The Truth About CORS

Maarten Mortier
Showpad
11/03/2016

Cross
Origin
Resource
Sharing

What is CORS

- Allows sharing resources from an origin with other origins, **exempting Same Origin Policy**

- CORS is there to protect **YOU** and/or a **SERVICE**
  from **XSS** attacks (typically code injections)

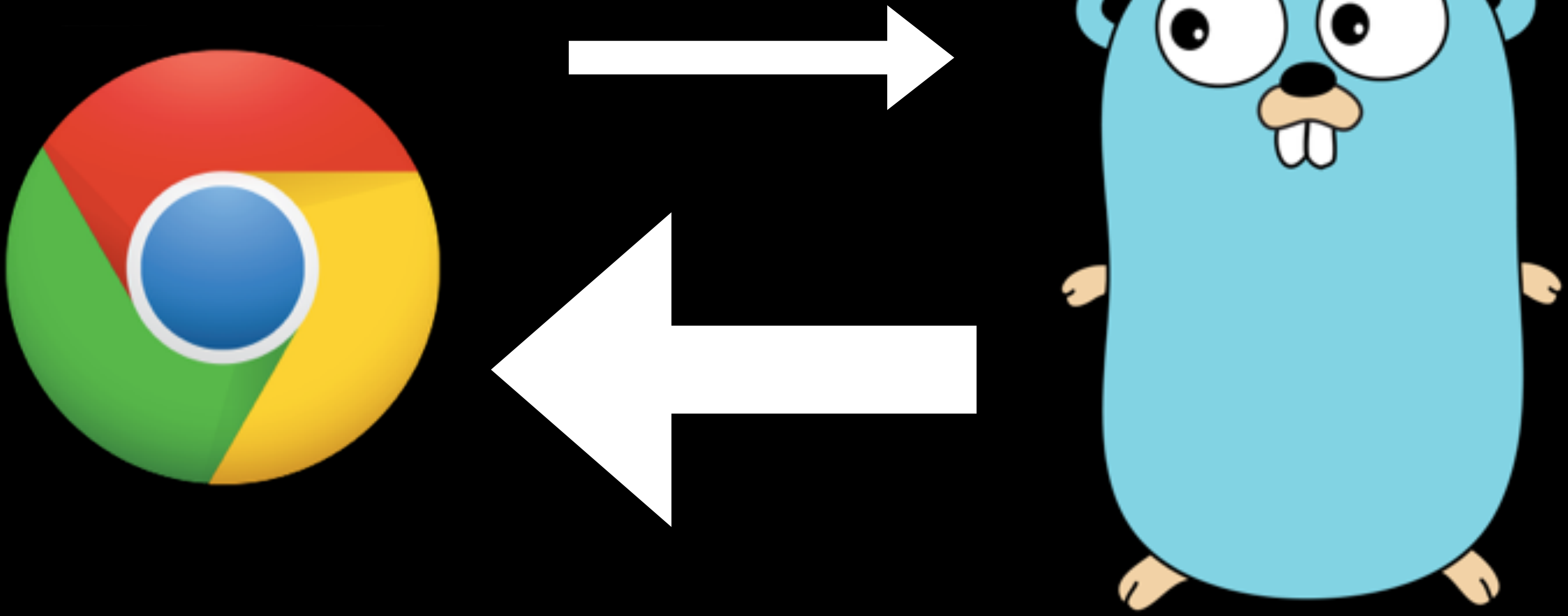
- Its restrictions are completely **CLIENT-SIDE**!

CORS

- Applies to JavaScript constructs that can fetch info from other origins
- XMLHttpRequests (AJAX)
- And to getImageData/toDataURL on Images from a different Origin

CORS

- If you think it applies to IFRAME or IMG you are confused
- JSONP is a different topic
- Want to show off? Also applies to Web Fonts and WebGL Textures



Setup

Origin

- Origins are understood and determined by Browsers
- Cannot be changed by JavaScript
- Usually relates to the address bar of the frame

Origin Equality

- start.showpad.dev != http://start.showpad.dev
- http://start.showpad.dev != https://start.showpad.dev
- https://start.showpad.dev != https://start.showpad.dev:8080
- file:// != http://start.showpad.dev
- Does not apply to data://

Preflight

- On **SOME** AJAX calls, MOST browsers will send PREFLIGHT requests

- They use the HTTP Verb “OPTIONS”

- They expect a 200 OK to succeed, and are used to consult SOME headers

Preflight

- When successful, the ‘real request’ is sent
- Preflight requests do not respect typical Cache-Control
- No way to programmatically alter/intercept the Preflights

When is there a Preflight?

- Non-traditional HTTP methods always require a Preflight (like HONK)
- Whenever you send some kind of custom header, Browsers will send a Preflight
- Whenever you send a cookie
- Whenever you POST data that is not `application/x-www-form-urlencoded`, `multipart/form-data`, or `text/plain`
- *Slight differences across browsers (Firefox sends more Preflights)

Cookies

- By default, AJAX requests do NOT send COOKIES to a different Domain

- XmlHTTPRequests that have “.withCredentials = true” will send Cookie data

- This only happens *if a certain Preflight check succeeded (see later)*

The CORS family

Preflight **responses**
Cross Origin **responses**

ACAO

- `Access-Control-Allow-Origin: <origin> | *`

  Allows cross origin requests from <origin> or all origins.
  No multiple origins :(

- If using specific origin, use:
  - `Vary: Origin`

ACEH

- `Access-Control-Expose-Headers: X-My-Custom-Header, X-Another-Custom-Header`

  Filters headers coming back from other origins

ACMA

- `Access-Control-Max-Age: <delta-seconds>`

  Allows Browsers to CACHE preflight requests

ACAC

- `Access-Control-Allow-Credentials: true | false`

  This needs to be set to TRUE to allow cookies to be sent
  Browsers do not accept this to be set TRUE when a wildcard origin is used

ACAM

- `Access-Control-Allow-Methods: <method>[, <method>]`

  Allows only certain HTTP Methods to access cross origin resources

ACAH

- `Access-Control-Allow-Headers: <field-name>[, <field-name>]*`

  Allows only certain HTTP Headers to be sent on cross origin requests

The CORS family

Preflight/request parameters

Origin

- `Origin: <origin> | null | “”`

  Lets the server know what Origin is used

ACRM

- `Access-Control-Request-Method: <method>`

  Lets the server know what method will be used, during preflight

ACAH

- `Access-Control-Request-Headers`

  Lets the server know what headers will be sent, during preflight

Redirects

- Clears “Origin” on most browsers and sets it to “null”, take this into account!
- When your request is ‘preflighted’, no redirects are allowed!

Safari vs Chrome

- file:// locations

How to turn off CORS

- In Develop menu in Safari
- Startup parameter for Chrome

AMA