**Размещаем проект на gh-pages через интерфейс GitHub**

Чтобы проект стал доступен любому пользователю, его нужно залить на сервер. Есть как бесплатные облачные серверы, так и платные. Например, у GitHub есть собственный хостинг, на котором можно бесплатно размещать свои проекты — GitHub Pages.

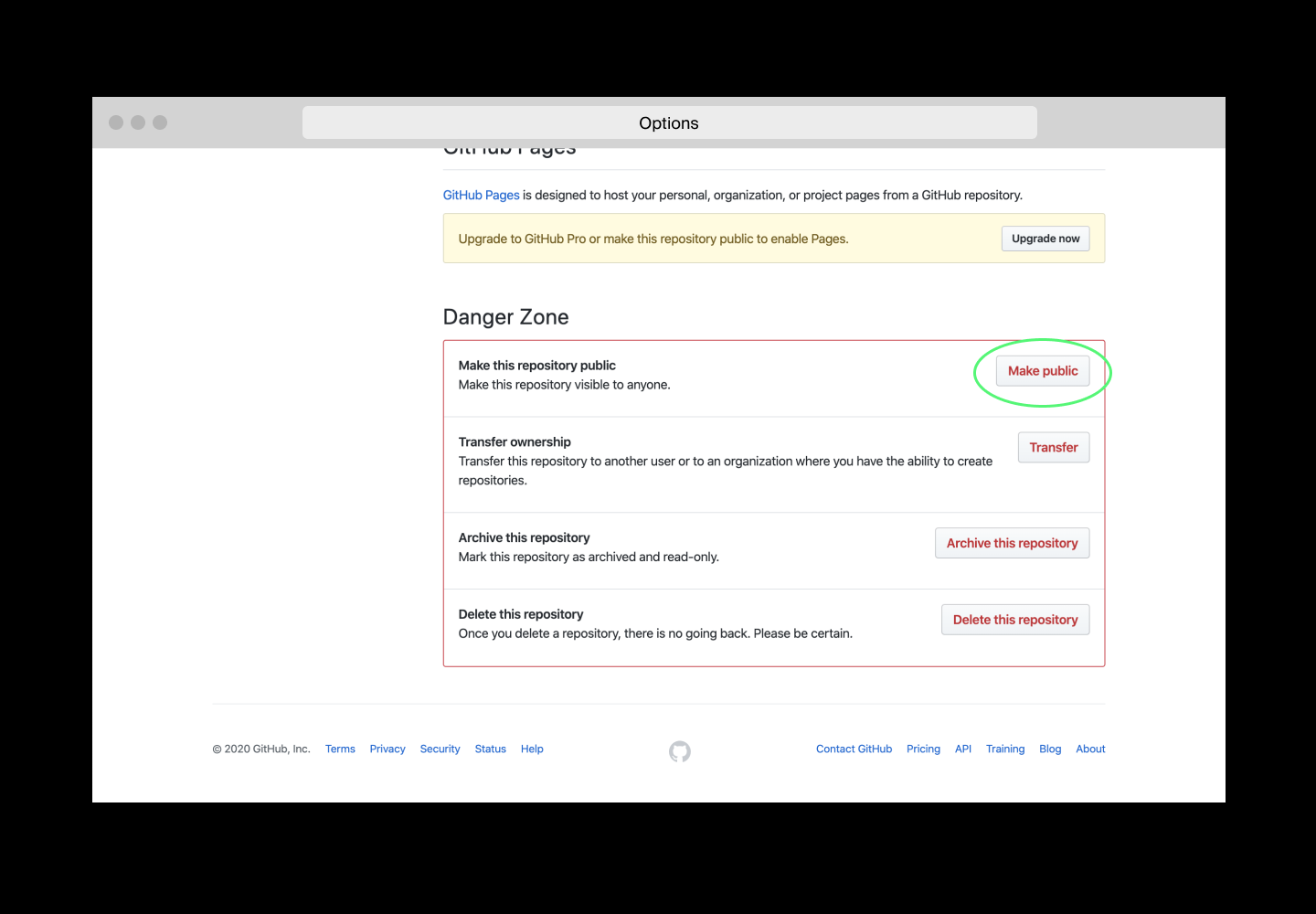
В этом уроке расскажем, как разместить проект на gh-pages на примере репозитория react-intensive пользователя students-yandex.

