Landing Page Piedpiper

Ссылка на заготовку

Посмотрим структуру заготовки.

- Файл **index.html,** с разметкой приложения.
- Файлы со стилями:
  - **style.css** — главный рабочий файл со стилями веб-страницы.
  - **overlay.css** — отдельный файл со стилями для меню в мобильной версии веб-страницы.
  - **fonts.css** — файл со стилями, который будем использовать для подключения используемых на веб-странице шрифтов, с помощью правил `@font-face`.
- Папка **images**, хранит все необходимые картинки веб-страницы.
- Папка **fonts** хранит все необходимые шрифты, которые мы будет подключать.
- В папке **videos** находится один файл с видео **video.mp4**, который используем для красивого динамического фона в верхней части страницы.

Вся страница будет иметь три представления

- Десктопное
- Для средних экранов (планшеты/нетбуки)
- Для маленьких экранов (мобильное представление)

Вся адаптивность будет реализована с применением медиа-выражений без каких либо CSS-фреймворков.

Описание лейаута

Как видим вся страничка состоит из **шести** отдельных блоков. Поэтому будет удобнее добавлять каждый и сразу же его описывать.

Блок `<head>` в заготовке уже содержит подключение всех необходимых ресурсов в виде CSS-таблиц, шрифтов и фавиконку.

На самом верхнем уровне (внутри тега `<body>`) будет три основных блока:

- Блок `<header class="header" …>` — этот блок будет отвечать за верхнюю часть веб-странички. В этом блоке будет находиться заголовок самого верхнего уровня `<h1>`. Иконка с фирменным логотипом и навигационное меню. Часто такие блоки еще называют **Hero.** В этом блоке предстоит познакомиться с понятиями относительного и абсолютного позиционирования элементов. Создавать адаптивный бэкграунд секциям с применением видео и изображения. Работу с тестом, CSS-трансформациями и даже немного JavaScript 😎 .

- Блок `<div class="sections" …>` — этот блок отвечает за содержание остальных контентных блоков веб-страницы:

  - `<div class="welcome" …>` — отвечает за приветствие пользователя на странице с предоставлением в сжатой форме информации о компании. В этой секции много работы с позиционированием элементов, картинками, текстом, адаптивностью.

  - `<div class="problem" …>` — этот блок содержит много текстовой информации, поэтому основная работа здесь сосредоточена на тонкой работе с текстом, ссылками, выравнивание элементов в блоке с помощью отступов и границ, для придания эстетического вида.

  - Блок `<div class="tech" …>` — представляет собой элемент, часто используемый на корпоративных сайтах. Много работы с обтеканием текста картинок, придание адаптивного представления.

  - `<div class="team" …>` — один из самых интересных блоков лендинга, так как содержит в себе работу с флекс-боксами, очень популярной, на данное время, спецификацией создания адаптивных сеток.

- Завершает основной лейаут нашей странички по традиции блок футера `<footer class="content-info" id="footer" …>`.

- Дополнительно есть еще мобильное меню `<div id="myNav" class="overlay">…`, где применяется так же много современных техник верстки.

Секция **header.**

Начнем построение нашего лейаута с секции header. Внутрь тела заготовки разместим его верстку:

```
<header class="header">
  <div class="video-wrapper">
    <video class="bg-video"
      src="videos/video.mp4"
      autoplay="" loop="" muted=""
    ></video>
  </div>
  <div class="title">
    <h1>PiperNet Is Here</h1>
    <p class="sub">Follow us into the future.</p>
  </div>
  <div class="logo-wrapper">
    <a href="#"><img src="images/logo.svg" alt="logo"></a>
  </div>
  <div class="home-header-container">
    <div class="nav-bar-links">
      <a class="nav-bar-link" href="#problem">the problem</a>
      <span class="divider">|</span>
      <a class="nav-bar-link tech-anchor" href="#tech">the tec
h</a>
      <span class="divider">|</span>
      <a class="nav-bar-link who-we-are-anchor" href="#team">t
he team</a>
    </div>
```

```
  </div>
  <div class="mobile-nav-button" onclick="openNav()">
    <i class="fa fa-bars"></i>
  </div>
</header>
```

Как видим этот блок состоит из блока для создания фона с помощью видео. Внутри блока находится тег `video` в котором через атрибут `src` указываем расположение видео-файла. Так же есть атрибуты `autoplay` - указывает на то, что видео начнет автоматически воспроизводиться после загрузки страницы, `loop` - видео будет воспроизводиться зациклено (по кругу), `muted` - будет воспроизводиться без звука.

