CMSC 240 Lecture 17

CMSC 240 Software Systems Development
Fall 2023

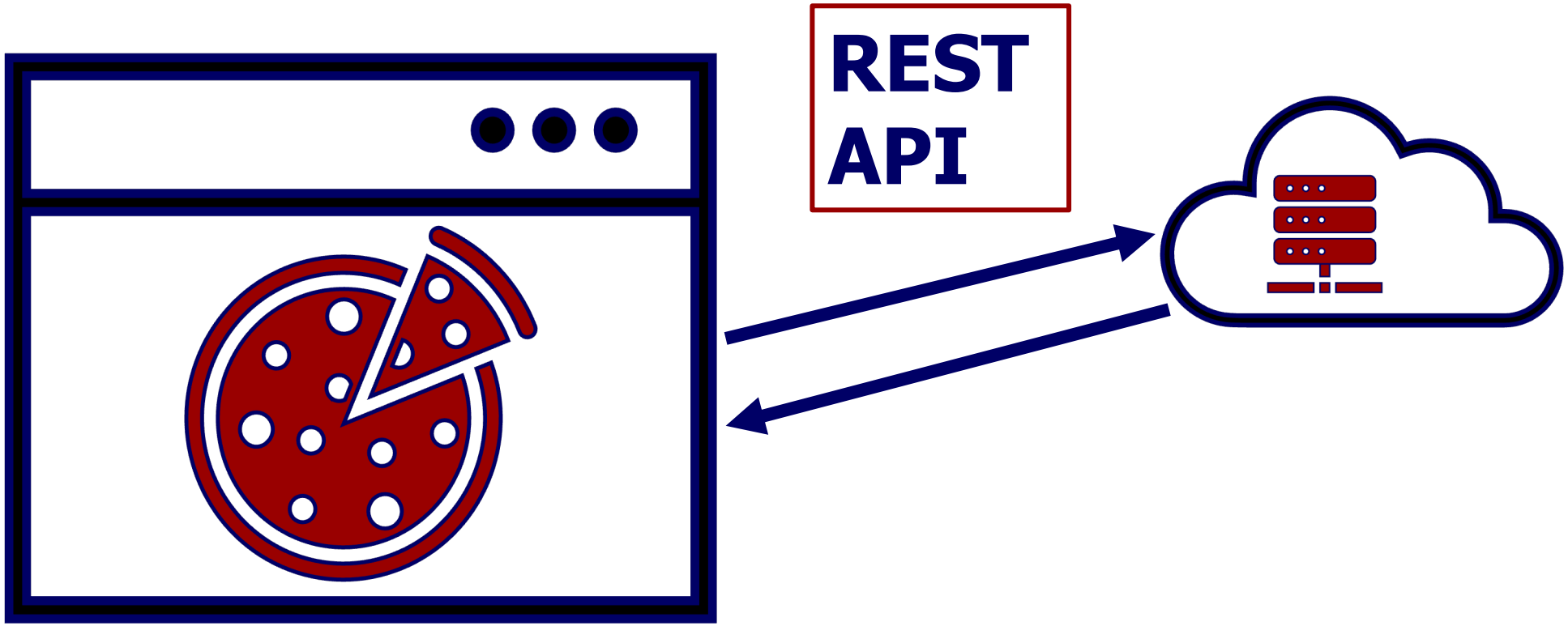
Today – REST APIs

- Project Introduction
- REST APIs
- In-Class Exercises

RICHMOND

REST API

- **Re**presentational **S**tate **T**ransfer
  - Communication between client and server "It's how they talk"
  - "RESTful" web service

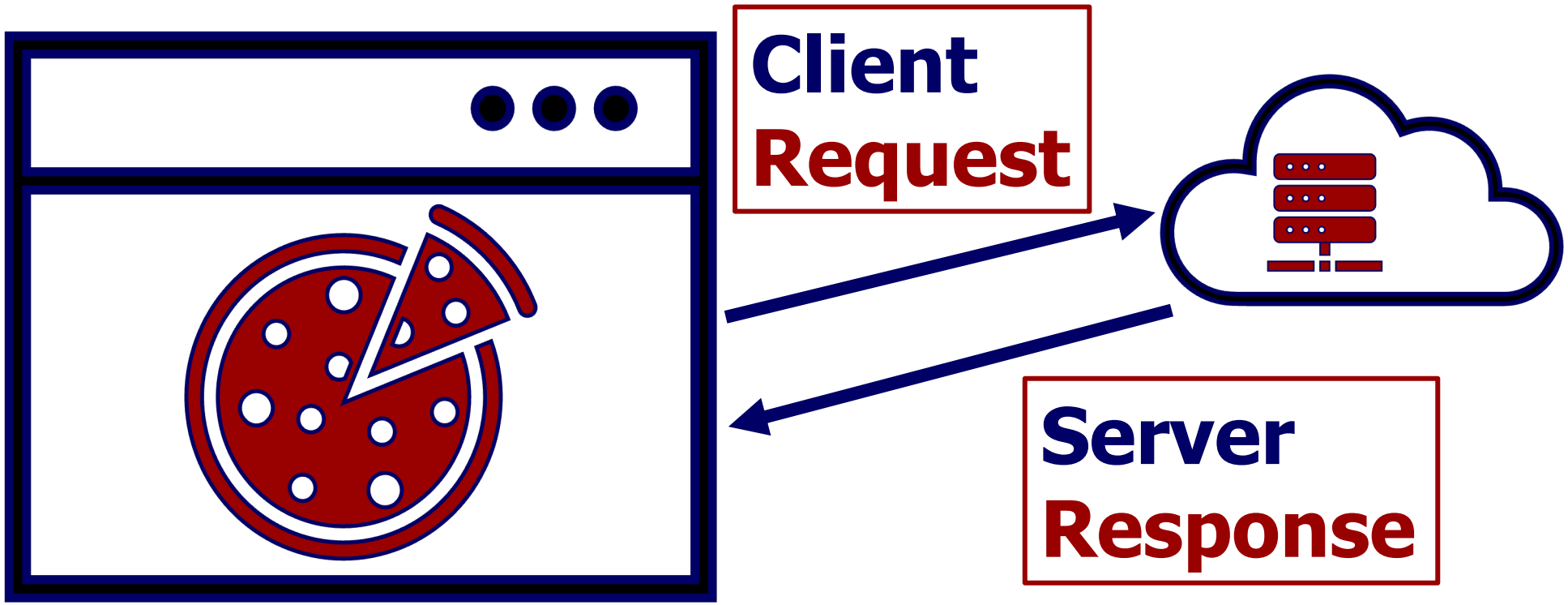


REST API

- Benefits of REST
  - Simple
  - Standardized
  - Scalable
  - Stateless
  - High Performance

