Programmazione Web

URI, HTTP, HTML

- URI (Uniform Resource Identifier)
  - Come identifico una risorsa
- HTTP
  - Come comunicano client e server
- HTML (HyperText Markup Language)
  - Come scrivo una pagina web
- CSS
  - Che aspetto ha una pagina Web

Uniform Resource Identifiers (URI)

- Sistema di indirizzamento su Web
  - stringhe di caratteri ASCII che identificano le risorse disponibili su Web
- Standard IETF (RFC 2396)
  - Internet Engineering Task Force
- Uniform Resource Identifiers (URI)
  - Uniform Resource Locators (URL)
  - Uniform Resource Names (URN)

Uniform Resource Identifiers (URI)

- URL
  - la risorsa è fisicamente accessibile
  - la stringa descrive il metodo (primario) per accedere alla risorsa
- URN
  - la stringa non descrive il metodo d'accesso
  - la risorsa può non essere fisicamente accessibile (es: namespace)
- Ci concentreremo sugli URL

URI

- Forma generale
  - <protocollo>:<parte-dipendente-dal-protocollo>
- Principali protocolli
  - http
  - ftp
  - mailto
  - file

URI

- Esempi:
  - http://www.repubblica.it/sport/
  - ftp://ftp.unina.it/pub
  - mailto:fioravanti@unich.it
  - file:///d:/sites/index.html
  - gopher://spinaltap.micro.umn.edu/00/
  - news:comp.infosystems.www.servers.unix
  - telnet://melvyl.ucop.edu/

Indirizzi Web (URL)

- URL (Universal Resource Locator)

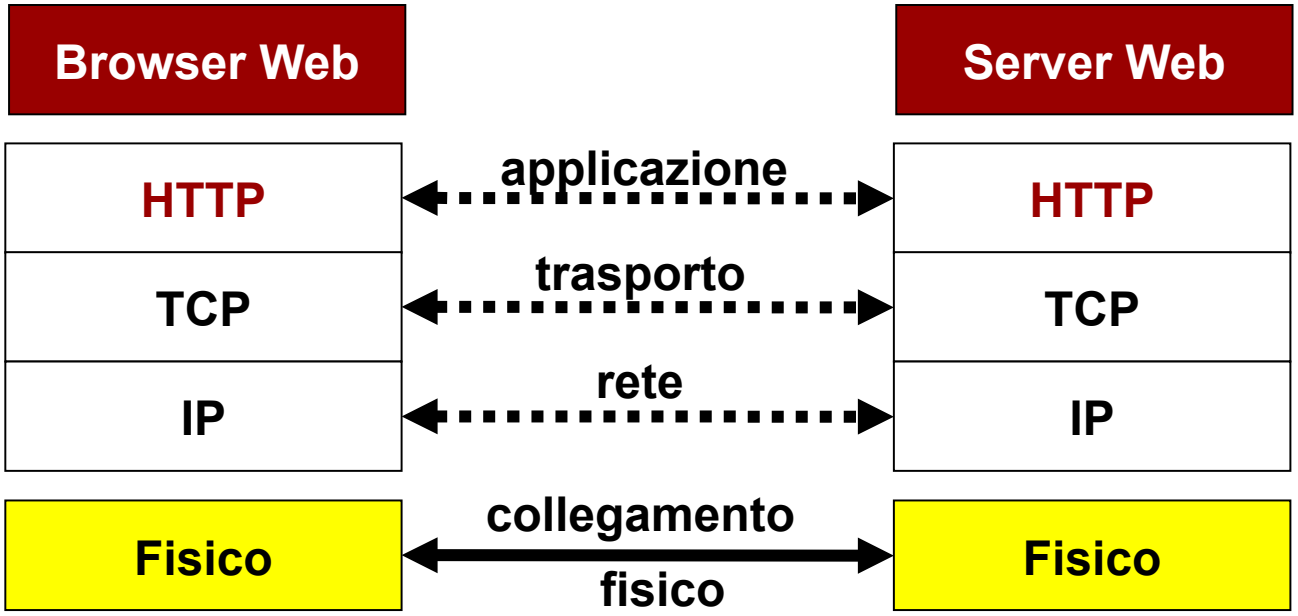
  http://www.sci.unich.it/~fioravan/index.html

  - http : il protocollo di comunicazione per il Web (Hyper Text Transfer Protocol)
  - www.sci.unich.it : l'indirizzo Internet della macchina server che il DNS (Domain Name Server) traduce in indirizzo IP oppure direttamente l'indirizzo IP
  - fioravan/index.html : l'indirizzo della risorsa (in questo caso un file HTML) relativo alla radice del Web server.

HTTP

HTTP 1.0

- Standard IETF (RFC 1945) ora superato da HTTP 1.1
- Protocollo di applicazione



HTTP 1.0: Transazioni

- Transazione HTTP
  - scambio di messaggi tra server e client
  - il client apre una connessione con il server
  - il client invia la richiesta sulla connessione
  - il server invia la risposta sulla connessione
  - la connessione viene chiusa

HTTP 1.0: Transazioni

- Caratteristiche del protocollo
  - una nuova connessione per ogni transazione
    - Migliorato in HTTP 1.1
  - privo di stato – nella transazione successiva non resta traccia di quanto avvenuto nelle transazioni precedenti
- Attenzione
  - la mancanza di stato influenza la scrittura delle applicazioni
    - Cookies, sessioni

HTTP 1.0: Transazioni

- Come nasce normalmente la richiesta
  - l'utente seleziona un URI nel browser
    es: http://www.sci.unich.it/~fioravan
  - l'URI può essere specificato esplicitamente
    es: nella barra degli indirizzi del browser
  - oppure può provenire da un collegamento ipertestuale selezionato dall'utente

HTTP 1.0: Transazioni in Dettaglio

- I Operazione
  - risoluzione del nome: il client utilizza il servizio DNS per risolvere il nome in num. IP
    es: www.sci.unich.it >> 192.167.92.35
- II Operazione
  - viene richiesta una connessione al numero IP e alla porta specificata
    es: 192.167.92.35:80

