JSON (JavaScript Object Notation) è un formato

per interscambio dati tra applicazioni

client-server.

È molto simile alla sintassi utilizzata per la

creazione di oggetti in JS utilizzando la

notazione letterale

I tipi di dati supportati all'interno di un JSON

sono: number, integer, string, boolean, object,

array, null.

I file contenenti questo tipo di sintassi hanno

estensione .json. A screenshot of a computer

Description automatically generated

Un documento JSON è composto da dati

organizzati in coppie chiave/valore (come gli oggetti

in JavaScript).

È importante notare alcune differenze rispetto alla

notazione letterale di JS: ad esempio un valore non

potrà mai essere una funzione o un metodo.

Inoltre, anche i nomi delle proprietà vanno inseriti

tra doppi apici (gli apici singoli o gli altri delimitatori

di stringa non sono accettati).

Le coppie chiave/valore sono sempre separate da

virgola, tranne nel caso risulti essere l'ultima

all'interno di un oggetto oppure nel caso il valore

risulti essere l'ultimo all'interno di un array.

A screenshot of a computer code

Description automatically generated

A close up of a logo

Description automatically generated

· **localStorage** e **sessionStorage** sono due

tecnologie di storage incluse nella grande

maggioranza dei browser moderni al fine di

salvare un piccolo quantitativo di dati in locale.

· localStorage salva dei dati in maniera

permanente, fino alla completa cancellazione

della cache del browser o fino alla loro specifica

rimozione.

· sessionStorage lavora come localStorage, ma alla

fine della sessione (quando il tab o il browser

viene chiuso) i dati salvati al suo interno vengono

automaticamente cancellati.

A diagram of a web storage

Description automatically generatedlocalStorage e sessionStorage utilizzano gli stessi

metodi per interagire con il salvataggio dei dati:

· **setltem**: salva un dato

· **getltem**: ritorna un dato salvato

· **removeltem:** rimuove un dato salvato

· **clear:** rimuove ogni dato sul dominio corrente

A screenshot of a computer

Description automatically generated

I dati all'interno dei motori di Web Storage possono

venire salvati solamente in formato stringa.

Molto spesso, tuttavia, è necessario salvare e

recuperare delle informazioni in formato più

complesso, come il JSON.

È possibile salvare dati in formato JSON all'interno

di localStorage/sessionStorage a patto di convertirli

precedentemente in formato stringa: per farlo è

possibile utilizzare un metodo integrato nell'oggetto

JSON di JavaScript: stringify().

Quando è necessario recuperare i dati dai motori di

Web Storage e ri-convertirli da stringa a oggetto, è

possibile utilizzare il metodo JSON.parse().

A screenshot of a computer code

Description automatically generated

Il contenuto dei motori di Web Storage può

tranquillamente essere esplorato dagli strumenti di

sviluppo dei browser.

A screenshot of a computer

Description automatically generated

A close-up of a logo

Description automatically generated

. HTTP è l'acronimo di HyperText Transfer Protocol

· È un protocollo utilizzato come principale sistema

per la trasmissione di informazioni sul web tra

client/server

. Un client invia una REQUEST al server

. Il server risponde al client con una RESPONSE

A diagram of a computer and a computer

Description automatically generated

**HTTP REQUEST**

. URL=> https://google.com/apis/test

. **Method=> GET, POST, PUT, PATCH, DELETE** ...

. Headers=> Authentication, cookies, Content-Type

· Payload =>data o empty

**HTTP REQUEST - URL**

· L'URL è l'indirizzo dell'endpoint che stiamo contattando

. Inizia con HTTP o HTTPS

· HTTPS è la versione sicura (crittografata) di http

**HTTP REQUEST - METHOD**

· Il metodo indica all'endpoint lo scopo della nostra richiesta

- GET = voglio delle informazioni

- POST = sto inviando delle informazioni

- PUT voglio modificare qualcosa

- DELETE voglio eliminare qualcosa

· Esistono molti altri metodi, questi sono i più utilizzati nelle API moderne

**HTTP REQUEST - HEADERS**

· Le intestazioni specificano alcune informazioni aggiuntive sulla richiesta

- Content-Type = invio dati in formato JSON

- Authorization = questa è la mia chiave per autorizzarmi

- Accept = mi aspetto di ricevere alcuni dati JSON

- Cookie = Vuoi un biscotto?