**Размещение работы на сервере**

Вашу работу сможет посмотреть весь интернет. Процесс перемещения кода на удалённый сервер называется ”deploy“. Переместить код нужно так, чтобы его работа не отличалась от работы на локальном компьютере.

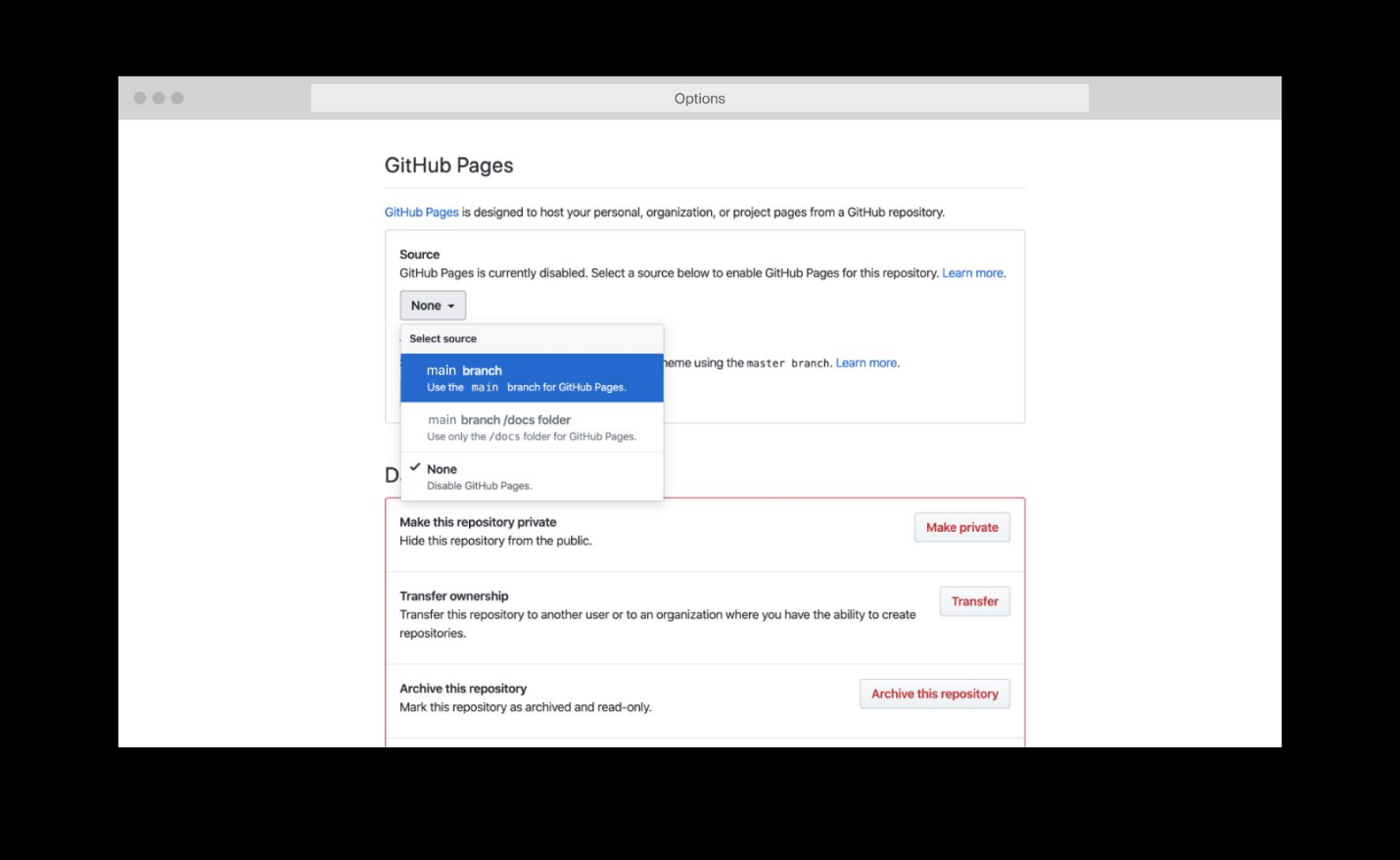
Ваши файлы уже есть на сервере — в удалённом репозитории, а процесс «деплоя» полностью берёт на себя Github. Нужно только задать пару настроек для правильной работы сайта.

