CORS DEMYSTIFIED

IT'S SIMPLER THAN YOU THINK

By Bill Parrott / @chimericdream

SPONSOR SHOUT OUT

WHO AM I?

- Software Engineer at Spring Venture Group
- Professional Web Developer for 10 years
- Web Tinkerer for 20+ years

WHO IS THIS TALK FOR?

THIS TALK IS FOR YOU

Source: Adobe Stock Photo

AND YOU

Source: Adobe Stock Photo

THIS TALK IS FOR...

- Front end developers who want to better understand what happens behind the scenes when they call `new XMLHttpRequest()` or `$.ajax(...)`
- API developers who want to better understand the ways in which CORS can be configured and the common scenarios for each
- Anyone who thinks reading documentation and the HTTP specification is "fun"

TOPICS WE'LL COVER

- What is CORS?
- A brief history of CORS
- Common use cases
- How CORS works
- CORS vs JSON-P
- When CORS isn't the answer

TOPICS WE'LL COVER

- What is CORS?
- A brief history of CORS
- Common use cases
- How CORS works
- CORS vs JSON-P
- When CORS isn't the answer

TOPICS WE'LL COVER

- What is CORS?
- A brief history of CORS
- Common use cases
- How CORS works
- CORS vs JSON-P
- When CORS isn't the answer

TOPICS WE'LL COVER

- What is CORS?
- A brief history of CORS
- Common use cases
- How CORS works
- CORS vs JSON-P
- When CORS isn't the answer

TOPICS WE'LL COVER

- What is CORS?
- A brief history of CORS
- Common use cases
- How CORS works
- CORS vs JSON-P
- When CORS isn't the answer

TOPICS WE'LL COVER

- What is CORS?
- A brief history of CORS
- Common use cases
- How CORS works
- CORS vs JSON-P
- When CORS isn't the answer

TOPICS WE'LL COVER

- What is CORS?
- A brief history of CORS
- Common use cases
- How CORS works
- CORS vs JSON-P
- When CORS isn't the answer

WHAT IS CORS?

CORS IS...

CORS IS...

DECEPTIVELY EASY TO SUMMARIZE

"Cross-origin resource sharing (CORS) is a mechanism that allows restricted resources (e.g. fonts) on a web page to be requested from another domain outside the domain from which the first resource was served."

Source: Wikipedia

CORS IS...

EASY TO MISUNDERSTAND

"Normcore truffaut waistcoat hexagon master cleanse. Slow-carb leggings whatever, flexitarian tumblr meh 8-bit unicorn franzen etsy wolf. Kombucha wayfarers dreamcatcher hella waistcoat, freegan photo booth bushwick shaman literally flannel synth."

Source: Hipster Ipsum

CORS IS...

A PAIN IN THE REAR

"CORS? Oh my god, I hate dealing with CORS issues."

Source: Every web developer ever

CORS IS...

BETTER DESCRIBED AS A BEHAVIOR THAN A TOOL OR TECHNOLOGY

Cross-origin resource sharing is what happens when JavaScript code on one website asks another site for a thing.

A BRIEF HISTORY OF CORS

YET ANOTHER ORIGIN STORY

2004	Proposed for VoiceXML browsers to allow safe cross-origin requests
2005	Separated into its own specification (NOTE) for general use beyond VoiceXML
2006	First W3C Working Draft submitted
2009	Renamed Cross-Origin Resource Sharing
2014	Accepted as a W3C recommendation

2014? SO IT'S PRETTY NEW, FOR A WEB THING.

NO, AND YES

COMMON USE CASES

(YOU'RE PROBABLY ALREADY DOING MOST OF THESE)

COMMON USE CASES

- Building/consuming APIs
- Access protected resources (i.e. endpoints/data requiring authentication)
- Serving content from a CDN
- Ads/Analytics

HOW CORS WORKS

BREAKING IT DOWN

A COUPLE IMPORTANT NOTES

Client != Browser

A COUPLE IMPORTANT NOTES

CORS is mostly passive.

It's all about configuration.

ANATOMY OF A CORS REQUEST

- Origin
- Request headers
- Response headers
- Pre-flight request
- XMLHttpRequest object

ANATOMY OF A CORS REQUEST

- Origin
- Request headers
- Response headers
- Pre-flight request
- XMLHttpRequest object

WHAT IS AN ORIGIN?

`https://www.somewebsite.com:1234/`

WHAT IS AN ORIGIN?

`https://www.somewebsite.com:1234/`

WHAT IS AN ORIGIN?

`https://www.somewebsite.com:1234/`

WHAT IS AN ORIGIN?

