NODE.JS NIGHTS

STRV

ARCHITECTURE

Ondrej Zivnustka, backend developer at STRV

STRV

BACKEND ARCHITECTURES

- Monolith
- Microservices
- Serverless

REQUEST LIFECYCLE

Each request goes through certain steps while being

- Data parsing
- Data validation
- Permission validation
- Business logic execution
- Response data mapping

DIVISION OF RESPONSIBILITIES

- app.js - server composition
- Routes
- Controllers - data validation, response mapping
- Operations - business logic execution
- Repositories - permanent storage access layer

ROUTES

- Define routes and bind them with controllers

```
'use strict'

const Router = require('koa-router')
const articles = require('../controllers/articles')

const router = new Router()

router.get('/articles', articles.getAll)
router.get('/articles/:id', articles.getById)
router.post('/articles', articles.create)

module.exports = router.routes()
```

CONTROLLERS

- Parse request data
- Validate request data
- Call operation(s)
- Set response

```
'use strict'

const { validate } = require('./../validations')
const operations = require('./../operations/articles')
const schemas = require('./../validations/schemas/articles')

async function create(ctx) {
  const article = {
    title: ctx.request.body.title,
    content: ctx.request.body.content,
    image: ctx.request.body.image,
    tags: ctx.request.body.tags,
  }
  validate(schemas.article, article)
  ctx.body = await operations.create(ctx.request.body)
}

module.exports = {
  create,
}
```

OPERATIONS

- Perform business logic
- Throw errors

```
'use strict'

const errors = require('../utils/errors')
const articleRepository = require('./../repositories/articles')

function getById(input) {
  const article = articleRepository.findById(input.id)
  if (!article) {
    throw new errors.NotFoundError()
  }
  return article
}

module.exports = {
  getById,
}
```

REPOSITORIES

- Abstract from specific database/ORM
- Simplify database calls for operations

```
'use strict'

const R = require('ramda')
const articles = require('./../database/articles.json')

function findById(id) {
  return R.find(R.propEq('id', id), articles)
}

function create(article) {
  article.id = articles.length + 1
  articles.push(article)
  return article
}

module.exports = {
  findById,
  create,
}
```

DIVISION OF RESPONSIBILITIES

- Config - application-wide configuration
- Database - database models
- Middleware - logic concerning all requests
- Utils - you know, that other stuff ...
- Validations - validation schemas and compositions

CONFIG

- Default config
- Environment config
- Config composition

Validation

- Throw error on fail
- Schemas grouped by entity
- Schemas should be strict as possible

Errors

- Add more descendants
- Handle in middleware (it concerns all requests)
- Convert errors to error responses
- Handle in routes

QUESTIONS

STRV