

# Модуль - Верстка

---

2/10 занятие

# Браузеры

---

Программа для интерпретации (просмотра в конечном виде) файлов .html

Популярные - Chrome, FireFox, EDGE

Интерфейс(UI):

- Адресная строка - запросы
- Viewport - область просмотра контента
- Элементы управления - Навигация и обновление
- DevTools

**ТЕГИ**

---

# ТЕГИ

---

**Тег (tag)** – инструкция для браузера о том, что необходимо создать графический элемент.

- Название в угловых скобках (< и > ).
- Теги бывают Одиночные и Парные

```
<single>  
<twin> ... </twin>
```

# ТЕГ = Элемент = Объект

---

Большинство элементов графические, но есть и невидимые(внутри тега head)

Графический элемент визуально - прямоугольник

Элемент обладает параметрами, такими как **стили**, **размер**, **ссылка**

**Объект** - структура данных в абстрактной форме

`<img>`



```
img = {  
    свойство1: значение;  
    свойство2: значение;  
}
```

# Вложенность тегов

---

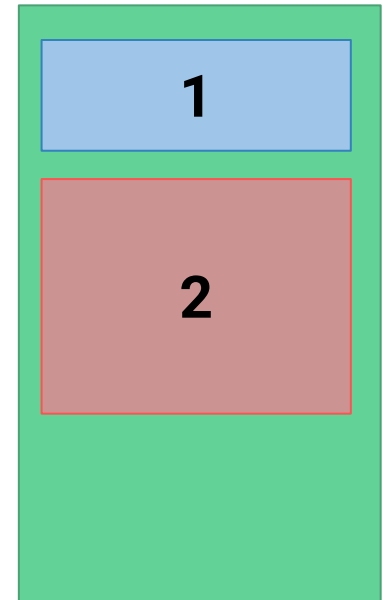
Теги можно вкладывать в другие парные теги, образуя структуры

Визуально это как пирамида в 2д вид сверху

Пары в парах образуют уровни (4 пробела [Space] или 1 таб [Tab])



```
<div>  
  <div>1</div>  
  <div>2</div>  
</div>
```



# ВИЗУАЛИЗАЦИЯ ТЕГОВ

---

- Все теги, при отображении, формируют прямоугольные фигуры в области отображения браузера(viewport).
- Фигуры выстраиваются в том порядке, в котором они описаны в коде, слева-направо\сверху-вниз (подобно тому, как мы привыкли печатать текст)
- Теги обладают визуальными свойствами(ширина, высота, цвет фона) и способны влиять на отображение своего содержимого - текста, картинок или других вложенных тегов.

❗ В html5 можно переопределить свойства и внешний вид абсолютно любого тега

# Одиночные и парные

---

- Одиночные теги используются самостоятельно
- Парные теги образуют “контент” между открывающим и закрывающим
- Контент - другие парные теги или текст
- **Закрывающий** тег – повторяет открывающий с символом /

Видимая на экране часть HTML-документа  
содержимое тега **<body>**.

Тег **<html>** всегда содержит **<body>** и **<head>**

```
<html>  
  <head>  
    скрытая информация  
  </head>  
  <body> Контент <img> </body>  
</html>
```



Важно соблюдать последовательность вложения.



# ОСНОВНАЯ РАЗМЕТКА

---

`<!DOCTYPE>` - тип документа, указывает браузеру, как интерпретировать веб-страницу

`<html>` - общий контейнер

`<head>` - контейнер, содержащий метаинформацию для браузеров и поисковых систем.