HTTP 1.0: Transazioni

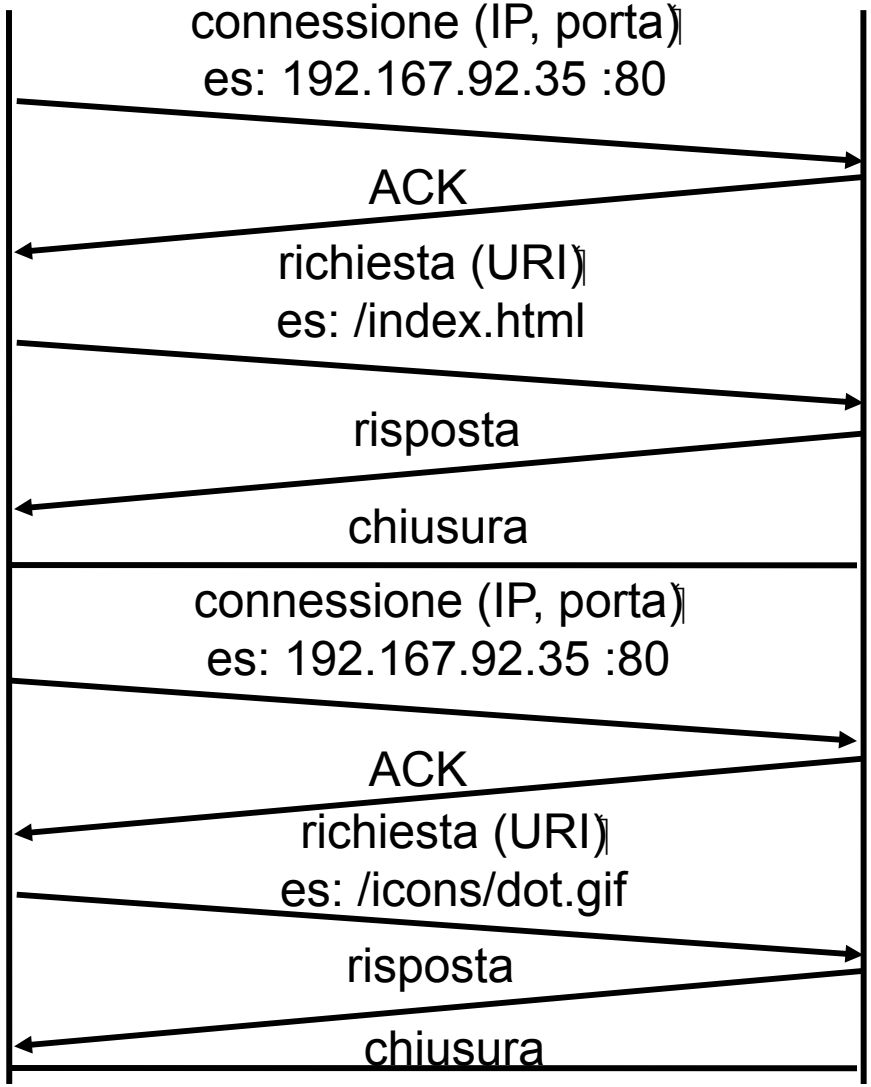
- III Operazione
  - ottenuta la connessione, il browser effettua una richiesta HTTP al server specificando il percorso e il nome della risorsa
    es: GET /fioravan/index.html HTTP/1.0
- IV Operazione
  - il server gestisce la richiesta e fornisce la risposta

HTTP 1.0: Transazioni

- Nota
  - le richieste HTTP sono raramente isolate
- Esempio
  - pagine HTML che contengono immagini
  - il codice HTML della pagina e le immagini sono risorse distinte, con URI distinti
  - viene richiesto il codice HTML
  - successivamente vengono richieste le immagini necessarie alla visualizzazione completa

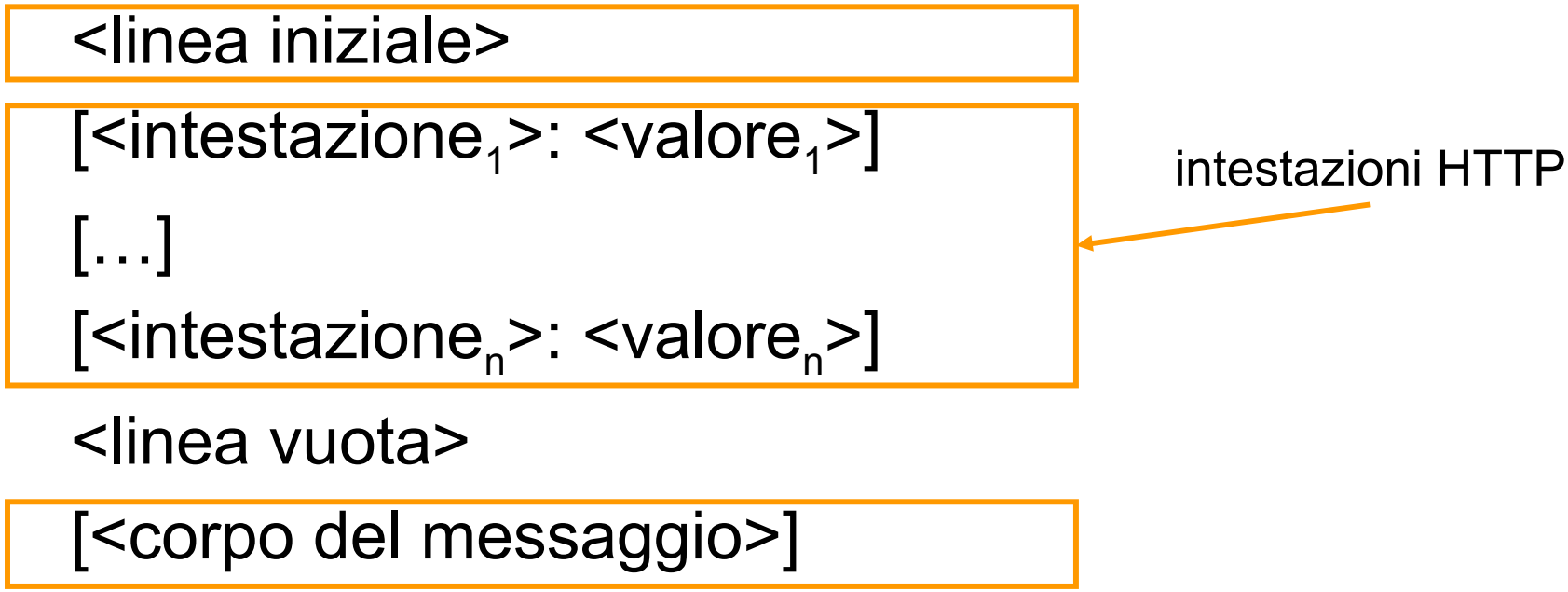
HTTP 1.0: Transazioni

browser



HTTP 1.0: Formato dei Messaggi

- Struttura generale dei messaggi
  - vale sia per la richiesta HTTP che per la risposta HTTP



