Web Application Components

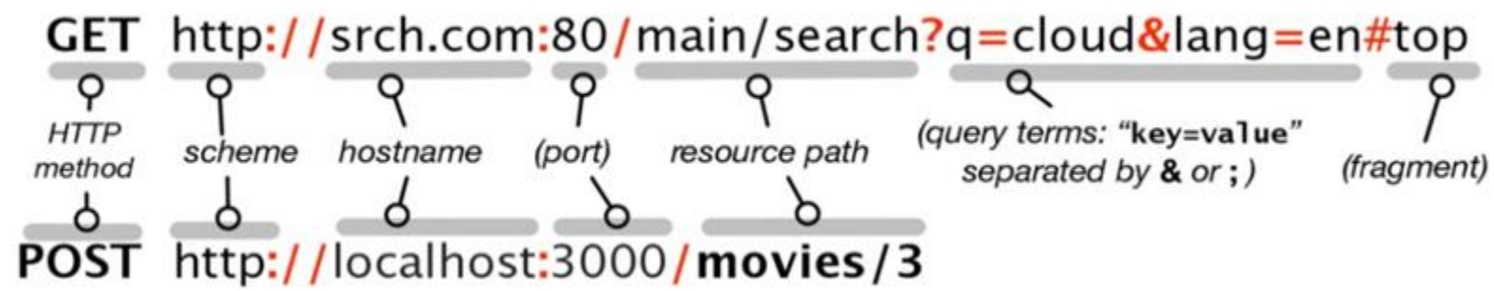
CPEN320

What is the Client/Server architecture?

HTTP Request and Response?

- What are they? What do they do?

HTTP Request



HTTP request = Method + **Scheme** + **URL**

- **Method**: GET
- **Scheme:** http
- **URL: srch.com:80/main/searh?q=cloud&lang=en**

Note: the fragment part of the URL (#) is not sent to the browser.

HTTP Request

Method URL Protocol Version

```
GET /main/searh?q=cloud&lang=en HTTP/1.1
Host: srch.com:80
User-Agent: Mozilla/5.0
Accept: text/html, */*
Accept-Language: en-us
Accept-Charset: ISO-8859-1,utf-8
Connection: keep-alive
```

Headers

blank line

Body (optional)

HTTP Response

Version Status Status Message

Headers

Body

```
HTTP/1.1 200 OK
Date: Thu, 11 Sep 2024 12:36:27 GMT
Server: Apache-Coyote/1.1
Content-Type: text/html;charset=UTF-8
Content-Length: 1846
blank line
<html>
...
</html>
```

HTTP response status codes

200 OK

- request succeeded, requested object later in this message

HTTP response status codes

200 OK

- request succeeded, requested object later in this message

301 Moved Permanently

- requested object moved, new location specified later in this message (Location:)

400 Bad Request

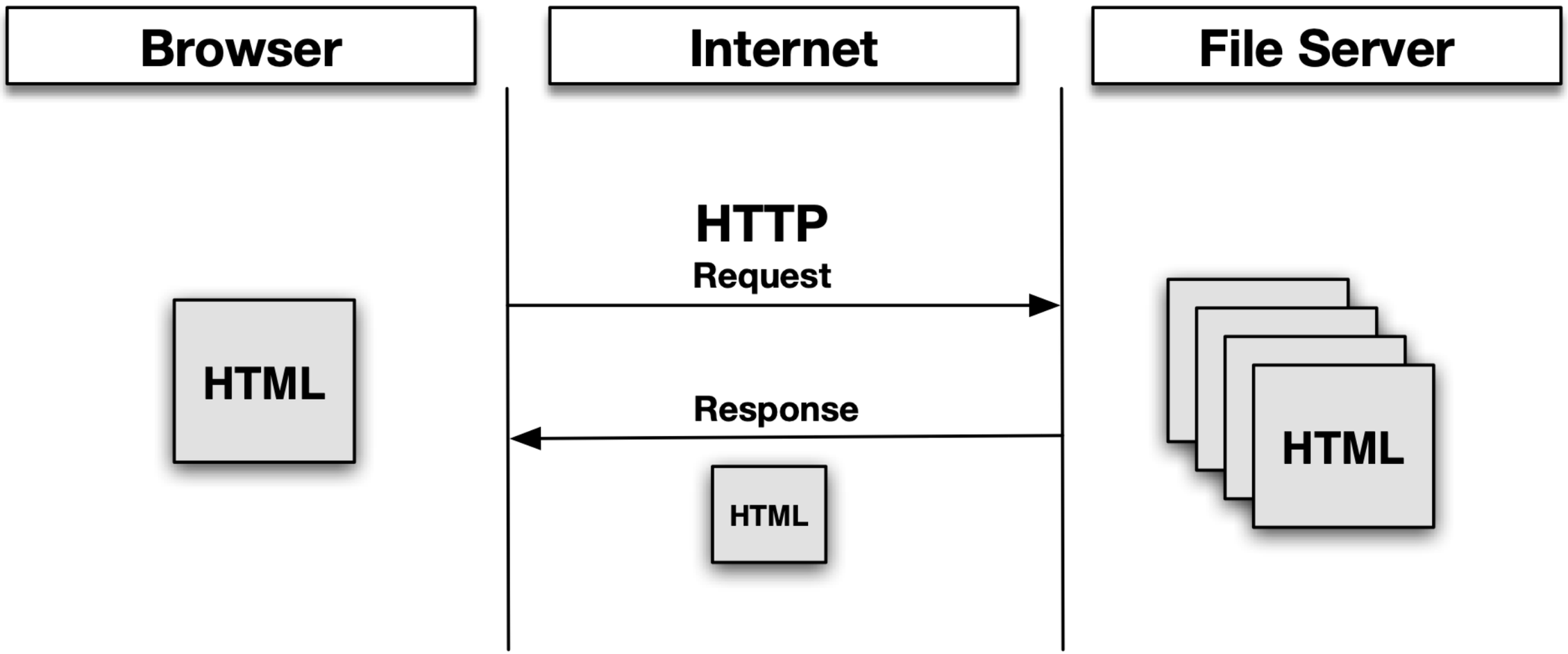
- request message not understood by server e.g. http://example.com/%%20badurl or ?sort=wrongvalue

404 Not Found

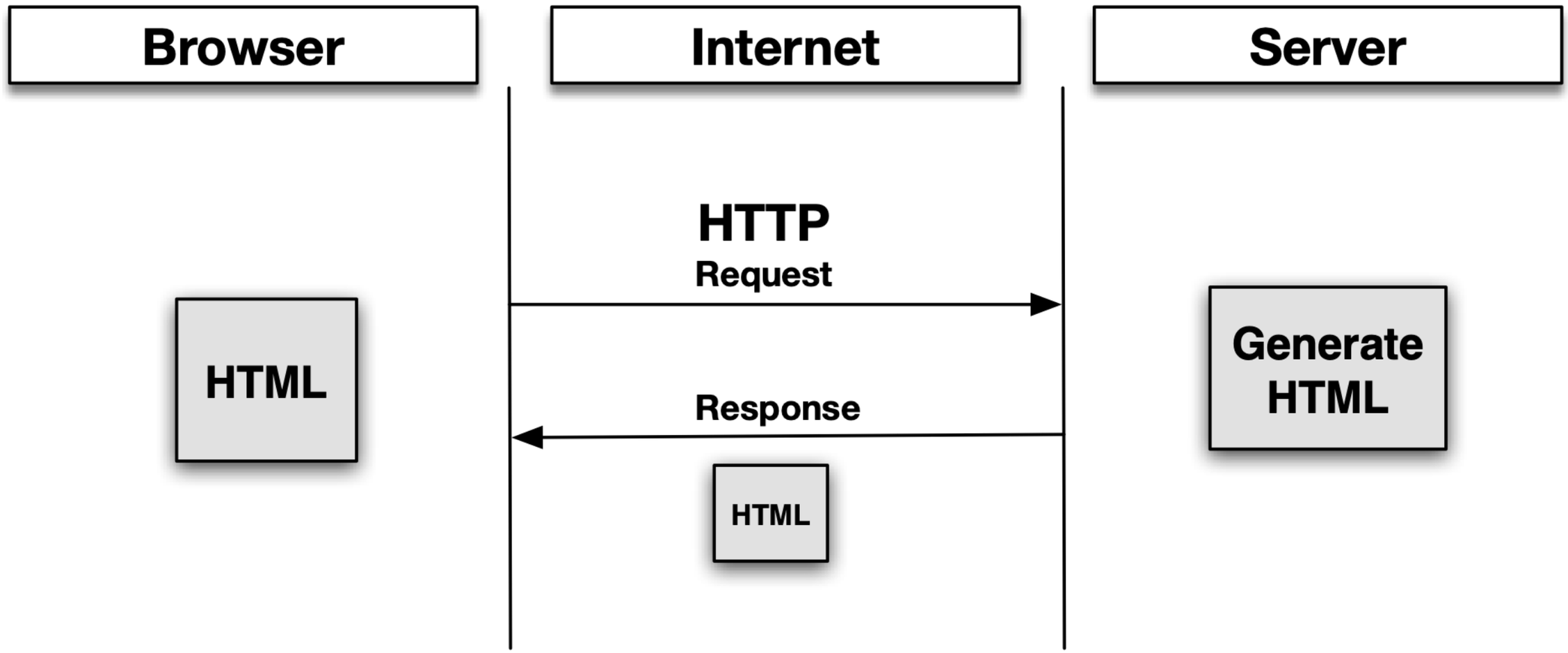
- requested document not found on this server

