

ANATOMIA DI UNA WEB APP

Parte 1 - Web Servers

LEONARDO TAMIANO

TABLE OF CONTENTS

- Overview Architettuale
- Web Server
 - Protocollo HTTP
 - Il linguaggio HTML
 - Uniform Resource Locators (URLs)
- Apache2
- Il limite delle pagine statiche

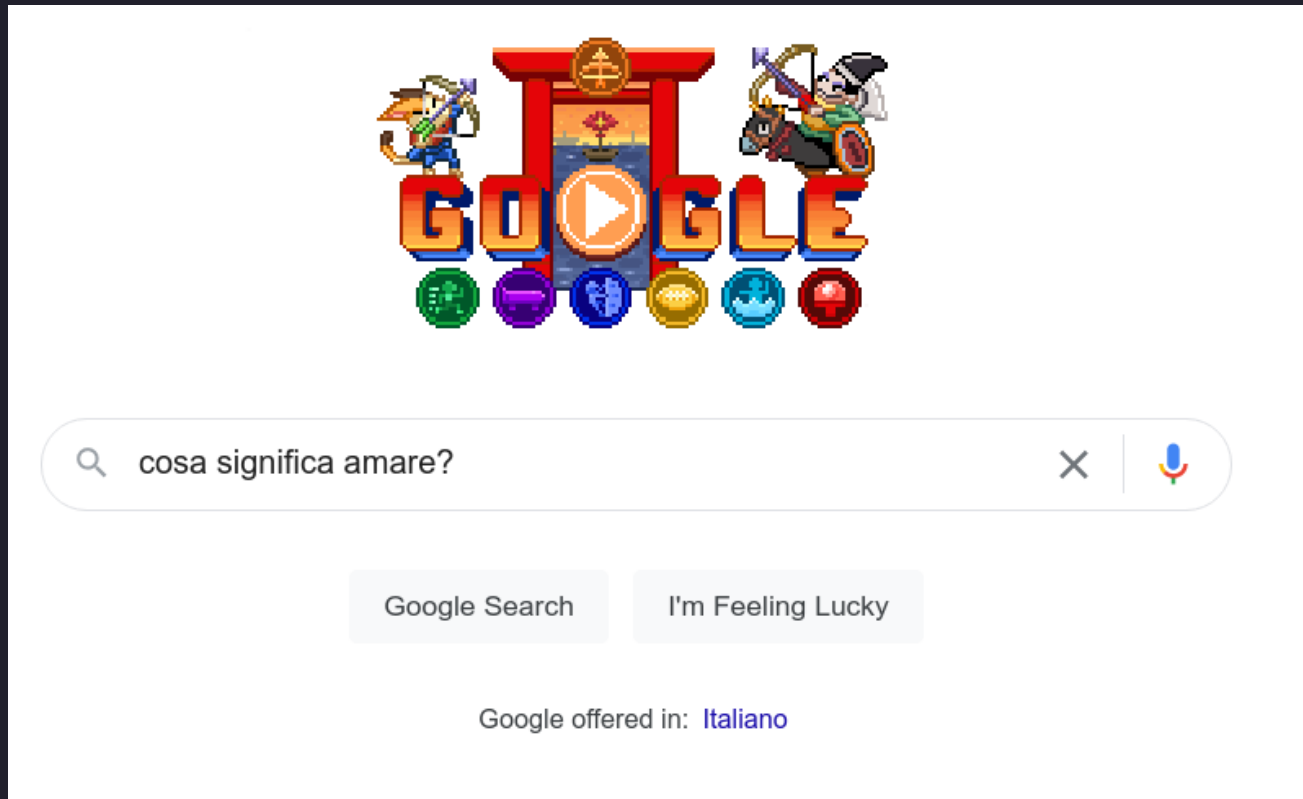
OVERVIEW ARCHITETTURALE

Tra tutti i servizi introdotti dall'informatica, il **Web** è senza dubbio quello che singolarmente ha avuto il maggior impatto nel mondo moderno.

Oramai è possibile accedere a quasi tutte le tipologie di informazioni tramite un **browser**.



I **motori di ricerca** sono diventati lo strumento principale utilizzato per soddisfare, nel bene e nel male, ogni nostro dubbio o curiosità.



La **programmazione web** è il contesto lavorativo più popolare e più ricompensato tra tutti i possibili lavori che hanno a che fare con l'informatica.

Ma quali sono i componenti principali di una
applicazione web?

Sono tanti i layers tecnologici associati ad una
applicazione web dinamica.

Alcuni sono associati al **server:**

Sono tanti i layers tecnologici associati ad una
applicazione web dinamica.

Alcuni sono associati al **server:**

- Un **web server**, utilizzato per scambiare files, principalmente pagine **html**, tra il server e i clients tramite il protocollo **http**.
applicazione web dinamica.
-

Alcuni sono associati al **server**:

- Un **web server**, utilizzato per scambiare files, principalmente pagine **html**, tra il server e i clients tramite il protocollo **http**.
applicazione web dinamica.
- Un **backend engine**, utilizzato per generare le pagine **html** e inviare al **server** ai vari clients.

- Un **web server**, utilizzato per scambiare files, principalmente pagine **html**, tra il server e i clients tramite il protocollo **http**.

- Sono tanti i layers tecnologici associati ad una **applicazione web dinamica**.
Un **backend engine**, utilizzato per generare le pagine **html dinamiche** inviate dal web server ai
-

vari clients.

Alcuni sono associati al **client**:

- Un **database**, utilizzato per memorizzare in modo persistente le informazioni critiche fornite dall'applicazione.

Sono tanti i layers tecnologici associati ad una
applicazione web dinamica.

Alcuni sono associati al **client**:

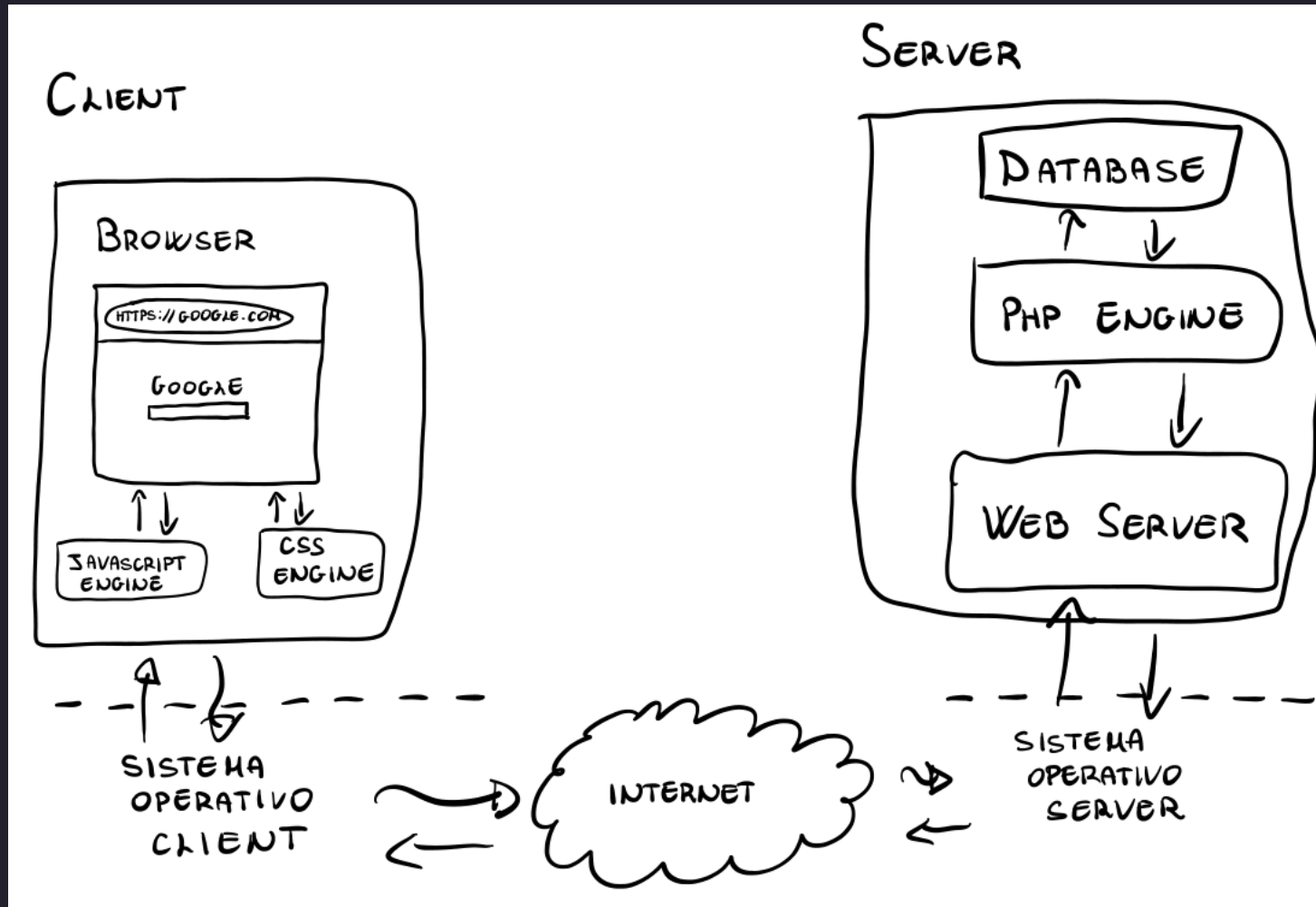
- Un **browser**, utilizzato per visualizzare le pagine **html** ritornate dal server.

Sono tanti i layers tecnologici associati ad una **applicazione web dinamica**.

Alcuni sono associati al **client**:

- Un **browser**, utilizzato per visualizzare le pagine **html** ritornate dal server.
- Un **frontend engine**, contenuto nel browser e utilizzato per rendere la pagina html dinamica agli occhi dell'utente.

Graficamente,



Cerchiamo adesso di capire in più dettaglio il ruolo di ciascuno di questi componenti, partendo in particolare dai **web servers**.

Tutti i comandi mostrati fanno riferimento alla seguente versione di **ubuntu**.

```
leo@leoPC:~/Desktop$ neofetch
```

```
  .-/+oosssso+/-.
  `:+ssssssssssss++:`
    -+ssssssssssssyyssss+-
    .ossssssssssssssdMMMNysssso.
    /ssssssssssshdmmNNmyNMMMMhssssss/
    +ssssssssshmydMMMMMMNdddyssssss++
    /ssssssssshNMMMyhhyyyyhNMMMNhssssss/
    .ssssssssdMMNhsSSssssssshNMMMdSSssssss.
    +ssshhhhyNMMNySSsssssssssyNMMMySSssss++
    ossyNMMMNyMMhSSssssssssshmmhSSssssso
    ossyNMMMNyMMhSSssssssssshmmhSSssssso
    +ssshhhhyNMMNySSsssssssssyNMMMySSssss++
    .ssssssssdMMNhsSSssssssshNMMMdSSssssss.
    /ssssssssshNMMMyhhyyyyhNMMMNhssssss/
    +ssssssssdmydMMMMMMNdddyssssss++
    /ssssssssshdmmNNmyNMMMMhssssss/
    .ossssssssssssssdMMMNysssso.
    -+ssssssssssssyyssss+-
    `:+ssssssssssss++:`
    .-/+oosssso+/-.

```

```
leo@leoPC
```

```
-----
OS: Ubuntu 20.04 LTS x86_64
Host: KVM/QEMU (Standard PC (Q35 + I
Kernel: 5.11.0-27-generic
Uptime: 8 mins
Packages: 1440 (dpkg), 6 (snap)
Shell: bash 5.0.16
Resolution: 918x964
DE: GNOME
WM: Mutter
WM Theme: Adwaita
Theme: Yaru [GTK2/3]
Icons: Yaru [GTK2/3]
Terminal: gnome-terminal
CPU: Intel Xeon E312xx (Sandy Bridge
GPU: 00:01.0 Red Hat, Inc. QXL parav
Memory: 701MiB / 1985MiB

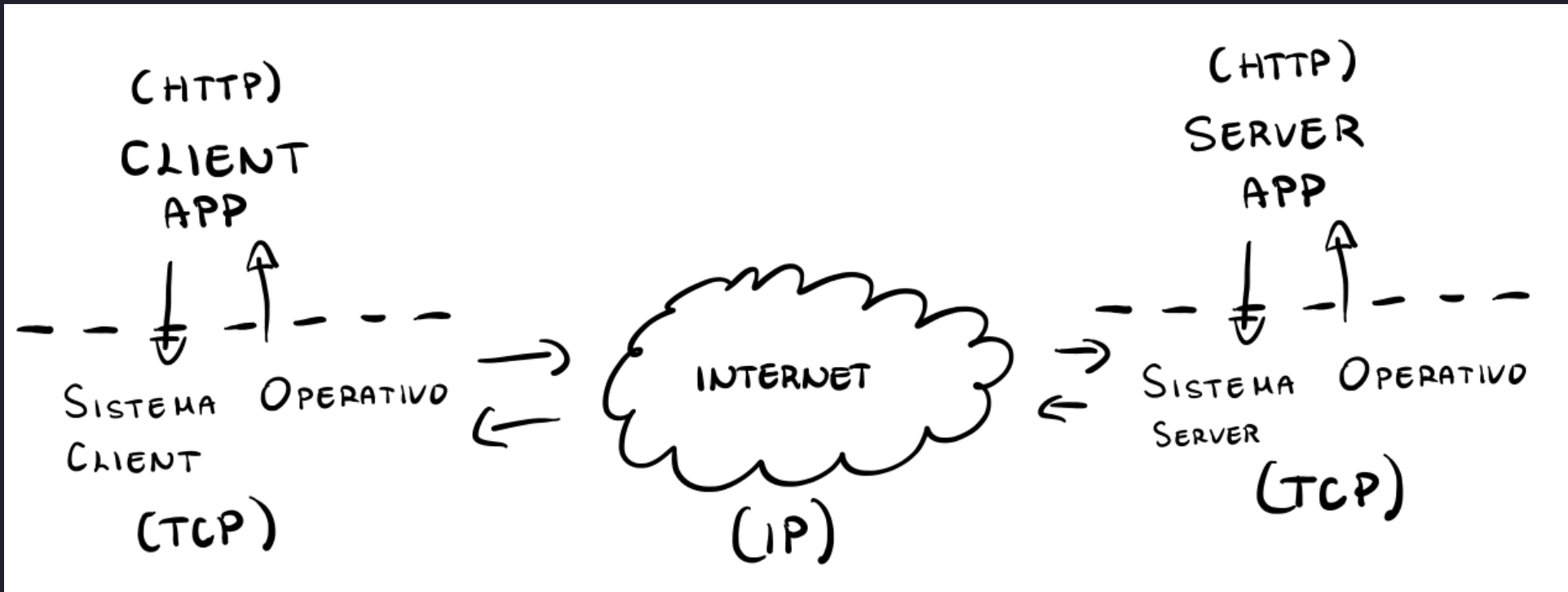
```



WEB SERVER

Il compito principale di un **web server** consiste nel mettere a disposizione delle **risorse** a tutti i **clients** in giro per il mondo.