HTTP 1.0: Formato dei Messaggi

- Linea iniziale
  - nella richiesta: contiene operazione e risorsa
  - nella risposta: contiene l'esito dell'operazione richiesta
- Corpo
  - nella richiesta è vuota o contiene la query
  - nella risposta contiene la risorsa
- Intestazioni
  - ne esistono varie

HTTP 1.0: Richiesta

- Linea iniziale della richiesta
  - <metodo> <URI> HTTP/1.0
- Metodo
  - GET: metodo ordinario per effettuare richieste specificando l'URI della risorsa
  - POST: metodo per effettuare richieste specificando l'URI ed una serie di dati e parametri **nel corpo** della richiesta
  - HEAD: variante di GET a scopo di controllo

HTTP 1.0: Richiesta

- Metodo GET
  - metodo molto comune
  - viene specificato l'URI della risorsa
  - eventuali parametri sono nell'URI come coppie parametro=valore
  - il corpo della richiesta è vuoto

  - GET /index.html HTTP/1.0
  - GET /didattica/index.html HTTP/1.0
  - GET /bollo.cgi?targa=AB123DE HTTP/1.0
  - GET /bollo.cgi?CF=19&alimentazione=benzina HTTP/1.0

HTTP 1.0: Richiesta

- Metodo POST
  - spesso utilizzato per inviare dati tramite moduli HTML (form)
  - viene specificato l'URI della risorsa senza parametri
  - parametri contenuti nel corpo del messaggio

  - POST /bollo.cgi HTTP/1.0
    (in questo caso i parametri sono nel corpo)

HTTP 1.0: Richiesta

- Metodo HEAD
  - variante di GET utilizzata principalmente a scopo di controllo (es: validità) e debugging
  - la richiesta è del tutto simile ad una GET
  - in risposta il server fornisce solo le intestazioni (ma non il corpo)
  - HEAD /index.html HTTP/1.0
  - HEAD /bollo.cgi?targa=AB123DE HTTP/1.0

HTTP 1.0: Richiesta

- Chi decide il metodo di richiesta?
  - Tipicamente non lo decide l'utente, ma l'applicazione usata
  - il client tipicamente utilizza il metodo GET
  - es: l'utente specifica un URI nella barra
  - es: l'utente seleziona un collegamento
- Metodo POST
  - quando l'utente sottomette dati tramite un modulo (form) il metodo può essere POST o GET

HTTP 1.0: Richiesta

- Intestazioni delle richieste HTTP
- Esempi
  - User-Agent – es: User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; Q312461)
    - Quale tipo e versione di browser si usa
  - If-Modified-Since – es:
    If-Modified-Since: Thu, 01 Mar 2013 08:00:00 GMT
    - Solo risorse modificate dopo una certa data
  - Authorization – es:
    Authorization: Basic ZGRpbjpvcGVuIHNl==
  - Referer – es. Referer: http://www.unich.it/index.html

HTTP 1.0: Risposta

- Linea iniziale della risposta
  - HTTP/1.0 <codice numerico> <descrizione>
- Codice numerico
  - 1xx: messaggio informativo
  - 2xx: richiesta servita con successo
  - 3xx: c'è stata una redirezione
  - 4xx: errore del client
  - 5xx: errore del server

HTTP 1.0: Risposta

- Esempi:
  - HTTP/1.0 200 OK
    - risorsa nel corpo del messaggio
  - HTTP/1.0 301 Moved Permanently
    HTTP/1.0 302 Moved Temporarily
    - nuovo URI nel corpo del messaggio
  - HTTP/1.0 404 Not Found
    HTTP/1.0 401 Unauthorized
  - HTTP/1.0 500 Server Error

HTTP 1.0: Risposta

- Intestazioni delle richieste HTTP
- Esempi
  - Content-Type – es: Content-Type: text/html
  - Content-Length – es: Content-Length: 650
  - Last-Modified –
    es: Last-Modified Thu, 01 Apr 2002 16:00:00 GMT
  - Pragma – es: Pragma: no-cache
  - Server – es: Server: Apache 1.3.20
  - Location –
    es: Location: http://www.unich.it/newindex.html
  - WWW-Authenticate –
    es: WWW-Authenticate: Basic realm="Area Privata"

HTTP 1.0: Un Esempio di GET

- Richiesta

GET /news/index.html HTTP/1.0

User-Agent: Mozilla/4.0
(compatible; MSIE 5.0;
Windows XP) Opera 6.0 [en]

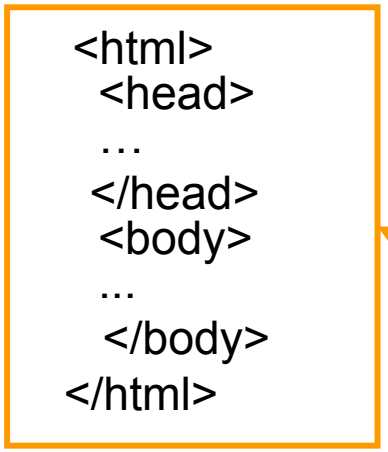
Referer: http://www.unich.it

<linea vuota>

- Risposta

HTTP/1.0 200 OK

Date: Thu, 01 Apr 2002 16:00:00 GMT
Content-Type: text/html
Content-Length: 1534



corpo della risposta: contenuto del file index.html

HTTP 1.0: Un Esempio di POST

- Richiesta

```
POST /bollo.asp HTTP/1.0
User-Agent: Mozilla/4.0
    (compatible; MSIE 5.0;
    Windows XP) Opera 6.0 [en]

targa=AB123CD&utente=Mario
    %20Rossi
```

si suppone che l'utente abbia riempito e sottomesso una maschera basata sul metodo POST

- Risposta

```
HTTP/1.0 200 OK
Date: Thu, 01 Apr 2002 16:00:00
    GMT
Content-Type: text/html
Content-Length: 2384
Pragma: no-cache
```

```
<html>
…
targa: AB123CD
EUR 110,00 ...
</html>
```

corpo della risposta: codice HTML generato dinamicam.

