Servicios Web y APIs

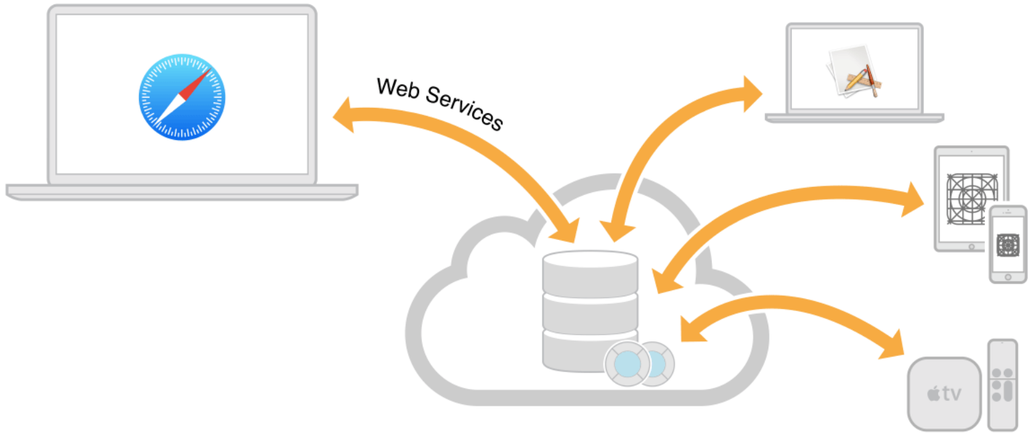
Servicio web

- Un **servicio web** (web service) es una tecnología que utiliza un conjunto de protocolos y estándares web que sirven para intercambiar datos entre aplicaciones

- Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes como Internet

- Existen dos formatos habituales para el intercambio de datos:

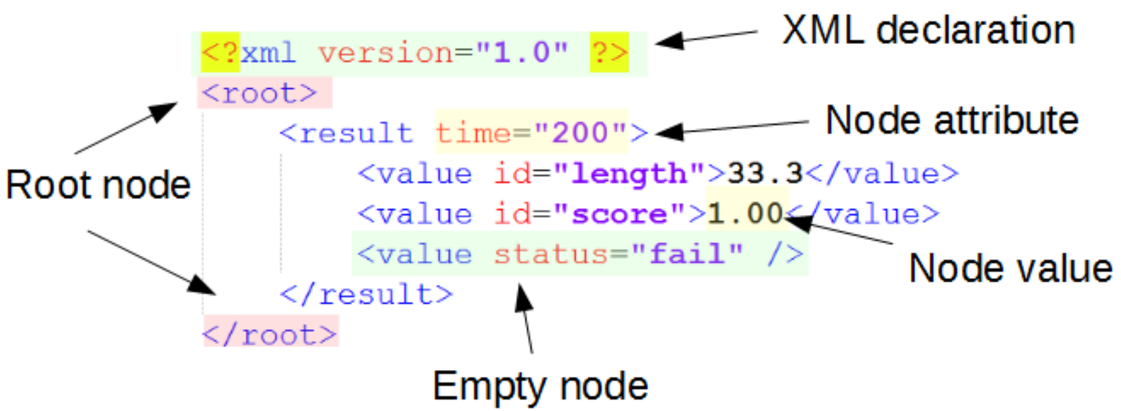
  - XML
  - JSON



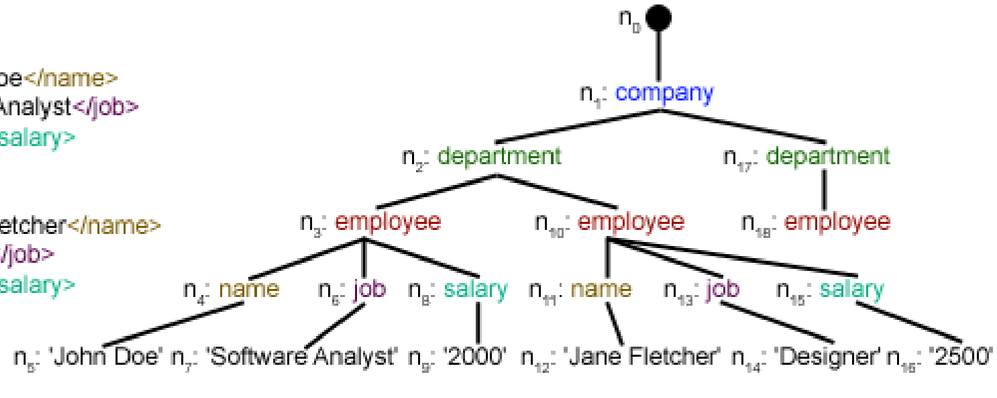
XML

- El formato **XML** (eXtensible Markup Language) es parecido al HTML, pero más estructurado.

- Los archivos XML forman una estructura de tipo árbol




```
<company>
 <department>
  <employee>
   <name>John Doe</name>
   <job>Software Analyst</job>
   <salary>2000</salary>
  </employee>
  <employee>
   <name>Jane Fletcher</name>
   <job>Designer</job>
   <salary>2500</salary>
  </employee>
 </department>
 <department>
  <employee>
  </employee>
 </department>
</company>
```



JSON

- **JSON** (JavaScript Object Notation) es un formato para el intercambios de datos. Un objeto json se forma con pares atributo-valor, éstos deben estar encerrados entre llaves { , } que es lo que definen el inicio y el fin del objeto.

- Una de las mayores ventajas que tiene el uso de JSON es que puede ser leído por cualquier lenguaje de programación.


```
{
    "orders": [
        {
            "orderno": "748745375",
            "date": "June 30, 2088 1:54:23 AM",
            "trackingno": "TN0039291",
            "custid": "11045",
            "customer": [
                {
                    "custid": "11045",
                    "fname": "Sue",
                    "lname": "Hatfield",
                    "address": "1409 Silver Street",
                    "city": "Ashland",
                    "state": "NE",
                    "zip": "68003"
                }
            ]
        }
    ]
}
```

JSON y XML vs CSV

student_id	age	score
1	12	77
2	12	68
3	11	75


```
[
 {
  "student_id":1,
  "age":12,
  "score":77
 },
 {
  "student_id":2,
  "age":12,
  "score":68
 },
 {
  "student_id":3,
  "age":11,
  "score":75
 }
]
```

