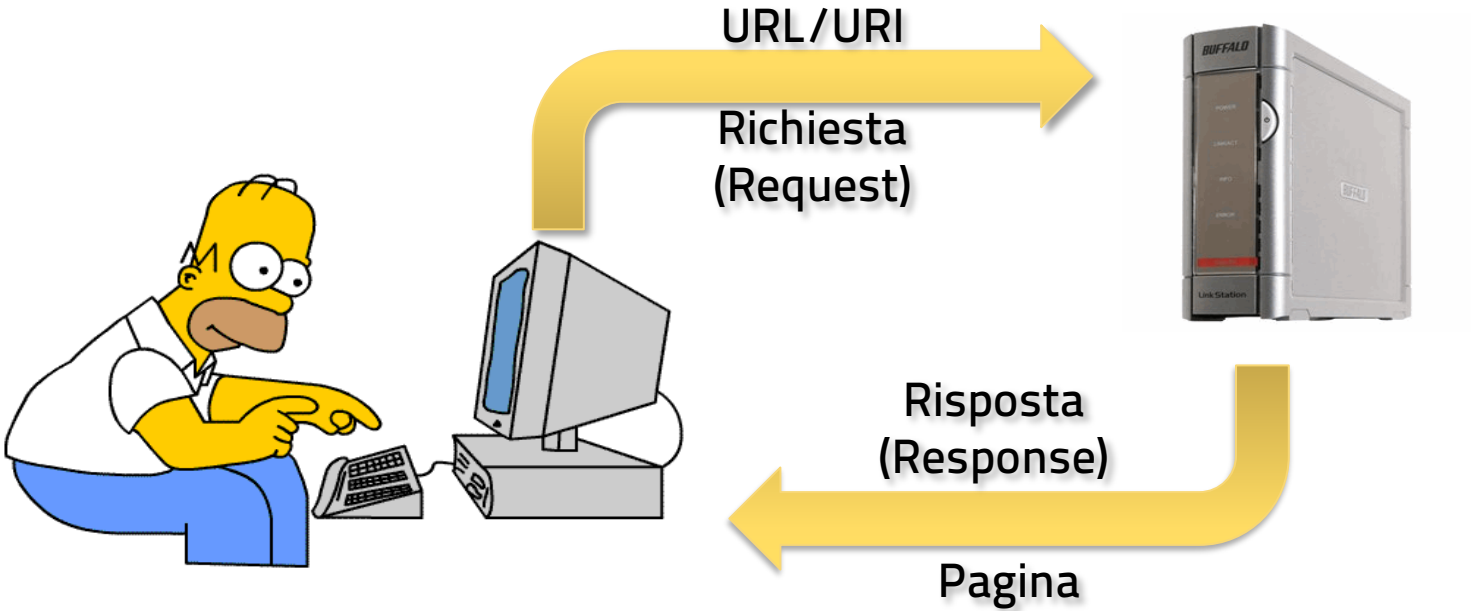
bruno@magicbusmultimedia.it

COME FUNZIONA INTERNET

ARCHITETTURA CLIENT SERVER



URL E URI

Uno **Uniform Resource Identifier** (URI, acronimo più generico rispetto ad "**URL**") è una stringa che identifica univocamente una risorsa generica che può essere un indirizzo Web, un documento, un'immagine, un file, un servizio, un indirizzo di posta elettronica, ecc. L'URL è un URI che indica una risorsa internet.

Un **Uniform Resource Locator** o **URL** è una sequenza di caratteri che identifica univocamente l'indirizzo di una risorsa in Internet, come un documento o un'immagine.

STRUTTURA DELL'URI

- La tipica struttura di un **URI** è:

  protocollo://indirizzo_risorsa

- In un URL **indirizzo_risorsa** può contenere informazioni aggiuntive:

  nomehost [:porta][/percorso][?querystring]

ESEMPIO DI URL

ARCHITETTURA CLIENT-SERVER

- Server
  - Programma *in ascolto* su una porta (punto di accesso)
  - Quando arriva una richiesta da un client, il server analizza questa richiesta (eventualmente con l'aiuto di altri programmi), elabora una risposta (anche in questo caso, eventualmente con l'aiuto di altri programmi) e la invia al client.
  - Un server, generalmente, può servire più client contemporaneamente
- Client
  - Un client è un programma che si connette ad un server, fa una richiesta ed aspetta una risposta

WEB SERVER

- Un Web Server (server che fornisce servizi sul Web) è sostanzialmente un HTTP Server (server che comunica mediante il protocollo HTTP) e gestisce 2 flussi di informazioni:
  - le richieste in arrivo dai client (HTTP request)
  - le risposte del server, inviate ai client (HTTP response)

BROWSER

- Un Web browser è un **HTTP client, cioè un** programma, dotato di interfaccia grafica, che:
  - **interagisce con un HTTP server , richiedendone** i servizi (per es. pagine Web)
  - riceve i dati dal server e li ricompone
  - **visualizza le pagine Web (ipertesti),** mostrandone il contenuto e interpretando correttamente i linguaggi che vengono utilizzati per descriverne i contenuti

LE APPLICAZIONI WEB

- Il passaggio di semplici documenti HTML tra il server e il client non permette lo sviluppo di applicazioni web complesse che coinvolgano una fase di elaborazione oltre che di passaggio di dati.
- Per questo motivo sono state sviluppate tecnologie che permettano una maggiore interazione dell'utente con il server web e una capacità di elaborazione sia del server che del client web.

PAGINE STATICHE E DINAMICHE

- Quando **ci connettiamo ad una risorsa in rete, identificata da un URL:**
  - Nel caso più semplice l'indirizzo di una pagina (generalmente scritta in HTML) il cui **contenuto è fisso (STATICA);**
  - In altri casi, l'URL può contenere l'indirizzo di una **pagina 'dinamica" (per esempio** scritta in ASP, PHP, o JSP) il cui **contenuto viene generato (selezionato, composto) al momento della richiesta;**

HTML

- **HTML (HyperText Markup Language) è un linguaggio di markup** (e NON un linguaggio di programmazione!) per scrivere pagine Web (ipertesti)
  - I **linguaggi di programmazione servono a scrivere programmi:** un programma è una sequenza di **istruzioni**
  - I **linguaggi di mark-up invece servono a scrivere documenti** ("formattati"): un documento formattato è un file di testo che contiene **istruzioni (tag) per la sua visualizzazione (struttura)**
  - I linguaggi di mark-up tendono a separare in modo chiaro **contenuto (testo) e aspetto (visualizzazione)**

PAGINE WEB DINAMICHE

- **Nelle pagine Web "dinamiche" il contenuto** viene generato (selezionato,composto) al momento della richiesta o della visualizzazione
  - Pagine Web "debolmente" dinamiche: queste utilizzano tecnologie **client-side**
  - Pagine Web autenticamente dinamiche: queste utilizzano tecnologie **server-side (i** programmi vengono eseguiti sul server web)

PAGINE WEB DINAMICHE

- Per visualizzare una pagina Web "debolmente" dinamica (che utilizza una tecnologia client-side) NON HO bisogno di un server
- Per visualizzare una pagina Web autenticamente dinamica (che utilizza una tecnologia server-side) HO bisogno di un server

PAGINE WEB DINAMICHE

- Client-side:
  - JavaScript
  - Java Applet
  - Flash
- Server-side:
  - ASP e ASP.NET di Microsoft
  - PHP
  - JSP (Java Server Pages)
- NB: Per le tecnologia Client-side è necessario che il Browser sappia interpretare le istruzioni!

HTML

FORMATTAZIONE...

- HTML è l'acronimo di HyperText Markup Language ("Linguaggio a marcatori per gli Ipertesti").
- Non è un linguaggio di programmazione non ha, cioè, meccanismi che consentono di prendere delle decisioni ("in questa situazione fai questo, in quest'altra fai quest'altro"), e non è in grado di compiere delle iterazioni ("ripeti questa cosa, finché non succede questo"), né ha altri costrutti propri della programmazione.
- Si tratta invece di un linguaggio di contrassegno (o 'di marcatura'), che permette di articolare gli elementi di una pagina in blocchi le cui caratteristiche vengono definite attraverso degli appositi marcatori, detti "tag".

... E SEMANTICA

- In origine HTML è stato concepito principalmente per formattare il testo: elementi con un preciso significato semantico si mischiavano con elementi di pura formattazione.
- Con l'evoluzione di html in xhtml e in html 5 i tag hanno sempre più assunto il compito di articolare la pagina i blocchi semantici:
  - Tutti i tag di pura formattazione (grassetto, corsivo, centrato) vanno considerati elementi deprecati.
  - Al contrario con html 5 sono stati introdotti tag che hanno l'unica funzione di definire in maniere più robusta le parti in cui è articolata una pagina Web (testata, piè di pagina, barra di navigazione, ecc.) dal punto di vista dell'organizzazione dei contenuti e dalla navigazione..
- Il compito di definire l'aspetto di una pagina è affidato ai fogli di stile.

I MARCATORI (TAG)

- I tag vanno inseriti tra parentesi uncinate: <TAG>
- La chiusura del tag viene indicata con una barra: </TAG>
- Il contenuto che il tag modifica va inserito tra l'apertura e la chiusura del tag medesimo:

```
Questa <span style="font-wieight:bold">parola</span> è in
                                grassetto.
```

- che nel rendering verrà reso:

Questa **parola** è in grassetto.

- Alcuni tag non hanno (o possono non avere) contenuto (empty tag) . Ad esempio l'interruzione di linea la indico così:

```
<br />
```

GLI ATTRIBUTI

- Le caratteristiche [illegible] del tag. Ogni tag ha [illegible] so modificare:

```
<TAG attri[illegible]TAG>
```

- Alcuni attribu[illegible]c.), altri sono specifici

```
<IMG widt[illegible] />
```

- Una caratteristica importante del codice HTML è che i tag possono essere annidati l'uno dentro l'altro.
- È quindi opportuno usare l'indentazione. Grazie ad essa il codice HTML risulta più leggibile.

```
<TAG1 attributo = "valore">
        contenuto 1
        <TAG2>
                contenuto 2
        </TAG2>
</TAG1>
```

ENTITY

- Per rappresentare i caratteri non coficabili con lo standard ASCII si è introdotta una codifica particola detta **entity**.
- Un entity è così composta:

&nnnn;

- Quando il parser HTML incontra una parola che inizia con **&** legge i successivi caratteri fino ad in contrare "**;**" e tenta di interpretare il tutto come un carattere secondo la tabella di codici definita dal W3C.

COMMENTI

- Un strategia importante, per rendere il nostro codice più leggibile è quella di inserire dei "commenti" nei punti più significativi:
- Un commento è un'indicazione significativa per il webmaster, ma invisibile al browser. Inserendo i commenti in punti specifici del documento ci permette di mantenere l'orientamento anche in file molto complessi e lunghi.
- La sintassi è la seguente:

```
<!-- questo è un commento -->
```

IL W3C

- L'organizzazione che si occupa di standardizzare la sintassi del linguaggio HTML (il W3C: World Wide Web Consortium).
- Ha rilasciato diverse versioni di questo linguaggio (HTML 2.0, HTML 3.2, HTML 4.0….);
- Allo stato attuale abbiamo a che fare con 3 versioni:
  - HTML 4.01 (24/12/1999)
  - XHTML 1.0 (01/08/2002)
  - HTML 5 (Bozza di lavoro: 19/10/2010)

CONTROLLO SINTASSI

- Il linguaggio HTML, pur essendo dotato di una sua sintassi precisa e codificata, non presuppone un controllo sintattico rigido.
- Per ragioni storiche (e strategiche) il fine principale di un browser è quella di fare vedere comunque qualcosa all'utente non formalizzandosi sul fatto che una pagina sia ben formata o meno.
- Se vi dimenticate di chiudere un tag, non verranno prodotti dei messaggi di errore; se non rispettate la sintassi probabilmente non otterrete la visualizzazione della pagina che desiderate, ma nient'altro.

XHTML

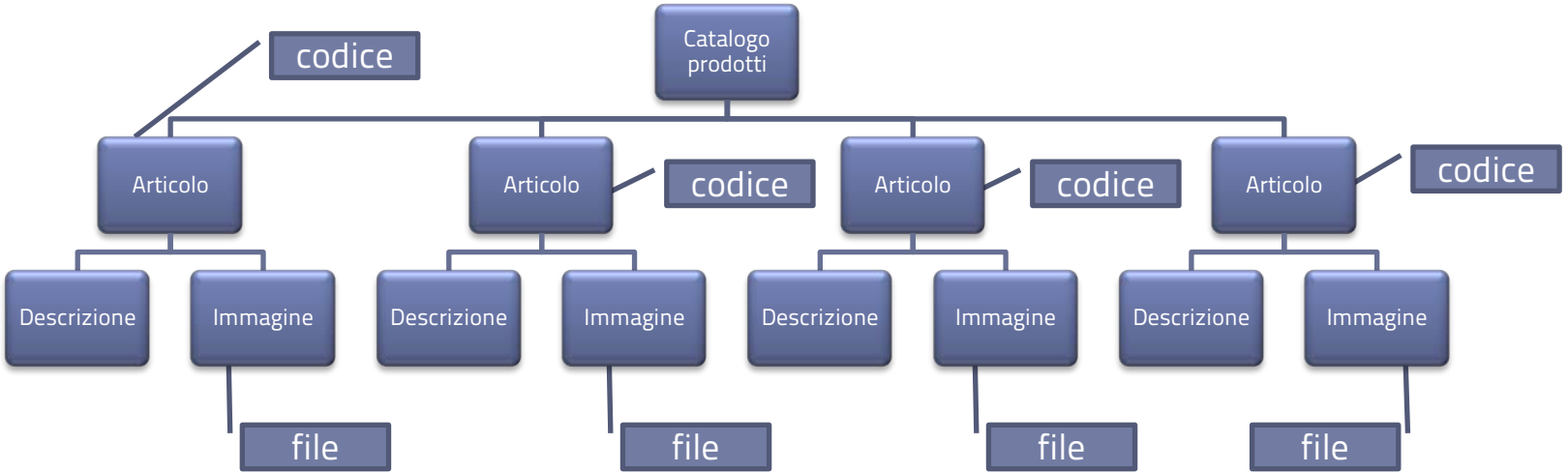
DEFINIZIONE DI XML

- Nel dicembre '97 il W3C rilascia le specifiche di XML come Proposed Recommendation.
- Gli obiettivi iniziali della nascita di XML erano rivolti alla soluzione di un problema di standard per il Web, ma ben presto ci si accorse che XML non era limitato al solo contesto Web.
- Esso risulta essere abbastanza generale per poter essere utilizzato nei più disparati contesti: dalla definizione della struttura di documenti allo scambio di informazioni tra sistemi diversi, dalla rappresentazione di immagini alla definizione di formati di dati.

DEFINIZIONE DI XML

- XML è un meta-linguaggio per definire la struttura di documenti e dati.
- Un documento XML è un file di testo che contiene una serie di tag, attributi e testo secondo regole sintattiche ben definite.
- Un documento XML è intrinsecamente caratterizzato da una struttura gerarchica. Esso è composto da componenti denominati elementi. Ciascun elemento rappresenta un componente logico del documento e può contenere altri elementi (sottoelementi) o del testo.
- Gli elementi possono avere associate altre informazioni che ne descrivono le proprietà. Queste informazioni sono chiamate attributi.
- L'organizzazione degli elementi segue un ordine gerarchico o arboreo che prevede un elemento principale, chiamato root element o semplicemente root o radice.

DEFINIZIONE DI XML



```
<?xml version="1.0" ?>
<catalogo>
   <articolo codice=“Codice del primo articolo">
      <descrizione>
         Blocco di testo della descrizione.
      </descrizione >
      <immagine file="immagine1.jpg"></immagine>
   </articolo>
   <articolo codice="Codice del secondo articolo">
      < descrizione >
                  Blocco di testo della descrizione.
      </descrizione>
   </articolo>
   <articolo codice="Codice del terzo articolo">
      <descrizione>
                  Blocco di testo della descrizione.
      </descrizione>
   </articolo>
</catalogo>
```

XML È UN METALINGUAGGIO

- A differenza dell'HTML in cui i tag sono predefiniti, XML ci lascia liberi di**definire i tag** che vogliamo.
- Per specificare un attributo per un elemento inseriamo il nome dell'attributo con il relativo valore all'interno del tag di apertura dell'elemento.
- L'organizzazione gerarchica degli elementi viene rappresentata in XML tramite il loro annidamento.
- Alcuni elementi possono essere vuoti, cioè possono essere privi di contenuto testuale. A differenza di quanto avviene per l'HTML, che consente l'utilizzo di elementi senza tag di chiusura, XML prevede che vengano sempre specificati i **tag di apertura e chiusura**.
- Un Dtd è un documento che descrive i tag utilizzabili in un documento XML, la loro reciproca relazione nei confronti della struttura del documento e altre informazioni sugli attributi di ciascun tag.

REGOLE DI BASE

- XHTML è la riformulazione di HTML come applicazione XML. Ciò significa essenzialmente una cosa: un documento XHTML deve essere valido e ben formato.
- Niente nuovi tag, attributi o metodi, cambiano le regole sintattiche.
- Le versioni di XHTML attualmente disponibili e pubblicate come raccomandazioni dal W3C sono tre.
  - XHTML 1.0 (26 gennaio 2000
    - DTD Strict
    - DTD Transitional
    - DTD Frameset
  - XHTML Basic (Dispositivi mobili)
  - XHTML 1.1

XHTML

- Un documento deve essere convalidato rispetto ad una delle tre DTD XHTML del W3C.
- Presenza di un elemento radice, corretto annidamento degli elementi, chiusura obbligatoria dei tag vuoti, etc.
- Ogni documento XML deve contenere un elemento radice. Si tratta dell'elemento che contiene al suo interno tutti gli altri. In un documento XHTML l'elemento radice deve essere <html>.
- L'elemento radice <html> deve contenere la dichiarazione di un namespace XML (spazio dei nomi) tramite l'attributo xmlns. Il namespace usato deve essere "http://www.w3.org/1999/xhtml".
- In un documento XHTML l'elemento radice deve essere preceduto da un elemento <!DOCTYPE>. All'interno di questo elemento è necessario specificare la DTD di riferimento e il suo URI.

XHTML

- Elementi devono correttamente annidati:

~~un test~~

un test

- Elementi e attributi devono essere in minuscolo
- Tutti gli elementi non vuoti devono essere chiusi
- I valori degli attributi devono essere posti tra virgolette:

<img src=test.gif>

~~<img src="test.gif">~~

XHTML

- Ogni attributo deve avere un valore:

~~<option selected>test</option>~~

<option selected="selected">test</option>

- Gli elementi vuoti devono terminare con />

~~<img src="test.gif">~~

- Per identificare un elemento si deve usare l'attributo id e non name.

DTD STRICT

- È la DTD più rigida, centrata esclusivamente sulla struttura del documento.
- Elimina diversi elementi ed esclude tutti gli attributi che definiscono la presentazione. Per questo scopo vanno usati i **CSS**.
- Tag non supportati: <applet>, <basefont>, , <dir>, <font>, <frame>, <frameset>, <iframe>, <isindex>, <menu>, <noframes>, <s>, <strike>, <u>

DTD TRASITIONAL

- Supporta tutti gli elementi e gli attributi di presentazione di HTML 4.0, anche quelli ritenuti sconsigliati.
- Garantisce la massima compatibilità con i vecchi browser.
- Consente anche sintassi fortemente sconsigliate che potrebbero non essere più supportate in futuro.

DTD FRAMESET

- È identica alla Transitional, ma va usata quando si utilizzano i frame.
- L'unica differenza è in pratica la sostituzione del tag <body> con <frameset> nella pagina principale

HTML 5

HEADER

- L'elemento <header> ha come scopo quello di racchiudere una porzione di codice che avrà ruolo di testata della pagina.

```
<header>
      <!-- il logo ed eventuale intestazione al sito -->
</header>
```

HGROUP

- L'elemento <hgroup> racchiude una porzione di codice composta da intestazioni

```
<header>
  <hgroup>
    <h1>Titolo del sito</h1>
    <h2>Sottotitolo</h2>
  </hgroup>
</header>
```

ARTICLE

- L'elemento <article> racchiude una porzione di codice semanticamente indipendente dal resto della pagina

```
<article>
  <header>
    <hgroup>
      <h1>Titolo del sito</h1>
      <h2>Sottotitolo</h2>
    </hgroup>
  </header>
</article>
```

TIME

- L'elemento <time> rappresenta una data di pubblicazione

```
<time datetime="2010-04-21" pubdate>21 Aprile 2009</time>
```

nav

- L'elemento <nav> specifica una porzione di codice destinato ala navigazione per esempio la lista di un menu.

```
<nav>
 <ul>
  <li><a href="#">link</a></li>
  <li><a href="#">link</a></li>
 </ul>
</nav>
```

ASIDE

- L'elemento <aside> viene utilizzato per definire una porzione di codice destinata correlata al contenuto ma comunque separata come per esempio una sidebar

FOOTER

- L'elemento <footer> definisce una porzione di codice che rappresenta di fatto il piede della pagina o del contenuto specifico.

```
<footer>&copy;2012 Mia Azienda Tutti i diritti riservati</footer>
```

VIDEO E CANVAS

- Le due novità non solo semantiche di HTML 5 che presuppongono uno specifico supporto del browser.
- Sostituiranno Flash ?

LA PAGINA HTML

STRUTTURA DELLA PAGINA

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

    <head>
        <title>La mia prima pagina XHTML</title>
    </head>

    <body>
        <h1>Benvenuto!</h1>
        <p>Questo &egrave; il mondo di XHMTL!</p>
    </body>

</html>
```

PROLOGO

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

- Dichiarazione XML (non obbligatoria):
  - version (il solo valore possibile è "1.0", in quanto non esistono altre versioni del linguaggio).
  - encoding: Serve a specificare la codifica del testo in cui è scritto il documento.
- Per massima compatibilità si può omettere la dichiarazione XML e usare il tag meta per indicare la codifica della pagina:

```
<meta http-equiv="Content-Type"
content="text/html;charset=UTF-8"/>
```

PROLOGO

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd>
```

- Un FPI (Identificatore Formale Pubblico) riferito ad una delle tre DTD XHTML
- L'URI della DTD
- Essa, dunque, ha lo scopo di stabilire a quale delle tre DTD XHTML intendiamo conformarci e di dire al browser dove essa è collocata.
- Il DOCTYPE non ha alcun effetto sulla presentazione della pagina. Serve solo al validatore per stabilire le regole della convalida.

ELEMENTO RADICE (ROOT)

```
<html xmlns="http://www.w3.org/1999/xhtml">
</html>
```

- <HTML> è obbligatorio
- L'elemento <html> può assumere questi attributi:
  - dir Determina la direzione del testo
  - lang Specifica il linguaggio di base dell'elemento quando è interpretato come HTML
  - xml:lang Specifica il linguaggio di base dell'elemento quando è interpretato come XML
  - xmlns Specifica il namespace predefinito per XHTML
- L'unico attributo obbligatorio è xmlns. Il W3C, come visto, specifica anche il valore obbligatorio di tale attributo: "http://www.w3.org/1999/xhtml".

HEAD

```
<head>
   <title>La mia prima pagina XHTML</title>
</head>
```

- La funzione principale della sezione <head> è quella di contenere informazioni che non vengono direttamente visualizzate nella pagina, ma che sono comunque di grande rilievo. Ecco l'elenco degli elementi che possono apparire nella testata:
  - <base> Usato per definire l'URL di base della pagina. Utilissimo per la risoluzione dei link relativi.
  - <link> Contiene informazioni su documenti esterni collegati. Usato soprattutto per i CSS.
  - <meta> Specifica informazioni di vario tipo sul documento.
  - <noscript> Usato per visualizzazioni alternative nei browser che non supportano gli script.
  - <object> Racchiude un oggetto.
  - <script> Contiene script di programmazione .
  - <style> Definisce le regole di formattazione per il documento corrente
  - <title> Specifica il titolo del documento che compare nella barra del titolo del browser

BODY

```
<body>
    <h1>Benvenuto!</h1>
    <p>Questo &egrave; il mondo di XHMTL!</p>
</body>
```

- Il corpo del documento è la sezione in cui si sviluppa il contenuto. È racchiusa, come in HTML, tra i tag <body>...</body>.
- Gli elementi che possono comparire all'interno del corpo sono in genere suddivisi in due categorie:
  - elementi blocco ed gli elementi blocco sono quelli che definiscono la struttura del documento. Possono contenere altri elementi blocco, elementi inline o testo. Quando sono inseriti danno origine ad una nuova riga nel flusso del documento.
  - elementi inline: quando sono inseriti non danno origine a una nuova riga e possono contenere solo dati (essenzialmente testo) o altri elementi inline.

ELEMENTI BLOCCO

- Il primo elemento della gerarchia dovrebbe essere , che definisce in pratica una sezione del documento. Al suo interno trovano posto gli altri elementi. Bisogna evitare annidamenti errati, che i browser fanno passare senza problemi, ma che il validatore segnala impietosamente in quanto violano le regole delle DTD.
- Esempio:

```
<p><div>Qui inserisco il mio testo</div></p>
<div><p>Qui inserisco il mio testo</p></div>
```

ELEMENTI BLOCCO

Elemento	Descrizione	DTD
<address>	Definisce un blocco di testo destinato a indirizzi, firme, indicazioni sull'autore. Non può contenere altri elementi blocco.	STF
<blockquote>	Usato per riportare citazioni da altri documenti. Il testo inserito viene indentato. Può contenere tutti gli elementi blocco.	STF
	Centra il testo che racchiude. Sconsigliato in HTML 4.0.	TF
<dir>	Crea una lista di tipo directory. Sconsigliato in HTML 4.0	TF
	Definisce un blocco di contenuto generico o una sezione del documento. Può contenere tutti gli elmenti blocco.	STF
<dl>	Crea una lista di definizione. Può contenere solo gli elementi <dt> e<dd>.	STF
<fieldset>	Usato per raggruppare campi di un form.	STF
<form>	Definisce un form. Può contenere i classici elementi dei form ma anche elementi blocco.	STF
<h1>..<h6>	Definiscono titoli e sottotitoli. Non possono contenere altri elementi blocco.	STF
<hr>	Inserisce una linea divisoria orizzontale. E' un elemento vuoto.	STF
<isindex>	Inserisce un elemento simile alle caselle di testo. Sconsigliato in HTML 4.0	TF
<menu>	Definisce una lista di tipo menu. Sconsigliato in HTML 4.0	TF
<noframes>	Inserisce contenuto alternativo per i browser che non supportano i frames.	STF

<noscript>	Inserisce contenuto alternativo per i browser che non supportano gli script.	STF
	Lista ordinata. può contenere solo l'elemento	STF
<p>	Definisce un paragrafo. Non può contenere altri elementi blocco, ma solo testo o elementi inline.	STF
<pre>	Definisce testo preformattato che mantiene le impostazioni dello spazio bianco.	STF
<table>	Definisce una tabella per l'inserimento di dati tabulari.	STF
<dd>	Descrizione di un termine in una lista di definizione.	STF
<dt>	Definizione di un termine in una lista di definizione.	STF
<frameset>	Definisce un frameset.	F
	Elemento di una lista ordinata o non ordinata.	STF
<tbody>	Definisce il corpo di una tabella. Con <thead> e <tfoot> serve a raggruppare le righe di una tabella.	STF
<td>	Cella di tabella.	STF
<tfoot>	Definisce il "piede" di una tabella.	STF
<th>	Intestazione di cella.	STF
<thead>	Definisce la testata di una tabella.	STF
<tr>	Riga di tabella.	STF

ELEMENTI INLINE

- Quando sono inseriti non danno origine a una nuova riga e possono contenere solo testo o altri elementi inline.

```
<p>Questo tasto è<b>grassetto</b></p>
```

- La parte delimitata dai tag ... non sarà posta su una nuova riga.
- Esempi come questo:

```
<b><p>Testo in grassetto</p></b>
```

- sono tollerati dai browser, ma non reggono al giudizio della validazione in quanto un elemento inline non può contenerne uno di tipo blocco.

ATTRIBUTI DI BODY

- Gli attributi per il testo, i link, il colore di sfondo e i margini dell'elemento <body> sono espressamente vietati solo nella DTD Strict, ma erano già considerati sconsigliati in HTML 4.0.

~~**alink**~~
~~**background**~~
~~**bgcolor**~~
~~**link**~~
~~**text**~~
~~**vlink**~~

OBSOLETI!!

VALIDARE LE PAGINE

- L'utilizzo delle dichiarazioni doctype consente l'uso dei validatori, applicazioni web che consentono di verificare se le pagine costruite soddisfano lo standard (DTD) dichiarato.
- Si può usare il validatore del W3C
- O il validatore alternativo del sito htmlhelp.com.
- Una volta validati i siti possono esporre l'icona che certifica la validazione.

CSS

I FOGLI DI STILE

- HTML serve informare il browser di quali sono le componenti necessarie a mostrare un documento e ad articolare il documento in blocchi semantici.
- I fogli di stile (Cascading Style Sheets) definiscono come i vari elementi che compongono un documento verranno resi su un media specifico (schermo, stampante, dispositivo mobile).

INSERIMENTO

- Un foglio di stile può essere esterno e interno:
  - È esterno un foglio di stile definito in un file separato dal documento.
  - Un foglio di stile si dice interno quando il suo codice è compreso in quello del documento.
- Un foglio esterno si carica:
  - Utilizzando l'elemento <LINK>.
  - Usando @import.
- Un foglio interno può essere compilato
  - utilizzando l'elemento <style>
  - Utilizzando l'attributo style di un singolo elemento

FOGLI COLLEGATI

- Uso dell'elemento <LINK>:
  - La dichiarazione va sempre collocata all'interno della sezione <HEAD> del documento (X)HTML:

```
<html>
    <head>
          <title>Inserire i fogli di stile in un documento</title>
          <link rel="stylesheet" type="text/css" href="stile.css">
    </head>
<body>
```

- L'elemento <link> presenta una serie di attributi di cui è importante spiegare significato e funzione:

Attributo	Descrizione
rel	descrive il tipo di relazione tra il documento e il file collegato. È obbligatorio. Per i CSS due sono i valori possibili: **stylesheet** e **alternate stylesheet**.
href	serve a definire l'URL assoluto o relativo del foglio di stile. È obbligatorio
type	identifica il tipo di dati da collegare. Per i CSS l'unico valore possibile è **text/css**. L'attributo è **obbligatorio**
media	con questo attributo si identifica il supporto (schermo, stampa, etc) cui applicare un particolare foglio di stile. Attributo **opzionale**.

FOGLIO COLLEGATI

- Un altro modo per caricare CSS esterni è usare la direttiva @import all'interno dell'elemento <style>:

```
<style>
    @import url(stile.css);
</style>
```

- Questo sistema è utile per fare in modo che solo i browser più recenti carichino il foglio di stile, garantendo così che lo interpretino correttamente e forzando i browser obsoleti (come Netscape 4. ad esempio) a rendere la pagina senza foglio di stile.

FOGLIO INCORPORATI

- I fogli incorporati sono quelli inseriti direttamente nel documento (X)HTML tramite l'elemento <style>. Anche in questo caso la dichiarazione va posta all'interno della sezione <head>:

```
<html>
  <head>
    <title>Inserire i fogli di stile in un documento</title>
      <style type="text/css">
        body {
          background: #FFFFCC;
        }
      </style>
  </head>
<body>
...
```

- <style> può avere due attributi:
  - type (obbligatorio)
  - media (opzionale)

FOGLIO IN LINEA

- L'ultimo modo per formattare un elemento con un foglio di stile consiste nell'uso dell'attributo 'style'.
- Esso fa parte della collezione di attributi (X)HTML definita Common: si tratta di quegli attributi applicabili a tutti gli elementi.
- La dichiarazione avviene a livello dei singoli tag contenuti nella pagina e per questo si parla di fogli di stile in linea. La sintassi generica è la seguente:

```
<elemento style="regole_di_stile">
```

REGOLE

- Un foglio di stile è costituito da una serie di regole che stabiliscono come un elemento (identificato da un selettore) viene reso su un media.

- Esempio:

```
p{font: 12px Verdana, arial;}
```

SINTASSI ABBREVIATA

- Ogni elemento presenta sui suoi quattro lati un certo margine rispetto a quelli adiacenti.
  - margin-top
  - margin-right
  - margin-bottom
  - margin-left
- La regola sarebbe questa:

```
div { margin-top: 10px;
      margin-right: 5px;
      margin-bottom: 10px;
      margin-left: 5px;
}
```

- Sintassi abbreviata:

```
div {margin: 10px 5px 10px 5px;}
```

SELETTORI

ELEMENTI

CSS 1.0

È il più semplice dei selettori. È costituito da uno qualunque degli elementi di (X)HTML.

```
h1 {color: #000000;}
p {background: white; font: 12px Verdana, arial, sans-serif;}
table {width: 200px;}
```

CSS 1.0

È possibile nei CSS raggruppare diversi elementi al fine di semplificare il codice.

```
h1 {background: white;}
h2 {background: white;}
h3 {background: white;}
```

```
h1, h2, h3 {background: white;}
```

CSS 2.1

```
* { color: black; }
```

ELEMENTI

CSS 1.0

Elementi che nella struttura ad albero di un documento siano discendenti di un altro elemento specificato nella regola.

```
div p {color: black;}
p strong {color: red;}
```

- Nel primo esempio verranno selezionati tutti i paragrafi (<p>) discendenti di elementi . Nel secondo tutti gli elementi che si trovino all'interno di un paragrafo.

CSS 2.1

Elementi che nella struttura ad albero di un documento siano i figli diretti di un elemento.

```
body > p {color: black;}
<body>
  <p>Primo paragrafo</p>
  <div>
    <p>Secondo paragrafo</p>
  </div>
</body>
```

IE7+

ELEMENTI

- Elementi che nella struttura ad albero di un documento siano discendenti di un altro elemento specificato nella regola.

CSS 1.0

```
div p {color: black;}
p strong {color: red;}
```

  - Nel primo esempio verranno selezionati tutti i paragrafi (<p>) discendenti di elementi . Nel secondo tutti gli elementi che si trovino all'interno di un paragrafo.

CSS 2.1

- Elementi che nella struttura ad albero di un documento siano i figli diretti di un elemento.

IE7+

```
body > p {color: black;}
<body>
  <p>Primo paragrafo</p>
  <div>
    <p>Secondo paragrafo</p>
```

ELEMENTI

CSS 2.1

Elementi che nel codice del documento siano immediatamente vicini (adiacenti) ad un altro.

IE7+

```
h1 + p {color: red;}
<h1>Titolo</h1>
<p>Primo paragrafo</p>
<p>Secondo paragrafo</p>
```

– In base a questa dichiarazione solo il primo dei due paragrafi avrà il testo rosso..

SELEZIONE DEGLI ELEMENTI IN BASE AI LORO ATTRIBUTI

IE7+

CSS 2.1

Attributo semplice

```
input [ id ] {background: red;}
```

– applicherà uno sfondo rosso a tutti gli elementi input per cui sia stato impostato un attributo id, a prescindere dal valore di id.

CSS 2.1

Attributo con valore

```
input [ id = "text" ] { backgorund: red; }
```

– applicherà un sfondo rosso a tutti gli elementi input che abbiano come valore dell'attributo id "text".

CSS 3.0

Attributo il cui valore contiene una stringa

```
img [ alt*= "foto" ] {margin: 10px;}
```

– La regola applicherà un margine di 10px a tutte le immagini in cui l'attributo alt contiene la stringa "foto".

CSS 3.0

Attributo il cui valore inizia con una stringa

```
img [ alt |= "figura" ] {margin: 10px;}
```

– selezionerà tutte le immagini in cui l'attributo alt inizia con la stringa "figura".

CLASSI E ID

- In questa pagina abbiamo assegnato al paragrafo l'attributo class="testorosso":
  ....
- Possiamo ora creare un regola e assegnargli il nome testorosso:

```
.testorosso {
   font: 12px arial, Helvetica, sans-serif;
   color: #FF0000;
}
```

- In un documento potrò avere senza problemi questa situazione:
  ....
  ....
  <table class="testorosso">...</table>
  ....
- E l'elemento seguirà la regola definita nella classe testorosso

CLASSI E ID

CSS 1.0

Per definire una classe si usa far precedere il nome da un semplice punto:

```
.testorosso {
   font: 12px arial, Helvetica, sans-serif;
   color: #FF0000;
}
```

CSS 1.0

Se scriviamo:

```
p.testorosso {color: red;}
```

- lo stile verrà applicato solo ai paragrafi che presentino l'attributo class="testorosso".

CSS 1.0

Sono possibili dichiarazioni di classi multiple:

```
p.testorosso.grassetto {color:red; font-weight:bold;}
```

- Questa regola applicherà gli stili impostati a tutti gli elementi in cui siano presenti (in qualunque ordine) i nomi delle classi definiti nel selettore.

IE7+

CLASSI E ID

CSS 1.0

La sintassi di un selettore ID è semplicissima. Basta far precedere il nome dal simbolo di cancelletto #:

```
#titolo {
  color: blue;
}
```

- assegniamo il colore blue all'elemento che presenti questa definizione:

```
<h1 id="titolo">...</h1>
```

CSS 1.0

Come per le classi è possibile usare una sintassi con elemento:

```
p#nome_id {
  color: red;
}
```

- Ma non ha senso perché l'id per sua natura dovrebbe essere unico.

PSEUDO-CLASSI

- Una pseudo-classe non definisce un elemento ma un particolare stato di quest'ultimo. In buona sostanza imposta uno stile per un elemento al verificarsi di certe condizioni.
- A livello sintattico le pseuso-classi non possono essere mai dichiarate da sole, ma per la loro stessa natura devono sempre appoggiarsi ad un selettore.

```
a:link {color: blue;}
```

  - La regola vuol dire: i collegamenti ipertestuali (<a>) che non siano stati visitati (:link) avranno il colore blue.

PSEUDO-CLASSI

CSS 2.1

:first-child

- Seleziona e formatta un elemento che si trovi ad essere il primo elemento figlio di un altro elemento.

IE 7+

CSS 1.0

:link

- Si applica solo all'elemento (X)HTML <a> che abbia anche l'attributo href. Definisce lo stile per questo elemento quando il collegamento punta ad un sito o ad una pagina non ancora visitati.

CSS 2.1

:hover

- Definisce lo stile per questo elemento quando il puntatore è sopra all'elemento.

IE 7+

CSS 1.0

:visited

- Si applica solo all'elemento (X)HTML <a> che abbia anche l'attributo href. Definisce lo stile per questo elemento quando il collegamento punta ad un sito o ad una pagina già visitata.

PSEUDO-CLASSI

CSS 1.0 :first-letter

- Imposta lo stile della prima lettera di un elemento contenente del testo.

IE 7+

CSS 1.0 :first-line

- Imposta lo stile della prima riga di un elemento contenente del testo.

IE 7+

CSS 2.1 :before

```
blockquote:before {content:“Nota"}
```

IE 8+

CSS 2.1 :after

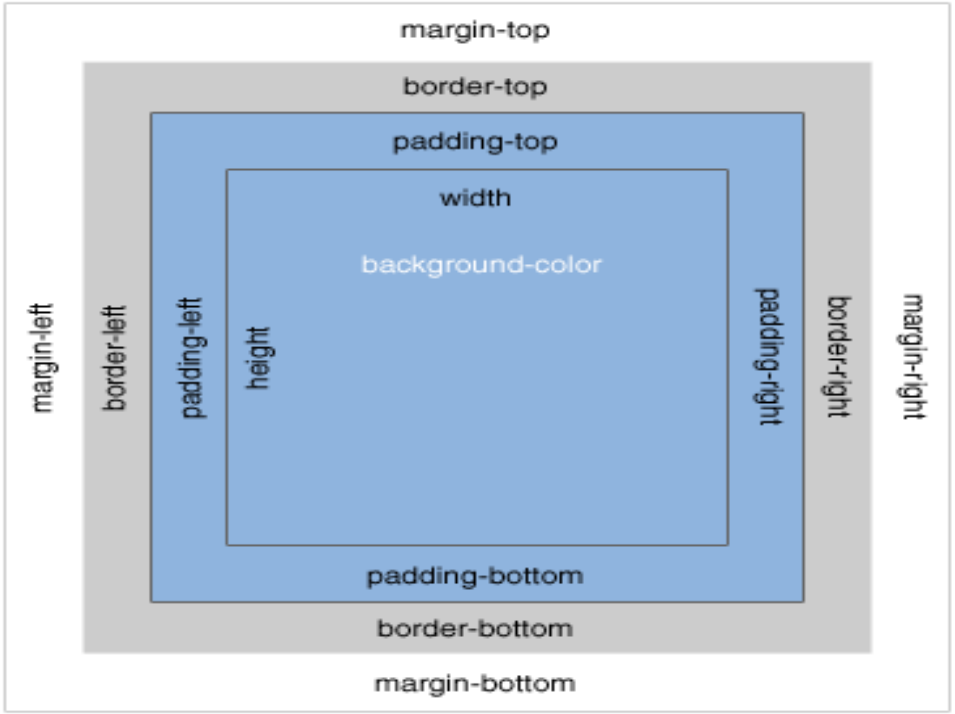
```
blockquote:after {content:“Nota"}
```

IE 8+

DETTAGLI

- http://kimblim.dk/css-tests/selectors/

PROPRIETÀ DEI BLOCCHI



BACKGROUND

- **background-color**
  - Definisce il colore di sfondo di un elemento. Questa proprietà non è ereditata.

```
selettore {background-color: #FFF;}
selettore {background-color: transparent;}
```

- **background-image**
  - Definisce l'URL di un'immagine da usare come sfondo di un elemento. Questa proprietà non è ereditata.

```
selettore { background-image: url(valore); }
selettore { background-image: none}
```

- **background-repeat**
  - Consente di definire la direzione in cui l'immagine di sfondo viene ripetuta.

```
selettore {background-repeat: valore;}
```

  - Valori: repeat, repeat-x, repeat-y, no-repeat

- **background-attachment**

```
selettore {background-attachment: valore;}
```

  - Valori: scroll, fixed.

BACKGROUND

- **background-position**
  - Definisce il punto in cui verrà piazzata un'immagine di sfondo.

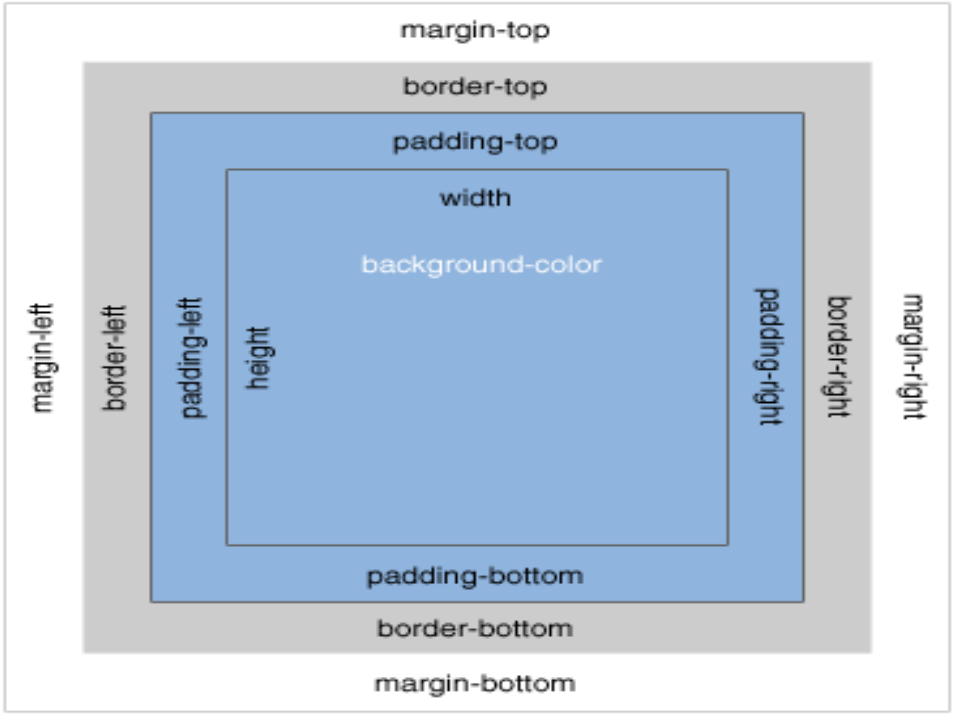
```
selettore {background-position: valoreOriz | valoreVert;}
```

  - Valori: valori in percentuale, valori espressi con unità di misura, parole chiave top, left, bottom, right.
- **background**
  - Per essere valida, la dichiarazione non deve contenere necessariamente riferimenti a tutte le proprietà viste finora, ma deve contenere almeno la definizione del colore di sfondo.

