

# Web Security 1

## Il Browser Web e HTTP

**Riccardo  
BONAFEDE**

Università degli Studi di  
Padova



<https://cybersecnatlab.it>



# Obiettivi

3

- Illustrare la struttura di una richiesta HTTP
- Capire come usare la console di debug di un browser

# Prerequisiti

4

- Modulo **NS\_1.1 - Fondamenti di reti di calcolatori**
- Conoscenza base in almeno un linguaggio di programmazione





# World Wide Web

7

- Il **World Wide Web** (WWW) è un servizio di internet che permette lo scambio di informazioni
- Nasce all'inizio degli anni 90, quando Tim Berners-Lee pubblica il primo sito web
- È composto da *siti web*, che sono contenitori di risorse che vengono rese pubbliche a utenti

# World Wide Web

8

- Il web è basato su una architettura **client-server**
- Ogni utente usa un **client** per accedere a una *risorsa* su un **server** attraverso la rete
  - Il client web è chiamato browser web
  - Il server è chiamato server web





# URLs

9

- Ogni risorsa nel web è identificata da un indirizzo, chiamato URL<sup>1</sup>
- Un URL è composto da varie parti, ognuna delle quali ha un preciso scopo

`https://www.example.com/index.html`

1: Uniform Resource Locator

# URLs

10

- Ogni risorsa nel web è identificata da un indirizzo, chiamato URL<sup>1</sup>
- Un URL è composto da varie parti, ognuna delle quali ha un preciso scopo

`https://www.example.com/index.html`

Protocollo: Può essere HTTP O HTTPS. Quest'ultimo è la versione crittografata di HTTP

1: Uniform Resource Locator

# URLs

11

- Ogni risorsa nel web è identificata da un indirizzo, chiamato URL<sup>1</sup>
- Un URL è composto da varie parti, ognuna delle quali ha un preciso scopo

`https://www.example.com/index.html`

Hostname: L'indirizzo del server a cui connettersi. Vedi modulo network

1: Uniform Resource Locator

# URLs

12

- Ogni risorsa nel web è identificata da un indirizzo, chiamato URL<sup>1</sup>
- Un URL è composto da varie parti, ognuna delle quali ha un preciso scopo

`https://www.example.com/index.html`

Path: La pagina da richiedere. Può contenere anche una directory

1: Uniform Resource Locator



# Protocollo HTTP

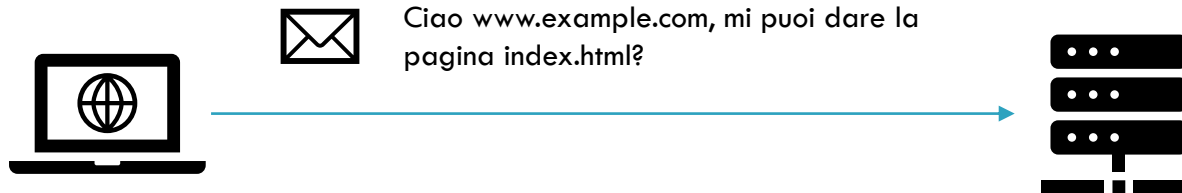
14

- Client e server scambiano informazioni usando un **protocollo**, chiamato **HTTP**
- La comunicazione viene sempre avviata dal client, che manderà richieste a un server

# Protocollo HTTP

15

- Client e server scambiano informazioni usando un **protocollo**, chiamato **HTTP**
- La comunicazione viene sempre avviata dal client, che manderà richieste a un server



# Protocollo HTTP

16

- Client e server scambiano informazioni usando un **protocollo**, chiamato **HTTP**
- La comunicazione viene sempre avviata dal client, che manderà richieste a un server
- Il server interpreta queste richieste e risponde al client





# Protocollo HTTP

17

## Richiesta

```
GET /index.html HTTP/1.1
Host: www.example.com
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

## Risposta

```
HTTP/1.1 200 OK
Content-Length: 88
Content-Type: text/html
Connection: Closed

<html>
...
```

# Protocollo HTTP

18

Metodo, path, versione

```
GET /index.html HTTP/1.1
```

```
Host: www.example.com
```

```
Accept-Language: en-us
```

```
Accept-Encoding: gzip, deflate
```

```
Connection: Keep-Alive
```

Versione, status code, status message

```
HTTP/1.1 200 OK
```

```
Content-Length: 88
```

```
Content-Type: text/html
```

```
Connection: Closed
```

```
<html>
```

```
...
```

# Protocollo HTTP

19

## Headers

```
GET /index.html HTTP/1.1  
Host: www.example.com  
Accept-Language: en-us  
Accept-Encoding: gzip, deflate  
Connection: Keep-Alive
```

## Headers

