибут, при наличии которого отображается панель управления проигрывателем |
| preload | задаёт режим предзагрузки аудио, имеет 3 возможных значения:   * none — не загружать ничего; * metadata — загрузить служебную мета-информацию; * auto — можно загрузить весь файл.   значение по умолчанию зависит от браузера |
| src | задаёт адрес аудиофайла |
| autoplay | пустой атрибут, при наличии которого воспроизведение звука начинается автоматически |



1. В index.html добавим еще один пост со ссылкой на новую страницу.

<section class="posts">

            <h2>Последние посты</h2>

            <article class="featured-post">

                <h3><a href="post-1.html">Чешуя на градиентах</a></h3>

                <p>В этом посте я расскажу как делать повторяющийся чешуйчатый фон на CSS-градиентах. Будем использовать градиенты в сочетании со свойством <code>background-size</code>.</p>

                <article class="featured-post">

                    <h3><a href="post-2.html">UX-тестирование</a></h3>

                    <p>В этом посте я расскажу про UX-тестирование.</p>

                <article class="featured-post">

                    <h3><a href="post-3.html">Тег audio, аудио</a></h3>

                    <p>В этом посте я расскажу про тег audio, аудио.</p>

            </section>

2. Создадим эту страницу post-3.html

<!DOCTYPE html>

<html>

<head>

<title>Тег audio, аудио</title>

<meta charset="utf-8">

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

</head>

<body>

<header class="page-header">

<div class="logo">

<img src="logo.svg" width="90" height="90" alt="Cat Energy">

</div>

<nav class="main-menu">

<ul>

<li><a href="#home">Главная</a></li>

<li><a href="#blog">Посты</a></li>

<li><a href="#me">Обо мне</a></li>

</ul>

</nav>

</header>

<main>

<article class="post">

<h1>Музыка для души</h1>

<time datetime="2014-04-05T15:25:00">

5 апреля 2014

</time>

<div class="intro">

<p>Звуки леса, травы и что-то электронное</p>

</div>

<div class="content">

<!-- содержание поста -->

</div>

</article>

</main>

<footer class="page-footer">

<div class="info">

<img src="keks.jpg" width="50" alt="Портрет Кекса">

Кекс, keks@htmlacademy.ru

</div>

<div class="copyright">

© 2008—2014 Cat Energy

</div>

</footer>

</body>

</html>

Задачи

1. Добавьте в содержание поста тег <audio>,
2. включите у него управляющую панель
3. и предзагрузку мета-информации.

<audio controls preload="metadata"></audio>



# 19. Форматы и источники звука

У звуковых файлов с поддержкой форматов дела обстоят лучше, чем у видео.

Для охвата большинства современных браузеров, достаточно использовать всего два формата:

* MP3
* OGG

И снова мы не можем указать только один файл в атрибуте src у тега <audio>. Мы должны так же, как и в случае с видео, перечислить адреса звуковых файлов в разных форматах с помощью тегов <source>:

<audio controls>

<source src="sound.mp3" type="audio/mpeg">

<source src="sound.oga" type="audio/ogg">

</audio>

Теги <source> в аудио работают так же, как и в видео.

Задачи

1. Добавьте в аудио первый источник с типом audio/mpeg и адресом sound.mp3,
2. второй с типом audio/ogg и адресом sound.oga