Дальше у нас идет блок с основным заголовком страницы `<div class="title" ...>`. Содержит заголовок `<h1>` и подпись к нему внутри параграфа `<p>`.
Блок `<div class="logo-wrapper" …>` предназначен для отображения фирменного логотипа, который заключен внутри тега `<a>` , что бы нажатие на логотип было ссылкой на главную.

`<div class="home-header-container">` - содержит основное меню страницы и представлено в виде ссылок с якорями на соответствующие блоки страницы, разделенные вертикальной чертой в тегах `<span>` .

Блок `<div class="mobile-nav-button">` хранит кнопку для открытия мобильного меню. На широком экране данный блок не будет отображаться.

Далее перейдем к рассмотрению стилей этого блока.
Все стили будем описывать в файле `style.css`, который находится внутри папки `styles`.
Этот файл уже содержит немного стилей, которые задают размеры и шрифт для заголовков и параграфов.
Добавим это часть стилей внутрь нашего файла `style.css`:

```
/* header */

.header {
  position: relative;
  height: 517px;
```

```
}
```

Как видим эти стили применяются к элементу с классом `header`.
Первое свойство `position: relative` — без указания свойств `left`, `top`, `right` и `bottom` никак не повлияет на позиционирование элемента. Но сделает так, что все дочерние элементы с позиционированием `position: absolute` будут выстраиваться относительно края этого родительского элемента, а не от края окна браузера. Это позволит нам гибко разместить вложенные элементы внутри этого блока.
`height: 517px` — задают блоку высоту в пикселях.

Подставим следующую часть стилей для **header*:***

```
.header .title {
  position: absolute;
  left: 50%;
  top: 60%;
  transform: translate(-50%, -50%);
  text-align: center;
}

.header .title h1, .header .title p {
  color: #fff;
  white-space: nowrap;
}
```

Видим, что эти стили применимы к элементу с классом `title` , который будет находиться внутри элемента с классом `header`.

Свойства `position: absolute`, `left: 50%` и `top: 60%` позиционируют этот элемент относительно нашего родительского с классом `header` у которого `position: relative` и сместит начало этого элемента относительно центра родителя.
Так как теперь верхний правый угол элемента будет находиться в центре своего родителя, его необходимо также отцентрировать относительно этой точки, в этом нам поможет правило `transform: translate(-50%, -50%)`, которое сместит вверх и влево сам элемент на 50% своей ширины и высоты.
`text-align: center` — указывает центрировать текст внутри элемента.

С текстом во втором селекторе все достаточно просто, устанавливаем тексту белый цвет в hex-формате. А `white-space: nowrap` необходим для того, что бы не переносился текст по пробелам при уменьшении размера окна браузера даже в случае переполнения контейнера.

Добавим стили к контейнеру нашего видео-фона и непосредственно ему же:

```
.header .video-wrapper {
  position: absolute;
  width: 100%;
  height: 100%;
  z-index: -1;
  background-image: url("../images/Video.jpg");
  background-repeat: no-repeat;
  background-size: cover;
  overflow: hidden;
}

.header .bg-video {
  min-width: 100%;
}
```

Как видим контейнеру тоже задаем `position: absolute`, а правила `width: 100%` и `height: 100%` позволят растянуть его на всю ширину и высоту родительского контейнера, а `z-index: -1` сместит его на задний план.
Через свойство `background-image` задаем статический фон в виде картинки, которая будет повторять графический паттерн видеофайла для мобильного отображения страницы, что бы не нагружать воспроизведением видеофайла мобильные платформы.
`background-repeat: no-repeat` - отменить возможность повторения фона, `background-size: cover` - масштабирует фоновое изображение до размеров родительского контейнера с сохранением пропорций самого изображения.