`https://www.somewebsite.com:1234/`

ANATOMY OF A CORS REQUEST

- Origin
- Request headers
- Response headers
- Pre-flight request
- XMLHttpRequest object

CORS REQUEST HEADERS

`ORIGIN`

The origin of the client making the request

CORS REQUEST HEADERS

`ACCESS-CONTROL-REQUEST-METHOD`

The HTTP method (e.g. GET, POST, etc) the client is trying to use

CORS REQUEST HEADERS

`ACCESS-CONTROL-REQUEST-HEADERS`

A list of the headers to be sent from the client as part of the request

ANATOMY OF A CORS REQUEST

- Origin
- Request headers
- Response headers
- Pre-flight request
- XMLHttpRequest object

CORS RESPONSE HEADERS

ACCESS-CONTROL-ALLOW-ORIGIN

Either * or a list of origins allowed to make CORS requests to the destination

CORS RESPONSE HEADERS

ACCESS-CONTROL-ALLOW-CREDENTIALS

Whether or not the browser may send credentials (Auth header, cookies, etc) with the request

CORS RESPONSE HEADERS

`ACCESS-CONTROL-EXPOSE-HEADERS`

A list of headers which may be sent to the client with the response to the CORS request

CORS RESPONSE HEADERS

ACCESS-CONTROL-MAX-AGE

The length of time the browser may cache the response to the pre-flight request

CORS RESPONSE HEADERS

ACCESS-CONTROL-ALLOW-METHODS

A list of HTTP methods (e.g. GET, POST, etc) which may be used for cross-origin requests

CORS RESPONSE HEADERS

ACCESS-CONTROL-ALLOW-HEADERS

A list of headers which may be sent by the client with the request

ANATOMY OF A CORS REQUEST

- Origin
- Request headers
- Response headers
- Pre-flight request
- XMLHttpRequest object

THE PRE-FLIGHT (`OPTIONS`) REQUEST

- Sent by the browser
- Includes `Origin`, `Access-Control-Request-Method`, and `Access-Control-Request-Headers` headers
- Transparent to the client
- Server's response determines whether the browser rejects or sends the client's original request

ANATOMY OF A CORS REQUEST

- Origin
- Request headers
- Response headers
- Pre-flight request
- XMLHttpRequest object

THE XMLHTTPREQUEST (XHR) OBJECT

- JavaScript API for fetching resources by URL
- Asynchronous by default, even before promises were cool

THE XHR OBJECT

NATIVE JAVASCRIPT (ES6)

```
const request = new XMLHttpRequest();
request.onreadystatechange = () => {
    if (request.readyState === XMLHttpRequest.DONE) {
        if (request.status === 200) {
            alert(request.responseText);
        } else {
            alert('Ruh roh!');
        }
    }
};

request.open('POST', 'http://somedomain.com/api/thing/123');
request.send();
```

