API

JSON — *JavaScript Object Notation*

JSON è un formato testuale derivato dalla sintassi di JavaScript, ma oggi usato universalmente (non solo in JavaScript).

Caratteristiche principali:

- Basato su **coppie chiave-valore**
- Usa **parentesi graffe** `{}` per oggetti e **parentesi quadre** `[]` per liste
- Tutte le **stringhe vanno tra virgolette doppie** `"`
- Molto usato per API, file di configurazione, e trasferimento dati web

ESEMPIO:

```
{
  "nome": "Luca",
  "eta": 25,
  "linguaggi": ["Python", "C", "JavaScript"],
  "isStudente": true
}
```

Il file JASON utilizza solo due concetti familiari:

- object(oggetto)
- array

Oggetto

E' una collezionenon ordinata di coppie `nome: valore`, racchiusa tra parentesi graffe `{}`.

- *nome*. stringa tra `" "`, unica all'interno dell'oggetto (a.k.a. "chiave o "campo").
- *valore*: numero, stringa, booleano, null array, object.

Array

Elenco ordinato di valori, separati da virgole, racchiuso tra parentesi quadre `[]`.
Esempio:

```
{
    "wasaWeekdays": ["tuesday", "thursday"]
}
```

YAML — *YAML Ain't Markup Language*

YAML è un formato alternativo a JSON, più **leggibile per gli esseri umani**.
È spesso usato per file di **configurazione** (es. Docker, GitHub Actions, Kubernetes).

Caratteristiche principali:

- Stessa struttura logica di JSON, ma più "naturale"
- Usa **indentazione (spazi)** invece di parentesi
- **Niente virgolette** obbligatorie per le stringhe
- Supporta **commenti** (`# ...`), cosa che JSON non fa

Esempio YAML (equivalente al JSON sopra):

```yaml
nome: Luca
eta: 25
linguaggi:
  - Python
  - C
  - JavaScript
isStudente: true
```

API - APPLICATION PROGRAMMING INTERFACE

Un'*API (Application Programming Iterface)* è la **definizione delle interazioni consentite** tra due parti di un software.
L'API funge da contratto, specificando come un pezzo di codice o un servizio può interagire con un altro.

Elementi di un Contratto

L'API definisce in dettaglio il "contratto" di interazione tra il ***CONSUMER***(client) e il ***PROVIDER***(servizio). Essa specifica:

- le Richieste possibili
- i Parametri delle richieste
- i valori di ritorno
- qualsiasi formto di dato richiesto (es. JASON, XML, YAML).

Benefici

L'adozione di un'API porta vantaggi fondamentali nell'architettura software:

- **Interfaccia Esplicita**: definisce chairamente le aspettative e le modalità di interazione
- ***Cotratto Infrangibile***: stabilisce un insieme di regole che entrambe le parti devono rispettare.
- ***Information Hiding*** (nascondimento delle informazioni): la logica interna dl Provider (come i dati vengono processati o archiviati) rimane nascosta al Consumer. Il Client deve solo conoscere l'interfaccia.

API pubbliche

Disponibili solo per l'uso da parte del pubblico. L'acceso può essere limitato solo ad alcuni utenti tramite ***API Tokens***.

API Private

Destinate all'uso interno di un azienda o un sistema chiuso. L'accesso è limitato ai componenti interni.

Interfaccia e Stabilità

- Stabilità dell'interfaccia: i cambiamenti protrebbero rompere la compatibilità con i client esistenti.
- Marcatori di Stato:
  - `beta`: indica che le parti potrebbero cambiare perchè non sono ancora stabili.
  - `deprecated`: indica che le parti verranno rimosse o non saranno più supportate in futuro.

Documentazione e Definizione

Per stabilire il contratto, l'API deve essere definita esplicitamente.

- La definizione può avvenire attraverso *documentazione*.
- Oppure, attraverso un lingaggio di descrizione standardizzato.
  Un linguaggio di descrizione formalizzato il contratto, consentendo la generazione automatica di documentazione, il codice client e validazione.

OAS - OpenAPI Specification

OAS (OpenAPI Specification) è un linguaggio di descrizione leader del settore per le API moderne baate su HTTP.

- è un format di descrizione vendor-neutral per le API remote basate su HTTP.
- rappresenta lo standard industriale per la descrizione delle API moderne.
- E' ampiamente adottato dalla comunità di sviluppo.
  I file OpenAPI sono tipicamente scritti in formato ***YAML***.
  ESEMPIO:

```
openapi: 3.0.0
info:
    title; An example OpenAPI document
    description:
        This API writing down marks on a TicTac board and requesting the state of the board or of the
individual cells.
    version: 0.0.1

paths: {} # gli endpoint dell'API verrebbero definiti qui.
```