505 HTTP Version Not Supported

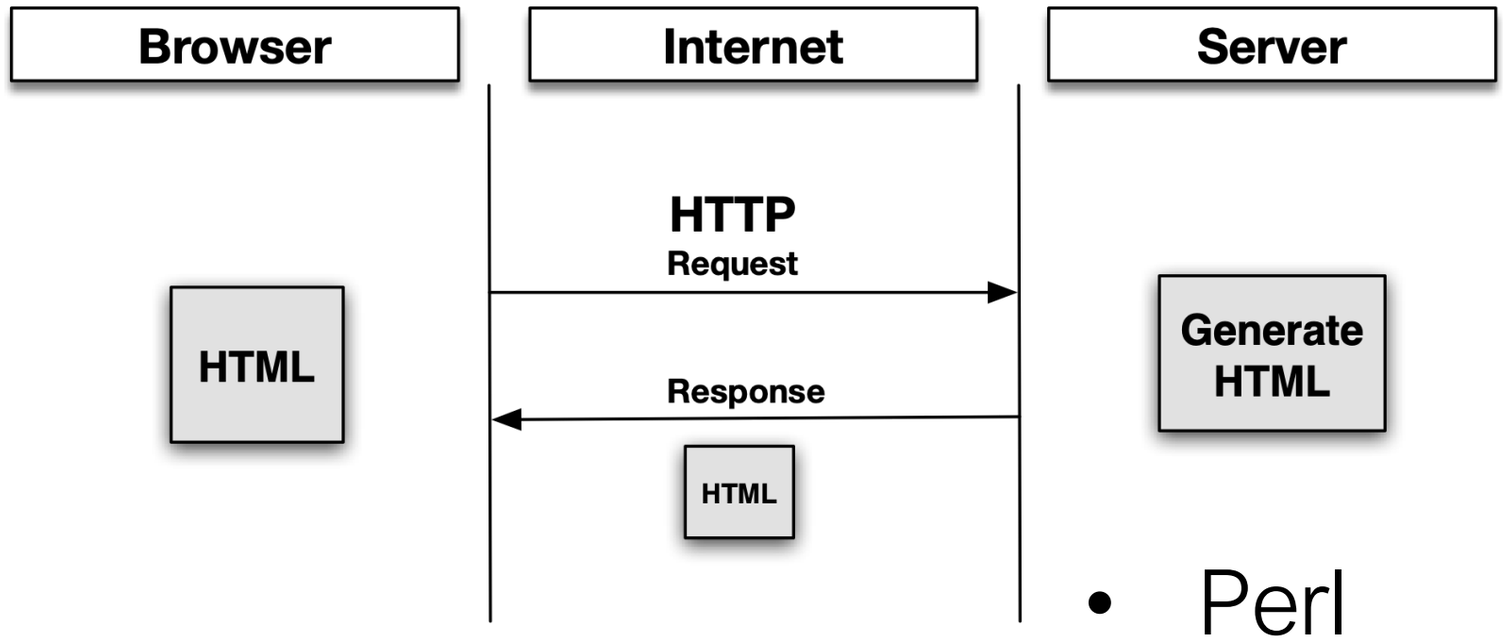
Static HTML Web Pages



DYNAMIC WEBSITES

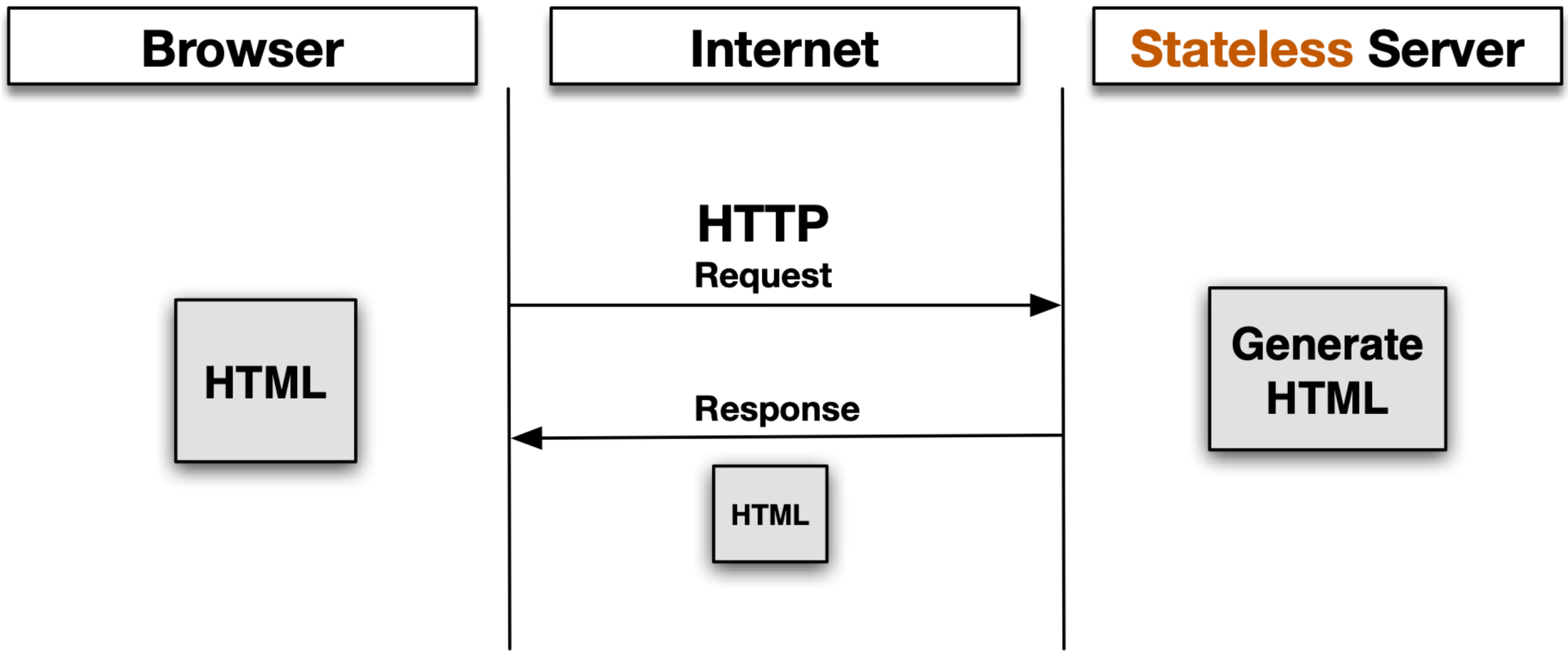


Dynamic Websites



- Perl
- PHP
- JSP/J2EE
- ASP
- Python
- …

RESTFUL WEBSITES



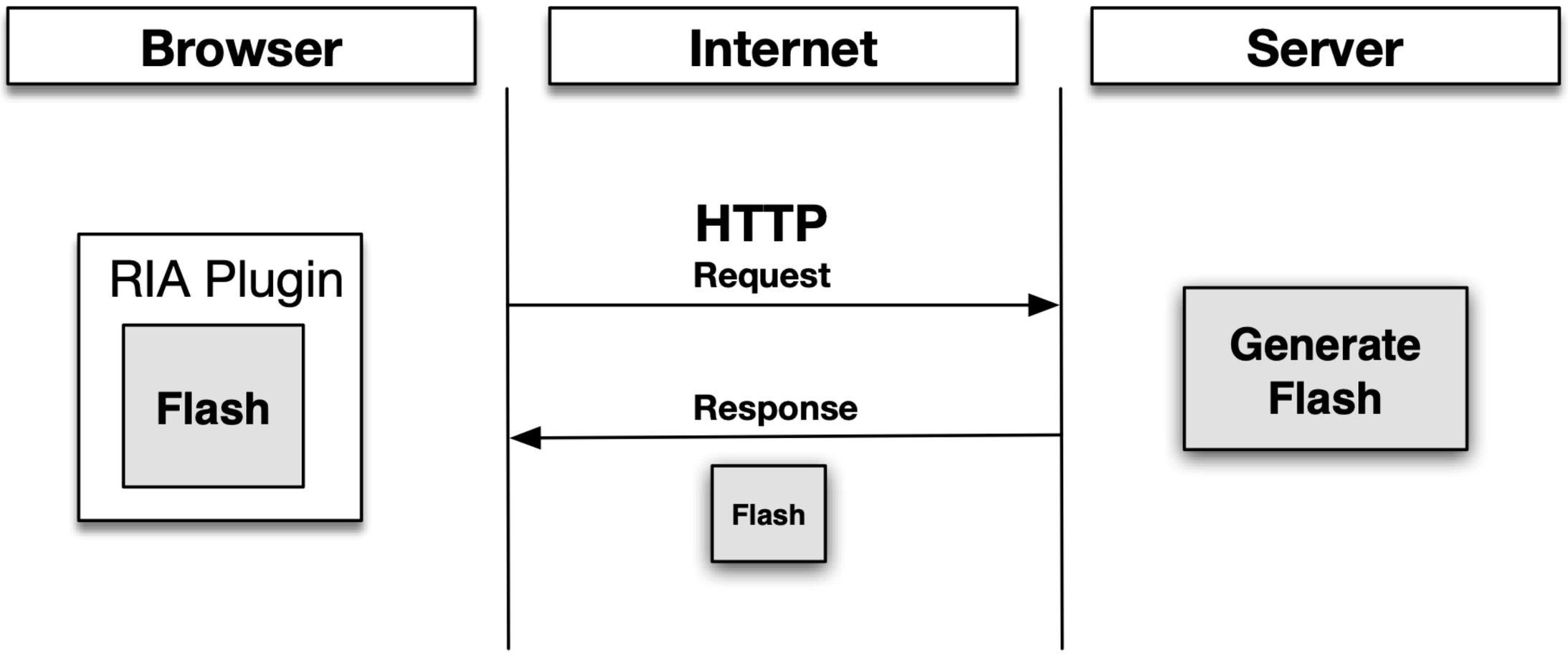
"Principled design of the modern Web architecture", Fielding and Tylor, ICSE 2000

REST APIs

Summary: RESTful APIs (Representational State Transfer)

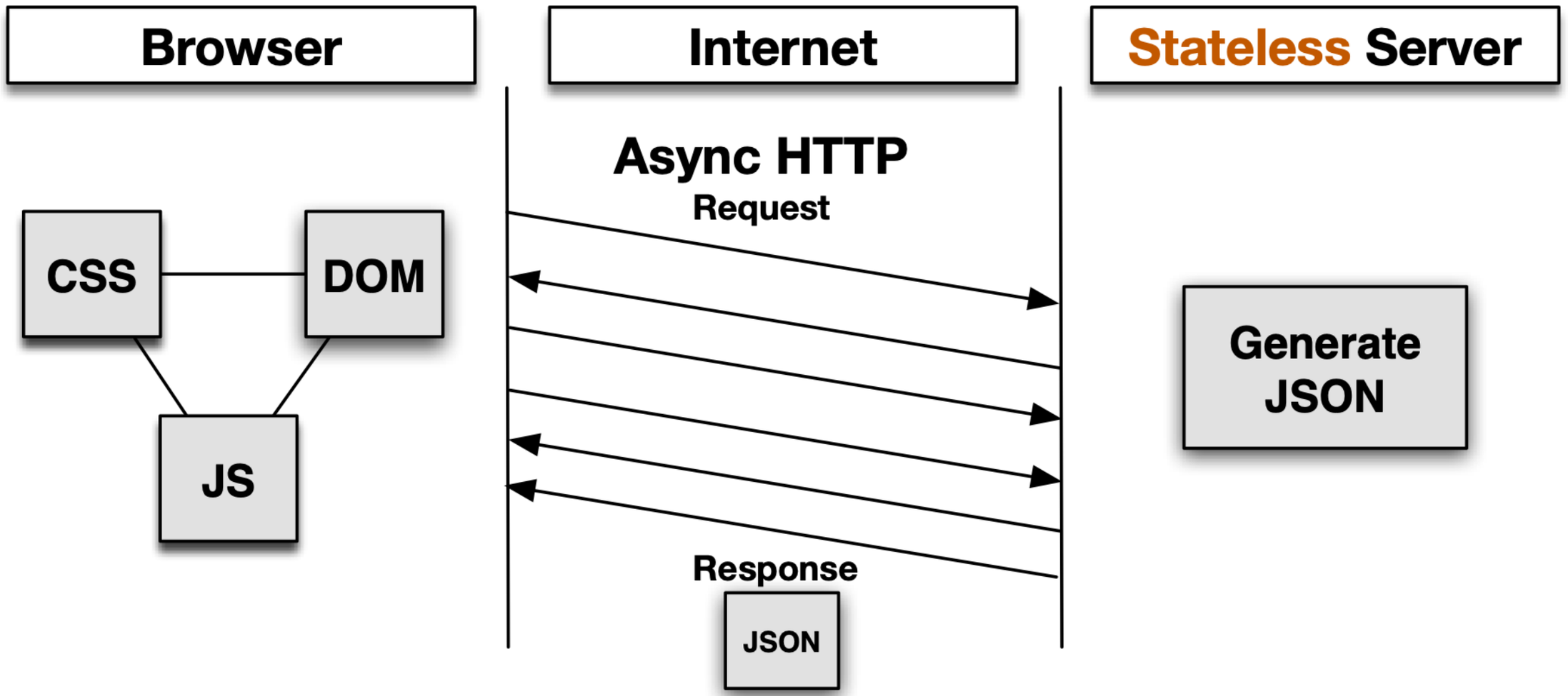
- *Everything is a resource.*
  - Basic ops: create, read, update, delete, list
  - Awkward to express desired operation?
    ➔ maybe new resource needs to be defined
- RESTful operation identifies resource, op to be done/side effects, and any other data needed to complete operation
  - resource relationship (eg A "owns" B) often reflected in route, e.g. `GET /a/:a_id/b/:b_id`, even if *b_id* uniquely identifies resource
- Resource can have different representations
  - Displayable HTML page for human user
  - JSON or XML data structure
  - hence *representational* in REST
- REST "won" because it's simple and matches engineering constraints of how Web evolved, e.g. stateless HTTP