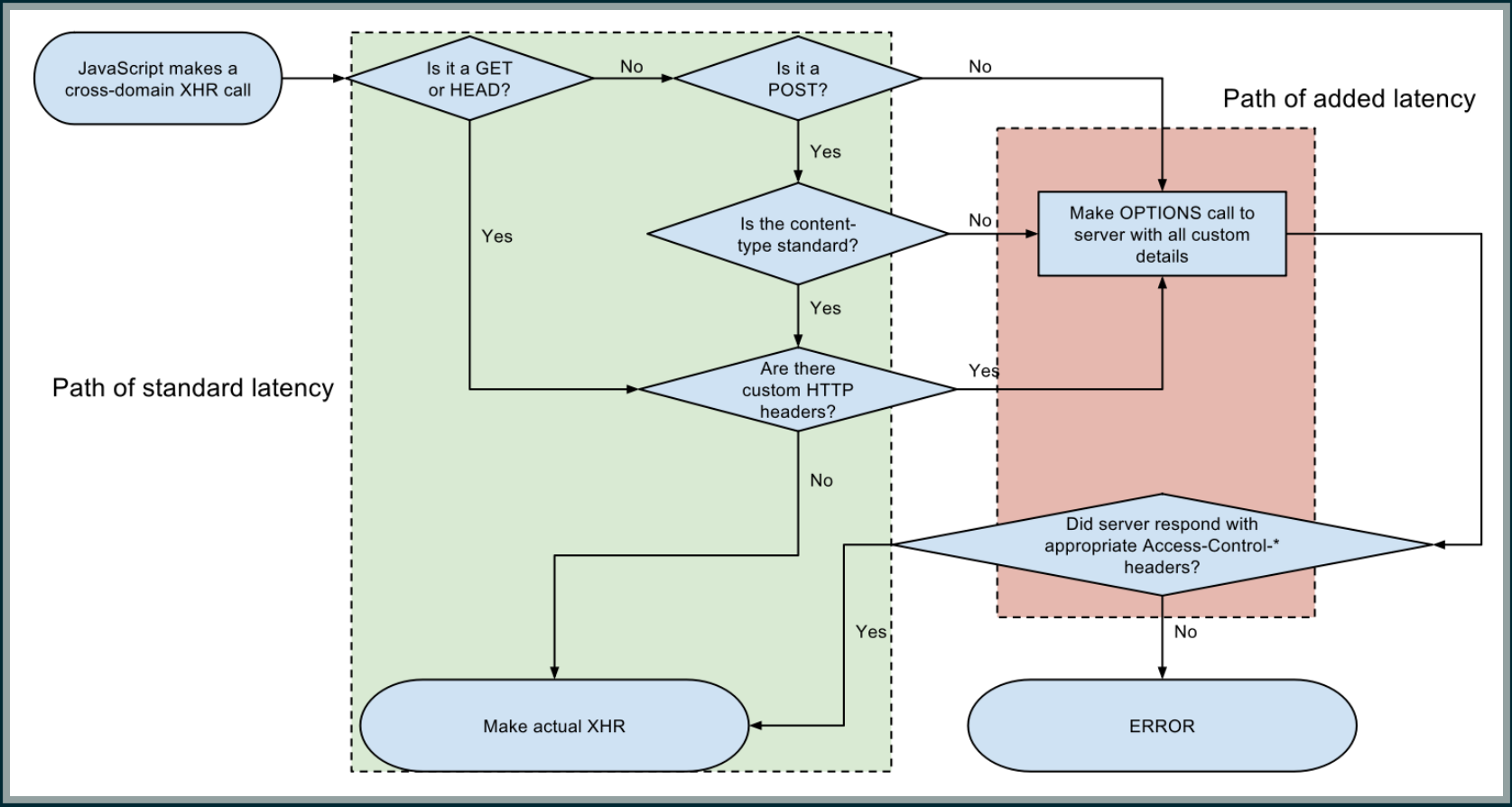
THE XHR OBJECT

JQUERY EQUIVALENT

```
$.ajax({
    'method': 'POST',
    'url': 'http://somedomain.com/api/thing/123'
}).done((data, status, xhr) => {
    alert(xhr.responseText);
}).fail((xhr, status, error) => {
    alert('Ruh roh!');
});
```

MAKING A CORS REQUEST

A "SIMPLE" CORS FLOWCHART



Source: Wikipedia

STEP BY STEP

1. Client makes XHR
2. Browser sends pre-flight request
3. Server responds with `Access-Control-*` headers
4. Browser sends (or blocks) XHR request
5. Server responds to request
6. Client parses response normally

STEP BY STEP

1. Client makes XHR
2. Browser sends pre-flight request
3. Server responds with `Access-Control-*` headers
4. Browser sends (or blocks) XHR request
5. Server responds to request
6. Client parses response normally

STEP BY STEP

1. Client makes XHR
2. Browser sends pre-flight request
3. Server responds with `Access-Control-*` headers
4. Browser sends (or blocks) XHR request
5. Server responds to request
6. Client parses response normally

STEP BY STEP

1. Client makes XHR
2. Browser sends pre-flight request
3. Server responds with `Access-Control-*` headers
4. Browser sends (or blocks) XHR request
5. Server responds to request
6. Client parses response normally

STEP BY STEP

1. Client makes XHR
2. Browser sends pre-flight request
3. Server responds with `Access-Control-*` headers
4. Browser sends (or blocks) XHR request
5. Server responds to request
6. Client parses response normally