HTTP 1.0: Un Esempio di POST

- Attenzione alle differenze
  - nel primo caso stiamo richiedendo il contenuto di un file (index.html)
  - nel secondo caso stiamo chiedendo l'esecuzione di un'applicazione, passando dei parametri
    - l'applicazione genera il codice HTML corrispondente al messaggio di risposta

HTTP 1.1

- Standard IETF (RFC 2616)
- Principali obiettivi
  - migliorare le prestazioni di HTTP 1.0
  - rendere il protocollo più flessibile
- Attualmente
  - è implementato da quasi tutti i server ed i browser
  - ma viene mantenuta compatibilità con il passato per via dei vecchi browser

HTTP 1.1

- Problemi di HTTP 1.0
  - lentezza e congestione nelle connessioni >> connessioni multiple
  - un solo server Web per ciascun IP
  - limiti del meccanismo di autorizzazione (password in chiaro)
  - limiti nel controllo dei meccanismi di caching

HTTP 1.1

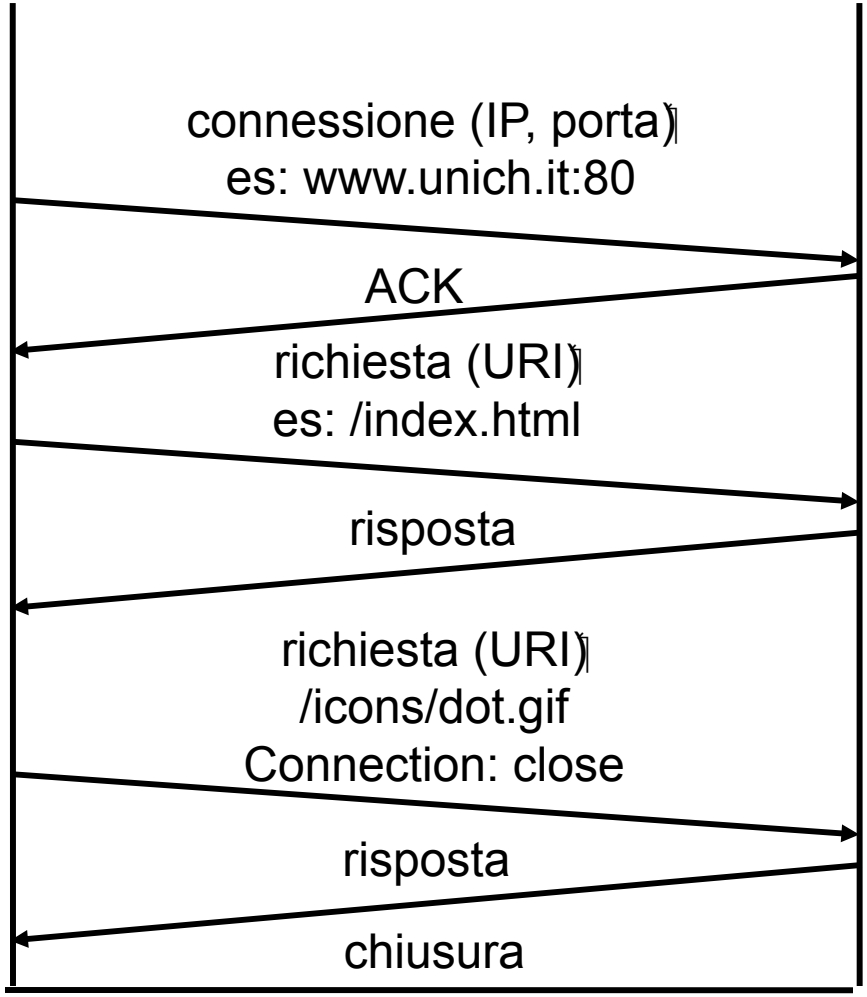
- Novità principali
  - connessioni persistenti
  - host virtuali
  - autenticazione crittografata (“digest”)
- Altre novità
  - nuovi metodi di accesso, miglioramento dei meccanismi di caching

HTTP 1.1: Connessioni Persistenti

- Modalità standard di HTTP/1.1
  - più di una transazione si può svolgere lungo la stessa connessione TCP
  - nuova intestazione del client
    Connection: close
  - nuovo messaggio del server
    HTTP/1.1 100 Continue
  - il server può chiudere la connessione unilateralmente dopo un certo “timeout”

HTTP 1.1: Connessioni Persistenti

browser



Server Web

HTTP 1.1: Host Virtuali

- Ad uno stesso IP possono corrispondere nomi diversi e server diversi
  - requisito importante per i “provider”
  - IP e porta non bastano più ad identificare il server
- Nuova intestazione del client **obbligatoria**
  - Host: serve a specificare il nome del server es: Host: www.tin.it

HTTP 1.1: Hosts Virtuali

- Indirizzo IP 192.168.3.109 con due host:
  - www.tin.it, www.virgilio.it
- Richiesta al sito 1:

  GET /news/index.html HTTP/1.1

  Host: www.tin.it
- Richiesta al sito 2:

  GET /news/index.html HTTP/1.1

  Host: www.virgilio.it

HTTP 1.1: Autenticazione “Digest”

- Le password non vengono trasmesse
- Il server invia al browser una stringa
  - “nonce”
- Il browser risponde con
  - nome utente
  - un valore crittografato basato su: nome utente, password, URI e nonce (algoritmo MD5, sunto di 128 bit in formato ASCII)
  - il browser ricorda l’autorizzazione

HTTP 1.1: Autenticazione “Digest”

- Esempio: Richiesta

GET /privato/index.htm HTTP/1.1

- Risposta

HTTP/1.1 401 Unauthorized

WWW-Authenticate: Digest
realm=“Area Privata",
nonce="dcd98b7102dd2f0“

- Il browser richiede nome utente e password all’utente

- Nuova Richiesta

GET /privato/index.htm HTTP/1.1

Authorization: Digest
username=“Pinco",
realm=“Area Privata",
nonce=" dcd98b7102dd2f0",
uri="/privato/index.htm",
response="6629fae49393a053
97450978507c4ef1“

- Nuova Risposta (2xx oppure 4xx)

HTTP 1.1: Autenticazione “Digest”

- Vantaggio
  - le password non vengono trasmesse direttamente sulla rete in chiaro
  - il meccanismo è decisamente più sicuro
- Ma
  - non è sicuro al 100%
  - è possibile intercettare la richiesta e riprodurla per accedere alle risorse protette

HTTPS: Cenni

- La soluzione: HTTPS
- HTTPS: HTTP over SSL (RFC 2818)
  - soluzione considerata più sicura
- SSL: Secure Socket Layer
  - protocollo di trasporto
  - tutti i messaggi sono crittografati
  - crittografia a chiave pubblica (certificato)
  - trasparente per lo sviluppatore

Modello concettuale per il Web

Un modello concettuale per il Web

- Contenuti + Presentazione + Comportamento
- Nell'editoria tradizionale:
  - Testo (contenuti)
  - Immagini e layout (presentazione visuale)
  - Fondamentale l'integrazione tra i due aspetti
- Il Web aggiunge il comportamento
  - Si seguono collegamenti ipertestuali
  - Si fanno ricerche
  - Si immettono informazioni
  - Si consultano cataloghi
  - SI effettuano pagamenti
  - …