Теперь перейдем к логотипу на внутри “шапки”.

```
.header .logo-wrapper {
  position: absolute;
  left: 2.5vw;
```

```
  top: 47px;
  width: 150px;
}
```

Так же позиционируем абсолютно от родительского контейнера, сдвигаем от края контейнера на 2,5% от viewport (то-есть от размера окна браузера) слева, на `47px` сверху и задаем ширину `150px`.
Стили для его центрирования на средних и маленьких экранах применим позже внутри необходимых медиа-выражений.

Перейдем к оформлению главного меню.

Рассмотрим вначале сам контейнер, в котором оно находится:

```
.header .home-header-container {
  position: absolute;
  top: 20px;
  right: 2.5vw;
}
```

Для него задаем тоже абсолютное позиционирование. Смещает от верху родительского контейнера на 20px и от правого края на 2.5% ширины окна браузера.

Непосредственно блок с пунктами меню:

```
.header .nav-bar-links {
  margin: 35px auto;
  text-align: right;
  letter-spacing: 1px;
}
```

`margin: 35px auto` - задаем границу сверху и снизу `35px`, авто по горизонтали.
`text-align: right` - само меню выровняем по правой стороне.
`letter-spacing: 1px` - немного увеличит отступы между буквами в элементах меню для лучшего визуального восприятия.

Теперь сами пункты меню

```
.header .nav-bar-link {
```

```
  color: #fff;
  font-weight: bold;
  text-decoration: none;
  text-transform: capitalize;
}

.header .nav-bar-link:hover {
  text-decoration: underline;
}
```

`color: #fff` и `font-weight: 700` - заменим стандартный цвет ссылок на белый, а тексту зададим жирное начертание (bold).
`text-decoration: none` - уберем стандартное подчеркивание у ссылок.
`text-transform: capitalize` - задаем правило, что все пункты меню будут наминаться с заглавной буквы.

Разделительная вертикальная черта:

```
.header .divider {
  color: #fff;
  margin: 0 10px;
}
```

`color: #fff` - белый цвет
`margin: 0 10px` - `10px` границы по горизонтали.

Стили для кнопки, которая будет отображаться в мобильном представлении, а на больших экранах будет отсутствовать:

```
.header .mobile-nav-button {
  position: absolute;
  left: 8vw;
  top: 44px;
  color: #fff;
  font-size: 28px;
  display: none;
  cursor: pointer;
}
```

`position: absolute`, `left: 8vw` и `top: 44px` - позиционируем абсолютно от родительского контейнера, сместим слева на 8% ширины экрана браузера и на `44px` сверху.
`color: #fff` - белый цвет.
`font-size: 28px` - задаем размер шрифта для используемой иконки.
`display: none` - скрываем саам элемент на широких экранах, изменим это свойство на `display: block` в соответствующем медиа-выражении.
`cursor: pointer` - изменим форму курсора, на соответствующую для кнопок и ссылок.

В итоге у нас должна получится заглавная часть страницы с отличным динамическим фоном. Адаптивности этой части мы додадим немного позже, используя для этого медиа-выражения.

---

sections

Теперь перейдем к созданию блока `<div class="sections" …>`, внутри которого будут находиться основные части страницы лендинга.
Создадим его сразу же после закрывающего тега `</header>`:

```
<div class="sections">


</div>
```

Секция **welcome**

Внутри созданного блока с классом `section` определим верстку для первой секции с классом welcome, краткое описание которой есть выше.

```
<div class="welcome">
  <div class="inner">
    <h3>Welcome to the web you never knew you wanted</h3>
    <div class="icons">
      <div class="icon ">
```

```
        <h4><strong>You</strong> own your data</h4>
        <img src="images/icon-data.png">
        <p>Hidden from the
Hoolis™ of the world</p>
      </div>
      <div class="icon ">
        <h4><strong>You</strong> control your identity</h4>
        <img src="images/icon-identity.png">
        <p>Decide what private
information is made public</p>
      </div>
      <div class="icon ">
        <h4><strong>You</strong> have the power</h4>
        <img src="images/icon-power.png">
        <p>With decentralized apps
and storage to make the
internet the fastest it can be</p>
      </div>
    </div>
  </div>
</div>
```

Тут вполне привычная нам верстка. Внутри структурных блоков есть заголовки 4-го уровня, параграфы и изображения.
Зададим для блока с классом `welcome` стили:

```
.welcome {
  text-align: center;
  background-color: #eee;
  padding: 82px 0 65px;
}
```

Рассмотрим за что отвечают данные стили:
`text-align: center` - отцентрировали внутри секции текст.
`background-color: #eee` - задали интересный светло-серый фон.

