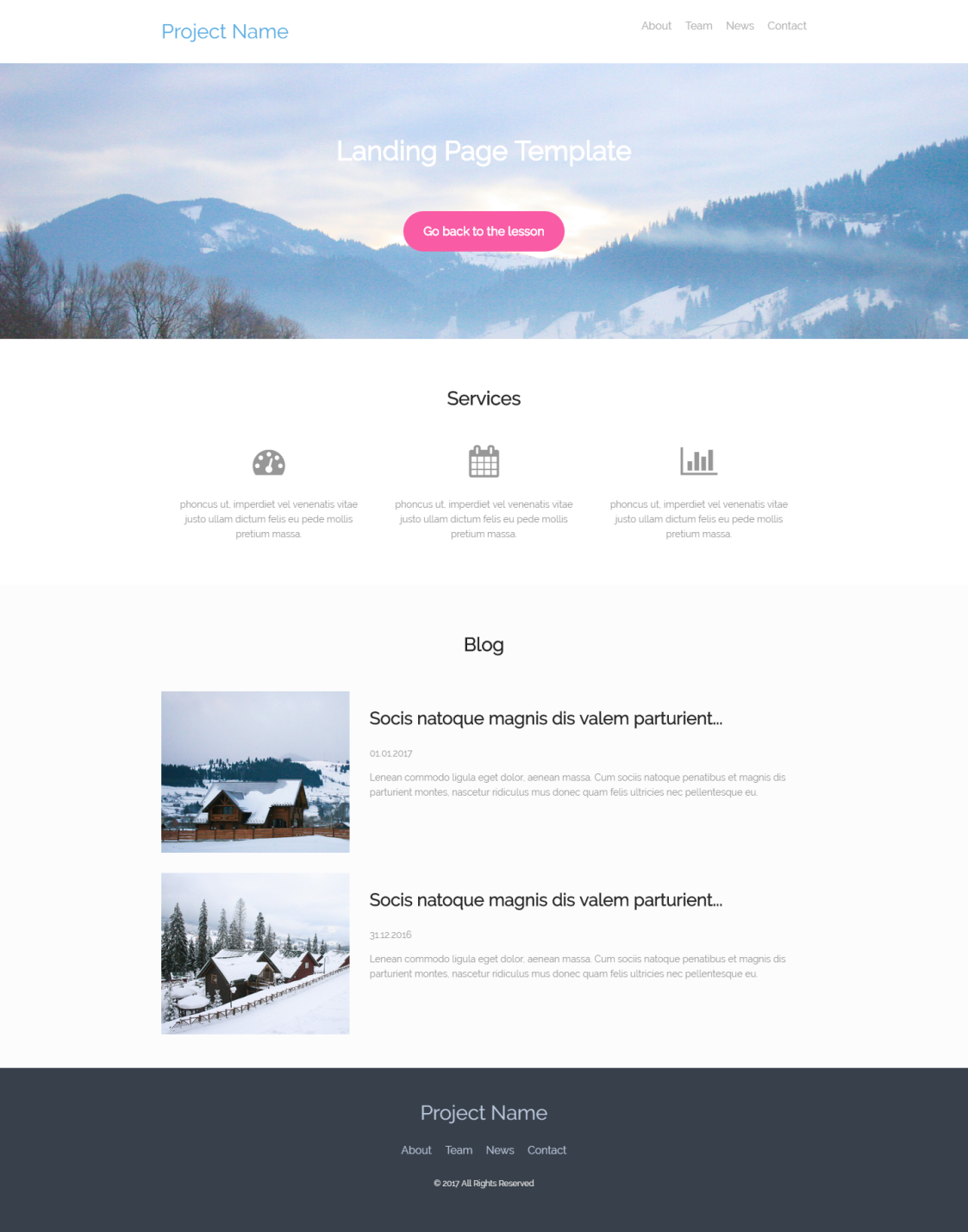
**Практическая работа Создание адаптивной лендинг страницы**

Открыв HTML-документ из архива, вы увидите готовый каркас посадочной страницы. В теге <head> подключины таблицы стилей, которые понадобятся для работы:

* normalize.css — файл для сброса стандартных стилей браузера;
* шрифт Raleway от Google Fonts;
* иконочный шрифт FontAwesome;
* style.css — файл, куда вы будете записывать все стили для лендинга.