Presentazione

- Per una comunicazione efficace è importante curare l'aspetto grafico e le immagini
- I web designer vogliono controllo sull'aspetto del documento
- Per questo motivo HTML contiene annotazioni strutturali e stilistiche
  - Font
  - center, attributo align
  - Background-color

Separazione stile/contenuto

- L'arricchimento di HTML con annotazioni stilistiche non è stata una buona idea
- Lo stile dovrebbe essere specificato in modo distinto dalla struttura
- CSS (Cascading Style Sheets)
  - Separano il contenuto dalle istruzioni per la presentazione (per i browser)
  - Le istruzioni sono di validità globale per la pagina o per il sito
  - Sono uno standard del W3C
  - Sono ormai supportati da quasi tutti i browser (in varia misura)

Comportamento

- Il Web è interattivo, le pagine hanno un comportamento
- Sono vere applicazioni informatiche interattive
  - Usabilità: facilità e soddisfazione con cui si svolge un certo compito
    - Navigare nel sito
    - Comprare un libro in un sito di commercio elettronico
    - Consultare l'orario dei treni, ecc…
  - Accessibilità: essere fruibile con facilità da qualsiasi categoria d'utente
- Servono capacità di programmazione
  - Pagine interattive (che cambiano in reazione ad eventi)
  - Pagine dinamiche (generate "al volo")
    - Basi di dati
    - Programmazione lato server (*back-end*)

Attività interdisciplinare

- Realizzare un sito Web richiede:
  - Capacità editoriali e di strutturazione del contenuto
  - Capacità di presentazione grafica
  - Capacità di programmazione
- Lavoro di gruppo
  - Editore di contenuti o architetto dell'informazione
  - Grafico
  - Programmatore
- Conoscenze interdisciplinari

Progettare con gli standard

- Standard per i tre aspetti:
  - Contenuti: HTML 4.1 e XHTML
  - Presentazione: fogli di stile (CSS2)
  - Programmazione: Javascript (ECMA script)
- Problema del supporto nei browser
- Conviene rispettare gli standard: compatibilità in avanti
- www.w3c.org, www.ecma-international.org www.ietf.org

Conclusione

- Per progettare un sito Web bisogna conoscere le specificità espressive del mezzo e comprendere la tecnologia usata “dietro le quinte”
- Progettare un sito Web richiede un buon equilibrio tra
  - Struttura
  - Presentazione
  - Interattività
- Usare tecnologie e strumenti che rispettano gli standard internazionali

HTML

Sintassi HTML

- HTML = HyperText Markup Language
  - Linguaggio usato per creare documenti multimediali ipertestuali
- Il documento contiene **elementi**
- Gli elementi sono delimitati da **tag**
- I tag sono contenitori per porzioni di documento (gli elementi):
  - tag di apertura
    - *<nome-elemento>*
  - tag di chiusura
    - *</nome-elemento>*
- Esempio elemento P
  - <P>Contenuto del paragrafo</P>

- Parental advisory: Il codice HTML mostrato contiene annotazioni stilistiche!!! (**sconsigliate**)
- Gli elementi possono avere ***attributi*** nel tag di apertura
  - nome_attributo="valore_attributo"

```
<font face="arial" size="+2">
     Esempio
</font>

<p align="right">
  Paragrafo allineato a destra
</p>
```

- I tag devono essere **annidati** correttamente

     Esempio

  e non   Esempio
  - come le parentesi graffe, quadre, tonde
- Nei documenti HTML
  - gli spazi consecutivi vengono considerati come unico spazio
  - gli "a capo" non hanno effetto sulla formattazione
- **<!-- Commento nel file HTML -->**

XHTML

- XHTML è un linguaggio simile ad HTML ma più rigoroso
  - una specializzazione di XML (eXtensible Markup Language)
  - Tutti gli elementi (anche se vuoti) si aprono e si chiudono
  - Nomi elementi ed attributi in **minuscolo**
  - Usano solo gli attributi previsti
- Esiste una notazione abbreviata per gli elementi vuoti

  `<br/>` equivale a `<br></br>`

Struttura minima del documento

Prologo... (prossima slide)

```
<html>
    <head>
     <title>Il titolo del documento</title>
      <meta …>
    </head>
    <body>
      <h1>Titolo del paragrafo di primo livello</h1>
      <p> Testo all'interno del paragrafo </p>

    </body>
</html>
```

- Scrivere questo codice in un file di testo
- Salvarlo come testo e con estensione html
- Visualizzarlo con il browser

Prologo: Document type declaration

- Inserito all'inizio del documento serve a dichiarare il tipo di linguaggio utilizzato
- strict.dtd = solo elementi non deprecati
  - <?xml version="1.0" encoding="iso-8859-1"> <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
- transitional.dtd = tollera anche elementi deprecati
  - <?xml version="1.0" encoding="iso-8859-1"> <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

Meta-dati

- nella parte di HEAD dati utili per: indicizzare la pagina HTML fornire informazioni al server web e/o al browser:
- <meta name="Author" content=“F.Fioravanti">
- <meta name="Keywords" content="html,php,web">
- <meta http-equiv="Content-Type"
  content="text/html; charset=ISO-8859-1">

Classificazione elementi

- Blocco
  - Generano blocchi di contenuti identificabili visualmente
  - p.e: ,<form>,<table>,<p>, <h1>,...,<h6>, <ul>,
- Inline
  - Si dispongono consecutivamente senza creare blocchi
  - p.e. <a>,<img>,<i>,<b>,,<input>

Titoli e paragrafi

- Titoli di diverso livello (fino a 6)

```
<h1>Un titolo di primo livello</h1>
<h2>Un titolo di secondo livello</h2>
<h3>Un titolo di terzo livello</h3>
<h2>Un altro titolo di secondo livello</h2>
```

- Paragrafi

```
<p>Questo è un primo paragrafo di testo</p>
<p>Questo è un secondo paragrafo di testo
un po' piu` lungo del primo</p>
```

Formattazione di base e semantica

- Formattazione semantica
  - sono annotazioni relative al significato (logiche), alla funzione assunta nel documento
  - Questo e` molto interessante!
  - Questo e` molto interessante!
- Formattazione di base
  - sono annotazioni stilistiche (fisiche), legate alla modalità di visualizzazione
  - Questo e` molto interessante! (italico)
  - Questo e` molto interessante! (grassetto)
- **Preferire formattazione semantica**

Stile logico e fisico

- Altri tipi di stile logico
  - <samp> output di un programma
  - <var> variabile in un programma
  - <acronym> acronimo
  - <abbr> abbreviazione
- <code, <samp, e kbd possono anche contenere pezzi di codice con >, < e &

- Esempi stile fisico
- <b> grassetto
- <u> sottolineato
- <blink> lampeggiante
- <strike> barrato
- <sub> subscript
- <sup> superscript

Riga orizzontale

- <hr> traccia una riga orizzontale
- Attributi:
  - size=valore spessore della riga in pixel.
  - width=valore larghezza in pixel o in percentuale (%)
  - noshade senza effetto ombra
  - align= "tipo“ left, right, o center.