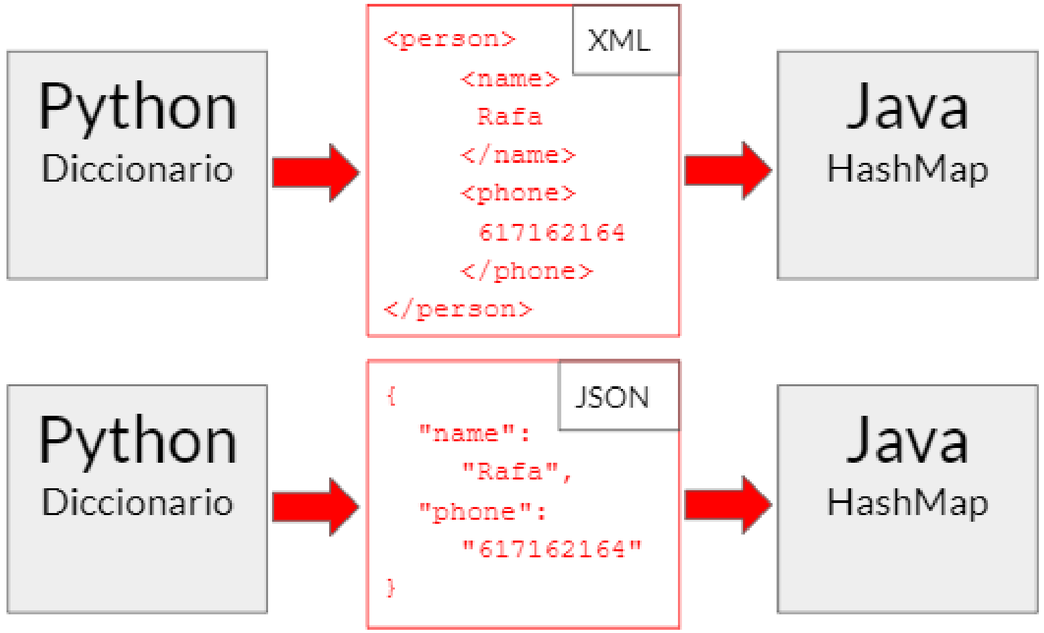

- XML y JSON son complementarios
- Los formatos CSV son en general más compactos que los formatos XML y JSON, siendo esta su principal ventaja.
- Por otro lado, CSV es el formato menos versátil y no permite crear jerarquías en los datos. Por ejemplo, los siguientes datos contienen jerarquías y es más difícil de plasmar en una tabla:

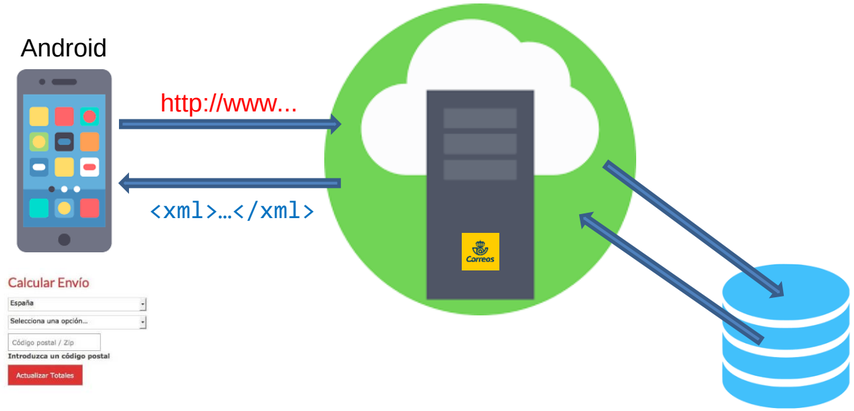

```
[
 {
  "student_id":1,
  "age":12,
  "subjects":{
   "mathematics":{
    "scores":[7,8,7,10],
    "final_score":8
   },
   "biology":{
    "scores":[6,6,5,7],
    "final_score":6
   }
  }
 }
]
```


NOTA: La estructura de un JSON puede ser bastante compleja. Es recomendable el uso de webs como codebeautify.org para representar los JSON en forma de árbol.

Serialización

La **serialización** consiste en un proceso de codificación de un objeto con el fin de transmitirlo a través de una conexión en red





Servicios Web

Principales **ventajas**

- Permiten que compañías en todo el mundo combinen fácilmente servicios y software para proveer servicios integrados.
- Son fáciles de entender debido a que fomentan los estándares y protocolos basados en texto.
- Disminuyen el tiempo de desarrollo de las aplicaciones.
- No están ligados a ningún Sistema Operativo o Lenguaje de Programación.