`padding: 82px 0 65px` - простые отступы сверху и снизу.

Стили для заголовка 3-го уровня. Заданы в общих стилях, их не будем изменять.

Теперь зададим стили для блока с содержимым контентом и непосредственно для каждого его элемента:

```
.welcome .icons {
  margin-top: 64px;
}

.welcome .icon {
  display: inline-block;
  width: 33%;
  vertical-align: top;
}
```

Рассмотрим эти стили сначала для контейнера.
`margin-top: 64px` - просто задаем верхний отступ.

`display: inline-block` и `width: 33%` для трех элементов содержимого отлично позволит расположить их три в ряд по горизонтали внутри контейнера.

Добавим немного стилей для более эстетического форматирования текста и картинок в наших элементах:

```
.welcome .icon h4 {
  color: #007765;
  font-family: GothamNarrow-Medium, serif;
  margin: 0;
  font-weight: 500;
}

.welcome .icon strong {
  font-family: GothamNarrow-Black, serif;
}

```

```
.welcome .icon img {
  margin: 34px auto 30px;
  max-width: 140px;
}
```

Стоит обратить только внимание только на ограничение размера в `140px` для картинок: `max-width: 140px`.

Секция **problem**

Подставим разметку для этой секции сразу же после секции welcome, внутри контейнера с классом `section`

```
<div class="problem">
    <div class="inner">
        <div class="content">
            <h2 id="problem">The Problem</h2>
            <p>Beneath the flashy homepages of your favorite s
ites lurk Hooli™ and other
                evil corporations out to hoard and sell your <
strong>most personal data.</strong></p>
            <p>What was built to be the ultimate platform for
the free sharing of knowledge
                has turned into a money-hungry monster feastin
g on our privacy and freedom.</p>
            <h5>But fear not, for Pied Piper has the answer! A
nd it’s <i>simple</i>:</h5>
            <p>An autonomous peer-to-peer network featuring di
stributed storage powered by
                universal compression, accelerated scheduling
allocation, and end-to-end encryption.</p>
            <h5 class="no-margin">Put even simpler — it’s the
internet, <strong>completely decentralized.</strong></h5>
```

```
            <p class="sub">(Still not getting it? <a class="link-out" href="#">Click here</a>).</p>
        </div>
    </div>
</div>
```

Как видим, это очень простая (с точки зрения разметки) секция, содержащая только текстовый контент.
Зададим форматирование для самого корневого элемента секции:

```
.problem {
    text-align: center;
    padding: 90px 0 120px;
}
```

Тут просто отцентрируем текст. И зададим необходимые отступы.
Зададим еще отступ для заголовка второго уровня:

```
.problem h2 {
  margin-bottom: 74px;
}
```

А для параграфов и заголовков 5-го уровня:

```
.problem p, .problem h5 {
    margin-bottom: 0.714em;
    color: #565656;
}
```

Просто добавим немного отступа снизу и изменим цвет текста.

Для нижней подписи (параграф с классом `sub`) добавим отдельное форматирование

```
.problem p.sub {
  font-size: 16px;
  letter-spacing: .5px;
}
```

Подпись будет иметь размер шрифта `16px` и увеличенное на `0.5px` расстояние между буквами.