Тело документа разделено на секции — шапка сайта (header) с навигацией (nav), большой блок на всю ширину (jumbotron), блок сервисов (services), записи блога (blog) и подвал сайта (footer). Каждая секция содержит соответствующий контент.

**Макет страницы выглядит следующим образом:**



**Таблица стилей**

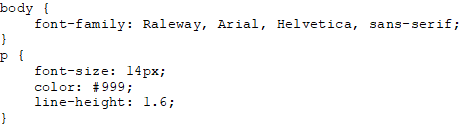
Откройте файл style.css из папки «css». Документ структурирован с помощью комментариев — для удобства разделен на части в соответствии со структурой HTML-страницы. Записывайте каждый стиль в соответствующий раздел: стили для шапки сайта помещайте под комментарием /\* Header \*/, стили для блога — под комментарием /\* Blog \*/, базовые стили и стили для повторяющихся элементов страницы — под комментарием /\* General \*/ и т. д.

Сначала объявим несколько основных стилей для тегов и классов, которые будут использоваться постоянно.

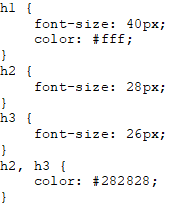
**Основные стили CSS**

На странице есть повторяющиеся элементы, а также теги, которым требуется задать определенный стиль, после чего он будет применен ко всем элементам сразу.

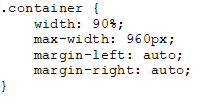
Начните с основного текста на сайте. Согласно дизайну макета, основной шрифт на странице — это Raleway, а все параграфы <p> должны иметь размер шрифта 14 пикселей, цвет #999 и высоту строки 1.6. Запишите эти данные в таблицу стилей, в раздел General:



Задайте стили для заголовков. На странице есть один заголовок <h1>, а также несколько заголовков <h2> и <h3>. Согласно макету, заголовок h1 имеет размер шрифта 40 пикселей и белый цвет. Остальные заголовки имеют цвет #282828. Размер шрифта для заголовков второго уровня — 28 пикселей, а для h3 — 26 пикселей.



Помимо текста есть еще несколько повторяющихся элементов. Например, класс .container — это блок, с ограниченной шириной в который помещается контент. Ширина нашего контента должна занимать 90% экрана, но при этом контейнер не должен растягиваться шире, чем 960 пикселей. Здесь используйте свойство max-width. Кроме того, этот контейнер должен размещаться по центру страницы, для чего воспользуйтесь свойством margin.



Следующие повторяющиеся элементы — блок с классом .brand и навигационное меню. Эти элементы присутствуют в хедере и футере. Они не будут выглядеть абсолютно идентично, однако часть стилей для них будут общими.

Оба элемента .brand будут иметь одинаковый размер шрифта:

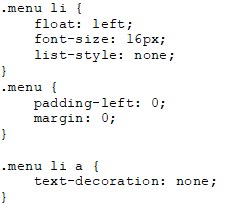
.brand {

font-size: 30px;

}

А оба навигационных меню будут горизонтальными и тоже будут иметь одинаковый размер шрифта. Для создания горизонтального меню можно использовать свойство float, применив его к пунктам меню:

.menu li {



Сохраните изменения в таблице CSS и обновите файл index.html в браузере посмотрите изменения.