<div class="content">

<audio controls preload="metadata">

<!-- источники звука -->

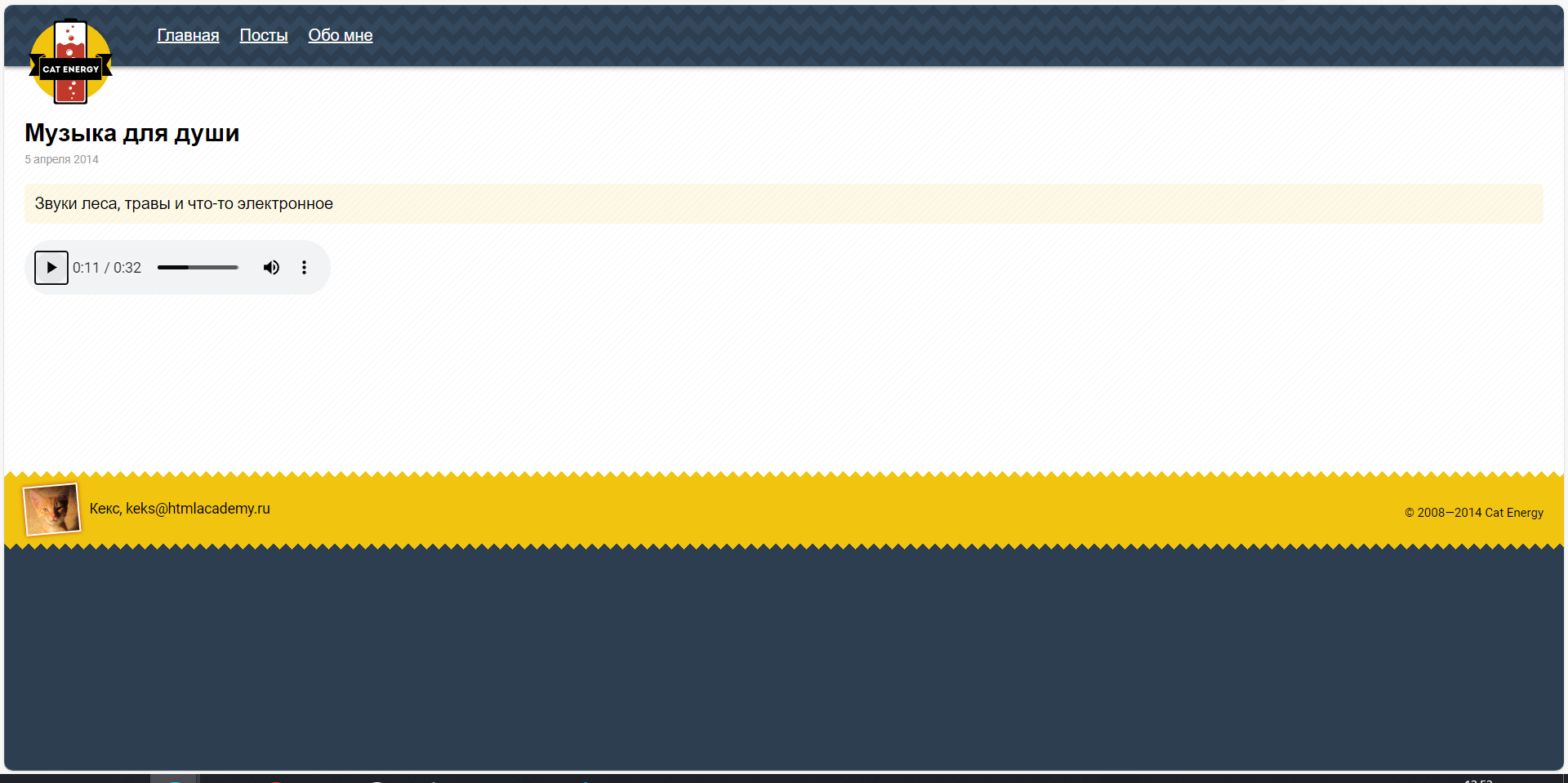
<source src="sound.mp3" type="audio/mpeg">

<source src="sound.oga" type="audio/ogg">

</audio>

</div>

Файл со звуком должен быть загружен.



# 20. Конспект «Возможности HTML и CSS»

## HTML5

Чтобы использовать HTML5, нужно задать правильный тип документа:

<!DOCTYPE html>

## header и footer

<header> — это не только привычная шапка сайта с логотипом и меню, но и «шапка» какой-нибудь статьи или раздела сайта, в которой могут содержаться заголовки, оглавление и так далее.

<footer> — это подвал сайта: с копирайтами, контактной информацией и так далее. Но может использоваться и в других разделах сайта, например в статье, чтобы разместить дополнительную информацию: данные об авторе, дополнительные ссылки и так далее.

## main

Тег <main> обозначает основное содержание сайта и по спецификации может использоваться на странице только один раз.

## article и section

Где и когда использовать разные контейнеры:

* <div> — контейнер общего назначения, не обязательно смысловой. Дивы используются для разметки мелких блоков, создания сетки и декоративных эффектов.
* <section> — более крупный логический контейнер, объединяющий содержание по смыслу. Например, блок «О компании», список товаров, раздел личной информации в профиле и так далее.
* <article> — самостоятельный, цельный и независимый раздел документа. Этот раздел можно в неизменном виде использовать в различных местах, в том числе и на других сайтах. Примеры: статья, пост в блоге, сообщение на форуме и так далее.

