

Programmazione Mobile

Nicola Noviello

nicola.noviello@unimol.it

Corso di Laurea in Informatica
Dipartimento di Bioscienze e Territorio
Università degli Studi del Molise
Anno 2023/2024

Servizi RESTful e Documentazione

- Scambio dati mediante JSON
- Documentazione dei servizi con Swagger e Postman
- Integrazione con Api RESTful pubbliche



Introduzione ai Servizi RESTful

Cosa sono le API (application programming interface)?

Sono insiemi di definizioni e protocolli che consentono a diversi software di comunicare tra loro. Sono essenzialmente ponti che consentono a due applicazioni di parlare tra loro, consentendo loro di accedere e condividere dati e funzionalità.



Ruolo delle API

Le API fungono da interfaccia tra il front-end e il back-end di un'applicazione, consentendo a diverse parti di un sistema software di interagire tra loro in modo uniforme e standardizzato. Possono essere utilizzate per accedere a risorse remote, eseguire operazioni specifiche e scambiare dati tra diverse applicazioni.

Sono usate per **scambiare dati** e **offrire informazioni**.




Introduzione ai Servizi RESTful

Rappresentano un paradigma di architettura software per lo sviluppo di applicazioni web.

Offrono un approccio semplice e flessibile per lo scambio di dati tramite interfacce API standardizzate.

Consentono ai sistemi software di comunicare tra loro attraverso il protocollo HTTP seguendo i principi dell'architettura REST (REpresentational State Transfer). Questo approccio si basa sull'uso dei metodi HTTP (GET, POST, PUT, DELETE) per operare su risorse identificate da URL.



Principi e caratteristiche dei Servizi RESTful

- **Stateless**: ogni richiesta contiene tutte le informazioni necessarie per essere processata, senza dipendere dallo stato precedente
- **Principio di scalabilità e alte performance**: le operazioni sono a basso accoppiamento e ad alta coesione
- **URL**: Univoci ed intuitivi
- **Verbi HTTP**: per indicare l'operazione da eseguire sulle risorse
- **JSON**: Leggero e flessibile, perfetto per la rappresentazione delle risorse



Vantaggi dell'utilizzo di Servizi RESTful

Sono presenti numerosi vantaggi, tra cui la facilità di integrazione, la scalabilità e la flessibilità. Grazie alla loro architettura semplice e leggera, i servizi REST permettono uno scambio di informazioni rapido e efficiente, adatto a una vasta gamma di applicazioni web e mobile.

È garantita una maggiore indipendenza tra client e server, consentendo uno sviluppo più agile e modulare delle applicazioni. Questo si traduce in una minore complessità e un più rapido time-to-market.



Scambio di dati mediante JSON

JSON (JavaScript Object Notation) è un formato leggero per lo scambio di dati strutturati, indipendente dal linguaggio di programmazione. È diventato lo standard de facto per lo scambio di dati nelle applicazioni web e mobile moderne, grazie alla sua semplicità e facilità di lettura e scrittura.

Attraverso l'utilizzo di JSON, le API RESTful possono scambiare agevolmente dati strutturati, come oggetti e array, tra il client e il server, facilitando l'integrazione tra sistemi e applicazioni diverse.



I metodi HTTP

Comandi che vengono inviati da un client a un server per specificare l'azione da eseguire su una risorsa identificata. Questi metodi definiscono il tipo di operazione che deve essere eseguita sul server.

Sono utilizzati per definire le operazioni CRUD (Create, Read, Update, Delete) su risorse Web. Ogni metodo ha un significato specifico e deve essere utilizzato in base all'azione richiesta.



