Laboratorio di Comunicazione mediante Calcolatore – A.A. 2020/2021
03 - HTML, e LaTeX

Leonardo Robol <leonardo.robol@unipi.it>,
Sergio Steffé <steffe@cs.dm.unipi.it>

16 Novembre 2020

Prossimi laboratori:

- 2 su LaTeX (questa e prossima settimana) → Test 2
- 1 su pagine web (30/11 e 03/12) → Test 3

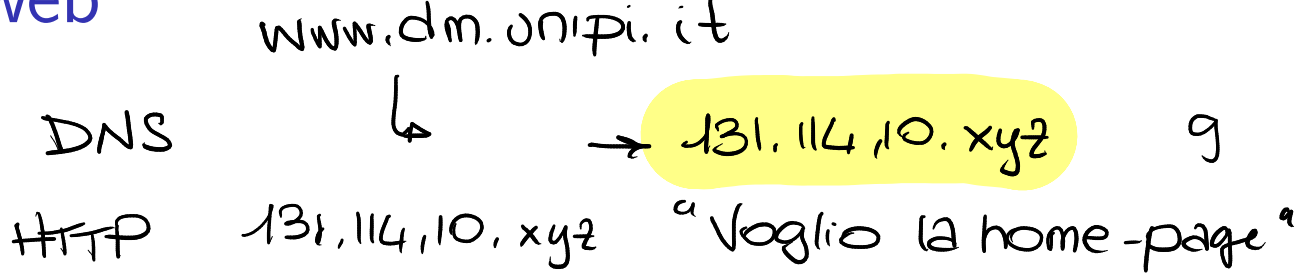
Reminder:

- Primo test entro il 20/11 (su Moodle)
- Check presenze su Moodle

Registrazione crediti:

- 7 presenze su 10.
- "sufficienza" nei test

Pagine web



- Quando visitiamo una pagina web, il PC deve richiederne il contenuto attraverso la rete.
- I primi passi usano le tecnologie che abbiamo già discusso (DNS, TCP/IP, . . .).
- Nei passi seguenti è necessario che il server e il client si accordino su come interpretare i dati, ovvero come descrivere il contenuto della pagina.

WWW e HTML

- Generalmente, HTTP (o la sua versione criptata HTTPS) sono i protocolli più usati per navigare in Internet.
- Altri protocolli (gopher, e in qualche misura anche FTP), sono quasi in disuso.

SFTP

Storia del WWW

- Nel 1991, Tim Berners Lee, lavorando al CERN, creò un sistema che permetteva di ospitare delle pagine con collegamenti (ipertesti);
- Il sistema era un'evoluzione di quanto aveva sviluppato negli anni prima; fino ad allora, internet veniva utilizzata per le email ma non esisteva un modo semplice di scambiare file.

Storia del WWW

- Nel 1991, Tim Berners Lee, lavorando al CERN, creò un sistema che permetteva di ospitare delle pagine con collegamenti (ipertesti);
- Il sistema era un'evoluzione di quanto aveva sviluppato negli anni prima; fino ad allora, internet veniva utilizzata per le email ma non esisteva un modo semplice di scambiare file.
- Con il tempo, sono nati dei browser, è stata aggiunta una componente "grafica" alle pagine, e la tecnologia si è evoluta. Ma il protocollo di trasmissione ha ricevuto pochi cambiamenti.

Collegamenti ipertestuali

- La grossa novità del WWW era la possibilità di passare da una pagina all'altra tramite "collegamenti".
- Questo permetteva di catalogare facilmente i contenuti!
- Un enorme passo in avanti è stato fatto poi dai ~~motori di~~ ricerca, che riescono a determinare quale pagina è importante e quale no (come? è un problema di autovalori e autovettori).

▷ Google n pag. ⇝ ▷ matrice n×n ⇒ V punteggi di importanza

Conversazione HTTP

- `GET / HTTP/1.0`: “vorrei vedere la home-page”
- Il server risponde con il codice HTML; questo potrebbe contenere link ad altri file da caricare (ad esempio, immagini).
- Il client richiede i file ausiliari:
  - `GET /images/image1.png HTTP/1.0`
  - `GET /images/image2.png HTTP/1.0`
  - `...`

Nei prossimi laboratori vedremo come creare una pagina web.

Pagine web "moderne"

W3C → HTML 5
→ Javascript

La maggior parte delle pagine che visitate non contengono solo codice HTML, che rappresenta contenuto "statico", ma anche codice Javascript, che permette di renderle dinamiche.

→ sintassi simil-C

- Esempio: Facebook, GMail, ...
- Questo apre la possibilità a problemi di sicurezza: il vostro computer esegue codice scaricato da un server, senza chiedervi conferma.
- Per questo, i browser utilizzano strategie particolare (e.g., sandboxing).

Server Proxy

A volte è preferibile non utilizzare una connessione diretta al server web, ma passare attravero un "proxy". Ad esempio, questo permette di accedere alla riviste scientifiche per cui l'università paga l'abbonamento:

- `ssh -D 8080 ssh1.dm.unipi.it`: Apertura di un proxy attraverso SSH;
- Configurare il proxy sul proprio PC (tipicamente Impostazioni $\rightarrow$ Rete $\rightarrow$ Proxy, oppure direttamente dal browser). Il tipo di protocollo si chiama SOCKS. $\rightarrow$ localhost , porta 8080
- Navigare su `mathscinet.ams.org`.

Browser

- Per visualizzare la pagine web, c'è bisogno di un browser.
- Potete scegliere il vostro preferito, Firefox, Chrome/Chromium, Edge, Opera, ...
- Linux dispone di alcuni comandi per scaricare dati:
  - `wget https://www.google.com` scarica la home page di Google sul vostro PC.
  - `curl https://www.google.com` effettua la stessa operazione; CURL ha molte opzioni, e permette anche di spedire dati, oltre che scaricarli.

lynx, links2

Privacy

Con il passare del tempo, e l'aumento della popolarità di Internet, si è cominciato a preoccuparsi della privacy degli utenti:

- Quante informazioni vengono condivise dal mio PC quando navigo in Internet?
- A che scopo possono essere utilizzate queste informazioni (ad esempio, per la pubblicità?).
- Posso negare il consenso?

Privacy in locale

Lo stesso problema si applica ai dati locali:

- Cosa succede se mi rubano il PC e/o riescono ad accederci da remoto?
- Esiste un sistema per prevenire il furto dei dati?
- Ce ne sono alcuni: cifratura della home / container / "trusted-platform", con diversi pro e contro.

Identificazione in rete

- Quali dati possono essere utilizzati quando navigo?
- Ci sono stati diversi tentativi di legiferare su questo argomento da parte dell'UE (Cookies, GDPR).
- Se vogliamo essere (ragionevolemente) sicuri che i nostri dati non siano utilizzati per la pubblicità – possiamo aprire una scheda in incognito.
- Alcuni dati però sono ancora disponibili: IP, geolocation, e se mi loggo perdo l'anonimato.

Come creare una pagina HTML?

Il linguaggio HTML è un linguaggio di markup. Una pagina base ha questo aspetto:

```
<!DOCTYPE html>
<html>
  <head>
    <title>La mia pagina personale</title>
  </head>

  <body>
    <h1>Home page</h1>
    <p>Benvenuti sulla mia pagina ...</p>
  </body>
</html>
```

- Tag: hanno parentesi angolate (, <head>) e si devono "chiudere" (, </head>). Possono contenere dell'altro HTML al loro interno, e/o avere attributi.

Aggiungiamo degli attributi

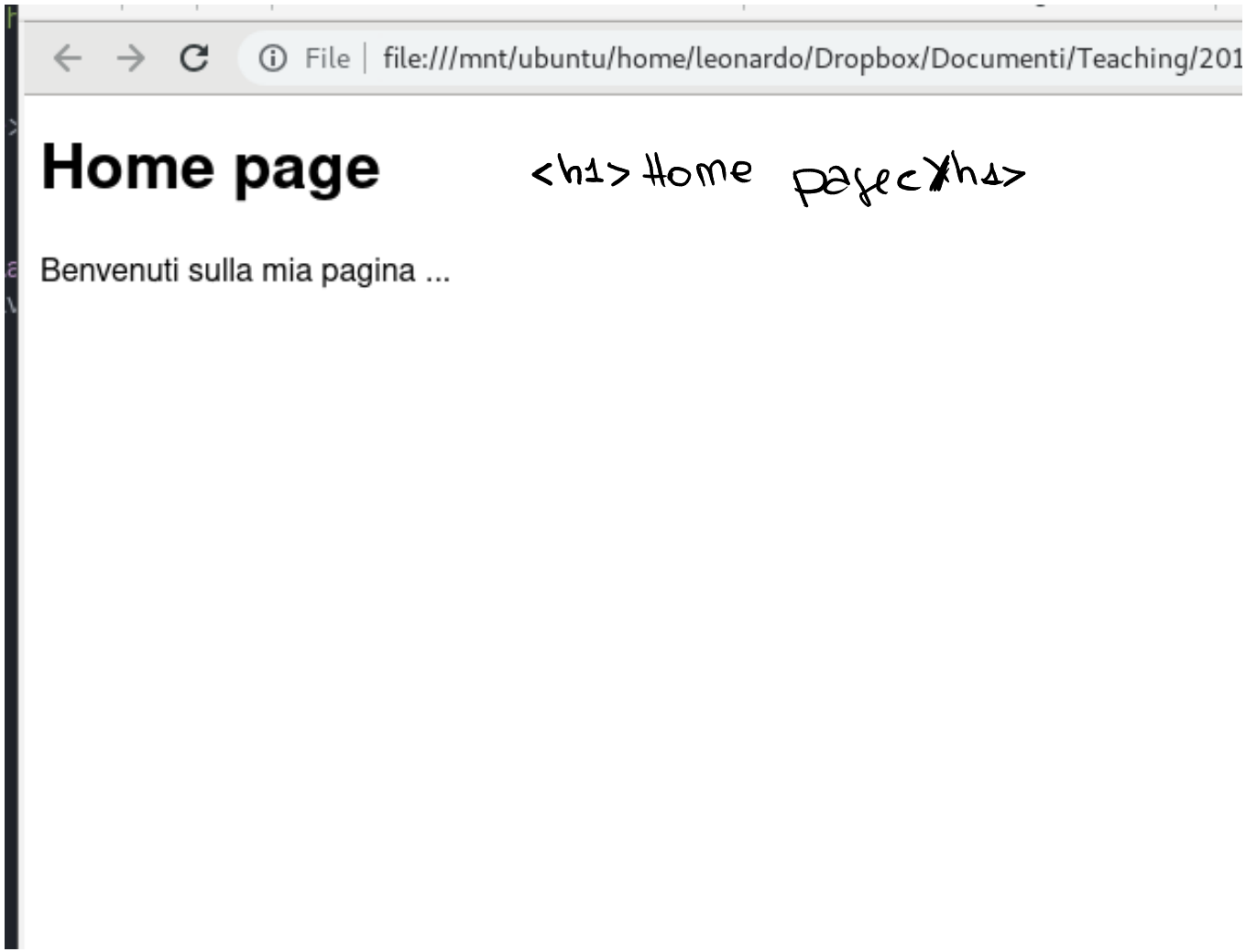
```
<!DOCTYPE html>
<html>
  <head>
    <title>La mia pagina personale</title>
  </head>

  <body>
    <h1 class="titolo" id="titolo-principale">Home page</h1>
    <p>Benvenuti sulla mia pagina ...</p>
  </body>
</html>
```

- `<h1>, ..., <h6>` sono tag per i titoli.
- `<p>` corrisponde ad un paragrafo.
- Ogni tag può avere una classe, ed un id. L'ID deve essere unico! Ci sono molti altri attributi possibili.

Alcuni esempi

- `<a href="http://www.google.com">Open google.com</a>`: un collegamento a Google (tag "anchor").
- `<img src="images/cat.png" />`: Un'immagine di un gattino: notate la sintassi finale `/>`: questo è equivalente ad aprire e chiudere un tag, ovvero a scrivere `<img src="images/cat.png"></img>`.
- Molti altri tag che vedremo in laboratorio.

Uno sguardo alla nostra pagina

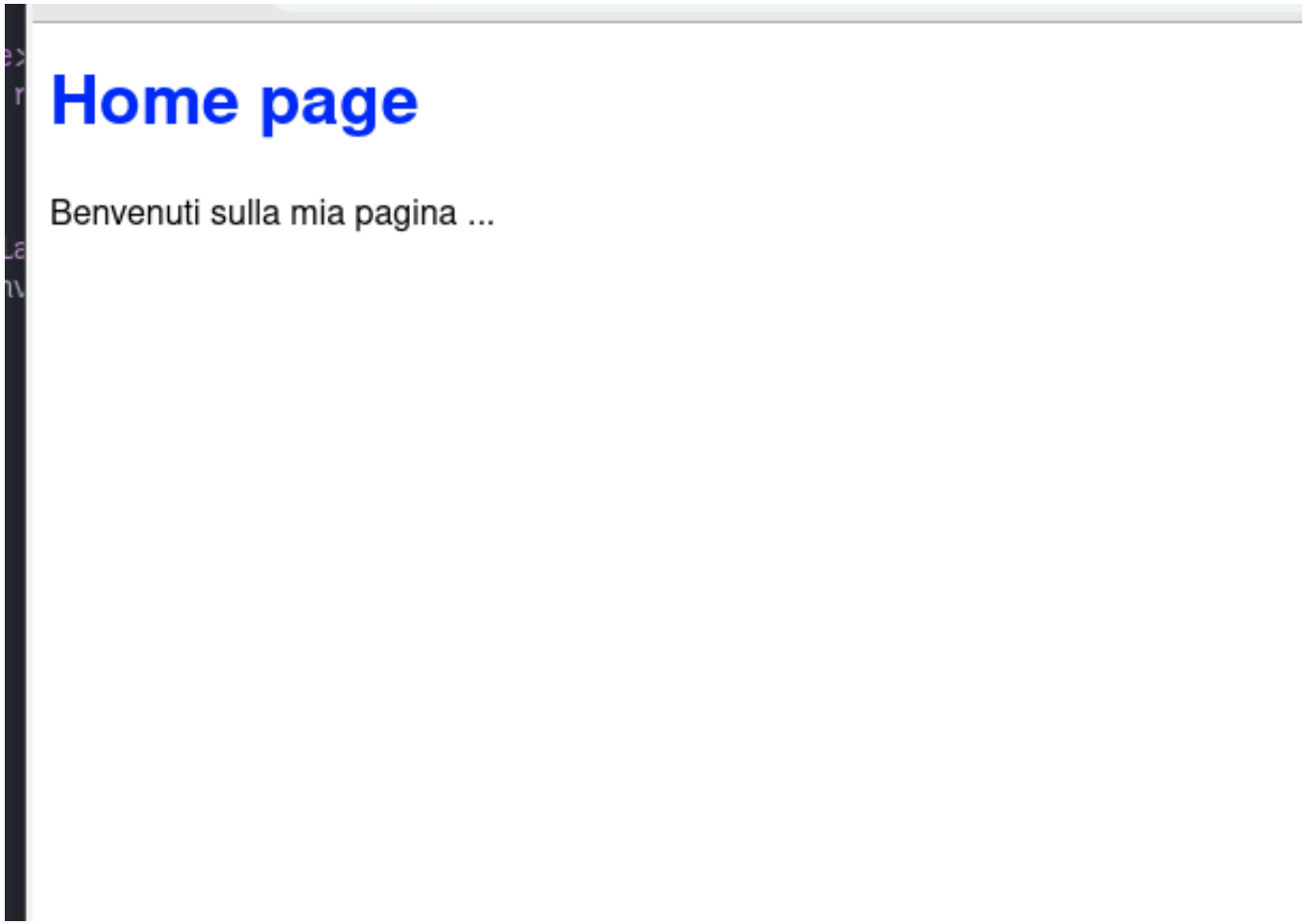


CSS

- La pagina funziona, ma è decisamente triste e spoglia.
- Per ovviare a questo problema, possiamo usare i fogli di stile (Cascading Style Sheets).
- Possiamo usare classi e id per specificare l'aspetto di varie parti della pagine, oppure cambiare l'aspetto di tutti i tag di un certo tipo.
- Il linguaggio dei CSS è diverso dall'HTML:

```
/* Il selettore .titolo matcha la class titolo */
.titolo {
    color: blue;
}
```

La nuova pagina



CSS

- Il risultato non è ancora entusiasmante, potremmo cambiare lo sfondo modificando il tag body – che rappresenta tutto il corbo del testo. Magari possiamo anche aggiungere un bordo al paragrafo.

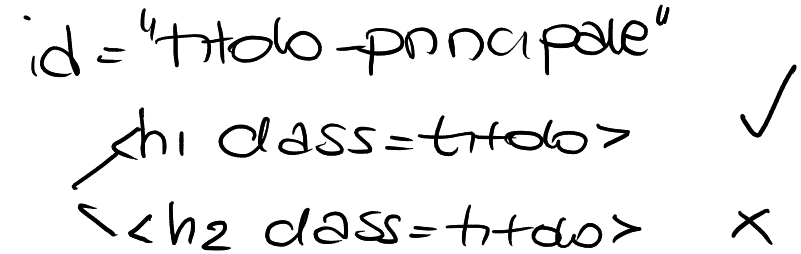
```
/* Il selettore .titolo matcha la class titolo */
.titolo {
    color: blue;
}
/* Si possono specificare anche nomi di tag */
body {
    background-color: red;
}
p {
    border: 1px solid green;
}
```

La nuova pagina

CSS

- Le possibilità sono infinite: bordi, margini, padding, caratteri, animazioni, ...
- È possibile selezionare anche elementi utilizzando il loro id, ad esempio `<h1 id="titolo-principale">Home page</h1>` si può selezionare con

```
#titolo-principale {
    ...
}
```



- Altre possibilità per i più esperti: selezionare tag che sono all'interno di certi altri tag, oppure solo in "condizioni" particolari, ad esempio solo quando il mouse è sopra di loro:

```
/* Colore rosso solo quando il mouse ci passa sopra */
#titolo-principale:hover {
    color: red;
}
```

TEX e LATEX

- Donald Knuth è l'autore di "The Art of Computer programming".
- Quando, alla fine degli anni 70, ottenne le bozze della seconda ristampa, rimase inorridito dalla bassa qualità tipografica.
- . . . e decise che doveva esistere qualche metodo più intelligente di affrontare il problema.
- Così è nato il TEX.

TEX e LATEX

- Donald Knuth è l'autore di "The Art of Computer programming".
- Quando, alla fine degli anni 70, ottenne le bozze della seconda ristampa, rimase inorridito dalla bassa qualità tipografica.
- ...e decise che doveva esistere qualche metodo più intelligente di affrontare il problema.
- Così è nato il TEX.

Ogni matematico ad un certo punto incontra il TEX – e certamente toccherà anche a voi quando dovrete scrivere la tesi oppure presentare un seminario.

Un po' di storia

- Knuth cominciò a lavorare al TEX nel 1977, progettando di finirlo nel seguente anno.
- In realtà, la versione finale fu rilasciata solo nel 1989 (TEX 3); da quel momento le nuove versioni hanno un numero che converge a π.
- Ad esempio, l'ultima release del TEX è la versione 3.14159265. Il numero di versione verrà settato a π — e lo sviluppo congelato — dopo la morte di Knuth.
- In maniera simile, il sistema di compilazione dei caratteri usati dal sistema TEX (MetaFont) ha raggiunto la versione 2, e da allora converge a $e = 2.71828\ldots$.

Funzionamento del TEX

Il TEX è un software di typesetting – ma anche un linguaggio di programmazione; dato un file in linguaggio TEX, compilarlo con il comando `tex` genera in output un file PDF.

- L'interprete TEX legge un file `esempio.tex`, tramite il comando `pdftex`.
- Valuta i vari comandi, e produce dei file ausiliari, insieme ad un file `esempio.pdf`.

Esempio minimale

Proviamo a compilare un documento contenente

```
Definiamo $y = x^2 - ab$.
\bye
```

e otteniamo

Definiamo $y = x^2 - ab$

Esempio minimale

Proviamo a compilare un documento contenente

```
Definiamo $y = x^2 - ab$.
\bye
```

e otteniamo

Definiamo $y = x^2 - ab$

Oppure possiamo usare comandi TEX, che cominciano con il simbolo \:

```
\[
  \frac{\partial u}{\partial t} =
    \int_0^1 u(x,t) \varphi(x) dx
\]
```

che produce

$$\frac{\partial u(x,t)}{\partial t} = \int_0^1 u(x,t)\varphi(x)dx$$

Macro

- In TEX è possibile definire delle macro, ovvero "comandi aggiuntivi".
- Queste si possono usare per estendere il TEX – e fornire funzionalità aggiuntive. Tipicamente queste funzionalità vengono raggruppate in pacchetti.
- Ad esempio, c'è un pacchetto per le lettere gotiche, un pacchetto per scrivere del codice, un pacchetto per le slide (Beamer, che ho usato per questi lucidi), un pacchetto per disegnare, per fare grafici, ecc.
- Il TEX è un linguaggio di programmazione, e come spesso succede andare a capo non è troppo diverso da inserire uno spazio; per andare a capo nel testo bisogna usare \\ o lasciare una riga vuota.

- Nell'uso di tutti i giorni, è necessario avere un set di regole preconfezionate per il layout di pagina – altrimenti è facile incorrere in errori tipografici.
- Per questa ragione, è stato sviluppato il LaTeX, un set di macro TeX che permette all'utente di descrivere la struttura del testo, senza preoccuparsi (troppo) della formattazione.
- Il LaTeX fornisce degli "ambienti".
- Impareremo anche questo a laboratorio.

Documento LaTeX di esempio

```
\documentclass{article}

\title{Il mio documento}
\author{Leonardo Robol}

\begin{document}
  \maketitle

  \section{Introduzione}
  Questo documento presenta la principali caratteristiche
  del \LaTeX; possiamo scrivere matematica in linea
  ($a = 2 \cdot b$), o in equazioni
  \begin{equation}
    \frac{1}{2\pi i} \int_{\partial \Gamma} f(z) dz = 0.
  \end{equation}
\end{document}
```

PREAMBOLO

Ambienti

La differenza essenziale fra TEX e LATEX è la presenza di ambienti, che si aprono con il comando `\begin` e si chiudono con `\end`:

- `\begin{document} ... \end{document}` contiene tutto il testo.
- `\begin{equation} ... \end{equation}` contiene un'equazione numerata.
- ...

Automatismi

LaTeX si occupa automaticamente di molte cose:

- La gestione delle sezioni, e i riferimenti; aggiusta anche la spaziatura come necessario secondo le regole tipografiche.
- La numerazione di equazioni, teoremi, ecc.
- La gestione della bibliografia.
- Il posizionamento ottimale di figure e tabelle.
- Liste puntate, numerate, ecc.

Possiamo usare i comandi `\label` e `\ref` per marcare delle sezioni / teoremi / equazioni, e per creare dei riferimenti.

Riferimenti

```
\begin{equation} \label{eq:uno}
  y = 2x
\end{equation}
Come visto in equazione~\eqref{eq:uno},
$y$ è il doppio di $x$.
```

Risulta nel seguente output:

$$y = 2x \tag{1}$$

Come visto in equazione (1), y è il doppio di x.

Lucidi

Generare lucidi è simile: il codice di questa slide è più o meno:

```
\begin{frame}{Lucidi}
  Generare lucidi è simile: il codice di questa slide è
  più o meno:

  [...] blocco di codice qui [...]

\end{frame}
```