URI: Uniform Resource Identifier

Angelo Di Iorio
Università di Bologna

(dal materiale del Prof. Fabio Vitali)

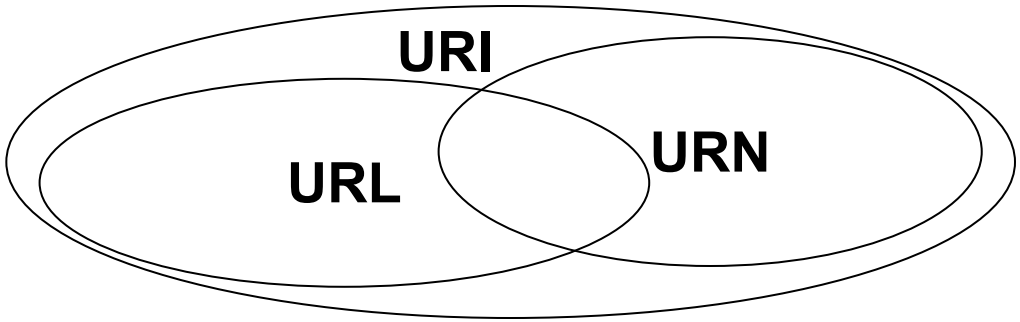
Uniform Resource Identifier (URI)

- Gli URI (Uniform Resource Identifier) sono una sintassi usata in WWW per definire i nomi e gli indirizzi delle risorse
- Sono stati verosimilmente il fattore determinante per il successo del WWW
- Attraverso gli URI, il WWW è stato in grado di identificare risorse accessibili tramite il proprio protocollo, HTTP, e tramite tutti gli altri protocolli esistenti (FTP, Telnet, ecc.).
- Il punto principale a cui gli altri sistemi non erano arrivati era una **sintassi universale**, **indipendente dal protocollo** e **facilmente memorizzabile** (o quasi) con cui identificare le risorse di rete.

URI, URL e URN

Gli **Uniform Resource Identifier (URI)** sono, per definizione:

- **Uniform Resource Locator (URL)**: una sintassi che contiene informazioni immediatamente utilizzabili per accedere alla risorsa (ad esempio, il suo indirizzo di rete)
- **Uniform Resource Names (URN)**: una sintassi che permette una etichettatura permanente e non ripudiabile della risorsa, indipendentemente dal riportare informazioni sull'accesso. Necessario quindi un meccanismo di traduzione verso gli URL



Risorsa vs. File

- Una risorsa non è necessariamente un file presente su un filesystem ma potrebbe essere:
  - in un database, e l'URI essere la chiave di ricerca
  - il risultato dell'elaborazione di un'applicazione, e l'URI essere i parametri di elaborazione.
  - una risorsa non elettronica (un libro, una persona, un pezzo di produzione industriale) e l'URI essere il suo nome, ad esempio nel caso di account Twitter, Instagram, etc.
  - un concetto astratto
- Per questo si usa il termine Risorsa, invece che File, e si fornisce una sintassi indipendente dal sistema effettivo di memorizzazione.

Organizzazione degli URI

Gli URI sono progettati per fornire **spazi di nomi organizzati gerarchicamente**:

URI = *schema* : [// *authority*] *path* [? *query*] [# *fragment*]

- Esempi:
  - http://www.ietf.org/rfc/rfc2396.txt
  - ftp://ftp.is.co.za/rfc/rfc1808.txt
  - https://purl.oclc.org/OCLC/PURL/FAQ
  - file:///Documenti/corsi/tw/slides/l1.html
  - mailto: angelo.diiorio@unibo.it
  -

Componenti degli URI (1)

schema **:** **[//** ***authority*****]** *path* [? *query*] [# *fragment*]

- Lo ***schema*** (negli URL è il protocollo) é identificato da una stringa registrata presso IANA usata come prefisso.
- L'autorità è a sua volta divisa in:
  - `authority = [userinfo @] host [: port]`
- La parte ***userinfo*** non deve essere presente se lo schema non prevede identificazione personale.
- La parte ***host*** è o un nome di dominio o un indirizzo IP. La ***port*** può essere omessa se ci si riferisce ad una well-known port (per http è la porta 80).

Componenti degli URI (2)

schema : [// *authority*] ***path*** [? *query*] [# *fragment*]

- La parte ***path*** è la parte identificativa della risorsa all'interno dello spazio di nomi identificato dallo schema e (se esistente) dalla authority.
- La parte path è divisa in blocchi separati da slash "/", ciascuno dei quali è un componente del path organizzato in gerarchia.
- In questo caso diventano significativi gli pseudo componenti "." e "..".

Componenti degli URI (3)

schema : [// *authority*] *path* **[? *query*]** [# *fragment*]

- La parte ***query*** individua un'ulteriore specificazione della risorsa all'interno dello spazio di nomi identificato dallo schema e dall'URI precedente.
- Di solito questi sono parametri passati all'URI (un processo) per specificare un risultato dinamico (es. l'output di una query su un motore di ricerca).
- Tipicamente ha la forma *nome1=valore1&nome2=valore+in+molte+parole*

Componenti degli URI (4)

schema : [// *authority*] *path* [? *query*] **[# *fragment*]**

- La parte ***fragment*** individua una risorsa secondaria (una risorsa associata, dipendente o in molti casi un frammento) della risorsa primaria
- E' tutta la parte che sta dopo al carattere di hash "#".
- Usata ad esempio per identificare sezioni all'interno di una pagina HTML

Alcuni schemi usati negli URI

- HTTP è lo schema più usato negli URI
- HTTPS prevede la cifratura dei messaggi, in entrambi i versi. Per il resto è identico ad HTTP:
  - `http[s] ://host[:port]/path[?query][#fragment]`
- Alcune note:
  - **host** é l'indirizzo TCP-IP o DNS della macchina su cui si trova la risorsa
  - **port** è la porta a cui il server é in ascolto per le connessioni. Per default, la porta è 80 per HTTP e 443 per HTTPS.

Lo schema FILE (RFC 8089)

- Dà accesso ai file di un file system locale (cioè del computer su cui gira il browser)
- Non girano applicazioni server-side, nessuna connessione HTTP. La sintassi è:

  file://<host>/<path> [#fragment]
- La parte host può essere eliminata, assumendo che sia localhost, che porta alla sintassi più frequente:

  file://localhost/path

  file:///c:/Users/mario/Pictures/img1.jpg
- MS Windows accetta anche "\", che però non è nello standard approvato!

Lo schema DATA (RFC 2397)

- Uno schema **non gerarchico**, che non fa riferimento ad una risorsa, ma CONTIENE la risorsa: tutti i dati della risorsa sono inseriti nell'URI vero e proprio.
- Usato per immagini inline su cui non si vuole attivare una connessione HTTP separata. La sintassi è:

```
data:[<media type>][;base64],<data>
```

- ***media type*** è un media type MIME (vedi lezione su MIME e base64)
- **base64: è un parametro opzionale per indicare che il dato è codificato in base 64**
- ***data*** sono i dati codificati con il media-type appena indicato

Esempi

Testo, codificato in UTF-8, da notare l'escaping del carattere 'spazio'

```
data:text/plain;charset=UTF-8;
some%20text%20for%20you
```

Immagine PNG, la sequenza di bit è codificata in base64

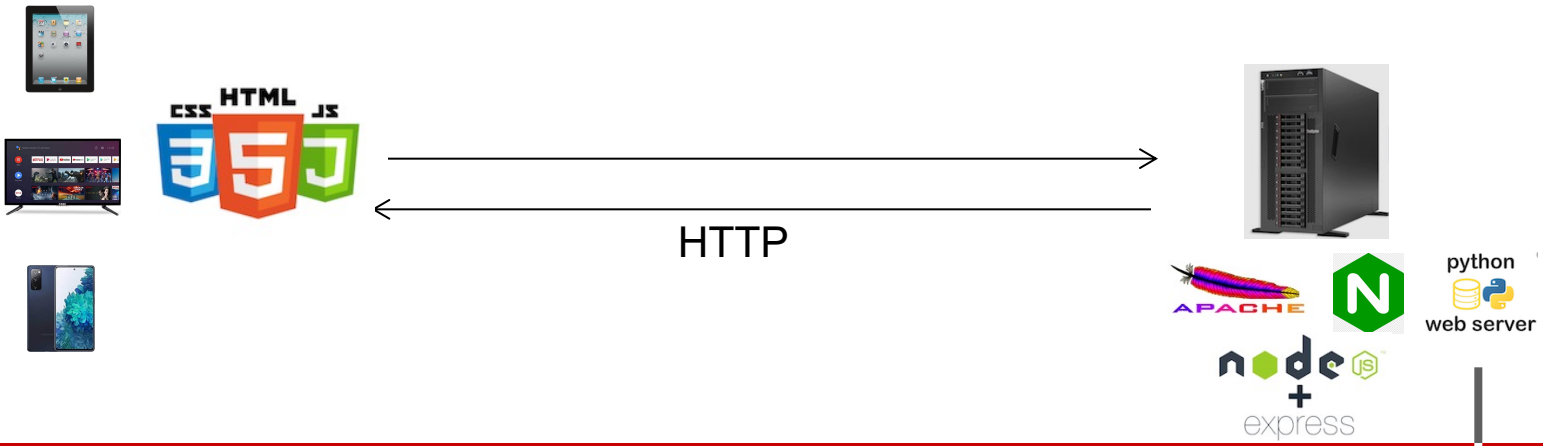
```

iVBORw0KGgoAAAANSUhEUgAAAAUAAAAFCAYAAACNbyblAAAAHElE
QVQI12P4//8/w38GIAXDIBKE0DHxgljN
BAAO9TXL0Y4OHwAAAABJRU 5ErkJggg==
```