http://urpizza.com/api/toppings

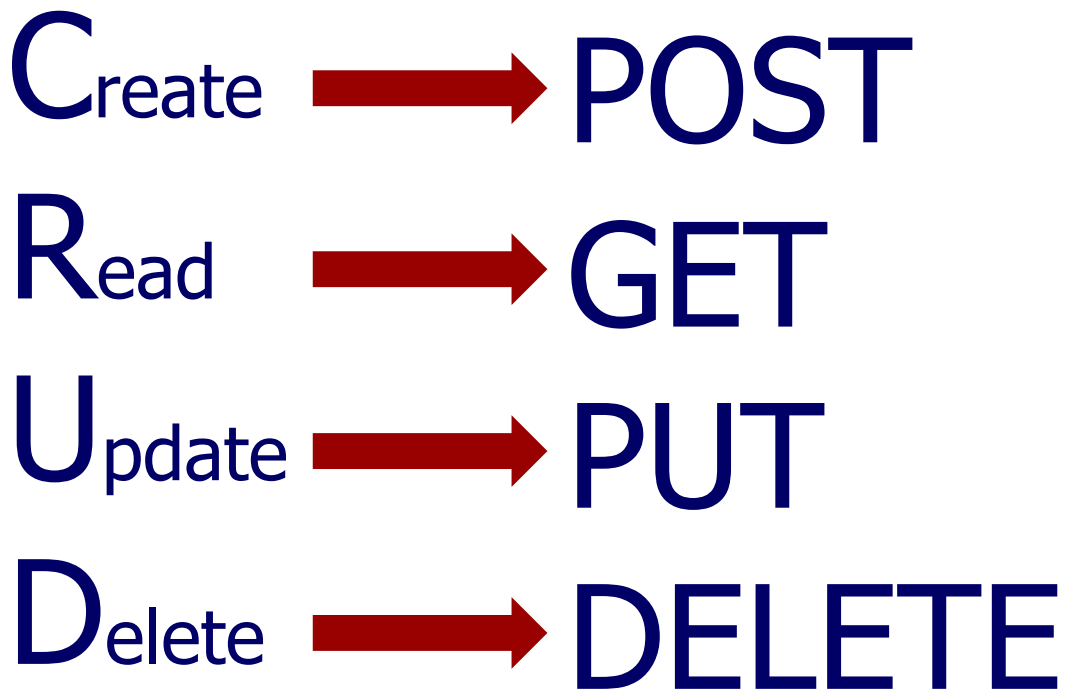
Resource



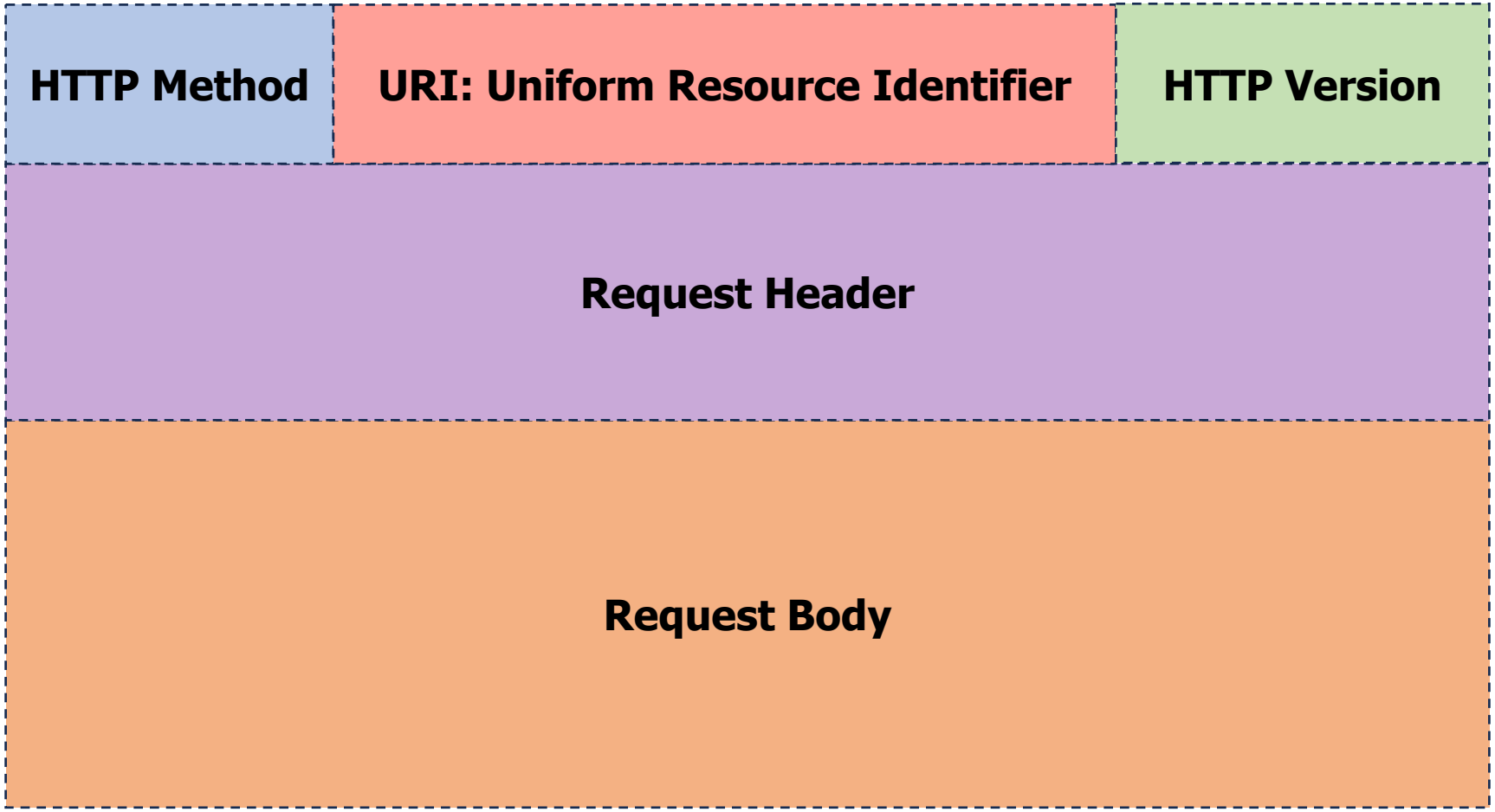
Request

What actions (verbs) would you want to perform on your resource?

