CS 4530
Software Engineering

Lecture 5 - HTTP + REST

Jonathan Bell, John Boyland, Mitch Wand
Khoury College of Computer Sciences

Zoom Mechanics

- Recording: This meeting is being recorded
- If you feel comfortable having your camera on, please do so! If not: a photo?
- I can see the zoom chat while lecturing, slack while you're in breakout rooms
- If you have a question or comment, please either:
  - "Raise hand" - I will call on you
  - Write "Q: <my question>" in chat - I will answer your question, and might mention your name and ask you a follow-up to make sure your question is addressed
  - Write "SQ: <my question>" in chat - I will answer your question, and not mention your name or expect you to respond verbally

Today’s Agenda

Administrative:

- HW1 Discussion, due **tomorrow at 10:00pm EST**
- Team formation out tomorrow, due next Friday

Today’s session:

- Lesson Q&A: Architecture, HTTP & REST
- Activity: Testing REST APIs with Postman
- Activity (Time permitting): REST transcript server extensions

TypeScript Warmup

```
const stringList: string[] = [];

function findStringStartingWith(prefix: string): string {
  return stringList.find(eachString => eachString.startsWith(prefix));
}
```

```
Type 'string | undefined' is not assignable to type 'string'.
  Type 'undefined' is not assignable to type 'string'.
```

```
const stringList: string[] = [];

function findStringStartingWith(prefix: string): string {
  const foundStr = stringList.find(eachString => eachString.startsWith(prefix));
  if (foundStr === undefined) {
    return 'String not found';
  }
  return foundStr;
}
```

No type errors (not sure that it makes sense to return the string ‘String not found’, but hopefully you get the idea...

Lesson 3.1 Learning Objectives

Software Architectures

- You should now be able to:
  - explain why software architecture is important
  - list a few of the properties that an architecture may have (the "ilities")
  - describe the basic ideas of the following architectures, with examples and pictures
    - monolithic
    - layered
    - pipeline
    - microkernel
    - event-driven
    - microservice

Lecture 3.2 Learning Objectives

HTTP Basics

- You should now be able to:
  - Explain the basic structure of the HTTP protocol
  - Define the following terms in the context of HTTP:
    - client, server
    - request, response
    - message header
    - message body
  - List the steps in the basic flow of an HTTP request
  - Explain what is meant by the following:
    - HTTP is stateless but not sessionless

```
GET /docs/index.html HTTP/1.1
Host: www.nowhere123.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
(blank line)
```

Lecture 3.3 Learning Objectives

REST Overview

- By the end of this lesson you should be able to:
  - Explain the basic principles of RESTful protocols
  - Examine a protocol and suggest ways in which it either adheres to or violates the REST principles.

- Anti-examples:
  - `/getCity/losangeles`
  - `/getCitybyID/50654`
  - `/Cities.php?id=50654`

Example REST API

```
POST /transcripts
 -- adds a new student to the database,
 -- returns an ID for this student.
 -- requires a body parameter 'name', url-encoded (eg name=avery)
 -- Multiple students may have the same name.
GET  /transcripts/:ID
 -- returns transcript for student with given ID.  Fails if no such student
DELETE /transcripts/:ID
 -- deletes transcript for student with the given ID, fails if no such student
POST /transcripts/:studentID/:courseNumber
 -- adds an entry in this student's transcript with given name and course.
 -- Requires a body parameter 'grade', url-encoded
 -- Fails if there is already an entry for this course in the student's transcript
GET  /transcripts/:studentID/:courseNumber
 -- returns the student's grade in the specified course.
 -- Fails if student or course is missing.
GET  /studentids?name=string
 -- returns list of IDs for student with the given name
```

Lesson 3.4 Learning Objectives

REST Implementation

- You should now be prepared to:
  - Explain the structure of a server in express.js
  - Define 'middleware' and 'route' in the context of an express.js server
  - Build a server for a simple REST protocol in express.js

Activity: Postman

REST API Testing

https://neu-se.github.io/CS4530-CS5500-Spring-2021/tutorials/week3-apis

This work is licensed under a Creative Commons Attribution-ShareAlike license