I metodi HTTP

- **GET**: Utilizzato per richiedere dati da una risorsa specificata
- **POST**: Utilizzato per inviare dati al server per creare una nuova risorsa
- **PUT**: Utilizzato per inviare dati al server per *sostituire* una risorsa esistente
- **DELETE**: Utilizzato per eliminare una risorsa specificata dal server
- **PATCH**: Utilizzato per applicare modifiche parziali a una risorsa (vedi *aggiornare*)



Struttura di una richiesta e risposta REST

Richiesta

La richiesta REST è composta da un metodo HTTP, un endpoint e, opzionalmente, parametri e un corpo dati in formato JSON


Risposta

La risposta REST contiene uno status code che indica l'esito dell'operazione e, opzionalmente, un corpo dati in formato JSON con i risultati

Header

Le intestazioni HTTP (header) forniscono informazioni aggiuntive sulla richiesta e sulla risposta, come il tipo di contenuto, l'autenticazione, la cache e altro

Gestione delle risposte e degli errori

- **Codici di Stato HTTP:** I codici di stato HTTP standard servono per comunicare l'esito delle richieste in modo chiaro e affidabile
 - **Messaggi di Errore Descrittivi:** Messaggi di errore dettagliati aiutano gli sviluppatori a comprendere e risolvere rapidamente i problemi
 - **Gestione delle Eccezioni:** Una solida gestione delle eccezioni garantisce la stabilità e l'affidabilità delle tue API RESTful
 - **Logging ed Analisi degli Errori:** Archiviare ed analizzare gli errori serve per identificare e risolvere rapidamente i problemi ricorrenti. Utile anche in ottica ML
- 

JSON (JavaScript Object Notation)

Introduzione a JSON (JavaScript Object Notation)


JSON è un formato di dati leggero e leggibile utilizzato, tra le altre cose, per lo scambio di informazioni tra client e server nelle applicazioni web e nelle API RESTful.

Basato sulla sintassi di JavaScript, JSON utilizza una struttura di coppie chiave-valore che lo rende facile da comprendere per gli sviluppatori e da elaborare per le applicazioni.

JSON ha anche la caratteristica di essere human-readable.




Vantaggi dell'utilizzo di JSON per lo scambio dati

- **Semplicità e Leggibilità:** La sintassi semplice e intuitiva di JSON lo rende facile da leggere e scrivere per gli sviluppatori. La struttura chiave-valore consente di organizzare i dati in modo chiaro e comprensibile
 - **Compatibilità e Flessibilità:** JSON è supportato da quasi tutti i linguaggi di programmazione e framework, rendendolo un formato di dati universale per lo scambio di informazioni tra diverse piattaforme e tecnologie
 - **Leggerezza e Efficienza:** JSON è un formato di dati leggero che richiede meno banda rispetto ad altri formati come XML, rendendolo ideale per le comunicazioni su Internet, soprattutto in applicazioni mobile e a bassa larghezza di banda
 - **Interoperabilità:** JSON facilita l'interoperabilità tra sistemi software eterogenei, consentendo loro di comunicare e scambiare dati in modo standardizzato e compatibile
- 

Esempio di oggetto JSON

```
{  
  "nome": "Mario",  
  "cognome": "Rossi",  
  "eta": 30,  
  "indirizzo": {  
    "via": "Via Roma",  
    "città": "Roma",  
    "CAP": "00100"  
  }  
}
```






Documentazione API con Swagger e Postman

Documentazione API con Swagger e Postman

La documentazione di API REST svolge un ruolo cruciale per la loro adozione e utilizzo. Due strumenti ampiamente utilizzati sono **Swagger** e **Postman**. **Swagger** fornisce una piattaforma per la creazione di documentazione interattiva, permettendo agli sviluppatori di esplorare e testare le API direttamente dal browser. **Postman**, invece, è un client API che facilita la gestione delle richieste, la visualizzazione delle risposte e il testing delle API.

Entrambi gli strumenti sono fondamentali per garantire una buona esperienza d'uso delle API, favorendo l'adozione da parte degli sviluppatori e la collaborazione all'interno dei team di sviluppo.