Секция **tech**

Эта секция очень интересна, так как тут применяется несколько интересных техник с позиционированием.
Запишем верстку:

```
<div class="tech">
  <div class="inner">
    <div class="content">
      <h2 id="tech">The Tech</h2>
      <h6>Only Pied Piper has the necessary tools to build the internet of the future, today.</h6>

      <div class="icons">
        <div class="icon">
          <div class="img-icon">
            <img src="images/icon-peer.png" alt="peer-to-peer">
          </div>
          <div class="description dir-right">
            <p class="title"><strong>P2P Computer Communication Patent</strong></p>
            <p>Pied Piper's decentralized internet is built upon a patented foundation of peer-to-peer computer communication. This is the basis of the "internet we deserve."</p>
          </div>
        </div>

        <div class="icon">
          <div class="description dir-left">
            <p class="title"><strong>Award-Winning Middle-Out Compression</strong></p>
```

```
            <p>Our revolutionary middle-out compression was th
e missing piece preventing Peter and Gavin from pursuing their
original patent. After shattering the theoretical limit of com
pression at TechCrunch Disrupt, we’re now able to transport ev
en the most complex data files across our network within secon
ds.</p>
          </div>
          <div class="img-icon">
            <img src="images/icon-compression.png" alt="compre
ssion">
          </div>
        </div>

        <div class="icon">
          <div class="img-icon">
            <img src="images/icon-storage.png" alt="storage">
          </div>
          <div class="description dir-right">
            <p class="title"><strong>Distributed Storage</stro
ng></p>
            <p>Middle-out reduces the size of your files so we
can distribute, store, and reassemble them seamlessly across o
ur network. Not to mention your data is fully encrypted and to
tally redundant across our storage nodes.</p>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
```

По большому счету у нас есть три параграфа и три изображения.
Зададим для начала стили для корневого контейнера секции:

```
.tech {
  background-color: #eee;
```

```
  padding: 90px 0 120px;
  text-align: center;
}
```

Мы добавили немного отступов, светло-серый фон для данной секции и отцентрировали текс внутри самой секции.

Отформатируем теперь главный заголовок секции, который есть заголовком 2-го уровня:

```
.tech h2 {
  margin-bottom: 40px;
}
```

Отступ снизу у заголовка будет в `40px`.

Видим, что под главным заголовком у нас есть еще подпись в виде заголовка 6-го уровня, зададим ему стилей:

```
.tech h6 {
  margin-bottom: 90px;
}
```

Тоже отступ снизу в `90px`.

Контент находится внутри блоков с классом `icon`, которые имеют общий родительский блок с классом `icons` .
Зададим небольшой отступ снизу для блоков с контентом:

```
.tech .icon {
    margin-bottom: 67px;
}
```

Блок с картинкой сделаем инлайн-блоком с шириной в `33%`, что бы он занимал треть родительского блока и отступ сверху примерно на высоту заголовка с классом `title`, используем для этого величину в `1em`:

```
.tech .img-icon {
    display: inline-block;
    padding-top: 1em;
    width: 33%;
}
```

А для самой картинки зададим блочное отображение, для того что бы задать границы и уменьшим ее размер на `20%` с помощью правила `transform: scale(0.8)` и отцентрируем всем известным способом `margin: 0 auto` :

```
.tech .img-icon img {
    display: block;
    max-width: 100%;
    margin: 0 auto;
    transform: scale(0.8);
}
```

Теперь перейдем к форматированию текстового содержимого.
Зададим тем параграфам, которые должны выстраиваются справа от своей картинки правое направление текста, а тех кто слева — левое. И выровняем текст по ширине блока. Для этого у нас есть классы `.dir-right` и `.dir-left`:

```
.tech .dir-right p {
    direction: rtl;
    text-align: justify;
}

.tech .dir-left p {
    text-align: justify;
    direction: ltr;
}
```

Немного для другой цели у нас есть класс `.description`. Его мы используем для задания инлайн-блочного представления с шириной в `66%`, что бы занять остальные 2/3 пространства. А с помощью CSS-правила `vertical-align: top` сможем добиться необходимого выравнивания по вертикали:

```
.tech .icon .description {
  display: inline-block;
  vertical-align: top;
  width: 66%;
}
```

Добавим немного стилей для параграфа с классом `.title` для форматирования:

```
.tech p.title {
  color: #007765;
  margin-bottom: 0.5em;
}
```

Теперь наша секция tech выглядит точно так, как и задумывалось.