RICH INTERNET APPS



MODERN WEB APPS

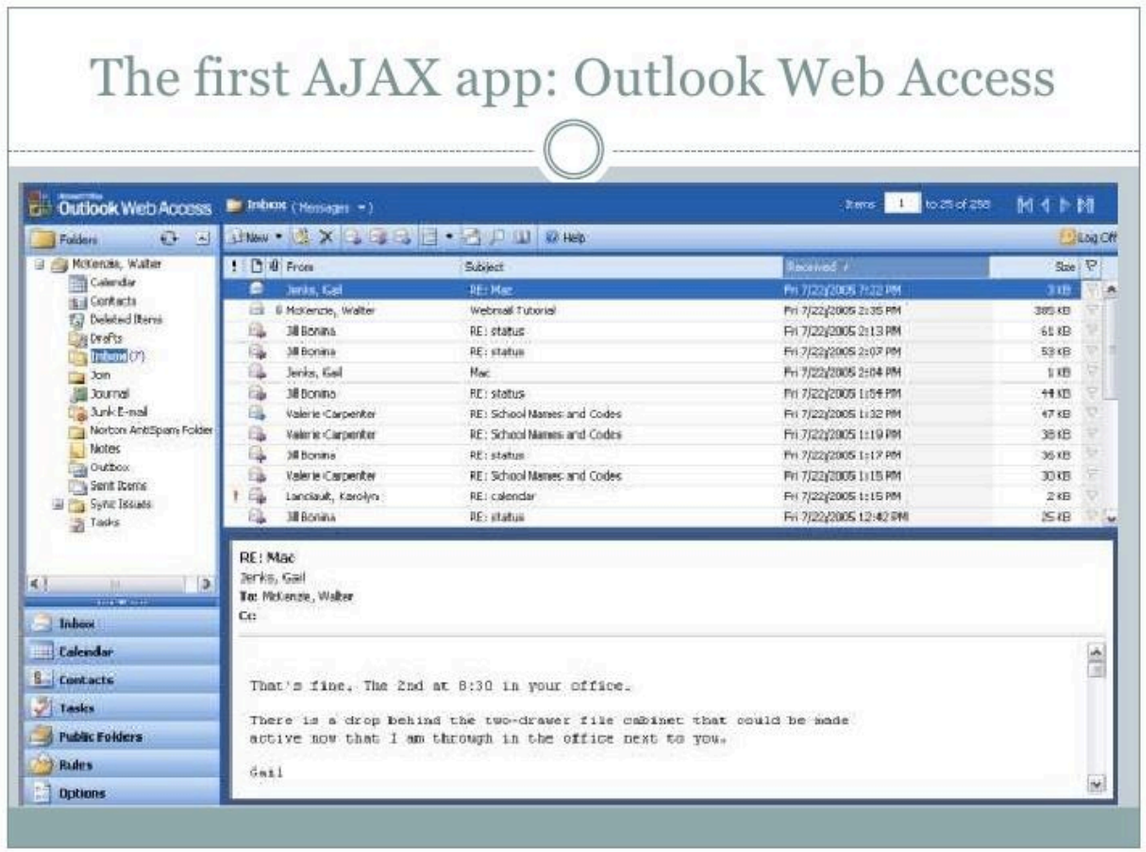
Web Standard-based



2005-2009: The Rise of AJAX

- AJAX (Asynchronous JAvaScript and XML)
  - Feature introduced by Microsoft IE in their Outlook Web Access (OWA) client in the early 2000s
  - Google used it for Google Maps and Gmail (2004), and suddenly it was all the rage (2005)
  - JavaScript became the new popular kid on the block, and JavaScript performance started to become increasingly important with Mozilla taking the lead
  - Google introduced Chrome in 2008 which was primarily about faster JavaScript execution and support – based on Webkit development engine

AJAX Applications

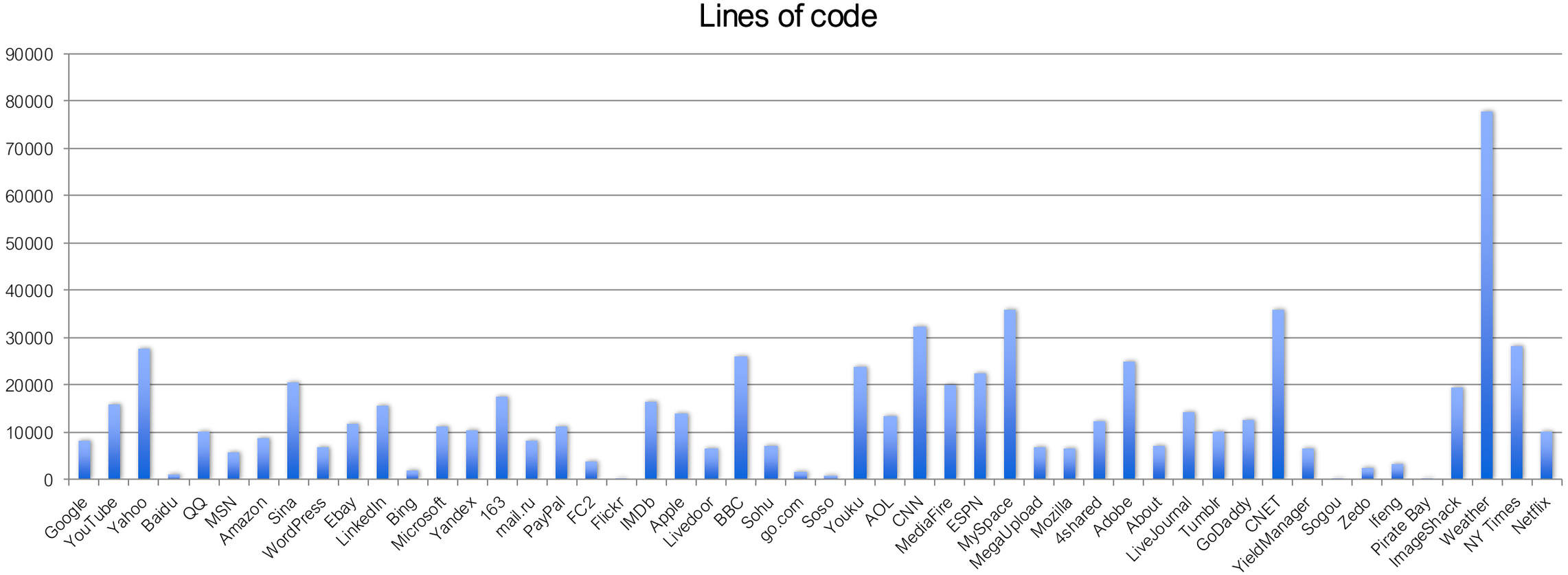


2010-2015: **JavaScript** Everywhere

- More and more webpages start heavily using JS, often with thousands of lines of minified code
- New applications (e.g., Google docs) and frameworks (e.g., JQuery)
- Academic papers written on analyzing JavaScript
- JavaScript becomes the most popular language on Github, Stackoverflow and is in the top 5 on the Tiobe Index
- JavaScript is also used to teach introductory CS by Khan Academy – often taught as the first language
- Many variants of JavaScript: TypeScript (MS), DART (Google) and Flow (Facebook). Also, HTML5
- Language for programming IoT devices (Samsung)

Prevalence of JavaScript (2012)

- 97 of Alexa top 100 websites use JavaScript
- Many of them with thousands of lines of JS code