Routing e URL assoluti e relativi

Un passo indietro: server Web

- Un Server Web è un software in grado di gestire richieste HTTP
- E' in ascolto su una porta TCP/IP (di default 80)
- La richiesta indica una risorsa tramite un URL e usa quindi lo schema HTTP o HTTPS



Route

- Una route è un'associazione della parte *path* di un URI ad una risorsa gestita o restituita da un server web
- ***Managed route***: il server associa ogni URI ad una risorsa o attraverso il file system locale (*risorse statiche*) oppure generate attraverso una computazione (*risorse* dinamiche).
    - Molto di moda oggi con node.js e express.js
- ***File-system route***: il server associa la radice della parte *path* ad una directory del file system locale e ogni filename valido all'interno di quella directory genera un URI corretto e funzionante.
    - Il vecchio approccio via web server come Apache

Una *managed route*

```
…
var router = require("express").Router();
function getName(req, res) { res.send("<p>Bob</p>"); }

function getEmail(req, res) {res.send("bob@unibo.it");}
router.get("/name", getName);
router.get("/email", getEmail);
app.use("/css", express.static('css'));
```

URI della richiesta	Risorsa restituita
http://www.example.com/name	Bob
http://www.example.com/email	bob@unibo.it
http://www.example.com/css/style.css	*contenuto del file* css/style.css

Organizzazione del file system	URI disponibili
/	-
var/	-
www/	-
index.html	http://www.example.com/
alice/	-
index.html	http://www.example.com/alice/
bruce/	-
index.html	http://www.example.com/bruce/
...	...
fabio/	-
index.html	http://www.example.com/fabio/
pages/	
doc1.html	http://www.example.com/fabio/pages/doc1.html
doc2.html	http://www.example.com/fabio/pages/doc2.html
index.html	http://www.example.com/fabio/pages/
img/	
img1.gif	http://www.example.com/fabio/img/img1.gif
img2.jpg	http://www.example.com/fabio/img/img1.gif
css/	
style.css	http://www.example.com/fabio/css/style.css

URI reference (o URI relativo)

- Un **URI assoluto** contiene tutte le parti predefinite dal suo schema, esplicitamente precisate.
- Un **URI** gerarchico può però anche essere **relativo**, (detto tecnicamente un **URI reference**) ed in questo caso riportare solo una parte dell'URI assoluto corrispondente "tagliando progressivamente parti da sinistra"
- Un **URI reference** fa sempre riferimento ad un **URI di base** (ad esempio, l'URI assoluto del documento ospitante l'URI reference) rispetto al quale fornisce porzioni differenti.
  - Es.: l'URL reference pippo.html posto dentro al documento di URI http://www.sito.com/dir1/dir2/pluto.html
    fa riferimento al documento il cui URI assoluto è
    http://www.sito.com/dir1/dir2/pippo.html

Risolvere un URI relativo

Risolvere un URI relativo significa identificare l'URI assoluto cercato sulla base dell'URI

	Dato il base URI http://www.site.com/dir1/doc1.html
Se inizia con "#", è un frammento interno allo stesso documento di base	#anchor1 si risolve come http://www.site.com/dir1/doc1.html#anchor1
Se inizia con uno schema, è un URI assoluto	http://www.site.com/dir2/doc2.html si risolve come http://www.site.com/dir2/doc2.html
Se inizia con "/", allora è un path assoluto all'interno della stessa autorità del documento di base, e gli va applicata la stessa parte autorità	/dir3/doc3.html si risolve come http://www.site.com/dir3/doc3.html

Risolvere un URI relativo

Altrimenti:	Dato il base URI http://www.site.com/dir1/doc1.html
Altrimenti, si estrae il path assoluto dell'URI di base, meno l'ultimo elemento, e si aggiunge in fondo l'URI relativo.	doc4.html si risolve come http://www.site.com/dir1/doc4.html dir5/doc5.html si risolve come http://www.site.com/dir1/dir5/doc5.html
Si procede infine a semplificazioni:	
"./" (stesso livello di gerarchia): viene cancellata	./doc6.html si risolve come http://www.site.com/dir1/./doc6.html che è equivalente a http://www.site.com/dir1/doc6.html
"../" (livello superiore di gerarchia): viene eliminato insieme all'elemento precedente.	../doc7.html si risolve come http://www.site.com/dir1/../doc7.html che è equivalente a http://www.site.com/doc7.html

Esercizio

Risolvere i seguenti URI relativi rispetto al seguente URI base:
http://www.sito.com/data/2050/index.html

a) **images/**

b) **/images/**

c) **../2020/images/img2.jpg**

d) **../../images/img1.png**

e) **./videos/**

f) **../videos/**

g) **/videos/images/**

h) **../index.html#y2021**