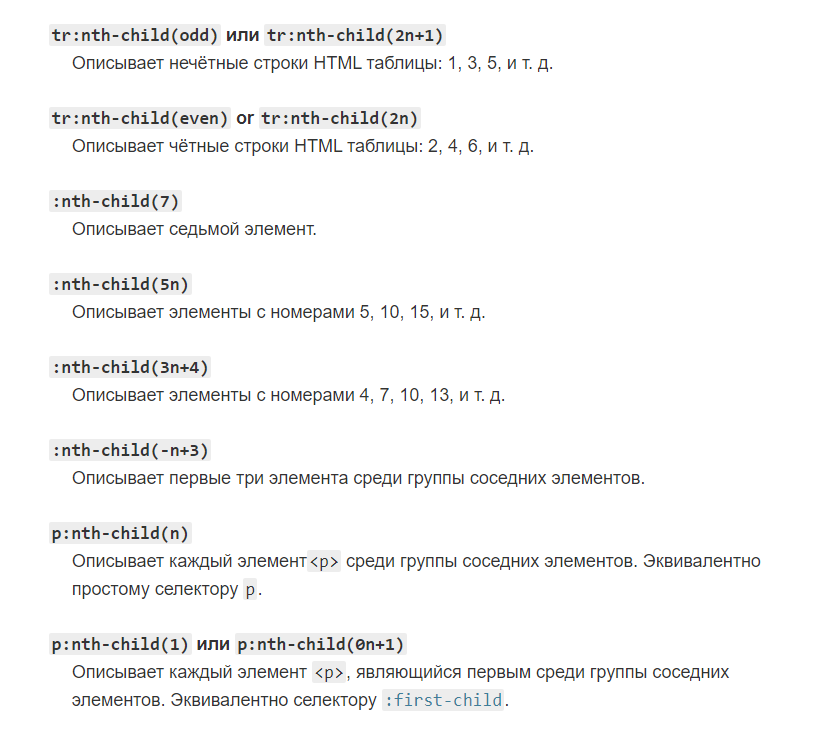
**Занятие 2**

Задача второго занятия - “раскрасить” макет, сделанный на первом занятии. Можно сделать примерно, как здесь: <https://preview.themeforest.net/item/jeffrey-personal-portfolio-template/full_screen_preview/24187700?_ga=2.18315711.763890385.1563976908-1245260935.1563976797>.

Но лучше выбрать здесь другой понравившийся макет <https://themeforest.net/search/portfolio?utf8=%E2%9C%93> и заюзать стили оттуда ;)

**Задать background-color (и вообще любые стили) для чередующихся блоков**

Это делается с помощью псевдокласса nth-child(). Подробнее про него можно почитать здесь <https://developer.mozilla.org/ru/docs/Web/CSS/:nth-child>. Для наглядности, самый полезный кусок ↴



**Подключение шрифтов**

1. Через сервис Google Fonts
2. Через правило @font-face из локальной папки:

@font-face {

font-family: ‘Test’;

src: url(‘fonts/test.ttf’)

}

**Псевдоэлементы**

С помощью css можно создать элемент не определяя его в html.

::before - до элемента

::after - после элемента

**Позиционирование**

По умолчанию все элементы на странице имеют статическое позиционирование (**position: static**), это означает, что элемент не позиционирован, и появляется в документе на своем обычном месте, то есть в том же порядке, как и в html-разметке.

Нет необходимости специально назначать это свойство какому-либо элементу, если только вам не требуется изменить ранее установленное позиционирование на дефолтное.

Относительное позиционирование (**position: relative**) позволяет вам использовать свойства: top, bottom, left и right, для расположения элемента относительно того места, где бы он появился при обычном позиционировании.

При абсолютном позиционировании (**position: absolute**), элемент удаляется из документа, и появляется там, где вы ему скажете. Таким образом, можно позиционировать любой элемент относительно страницы, однако этого не достаточно. Чаще нам необходимо позиционировать элементы относительно родительского блока. И тут без относительного позиционирования у родителя “абсолютного” элемента не обойтись. Назначив родительскому блоку относительное позиционирование (position: relative), мы сможем позиционировать любые дочерние элементы, относительно его границ.