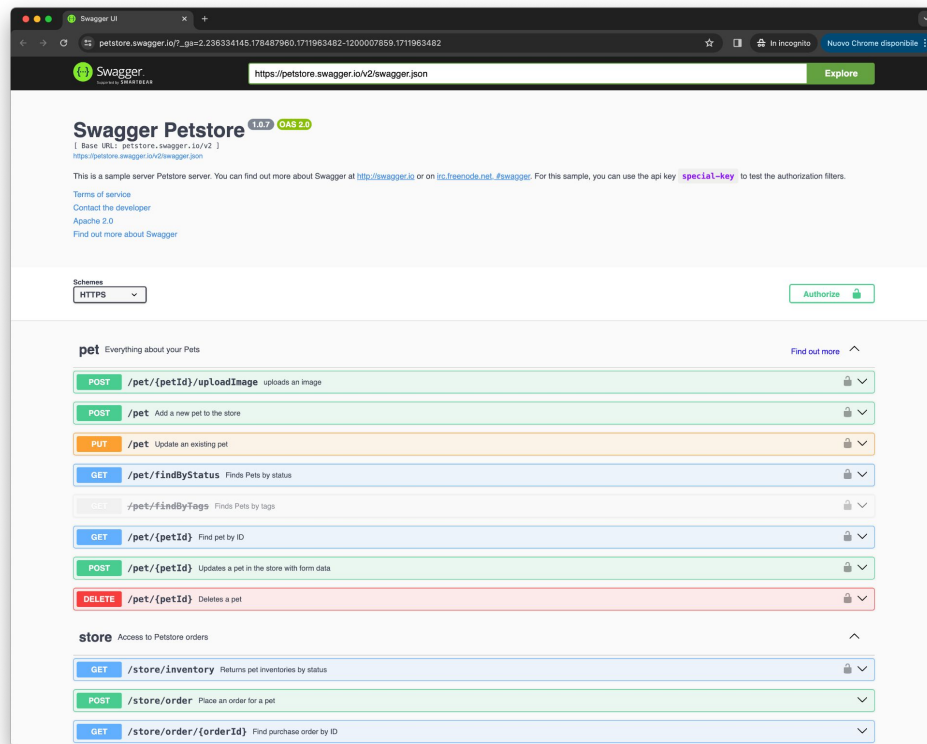
Documentazione dei servizi con Swagger

Swagger è un framework open-source per la progettazione, la documentazione e il testing di API RESTful. OpenAPI è un formato di specifica che consente di descrivere in modo standardizzato le API RESTful utilizzando JSON o YAML.

Swagger semplifica il processo di documentazione, fornendo una rappresentazione visiva dell'API e consentendo agli sviluppatori di esplorare e testare l'API direttamente dall'interfaccia Swagger UI.



