

PHASE 2

WEEK 2

DAY 4



План

1. Секреты в разработке и где их хранить
2. `process.env`
3. Сервисы для развёртывания: Heroku, Render

Секреты в веб-разработке

- хосты и порты серверов (например, сервера БД)
- логины и пароли
- API-ключи
- секретные слова для шифрования
- ... и другие

Вопрос: где хранить секреты?

- Можно ли прописывать секреты в коде приложения?
- Нельзя.
- А где тогда?
- `process.env`



process.env

process

Глобальная переменная, доступная в Node-приложениях.

Содержит объект с описанием запущенного Node-процесса.

process.env, environment variable

Объект с переменными окружения, в котором запущен Node-процесс.

Переменная окружения (**environment variable**) — текстовая переменная из операционной системы.

Задать env через командную строку

Linux, Mac

```
PORT=1234 node app.js
```

Windows

```
set PORT=1234 && node app.js
```


Задать env через файл

1. Создать файл `.env` и добавить его в `.gitignore`

2. Прописать необходимые переменные в виде

“KEY=value”:

```
MY_IMAGINARY_DOG=very good boy
```

```
YOU_ARE_FABULOUS=true
```

```
IMAGINATION_DEPLETED=1234
```

3. Установить пакет 'dotenv': `npm i dotenv`

4. `require('dotenv').config();`

Прочитать env-переменные в коде

// Прочитать переменную из окружения.

// Если такой переменной нет,

// передать стандартное несекретное значение:

```
const DB_HOST = process.env.DB_HOST || 'localhost';
```

// или ('??' — оператор нулевого сравнения)

```
const DB_HOST = process.env.DB_HOST ?? 'localhost';
```

Heroku

Heroku

- Сервис для публикации веб-приложений
- Приложение будет работать, даже если сервер на вашем компьютере выключен
- Для доступа к проекту можно использовать ссылку с доменным именем

Heroku, Heroku CLI

Установить пакет 'heroku' на Ubuntu:

```
sudo snap install --classic heroku;
```

Установить пакет 'heroku' на другие ОС:

<https://devcenter.heroku.com/articles/heroku-cli>






Env через веб-интерфейс Heroku

Config Vars

Config vars change the way your app behaves. In addition to creating your own, some add-ons come with their own.

Config Vars

[Hide Config Vars](#)

DB_USER	postgres	 
DB_PASS	postgres	 
KEY	VALUE	

Env в Heroku через .env (Linux || Mac)

- Записать в конфиг Heroku из .env:

```
heroku config:set $(cat .env | sed '/^$/d; /#[[:print:]]*$/d')
```

- Прочитать из конфига в Heroku в .env:

```
heroku config | sed 's/: */=/g; /^=/d' >> .env
```

Важное при работе с Heroku

Номер порта **обязательно** должен формироваться через переменную окружения **PORT** (Heroku задаст её само), иначе будет ошибка запуска сервера.

Heroku – базы данных

В **Heroku** так же можно создать и базу данных (например **PostgreSQL**), чтобы ваше приложение могло подключаться и работать с ней после деплоя.

Heroku: базы данных, конфигурация

```
// config.json

"production": {
  "use_env_variable": "DATABASE_URL",
  "dialectOptions": {
    "ssl": {
      "rejectUnauthorized": false
    }
  }
}
```

Heroku: базы данных, скрипты запуска

```
// package.json
```

```
"start": "sequelize db:migrate && node app.js"
```

Render.com

Render: этапы развёртывания

- Login на render.com
- Подготовить облачную БД с PostgreSQL на render.com через ``New` > `PostgreSQL``
- Создать приложение внутри сервиса Render.com через ``New` > `Web Service``
- Заполнить форму для дальнейшего развёртывания с указанием окружения ``Node.js``
- Подвязать ``GitHub`/`GitLab`` аккаунты для связи с проектом/репозиторием или используйте публичную ссылку
- Заполнить переменные окружения, ``DATABASE_URL`` и другие...
- Заполнить ``Build Command``, например: ``npm i && npm run db-migrate`` и ``Start Command``, например: ``node app.js``
- Начать развёртывание с отслеживанием логов
- Дождаться окончания развёртывания, иногда около 5 минут