· Ce ne sono molti altre e puoi anche creare le tue intestazioni

**HTTP REQUEST - PAYLOAD**

· Nel payload invii effettivamente alcune informazioni

· È strettamente correlato all'intestazione Content-Type

· Inviamo il payload solo su richiesta POST (crea) e PUT (modifica)

· È anche noto come il corpo (body) della richiesta

**HTTP RESPONSE**

Viene inviata dal server per rispondere alla Request

· Payload = contiene le informazioni richieste dalla request

. Status Code fornisce il risultato dell'operazione

· Headers = meta-informazioni aggiuntive

**HTTP RESPONSE - STATUS CODE**

· 200 OK = contiene le informazioni richieste dalla request

· 201 Created = fornisce il risultato dell'operazione

· 400 Bad request #c'è qualcosa che non va nella tua request

· 401 Unauthorized = problema nell'autorizzazione

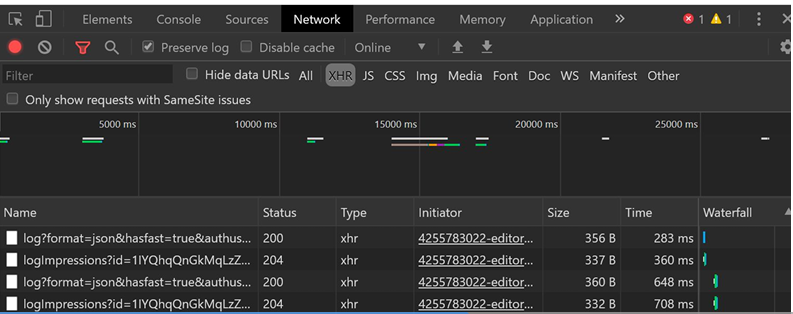
. 404 Not found = non riesco a trovare le informazioni che stai cercando

· 500 Internal server error = qualcosa non va, lato server

**HTTP EXAMPLES**

. Negli developer tools del tuo browser troverai la scheda "network"

· Da lì è possibile vedere ed esplorare ogni richiesta che il tuo browser sta inoltrando



**AJAX**

. Asynchronous JavaScript and XML

· AJAX permette di inviare richieste HTTP dalla nostra applicazione in modo asincrono

· **Con AJAX puoi:**

- Leggere dati da un server web - dopo che la pagina è stata caricata

- Aggiornare una parte del documento web senza ricaricare la pagina

- Inviare dati a un server web in background

A close up of a sign

Description automatically generated

REST definisce un insieme di principi

architetturali per la progettazione di un

sistema.

REST è l'approccio più recente per utilizzare

API, è flessibile e molto adatto per le

applicazioni Web poiché si basa su HTTP per

comunicare con il server.

REST = REpresentational State Transfer

A diagram of a cloud

Description automatically generated

**Principi fondamentali di REST:**

. Identificazione delle risorse

· Utilizzo esplicito dei metodi HTTP

. Risorse autodescrittive

· Comunicazione stateless

**Identificazione delle risorse**

· Le risorse sono gli elementi fondamentali su cui si

basano i Web Service RESTful

· Per risorsa si intende un qualsiasi elemento

oggetto di elaborazione

. Ciascuna risorsa deve essere identificata

univocamente

· Il meccanismo più naturale per individuare una

risorsa è dato dal concetto di URI

A close-up of a website

Description automatically generated

**Utilizzo esplicito dei metodi http**

L'uso esplicito dei metodi HTTP ci indica di sfruttare

i metodi (o verbi) predefiniti di questo protocollo, e

cioè GET, POST, PUT e DELETE.

A close-up of a list

Description automatically generated

**Risorse autodescrittive**

· Le risorse sono concettualmente separate dalle rappresentazioni restituite al client.

. Un Web Service non invia al client direttamente un record del suo database, ma una sua rappresentazione in una codifica dipendente dalla richiesta del client e/o dall'implementazione del servizio.

**Comunicazione stateless**

· La comunicazione client-server è vincolata in modo che nessun contesto client venga memorizzato sul server tra le richieste

· Ciascuna richiesta dai vari client contiene tutte le informazioni necessarie per richiedere il servizio e lo stato della sessione è contenuto nel client

**CRUD**

· CRUD significa Create, Read, Update, Delete

· Queste sono le quattro funzioni di base del PERSISTENT STORAGE

L'implementazione del CRUD in un'applicazione Web consente di leggere e scrivere contenuti in modo persistente

Questi comandi vengono eseguiti utilizzando metodi http

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Postman** è uno strumento che permette di eseguire

richieste HTTP ad un server.

Possiamo effettuare una richiesta specificando

diversi parametri e ispezionare ogni informazione

ricevuta dalla response.

<https://www.postman.com/>

· In alto a destra, con il pulsante new, possiamo

creare le richieste che vogliamo testare.

· Postman permette di creare anche collezioni e

altri oggetti avanzati.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

. Creiamo la prima richiesta, inserendo il nome con

la quale salvarla e la collection dove salvarla.

· Faremo una semplice get a questo indirizzo:

· https://swapi.dev/api/people/1

· Selezioniamo il metodo GET e send senza altri

parametri.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

· Nella parte bassa della finestra abbiamo la

risposta che possiamo formattare con diverse

opzioni.

. Oltre al body della richiesta è interessante notare

gli headers ricevuti.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

. Con Postman possiamo testare tutte le API come

abbiamo fatto con la GET.

· In POST dovremo inviare dei dati.

· Quindi impostiamo la chiamata con metodo post

e la url.

· Nella parte body scegliamo l'opzione

x-www-form-urlencoded.

· Inseriamo le coppie key-value.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated