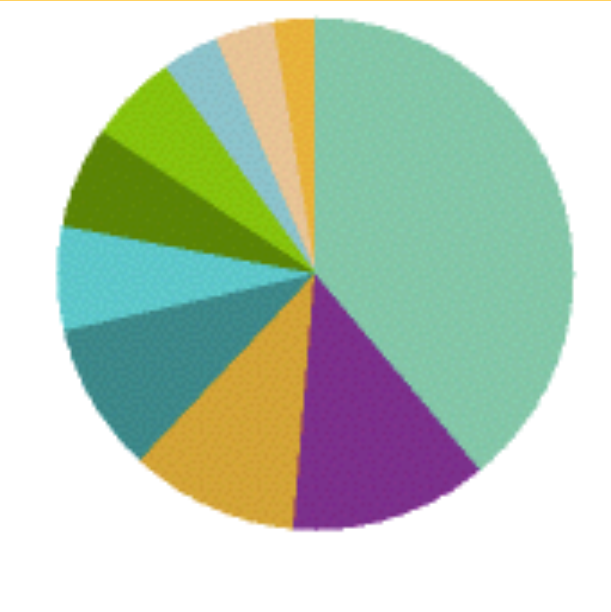


Javascript, parte VII

Acceso a APIs externos

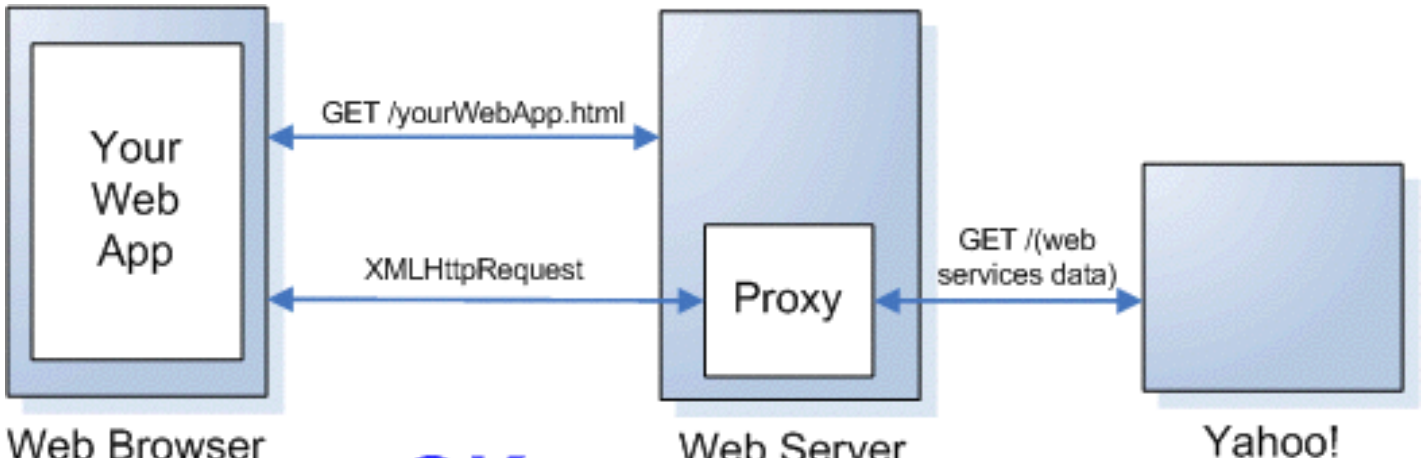
La web como plataforma

APIs accesibles desde el cliente

- El problema principal para poder usar un API web desde Javascript no es el lenguaje en sí, sino la **política de seguridad del "mismo origen"**
- Esta política implica que en general desde una página con cierto origen (URL) no se puede acceder a ninguna información de otra página de un origen distinto
  - En general esto es OK, a nadie le gustaría que código malicioso de una URL estuviera accediendo a ningún dato de la solapa del navegador que está abierta con nuestra cuenta bancaria
  - Pero también es un problema, por ejemplo con AJAX: una página de http://localhost en principio no puede hacer una petición AJAX a Google
  - Como consecuencia, se han tenido que idear diversos "trucos"/técnicas para intentar sobrepasar este límite

Opción 1: Usar un proxy en el servidor

- Podemos colocar en nuestro servidor un programa que simplemente "retransmita" la petición al *host* al que queremos llegar
  - Como las peticiones desde el lado del servidor no están restringidas, no habrá problema
  - Ejemplo: http://developer.yahoo.com/javascript/howto-proxy.html



Opción 2: Usar el tag <script>

- Las restricciones de seguridad no se aplican a la etiqueta <script>. Con ella podemos cargar (y ejecutar!!) código Javascript de cualquier origen
- Podríamos cargar en un punto del documento la respuesta del servidor en formato JSON, por ejemplo:

```
<script
   src="http://api.flickr.com/services/rest?
format=json&method=flickr.photos.search&api_key=<TU_API_KEY>&
tags=gatitos">
</script>
```

  - Pero esto no sirve de mucho. Tendríamos el JSON, pero no se haría "nada útil" con esa información

JSONP

- Si consiguiéramos ejecutar una función nuestra que recibiera como parámetro el JSON que envía el servidor todo estaría resuelto
- En los servicios que admiten **JSONP**, debemos pasar un parámetro (normalmente se llama "callback" o algo similar) con el nombre de la función a llamar.

```
http://api.flickr.com/services/rest?
format=json&method=flickr.photos.search&api_key=<TU_AP
I_KEY>&tags=gatitos&jsoncallback=miFuncion
```

- El servidor nos devolverá un Javascript del estilo:

```
miFuncion(JSON_RESULTADO_DE_LA_PETICION)
```

- Es decir, se ejecutará la función "miFuncion" recibiendo como argumento el JSON. En "miFuncion" procesaríamos los datos (los filtraríamos y mostraríamos en la página, por ejemplo)

Ejecutando JSONP “a petición”

- Según los ejemplos anteriores, parece que la llamada al servicio se tenga que hacer cuando se carga la página, pero también se puede hacer en respuesta a un evento
- Una etiqueta <script> creada dinámicamente se ejecuta en el momento en que se inserta en el documento

```
<body>
<script>
  function llamarServicio() {
      var s = document.createElement("script")
      s.src = “http://api.flickr.com/services/rest?
format=json&method=flickr.photos.search&api_key=<TU_API_KEY>&jsoncal
lback=miFuncion&tags=gatitos"
      document.body.appendChild(s);
  }
  function miFuncion(json) { alert(JSON.stringify(json))}
</script>
<input type="button" onclick="llamarServicio()" value="JSONP">
</body>
```

Opción 3: CORS

- *Cross-Origin Request Sharing*: No es un "truco", es el mecanismo estándar para hacer llamadas AJAX "cross-domain"
  - O sea, es el reconocimiento "oficial" de que en ciertos casos el navegador debería permitirlas
- Si el **servidor** al que le haces la petición "dice" que OK a la petición "cross-domain" (**enviando la cabecera Access-Control-Allow-Origin**), el navegador también dejará que continúe

```
HTTP/1.1 200 OK
Server: Apache/2.0.61
Access-Control-Allow-Origin: *
```

- Por desgracia no demasiados APIs lo ofrecen en la actualidad, pero van aumentando en número
  - Lista de APIs en http://enable-cors.org/resources.html

OAuth

- Para que una *app* pueda acceder a servicios de terceros sin que el usuario tenga que darle a la app sus credenciales del servicio
  - Ejemplo: una app que permite publicar en tu muro de FB, pero en la que no confías lo suficiente como para meter tu login y password de FB
- Es el estándar en APIs REST abiertos a terceros
- Se basa en el uso de un *token* de sesión

Flujos en OAuth

- OAuth especifica varios **flujos de autenticación**, que indican la secuencia de pasos que debe seguir el **usuario**, la **aplicación** y el **servicio** para acceder a un recurso protegido

- Ejemplo de flujo de autenticación con Facebook: https://developers.facebook.com/docs/facebook-login/manually-build-a-login-flow/ (Implicit flow, según OAuth 2.0)
  - El navegador debe ir a una URL de autenticación, en FB, pasándole como parámetro HTTP el **id de la aplicación** y algunos datos más
    - La aplicación debe estar registrada en FB
  - Desde esa URL (de FB) se hace el login
  - Desde FB se le redirige a una URL propia de la aplicación, pasándole un "**access token**", que se debe usar en todas las operaciones restringidas