Il mezzo sottostante che si utilizza per mettere in contatto i server web e i clients web è la rete **internet**.



Per poter comunicare però server e client necessitano di parlare lo stesso **linguaggio**.

PROTOCOLLO HTTP

Il linguaggio principale utilizzato nel web è il protocollo **HTTP**, abbreviazione di "Hypertext Transfer Protocol".

Il protocollo HTTP definisce

1. Il formato dei messaggi che client e server si possono inviare per comunicare.
2. Come questi messaggi devono essere interpretati da client e server.

I messaggi definiti dal protocollo HTTP sono semplici da leggere, in quanto non contengono dati in binario, ma sono scritti in **plaintext**, ovvero utilizzando parole riprese dal linguaggio comune.

Utilizzando il comando **curl** con la flag **--verbose** è possibile sia effettuare delle richieste HTTP, che vedere quali sono i formati delle richieste e delle risposte.

Ad esempio,

```
curl --verbose https://leonardotamiano.xyz > /dev/null
```

osserviamo la seguente richiesta,

```
GET /index.html HTTP/1.1  
Host: leonardotamiano.xyz  
User-Agent: curl/7.77.0  
Accept: */*
```

a cui il server risponde,

```
HTTP/1.1 200 OK
```

```
Server: nginx
```

```
Date: Thu, 26 Aug 2021 22:11:47 GMT
```

```
Content-Type: text/html; charset=utf-8
```

```
Content-Length: 28869
```

```
Last-Modified: Wed, 25 Aug 2021 20:55:33 GMT
```

```
Connection: keep-alive
```

```
ETag: "6126ae45-70c5"
```

```
Accept-Ranges: bytes
```

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
    <meta name="generator" content="Hugo 0.83.1" />
```

```
    <title>Leonardo Tamiano's Cyberspace</title>
```

```
...
```

```
</html>
```

È possibile leggere tutti i dettagli relativi alle varie versioni del protocollo HTTP nei famosi documenti noti con il nome **Request For Comments** (RFC).

Hypertext Transfer Protocol -- HTTP/1.1

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

Abstract

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers [\[47\]](#). A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred.

HTTP has been in use by the World-Wide Web global information initiative since 1990. This specification defines the protocol referred to as "HTTP/1.1", and is an update to [RFC 2068](#) [\[33\]](#).

IL LINGUAGGIO HTML

Il linguaggio **html** (hypertext-markup-language) è un linguaggio di **markup** e viene utilizzato per rappresentare

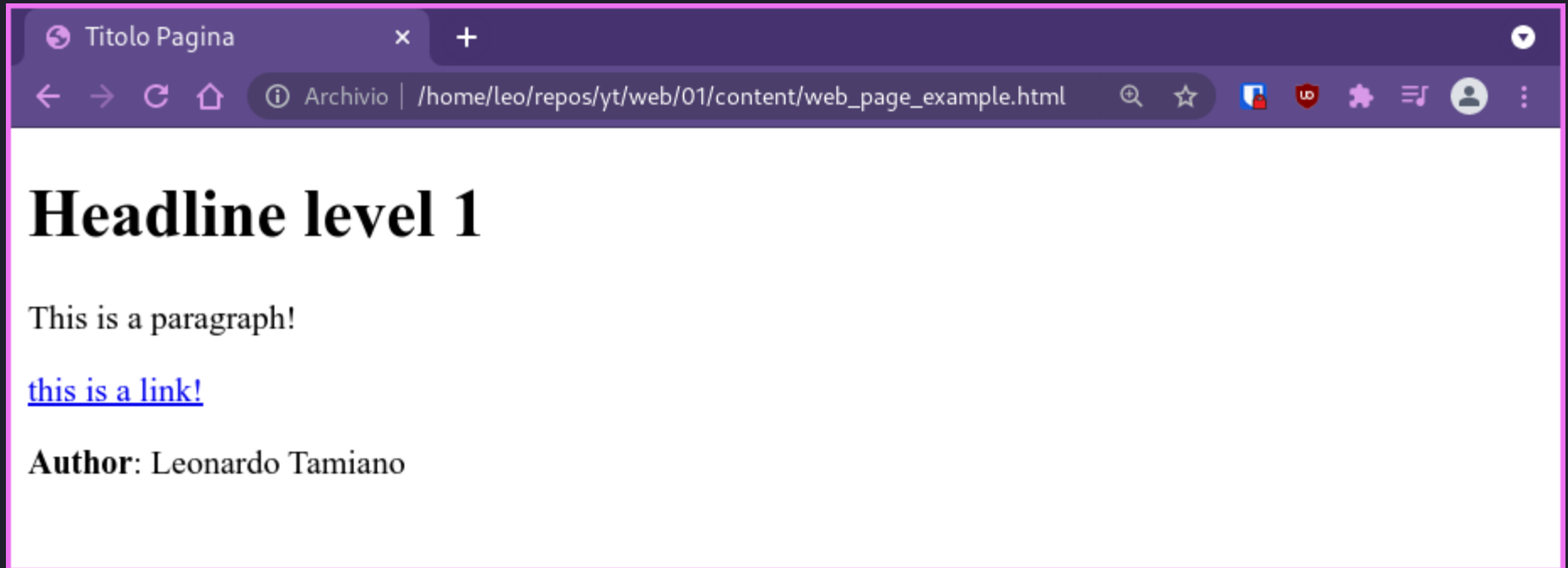
1. Il contenuto del documento.
2. La struttura del documento.

I documenti html infatti sono documenti **strutturati** ed **ipertestuali**.