```
HTTP/1.1 200 OK  
Content-Length: 88  
Content-Type: text/html  
Connection: Closed
```

```
<html>
```

```
...
```

# Protocollo HTTP

20

Body (qui non presente)

```
GET /index.html HTTP/1.1
Host: www.example.com
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

Body

```
HTTP/1.1 200 OK
Content-Length: 88
Content-Type: text/html
Connection: Closed
```

```
<html>
...
```

# Argomenti

21

- Introduzione a HTTP
  - URLs
  - Overview del protocollo
- **Il browser web**
  - **Struttura delle pagine**
  - Strumenti di sviluppo

# Pagine WEB

22

- Il body della risposta del server viene interpretato dal browser, a seconda del tipo di contenuto
- Le pagine che si visualizzano navigando la rete, sono renderizzate interpretando un linguaggio di markup, chiamato **HTML**

# HTML

23

- HTML funziona a blocchi, chiamati **tag**
- Ogni tag ha degli **attributi**, che ne vanno a modificare il comportamento
- Ogni tag può contenere altri tag

# HTML

24

```
<html>
  <head>
    <title>Esempio</title>
  </head>
  <body>
    <h1>Hello World!</h1>
    <span style="color:red">Testo rosso</span>
  </body>
</html>
```

**Hello World!**

Testo rosso



# HTML

25

```
<html>
  <head>
    <title>Esempio</title>
  </head>
  <body>
    <h1>Hello World!</h1>
    <span style="color:red">Testo rosso</span>
  </body>
</html>
```

**Hello World!**

Testo rosso

# HTML

26

- Le pagine possono includere altri tipi di risorse:
  - Immagini
  - Video
  - Audio
  - Fogli di stile (CSS)
  - **Script**
- Gli script sono dei veri e propri programmi, che agiscono sulle pagine e consentono di renderle più interattive

# Javascript

27

- Gli script nella pagina web vengono inclusi utilizzando il tag `<script>`
- Sono scritti in Javascript, un linguaggio inizialmente nato proprio per essere incluso in pagine web (e ora utilizzato anche in parecchie altre applicazioni)

```
<!-- Inclusione diretta -->
<script>
  alert('Hello World');
</script>

<!-- Inclusione come file esterno -->
<script src="/app.js"></script>
```

# Argomenti

28

- Introduzione a HTTP
  - URLs
  - Overview del protocollo
- Il browser web
  - Struttura delle pagine
  - **Strumenti di sviluppo**