Секция ***team***

Тут мы создадим адаптивную сетку, используя флекс-боксы.
Запишем верстку:

```
<div class="team">
  <div class="inner">
    <div class="content">
      <h2 id="team">The Team</h2>
      <h6>Meet the Pipers</h6>
        <div class="photos-wrapper">
          <div class="photo who-we-are-info">
            <div class="img-photo">
              <img
                src="images/photo-richard.png"
                alt="Richard Hendricks"
              />
            </div>
            <p>Richard Hendricks<br /><i>Founder &amp; CEO</i>
</p>
          </div>
          <div class="photo who-we-are-info">
            <div class="img-photo">
              <img src="images/jared.png" alt="Jared Dunn" />
            </div>
            <p>Jared “Donald” Dunn<br /><i>Chief Operating Off
icer</i></p>
```

```
          </div>
          <div class="photo who-we-are-info">
            <div class="img-photo">
              <img src="images/monica.png" alt="Monica Hall" />
            </div>
            <p>Monica Hall<br /><i>Chief Financial Officer</i></p>
          </div>
          <div class="photo who-we-are-info">
            <div class="img-photo">
              <img src="images/chugtai.png" alt="Dinesh Chugtai" />
            </div>
            <p>Dinesh Chugtai<br /><i>Senior Programmer</i></p>
          </div>
          <div class="photo who-we-are-info">
            <div class="img-photo">
              <img src="images/gilfoyle.png" alt="Bertram Gilfoyle" />
            </div>
            <p>Bertram Gilfoyle<br /><i>Chief Systems Architect</i></p>
          </div>
          <div class="photo who-we-are-info">
            <div class="img-photo">
              <img src="images/bighetti.png" alt="Nelson Bighetti" />
            </div>
            <p>
              Nelson “Big Head” Bighetti<br /><i>Majority Investor</i>
            </p>
```

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

Видим, что секция состоит из двух областей, первая это заголовок, а вторая это шесть равнозначных карточек с изображениями и текстовыми подписями к ним.
Для начала зададим стили для корневого контейнера секции:

```
.team {
  text-align: center;
  padding: 76px 0 96px;
}
```

Тут делаем выравнивание по центру контента внутри контейнера и задаем отступы.
Для части с заголовком понадобится очень мало стилей, просто зададим отступы:

```
.team h2 {
  margin-bottom: 10px;
}

.team h6 {
  margin-bottom: 70px;
}
```

И приступим к сетке с карточками.
Общим контейнером для всех карточек является блок с классом `row-wrapper`.
Зададим этому блоку отображение `flex`. Укажем, что необходимо переносить контент на новую строку с помощью правила `flex-wrap: wrap`. А так же распределим пространство между элементами блока с помощью правила: `justify-content: space-around`:

```
.team .photos-wrapper {
    display: flex;
    flex-wrap: wrap;
    justify-content: space-around;
}
```

Очень показательно, как используя всего несколько правил нам получилось сделать отличную адаптивную сетку.

Каждая отдельная карточка имеет класс `photo`, зададим для них немного стилей для отступов, выравнивания и красивый эффект перекрытия курсором:

```
.team .photo {
  width: 297px;
  margin-bottom: 48px;
}

.team .photo:hover {
  opacity: 0.4;
}
```

Останется только немного отформатировать подписи к карточкам:

```
.team .img-photo {
  margin-bottom: 30px;
}

.team p {
  color: #007765;
  font-size: 18px;
  margin-bottom: 0;
}

.team i {
  color: #565656;
}
```

Секция ***footer***

Секция подвала (футера) как обычно будет содержать блок с иконками социальных сетей и копирайт.

Посмотрим разметку:

```
<footer class="footer">
    <div class="social-icons">
        <a href="#">
        <span class="icon-wrapper twitter">
          <i class="fa fa-twitter footer-icon"></i>
        </span>
        </a>

        <a href="#">
        <span class="icon-wrapper facebook">
          <i class="fa fa-facebook footer-icon"></i>
        </span>
        </a>

        <a href="#">
        <span class="icon-wrapper instagram">
          <i class="fa fa-instagram footer-icon" aria-hidden="true"></i>
        </span>
        </a>
    </div>
    <p class="footer-text">
        All other trademarks and copyrights are the property of their
        respective owners. Use of these names, trademarks and brands does not
        imply endorsement.
    </p>
</footer>
```

Для иконок используем библиотеку Font Awesome.