```
<!DOCTYPE html>
<head>
  <title> Titolo Pagina </title>
  <meta name="viewport"
        content="width=device-width, initial-scale=1">
  <meta http-equiv="Content-Type"
        content="text/html; charset=UTF-8">
  <meta name="generator"
        content="Org-mode">
  <meta name="author"
        content="Leonardo Tamiano">
</head>

<body>
  <div id="content">
```

I **browsers** sono dei programmi in grado di leggere i documenti html e processarli in modo da renderli accessibili agli utenti finali.



UNIFORM RESOURCE LOCATORS (URLS)

Per specificare la **risorsa** a cui si vuole accedere nel web, il client utilizza un **URL** (Uniform Resource Locator).

Un **URL** specifica:

1. Il server di interesse.
2. Il protocollo che si vuole utilizzare.
3. Il particolare file, nel server di interesse.

Alcuni esempi di URLs:

`https://leonardotamiano.xyz`

`https://leonardotamiano.xyz/index.html`

`https://leonardotamiano.xyz/ppa/README.html`

`https://leonardotamiano.xyz:80/index.html`

`https://leonardotamiano.xyz:90/index.html`

Per poter processare questi URLs e trovare gli effettivi server però è necessario utilizzare anche il protocollo **DNS** (Domain Name System).

Il DNS si occupa di tradurre i **nomi simboli** in indirizzi **IP**.

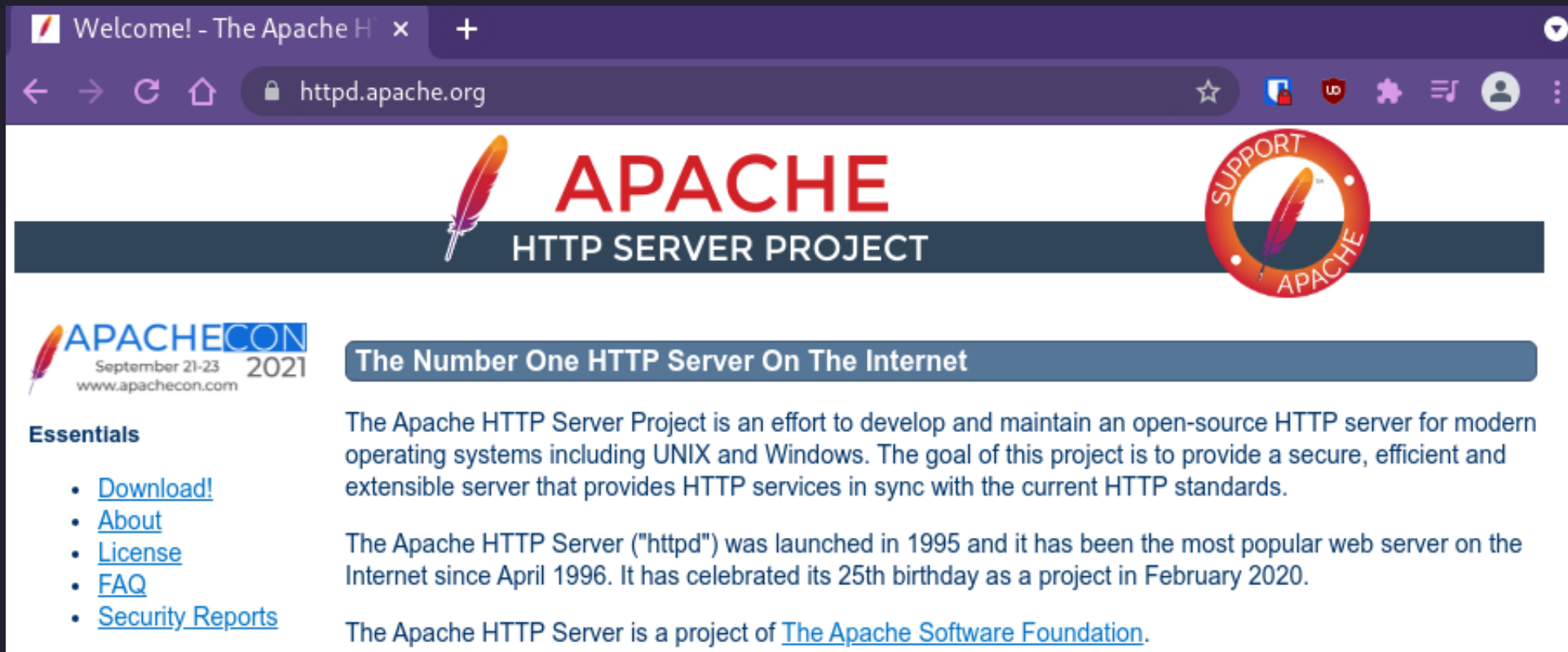
```
leonardotamiano.xyz - - - - -> 45.76.93.206
```

Riassumendo, per il funzionamento di un solo web server sono necessarie le seguenti tecnologie:

- Protocolli
 - HTTP, DNS, TCP, IP
- Linguaggio HTML
- Browser
- Web Server
- Internet
- Sistema Operativo

APACHE2


Apache2 è uno dei più importanti web servers open-source, ed è stato fondamentale per lo sviluppo del web.



The screenshot shows a web browser window with the URL `httpd.apache.org`. The page features the Apache logo (a feather) and the text "APACHE HTTP SERVER PROJECT". A banner for "APACHECON" is visible on the left, dated September 21-23, 2021. The main content area is titled "The Number One HTTP Server On The Internet" and describes the project's goals and history. A sidebar on the left lists "Essentials" including links for Download, About, License, FAQ, and Security Reports. The footer mentions the project is part of The Apache Software Foundation.


Welcome! - The Apache HTTP Server Project


← → ↻ 🏠 🔒 httpd.apache.org ☆ 🔍 ⚙️ 👤 ⋮



APACHE

HTTP SERVER PROJECT





September 21-23 2021
www.apachecon.com

The Number One HTTP Server On The Internet

The Apache HTTP Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows. The goal of this project is to provide a secure, efficient and extensible server that provides HTTP services in sync with the current HTTP standards.

The Apache HTTP Server ("httpd") was launched in 1995 and it has been the most popular web server on the Internet since April 1996. It has celebrated its 25th birthday as a project in February 2020.

The Apache HTTP Server is a project of [The Apache Software Foundation](#).

Essentials

- [Download!](#)
- [About](#)
- [License](#)
- [FAQ](#)
- [Security Reports](#)

Per installarlo possiamo procedere come segue

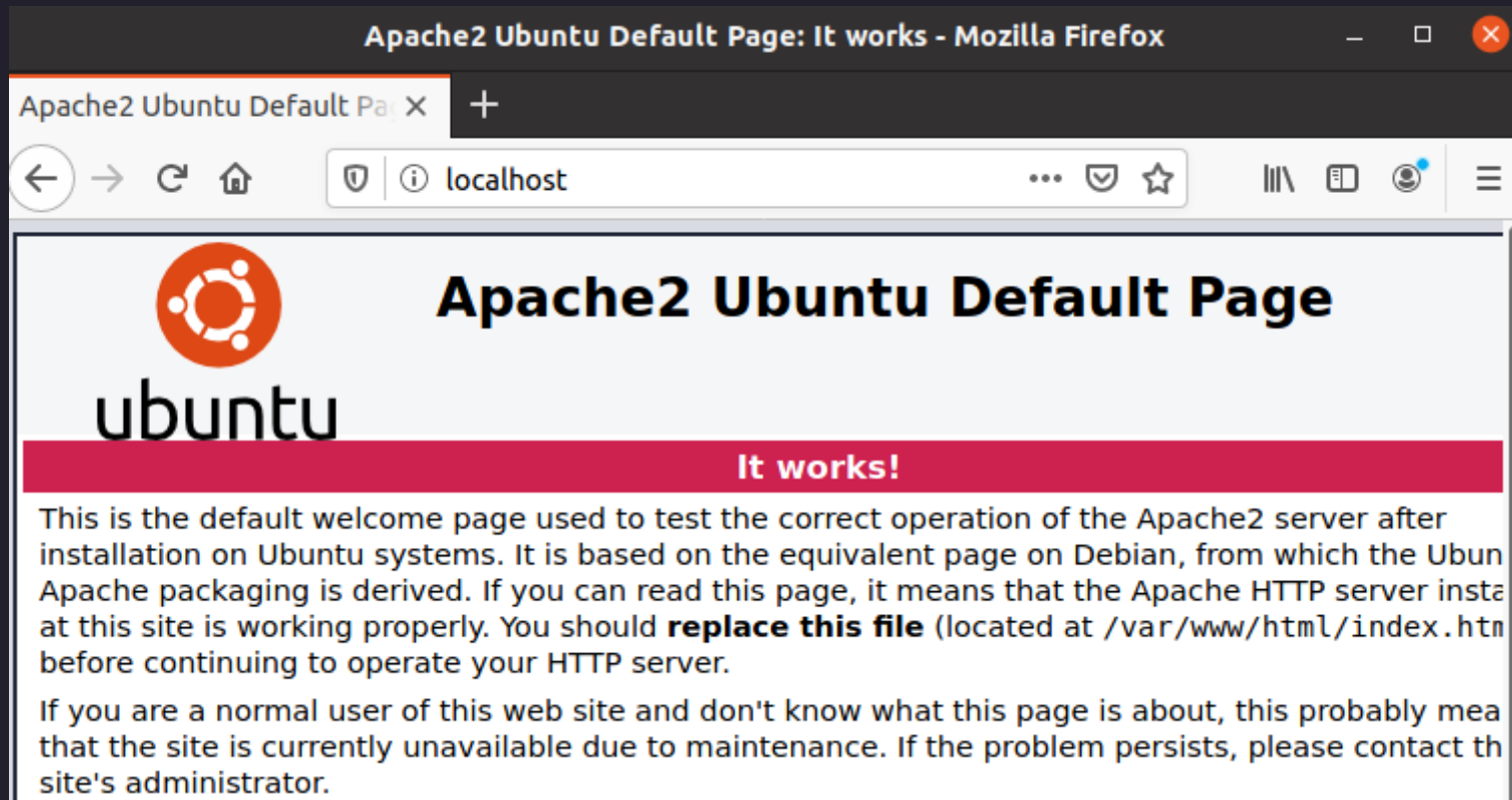
```
sudo apt update  
sudo apt install apache2
```

Una volta installato lo possiamo attivare