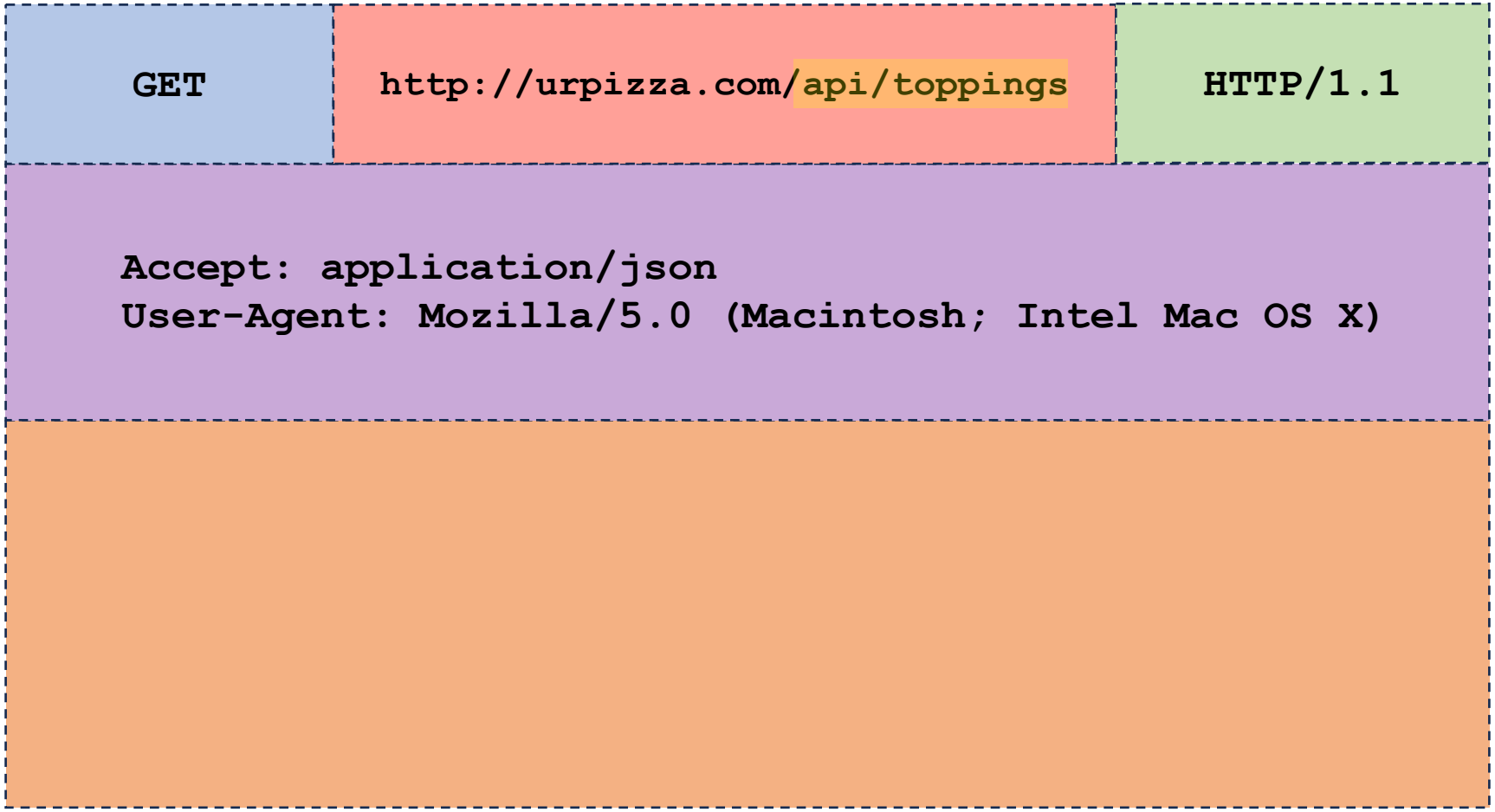
HTTP Methods/Operations



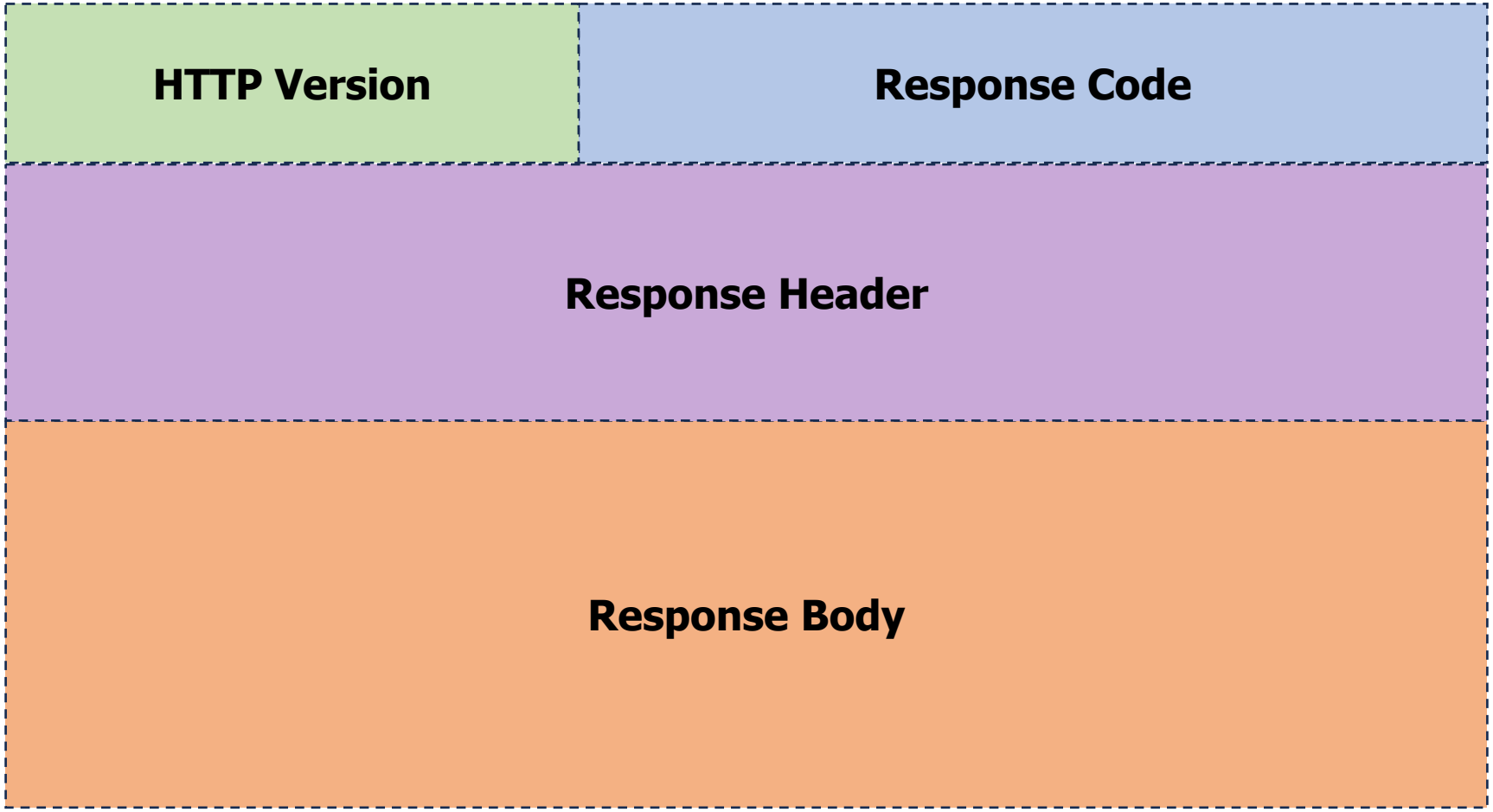
HTTP Request



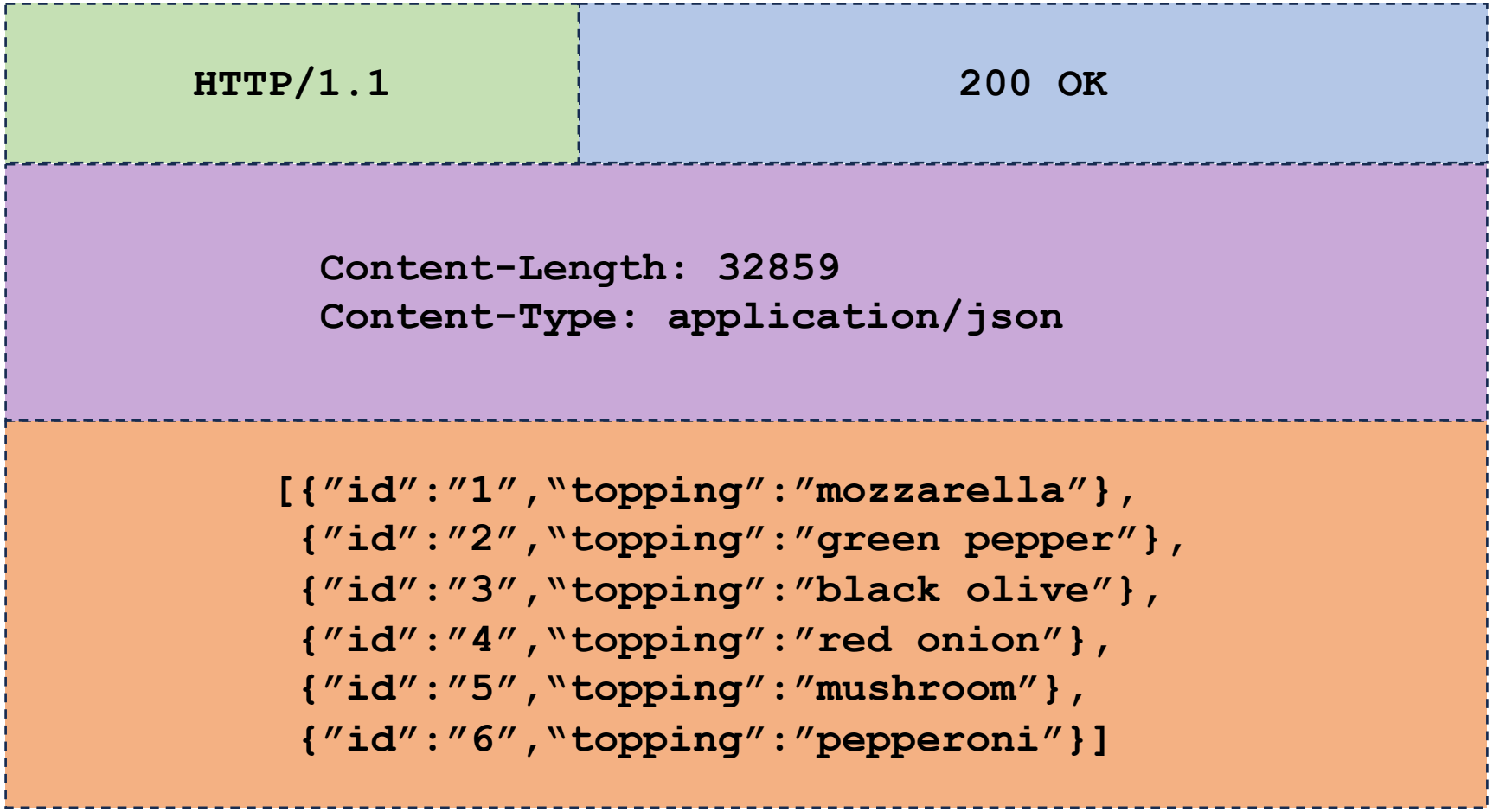
GET Request
http://urpizza.com/api/toppings



HTTP Response



GET Response
http://urpizza.com/api/toppings



JSON (**J**ava **S**cript **O**bject **N**otation)

- JSON Syntax Rules
  - Data is in name/value pairs
  - Data is separated by commas
  - Curly braces hold objects
  - Square brackets hold arrays

```
{
  "array": [
    1,
    2,
    3
  ],
  "boolean": true,
  "string": "Hello World",
  "null": null,
  "number": 123,
  "object": {
    "a": "b",
    "c": "d"
  }
}
```