Le immagini

-
  - visualizza l'immagine contenuta nel file specificato nell’attributo src
-
  - Specifica del testo alternativo
-
  - Specifico larghezza e altezza

I collegamenti (link)

- Per i collegamenti si usa il tag <a> con attributo href, il cui valore è l'URL della pagina che vogliamo collegare.

```
<a href="pagina.html">Pagina collegata</a>
```

- Il testo tra <a> e è il testo del collegamento (che sarà visualizzato in blu e sottolineato dai browser)
- Link assoluti e relativi:

```
<a href="http://www.w3.org/">W3C</a>
<a href="didattica/wcmel">
   <img src="logo.gif">
</a>
```

- Quando possibile preferire link relativi

Collegamenti interni alle pagine

- Per poter saltare nel mezzo di una pagina è necessario predisporre un'**àncora**

```
<a name="qui">
  <h3>Sezione 1</h3>
</a>
<a href="#qui">Salta a sezione 1</a>
```

- Si può anche saltare nel mezzo di una pagina diversa

```
<a href="pagina2.html#li">Salta li</a>
```

Liste ordinate e non

- **Lista non ordinata / non numerata (unordered list)**
  <ul>
  **
il primo elemento della lista
**
  il secondo elemento
  il terzo elemento
  **
**
- **Lista ordinata / numerata (ordered list)**
  <ol>
  il primo elemento della lista
  il secondo elemento
  il terzo elemento
  **
**
  - **Lettere minuscole (a,b,c) o maiuscole (A,B,C)**
  - **Numeri (1,2,3), Numeri romani minuscoli (i,ii,iii), maiuscoli (I,II,III)**
- **Stesso tag per gli elementi della lista (list item)**

Liste di definizioni

- <dl>
  <dt>primo elemento</dt>
  <dd>definizione</dd>

  <dt>secondo elemento</dt>
  <dd>definizione</dd>

  <dt>terzo elemento</dt>
  <dd>definizione</dd>
  </dl>

Andare a capo e inserire spazi

- Gli spazi e gli “a capo” nel file vengono ignorati dal browser
  - Per andare a capo si usa
  - Per introdurre spazi si usa il carattere speciale   (spazio non "interrompibile").
- Esempi

  Fabio Fioravanti

  Dipartimento di Scienze

  Viale Pindaro

     Fabio   Fioravanti

Caratteri speciali

à	à à
è	è è
é	é é
<	&lt;
>	&gt;
&	&amp;
spazio	

©	&copy;
®	&reg;
“	&quot;
‘	&…
#	

NOTA: alcuni caratteri non possono essere usati direttamente nei documenti HTML perché hanno un significato speciale, p.e. & < > “ ‘

Bisogna usare un codice che inizia con &…

Tabelle

```
<table border="1“>
    <tr>
        <th>Anno</th>
        <th>Vendite</th>
    </tr>
    <tr>
        <td>2008</td>
        <td>18M</td>
    </tr>
    <tr><td>2009</td><td>25M</td></tr>
</table>
```

Attributi delle tabelle

- border = dimensione
- width = dimensione
- height= dimensione
- cellspacing= dimensione
- cellpadding= dimensione

- Dimensioni espresse in valori assoluti e/o relativi
  - dipende dall'attributo

Tabelle senza bordo

- Le tabelle servono per dati strutturati ma non solo …
- Le tabelle senza bordo sono tuttora uno strumento molto usato (anche abusato) per disporre gli oggetti nella pagina
  - Attributo border="0“
- Preferire, quando possibile, layout CSS

Ampiezza di tabelle e colonne

```
<table border="1" width="80%">
   <tr><th>Anno</th><th>Vendite</th></tr>
   <tr><td>2008</td><td>18M</td></tr>
   <tr><td>2009</td><td>25M</td></tr>
</table>
```

- Attributo width
  - funziona anche sulle celle
  - valore assoluto (punti) o relativo (%)

Cellpadding e cellspacing

- Cellpadding definisce lo spazio interno ad ogni cella (rif. “box model”)

- Cellspacing definisce lo spazio tra le celle

```
<table border="1" cellpadding="10">
   <tr><th>Anno</th><th>Vendite</th></tr>
   <tr><td>2008</td><td>18M</td></tr>
   <tr><td>2009</td><td>25M</td></tr>
</table>
```

Altri elementi delle tabelle

- intestazione <thead>…</…>
- contenuto <tbody>…</…>
- footer <tfoot>…</…>
- didascalia <caption>…</…>
- riassunto <summary>…</…>

Righe e celle

- Riga
  - <tr ... > ... </tr>
- Cella
  - Intestazione: <th ... > ... </th>
  - Dato: <td ... > ... </td>

Allineamento dei contenuti

- Attributo align di righe o celle
  - Il valori possono essere “left”, “right”, “center”
- Attributo valign, per allineare in verticale
  - I valori possono essere “top”, “middle”, “bottom”
- NOTA: Le celle di tabelle con bordi che non hanno contenuti potrebbero avere un aspetto non gradevole
  - Soluzione: riempirle con

Celle su più colonne/righe

- L’attributo colspan="n" di una cella dice che quella cella si estende per n colonne.
- L’attributo rowspan="n" di una cella dice che quella cella si estende per n righe.

Esempio:

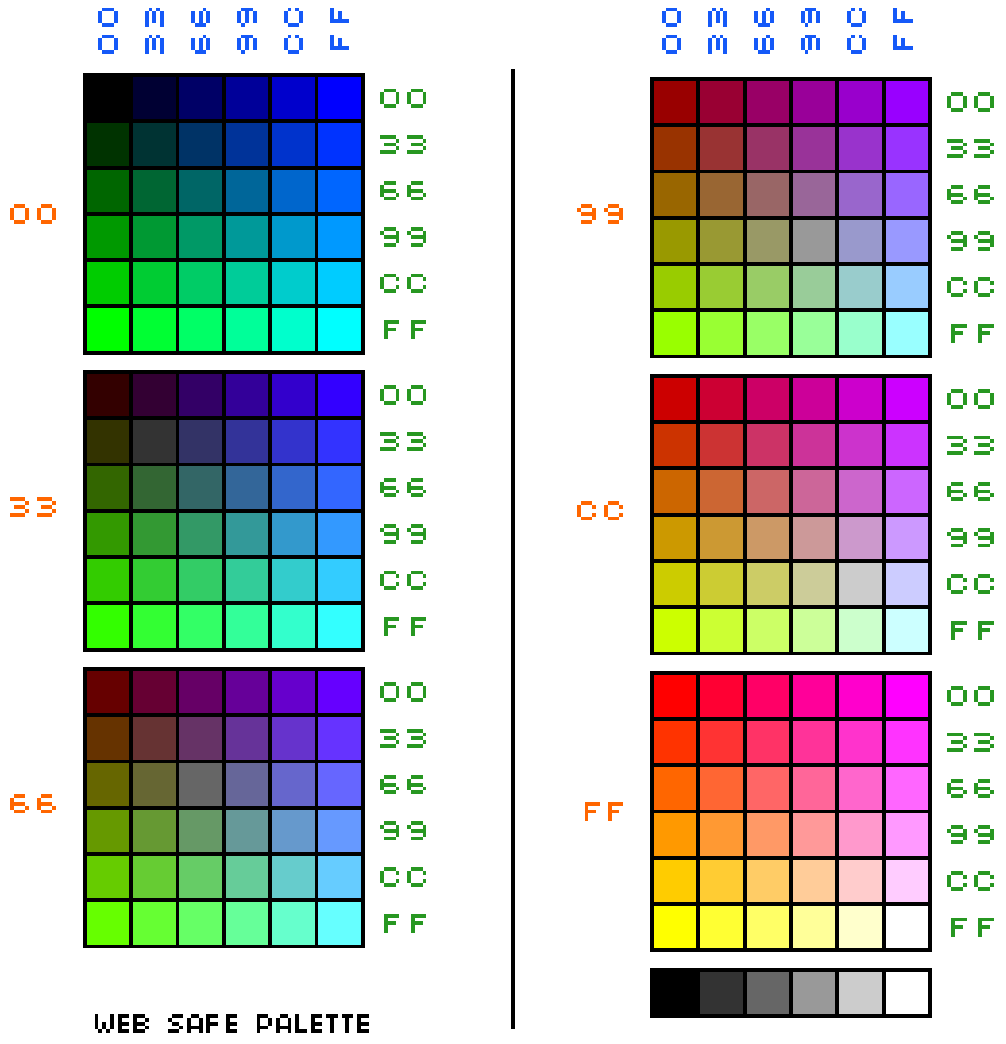
```
<table>
  <tr><td colspan="2">Titolo</td></tr>
  <tr><td>Uno</td><td>Due</td></tr>
</table>
```

Colori

- Colori: un certo numero sono disponibili con il loro nome in inglese
  - Blue, green, yellow, red, magenta …
  - versione light e versione dark
- I colori sono codificati con un numero di 6 cifre esadecimali (codifica RGB: red/green/blue)
  - `#ff0000` corrisponde al rosso
  - `#00ff00` corrisponde al verde
  - `#0000ff` corrisponde al blu
  - `#000000` corrisponde al nero
  - `#ffffff` corrisponde al bianco
- Si può usare una codifica abbreviata
  - `#ff0000` abbreviato con `#f00`
  - `#0f0000` non abbreviabile
- Si usa l'attributo bgcolor di body, td … per colorare lo sfondo della pagina o di una cella

Colori

- Colori Web-safe
- 256 colori
- composti usando solo i valori 00,33,66,99,cc,ff



Font

- ...
  - scelta del carattere per il blocco di testo incluso tra i tag
  - deprecato, uso sconsigliato (usare CSS)
- attributi:
  - size= dimensione
  - color= colore
  - face= font-family …
- Unità di misura dimensione:
  - pt, px, in, cm,
  - %, N (=1…7, default=3), +N, –N
  - consigliabili % +N –N

PRE

- Il tag <pre> si usa per testi preformattati.
  - Un testo racchiuso tra i tag <pre> </pre> mantiene gli spazi e le interruzioni di linea del documento HTML.

Frame e frameset

- La pagina è divisa in zone il cui contenuto è all'interno di altri file HTML
- uso sconsigliato (usare CSS e "include")
- frameset possono essere annidati
- il contenuto di ciascun frame è specificato tramite:
  - <frame src=URI name=...>
  - L'attributo name è importante per i link dagli altri frame
    - …

Esempio frame

- Esempio

```
<frameset cols=“100,*”>
    <frame src="menu.html">
    <frame src=“contenuto.html"
      name="content">
</frameset>
```

- Non tutti i browser supportano i frame
  - <noframe>…</…>: testo da visualizzare sui browser che non supportano i frame

Dimensioni frame

- Dimensioni dei frame nella pagina
  - percentuale (dello spazio disponibile)
  - valore assoluto (in punti o pixel)
  - * per indicare di usare tutto lo spazio rimasto
  - Se le dimensioni sono superiori si verifica “overflow” e si attivano le barre di scorrimento “scrollbar” (orizzontali/verticali)
- Esempio suddivisione orizzontale
  - <frameset rows=“30%,50%,20%”>
- Esempi suddivisione verticale:
  - < frameset cols=“30%,70%”>
  - < frameset cols=“200,*,100”>

IFRAME

- Inline Frame
- Inserisce un frame all'interno di un documento convenzionale (non è usato all'interno di un FRAMESET)
- Il frame inserito, relativamente al suo posizionamento rispetto al resto del testo, si comporta come se fosse un'immagine
- Ricordarsi sempre di chiudere il tag IFRAME

Frame

- Vantaggi
  - evitare di ricaricare le parti comuni
  - condivisione layout comune
- Svantaggi
  - Non è ben supportato dai browser
  - Cattiva indicizzazione dai motori di ricerca

DIV e SPAN

- introdotti in HTML 4.0 servono a raggruppare parti per poter applicare più facilmente una formattazione
- DIV identifica un blocco
  - tipicamente i browser vanno a capo alla fine del blocco
  - Molto usati per creare (con un opportuno CSS) un layout di pagina senza usare table o frame
- SPAN identifica una parte all'interno di un blocco.
- "id" e/o "class" permettono di accedere tramite CSS e Javascript
  -
  -

Strumenti

- http://validator.w3.org
  - verifica se una pagina aderisce agli standard
- http://cgi.w3.org/cgi-bin/tidy
  - corregge gli errori nel codice HTML
- http://validator.w3.org/checklink
  - Verifica l'esistenza delle pagine usate nei link

Licenza

- Creative Commons
  - http://creativecommons.org/licenses/by-nc/2.0/it/
    Maria Simi – UNIPI
  - http://creativecommons.org/licenses/by-sa/1.0/
    G. Mecca - UNIBAS