`<body>` - контейнер для видимого содержимого веб-страницы, отображаемого в окне браузера.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Название страницы</title>
  </head>
  <body>
    <h1>Заголовок</h1>
    <!-- Содержимое документа -->
    <p>Форматированный текст</p>
  </body>
</html>
```

# ТЕГИ ВНУТРИ <HEAD>

---

```
<!DOCTYPE html>
<html>
  <head>
    <title>Название страницы</title>
    <meta charset="UTF-8">
    <meta name="viewport"
      content="width=device-width,
      initial-scale=1">
    <link rel="stylesheet" href="style.css">
    <script type="text/javascript"
      src="script.js"></script>
  </head>
  <body>
    ...
  </body>
</html>
```

**<title>** - Текст, отображающийся во вкладке браузера

**<meta>** - содержит скрытую информацию

**<link>** - запрашивает дополнительные файлы с кодом (ресурсы)

**<script>** - запрашивает файлы JavaScript

# ТЕГИ ЗАГОЛОВКОВ

---

**<h1>** {Heading} Заголовок 1-ого уровня

**<h2>** {Heading} Заголовок 2-ого уровня

...

**<h6>** {Heading} Заголовок 6-ого уровня

**Заголовок первого уровня**

**Заголовок второго уровня**

**Заголовок третьего уровня**

**Заголовок четвертого уровня**

**Заголовок пятого уровня**

**Заголовок шестого уровня**

Меняет размер шрифта от большего **<h1>** к меньшему **<h6>**

```
<h1>Огромный заголовок</h1>
```

```
<h2>Большой заголовок</h2>
```

```
<h3>Средний заголовок</h3>
```

```
<h4>Небольшой заголовок</h4>
```

```
<h5>Маленький заголовок</h5>
```

```
<h6>Крошечный заголовок</h6>
```



# ТЕГИ ФОРМАТИРОВАНИЯ

---

<p>

{Paragraph} Параграф

<br>

{Brake} Разрыв строки

<p>Произвольный текст, в котором  
не учитывается перенос строк,  
даже если в редакторе вы его сделали</p>

<p>Принудительный<br/>  
перенос<br/>строк</p>

# ТЕГИ НАЧЕРТАНИЯ

---

<b>&lt;i&gt;</b> , <b>&lt;em&gt;</b>	{ <i><u>I</u>tal</i> ic}, { <u>E</u> mphasis} <u>Курсивный</u> текст, <u>акцент</u>
<b>&lt;b&gt;</b> , <b>&lt;strong&gt;</b>	{ <u>B</u> old}, { <u>S</u> trong} <u>Жирный</u> текст
<b>&lt;u&gt;</b>	{ <u>U</u> nderline} <u>Подчеркнутый</u> текст

```
<p>Текст, в котором некоторые слова выделены <i>курсивом</i>,  
<b>жирным</b> или <u>подчеркнуты</u>.</p>
```



# ТЕГИ СПИСКОВ

---

**<ol>**

{Ordered list} Упорядоченный список

**<ul>**

{Unordered list} Маркированный список

**<li>**

{list item} Элемент любого из списков

```
<ul>
  <li>Элемент списка с маркером</li>
  <ul>
    <li>Элемент списка с маркером</li>
    <li>Элемент списка с маркером</li>
  </ul>
</ul>
<ol>
  <li>Элемент списка</li>
  <li>Элемент списка</li>
</ol>
```

<>

- Элемент списка с маркером
  - Элемент списка с маркером
  - Элемент списка с маркером
- 1. Элемент списка
- 2. Элемент списка

# ТЕГИ ДЛЯ КОНСТРУИРОВАНИЯ

---

**<div>**

{Division} Разделение контента на логические блоки

**<span>**

{Span} Выделяет(Охватывает) текст

