Best Practices for designing RESTful API

Tue Nguyen - Oct 29, 2020

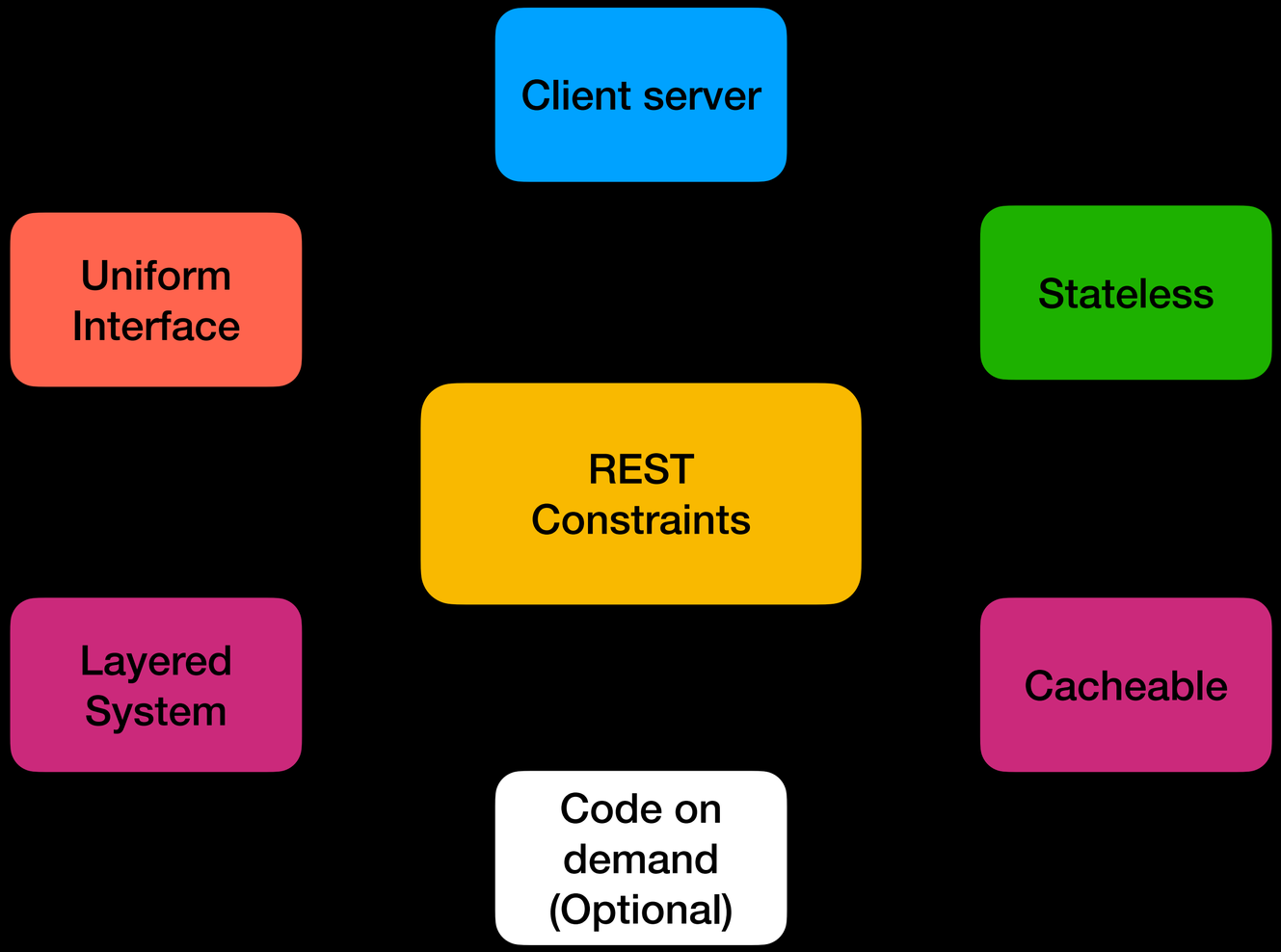
Agenda

1. Introduction
2. Conventions
3. Discussion

Introduction

Introduction

- REST: representational state transfer
- Basic terminology:
  - Resource
    - Collection
    - Instance/Singular
  - REST API or service
  - Resource identifier
  - Root resource
  - Sub-resource



Conventions

Endpoints

- API first development
- Behavior: HTTP methods
  - GET (SELECT)
  - POST (CREATE)
  - PUT (UPDATE)
  - PATCH (UPDATE)
  - DELETE (DELETE)
- Behavior should implement based on CRUD, unless there's a specific requirement otherwise

Conventions

Endpoints

- Use plural nouns
  - /employees
  - /departments
  - /products
- Use uniform endpoint for each functionality
- Don't use verbs:
  - /add_employee
  - /edit_employee
  - /delete_employee

Conventions

Endpoints

- GET /tickets - Retrieves a list of tickets
- GET /tickets/12 - Retrieves a specific ticket
- POST /tickets - Creates a new ticket
- PUT /tickets/12 - Update ticket #12
- PATCH /tickets/12 - Partially updates ticket #12
- DELETE /tickets/12 - Delete ticket #12

Conventions

Relations

- GET /tickets/12/messages - Retrieves list of messages for ticket #12
- GET /tickets/12/messages/5 - Retrieves message #5 for ticket #12
- POST /tickets/12/messages - Creates a new message in ticket #12

- GET /messages?ticketId=12 - Retrieves list of messages for ticket #12
- POST /messages - Creates a new message

Conventions

What about actions that don't fit into the world of CRUD operations?

Conventions

Actions

- PUT /tickets/:id/activate
- PUT /tickets/:id/deactivate

Conventions

Paging, Filtering, sorting & searching

- GET /tickets?state=open - query parameter that implements a filter
- GET /tickets?sort=-priority - retrieves a list of tickets in descending order of priority
- GET /tickets?q=return&stage=open&sort=-priority,createdAt - Retrieve the highest priority open tickets mentioning the word 'return'
- GET /tickets?limit=20&offset=10

Response

Status code

- 200 OK - [GET/PUT/PATCH/DELETE]
- 201 CREATED - [POST]
- 400 BAD REQUEST - client sent bad data to server
- 401 UNAUTHORIZED - client lacks valid authentication credential for the target resource
- 403 FORBIDDEN - When authentication succeeded but authenticated user doesn't have access to the resources
- 404 NOT FOUND - the client referenced a nonexistent resources or collection
- 429 TOO MANY REQUESTS - When a request is rejected due to rate limiting
- 500 INTENAL SERVER ERROR - The server encountered an unhandled error

Authentication - Authorization

Stateless authentication

- Access token: short live
- Refresh token: long live
- Logout / invalidate token?

Thank you and goodbye!!!