```
sudo service apache2 start
```

Per verificare se è stato attivato correttamente
possiamo andare nel seguente URL:

<http://localhost>



A questo punto ci possiamo chiedere:

*come possiamo esporre pubblicamente
un nuovo file?*

Per rispondere a questa domanda è necessario introdurre il concetto della **root directory** nel contesto dei web servers.

La **root directory** è una cartella del sistema operativo su cui gira il web server.

Il web server espone ai vari clients proprio le risorse che si trovano nella root directory, o in una sotto-cartella che passa per la root directory.

La root directory di default per apache2 è

`/var/www/html`

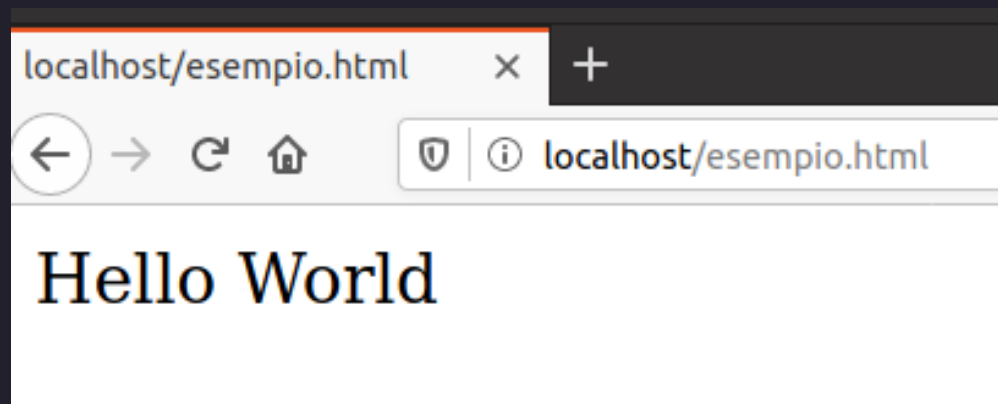
Se vogliamo esporre pubblicamente un nuovo file, dobbiamo copiarlo nella root directory del web server che stiamo configurando.

Nel nostro caso, eseguendo

```
echo "Hello World" > /var/www/html/esempio.html
```

creiamo una nuova risorsa, che può essere vista andando al seguente url

<http://localhost/esempio.html>



IL LIMITE DELLE PAGINE STATICHE

Tramite un web server siamo in grado di fornire ai vari clients delle risorse.

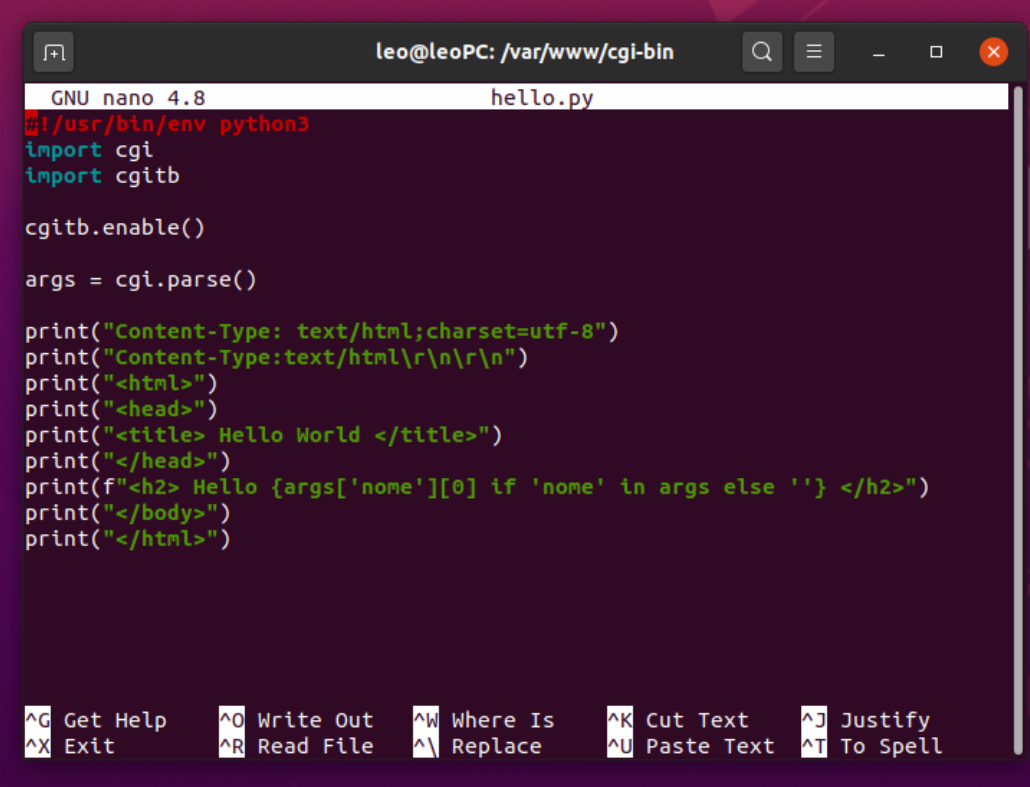
Le risorse che possiamo offrire così però sono risorse **statiche**, ovvero non cambiano a seconda di chi sta richiedendo la risorsa.

Tutti i client vedono le stesse risorse.

Questo pone dei limiti rispetto a ciò che possiamo fare con delle pagine web.

Per cercare di superare questi limiti è stata introdotta la **Common Gateway Interface** (cgi), che serve per collegare il server a degli scripts presenti nel file system del server.

Questi scripts possono essere utilizzati per rendere dinamico il contenuto offerto dal sito (1/4)



The image shows a terminal window with a dark background. The title bar at the top reads 'leo@leoPC: /var/www/cgi-bin'. Below the title bar, the text 'GNU nano 4.8' is visible on the left and 'hello.py' on the right. The main area of the terminal contains a Python script. The script starts with a shebang line '#!/usr/bin/env python3', followed by imports for 'cgi' and 'cgi.tb'. It then calls 'cgi.tb.enable()', parses command-line arguments, and prints a series of headers and HTML tags to output a 'Hello World' message. The script uses an if-statement to check for a 'nome' parameter in the arguments. At the bottom of the terminal, there is a row of keyboard shortcuts for nano editor functions: '^G Get Help', '^O Write Out', '^W Where Is', '^K Cut Text', '^J Justify', '^X Exit', '^R Read File', '^_ Replace', '^U Paste Text', and '^T To Spell'.