Concepto de API

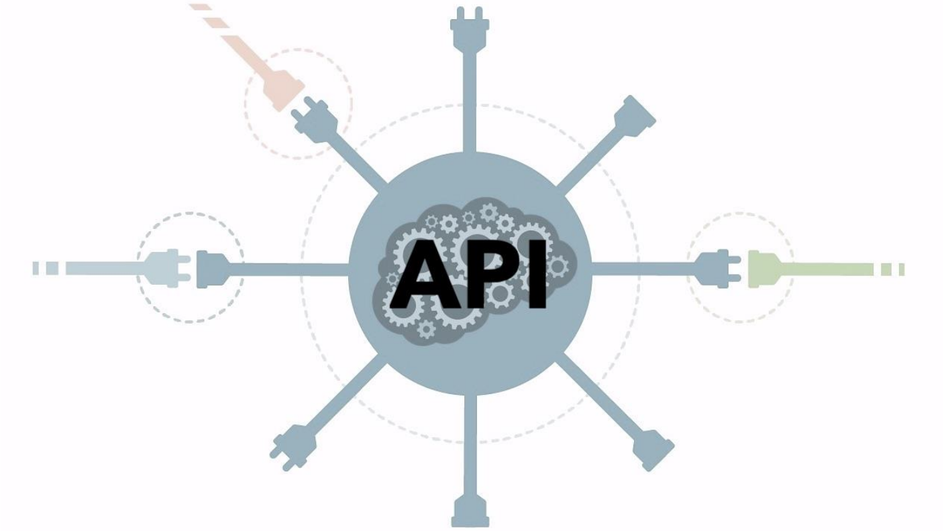
Hasta ahora tenemos:

- Capacidad de intercambiar datos entre aplicaciones utilizando el protocolo HTTP
- Un modo de representar estructuras complejas de datos para poder enviar y recibir datos entre las aplicaciones a través de XML o JSON

El paso siguiente consiste en definir y documentar "contratos" entre aplicaciones usando estas técnicas. Estos contratos reciben el nombre de **Interfaces de Programación de Aplicaciones** (***A****pplication* ***P****rogramming* ***I****nterfaces*), o **API**s

Concepto de API

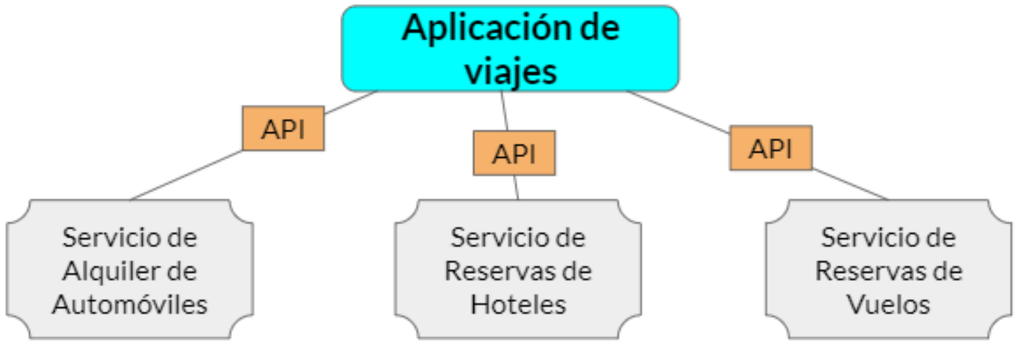
Cuando se utiliza una API, un programa crea un conjunto de servicios disponibles para que los usen otras aplicaciones y publica las APIs (“reglas”) que deben seguirse para acceder a los servicios proporcionados por el programa, permitiendo que las aplicaciones se conecte, se comuniquen y compartan información.



Arquitectura Orientada a Servicios (SOA)

Cuando los programas acceden a servicios proporcionados por otros, se utiliza un planteamiento llamado **Arquitectura Orientada a Servicios** o **SOA**

Un planteamiento SOA es aquel en el que una aplicación principal usa los servicios de otras aplicaciones



Tipos de APIs

- **APIs de servicios web**: permiten el intercambio de información entre un servicio web y una aplicación

- APIs basadas en bibliotecas
- APIs basadas en clases
- APIs de sistemas operativos