Web Apps

Google Drive

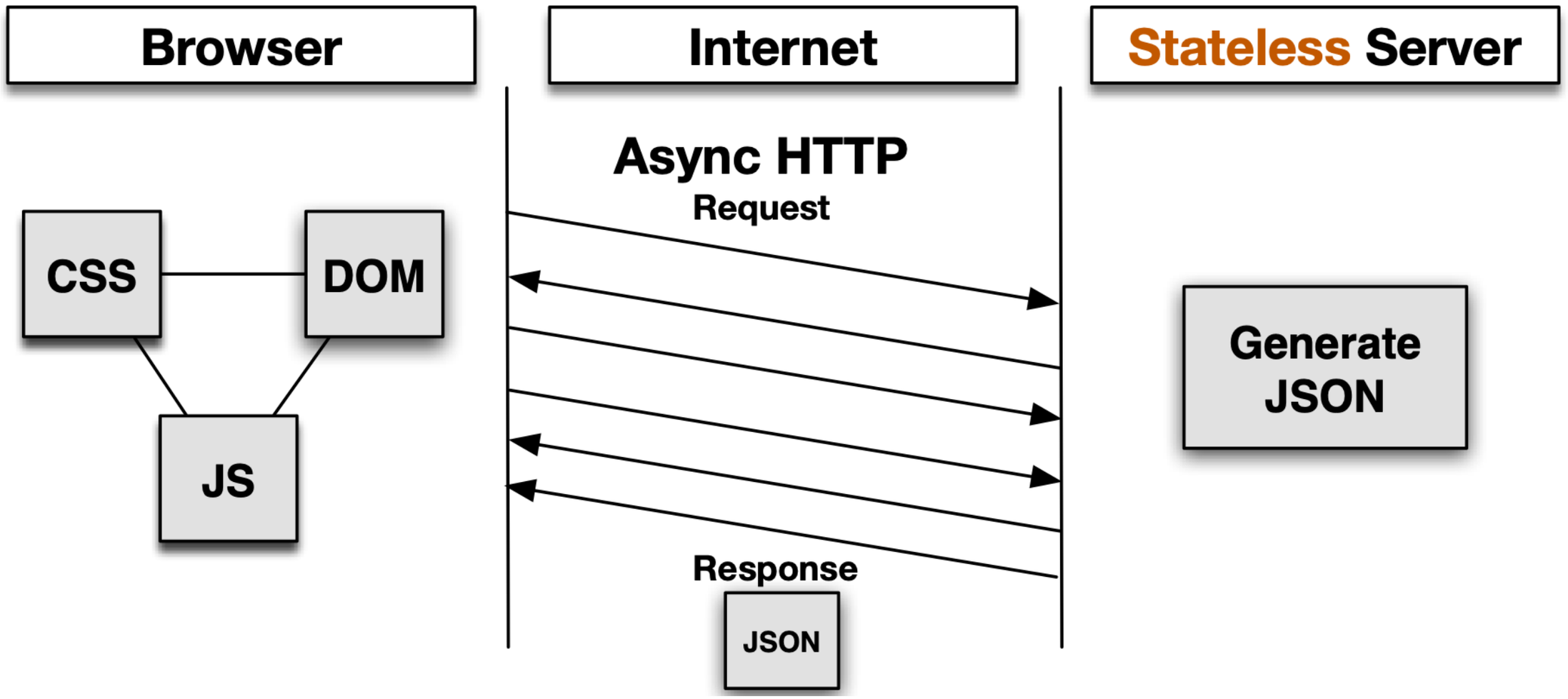
What makes JavaScript a challenging language?

JavaScript: the Difficult Parts

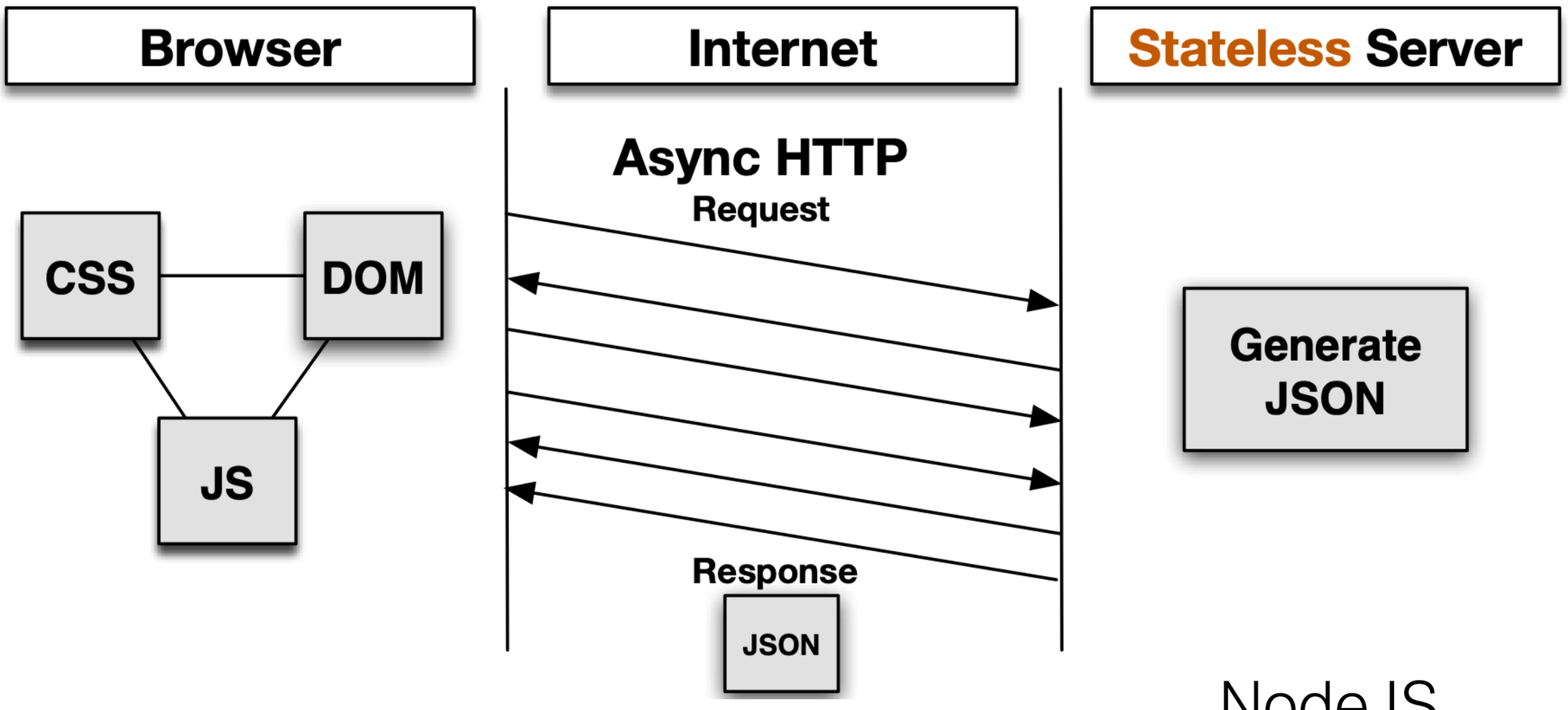
1. Dynamic and weakly-typed
   - types can change dynamically
   - objects/functions can change
2. Event-driven
3. First-class functions
4. Asynchronous callbacks
5. Prototype-based
   - objects inherit from objects
   - can be redefined at runtime
6. Constructs such as `eval`
7. Interactions with DOM
8. Interpreted (not compiled)
9. …

Make Software Analysis Challenging

JS on the client (browser)