# Console del browser

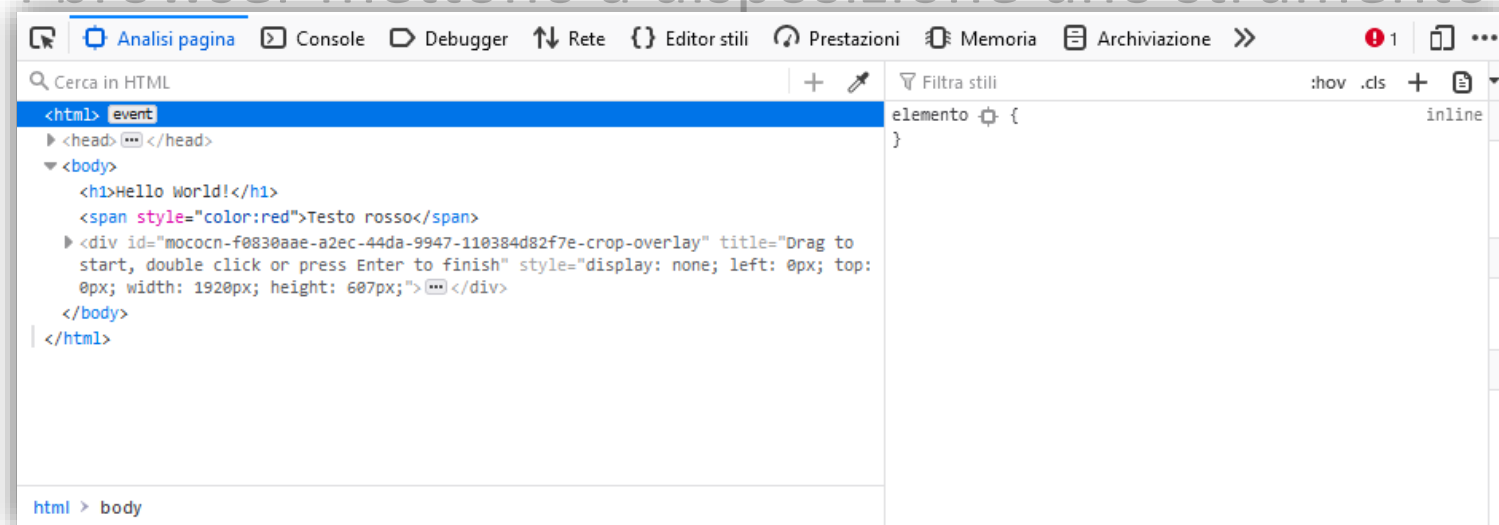
29

- I browser mettono a disposizione uno strumento per analizzare pagine web, detto "*browser console*"
- Per aprire questa console, generalmente basta premere il tasto F12

# Console del browser

30

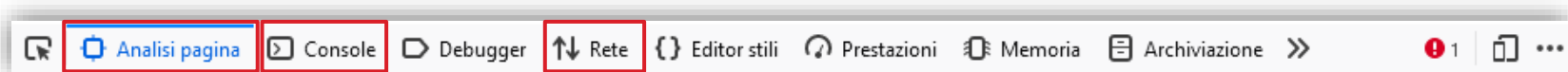
- I browser mettono a disposizione uno strumento



# Console del browser

31

- Le tab a cui dare attenzione sono:
  - **Analisi pagina**: permette di interagire con l'HTML
  - **Console**: permette di interagire con Javascript
  - **Rete**: permette di esaminare le richieste HTTP



Nota bene: su browser diversi da Firefox i nomi possono variare

# Console del browser – Console Javascript

32

- Come anticipato, la console Javascript permette di interagire con codice Javascript
  - Eseguire codice
  - Leggere il contenuto di variabili
- Per sperimentare, basta creare una pagina html
  1. Crea un nuovo file
  2. Rinomina in *test.html*
  3. Aggiungi del codice usando un editor di testo<sup>1</sup>
  4. Aprire il file con un browser

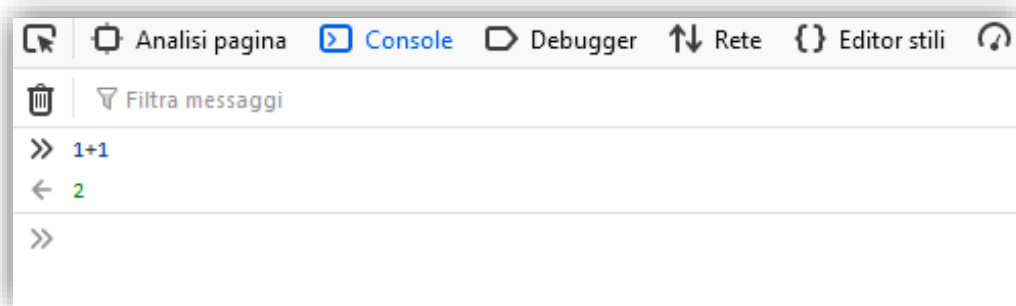
1: Visual Studio Code per esempio, o il blocco note di windows



# Console del browser – Console Javascript

33

- Aperta la pagina, bisogna ora aprire la console:
  - *Tasto F12 → console*
- Dopo i due >, è possibile scrivere codice Javascript
- Ad esempio, scrivendo *1+1* la console risponderà con 2



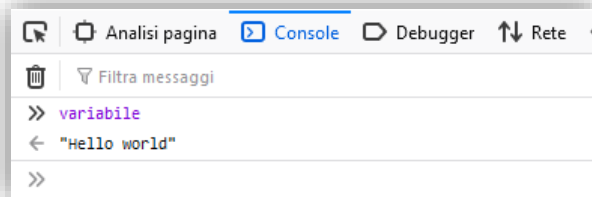
# Console del browser – Console Javascript

34

- È possibile interagire con il codice Javascript presente nella pagina.
- Ad esempio, se si inserisce il seguente codice nella pagina *test.html*

```
<script>  
var variabile = "Hello world";  
</script>
```