## nav

Тег <nav> — специальный элемент для навигационных блоков. Меню и блоки ссылок лучше оборачивать не в дивы, а в навы.

## aside

<aside> — это дополнительное содержание, не связанное напрямую с основным. Ещё такие блоки часто называют «сайдбарами» или боковыми панелями.

## time

<time> — специальный тег для обозначения даты и времени. У этого тега есть атрибут datetime — для указания даты в машиночитаемом формате ISO 8601. Содержимое datetime не отображается.

<time datetime="2014-04-20">Вчера</time> мы готовили тренажёр к публикации.

## figure и figcaption

<figure> — цельный и независимый блок содержания. Внутри этого тега размещают демонстрационный материал: изображения, схемы, куски кода и так далее.

<figcaption> — разъясняющий комментарий, или «легенда». Размещается первым или последним элементом внутри <figure>. Пример:

<figure>

схема,

график,

диаграмма

и так далее

<figcaption>Легенда</figcaption>

</figure>

## video

Парный тег <video> предназначен для вставки видео. Его основные атрибуты:

* width и height — задают ширину и высоту видео;
* controls — пустой атрибут, при наличии которого отображается панель управления видео;
* preload — задаёт режим предзагрузки видео, имеет 3 возможных значения:
  + none — не загружать ничего;
  + metadata — загрузить служебную мета-информацию (длительность, первый кадр и так далее);
  + auto — можно загрузить всё видео.
* src — задаёт адрес видеофайла;
* autoplay — пустой атрибут, при наличии которого воспроизведение видео начинается автоматически;
* poster — задаёт адрес картинки-обложки, которая отображается, когда видео ещё не загрузилось или не воспроизводится.

Разные браузеры поддерживают разные форматы, поэтому лучше указывать адреса файлов в нескольких форматах. Делается это с помощью тегов <source>:

<video controls>

<source src="video.mp4" type="video/mp4">

<source src="video.ogv" type="video/ogg">

<source src="video.webm" type="video/webm">

</video>

В атрибуте src указывается адрес видеофайла, а в атрибуте type его тип (также там могут указываться и кодеки). Браузер выбирает из списка видеофайлов первый, который может проиграть, и загружает его.

## audio

Тег <audio> предназначен для вставки звука. Его основные атрибуты:

* controls — пустой атрибут, при наличии которого отображается панель управления проигрывателем;
* preload — задаёт режим предзагрузки аудио, имеет 3 возможных значения:
  + none — не загружать ничего;
  + metadata — загрузить служебную мета-информацию;
  + auto — можно загрузить весь файл.
* src — задаёт адрес аудиофайла;
* autoplay — пустой атрибут, при наличии которого воспроизведение звука начинается автоматически.

Лучше указывать адреса файлов в нескольких форматах. Для этого используют теги <source>:

<audio controls>

<source src="sound.mp3" type="audio/mpeg">

<source src="sound.oga" type="audio/ogg">

</audio>

## Шрифты

Есть возможность подключать и использовать на странице любые нестандартные шрифты. Подключение веб-шрифтов производится с помощью CSS-правила @font-face. Например:

@font-face {

font-family: "Roboto";

src:

local("Roboto Regular"),

url("roboto.woff") format("woff");

}

Нужно указать название шрифта, которое вы будете использовать в font-family, и источники, из которых браузер сможет загрузить шрифт. Сначала указывают название шрифта в системе, чтобы браузер попытался найти его локально, а затем указывают адрес файла шрифта в интернете.

Если шрифт очень редкий и его нет ни в одном из шрифтовых сервисов, поступают так:

1. Берут файл шрифта (например, .ttf) и конвертируют в веб-формат в сервисе наподобие [Font Squirrel](http://www.fontsquirrel.com/tools/webfont-generator) (для кириллических шрифтов надо указать дополнительные параметры конвертации).
2. Затем сконвертированные файлы шрифта размещают у себя на сервере.
3. И подключают шрифт с помощью @font-face.

## SVG

SVG — векторный формат, который хорошо поддерживается браузерами. Векторные изображения можно уменьшать и увеличивать без потери качества.

# 21. Испытание: другой вариант главной

Доверстайте второй вариант главной страницы блога, используя новые теги, изученные в тренажёре.

Вёрстка практически готова. Не хватает только некоторых тегов. Вам нужно использовать подходящие теги с правильными классами.

В качестве подсказки используйте селекторы в CSS-коде.