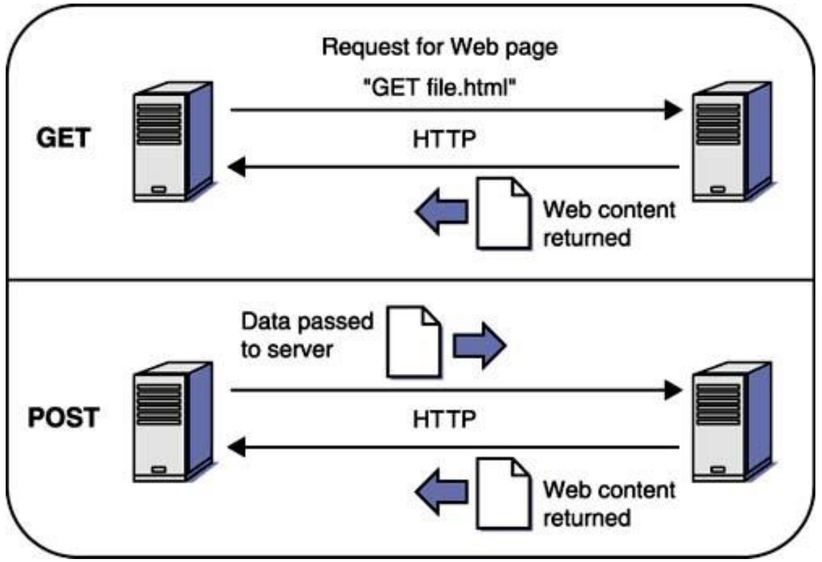
API REST

- La mayoría de las empresas utilizan API REST para crear servicios web
- Funcionan de manera similar a una web

- Las llamadas a la API se implementan como peticiones HTTP
- Las operaciones que nos permiten obtener datos en HTTP son **GET** y **POST**
- Ejemplo: https://docs.openaq.org/

Métodos HTTP: GET y POST

- El concepto **GET** consiste en obtener información del servidor. Esto puede realizarse a través de un navegador.
- El concepto **POST** consiste en **enviar** información desde el cliente para que sea procesada y actualice o agregue información en el servidor (por ejemplo, el envío de un formulario). No puede realizarse con un navegador
- Existen herramientas, como **Postman**, que permiten realizar peticiones HTTP, y probar cualquier API REST



Llamada a una API

- En primer lugar, se recomienda leer detenidamente la **documentación** de la API

- Ejemplo:

  Para llamar a esta API nos conectamos a esta URL “fija”, a la cual se le introducen los parámetros

Latest - GET

Provides the latest value of each available parameter for every location in the system.

GET

```
https://api.openaq.org/v1/latest
```

Parámetro

Campo		Tipo	Descripción
city	opcional	string	Limit results by a certain city.
country	opcional	string	Limit results by a certain country.
location	opcional	string	Limit results by a certain location.
parameter	opcional	string	Limit to only a certain parameter. Valores permitidos: `pm25` , `pm10` , `so2` , `no2` , `o3` , `co` , `bc`

https://api.openaq.org/v1/latest?city=Madrid&parameter=no2

Se utiliza el interrogante (?) para introducir parámetros, separados por &

¿Por qué utilizer APIs?

- Permiten **ofrecer** datos a otras aplicaciones o desarrolladores en un formato estándar
- Permiten **consumir** datos de otras aplicaciones
- Existen infinidad de proveedores de APIs (Facebook, Youtube, Amazon, Twitter, etc.)
- Directorio de proveedores de APIs:

https://www.programmableweb.com/apis/directory

Seguridad y uso de APIs

- Muchas veces se necesita autenticación para hacer uso de APIs comerciales
- Esto permite saber quién usa los servicios y cómo los usa
- Pueden existir distintos niveles de servicios (gratuitos y de pago), o políticas que limitan el número de peticiones que un usuario puede realizar durante un determinado periodo de tiempo

Seguridad y uso de APIs

Existen tres métodos de autenticación:

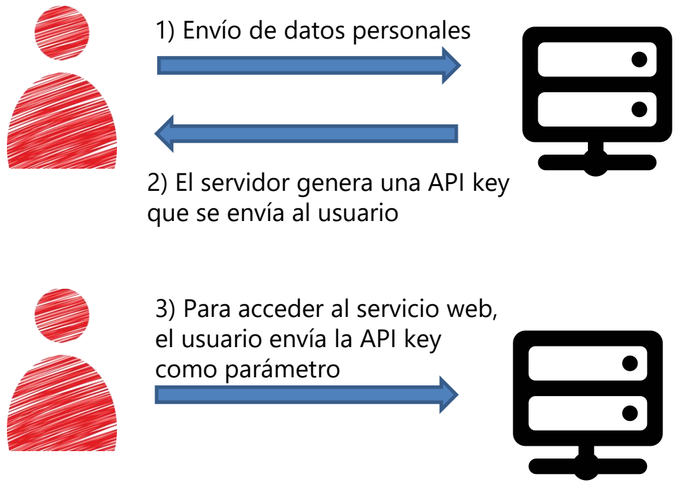
- **HTTP Basic Authentication**: utilizando un usuario y contraseña que se envía de forma codificada en la cabecera de la petición HTTP. Este método es el menos seguro.

- **API key**: cadena larga de caracteres, que se asigna a un consumidor de una API cuyo valor es único y es utilizada por parte de ese consumidor en cada una de las solicitudes a la API.

- **Open Authorization** (**OAuth**) es un protocolo que permite la autorización segura de una API de modo estándar y simple, basada en el uso de un token de acceso (o **access token**)

Seguridad y uso de APIs

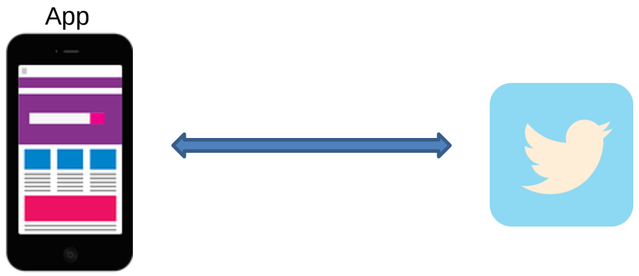
API key



Seguridad y uso de APIs

OAuth

- OAuth es un framework que permite delegar la autorización de acceso a las APIs
- Los tokens de acceso no contienen información sobre la identidad del usuario
- Empresas como Twitter o Facebook utilizan este framework



APIs de Inteligencia Artificial

- No todas las APIs sirven para devolver datos, algunas realizan determinadas operaciones o cálculos, incluso aplican Inteligencia Artificial
- La plataforma **meaningcloud**, por ejemplo, permite hacer procesamiento de lenguaje natural
- Ejemplo: **Análisis de sentimientos**

APIs de Google Cloud

- **Cloud Vision API**: Detección de etiquetas, texto, contenido explícito, rostros, atributos de imágenes , logos, búsqueda en internet de imágenes similares, etc.
- **Cloud Video Intelligence API**: Detección de etiquetas, cambios de escena, transcripción de audio a inglés, moderación de contenido
- **Cloud Translation API**: Detección del lenguaje , traducción de texto
- **Cloud Natural Language API**: Análisis de texto, sentimiento
- **Cloud Speech API**: Reconocimiento automático de voz, contenido inapropiado, etc.

APIs de Microsoft Azure

https://azure.microsoft.com/es-es/services/cognitive-services/

- **Emotion API**: Detección de emociones
- **Computer Vision API**: Análisis de imágenes (Obtener información del contenido de la imágenes, detección de contenido adulto, detección de esquemas de color), detección de texto
- **API de reconocimiento facial**: Comparación de rostros
- **Video Indexer**: Obtener información de vídeos (transcripción de audio, seguimiento e identificación de rostros, reconocimiento de texto,…)

APIs de Microsoft Azure (lenguaje)

https://azure.microsoft.com/es-es/services/cognitive-services/

- **Translator Text API**: API de traducción automática de texto
- **Bing Spell Check API**: Detección y corrección de errores ortográficos
- **Text Analytics**: Análisis de texto (idiomas, frase claves, opiniones…)
- **Linguistic Analysis**: Análisis de la estructura de un texto (sintaxis, tokens,…)

Gracias

Rafa Zambrano

rafael@thebridgeschool.es

THE BRIDGE