Esempio di Documentazione Swagger



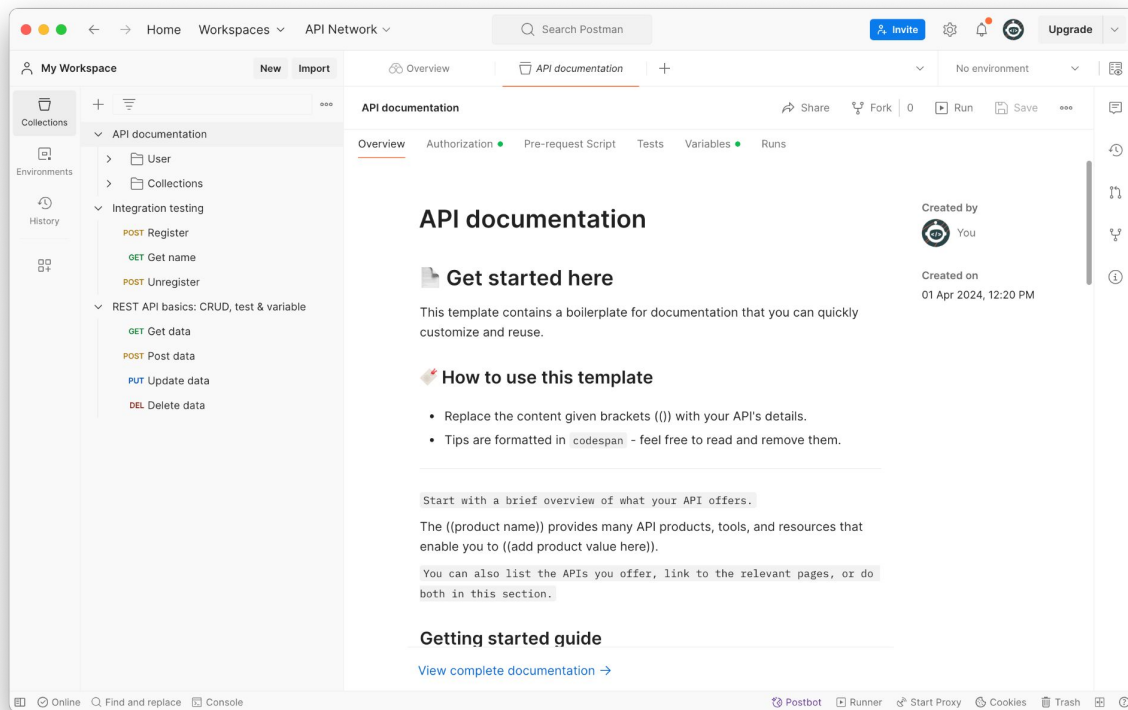
Documentazione dei servizi con Postman

Postman è una piattaforma che consente agli sviluppatori di testare e documentare API. È ampiamente utilizzato per eseguire richieste HTTP, testare servizi e generare documentazione.

Offre una suite completa di strumenti per la documentazione, compresi l'organizzazione delle richieste in raccolte, la creazione di ambienti di test e la generazione automatica di documentazione.



Esempio di Documentazione Postman



API RESTful pubbliche

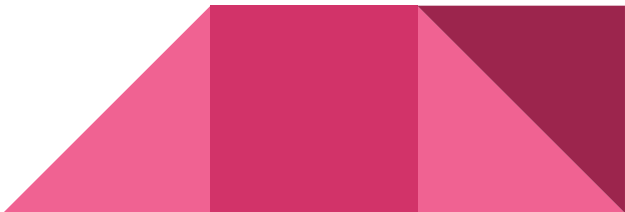
Panoramica sull'integrazione con API pubbliche

Le API pubbliche offrono un'ampia gamma di servizi e funzionalità che possono essere utilizzati nelle proprie applicazioni senza la necessità di sviluppare da zero tutte le funzionalità.

Queste API consentono agli sviluppatori di accedere a servizi esterni e a dati di terze parti in modo rapido ed efficiente, consentendo loro di arricchire le proprie applicazioni con funzionalità avanzate senza dover scrivere codice personalizzato per ogni singola funzionalità.



Vantaggi dell'integrazione con API pubbliche

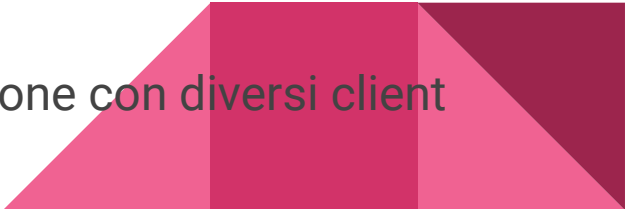
- **Risparmio di tempo e risorse:** È possibile risparmiare tempo e risorse evitando di dover sviluppare da zero funzionalità già disponibili attraverso API esterne
 - **Accesso a funzionalità avanzate:** Offrono spesso funzionalità avanzate e dati di alta qualità che sarebbe difficile o costoso da ottenere autonomamente
 - **Aggiornamenti e manutenzione:** Consente agli sviluppatori di beneficiare degli aggiornamenti e delle migliorie apportate dagli sviluppatori delle API, riducendo la necessità di mantenere e aggiornare costantemente le funzionalità interne
- 

Esempi di API pubbliche

- **GitHub:** <https://docs.github.com/en/rest?apiVersion=2022-11-28>
- **Reddit:** <https://www.reddit.com/dev/api/>
- **MediaWiki (Wikipedia):** https://www.mediawiki.org/wiki/API:Main_page#
- **Varie (200+ API):**
<https://mixedanalytics.com/blog/list-actually-free-open-no-auth-needed-apis/>