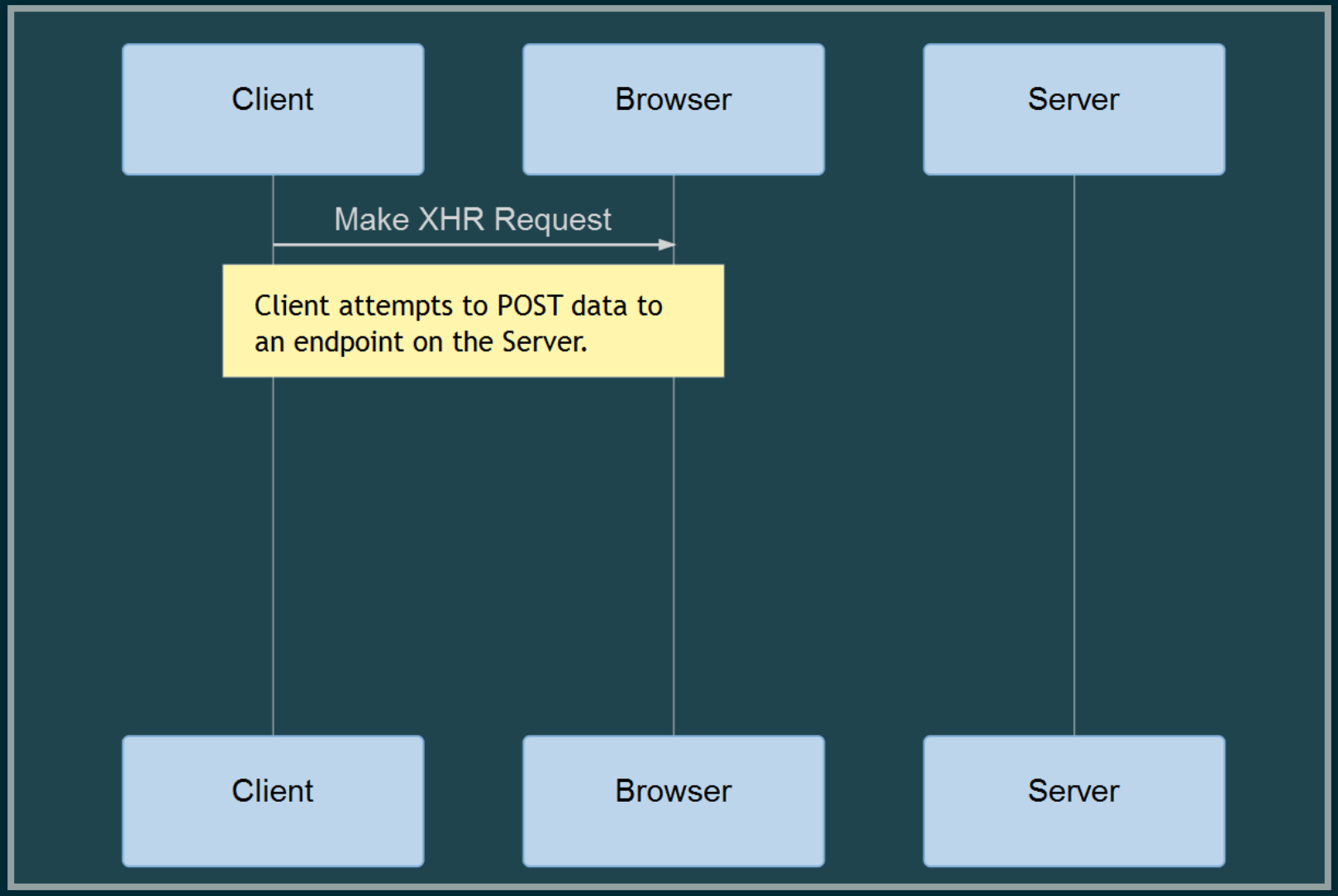
STEP BY STEP

1. Client makes XHR
2. Browser sends pre-flight request
3. Server responds with `Access-Control-*` headers
4. Browser sends (or blocks) XHR request
5. Server responds to request
6. Client parses response normally

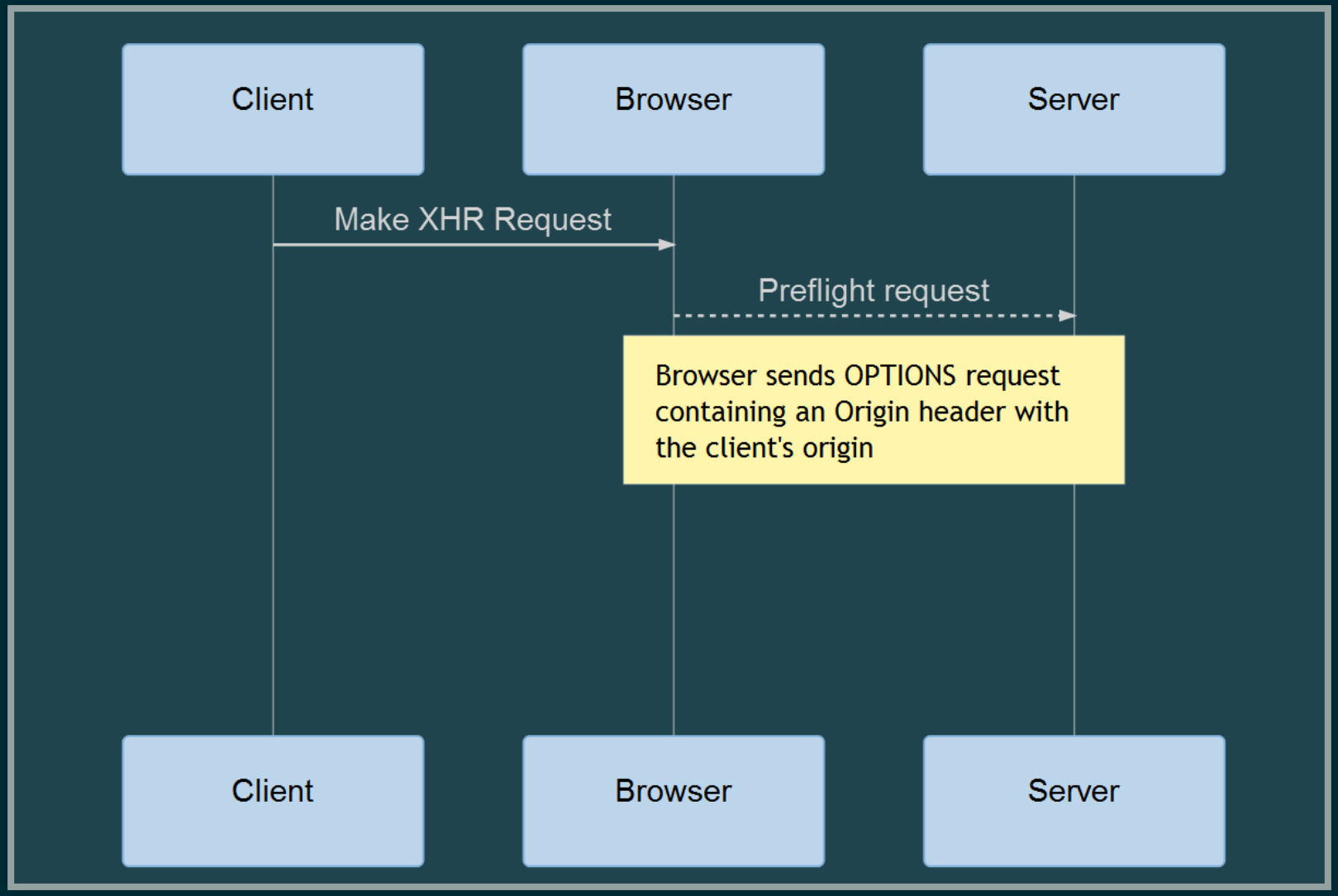
STEP BY STEP

1. Client makes XHR
2. Browser sends pre-flight request
3. Server responds with `Access-Control-*` headers
4. Browser sends (or blocks) XHR request
5. Server responds to request
6. Client parses response normally

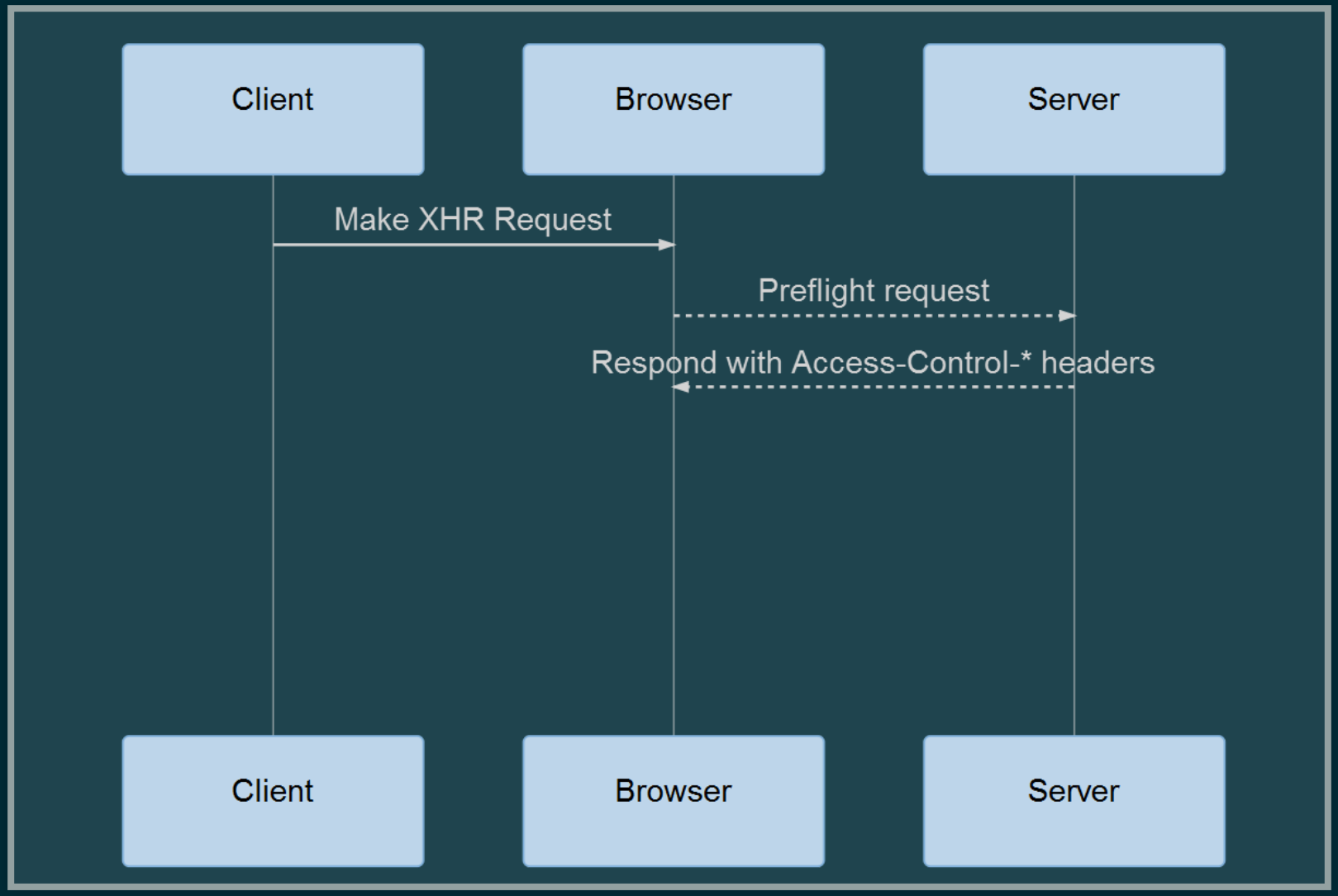
STEP BY STEP


Client
Browser
Server
Make XHR Request
Client attempts to POST data to an endpoint on the Server.
Client
Browser
Server