Фиксированное позиционирование (**position: fixed**), является подразделом абсолютного позиционирования. Единственное его отличие в том, что он всегда находится в видимой области экрана, и не двигается во время прокрутки страницы. В этом отношении, он немного похож на фиксированное фоновое изображение.

Порядком наложения спозиционированных элементов можно свойством **z-index**. Значения от 0 до бесконечности.

**Прокинуть стили через props** <https://ru.reactjs.org/docs/dom-elements.html#style>

**SVG-Иконки**

Лучший формат для иконок - svg. Можно их взять на бесплатных сервисах:

<https://www.svgrepo.com/>

<https://freeicons.io/>

**Также можно использовать шрифтовые иконки (Font Awesome)**

Их преимущество в простоте стилизации. Для их стилизации используются те же св-ва, что и для стилизации шрифта. Например, если нужно поменять цвет иконке, используется св-во **color**, если нужно изменить размер - используется св-во **font-size**

(Для typeScript см. инструкцию 👇 <https://www.npmjs.com/package/@types/react-fontawesome>)

1. Переходим сюда <https://github.com/FortAwesome/react-fontawesome#get-started>
2. В терминале:
3. устанавливаем саму библиотеку:

$ npm i --save @fortawesome/fontawesome-svg-core

$ npm i --save @fortawesome/free-solid-svg-icons

$ npm i --save @fortawesome/react-fontawesome

или

$ npm i --save @fortawesome/fontawesome-svg-core @fortawesome/free-solid-svg-icons @fortawesome/react-fontawesome

через yarn:

$ yarn add @fortawesome/fontawesome-svg-core

$ yarn add @fortawesome/free-solid-svg-icons

$ yarn add @fortawesome/react-fontawesome

или

$ yarn add @fortawesome/fontawesome-svg-core @fortawesome/free-solid-svg-icons @fortawesome/react-fontawesome

1. устанавливаем бесплатные иконки:

$ npm i --save @fortawesome/free-brands-svg-icons

$ npm i --save @fortawesome/free-regular-svg-icons

1. В компоненте, в которой будем использовать, импортируем:

**import** { *FontAwesomeIcon* } **from '@fortawesome/react-fontawesome'** и b и нужные иконки:

**import { faReact } from '@fortawesome/free-brands-svg-icons',**

**import { faEnvelope } from '@fortawesome/free-solid-svg-icons'**

1. Вставляем иконку в нужное место <**FontAwesomeIcon icon=**{faReact}/> и радуемся :)

Еще один нужный хак💥

**Для того, чтобы ваши поля не были стандартного цвета, при выборе из списка ранее вводимых значений, нужно добавить:**

**input**:**-webkit-autofill**,

**input**:**-webkit-autofill**:**hover**,

**input**:**-webkit-autofill**:**focus**

**input**:**-webkit-autofill**,

**textarea**:**-webkit-autofill**,

**textarea**:**-webkit-autofill**:**hover**

**textarea**:**-webkit-autofill**:**focus**,

**select**:**-webkit-autofill**,

**select**:**-webkit-autofill**:**hover**,

**select**:**-webkit-autofill**:**focus** {

**border**: 0;

**-webkit-text-fill-color**: **#1f1f20**;

**-webkit-box-shadow**: 0 0 0 1000**px transparent inset**;

**transition**: **background-color** 5000**s ease-in-out** 0**s**;

**background**: **-webkit-linear-gradient**(**top**, **rgba**(255,255,255,0) 0%,**rgba**(0,174,255,0.04) 50%,**rgba**(255,255,255,0) 51%,**rgba**(0,174,255,0.03) 100%);

}

где вместо **transparent** можно указать свой цвет.

**Дополнительно:**

[**Короткий и простой видос про sass**](https://www.youtube.com/watch?v=Zz6eOVaaelI) **(на английском, но там все понятно)**

Если после установки Sass не работает, то попробуйте установить его, указав версию 4.14.1 - **npm install node-sass@4.14.1**