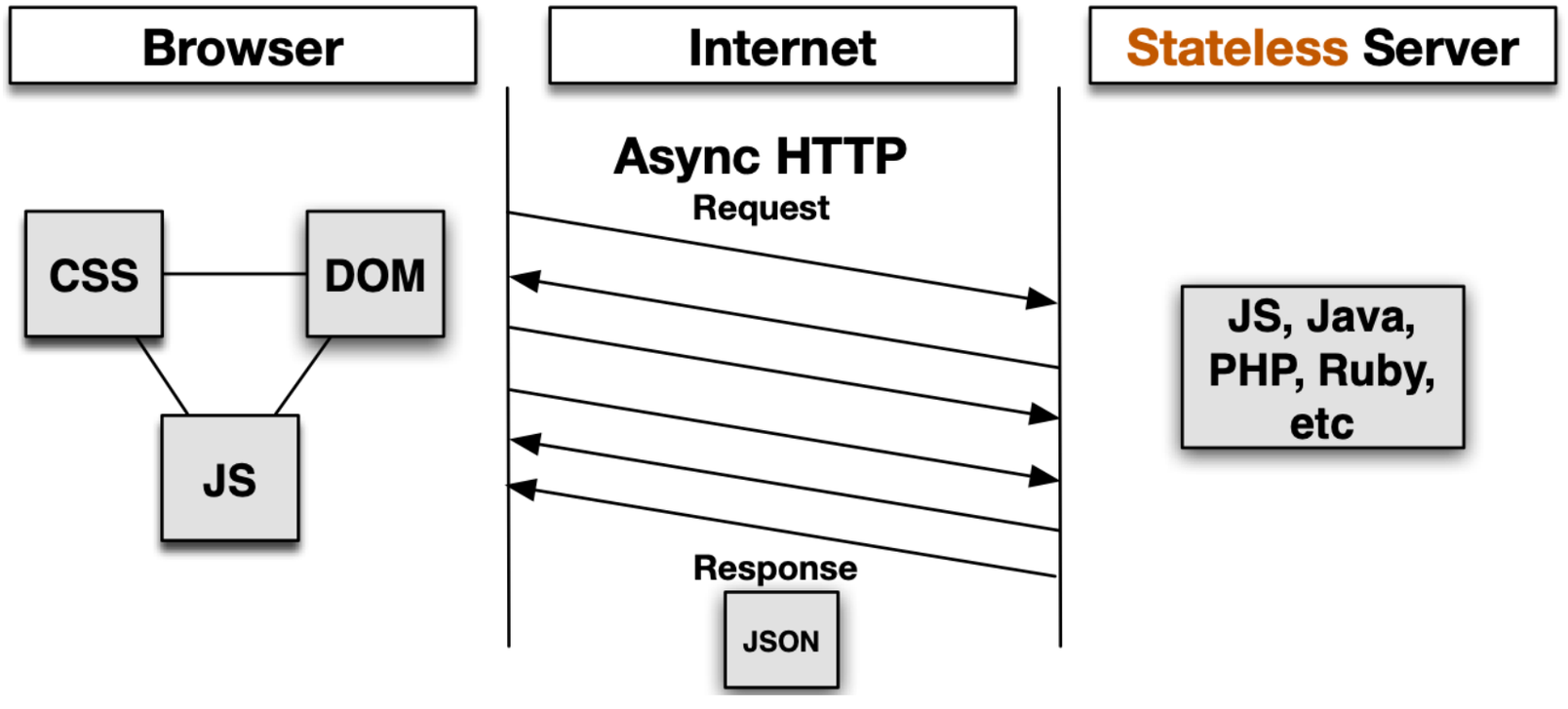


JS on the server



NodeJS

Challenges



Heterogeneous

Distributed

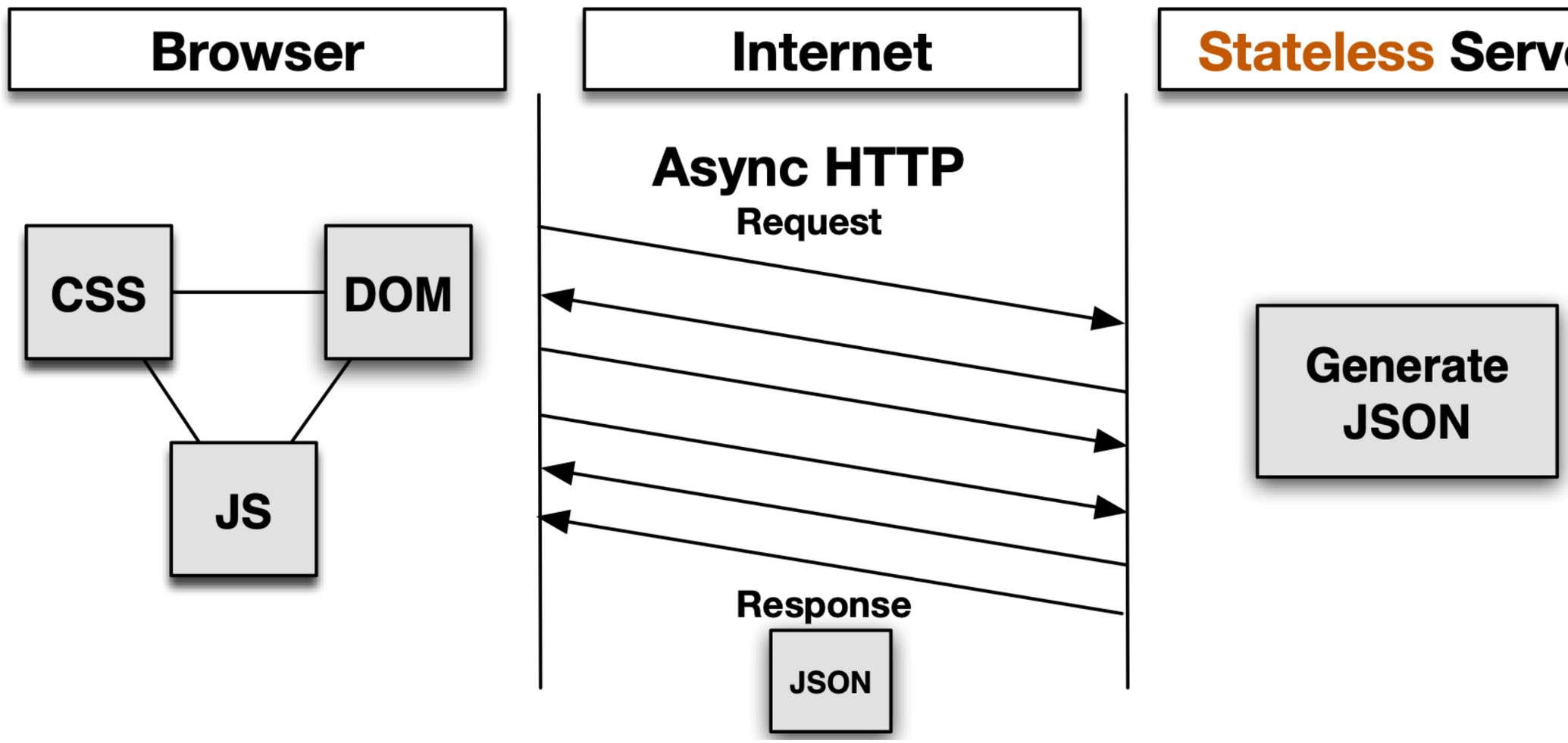
Stateless

Event-driven

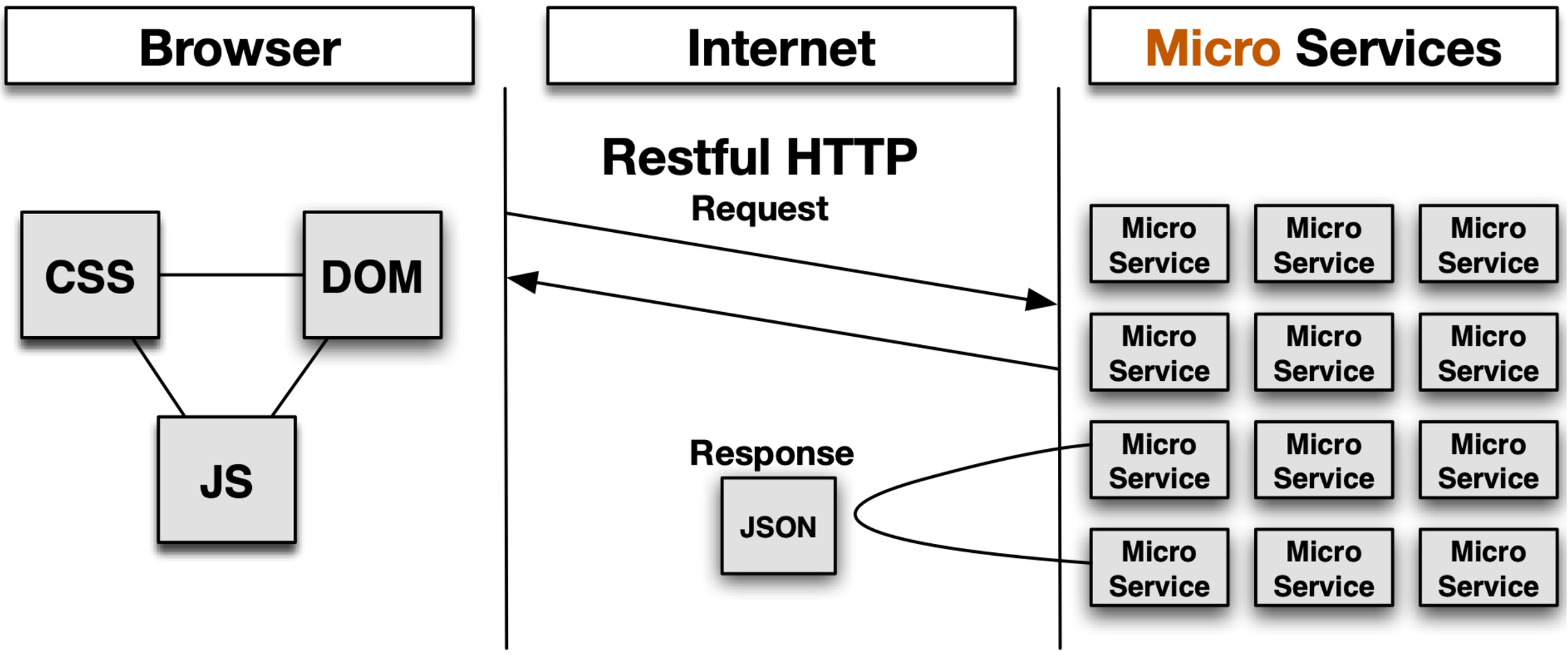
Async

Dynamic Interactions

FROM MONOLITHIC APPS



TO MICRO SERVICES



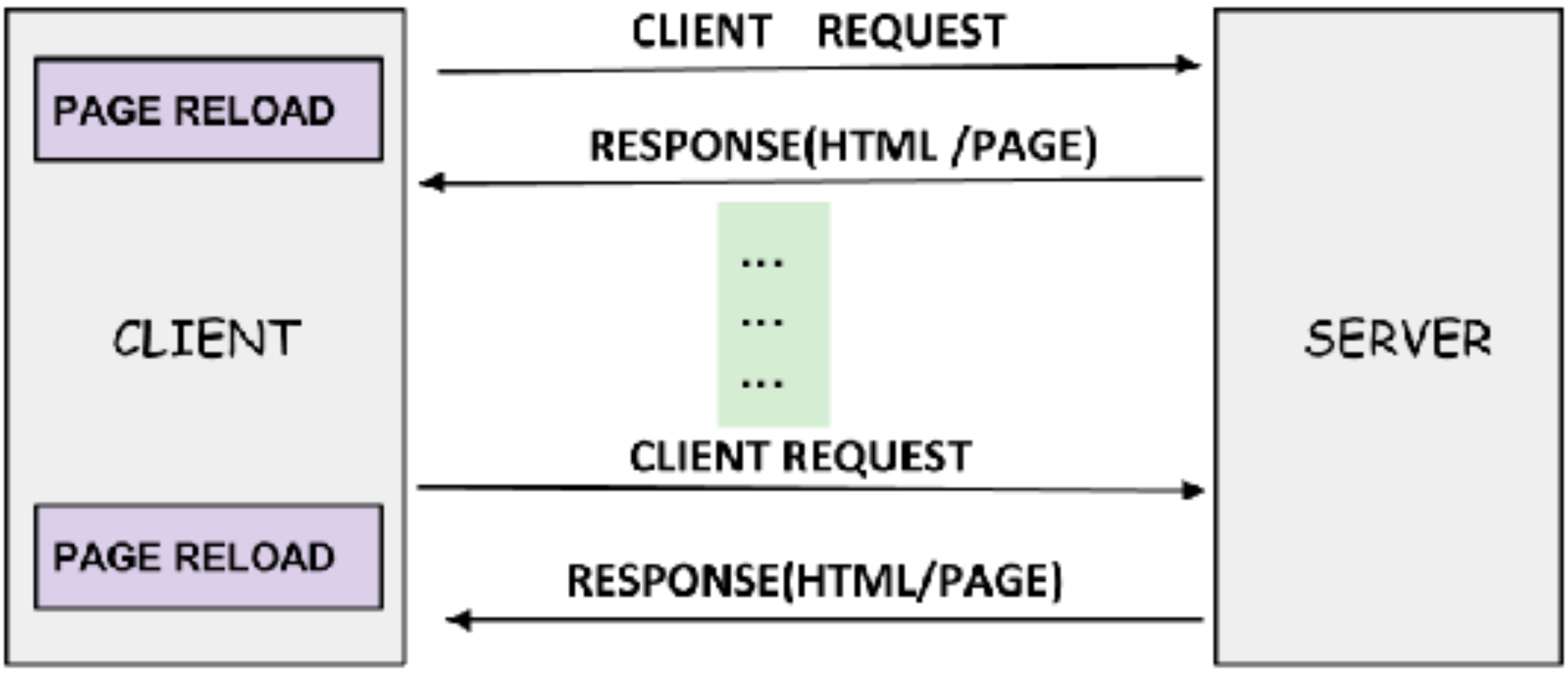
Web Application Components

- What are the three main components of a web application?