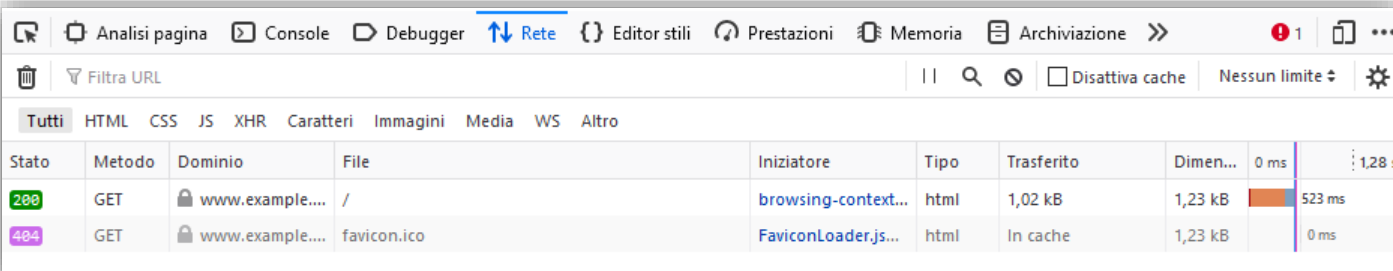
- È possibile recuperare il valore della variabile "variabile" dalla console



# Console del browser - Rete

35

- La tabella Rete della console permette di visualizzare le richieste eseguite da una pagina

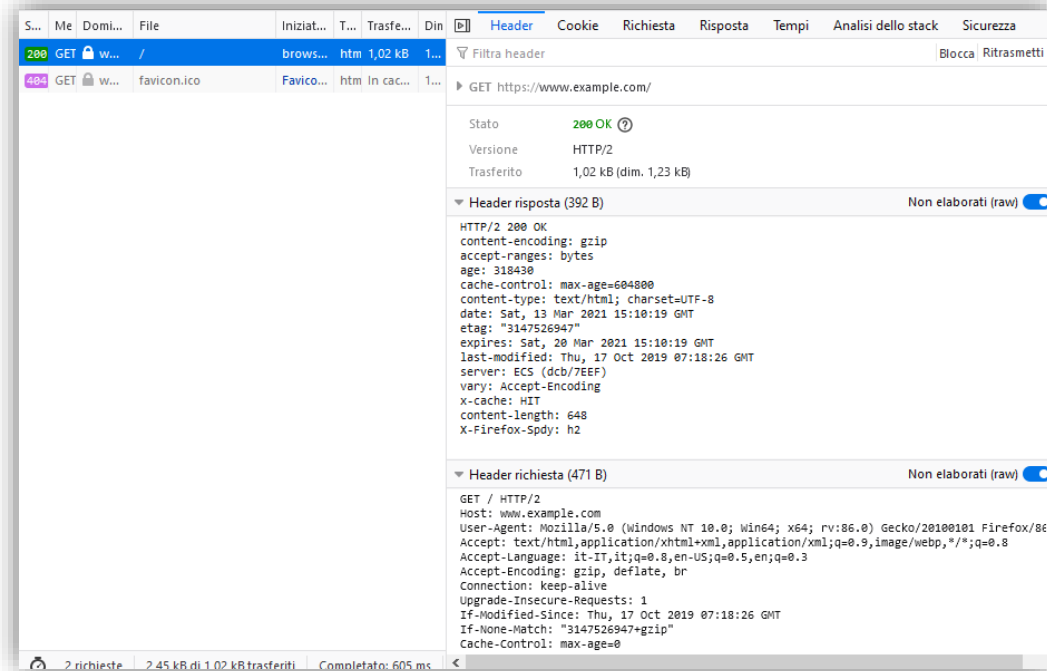


The screenshot shows the Network tab of a browser's developer console. The table displays the following data:

Stato	Metodo	Dominio	File	Iniziatore	Tipo	Trasferito	Dimen...	0 ms	1,28 s
200	GET	www.example....	/	browsing-context...	html	1,02 kB	1,23 kB	523 ms	
404	GET	www.example....	favicon.ico	FaviconLoader.js...	html	In cache	1,23 kB	0 ms	

# Console del browser - Rete

36



Risposta grezza

Richiesta grezza

# Web Security 1

## Il Browser Web e HTTP

**Riccardo  
BONAFEDE**

Università degli Studi di  
Padova



<https://cybersecnatlab.it>