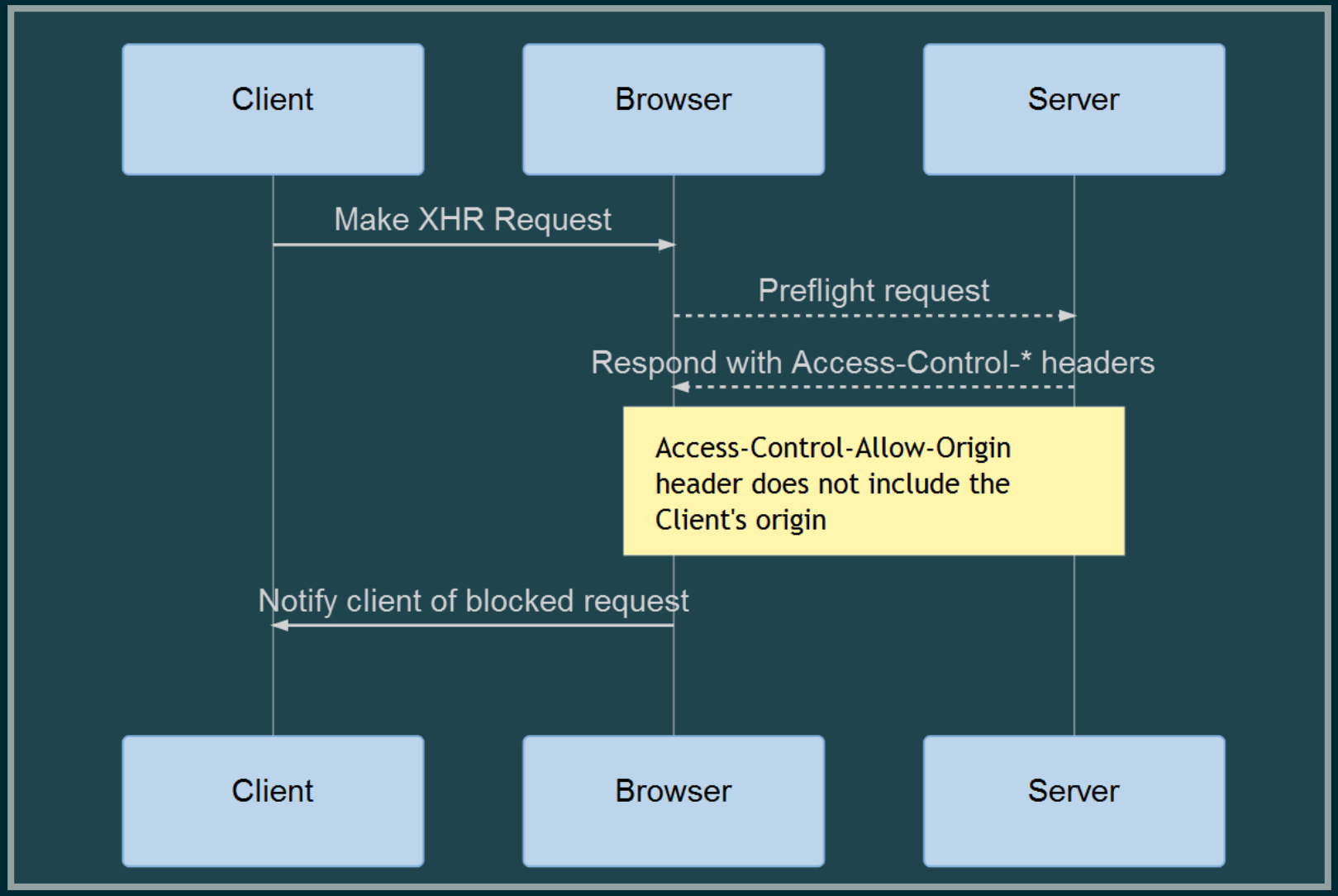
STEP BY STEP


Client
Browser
Server
Make XHR Request
Preflight request
Browser sends OPTIONS request containing an Origin header with the client's origin
Client
Browser
Server