Considerazioni sull'integrazione con API pubbliche

- **Limiti e vincoli:** Le API pubbliche possono avere limiti di utilizzo, come limiti di rate e limiti di accesso ai dati, che gli sviluppatori devono tenere in considerazione durante l'integrazione
 - **Politiche di utilizzo:** È importante leggere e comprendere le politiche di utilizzo e le condizioni d'uso delle API pubbliche per assicurarsi di utilizzarle in modo conforme e appropriato.
 - **Sicurezza:** Gli sviluppatori devono prestare attenzione alla sicurezza durante l'integrazione con API pubbliche, assicurandosi di utilizzare metodi di autenticazione e autorizzazione sicuri per proteggere i dati sensibili e prevenire possibili violazioni della sicurezza
 - **SDK:** Spesso sono presenti anche SDK per l'integrazione con diversi client
- 

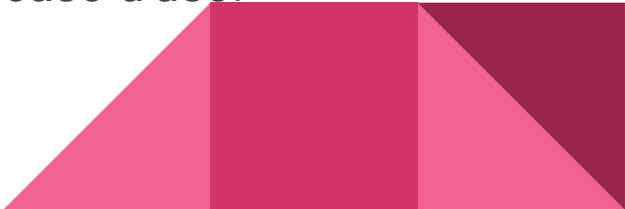


Tutto è RESTful?

SOAP e GraphQL

In aggiunta ai servizi RESTful, esistono altri paradigmi di comunicazione sincrona tra client e server, come SOAP (Simple Object Access Protocol) e GraphQL. SOAP è uno standard basato su XML per l'invio di messaggi, mentre GraphQL è un linguaggio di query per API che offre maggiore flessibilità nell'accesso ai dati rispetto all'approccio REST tradizionale.

Ciascuno di questi approcci presenta vantaggi e svantaggi, e la scelta dipende dalle esigenze specifiche dell'applicazione. È importante conoscere i diversi paradigmi per poter scegliere quello più adatto al proprio caso d'uso.



Lab 2

Laboratorio sull'utilizzo di Postman



Esercizio

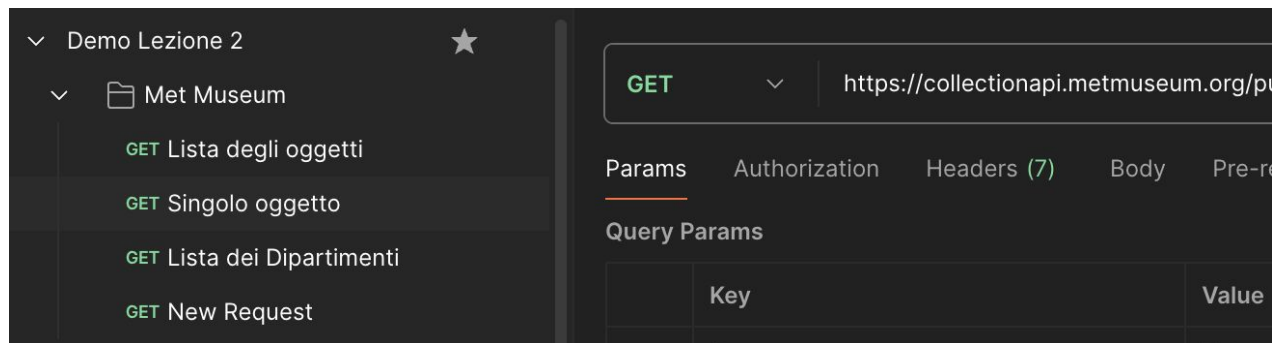
Esercizio: Valutazione di API e Integrazione in un'App

L'obiettivo di questo esercizio è quello di valutare API pubbliche, selezionare alcuni endpoint e creare una collezione Postman. Successivamente, si richiede di disegnare un wireframe che mostri come i dati acquisiti dalla collezione Postman possono essere integrati nella schermata di un'applicazione.



