

Architettura del Web

Laboratorio di basi di dati
Dipartimento di Informatica
Università degli Studi di Milano



Introduzione al web

- Il Web attuale si fonda su un'architettura gerarchica conforme al modello client/server
- Le macchine server ospitano le informazioni organizzate in siti web e le rendono disponibili ad eventuali richieste di macchine client
- Tipicamente, i client sottopongono richieste HTTP ai server mediante un browser
- Le comunicazioni possono veicolare diverse tipologie di contenuto (testo, dati binari)



WWW e HTTP

WWW e HTTP

World Wide Web (WWW)

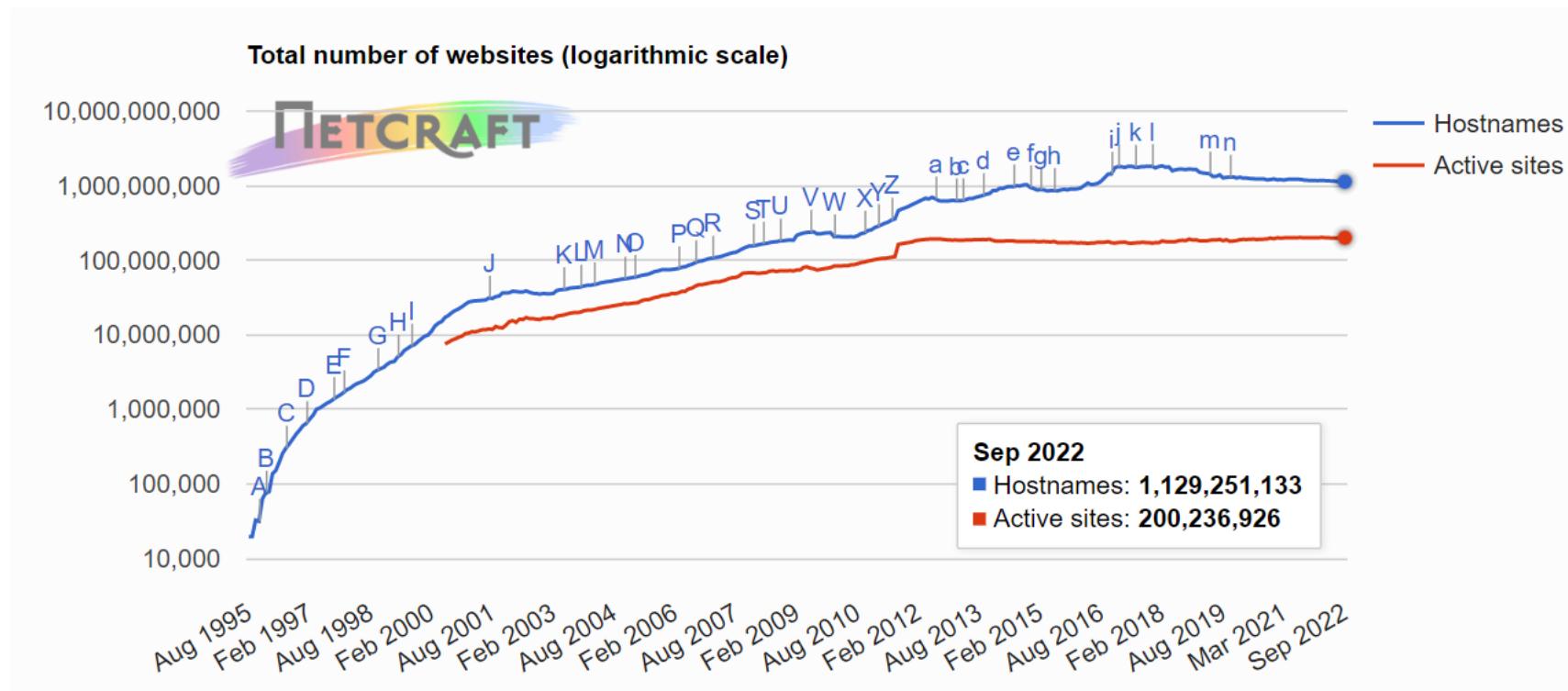
- World Wide Web applicazione Internet più usata
 - Definita da Tim Berners-Lee nel 1990 (CERN – Ginevra)
 - Fornisce accesso a contenuto multimediale
 - Nodi (pagine Web) memorizzati su macchine in tutto il mondo e connesso alla rete Internet
 - Open service
- Architettura software che fornisce accesso a documenti *linkati* distribuiti su migliaia/milioni di macchine

Importanza del Web

- All'inizio della storia di Internet, il trasferimento di dati FTP contava un terzo di tutto il traffico
- All'inizio del 1990, viene introdotto il Web
 - Crescita esponenziale
- Nel 1995, il traffico Web supera quello FTP
 - Greatest bandwidth consumer
- Nel 2000, il traffico Web supera di gran lunga quello di altri servizi

Alcune Statistiche - 1

- 1991-1997: Explosive growth, at a rate of 850% per year.
- 1998-2001: Rapid growth, at a rate of 150% per year.
- 2002-2006: Maturing growth, at a rate of 25% per year.
- September 2020: 1,129 billions of hostnames (200 millions active sites)

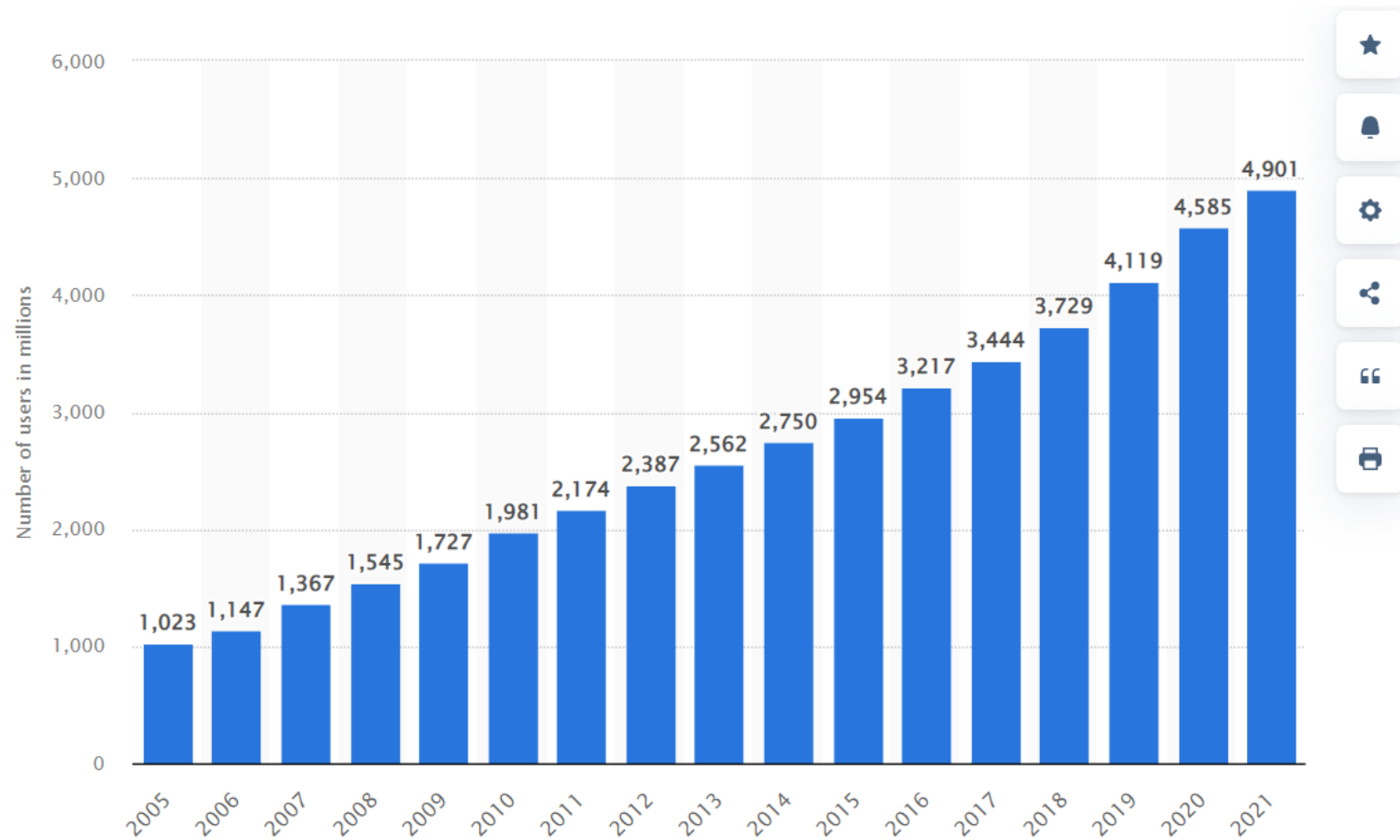


Alcune Statistiche - 2

- Il Web ha un impatto maggiore delle statistiche
 - Ogni persona nei paesi sviluppati accede al Web
 - WWW è un “diritto dell’uomo”
 - Internet.org tenta di raggiungere tutti

Alcune Statistiche - 3

- Numero di utenti internet nel mondo



<https://www.statista.com/statistics/273018/number-of-internet-users-worldwide/>

The background consists of a dark blue field with a lighter blue square on the left and a large, semi-transparent light blue circle on the right.

World Wide Web (WWW)

World Wide Web (WWW)

- L'applicazione internet più usata
- Può essere definita come un insieme di ipertesti multimediali distribuiti su molti nodi
- Fornisce accesso a documenti ipertestuali attraverso la rete internet
 - Distribuito e scalabile

Iper testo

- Documento con una struttura non sequenziale
- Composto di diverse sezioni linkate
 - Link interni ed esterni
 - Permette agli utenti di ricercare e muoversi tra argomenti collegati
 - Indipendente dall'ordine di presentazione
- C'è più di solo testo (hypermedia)
 - Immagini, musica, video, ...

Componenti architettonici

- Paradigma a ipermedia distribuito
 - Iper: link per riferirsi ad altri documenti
 - Media: più di semplice testo
 - Consiste di pagine web (documenti ipermediali)
- Building block del web
 - Web browser: programma applicativo che permette agli utenti di accedere e visualizzare pagine web
 - Web server: entità che fornisce le pagine web al web browser

Architettura

- Architettura client-server
 - Il **client** (Web browser) è un programma che si connette al server, sottomette una richiesta e aspetta una risposta
 - Il **server** (Web server) è un programma “in ascolto” (su una porta TCP) che fornisce un servizio
- Data una richiesta del client, il server
 - Analizza la richiesta
 - Prepara la risposta
 - Ritorna la risposta al client

Comunicazione via Web

Quando l'utente fa clic su un collegamento nel browser, il browser identifica il server Web e invia una richiesta per quella pagina.



**Computer su cui è
installato un
browser Web**



Server Web

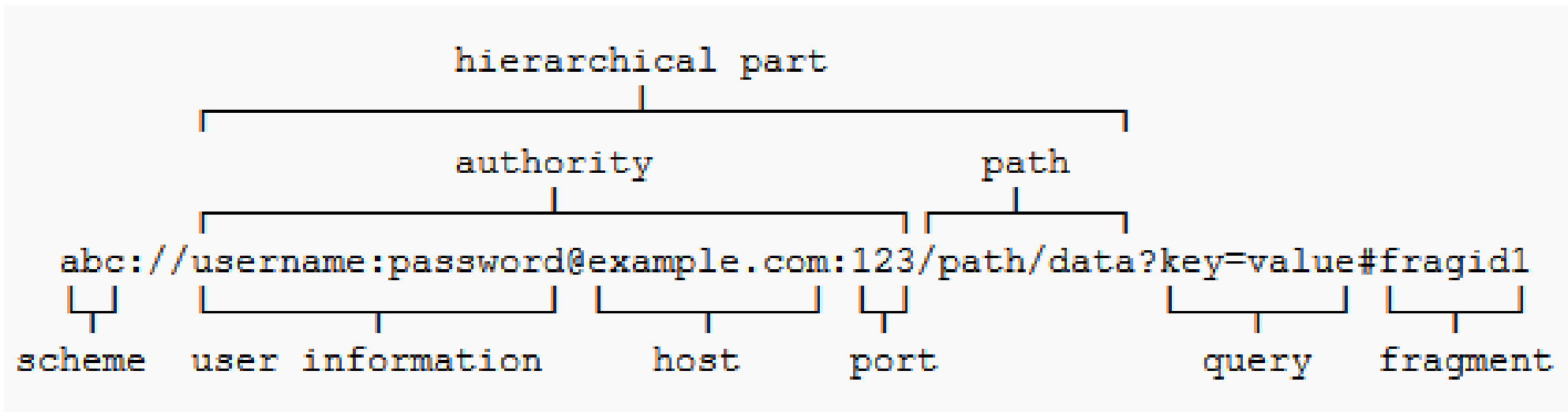
**Il server individua la pagina
specificata nelle sue directory/cartelle
e invia indietro la pagina al browser
per visualizzarla.**

Concetti base

- Uniform Resource Locator (URL)
- HyperText Transfer Protocol (HTTP)
- HyperText Markup Language (HTML)

Web Page Identifier

- Miliardi di pagine accessibili attraverso il web
- Ogni pagina deve avere un nome univoco
- Uniform Resource Identifier (URI)
 - Sintassi universale con cui identificare le risorse di rete
 - Indipendente dal protocollo
 - Basato su un set di caratteri ristretto, per evitare problemi di cattiva codifica
 - Non definisce la semantica di una risorsa



Web Page
Identifier

Web Page Identifier

- Uniform Resource Locator (URL) è un'URI che in aggiunta a identificare una risorsa fornisce anche un meccanismo per localizzarla
 - Sottoinsieme delle URI
- Rispetta la sintassi URI
 - `http: // hostname [: port] / path [? query]`
(versione semplificata)
 - Scheme definisce lo schema usato per la comunicazione
 - Il formato dell'URL dipende dallo schema
 - Campi tra parentesi quadre sono opzionali

Web Page Identifier

- Encoding
 - Protocollo usato (e.g., http)
 - Nome di dominio del server (o indirizzo IP)
 - Numero di porta del protocollo (opzionale)
 - Necessaria solo quando una porta di default non è specificata per il protocollo
 - Percorso attraverso il file system del server
 - Query (opzionali)

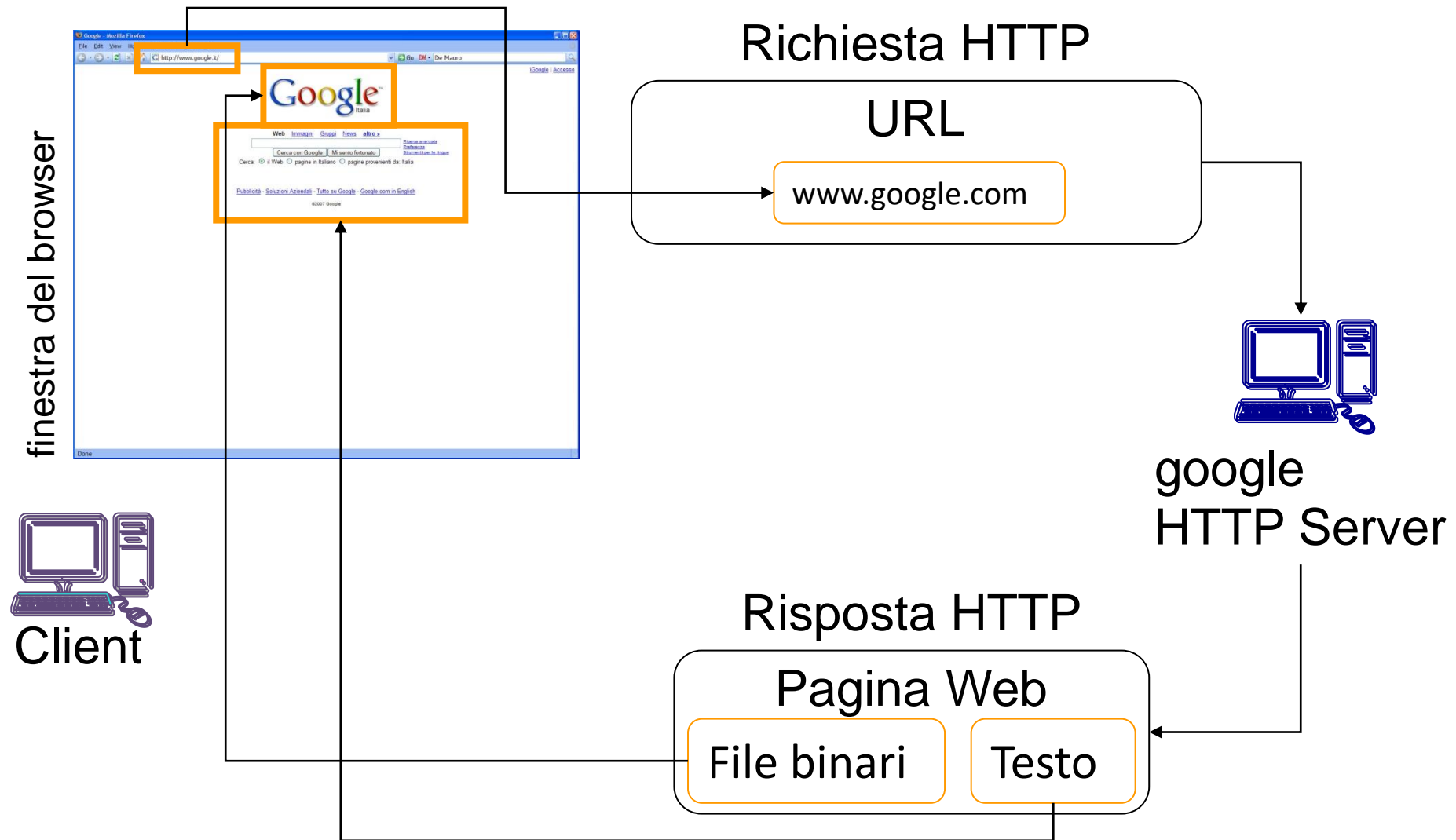
Web Page Identifier

- URL può assumere due diversi formati
 - **Assoluto**
 - Usato dagli utenti del web
 - Contiene una specifica completa dell'URL
 - E.g., homes.di.unimi.it/bellandi
 - **Relativo**
 - Significativo solo quando un server è già stato identificato
 - Omette l'indirizzo del server
 - E.g., /bellandi/didattica/

Web Standards

- Standard separati per
 - **Trasferimento (HTTP)**
 - **Rappresentazione (HTML)**

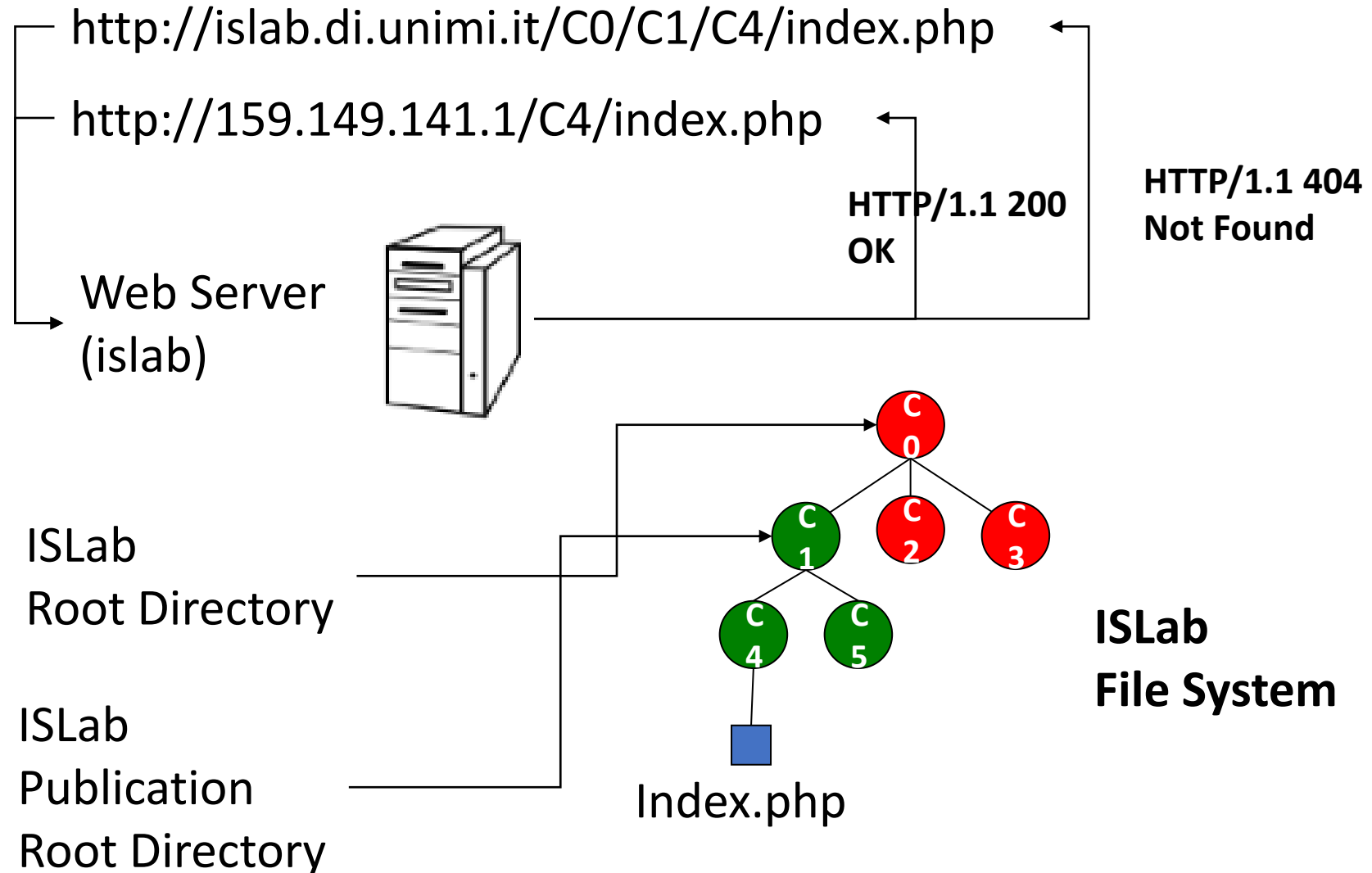
Esempio di interazione HTTP



Struttura di un server Web

- Un Web server è un elaboratore che ospita un software per la gestione di comunicazioni HTTP (software denominato HTTP server)
- Esistono diversi software di tipo HTTP server
- Il Web server espone (pubblica) verso la rete **una porzione del proprio file system** il cui contenuto è consultabile tramite richieste HTTP
- Tale porzione è identificata mediante una *publication root directory* (radice di pubblicazione)

Esempio di risoluzione di una richiesta



The background consists of a dark blue gradient. A large, semi-transparent circle is positioned on the right side of the image. A vertical line of a slightly lighter blue shade runs through the center of the circle, extending from the top to the bottom of the frame.

HTTP

Accesso al web

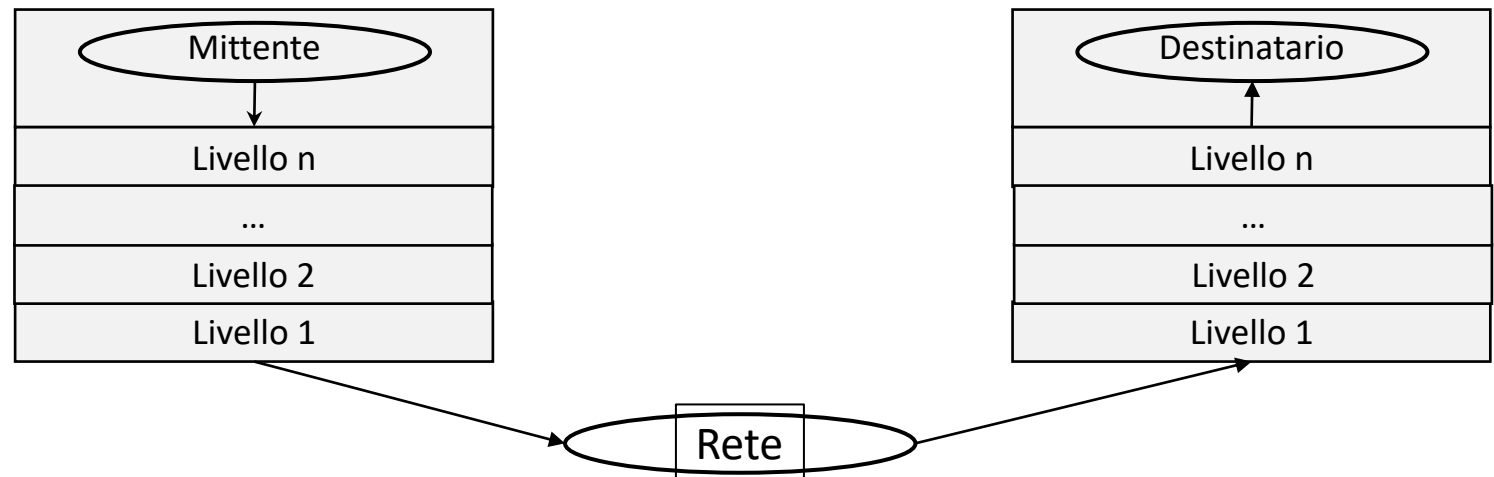
- Semplice e intuitivo
 - Utente inserisce un'URL o clicca su un link
 - Il browser parse l'URL, estrae le informazioni e riceve una copia della pagina
 - Il server riceve la richiesta, seleziona la pagina e la ritorna al browser
 - Il browser parse la pagina HTML e la mostra all'utente

Trasferimento

- Protocollo *HyperText Transfer Protocol (HTTP)*
- Si basa su **TCP**
- Usato tra browser e Web server
- Definisce in che modo
 - i client richiedono le pagine ai server Web
 - i server Web trasferiscono le pagine ai client

Background: stratificazione dei protocolli di rete

- I sistemi di comunicazione dei dati operano tramite un **insieme di protocolli cooperativi**
- I problemi da gestire sono molteplici → ogni livello (protocolli) responsabile di gestirne una parte
 - Guasti dell'HW
 - Congestione delle reti
 - Ritardo e/o perdita di pacchetti in transito
 - Danneggiamento dei dati
 - ...
- Modello concettuale dei protocolli: **stratificazione in livelli**



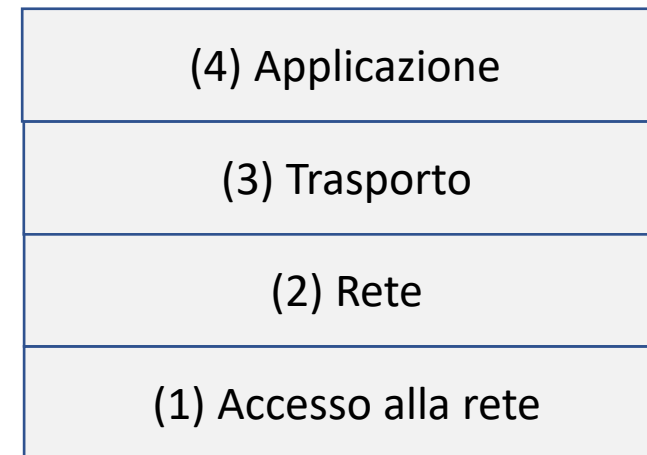
Background: il modello ISO/OSI

- Modello concettuale che struttura la comunicazione fra sistemi in **7 livelli di astrazione**
 - Da livello più basso (trasmissione dei singoli bit, livello 1)
 - A livello più alto (protocolli di comunicazione e interfacce degli host, livello 7)
- Ogni livello ***I*** fornisce funzionalità al livello successivo ***I+1*** e usa funzionalità del livello precedente ***I-1***



Background: il modello TCP/IP

- Modello concettuale che struttura la comunicazione fra sistemi in **4 livelli di astrazione**
 - Da livello più basso (trasmissione dei singoli bit, livello 1)
 - A livello più alto (protocolli di comunicazione e interfacce degli host, livello 4)
- Ogni livello ***l*** fornisce funzionalità al livello successivo **$l+1$** e usa funzionalità del livello precedente **$l-1$**



Background: il modello ISO/OSI Vs. TCP/IP

(7) Applicazione
(6) Presentazione
(5) Sessione
(4) Trasporto
(3) Rete
(2) Data Link
(1) Fisico

(4) Applicazione
(3) Trasporto
(2) Rete
(1) Accesso alla rete

e.g., **HTTP**,
FTP, **SMTP**

e.g., **TCP**,
UDP

e.g., **IP**

Background: imbustamento

- I dati da trasmettere sono suddivisi in segmenti composti da informazioni di controllo (e.g., intestazioni) e dati veri e propri (payload)
 - Le intestazioni sono aggiunte in ciascuno strato da ciascun protocollo di rete utilizzato nella comunicazione
- Ad ogni livello un protocollo di rete aggiunge alla rete funzioni di complessità via via crescenti, basandosi sui servizi offerti dai livelli inferiori

(4) Applicazione
(3) Trasporto
(2) Rete
(1) Accesso alla rete

Ad esempio, una pagina web:

1. viene imbustata con intestazioni HTTP (livello 4), per essere trasmessa
2. su una connessione TCP (livello 3). TCP segmenta il flusso di dati in pacchetti di dimensione adatta alla rete sottostante, aggiunge ad ogni pacchetto le sue intestazioni, e
3. li passa a IP (livello 2), che aggiunge la propria intestazione e
4. la passa al protocollo di rete locale (e.g., Ethernet, livello 1) che aggiunge le proprie intestazioni e
5. passa il pacchetto da trasmettere al trasmettitore che invia sul mezzo fisico (e.g., un cavo in rame).

Trasferimento

- Quando viene richiesta una pagina Web (con URL o cliccando su hyperlink)
 - Browser invia al server messaggi di **richiesta** HTTP per gli oggetti nella pagina
 - Il server risponde con messaggi di **risposta** HTTP che contengono gli oggetti
- Nota: operando sopra TCP, richieste e risposte HTTP (**prima o poi**) arriveranno a destinazione
 - HTTP (livello più alto negli stack) non si preoccupa di dati smarriti o in disordine (in capo a TCP e protocolli di livello inferiore)

HyperText Transfer Protocol (HTTP)

- Protocollo di comunicazione per lo scambio di messaggi e ipertesti
 - Set-up della connessione
 - Web browser verifica la disponibilità del server
 - HTTP request
 - HTTP response
 - Connection closed

Caratteristiche

- Livello applicativo
 - Assume un protocollo di trasporto affidabile e orientato alla connessione
 - Non fornisce affidabilità e ritrasmissione
- Paradigma richiesta/risposta
 - La comunicazione inizia con una richiesta HTTP
- Stateless
 - Il server non mantiene lo stato della connessione

Caratteristiche

- Permette trasferimenti bi-direzionali (ad es., form)
- Negoziamenti delle capability
 - Il mittente specifica le sue capability e il ricevente le accetta tutte o in parte (ad es., negoziazione del character set)
- Supporto per caching
 - Diminuisce il tempo di risposta
 - Per richieste alla stessa pagina, il browser determina se la pagina è cambiata
- Supporta intermediari nella comunicazione
 - Proxy server

Caratteristiche (Sommario)

- Protocollo a livello applicativo per sistemi informativi ipermediali, collaborativi e distribuiti
 - Orientato agli oggetti, generico, stateless
 - Può essere usato per molte applicazioni come i sistemi distribuiti di gestione degli oggetti
 - Protocollo principale del Web
 - Uno dei pochi protocolli applicativi per cui è garantita la connettività tra qualsiasi coppia di punti su Internet (firewall traversal)

Connessioni HTTP

- Browser invia la richiesta a cui il server risponde
 - Accede all'URL
 - Estrae l'hostname
 - Contatta il DNS per ottenere l'indirizzo IP
 - Crea una connessione TCP con il server
 - Una volta che la connessione TCP è pronta, HTTP è usato per comunicare
 - Il browser invia una query e il server la soddisfa
- HTTP può usare connessioni TCP persistenti (default) e non persistenti

Connessioni HTTP: passaggio parametri

- Ad una richiesta HTTP, è **possibile** associare una lista di parametri
- A tale scopo sono disponibili il metodo **GET** e il metodo **POST**
- Sintassi **GET**
 - `http://URL?par0=valore0&par1=valore1`
- Sintassi **POST**
 - `par0=valore0`
 - `par1=valore1`

Connessioni HTTP: passaggio parametri

- **GET**
 - la lista di parametri viene specificata nella URL della richiesta HTTP
 - il numero di caratteri utilizzabili per i parametri è limitato (i.e., 255 byte)
 - sono ammessi solo parametri in formato testuale
- **POST**
 - la lista di parametri viene specificata nel corpo della richiesta HTTP
 - il numero di caratteri utilizzabili per i parametri non ha un limite prefissato
 - sono ammessi parametri in diversi formati (e.g., video, audio)

Trasferimento con connessioni TCP non persistenti

Trasferimento di una pagina Web che contiene

- File di base html all'URL
`http://www.someSchool.edu/someDepartment/home.html`
- 10 immagini JPEG sullo stesso server

1. Il client HTTP inizia una connessione TCP con il server `www.someSchool.edu` (porta 80)
2. Tramite una socket associata alla connessione TCP, il client HTTP invia al server una richiesta HTTP che contiene il percorso `/someDepartment/home.html`
3. Il server recupera l'oggetto al percorso, lo incapsula in una risposta HTTP, e lo invia al client
4. Il server chiude la connessione TCP (chiusa quando la risposta HTTP è ricevuta)
5. Il client esamina la risposta, trova i riferimenti alle 10 immagini JPEG, ripete i passi 1-4 per ognuna delle immagini

Trasferimento con connessioni TCP persistenti

Trasferimento di una pagina Web che contiene

- File di base html all'URL
`http://www.someSchool.edu/someDepartment/home.html`
- 10 immagini JPEG sullo stesso server
- Funzionamento analogo a quanto avviene con connessioni non persistenti, ma la connessione TCP non viene chiusa fra un trasferimento di un oggetto ed un altro
 - Migliori performance in quanto il setup di una connessione TCP richiede allocazione e mantenimento di buffer e variabili TCP lato server
 - In genere, connessione TCP chiusa dopo un timeout
- Senza pipelining / con pipelining (default)
 - Richiesta di un nuovo oggetto solo dopo aver ricevuto oggetto precedente / indipendentemente dall'averlo ricevuto

Formato messaggi HTTP

- HTTP request
 - Specifica del metodo di invio (POST/GET)
 - Specifica della URL
 - Specifica della versione del protocollo
- HTTP response
 - Specifica il codice della risposta (e.g., ok, not found, bad request, unauthorized)
 - Specifica il tipo di dati inviato
 - Specifica la dimensione del pacchetto di risposta

Formato messaggi HTTP: richiesta

```
GET /someDirectory/page.html HTTP/1.1
Host: www.someSchool.edu
Connection: close
User-agent: Mozilla/5.0
Accept-language: en-GB, en-US
```

- Prima riga (richiesta):
 - Campo metodo (GET, POST, HEAD, ...)
 - Campo URL
 - Campo versione
- Righe successive (intestazione)
 - Host (necessario per proxy anche avendo già connessione TCP attiva)
 - Tipo di connessione TCP (persistente o no)
 - User-agent
 - Lingua della risorsa preferita (una delle negoziazioni possibili)
 - ...

Formato messaggi HTTP: risposta

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 03 Jul 2003 12:00:20 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Tue, 01 Jul 2003 09:00:15 GMT
Content-Length: 6281
Content-type: text/html
(data data data data data data ...)
```

- Prima riga (stato)
 - Versione protocollo, codice stato, messaggio stato
- 6 righe (intestazione)
 - Intenzione di chiudere la connessione TCP
 - Creazione e invio della risposta HTTP
 - Web server
 - Ultima modifica dell'oggetto incapsulato nella risposta
 - Dimensione e tipo dell'oggetto restituito
- Oggetto vero e proprio

Assenza di stato

- HTTP è un protocollo stateless
- Nelle diverse richieste di risorse che compongono una transazione Web non viene mai conservata alcuna informazione sullo stato di esecuzione
- L'assenza di stato implica la NON persistenza delle informazioni fra le pagine che compongono l'applicazione
- Allo scopo di simulare la persistenza di stato sono state individuate due soluzioni:
 - **Sessioni.** Memorizzano sul server le informazioni da mantenere per la persistenza
 - **Cookies.** Delegano al client la conservazione delle informazioni

Cookies

- Previsti dal protocollo HTTP
- Sistema per immagazzinare dati lato client
- Il browser, ad ogni richiesta, invia al server i dati immagazzinati nei cookie all'interno dell'header (intestazione) HTTP
- Generalmente composti da
 - Stringhe di testo
 - Data di scadenza
 - Pattern per riconoscere il dominio da cui sono stati definiti (possono essere letti solo da chi li ha definiti)

Cookies

- Possono essere classificati sulla base di:
 - Tipologia
 - Temporanei/di sessione: memorizzati fino alla chiusura del browser
 - Persistenti/di tracciamento: memorizzati fino a scadenza o cancellazione
 - Sorgente
 - First party
 - Third party
 - Scopo
 - Tecnici, analitici, ...

Cookies



Prima richiesta

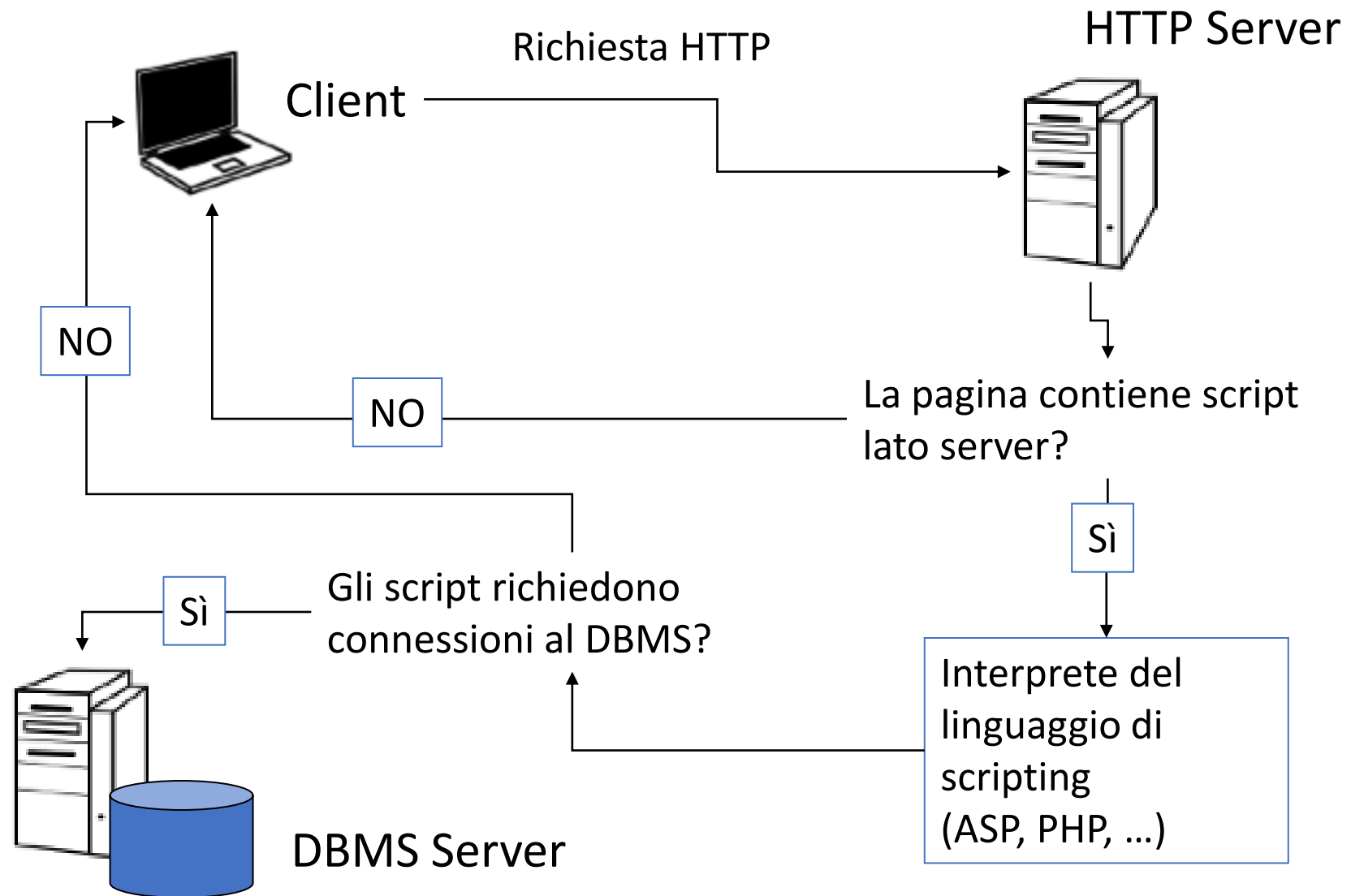
Successive richieste



Web statico e web dinamico

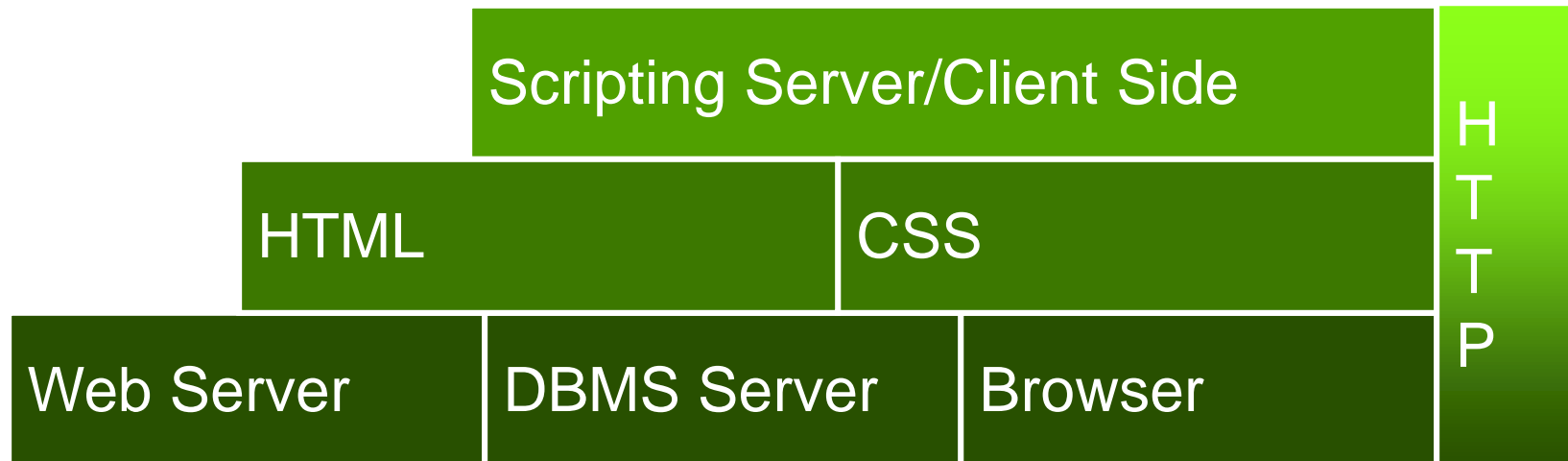
- I siti Web possono essere di natura statica o dinamica:
 - **Statica:** l'interazione con l'utente non prevede forme di elaborazione e interscambio di informazioni
 - **Dinamica:** con la richiesta di pagine, l'utente invia informazioni che richiedono al server una fase di elaborazione e modificano il contenuto e la forma degli oggetti restituiti
- **Web dinamico:**
 - Le pagine web non contengono solo istruzioni HTML per il browser ma anche istruzioni in linguaggio di programmazione
 - Tali istruzioni vengono eseguite (nel caso di script lato server) presso il server HTTP
 - Tali istruzioni consentono anche l'accesso alle risorse lato server, tra le quali sono fondamentali i DBMS

Web statico e web dinamico



Gli attori del web

THE WEB ACTORS



The background is a solid dark blue. A large, lighter blue semi-circle is positioned on the right side, with its flat edge facing left. A thin, vertical, medium-blue line runs from the top to the bottom of the frame, passing through the center of the semi-circle.

HTML

Linguaggio di markup

- Linguaggio di markup modellato per rendere esplicita una particolare interpretazione di un testo
 - Per esempio, tutte quelle aggiunte al testo scritto che permettono di renderlo più fruibile
 - Oltre a rendere il testo più leggibile, il markup permette anche di specificare ulteriori usi del testo
- Con il markup per sistemi informatici, specifichiamo le modalità esatte di utilizzo del testo nel sistema stesso
- I linguaggi di markup sono i linguaggi più opportuni per strutturare e marcare i documenti in maniera indipendente dall'applicazione, favorendo la riusabilità, la flessibilità e la apertura ad applicazioni complesse

HyperText Markup Language (HTML)

- HTML linguaggio per la creazione di pagine Web
 - Standard per la rappresentazione
 - Descrive la struttura della pagina
 - Non è un linguaggio di programmazione, è un linguaggio di markup
 - Descrive dati e regole su come mostrarli
 - Poche e semplici regole sintattiche
- File di testo

Browser - 1

- Programma per la navigazione nel Web
- Richiede risorse attraverso il web (all'indirizzo richiesto) e le mostra nella finestra
- Legge e interpreta i documenti HTML, CSS, JavaScript, immagini sulla base di specifiche e standard
- Browser: Internet Explorer, Mozilla Firefox, Chrome, Opera, Safari

Browser - 2

- Contiene tool per gli sviluppatori
 - Debug pagina
 - Analisi stili e sorgenti dei file
 - Visualizzazione HTTP request e response
 - Accesso a cookie e storage
- Molto altro...

Scrivere una pagina Web

- Gli editor HTML sono strumenti che permettono la scrittura di codice HTML (e quindi di pagine HTML)
 - Creazione file .html o .htm
- In generale, qualsiasi editor testuale può fungere da editor HTML. Tuttavia, esistono strumenti specifici
- Si distinguono tre tipologie di editor
 - Editor testuali: Permettono solo la scrittura di codice HTML
 - Editor wysiwyg: Aggiungono scrittura visuale e drag & drop
 - Suite avanzate: Aggiungono debugging

Validazione di una pagina Web

- Il W3C (World Wide Web Consortium) definisce e cura i linguaggi e gli standard per il Web
- W3C mette a disposizione un validatore online che consente di verificare se una pagina Web è ben scritta (cioè è formalmente corretta) rispetto agli standard (HTML e CSS)
 - <http://validator.w3.org/>

Struttura della pagina HTML

HTML - Tag

- Tag sono marcatori che identificano porzioni di testo
- Permettono la personalizzazione della pagina, e di identificare e processare le porzioni di testo contrassegnate da una certa marcatura
- Nome tag = nome funzione

Struttura tag

- Tag contenuto tra parentesi angolari
- Tag di inizio, tag di fine
- Contenuto del tag
- `<tag attributi>contenuto</tag>`

Esempio tag

- Tag con contenuto
 - `testo`
- Tag senza contenuto
 - ``

Struttura della pagina - 1

- Riga di intestazione

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01  
Transitional//IT">
```

- Indichiamo le specifiche W3C adottate
- Dice al browser quale linguaggio è stato usato per il rendering

Struttura della pagina - 2

- Il tag `<html>` indica l'inizio di una pagina HTML
- Tutto ciò compreso all'interno del tag `<html>` è il codice HTML

`<html>`

... codice HTML ...

`</html>`

Struttura della pagina - 3

- Documento HTML diviso in due
 - Testa
 - Corpo

Testa

- Contiene informazioni utili per la gestione della pagina
 - Ad es., il titolo che apparirà sui motori di ricerca e sui tab del browser ed altri
- Descrive come il documento deve essere letto e interpretato
- Contiene meta-tag, script, stili
- Racchiuso tra il tag <head>

Corpo

- Contiene il documento vero e proprio
- Contiene tutti i tag per la realizzazione del sito Web (ad esempio)
- Racchiuso tra il tag <body>

Esempio

- Scrivere una pagina HTML con i tag essenziali (html, head, body) inserendo una stringa all'interno del corpo
- primo.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01  
Transitional//IT">
```

```
<html>
```

```
<head>
```

```
</head>
```

```
<body>
```

```
Hello World!
```

```
</body>
```

```
</html>
```

Indentazione

- Buona norma utilizzare caratteri di tabulazione
- Aumenta leggibilità
- Diminuisce tempi di modifica

Indentazione - esempio

```
<html>  
  <head>  
  </head>  
  <body>  
    Hello World!  
  </body>  
</html>
```

```
<html><head></head>  <body>  
  Hello World!</body></html>
```

Annidamento

- Caratteristica HTML: tag annidati

<tag1 attributi>

contenuto1

<tag2 attributi>

contenuto2

</tag2>

</tag1>

- Formattazioni successive

Annidamento

- Esempio

```
<font color="blue">
```

```
    Hello
```

```
        <font color="red">
```

```
            World!
```

```
        </font>
```

```
    </font>
```

- Hello World!

Commenti

- Aumenta la leggibilità del codice
- Buona norma commentare le parti significative
- Permettono, insieme all'indentazione, di orientarsi all'interno di un grosso documento

Esempio - Commenti

- `<!--` = inizio commento
- `-->` = fine commento
- `<!-- questo è un commento -->`

Maiuscolo o minuscolo

- Case unsensitive
 - `testo`
 - `testo`
- Consigliabile usare carattere minuscolo

Altre caratteristiche

- Non è sensibile agli spazi

```
<font color="blue">Hello   World   !           </font>
```

- Non è sensibile alle linee vuote

```
<font color="blue">Hello
```

```
World !
```

```
</font>
```

- Stringa visualizzata: Hello World !

Testa

- Contiene informazioni/metadati relativi al documento
 - Ad esempio, titolo, keyword, altri dati non considerati contenuto del documento
- Browser di solito non presentano elementi che compaiono nella testa (HEAD)
- Rendono disponibili queste informazioni attraverso altri meccanismi

Testa: elemento title

- Elemento <title>
- Indica il titolo della pagina
- Viene visualizzato in alto a sinistra nei browser più datati o nella scheda nei browser più moderni

Metadati

- Elemento **<meta>**
 - Definisce informazioni relative alla pagina piuttosto che contenuto vero e proprio
- Contiene tre attributi principali
 - **Name, content, http-equiv**
 - **Name** contiene il nome della proprietà
 - **Author, Description, Copyright, Generator, Language, Keywords**
 - **Content** contiene il valore della proprietà
 - **http-equiv** specifica caratteristiche dell'HTTP response header
 - Esempio: `<meta http-equiv="refresh" content =5>`

Esempi

<META NAME="author" CONTENT="Valerio">

<META NAME="description" CONTENT="...">

<META NAME="copyright" CONTENT="...">

<META NAME="keyword" CONTENT="...">

Meta Tag - Keywords

- Indica alcuni informazioni sul contenuto del sito
- Separate da virgola, punto e virgola o spazio
- Sono chiavi di ricerca

Corpo (<body>)

Corpo

- Il corpo di un documento contiene il contenuto vero e proprio
- Il browser presenta il contenuto in diversi modi
 - Browser visuali: tela su cui disegnare testi, immagini, colori...
 - Agenti audio: lo stesso contenuto è parlato
- Contiene tutte le informazioni per la visualizzazione della pagina
 - Siccome gli style sheet sono il modo suggerito per specificare come verrà presentato un documento gli attributi del BODY sono stati deprecati

Elementi inline vs block

- Gli elementi HTML nel body possono essere "block-level" o "inline" ("text level")
- Content model
 - Elementi block-level possono contenere elementi inline e altri elementi block-level
 - Elementi inline contengono solo dati e altri elementi inline
- Formatting
 - Elementi block-level sono formattati in maniera differente dagli elementi inline
 - Elementi block-level iniziano su una linea nuova, elementi inline no
 - Elementi block-level occupano la larghezza massima disponibile, elementi inline solo la larghezza necessaria

Testo – Tag predefiniti

- Tag-contenitori di testo (elementi block-level, unica eccezione ``)
 - `<h1>`, `<h2>` ... (h = heading)
 - `<p>`
 - ``
 - `<div>`

Titoli - Heading

- Sono previste sei grandezze predefinite per i titoli
- Si va da <h1> che è la grandezza maggiore fino ad <h6> che è la grandezza minore
- Sintassi: <h1>Titolo</h1>

Esempio

- `<h1>Titolo1</h1>`
- `<h2>Titolo2</h2>`
- `<h3>Titolo3</h3>`
- `<h4>Titolo4</h4>`
- `<h5>Titolo5</h5>`
- `<h6>Titolo6</h6>`

Paragrafo

- `<p>`
- Unità base di suddivisione del testo
- Sintassi: `<p>paragrafo</p>`
- `<p>` lascia una riga vuota prima e dopo il testo

Blocco testo

- `<div>` è un blocco contenitore
- Blocco testo va a capo ma non lascia righe di spazio
- Sintassi: `<div>blocco 1</div>`
- Non impone rappresentazioni di default (a parte il fatto di essere un elemento block-level)

Contenitore

- ``
- Contenitore generico
- Può essere annidato
- Elemento inline (continua sulla stessa riga del tag che lo contiene (ad es. `<div>`))
 - Non va a capo
- Non impone rappresentazioni di default (a parte il fatto di essere un elemento inline)

Esempio

- Pagina contenente un titolo, un paragrafo, un blocco di testo e un contenitore

...

```
<h2>Titolo della pagina</h2>
```

```
<p>paragrafo 1</p>
```

```
<p>paragrafo 2</p>
```

```
<div>blocco 1</div>
```

```
<div>blocco 2
```

```
  <span>contenitore 1</span>
```

```
  <span>contenitore 2</span>
```

```
</div>
```

Struttura della pagina con tag <div>



È uno tra i tag più utilizzati nella creazione di pagine Web, soprattutto nella creazione di layout



Fornisce un vero e proprio elemento strutturale della pagina



Suddivide gli spazi in zone per progettare il sito in modo semplice e dettagliato



Molto utile quando usato insieme a fogli di stile

Struttura della pagina con tag <div>

- Facciamo subito un esempio: immaginando di dover costruire un sito pensiamo a come esso debba essere strutturato, tipicamente abbiamo
 - Un contenitore (container)
 - Una parte alta (header)
 - Un corpo centrale (middle)
 - Un menù (navigation)
 - Una parte bassa (footer)
 - Spesso anche un pannello laterale (sidebar)

Struttura della pagina con tag <div>

```
<div id="container">  
  <div id="header">  
    <div id="navigation"></div><!--#navigation-->  
  </div><!--#header-->  
  <div id="main"></div><!--#main-->  
  <div id="sidebar"></div><!--#sidebar-->  
  <div id="footer"></div><!--#footer-->  
</div><!--#container-->
```

Struttura della pagina con tag <div>



Testo - Stili

- Due tipi di stili
 - Fisici: definiscono lo stile grafico
 - Logici: forniscono informazioni sul ruolo svolto dal testo
 - Opzionale: impongono uno stile grafico

Stili fisici - 1

- `` bold: formatta testo in **grassetto**
- `<i>` italic: formatta il testo in *corsivo*
- `<u>` underline: sottolinea il testo
- `<strike>`: testo barrato (usato per correzioni)
- Tutti elementi inline

Stili logici - esempi

- `<abbr>` abbreviazione: non comporta nessun cambiamento grafico
- `<address>` indirizzo, `<cite>` citazioni: testo in corsivo
- `<samp>` esempio: testo a spaziatura fissa

Testo - Font

- Font = colore, dimensione e tipo di carattere
- Carattere predefinito Times new roman
 - Poco leggibile
 - Meglio Verdana, Arial o Helvetica

Esempio

- `testo`
- E' buona norma:
 - Utilizzare caratteri sicuri (sicuramente visualizzabili)
 - Non indicare un solo carattere
 - Solo come ultima spiaggia sarà usato il Times

Dimensione del testo

- Attributo size del tag font
- Due modi per impostare la dimensione
 - Valore tra 1 e 7
 - `Dimensione 3`
 - Valori relativi alla dimensione base del font
 - `Dimensione +2`
 - Dimensione del font di base
 - Tag `<basefont>` permette di cambiare il font di base

Elenchi

- Elenchi ordinati
- Elenchi non ordinati
- Elenchi di definizioni

Sintassi

<elenco>

<elemento> primo elemento

<elemento> secondo elemento

</elenco>

- Chiusura elemento opzionale
- Sintassi degli elenchi di definizione leggermente diversa

Elenchi ordinati

- Numerazione degli elementi
- Numerazione progressiva
- Tag elenco ordinato `` (ordered list)
- Tag elemento `` (list item)

Esempio elenchi ordinati

```
<div>Titolo</div>
```

```
<ol>
```

```
  <li>Primo elem
```

```
  <li>Secondo elem
```

```
    <ol>
```

```
      <li> Primo sub-elem secondo elem
```

```
      <li> Secondo sub-elem secondo elem
```

```
    </ol>
```

```
</ol>
```

Elenchi non ordinati

- Nessuna numerazione
- Elenchi puntati
- Tabella tag `` (unordered list)

Esempio elenchi non ordinati

 Primo elemento

 Secondo elemento

 Primo elemento

 Secondo elemento

Iperestualità (Link)

Introduzione

- Testi vs ipertesti
- Ipertesti caratteristica che ha reso grande il web
- Link = ponte tra un testo e un altro

Link

- I link sono formati da due componenti
 - Il contenuto (testo o immagine) che nasconde il collegamento
 - La risorsa puntata
- Sintassi
 - Clicca `` qui `` per visualizzare il collegamento.
 - La testa del link è *qui*
 - La coda è *indirizzo*
- Coda
 - Pagina HTML, immagine, documenti, altri file

Destinazioni

- Pagina HTML: aperta nel browser
- Immagine (.gif, .jpeg): aperta nel browser
- Altri documenti (.doc, .pdf):
 - Visualizzata nel browser
 - Richiesta di salvataggio
- Altri file (.zip, .exe) richiesta di salvataggio

Esempio

- Link a pagina HTML
 - `Clicca qui`
- Link a documenti
 - `Clicca qui`
- Link ad immagini
 - `Clicca qui`
- Link ad altri file
 - `Clicca qui`

Caratteristiche link

- Diversi stati
 - Link a riposo
 - Colore blu #0000FF
 - Link visitato
 - Colore violetto
 - Link attivo
 - Passaggio da una pagina all'altra
 - Utile nel passato
 - Link al passaggio del mouse
 - Solo con fogli di stile

Percorsi assoluti

- Leggi l'esempio ` qui `
- Nel caso di file sull'hard disk
- Leggi l'esempio ` qui `

Percorsi relativi

- `Clicca qui`
 - Indica al browser di cercare il file nella stessa directory
- `Clicca qui`
 - Indica al browser di cercare il file nella sottodirectory sottodir
- `Clicca qui`
 - .. Indica al browser di cercare il file nella directory padre

Considerazioni

- Nomi dei file
 - Evitare gli spazi (mio_file.html)
 - Ricordarsi che maiuscole e minuscole fanno la differenza
- `` non funziona quando caricate la pagina sul vostro sito
- Allora diventerà
 - ``

Link interni o ancore

- Permette di creare un indice
- Ancora
 - ` ancora `
- Riferimento all'ancora
 - ` vai all'ancora `
- Riferimento ad inizio pagina
 - ` torna su `

Tabelle (table)

Tabelle

- Definizione tramite l'elemento `<table>`
- Ogni riga definita con il tag `<tr>` contenuto in `table`
- Ogni cella definita con il tag `<td>` contenuto in `<tr>`
- Una riga può essere anche divisa in table heading usando l'elemento `<th>`
 - Di solito mostrati centrati e in grassetto
- Elemento `<caption>` per definire la caption della tabella

Esempio

```
<table style="width:100%">
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
  <tr>
    <td>John</td>
    <td>Doe</td>
    <td>80</td>
  </tr>
</table>
```

Bordi e colonne/righe multiple

- Attributo border
 - Se non specificato nessun bordo
 - `<table border="1" style="width:100%">`
- Attributi rowspan e colspan
 - `<td rowspan="2">`
 - `<td colspan="2">`

The background is a solid dark blue. A large, semi-transparent circle of a slightly lighter blue shade is positioned on the right side, centered vertically. A thin, vertical line of a medium blue shade runs through the center of the circle, extending from the top to the bottom of the frame.

Form

Introduzione

- Form usate per collezionare input utente
- Basato sull'elemento `<form>`
- Specifica diversi elementi
 - input, checkbox, radio button, submit button...

Elemento input

- Elemento più importante
- Diverse varianti che dipendono dall'attributo type
 - Text: definisce input testo normale
 - Radio: definisce un input radio button (per selezionare una tra diverse scelte)
 - Submit: definisce un bottone submit (per sottomettere la form)

Text input

- `<input type="text">`
 - Definisce un campo testo di una linea
 - Esempiof1.html
 - `<input type="password">` un tipo speciale di campo testo dove il valore inserito viene nascosto

```
<form>  
First name:<br>  
<input type="text" name="firstname">  
<br>  
Last name:<br>  
<input type="text" name="lastname">  
</form>
```

Text area

- `<textarea>`
 - Definisce un campo testo a linee multiple
 - Esempiof1.html

```
<textarea name="message" rows="10" cols="30">  
The cat was playing in the garden.  
</textarea>
```

Radio button input

- `<input type="radio">`
 - Definisce un radio button che permette **una** scelta tra diverse opzioni
 - Esempiof2.html

```
<form>  
  <input type="radio" name="sex" value="male"  
checked>Male  
  <br>  
  <input type="radio" name="sex" value="female">Female  
</form>
```


Checkbox input

- `<input type="checkbox">`
 - Definisce una serie di checkbox che permettono una scelta **multipla** tra diverse opzioni
 - Esempiof2.html

```
<form>
```

```
  <input type="checkbox" name="vehicle1" value="Bike"> I  
  have a bike
```

```
  <br>
```

```
  <input type="checkbox" name="vehicle2" value="Car"> I  
  have a car
```

```
</form>
```

Drop-down list

- `<select>`
 - Definisce un menù a tendina
 - Elemento `<option>` definisce le possibili opzioni
 - `<option value="fiat" selected>Fiat</option>`
 - Esempiof3.html

```
<select name="cars">  
  <option value="volvo">Volvo</option>  
  <option value="saab">Saab</option>  
  <option value="fiat">Fiat</option>  
  <option value="audi">Audi</option>  
</select>
```

Submit button

- `<input type="submit">`
 - Definisce un bottone per sottomettere la form ad un handler
 - Handler è una pagina lato server che processa i dati della form
 - Handler è definito con l'attributo action
 - Esempiof4.html

```
<form action="action_page.php">  
  First name:<br>  
  <input type="text" name="firstname" value="Mickey">  
  <br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse">  
  <br><br>  
  <input type="submit" value="Submit">  
</form>
```

- Permette anche di eseguire semplici azioni
 - `<button type="button" onclick="alert('Hello World!')">Click Me!</button>`

Action attribute

- Definisce le azioni che devono essere fatte quando la form è inviata
- Sottomissione fatta solitamente con il bottone submit
- La pagina viene inviata a una pagina web lato server
 - Ad esempio uno script lato server gestisce la form inviata
 - `<form action="action_page.php">`
 - Se l'attributo action è omissso si considera la pagina corrente di default

Method attribute

- Specifica l'operazione HTTP (GET o POST) da usare per inviare la form
 - `<form action="action_page.php" method="get">`
 - `<form action="action_page.php" method="post">`
- GET (default) usata per invio di form passive (ad es., search engine query) e senza informazioni sensibili
- I dati inviati tramite la form sono visibili nell'indirizzo
 - `action_page.php?firstname=Mickey&lastname=Mouse`
- GET è ottima per piccola quantità di dati

Method attribute

- Specifica l'operazione HTTP (GET o POST) da usare per inviare la form
 - `<form action="action_page.php" method="get">`
 - `<form action="action_page.php" method="post">`
- GET (default) usata per invio di form che aggiornano dati
- POST fornisce maggiore sicurezza perchè i dati non sono visibili nell'indirizzo (ad es., password)

Name attribute

- Ogni campo input deve avere un nome

```
<form action="action_page.php">  
  First name:<br>  
  <input type="text" name="firstname" value="Mickey">  
  <br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse">  
  <br><br>  
  <input type="submit" value="Submit">  
</form>
```

Altri attributi

- Attributo value specifica il valore iniziale
 - `<input type="text" name="firstname" value="John">`
- Attributo readonly specifica che il valore non può essere cambiato
 - `<input type="text" name="firstname" value="John" readonly>`
- Attributo disabled specifica un campo disabilitato e non utilizzabile (non viene inviato)
 - `<input type="text" name="firstname" value="John" disabled>`