Web Application Components

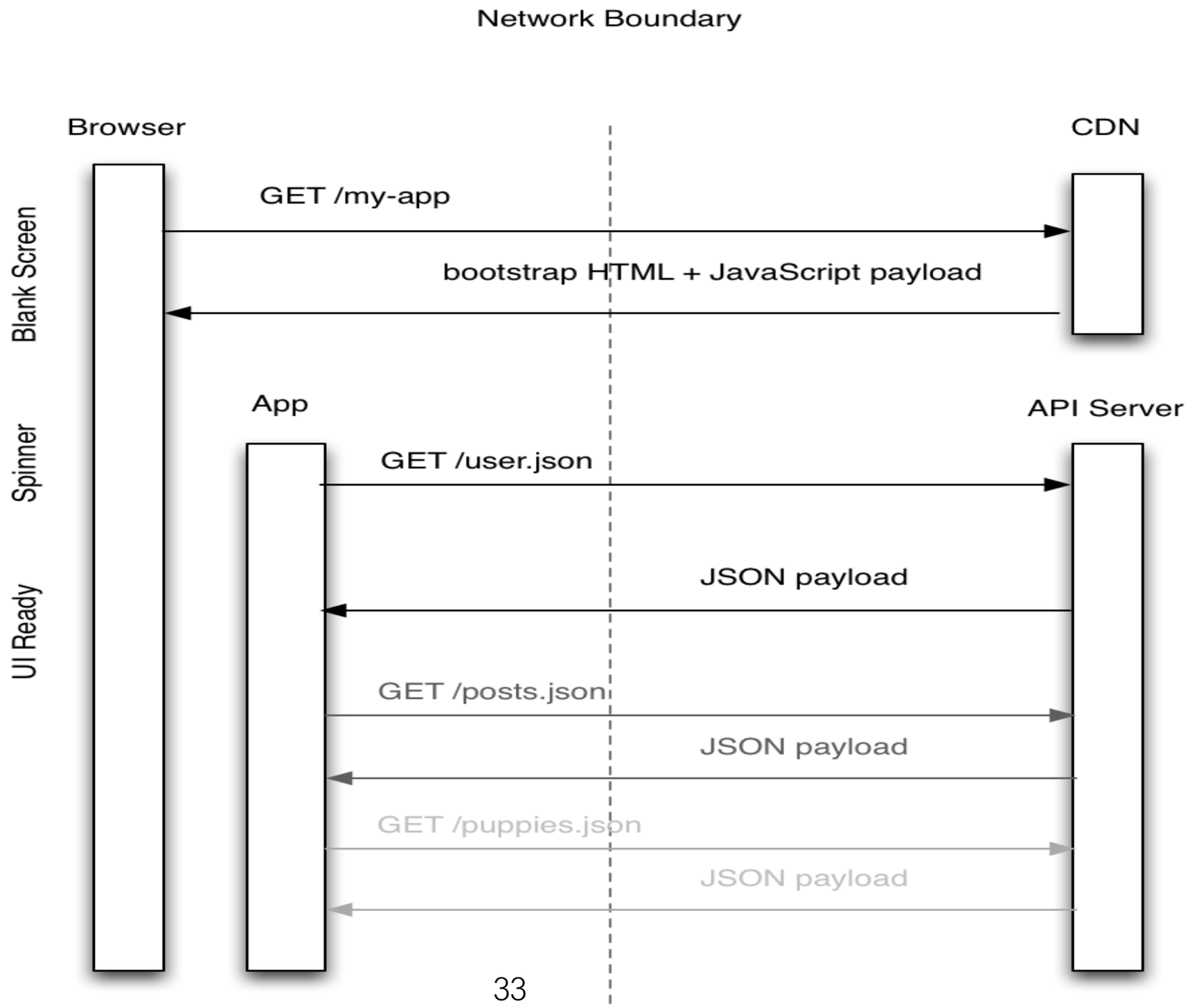
- Three pre-requisites for a web application
  - **Server**: To “serve” the web page and to send content to the client
  - **Client**: To receive content from the server and display them on the web browser window
  - **HTTP** connection for client-server interactions
- Everything else is optional.