Deliverable dell'esercizio

- **Valutazione delle API:**
 - Ricerca e selezione di almeno tre API pubbliche rilevanti per il progetto
 - Analisi degli endpoint disponibili su ciascuna API
 - Scelta di quattro o cinque endpoint interessanti per il progetto
- **Creazione della Collection Postman:**
 - Utilizzando l'applicazione Postman, creare una nuova collezione
 - Aggiungere gli endpoint selezionati alla collezione
 - Configurare le richieste per ciascun endpoint, includendo eventuali parametri necessari



Deliverable dell'esercizio

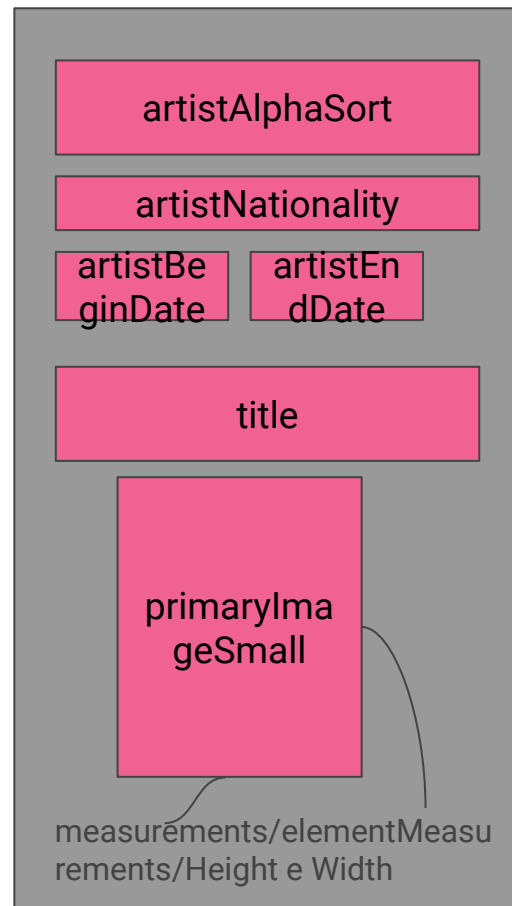
- **Test delle API:**

- Eseguire le richieste nella collezione Postman per verificare che restituiscano i dati previsti
- Verificare la correttezza dei dati ricevuti rispetto alle specifiche delle API

```
[,
  "department": "Modern and Contemporary Art",
  "objectName": "Drawing",
  "title": "Untitled (\"Letter of Love\" Illustration)",
  "culture": "",
  "period": "",
  "dynasty": "",
  "reign": "",
  "portfolio": "",
  "artistRole": "Artist",
  "artistPrefix": "",
  "artistDisplayName": "Andy Warhol",
  "artistDisplayBio": "American, Pittsburgh, Pennsylvania 1928-1987 New York",
  "artistSuffix": "",
  "artistAlphaSort": "Warhol, Andy",
  "artistNationality": "American",
  "artistBeginDate": "1928",
  "artistEndDate": "1987",
```

Deliverable dell'esercizio

- **Disegno del Wireframe:**
 - Utilizzando carta e penna o strumenti digitali, disegnare un wireframe che rappresenti una schermata dell'applicazione
 - Indicare chiaramente dove verranno visualizzati i dati provenienti dalle API nella schermata dell'app
- **Integrazione dei Dati:**
 - Sulla base del wireframe disegnato, identificare almeno un'area in cui i dati provenienti dalla collezione Postman possono essere integrati
 - Descrivere brevemente come i dati saranno presentati e utilizzati all'interno dell'applicazione



Esercizio: Valutazione di API e Integrazione in un'App

Consegna:

- Una collezione Postman contenente le richieste agli endpoint selezionati, esportata in formato JSON o Postman Collection v2;
- Il wireframe disegnato, mostrante la schermata dell'applicazione e indicante l'integrazione dei dati provenienti dalla collezione Postman;
- Una brevissima relazione che descrive il processo seguito per la valutazione delle API, la creazione della collezione Postman e il disegno del wireframe, includendo eventuali considerazioni e scelte progettuali;
- **Inviare tutto a nicola.noviello@unimol.it utilizzando come oggetto della e-mail "Lezione2: Nome Cognome Matricola";**
- 2h di tempo dalla fine della lezione, da svolgere in aula;
- Non obbligatorio, ma valutato positivamente ai fini dell'esame.



Link al Materiale

- Link (temporaneo) al materiale:

https://github.com/nicolanoviell/mobile_programming