```
#!/usr/bin/env python3
import cgi
import cgi.tb

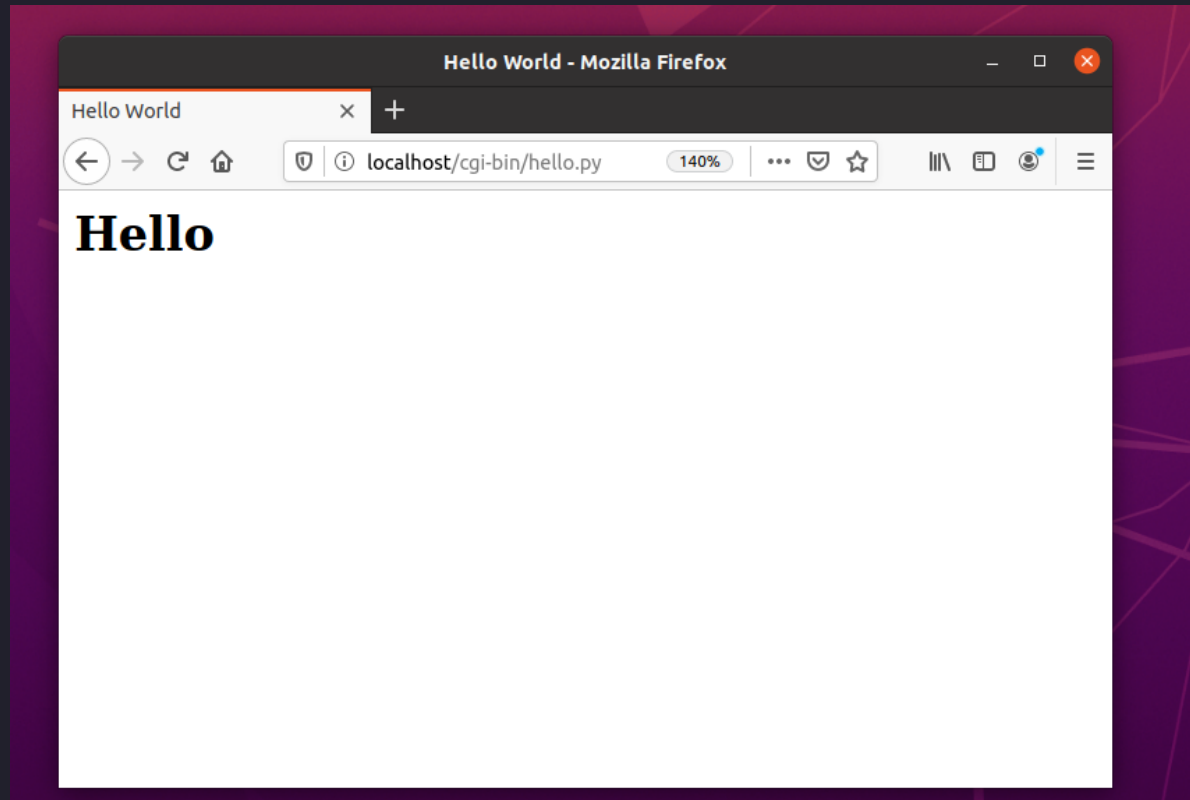
cgi.tb.enable()

args = cgi.parse()

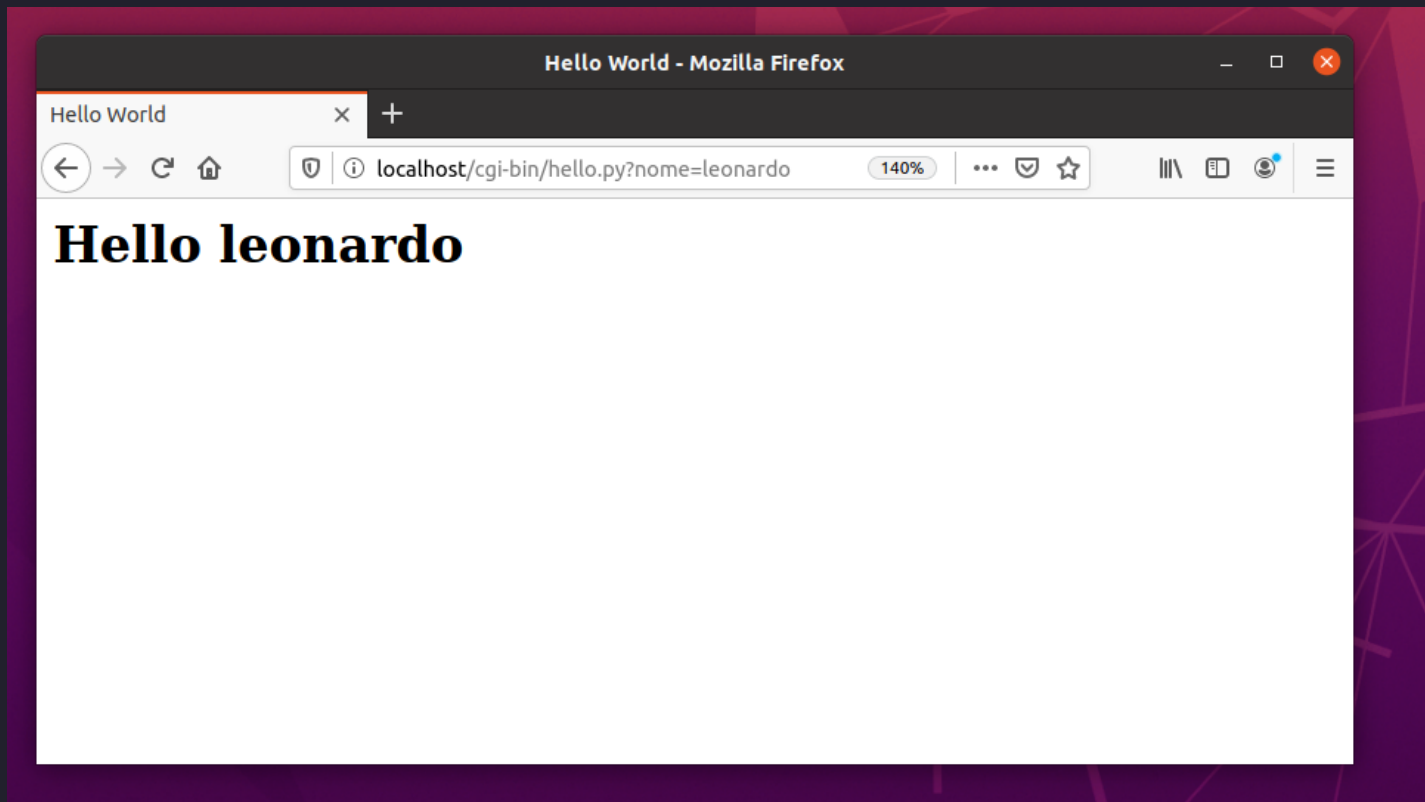
print("Content-Type: text/html;charset=utf-8")
print("Content-Type:text/html\r\n\r\n")
print("<html>")
print("<head>")
print("<title> Hello World </title>")
print("</head>")
print(f"<h2> Hello {args['nome'][0] if 'nome' in args else ''} </h2>")
print("</body>")
print("</html>")
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File ^_ Replace ^U Paste Text ^T To Spell