Traditional Web Applications



Limited or no processing of content on the client side

Modern Web Applications



Components of a modern web app?

Components of a modern web app

- Client-side components
  - HTML/DOM
  - CSS
  - JavaScript
- Network:
  - REST APIs
  - Async interactions (JSON)
- Server code (Node.js or any other platform)

Client: HTML/DOM

- **Hyper-text Markup Language** (HTML) to describe the structure and contents of the initial page
  - Also has pointers to the JavaScript code (e.g., <script>)
- Is retrieved by the browser and parsed into a tree called the **Document Object Model** (DOM)
  - Common way for elements to interact with the page
  - Can be read and modified by the JavaScript code
  - Modifications to the DOM are rendered by browser

Client: CSS

- CSS (**Cascading style sheets**) separate the content of the page from its presentation
- Written in the form of declarative rules with a element on LHS and action to apply on RHS
- Ensure uniformity by applying the rule to all elements of the webpage in the DOM

Client: JavaScript

- Unique to modern web applications and provides active functionality

- Executed when <script> tag is encountered or when events are triggered on DOM elements

- Is a full-fledged programming language with many advanced features (and some bad ones)

Async Messages

- Way for client and server to interact with each other without navigating to a new webpage or blocking the page
- Initiated by the client by sending a message asynchronously (activities can happen in parallel after send on the client)
- Request is processed by server, and a response is returned. Response triggers a registered callback at the client.
- Call-back can do whatever it wants with the response, though typically it uses it to update the DOM

Server

- Server can be written in whatever language one wants, as long as it serves the content and responds to HTTP requests from the client

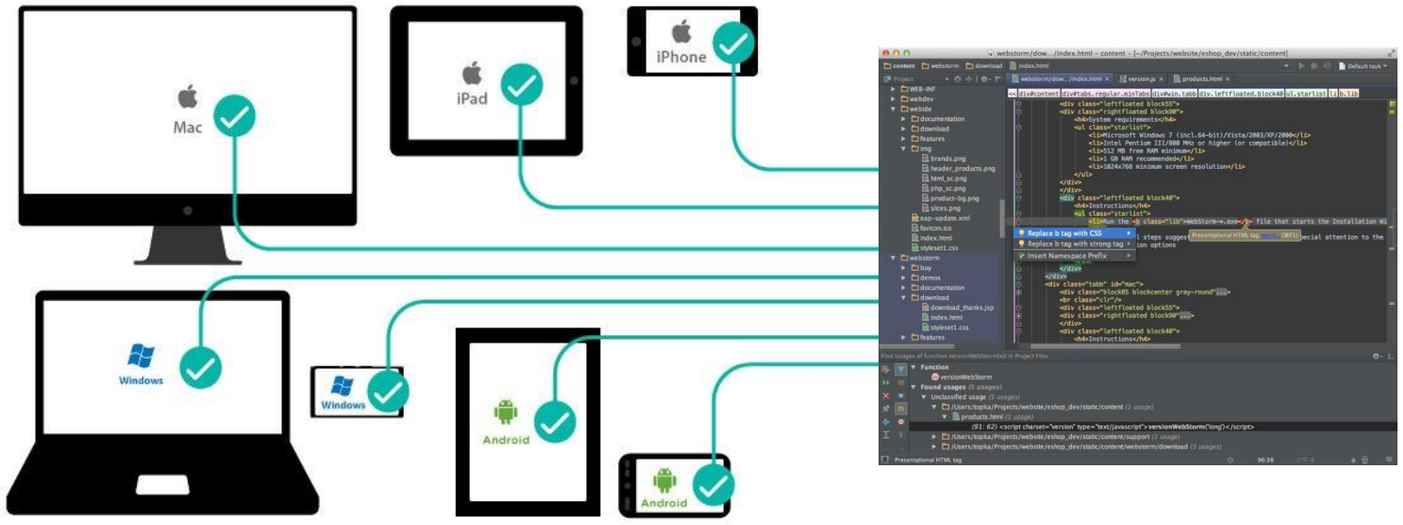
- Typical languages used: Java, RoR, PHP etc.

- Recently, JavaScript is also being used to write server-side applications for uniformity and portability. Example of this is Node.js

Other state elements

- Cookies
- Persistent local storage (HTML5 and variants)
- Server Database (e.g., MongoDB)
- Canvas (HTML5)

Why build web apps?



“write once, run anywhere”

Instantaneous software delivery

- **Ease of installation and use**
- **Standard UI and interface across platforms**
- **Service provider retains control of code distribution. No need to patch, upgrade etc.**
- **Allow compute and data-storage intensive functionality to be offloaded to the server.**

HTML: What is it?

HTML: What is it?

- HTML: HyperText Markup Language

- **Hierarchical** way to organize documents and display them (typically in a web browser)

- Combines semantics (document structure) with presentation (document layout)

- Allows tags to be interspersed with document content e.g., <head> - these are not displayed, but are directives to the layout engine

HTML Evolution

Influenced by browser implementation quirks

What to do if you see “ Some text” (missing closing)?

- Complain bitterly about malformed HTML.
- Figure out there was a missing , add it, and continue processing.

Forked into HTML and XHTML (XML-based HTML)

XHTML is more strict about adhering to proper syntax

Users came to depend on browser quirks, so browsers couldn't change. -> caused a lot of headaches for developers too

XHTML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN”
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
lang="en">
   <head>
      <title>Hello World</title>
   </head>
   <body>
      <p>Hello world!</p>
   </body>
</html>
```

HTML: Example

head

```
<!DOCTYPE html>
<html>
  <head>
    <title>Photo Gallery</title>
  </head>

  <body>
    <div class="photo">
      <h3>My first photo</h3>
      <img src="picture1.jpg"/>
    </div>
  ...
  </body>
</html>
```

body

HTML: Head

- Is typically NOT displayed by web browser
- Contains metadata to describe the page
  - Title of the webpage: <title> TITLE </title>
  - Style of the webpage: <style> style rules </style>
  - Link to CSS stylesheets: <link rel="stylesheet" type="text/css" href="">
  - For search engines: <meta name="" content="">
  - Embed JavaScript: <script > Javascript code </script> OR <script src="Javascript file"></script>

HTML: Body

- Contains the actual contents of the page with HTML tag descriptors for the structure

- Common **structural** tags used in HTML

<**div**>	group elements spanning multiple lines line break before and after
<**span**>	group elements within a single line
<**p**>	new paragraph
<**br**>	line break

More HTML Tags

<**h1**>,…,<**h6**>	headings
<**img** src=‘’>	images
<**a** href=‘’>	hyperlink
<**table**><**tr**><**td**>	tables
<**ul**><**li**>	unordered list
<**ol**><**li**>	ordered list
<**form**><**input**>	forms that take in user input

Newer HTML5 Tags

Additions tags to allow content definition

- <article>, <section>, <header>, <footer>, <summary>, <aside>, <details>, <mark>, <figcaption>, <figure>, <nav>
- Drawing
  - <svg> - Scalable Vector Graphics - Draw shapes
  - <canvas> - Draw from JavaScript - 3D with WebGL
- Timed media playback: <video> and <audio>

Example of HTML Raw content text

Introduction

There are several good reasons for taking CPEN322. You will learn a variety of interesting concepts. It may inspire you to change the way software is developed. It will give you the tools to become fabulously wealthy.

Example of HTML Annotate with tags

```
<h2>Introduction</h2>
<p>There are several good reasons for taking
<i>CPEN322</i>.
</p>
<ul>
<li>You will learn a variety of interesting concepts.</li>
<li>It may inspire you to change the way software is
developed.</li>
<li>It will give you the tools to become fabulously
wealthy.</li>
</ul>
```

Example of HTML Browser output

Introduction

There are several good reasons for taking *CPEN322*.

- You will learn a variety of interesting concepts.
- It may inspire you to change the way software is developed.
- It will give you the tools to become fabulously wealthy.