HTTP Response Codes For Success

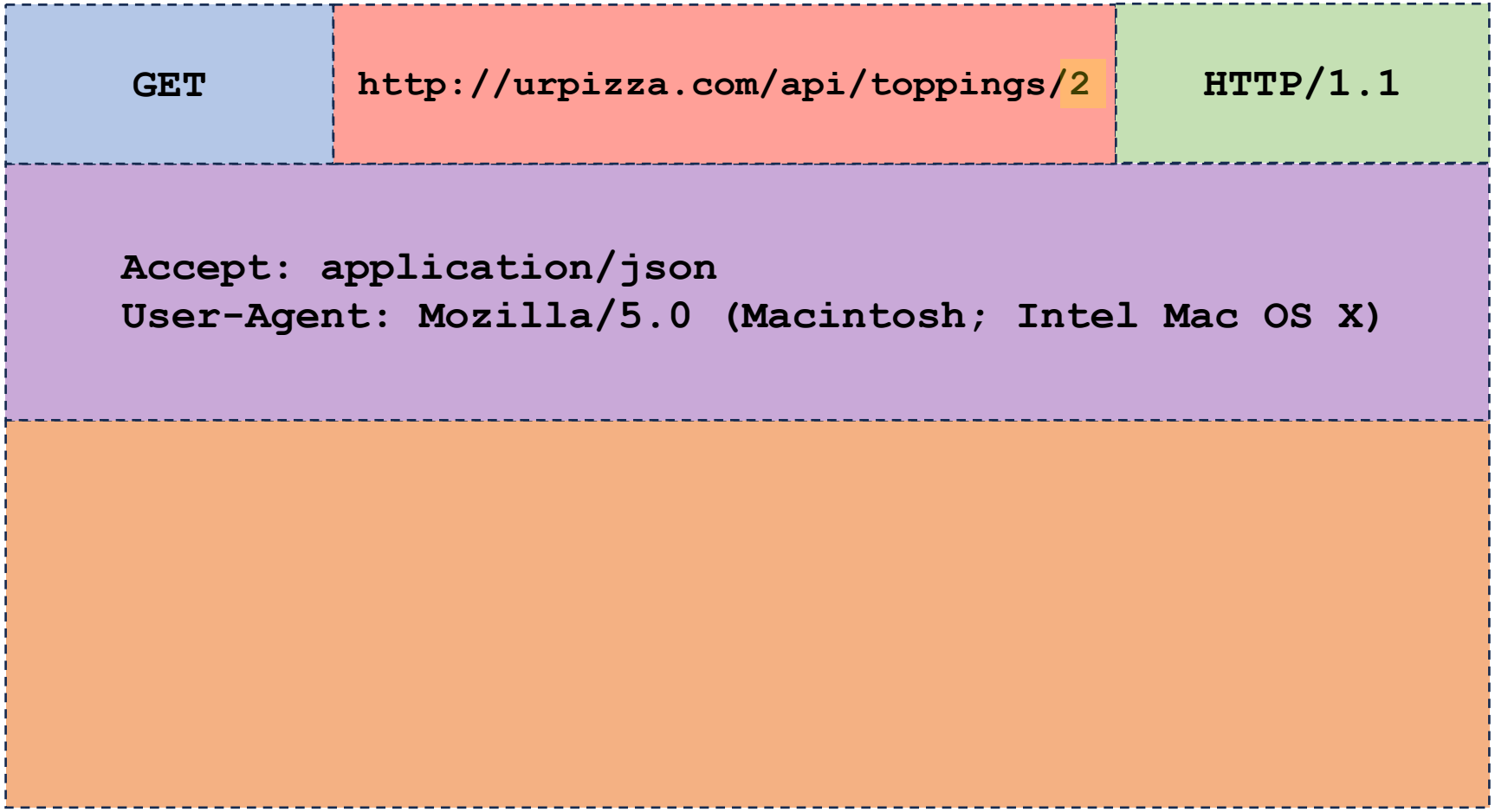
- 2xx success
  - 200 OK
    - Standard response for successful HTTP requests
    - Use for successful GET and PUT requests
  - 201 Created
    - The request has been fulfilled, resulting in the creation of a new resource
    - Use for successful POST requests
  - 204 No Content
    - The server successfully processed the request, and is not returning any content
    - Use for successful DELETE requests