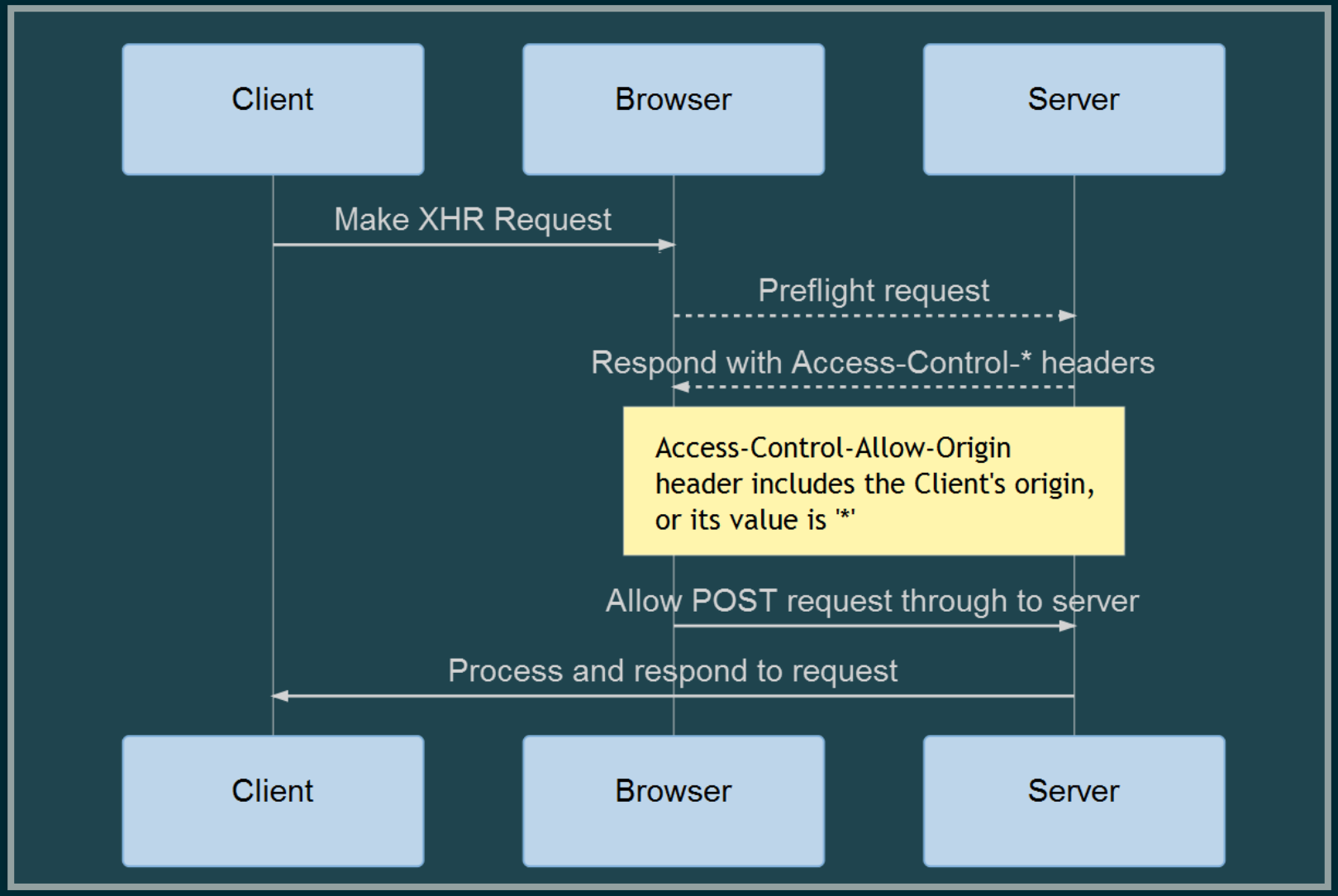
STEP BY STEP


Client
Browser
Server
Make XHR Request
Preflight request
Respond with Access-Control-* headers
Client
Browser
Server

STEP BY STEP


Client
Browser
Server
Make XHR Request
Preflight request
Respond with Access-Control-* headers
Access-Control-Allow-Origin header does not include the Client's origin
Notify client of blocked request
Client
Browser
Server

STEP BY STEP


Client
Browser
Server
Make XHR Request
Preflight request
Respond with Access-Control-* headers
Access-Control-Allow-Origin header includes the Client's origin, or its value is '*'
Allow POST request through to server
Process and respond to request
Client
Browser
Server

DEMO TIME!

CORS VS JSON-P

JSON-P ≈ "CORS LITE"

JSON-P IN A NUTSHELL

JSON-P IN A NUTSHELL

- `<script>` tag on client page
- Server responds (in theory) with JSON object wrapped in function call
- Browser evaluates and executes response as plain JavaScript

A JSON-P EXAMPLE

CLIENT

```
<script
    src="http://somedomain.com/api/get/1234?callback=evalMe">
</script>
```