Сперва сделаем репозиторий публичным:



*Прокручиваем страницу вниз до пункта ”Danger Zone“ в пункте ”Settings” и нажимаем на кнопку ”Make public“*

Чуть выше в настройках находится пункт ”Source“. В выпадающем меню нужно выбрать ветку, код которой будет опубликован. Это может быть ветка main:



*В выпадающем меню выбираем ветку ”main”*

После выбора ветки появится ссылка на опубликованный проект. Обычно он моментально публикуется на GitHub Pages, но иногда это занимает до 20 минут.

Есть неприятность, с которой вы можете столкнуться: проект не будет доступен по выданной ссылке. Чтобы это исправить, допишите в конце адреса index.html:

Скопировать код

https://students-yandex.github.io/react-intensive/index.html

У GitHub Pages есть возможность держать финальную версию в ветке main, а выкладывать в интернет из другой ветки. Для ветки зарезервировано специальное название — gh-pages. Сайт будет виден по тому же адресу:

Скопировать код

https://students-yandex.github.io/react-intensive/index.html

Новую ветку нужно указать в настройках Github в разделе ”Source“. Так вы разделите релизную ветку и main. При этом в ветке main можно будет продолжать улучшать код.

**Итого**

Ваш код попал из ветки main в ветку gh-pages. Он размещён на GitHub Pages и доступен всему миру по ссылке:



Вы познакомились с интерфейсом работы Github Pages. Так как мы пишем на React, то на GitHub Pages должен попадать только собранный проект, ведь код нужно транспилировать. Разберёмся с этим в следующем уроке.

**Добавляем скрипт для деплоя на GitHub Pages**

В предыдущем уроке мы разобрались с тем, как деплоить проект на gh-pages с помощью интерфейса GitHub. Но есть более современный и быстрый способ обновлять код приложения в GitHub Pages.

**Наcтройка Github Pages**

Чтобы добавить проект в сервис Github Pages, нужно создать в репозитории ветку gh-pages и загрузить туда собранный проект. Но проще сделать это [специальным пакетом](https://www.npmjs.com/package/gh-pages) — gh-pages.

1. Установите его:

Скопировать кодBASH

npm install gh-pages --save-dev

В файле package.json в разделе devDependencies появится новая строка:

Скопировать кодJSON

devDependencies: {

...

"gh-pages": "^3.1.0"

...

}

2. Добавьте в package.json новый скрипт — deploy. Этот скрипт должен вызывать пакет gh-pages и передавать ему папку, в которой находится собранный проект. В нашем случае это папка build:

Скопировать кодJSON

scripts: {

...

"deploy": "gh-pages -d build"

}

3. Обычно перед деплоем нужно собрать проект командой npm run build. Чтобы не делать это каждый раз вручную, можно сказать npm, чтобы перед каждым вызовом скрипта deploy вызывался build. Для этого в раздел scripts добавьте скрипт predeploy:

Скопировать кодJSON

scripts: {

...

"predeploy": "npm run build",

"deploy": "gh-pages -d build"

}

Такой скрипт будет вызываться при каждом вызове скрипта deploy.

4. Добавьте в package.json указание домашней директории для проекта. Это необходимо для работы задеплоенного приложения:

Скопировать кодJSON

"homepage": "./",

5. Настройки готовы. Запустите команду npm run deploy. Проект соберётся и содержимое папки dist попадёт в удалённую ветку gh-pages. При этом может возникнуть такая ошибка:

Скопировать код

Couldn't find remote ref refs/heads/gh-pages

Тогда введите команду npx gh-pages-clean и заново проделайте этот шаг.

6. Зайдите на Github и убедитесь, что там появилась ветка gh-pages со сборкой. 7. Через несколько минут сайт будет доступен по адресу:

Скопировать код

https://yandex-studens.github.io/react-intensive

Только замените yandex-studens на своё имя пользователя, а react-intensive – на название репозитория.

Теперь, после того, как вы внесли изменения в проект, достаточно вызвать команду npm run deploy и изменения отобразятся на сайте.

Добавьте ссылку на сайт в описание репозитория на Github. Пользователи, которые зайдут в него, смогут посмотреть не только код, но и сам сайт, и быстро его протестировать. Удобно, правда?

С роутингом на GitHub Pages много проблем. Варианты решений есть тут <https://create-react-app.dev/docs/deployment/#notes-on-client-side-routing>, но они не простые.