HTTP Response Codes For Client Errors

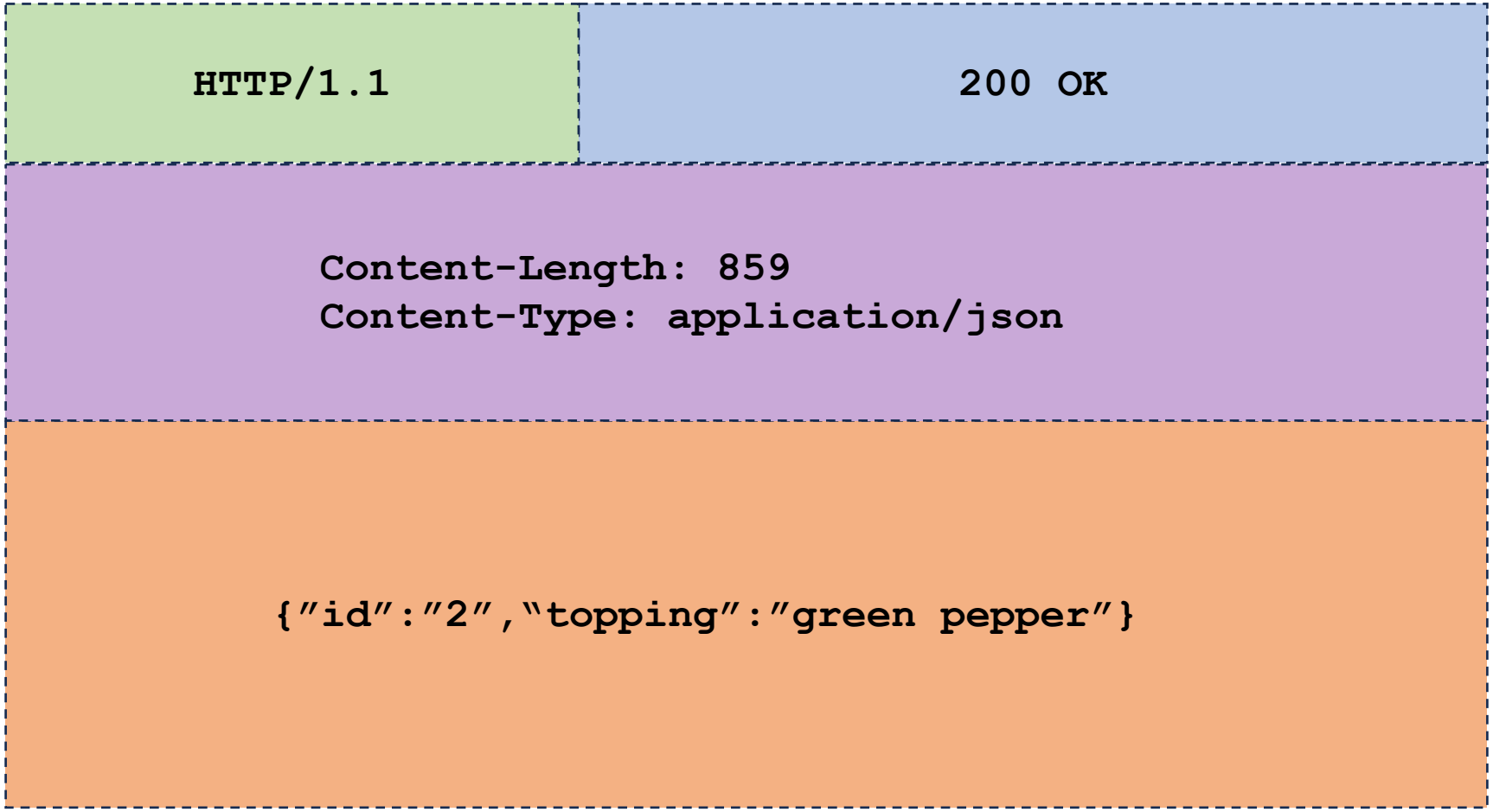
- 4xx client errors
  - 400 Bad Request
    - The server cannot or will not process the request due to an apparent client error e.g., malformed request syntax, size too large, invalid request message
    - Use for unsuccessful POST and PUT requests when JSON parsing fails
  - 401 Unauthorized
    - When authentication is required and has failed or has not yet been provided
  - 404 Not Found
    - The requested resource could not be found
    - Use for unsuccessful GET, PUT, and DELETE requests when resource is not found
  - 418 I'm a teapot
    - The server refuses to brew coffee because it is, permanently, a teapot