A JSON-P EXAMPLE

SERVER RESPONSE

```
evalMe({"key1": "val1", "key2": "val2"})
```

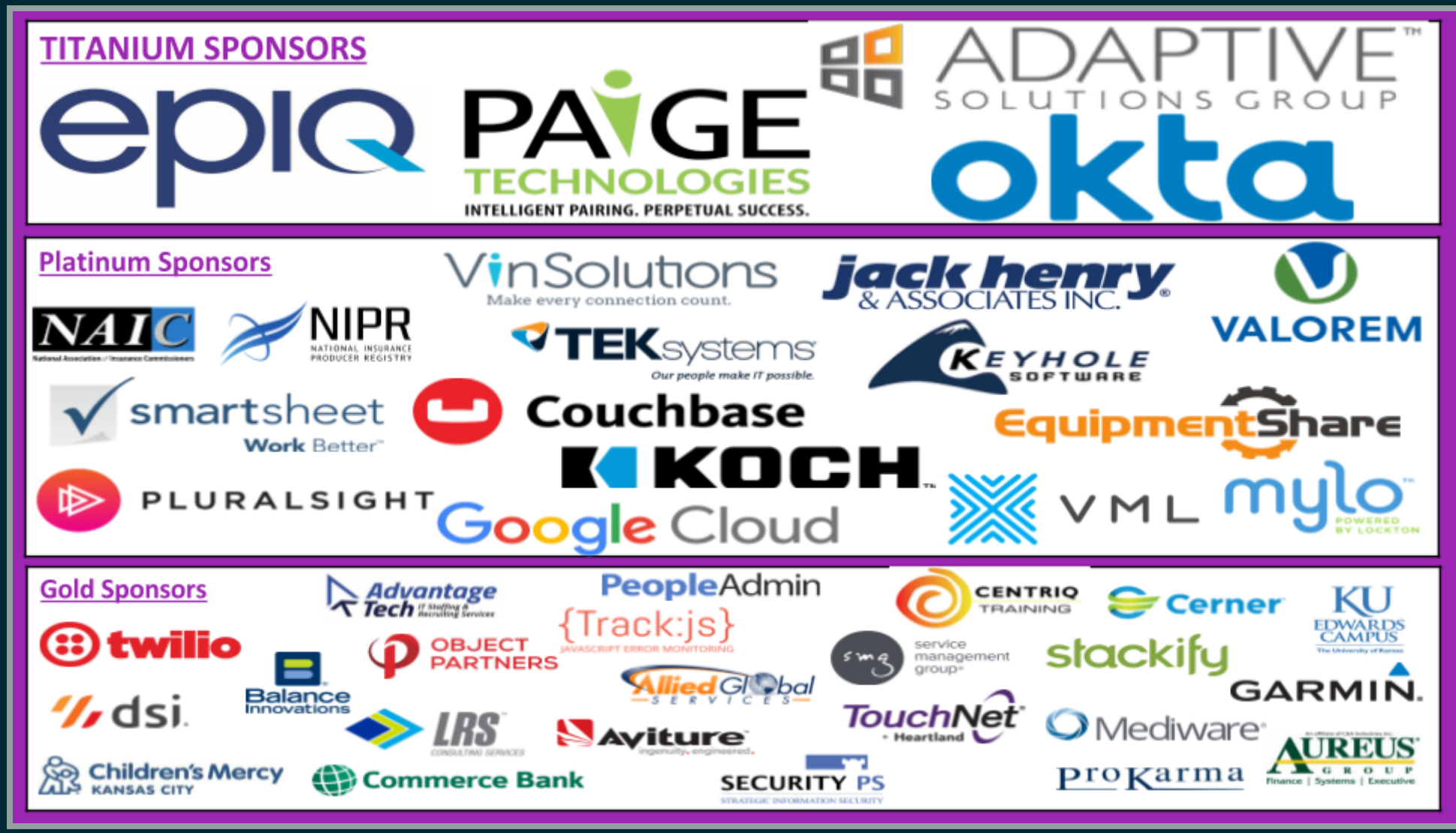
CORS VS JSON-P

CORS	JSON-P
All HTTP methods	GET only
All current browsers (Chrome, Firefox, IE 10+, etc)	Current & legacy browsers
Resistant to XSS	Vulnerable to XSS

BUT WHAT ABOUT?

WHEN CORS MAY NOT BE THE ANSWER

- Your website needs to work offline
- Your website exists in a controlled environment with a single origin
- Your website exists in a controlled environment and only makes GET requests
- You're stuck in 2006 and need to support browsers like IE 7


TITANIUM SPONSORS
epiq
PAIGE
TECHNOLOGIES
INTELLIGENT PAIRING. PERPETUAL SUCCESS.
ADAPTIVE
SOLUTIONS GROUP
okta
Platinum Sponsors
VinSolutions
Make every connection count.
jack henry
& ASSOCIATES INC.
VALOREM
NAIC
NIPR
NATIONAL INSURANCE PRODUCER REGISTRY
TEKsystems
Our people make IT possible.
KEYHOLE SOFTWARE
smartsheet
Work Better
Couchbase
EquipmentShare
KOCH
PLURALSIGHT
Google Cloud
VML
mylo
POWERED BY LOCKTON
Gold Sponsors
Advantage Tech
PeopleAdmin
CENTRIQ TRAINING
Cerner
KU EDWARDS CAMPUS
twilio
{Track:js}
JAVASCRIPT ERROR MONITORING
OBJECT PARTNERS
smg
service management group
stackify
GARMIN
dsi
Balance Innovations
Allied Global SERVICES
TouchNet
Heartland
Mediware
LRS
Aviture
AUREUS GROUP
Finance | Systems | Executive
Children's Mercy KANSAS CITY
Commerce Bank
SECURITY PS
STRATEGIC INFORMATION SECURITY
ProKarma

QUESTIONS?

THANK YOU!

Slides and demos available at:

https://github.com/chimericdream/cors-demystified