**CSS для шапки сайта**

Хедер будет иметь белый фон и отступ сверху в размере 30 пикселей. Запиште это в таблицу стилей:

header {

background-color: #fff;

padding-top: 30px;

}

Блок с названием проекта будет прилеплен к левой стороне контейнера, иметь цвет #59abe3 и отступ внизу в 30 пикселей:

header .brand {

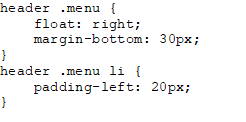
float: left;

color: #59abe3;

margin-bottom: 30px;

}

Навигационное меню приклеим к правой стороне шапки, добавьте внизу поле высотой 30 пикселей, а также сделайте отступы по 20 пикселей слева для всех пунктов меню, чтобы они не сливались в одно целое:



Установите цвет текста #aaa для пунктов меню.

header .menu li a {

color: #aaa;

}

Обновите страницу в браузере для просмотра изменений.

**CSS для блока Jumbotron**

Jumbotron (джамботрон) на сайте — это большой, привлекающий внимание блок, растянутый на всю ширину страницы. Фоном джамботрона послужит изображение, растянутое на всю ширину, а текстовое содержимое будет находиться в ограничивающем блоке .container.

Фоновое изображение находится в папке «img».

.jumbotron {

background-image: url(../img/jumbotron.jpg);

background-size: cover;

background-position: top center;

background-repeat: no-repeat;

}

Добавьте к стилю джамботрона — для создания вертикальных отступов по 80 пикселей и установки минимальной высоты в 250 пикселей, для выравнивания внутреннего содержимого по центру блока, а также для сброса обтекания, которое было задано навигационному меню:

.jumbotron {

background-image: url(../img/jumbotron.jpg);

background-size: cover;

background-position: top center;

background-repeat: no-repeat;

padding-top: 80px;

padding-bottom: 80px;

min-height: 250px;

text-align: center;

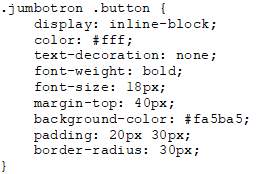
clear: both;

}

Перейдем к созданию яркой кнопки, расположенной под джамботроном

Во-первых, поскольку тег <a> является строчным элементом, то первым делом применим к нему свойство display со значением inline-block. Таким образом кнопка будет отображаться как блок, а поведение останется, как у строки.

Текст кнопки будет белым и без подчеркивания, шрифт — жирный 18 пикселей, верхний margin — 40 пикселей, а цвет фона — #22a7f0. Чтобы это больше было похоже на кнопку, необходимо добавить отступы с четырех сторон — в нашем случае по 20 пикселей вертикально и по 30 пикселей горизонтально. Также мы округлим углы элемента с помощью свойства border-radius:



**CSS для блока Services**

Данная секция будет включать в себя три блока, расположенных в один ряд. Для этого можно использовать свойство float. Все блоки будут иметь одинаковую ширину, указанную в процентах, и в сумме они займут всю ширину контейнера. Разделив число 100 на 3, получаем 33,333333. Именно его мы записываем в стиль для класса .service. Дополнительно зададим каждому блоку внутренние отступы в размере 15 пикселей и внешние вертикальные отступы в 30 и 50 пикселей:

.services .service {

width: 33.333333%;

float: left;

padding-left: 15px;

padding-right: 15px;

margin-top: 30px;

margin-bottom: 50px;

}

Сейчас вы можете обновить страницу и увидеть, как третий блок съехал вниз. Догадаетесь, почему так произошло? Мы добавили нашим блокам свойство padding, и сумма их ширин превысила ширину контейнера, поэтому один из них не помещается в ряд. Добавьте к стилю элемента .service свойство box-sizing со значением border-box, после чего обновите страницу в браузере. Вы увидите, что теперь все блоки выстроились так, как надо.

Добавим стили для самого блока .services — сделаем верхний отступ в 50 пикселей и центрируем всё текстовое содержимое:

.services {

padding-top: 50px;

text-align: center;

}

В качестве иконок блоков мы использовали иконочный шрифт FontAwesome. Мы не будем углубляться в эту тему (на официальном сайте FontAwesome есть очень простая и понятная инструкция по использованию), но мы изменим цвет иконок на более подходящий нам. Каждая иконка представляет собой векторный рисунок, перекрасить который не составит труда. Мы просто запишем стиль, как будто задаем цвет для текста:

.services .fa {

color: #979797;

margin-bottom: 15px;

}

Обновите страницу в браузере и посмотрите результатом.