Есть контейнер с классом `social-icons`, который хранит внутри себя иконки и контейнер с классом `footer-text`, в котором находится текст с копирайтом.

Зададим стили непосредственно для самого футера:

```
.footer {
  text-align: center;
  background-color: #3d3d3d;
  min-height: 261px;
}
```

Тут просто отцентрируем содержимый текст, задаем цвет фона и фиксированную минимальную высоту.

Запишем стили для блока с иконками и содержимому блока соответственно:

```
.social-icons {
    padding: 50px 0;
}

.social-icons a {
    text-decoration: none;
}

.icon-wrapper {
    width: 50px;
    height: 50px;
    background-color: #fff;
    display: inline-block;
    border-radius: 50%;
    line-height: 50px;
    margin: 0 6px;
}

.footer-icon {
    color: #3d3d3d;
    font-size: 30px;
```

```
    vertical-align: middle;
}
```

Задаем для блока с иконками поля сверху и снизу по 50px.
Для ссылок `text-decoration: none`, что бы убрать подчеркивание под ними.
Каждая иконка соответственно находится внутри элемента с классом `icon-wrapper`. Задаем ему высоту и ширину `50px` и цвет фона `#fff` — серый. Сделаем вмещающие элементы круглой формы с помощью `border-radius: 50%`. Через `line-height: 50px`, сможем выровнять линию по которой будут выстроенные иконки строго по центру, а `margin: 0 6px` просто задает отступы между ними.

```
.footer-icon {
    color: #3d3d3d;
    font-size: 30px;
    vertical-align: middle;
}
```

Зададим иконкам размер шрифта и цвет. А `vertical-align: middle` позволит отцентрировать по вертикали иконки по линии, которую мы в классе `.icon-wrapper` поставили по центру элемента.

И теперь только немного стилей для текста:

```
.footer-text {
    width: 500px;
    line-height: 22px;
    margin: 30px auto;
    font-size: 16px;
    color: #fff;
}
```

Просто задаем фиксированную ширину самого блока в `500px`. Немного увеличим растояние мужду строками текста с помощью `line-height: 22px`. Отступы по вертикали в `30px` и `auto` по горихотали для выравнивания блока по центру. Размер шрифта `16px` и цвет серый (`#fff`).

@media

Теперь настало время добавить немного адаптивности для нашей странички. Использовать для этого будем стандартный подход с использованием медиа-выражений.
Всего у нас будет 4 опорных точки расширения экрана, для которых будем адаптировать стили.
Начнем с первой, это расширение экрана до максимального значения `1280px` :

```
@media (max-width: 1280px) {
  h2 {
    font-size: 67px;
  }

  h6 {
    font-size: 25px;
  }

  h4 {
    font-size: 25px;
  }

  .inner {
    max-width: 95%;
  }
}
```

Тут мы немного адаптируем размер шрифта для всех заголовков 2-го, 6-го и 4-го уровней, а так же зададим ширину контейнера с классом `inner`, равной `95%` от ширины родительского контейнера.

Следующей точкой будет расширение экрана до максимального значения `1080px`:

```
@media (max-width: 1080px) {
  h2 {
    font-size: 67px;
  }
```

```
5
6   h5 {
7     font-size: 29px;
8   }
9
10  .inner {
11    max-width: 90%;
12  }
13
14  .header {
15    height: 420px;
16  }
17
18  .header .logo-wrapper {
19    left: 50%;
20    top: 44px;
21    transform: translateX(-50%);
22  }
23
24  .header .home-header-container {
25    top: 110px;
26    right: 50%;
27    transform: translateX(50%);
28    font-size: 14px;
29    letter-spacing: 0.7px;
30  }
31
32  .welcome .icon {
33    float: none;
34    width: 45%;
35    display: inline-block;
36  }
```

```
37
38  .welcome .icon:last-child {
39    margin-top: 64px;
40  }
41
42  .welcome .icons {
43    text-align: center;
44  }
45
46  .problem p {
47    font-size: 21px;
48    white-space: normal;
49  }
50
51  .tech .icon {
52    display: flex;
53    flex-direction: column;
54    margin-bottom: 30px;
55  }
56
57  .tech .icon .description {
58    width: 100%;
59    order: 2;
60  }
61
62  .tech .img-icon {
63    width: 100%;
64    margin-bottom: 20px;
65  }
66
67  .tech .dir-right p,
68  .tech .dir-left p {
```

```
    text-align: center;
  }



}
```

