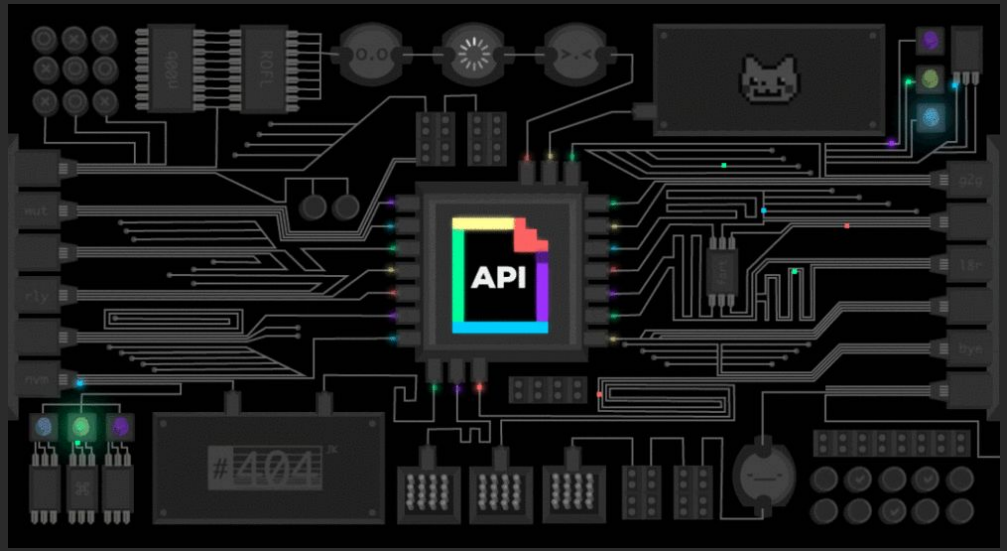
Introduzione all'architettura REST

Introduzione al formalismo REST per il trasferimento di informazioni tra applicazioni tramite rete.



Riccardo Bastianini - 17 Maggio 2017
Immagini via giphy - http://giphy.com

Sommario

- Introduzione a REST
  - Che cos’è
  - Che cosa non è
  - Elementi costitutivi
  - REST + HTTP
- Modellare un’API per un servizio di condivisione immagini

Chi sono / di cosa mi occupo?

- ★ Software Engineer @ Vendini (4+ anni)
  - Sviluppo API della piattaforma (~2 anni)
  - Sviluppo sistema multi-site basato su Drupal
- ★ Prima: mancato miliardario
  - Sviluppo web service di statistiche per social network
- ★ Ancora prima:
  - Ex-studente di Informatica @ UNIPG
  - DCM - www.duricomeilmetallo.net

REST a chi?

REpresentational

State

Transfer

“Architectural Styles and the Design of Network-based Software Architectures“ Roy Fielding (2000)

Che cosa non è?

Non è uno standard.

Non è un formato di dati.

Non è un protocollo.

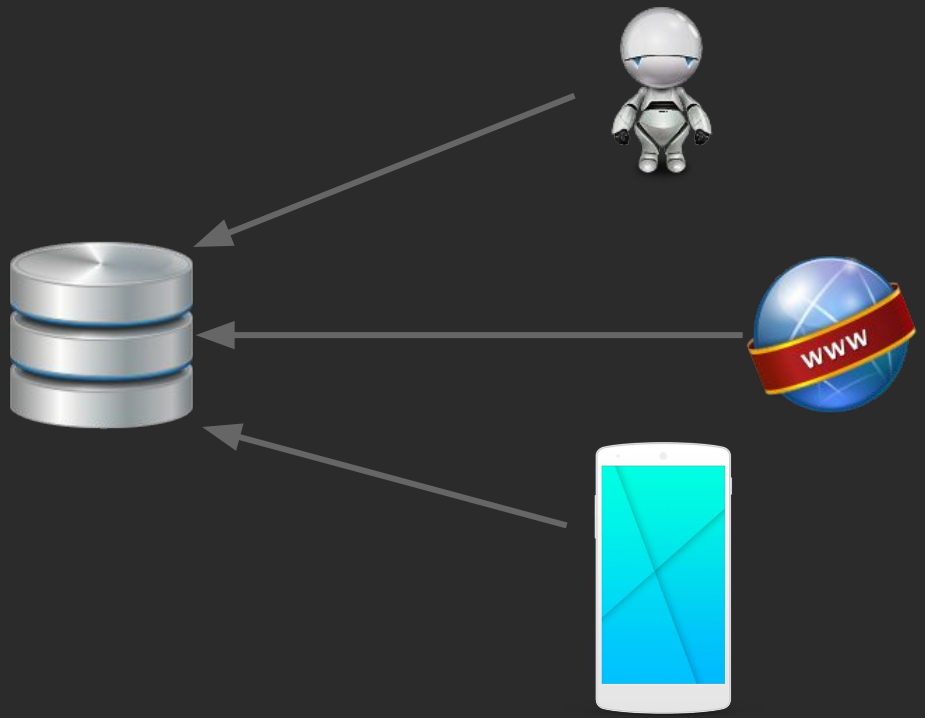
Che cos’è?

Stile architetturale per sistemi connessi via rete.

Descrive componenti, connessioni, ruoli e vincoli di un sistema ipermediale, deliberatamente ignorando i dettagli implementativi.

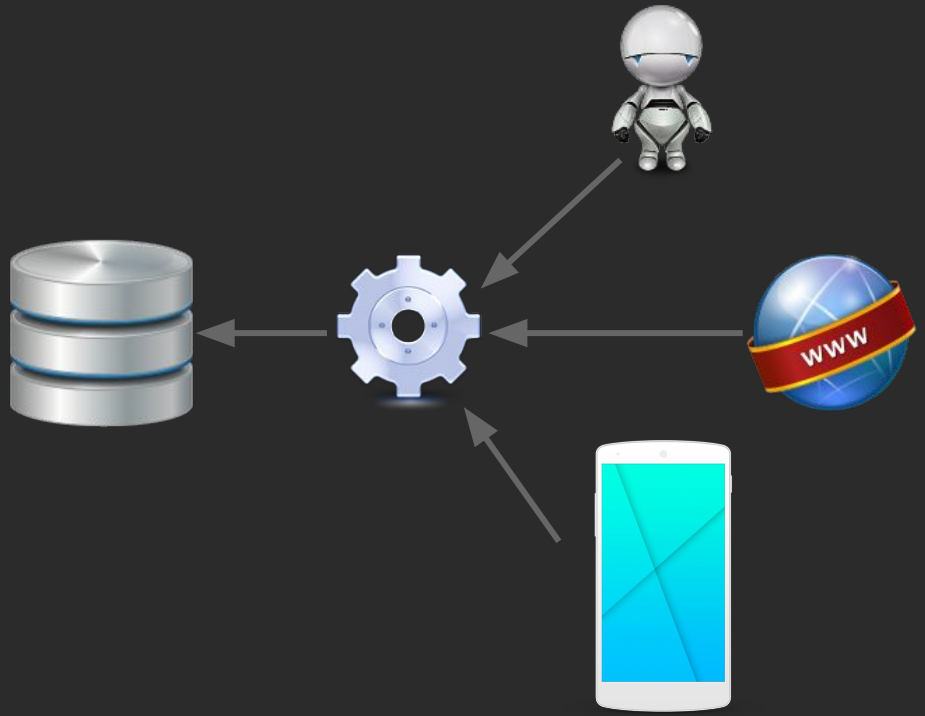
A cosa serve?

Realizzare API comprensibili



A cosa serve?

Realizzare API comprensibili



Elementi Architetturali

Resource-oriented

- Trasmette, riceve, crea e modifica “risorse”.

Elementi Architetturali (2)

Resource-oriented

Elementi Architetturali (3)

Resource-oriented

Qualunque “cosa” possa essere richiesta, manipolata o referenziata è una risorsa

- Ogni risorsa ha un identificatore (URI)
  - Più URI possono referenziare la stessa risorsa
- Privilegia i “nomi” (risorse) sui “verbi” (azioni)
  - ≠ RPC - SOAP

Elementi Architetturali (6)

Resource-oriented

- Lo scopo dei messaggi è di ottenere o modificare lo stato di una risorsa
- I messaggi non contengono **mai** la risorsa vera e propria, ma una sua rappresentazione

Vincoli di un sistema REST

Client - Server

- Due ruoli distinti
- Promuove la separazione dei compiti

Vincoli di un sistema REST (2)

Stateless

- Il server non mantiene informazioni di stato durante la comunicazione con i client
- (La sessione viene mantenuta sul client)
- Ogni messaggio contiene tutte le informazioni necessarie alla sua processazione

Vincoli di un sistema REST (3)

Cacheable

- Il server deve dichiarare la possibilità di mantenere (o meno) alcune risposte in cache per ottimizzare le prestazioni

Vincoli di un sistema REST (4)

Uniform Interface

- Contratto (Interfaccia) ben definita tra client e server
- Tutti i messaggi vengono scambiati secondo questa interfaccia
- Disaccoppia e semplifica l’architettura

Vincoli di un sistema REST (5)

Layered system

- Il sistema può essere composto di più livelli
- Ciascun livello interagisce (e conosce) solo con il livello immediatamente successivo

Vincoli di un sistema REST (6)

Code on demand (Optional)

- I server possono trasmettere logica ai client, arricchendone (temporaneamente) le capacità.

REST + HTTP

- Rappresentazioni di Risorse
  - Internet Media Type (application/JSON, text/xml)
- Resource ID
  - URI
- Messaggi
  - HTTP Request
- Metadati
  - HTTP Request Header

REST + HTTP (2)

I verbi:

HTTP Method	Utilizzo
GET	Ottiene una risorsa. **In nessun caso una richiesta GET deve modificare lo stato del sistema!**
POST	Crea una risorsa.
PUT	Sostituisce una risorsa.
PATCH	Modifica una risorsa.
DELETE	Elimina una risorsa.

Modelliamo l’API per un servizio di upload di foto

(Di gatti… Tanto sappiamo che va finire così)

Il nostro servizio

- ~~Registrazione~~
- ~~Autenticazione~~
- Caricare / modificare / cancellare foto
- Vedere le foto di un utente
- Mettere un “😻“ alle foto
- Organizzare le foto in album

La risorsa “Foto”

- ID
- Nome
- Descrizione
- Formato
- Data caricamento
- Autore

La risorsa “Utente”

- Nome
- Cognome
- Biografia
- Data Iscrizione
- Foto

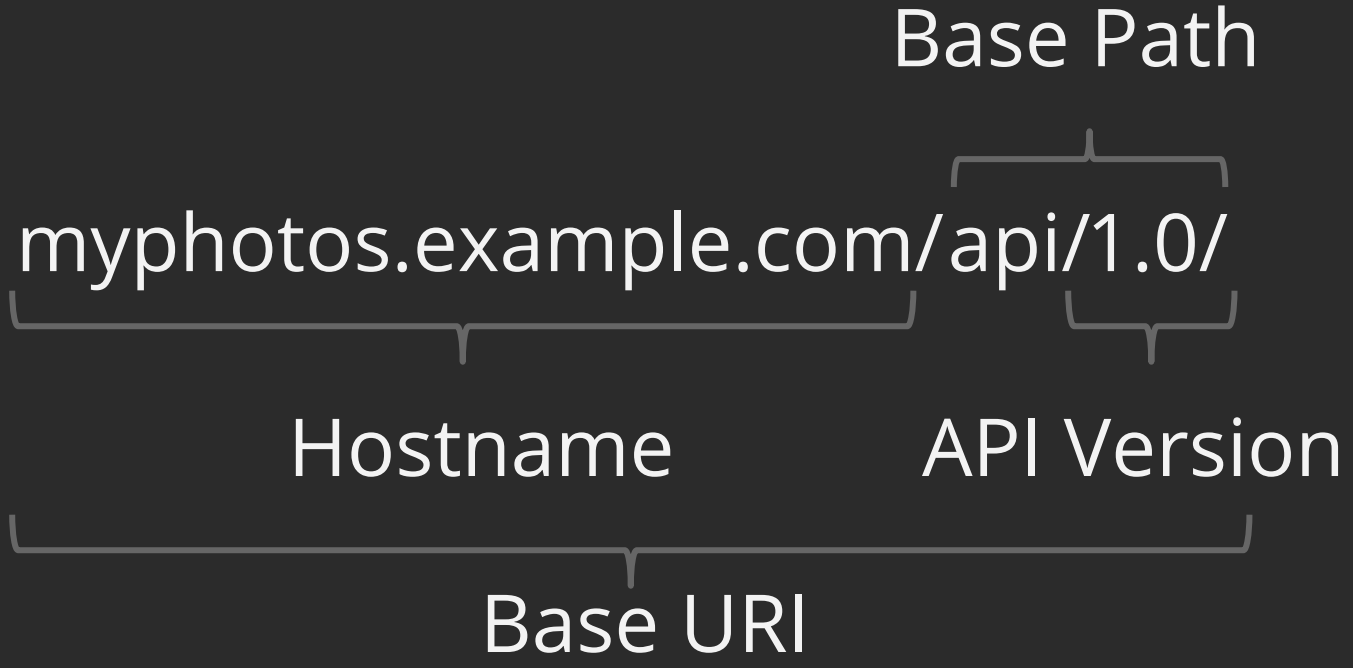
La risorsa “Album”

- Nome
- Descrizione
- Data creazione
- Foto

La risorsa “😻”

- Data creazione
- Foto
- Autore

Strutturare gli URI



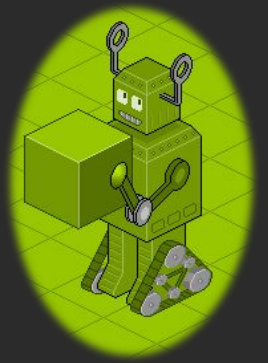
Strutturare gli URI (2)

myphotos.example.com/api/1.0/{risorsa}

Strutturare gli URI (2)

myphotos.example.com/api/1.0/{risorsa}
myphotos.example.com/api/1.0/photos
myphotos.example.com/api/1.0/users
myphotos.example.com/api/1.0/albums
myphotos.example.com/api/1.0/ 😻

Modellare l'API: Swagger



- Open API Specification
- Standard per rappresentare interfacce REST
- http://swagger.io - http://openapi.org
- Human-readable e Machine-readable

VENDINI

Creare risorse

Metodo: POST

URL: /{risorsa}

Body: Rappresentazione della risorsa da creare

Risultato: Rappresentazione della risorsa creata

Accedere Risorse Specifiche

Metodo: GET

URL: /{risorsa}/{id}

Body: N/A

Risultato: Rappresentazione della risorsa

Accedere Risorse Generiche

Metodo: GET

URL: /{risorsa}/

Body: N/A

Risultato: Una collezione di {risorsa} (possibilmente paginata)

Collezioni

Perchè paginata?

- Performance
- Praticità

Come passare da una pagina all'altra?

- GET /{risorsa}?page={numero-pagina}
- GET /{risorsa}/page/{numero-pagina}

Collezioni (2)

- Informazioni sulla collezione
- Numero di pagine totali
- Elementi per pagina
- “Dritte per cambiare pagina”
  - (Link alla pagina precedente / successiva)
- Contenuto
  - Risorse
  - Rappresentazioni “leggere”
  - Riferimenti

Modificare Risorse

Metodo: PUT

URL: /{risorsa}/{id}

Body: Rappresentazione dell’intera risorsa

Risultato: Rappresentazione dell’intera risorsa

Modificare Risorse (2)

Metodo: PATCH

URL: /{risorsa}/{id}

Body: Rappresentazione degli elementi da aggiornare

Risultato: Rappresentazione della risorsa.

Cancellare Risorse

Metodo: DELETE

URL: /{risorsa}/{id}

Body: N/A

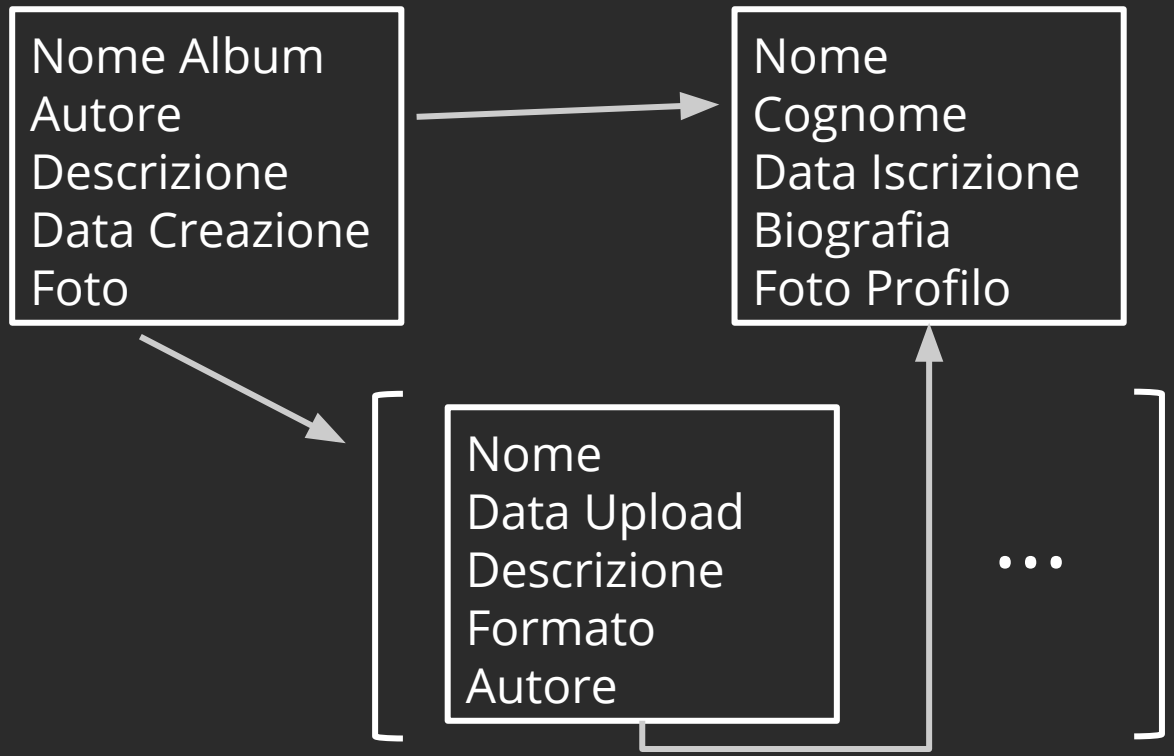
Risultato: Rappresentazione della risorsa cancellata / Vuoto

Tabella Riassuntiva

URL	Method	Significato
/{risorsa}	GET	Ottieni collezione
	POST	Crea risorsa
	PATCH	Modifica più risorse
	DELETE	Cancella più risorse
/{risorsa}/{id}	GET	Ottieni risorsa
	PUT	Sostituisci risorsa
	PATCH	Modifica risorsa
	DELETE	Elimina risorsa

Risorse Innestate

Cosa restituisce GET /albums/{id} ?



Risorse Innestate (2)

GET /albums/{id}

Nome Album
Autore
Descrizione
Data Creazione
~~Foto~~

Risorse Innestate (3)

GET /albums/{id}/photos /page/2

Risorse “strane”: 😻

Che cosa restituisce GET /meows/ ?

Che cosa restituisce GET /meows/{id} ?

… Che me ne faccio?

Risorse “strane” (2)

- Non sono utili da sole
- Hanno senso con altre risorse
  - Foto
  - Utente
- Limitare l’accesso alle risorse
  - GET /photos/{id}/meows
  - GET /users/{id}/meows

Risorse “Strane” (3)

URL	Method	Significato
/users/{id}/meows	GET	Foto che piacciono all’utente
/photos/{id}/meows	GET	Utenti a cui piace la foto
/photos/{id}/meows	POST	Aggiungi un “Mi piace” alla foto
/photos/{id}/meows	DELETE	Rimuovi “Mi piace” alla foto

Risorse “Più strane”

Nuovo vincolo:

- Gli utenti possono richiedere di pubblicare un proprio Album in home page
- 3 giurati devono valutare gli album prima che possano andare in home page

Risorse “Più strane” (2)

Nome Album
Autore
Descrizione
Data Creazione
~~Richiesta Inviata~~

- Non permette di inviare più richieste per lo stesso album
- Il fatto che sia stata inviata una richiesta non è una proprietà dell’album
- Richiede che il client conosca la logica del sistema

Risorse “Più strane” (3)

POST /albums/{id}/promote

- Promote non è una risorsa (è un verbo)!

Risorse “Più strane” (4)

POST /front-page/

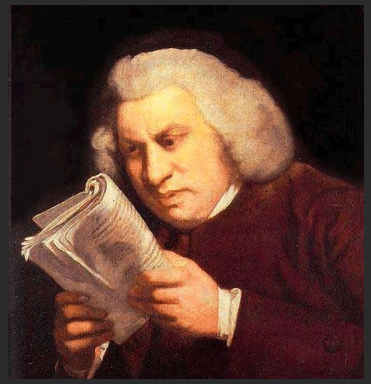
Body: Oggetto Album da promuovere.

- Interazione RESTful ✔
- Non posso ottenere informazioni sullo stato dell’operazione

Risorse “Più strane” (5)

La soluzione “più RESTful” richiede di *reificare* il processo di promozione di un album.

> “La reificazione è una rappresentazione indiretta che prevede l’utilizzo di determinate espressioni del linguaggio per descrivere entità del mondo reale intuitivamente associate ad espressioni di tipo diverso.”



Risorse “Più strane” (6)

Reificare significa ~ “rendere manipolabile un concetto astratto”.

Trasformare l’azione in una risorsa.

URL	Method	Significato
/promotion-requests/	POST	Invia una richiesta di promozione
/promotion-requests/	GET	Lista delle richieste di promozione
/promotion-requests/{id}	GET	Dettagli sulla richiesta di promozione
/promotion-requests/{id}	PATCH	Aggiorna stato della richiesta di promozione

Per saperne di più

- http://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf
- http://www.restapitutorial.com/
- https://github.com/OAI/OpenAPI-Specification/blob/master/versions/1.2.md
- http://rest.elkstein.org/2008/02/what-is-rest.html

Domande?