**CSS для блока Blog**

Прежде всего необходимо отменить обтекание для блока .blog. Он будет иметь вертикальные внутренние отступы в 50 пикселей, а также легкий серый фон, в соответствии с макетом:

.blog {

clear: both;

padding-top: 50px;

padding-bottom: 50px;

background-color: #fcfcfc;

}

Выровняем заголовок этого блока по центру:

.blog h2 {

text-align: center;

}

Каждому блоку с классом .post присвоим внутренний отступ в 30 пикселей сверху и поведение строчно-блочного элемента:

.blog .post {

padding-top: 30px;

display: inline-block;

}

Для миниатюр постов зададим обтекание и внешний отступ 30 пикселей с правой стороны:

.blog .post img {

float: left;

margin-right: 30px;

}

После чего дополните стиль элементов .post строкой clear:both, благодаря которой каждый блок будет отображаться корректно и не съезжать.

**CSS для футера**

Для основного блока подвала понадобится сбросить обтекание, задать вертикальные отступы в 50 пикселей, выравнивание текста по центру и темно-серый фон:

footer {

clear: both;

padding-top: 50px;

padding-bottom: 50px;

text-align: center;

background-color: #38404b;

}

Установим цвет текста для элемента .brand (он будет отличаться от стиля, применяемого к элементу .brand в шапке сайта):

footer .brand {

color: #bcc9dd;

}

Для пунктов меню в футере зададим отступы и цвет ссылок:

footer .menu li {

padding-left: 10px;

padding-right: 10px;

}

footer .menu li a {

color: #bcc9dd;

}

**Обратите внимание:** чтобы свойство text-align:center, примененное к тегу <footer>, повлияло на блок с меню, необходимо сделать его строчно-блочным:

footer .menu {

display: inline-block;

margin-top: 30px;

margin-bottom: 15px;

}

Напишем небольшой стиль для последнего элемента — .copyright:

footer .copyright {

font-size: 12px;

color: #eee;

}

Теперь обновите страницу и посмотритена результат работы. Если вы всё сделали без ошибок, то у вас получилась веб-страница, которую вы видели на макете

**Проверка адаптивности**

Проверьте, насколько верстка хорошо адаптируется под разные размеры экрана. Измените ширину окна браузера и понаблюдайте за поведением элементов. Вы обнаружили несколько недочетов.

Три блока .service, каждый из которых имеет ширину 33,333333% от ширины контейнера. На узких экранах текст в этих блоках становится трудночитаемым.

Это можно исправить, использовав медиазапрос. В медиазапросе можно указать, при какой ширине экрана какой стиль отображать.

Блоки начинают выглядеть плохо на экранах шириной менее 576 пикселей. Запишите специальный стиль для данной ширины (поместите эту запись в самый низ таблицы стилей):

@media (max-width: 575px) {

.services .service {

width: 100%;

float: none;

}

}

Теперь, если ширина экрана пользователя будет менее чем 576 пикселей, каждый блок .service будет иметь ширину 100% и не иметь обтекания.

На маленькие экраны нормально не помещаются блоки .post. Миниатюра поста чересчур большая для небольшого экрана и следующий за ней текст некрасиво обтекает ее. Исправьте это следующим стилем (просто добавьте код к уже созданному медиазапросу):

.blog .post img {

float: none;

display: block;

margin-right: auto;

margin-left: auto;

}

.blog .post {

max-width: 280px;

margin-left: auto;

margin-right: auto;

display: block;

}

Добавьте еще строки кода, которые помогут всем изображениям на странице стать адаптивными. Внимание: этот стиль добавляем в раздел General!

img {

display: block;

max-width: 100%;

height: auto;

}