```
<div>
  <div>
    <span>Внимание!</span>
  </div>
  <div>
    Ваша <span>скидка</span> <span>30</span>% только сегодня!
  </div>
</div>
```

# Задание 1.1

---

- перенести структуру с фигмы  
в файл card.html

## Тексты

- \$20/mth
- Business plan
- Billed annually.
- 200+ integrations
- Advanced reporting and analytics
- Up to 20 individual users
- 40GB individual data each user
- Priority chat and email support
- Another feature
- Another feature
- Another feature
- Another feature
- Another feature
- Get started
- Chat to sales

## \$20/mth

### Business plan

Billed annually.

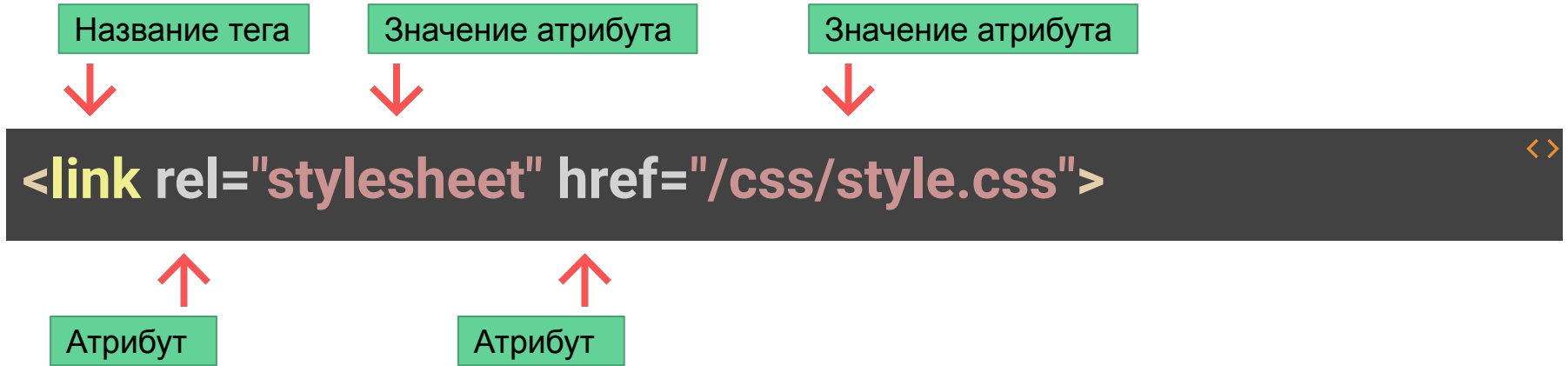
- ✓ 200+ integrations
- ✓ Advanced reporting and analytics
- ✓ Up to 20 individual users
- ✓ 40GB individual data each user
- ✓ Priority chat and email support

Get started



# АТТРИБУТЫ

---



Атрибуты расширяют возможности отдельных тегов.

У одного тега может быть несколько атрибутов, в таком случае, они пишутся через [Пробел].

# ОСОБЕННОСТИ АТТРИБУТОВ

---

- Указываются только в открывающем или одиночном теге
- Атрибут может быть пустым (<a href="" >, <input checked>)
- Порядок атрибутов в тегах не имеет значения
- Атрибуты могут содержать путь к файлу и др. ссылки

```
<html lang="ru">  
<meta charset="UTF-8">  
<link rel="stylesheet" href="/css/style.css" >  
  
<a href="" >...</a>  
<p height="118" width="438">...</p>  
<input type="radio" checked>
```



# ИЗОБРАЖЕНИЯ

---

**<img>** {*image*} Выводит на экран изображение

- Атрибут **src** указывает на расположение картинки
- Атрибут **alt** содержит текст, который будет показан, если изображение недоступно

```

```

<>

Форматы изображений для web:

- png
- jpg\jpeg
- svg
- webp



# ССЫЛКИ

---

**<a>** {*anchor*} Делает содержимое гиперссылкой(якорем)

- Обязательный атрибут **href** - путь до ресурса(файл, страница)
- Атрибут **target** - указывает браузеру где открыть,  
в этом же окне или в новом

```
<a href="google.com" target="_blank">ссылка </a>
```



# ПУТИ

---

Пути могут быть абсолютными и относительными