Тут немного сузим контейнер с классом `inner` до `90%` от ширины родительского контейнера.
Дальше все стили касаются только “шапки ” страницы. Задаем фиксированную высоту в `420px`. Центрируем логотип уже знакомым нам методом. И блок с пунктами меню заголовка тоже центрируем.
Делаем более адаптивным блок **welcome**, запретив обтекание и задав ширину в `45%` при отображении их как инлайновый блок.

Для секции **problem** просто уменьшим размер шрифта параграфов.

Для секции **tech** зададим классу `.icon` зададим флекс-отображение с вертикальной осью. Это позволит выстроить ее элементы вертикально один за другим, задав им ширину в `100%` и немного отступов. А свойство `order` позволит поменять местами параграф и картинку в том блоке, в котором на широком экране было выравнивание по правой стороне контейнера.
И отцентрируем текст в параграфах.

Следующей опорной точной у нас будет `767px`. это мобильно представление:

```
@media (max-width: 767px) {
  h1 {
    font-size: 45px;
  }

  h2 {
    font-size: 39px;
  }

  h4 {
    font-size: 18px;
  }
```

```
13
14  h5 {
15    font-size: 18px;
16  }
17
18  h6 {
19    font-size: 18px;
20  }
21
22  p {
23    font-size: 18px;
24  }
25
26  .inner {
27    max-width: 540px;
28  }
29
30  .tech h6 {
31    margin-bottom: 50px;
32  }
33
34  video {
35    display: none;
36  }
37
38  .header .nav-bar-links {
39    display: none;
40  }
41
42  .header .mobile-nav-button {
43    display: block;
44  }
```

```
45
46  .welcome {
47    background-color: #fff;
48    padding-top: 30px;
49    padding-bottom: 30px;
50  }
51
52  .welcome .content h3 {
53    font-size: 31px;
54  }
55
56  .inner {
57    max-width: 540px;
58  }
59
60  .welcome .icons {
61    margin-top: 30px;
62  }
63
64  .welcome .icon {
65    width: 100%;
66    background-color: #eee;
67    padding: 30px 0;
68    margin-top: 0;
69    margin-bottom: 25px;
70  }
71
72  .welcome .icon p {
73    font-size: 18px;
74  }
75
76  .welcome .icon:last-child {
```

```
    margin-top: 0;
  }

  .problem p {
    font-size: 16px;
  }

  .footer-text {
    width: 80%;
  }
}
```

Тут больше всего достаточно тривиальных стилей. Изменяем отступы и размер шрифтов.
Стоит обратить внимание на то, что на этом расширение экрана мы прячем видео на фоне шапки и меню. А покажем кнопку мобильного меню, которое самое время добавить в наш документ после секции футера:

```
<div id="myNav" class="overlay">
  <a href="javascript:void(0)" class="closebtn" onclick="closeNav()">&times;</a>
  <div class="overlay-content">
    <a href="#problem" onclick="closeNav()">the problem</a>
    <a href="#tech" onclick="closeNav()">the tech</a>
    <a href="#team" onclick="closeNav()">the team</a>
  </div>
</div>
<script>
  function openNav() {
    document.getElementById("myNav").style.height = "100%";
  }

  function closeNav() {
    document.getElementById("myNav").style.height = "0%";
  }
```

```
</script>
```

Обратите внимание на секцию `<script>`, она крайне необходима для работоспособности нашего мобильного меню.

Следующую опорную точку (`639px`) можно назвать “косметической”. просто адаптируем максимальную ширину контейнера `inner` :

```
@media (max-width: 639px) {
  .inner {
    max-width: 90%;
  }
}
```