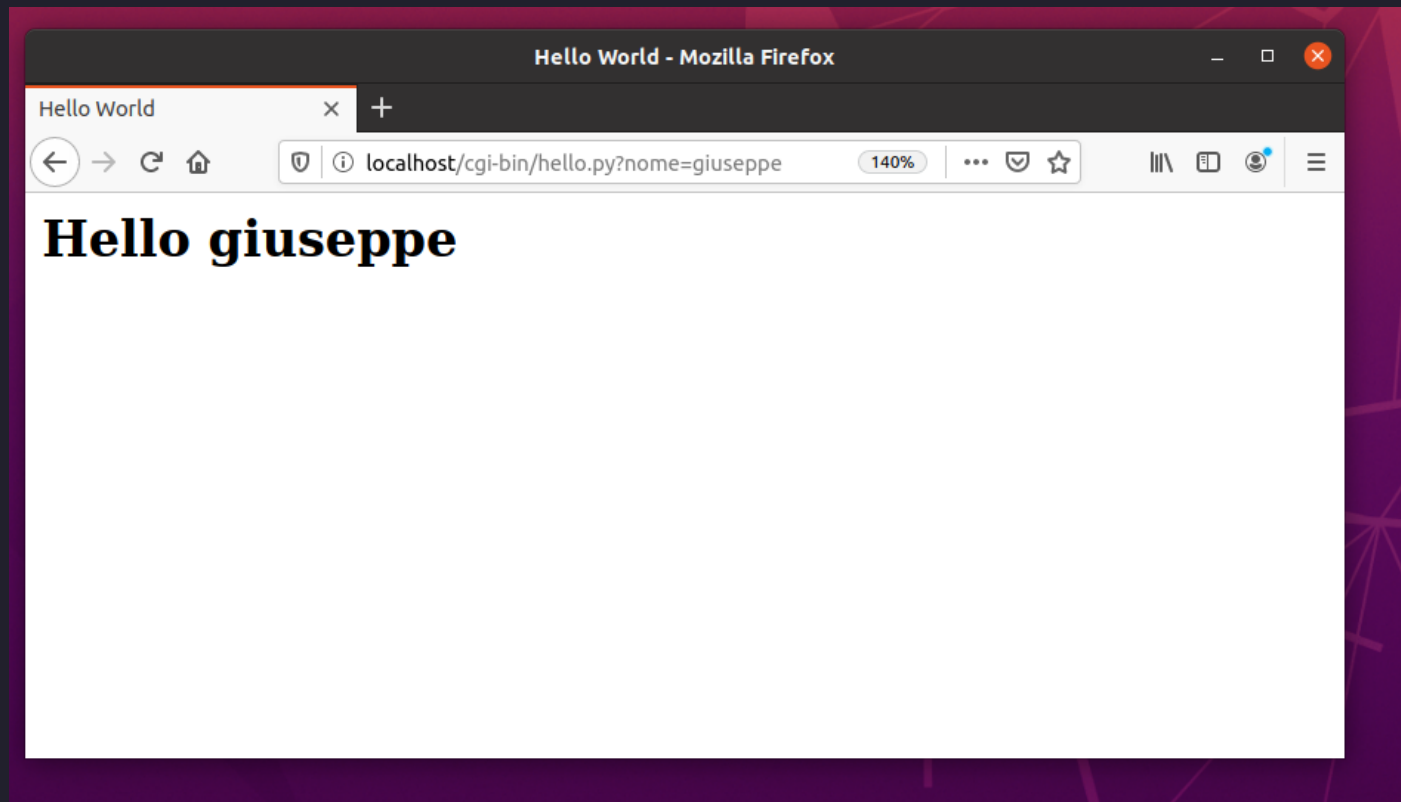
Questi scripts possono essere utilizzati per rendere dinamico il contenuto offerto dal sito (2/4)



Questi scripts possono essere utilizzati per rendere dinamico il contenuto offerto dal sito (3/4)

