HTTP and Web Servers

* 10,000,000 feet high description of the internet
* The internet is basically computers messaging each other
* The primary way computers on the internet message each other is via HTTP
* HTTP (Hyper Text Transfer Protocol)
  + Request response messaging system
  + You make an HTTP request and you get back a HTTP response
  + This is always 100% true
* Web Server
  + Any computer that can handle HTTP requests.
  + Can be written in any programming language.
    - Java
    - Go
    - Rust
    - Node.js (JavaScript)
    - Python
* Javalin
  + Java Dependency for creating web servers.
  + Similar to express in Node.js.
  + Event driven.
    - Each http request is an event which triggers a lambda function in response.

The problem

* You have an internet full of web servers written in a dozen different programming languages.
* You want to make sure that they can all communicate with each other.
* Every programming language has different data types.
  + Custom classes
  + Some languages only have number type.
  + Some have a bunch
    - Java has an int and a double/ float
    - C has signed unsigned numbers

The Solution

* Universal format that any programming language could understand.
* Strings are a data type found in essentially every single programming language.
* Strings needed a universal format that any programming language could parse into it’s own objects
  + It is impossible for a Java byte code object in the JVM to be correctly interpreted by Python.
* JSON (JavaScript Object Notation)
  + De Facto string format for sending information on the web
  + Based on JavaScript objects
  + IT IS JUST A FORMATTED STRING

name = Adam Ranieri

age = 19

profession = Trainer

=>

JSON format of the above information

numbers do not get quotation marks

key value pairs

{"name":"Adam Ranieri","age":19, "profession":"trainer"}

REST

* REST (REpresentational State Transfer).
* Type of Web Server API
  + Application Program interface
* Theoretically a web server could responds to http requests in anyway it wants.
  + You could set up routes however you wanted
  + Return whatever you wanted.
  + NOT A GOOD IDEA
* RESTful web server (service)
  + Web Server API that follows a specific convention and standard.
  + Standardizes how we can communicate over the web.
  + The most common top of web server.
* Important features
  + Uses HTTP or HTTPS
  + REST is based on RESOURCES (entities)
    - A Resource is a collection of related objects.
    - These objects/resources are accessed via purposefully named URI (Uniform RESOURCE interface) paths.
    - HTTP verbs indicate what is happening to that resource.
* Example REST API
  + A company might have a bunch of employees.
  + HTTP GET /employees
    - Return all employees at the company.
  + HTTP POST /employees
    - Create a new employee.
  + HTTP GET /employees/ 35
    - Return 1 employee whose ID is 35
  + HTTP PUT /employees/41
    - Perform an UPDATE on employee whose ID is 41
  + HTTP DELETE /employees/54
    - Delete the employee whose ID is 54
* Status Codes
  + Web servers and REST applications are supposed to be language agnostic
  + Errors should be language agnostic
    - You do not want to send the person using your API a Java Error code or exception.
  + Simple numeric identifier of the success or failure of a http response
    - 100’s Information
    - 200’s Success
      * 201 : You successfully created something
    - 300’s redirects
    - 400’s Client side errors
      * 404 : You requested something that does NOT exist
    - 500’s server side errors