  - https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

GET Request
http://urpizza.com/api/toppings

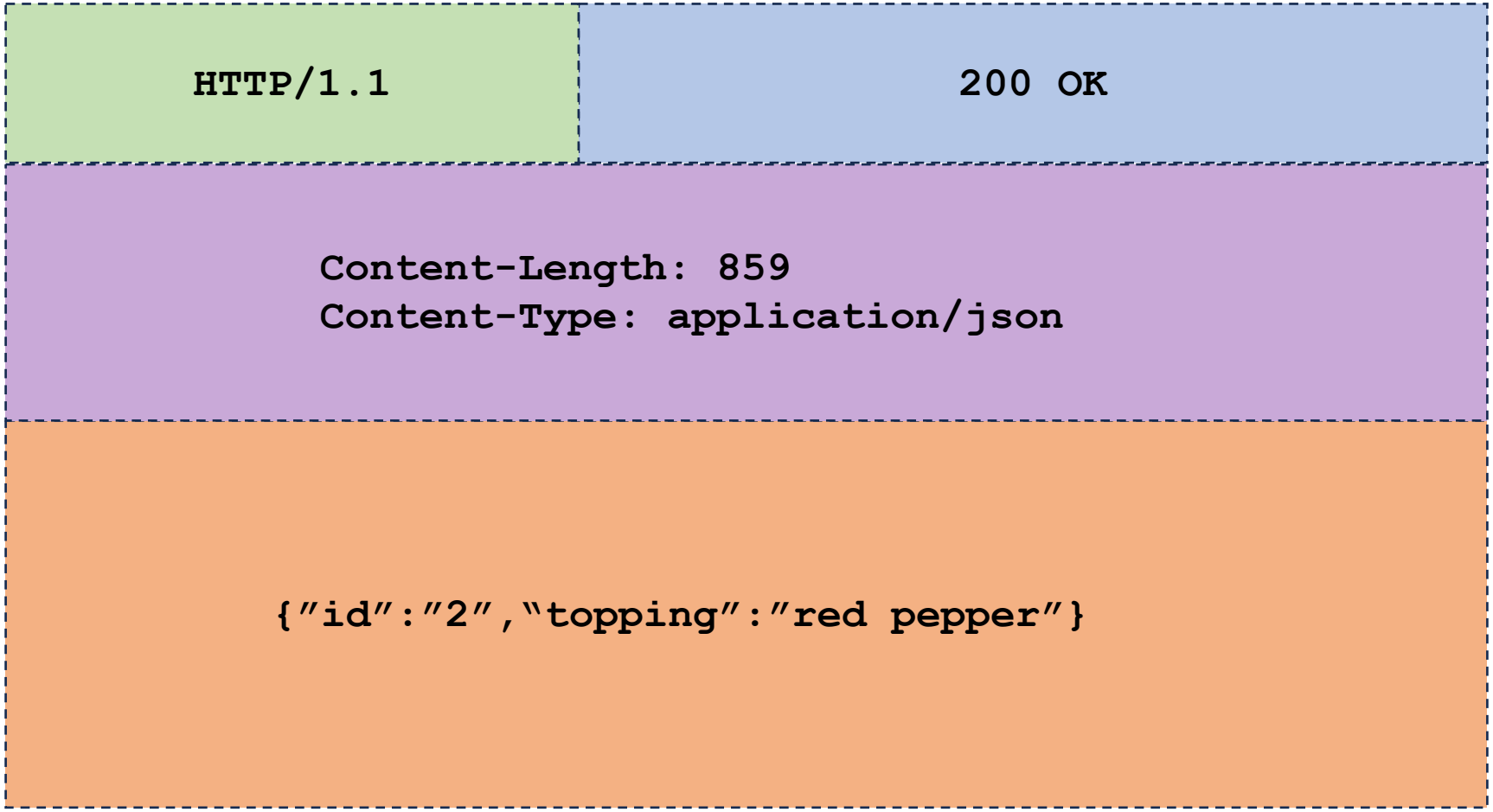


GET Response
http://urpizza.com/api/toppings

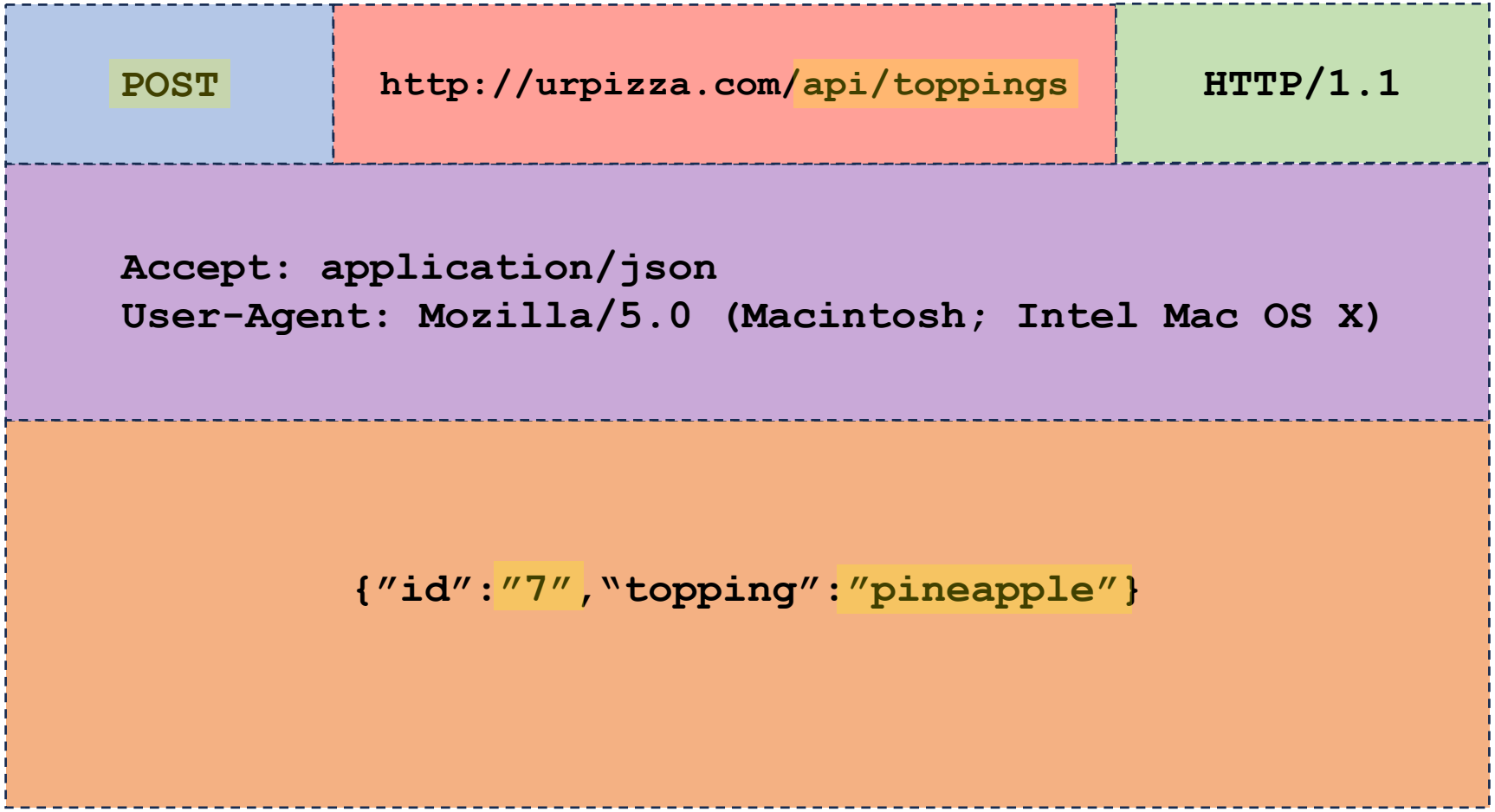


PUT Request
http://urpizza.com/api/toppings

PUT Response
http://urpizza.com/api/toppings

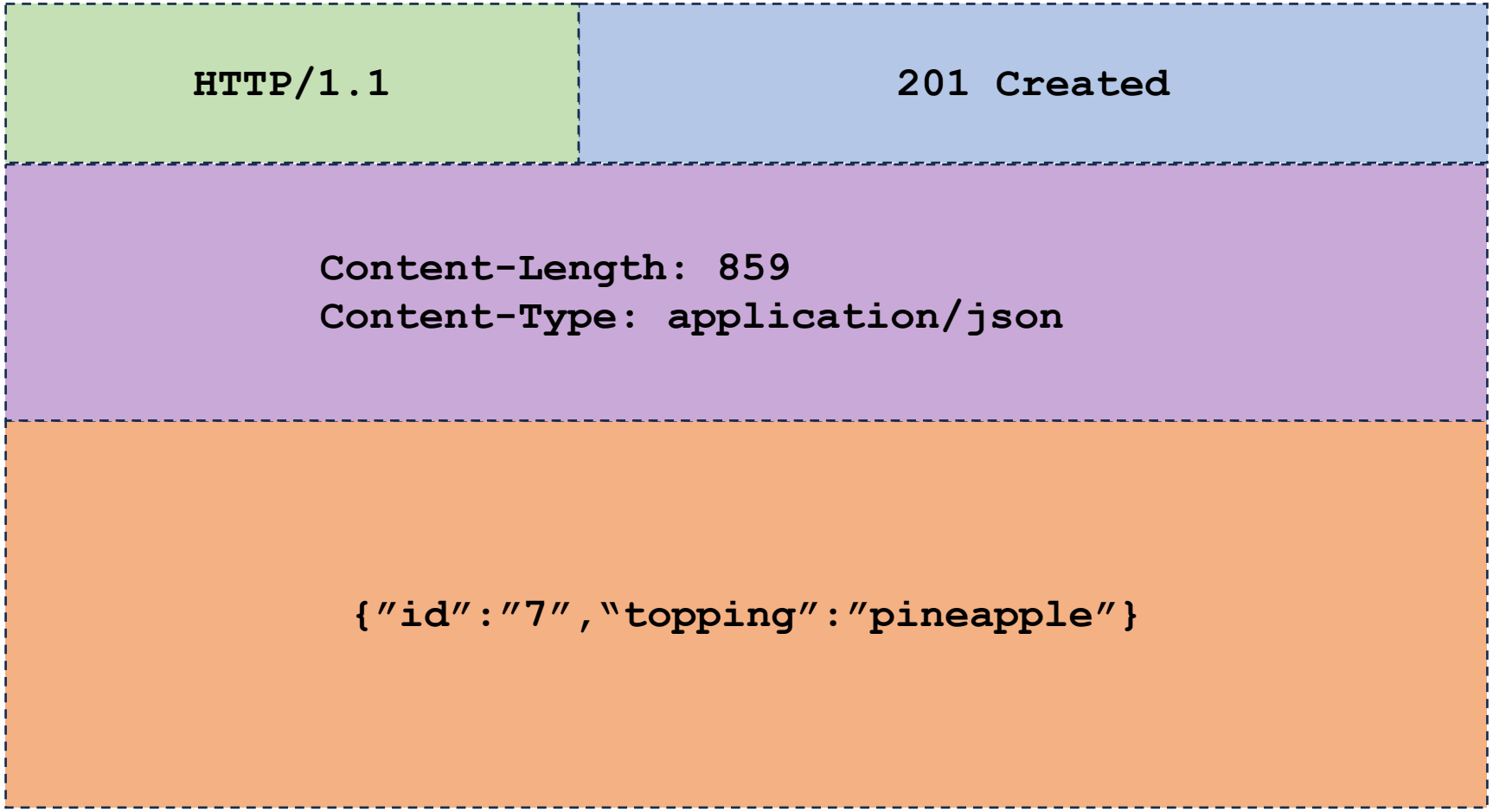


POST Request
http://urpizza.com/api/toppings

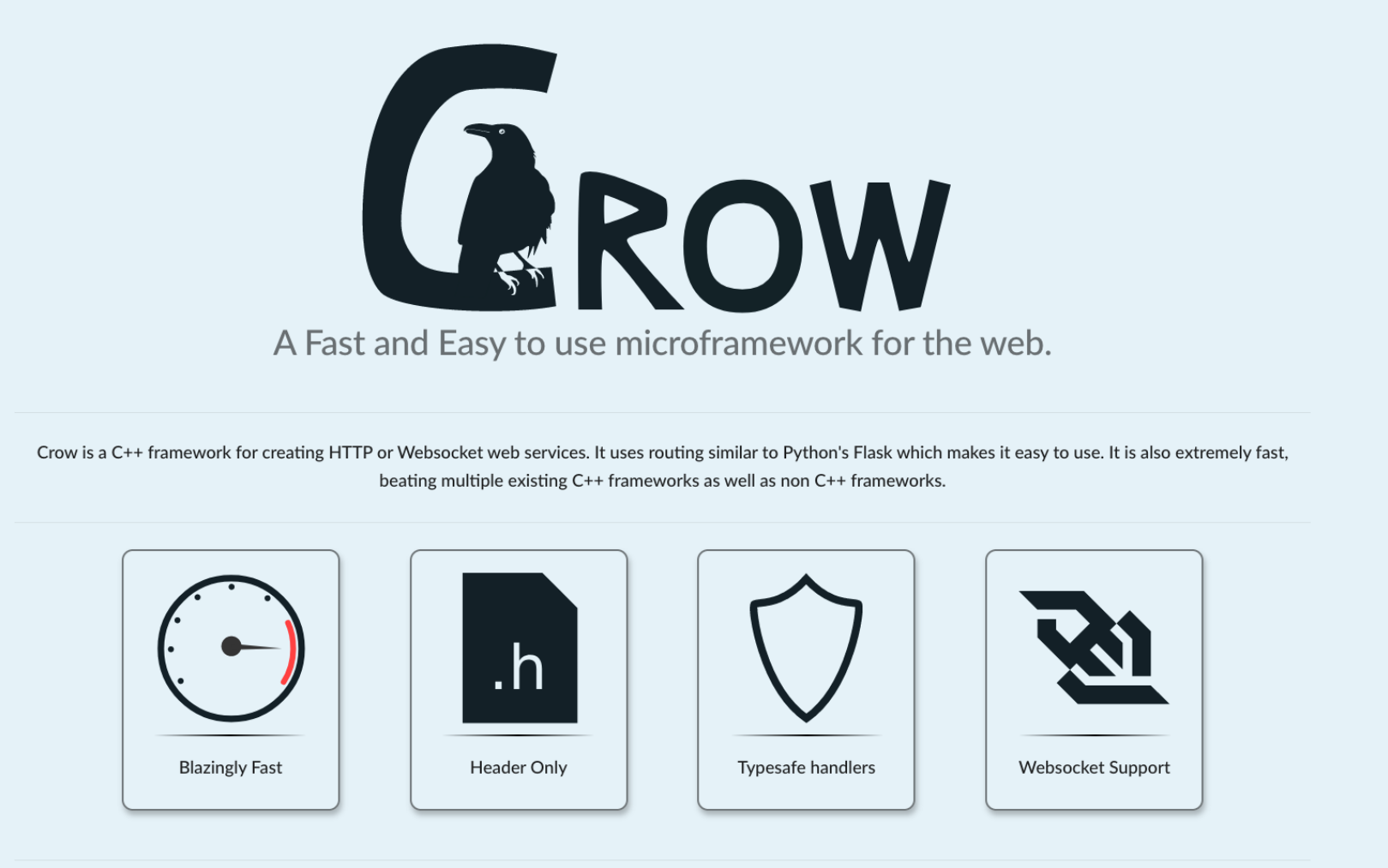


POST Response

http://urpizza.com/api/toppings



C++ Library for RESTful web service



Hello REST

```cpp
#include <crow.h>
#include <string>
using namespace std;
using namespace crow;

response handleHelloGetRequest()
{
    string body = "Hello, world!";

    // Return the integer HTTP response code, and the response body as a string.
    return response(200, body);
}

int main()
{
    // Create a simple crow application.
    SimpleApp app;

    // Define your endpoint and provide a function to handle the request.
    CROW_ROUTE(app, "/api/hello").methods(HTTPMethod::Get)(handleHelloGetRequest);

    // Set the port and run the app.
    app.port(18080).run();
}
```