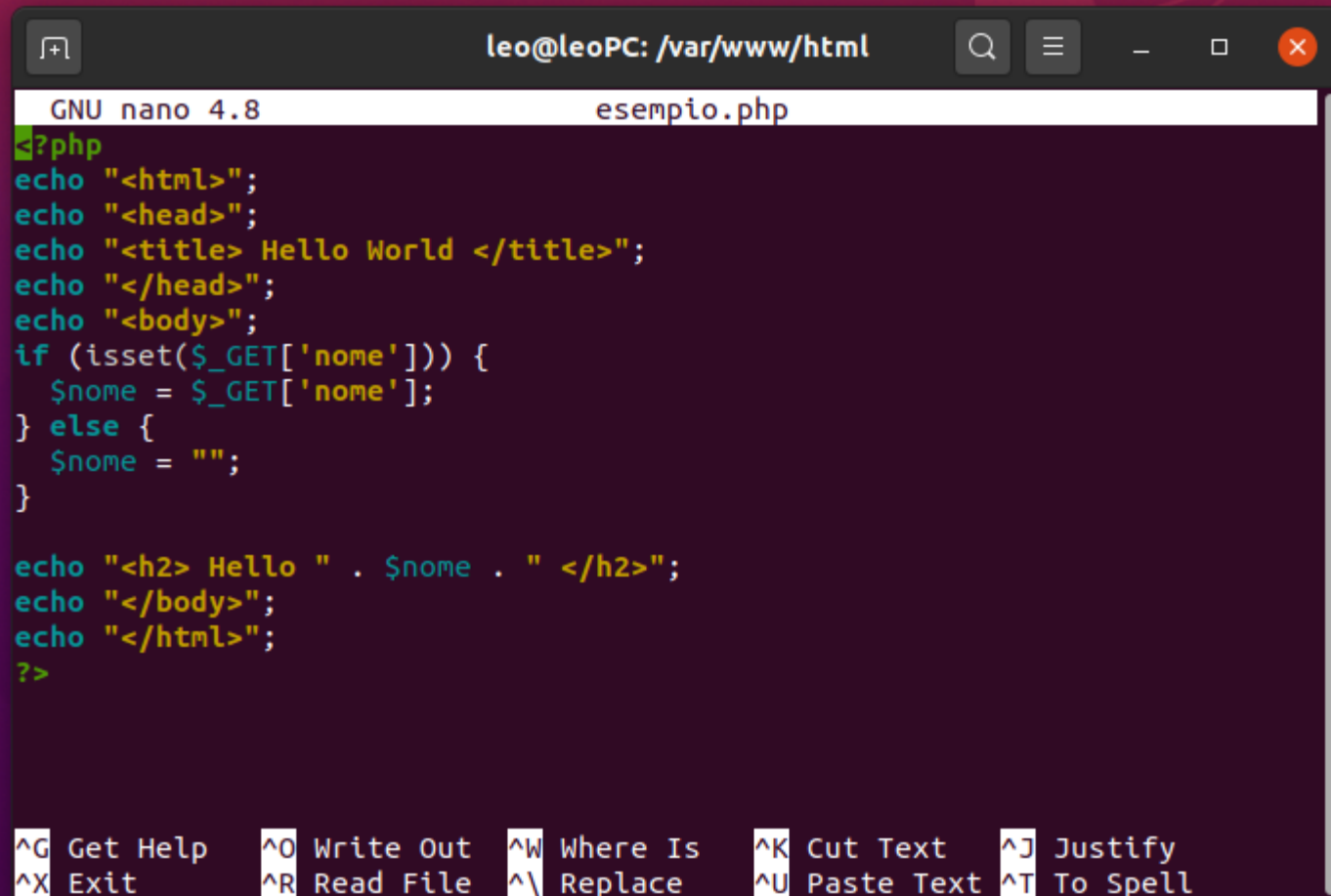


Questi scripts possono essere utilizzati per rendere dinamico il contenuto offerto dal sito (4/4)



Col passare del tempo sono stati introdotti dei veri e propri linguaggi, come il linguaggio **php**, per effettuare questa generazione dinamica della pagine.

L'esempio di prima in **php** (1/3)

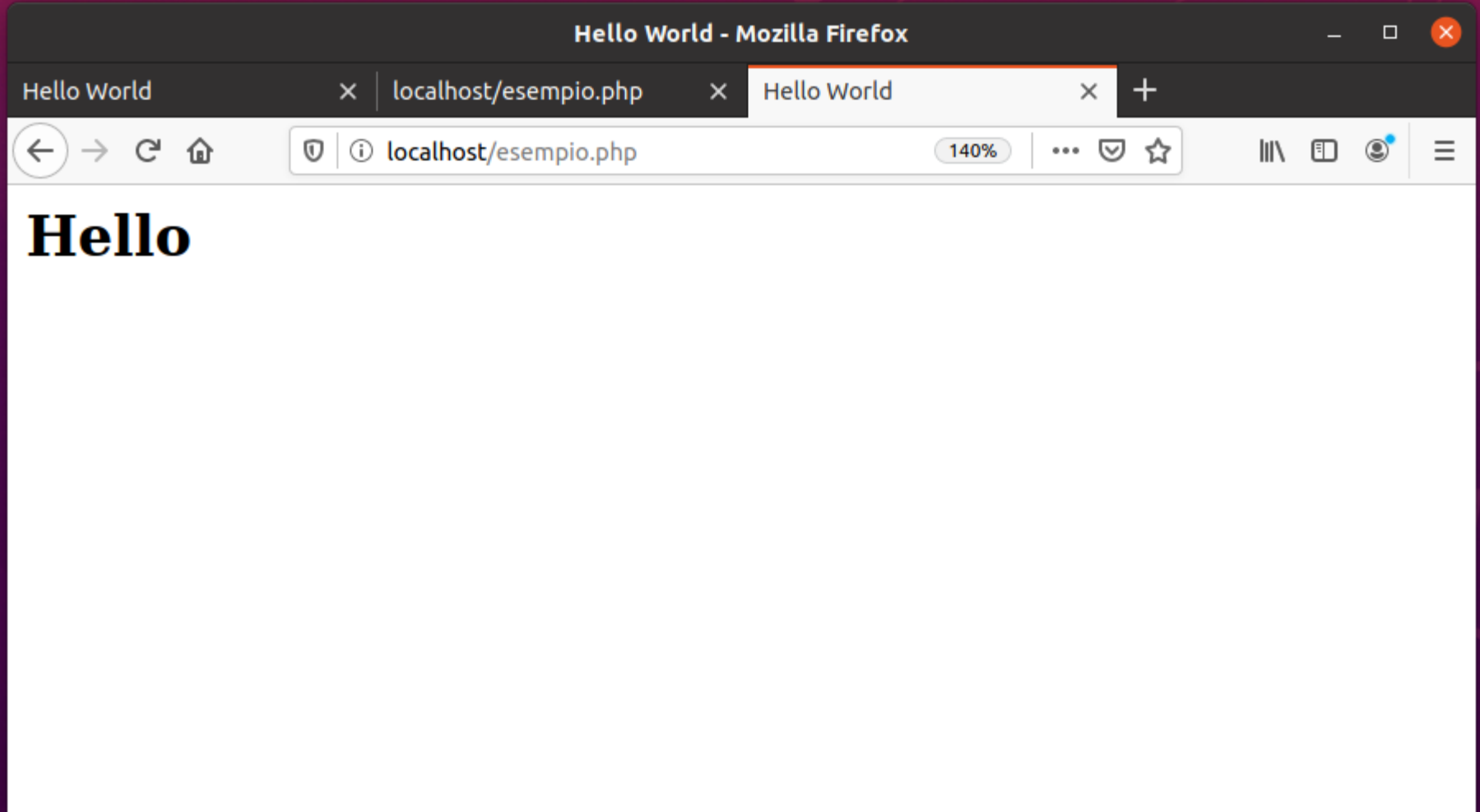


```
leo@leoPC: /var/www/html
GNU nano 4.8      esempio.php
<?php
echo "<html>";
echo "<head>";
echo "<title> Hello World </title>";
echo "</head>";
echo "<body>";
if (isset($_GET['nome'])) {
    $nome = $_GET['nome'];
} else {
    $nome = "";
}

echo "<h2> Hello " . $nome . " </h2>";
echo "</body>";
echo "</html>";
?>
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell

L'esempio di prima in **php** (2/3)



L'esempio di prima in **php** (3/3)



Le **backend engine** servono proprio per questo:
generare delle pagine il cui contenuto è dinamico,
ovvero cambia a seconda di chi sta facendo la richiesta.