```
<a href="www.google.com" title="Подсказка">Абсолютный путь к ресурсу</a>  
<a href="page1.html">Путь, относительно текущей страницы</a>  
<a href="/pages/page2.html">Путь относительно корня сайта</a>  
<a href="../page2.html">Путь, с переходом в родительский каталог</a>  
<a href="page3.html" target="_blank">Ссылка откроется в новой вкладке</a>
```

Хорошая практика - использовать относительные пути



## Задание 1.2

---

- создать файл links.html
- добавить в него картинку(локально)  
<https://w.forfun.com/fetch/a4/a4e35e08c53cf649cb8cad61d99a01f7.jpeg>
- Добавить ссылку на любой ресурс

**перерыв**

# СИСТЕМА КОНТРОЛЯ ВЕРСИЙ GIT

---





# ОСНОВНЫЕ ТЕРМИНЫ

---

Git — это набор консольных утилит, которые отслеживают и фиксируют изменения в файлах (исходный код и бинарные файлы).





С помощью системы `git`, в **папке с рабочими файлами**, создается локальный репозиторий (Хранилище версий).

 **Локальный репозиторий** - скрытая папка `.git`, в которой хранятся зафиксированные версии проекта (бэкапы или иначе - копии файлов в состоянии, на момент фиксации)

 **“Проект”** (в лексическом контексте) - папка с файлами сайта или приложения.

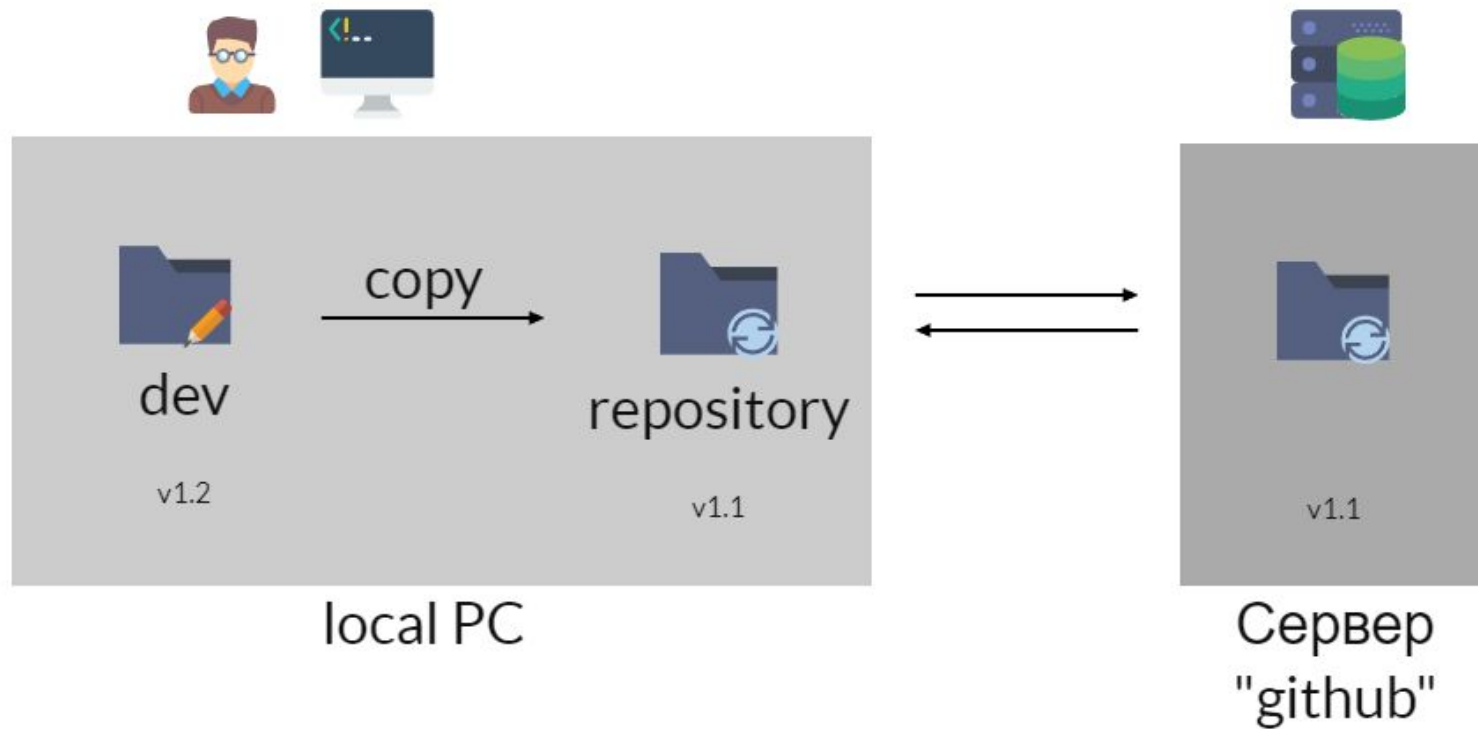
# ВОЗМОЖНОСТИ

---

-  Сохранение истории изменений в проекте
-  Откат изменений к любой версии проекта
-  Сравнение версий между собой
-  Совместная работа над проектом

# ЖИВОЙ ПРИМЕР

---



# КОНТРОЛЬ ВЕРСИЙ

---

Каждому созданному гитом "слепок" назначается уникальное **хэш**-имя и дата создания.

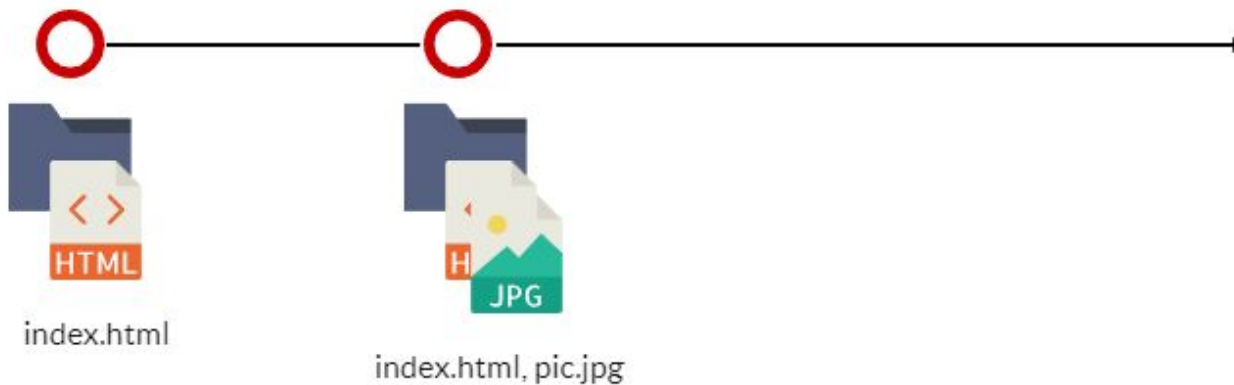
Например:

```
24b9da6552252987aa493b52f8696cd6d3b00373
```



11 янв (v.1)

15 янв (v.1.1)



# СОЗДАНИЕ И НАСТРОЙКА РЕПОЗИТОРИЯ

---

## Основные команды для консоли:

Создать локальный репозиторий в текущей директории:

➞ `git init`

Глобально настроить свои персональные данные и перенос строк(Win\Mac):

➞ `git config --global user.name "ваше_имя"`

➞ `git config --global user.email "ваша_почта"`

➞ `git config --global core.autocrlf true`

➞ `git config --global core.autocrlf input` (только для Mac\Linux)

➞ `git config --global core.safecrlf true`

# ФИКСАЦИЯ ИЗМЕНЕНИЙ

---

Добавить содержимое рабочей директории в "индекс":

⇒ `git add имя_файла`

⇒ `git add *` (*выберет все видимые файлы*)

Показать состояние файлов в "индексе":

⇒ `git status`

Удалить файлы из "индекса"

⇒ `git reset *`

Фиксация "версии", иными словами, создание слепка из добавленных в индекс файлов. (Можно добавить комментарий ключом `-m"комментарий"`):

⇒ `git commit -m"комментарий"`

# УДАЛЕННЫЙ РЕПОЗИТОРИЙ

---

Скачивание содержимого удаленного репозитория в пустую директорию:

⇒ `git clone https://github.com/логин/репозиторий.git`

Указать, в какой удаленный репозиторий будет отправляться копия локального репозитория:

⇒ `git remote add https://github.com/логин/репозиторий.git`

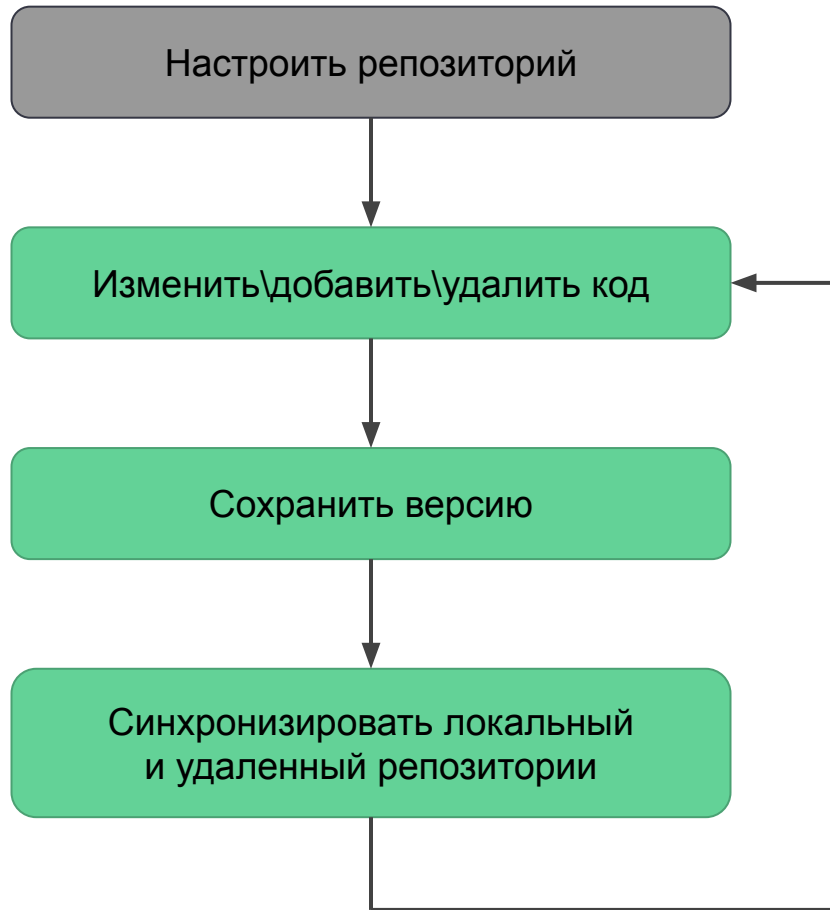
Отправка локальных слепков в удаленный репозиторий например github:

⇒ `git push репозиторий ветка`

Получение слепков из удаленных репозитория в локальный(авто слияние):

⇒ `git pull репозиторий ветка`

# ЦИКЛ РАБОТЫ С GIT





# ШАГ 1

---

Откройте консоль и настройте персональные данные и преобразование конца строк:

```
git config --global user.name "ваше_имя"  
git config --global user.email "ваша_почта"  
git config --global core.safecrlf true
```

Windows

```
git config --global core.autocrlf true
```

Mac, Linux

```
git config --global core.autocrlf input
```

Перейдите в директорию C:/work/repository/ и инициализируйте новый репозиторий:

```
cd C:/work/repository/  
git init
```

Создайте в директории [about.html](#) с основной разметкой и текстом в <body>: *"It is my first document"*

## ШАГ 2

---

Проверьте статус файлов, что видит git(новые\измененные\нет изменений)

```
git status
```



Проиндексируйте всё(\*) содержимое **/repository** и проверьте результат:

```
git add *  
git status
```



## ШАГ 3

---

Создайте первый коммит(**commit**) с комментарием "произвольный текст"

```
git commit -m "initial version"
```



Просмотрите список коммитов(история проекта):

```
git log
```



Добавьте в **about.html** новую строку с текстом: *Привет, я Имя!*

## ШАГ 4

---

Теперь, когда файл в проекте изменился, "версия" проекта, в целом, отличается от последней сохраненной. Зафиксируйте текущую.

```
git status
git add *
git status
git commit -m"added current date"
```



Если открылся Vi или Vim, то сохранить комментарий и закрыть его можно командой **:wq**

```
git config --global core.editor "'путь_до_visual_studio_code/vscode.exe' open -n -w"
git config --global core.editor "путь_до_visual_studio_code/vscode.exe"
```

## ШАГ 5

---

Создайте новый репозиторий на github и свяжите с ним локальный репо:

```
git remote add origin https://github.com/username/название_репозитория.git <>
```

Отправьте в него версию локального репо и проверьте на сайте гитхаба:

```
git push origin master <>
```

Теперь можно скачать репозиторий с гитхаба на любой ПК, где установлен GIT

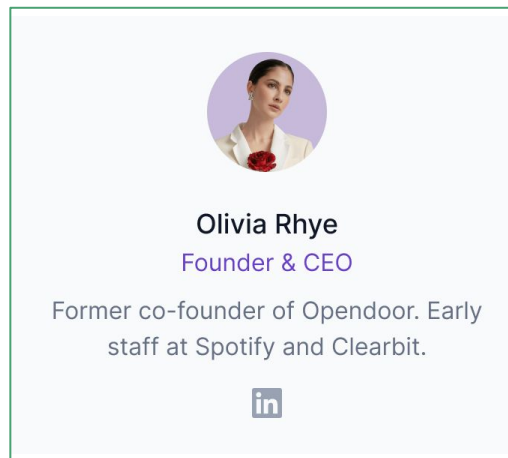
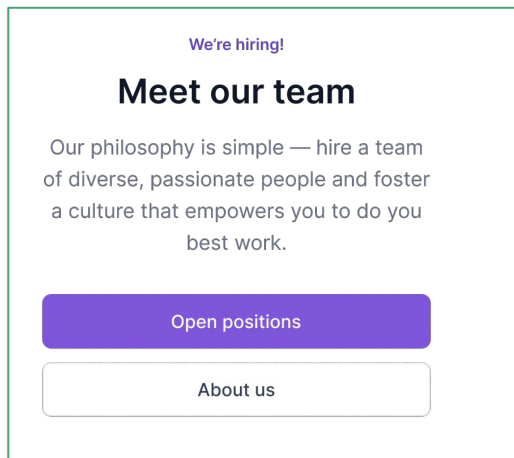
```
git clone https://github.com/username/название_репозитория.git <>
```

**Конец занятия**

# Задание (домашнее)

---

1. HTML теги - дополните about кратким эссе в пару предложений о себе, используя теги h1, h2, p, b, i, u, br, ul li, table\*
2. Структура - перенесите структуру блоков(используя div'ы) в team.html



3. GIT
  - Добавить в локальный репозиторий картинку
  - Сделайте commit новой версии
  - Сделайте push локальной версии на github в персональный репозиторий

# ТАБЛИЦЫ - ДЗ\*

---

**<table>**

{Table} Контейнер таблицы

**<tr>**

{Table row} Ряд внутри таблицы

**<td>**

{Table data} Ячейка с данными внутри ряда

**<th>**

{Table heading} Ячейка с заголовками столбцов

```
<table>
  <tr>
    <th>Товары</th>
    <th>Вес</th>
  </tr>
  <tr>
    <td>Мука</td>
    <td>100 кг</td>
  </tr>
</table>
```