Riferimenti

- Standard
  - http://www.w3.org/html/
- Guide HTML.it
  - HTML
    - http://xhtml.html.it/guide/leggi/51/guida-html/
  - XHTML
    - http://xhtml.html.it/guide/leggi/52/guida-xhtml/
- In inglese (autore Dave Raggett)
    - http://www.w3.org/MarkUp/Guide/
    - http://www.w3.org/MarkUp/Guide/Advanced.html

Form (moduli)

- I moduli rendono le pagine WEB interattive
- L'utente interagisce con una pagina che contiene un(a) form
- L'utente inserisce dati e invia premendo un pulsante
- Il server riceve i dati tramite un messaggio di richiesta HTTP con metodo GET o POST (POST piu' robusto, preferibile se il server genera effetti collaterali)
- Il server elabora i dati ricevuti ed invia all'utente la pagina contenente la risposta

Tipi di controlli

- Bottoni
- Caselle di spunta
- Radio button
  - Come caselle di spunta, ma mutuamente esclusive
- Liste di selezione
- Caselle di testo
- Selezione di file
- Controlli nascosti

- <form id=“bollo“ action=“bollo.php“ method=“POST">
  ...
  </form>
- id: codice identificativo del modulo
- action: pagina che elabora i dati
- method: metodo HTTP
- enctype: tipo di codifica dei dati
  - "text/plain“
  - “application/x-www-form-urlencoded” - con POST
  - "multipart/form-data" - obbligatorio se si inviano file

INPUT

- L'elemento INPUT definisce un campo di input.
- Attributi principali
  - type tipo di controllo
  - name nome del controllo
  - value valore iniziale del controllo (o etichetta)
- Altri attributi validi solo per alcuni valori di type
  - size, checked, maxlength, src

INPUT

- Ulteriori attributi di INPUT; valgono anche per altri controlli
  - tabindex — indice di tabulazione
  - accesskey — scorciatoia da tastiera
  - readonly — sola lettura, valore inviato al server
  - disabled — disabilitato, valore non inviato al server

INPUT

- <input type="…" name="…" value="…">
  - type definisce il tipo di controllo
  - name
  - value
    - Valore se type in {text, password, file, hidden}
    - Etichetta se type in {submit, reset, button}

INPUT

- type=”text”
  - Campo di testo di una riga
- type=”password”
  - Come TEXT, ma con l'input mascherato (*****)
- type=”hidden”
  - non visualizzato ma inviato al server
- type=”file”
  - selezione di file

INPUT

- type=”checkbox”
  - casella di controllo. selezionata o non selezionata
- type=”radio”
  - casella per la scelta di una tra varie alternative.
- type=”submit”
  - pulsante per l'invio dei contenuti del modulo
- type=”reset”
  - pulsante per ripristinare i contenuti di un modulo ai valori di default
- type=”image”
  - utilizzare immagini come bottoni (attributi SRC ed ALT)

INPUT TYPE="TEXT“

- MAXLENGTH
  - Massimo numero di caratteri inseribili
- SIZE
  - Dimensione della casella
- VALUE
  - Valore predefinito
- Esempio
  - <input type="text" name="domanda“ maxlength=“25” size="5" value=”Inserire qui la domanda”>

INPUT TYPE="PASSWORD"

- Caratteri inseriti mascherati con asterischi
- Non visualizzato a video ma inviato "*in chiaro*"
- Attributi
  - MAXLENGTH
  - SIZE
  - VALUE

INPUT TYPE="HIDDEN"

- Campo nascosto a video ma inviato al server
- Usato per passare informazioni allo script sul server senza “appesantire” visivamente il form

INPUT TYPE="CHECKBOX"

- Casella di controllo
- VALUE
  - Valore inviato allo script se (e solo se) la casella è selezionata
- CHECKED
  - Pre-seleziona la casella
- Esempio

```
<input type="checkbox" name=“bianco“ value="si"> bianco <br>
<input type="checkbox" name=“rosso“ value="si" checked=”checked”>
    rosso<br>
<input type="checkbox" name=“verde“ value="si"> verde <br>
```

INPUT TYPE="RADIO"

- Pulsanti di opzione
- Scelta esclusiva: può essere selezionata soltanto una delle opzioni nello stesso gruppo
- Opzioni raggruppate usando stesso nome ma valori diversi
- Scelta non obbligatoria: se non si seleziona nulla, nulla viene inviato allo script
- Esempio
  - HTML<input type="radio" name=“argomento" value="html“ /> CSS <input type="radio" name=" argomento " value="css“ />

INPUT TYPE="FILE"

- Selezione di un file locale per inviarlo al server con il modulo
- Il browser aprirà una finestra di dialogo

INPUT TYPE=”SUBMIT” e “RESET”

```
<input type="submit“ value="invia">
<input type="reset“ value="cancella”>
```

TEXTAREA

- Inserimento testo su più righe.
  - NAME: nome del controllo
  - ROWS: numero di righe
  - COLS: numero di colonne

- Esempio

```
<textarea name=“corpo" rows="6" cols="35">
Testo predefinito.
Ciao come stai?
</textarea>
```

SELECT

<select> … </select>

- Lista di selezione o menù a discesa
  - Elementi della lista specificati con OPTION
- NAME
- MULTIPLE (attributo booleano)
  - Selezione di molteplici opzioni
- SIZE
  - Numero di opzioni mostrate nel controllo

OPTION

- Elemento della lista del tag SELECT
- Attributi principali:
  - VALUE
    - Indica il valore da inviare al server.
  - SELECTED
    - Preselezione di una voce del menù (**selected=”selected”**)
    - Anche su più elementi di tipo option se multiple=”multiple”

Raggruppare opzioni

- <optgroup label=“Argomenti lezione">
  <option value=“html"> HTML </option>
  <option value=“css"> CSS </option>
  </optgroup>

Raggruppare elementi

- <form action="…">
  <fieldset>
  <legend>Anagrafica</legend>
  <input type="text" name="nome">
  <input type="text" name="cognome">
  <input type="text" name="eta">
  </fieldset>
  </form>

Etichette dei controlli

- Elemento label associa un'etichetta ai controllo in maniera semantica (quindi maggiormente accessibile) invece che esclusivamente in maniera visuale (p.e. tabelle)
- <label>Cognome:
  <input type="text“ name=“cognome”>
  </label>
- <label for=”**cogn**”>Cognome:</label>
  <input type="text“ id=”**cogn**” name=“cognome”>

Note XHTML

- multiple="multiple"
- checked=“checked”
- disabled=“disabled”
- readonly=“readonly”

- <input … />