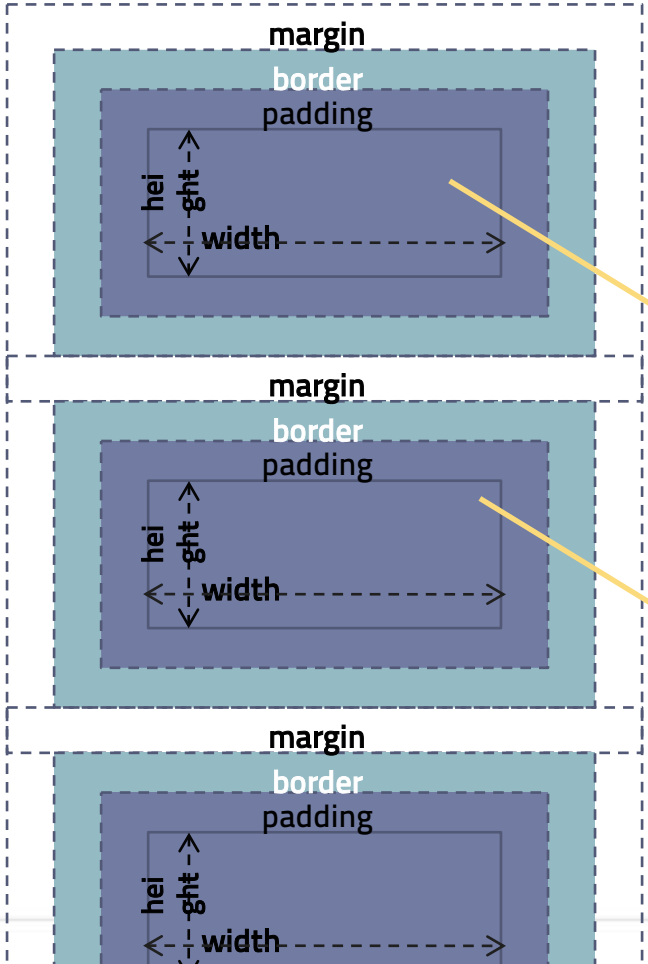
```
selettore { background: background-color
                        background-image
                        background-repeat
                        background-attachment
                        background-position;
          }
```

FLOAT

PROPRIETÀ DEI BLOCCHI



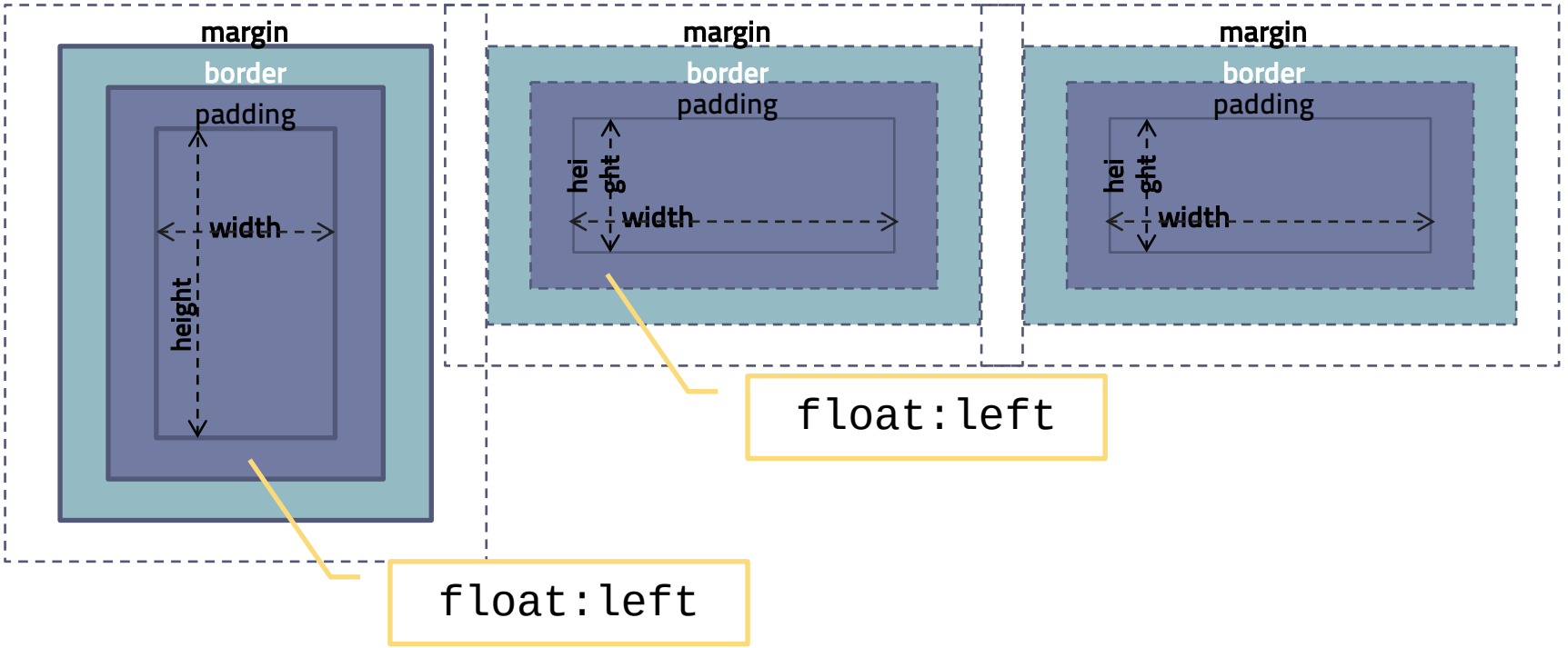
FLOAT



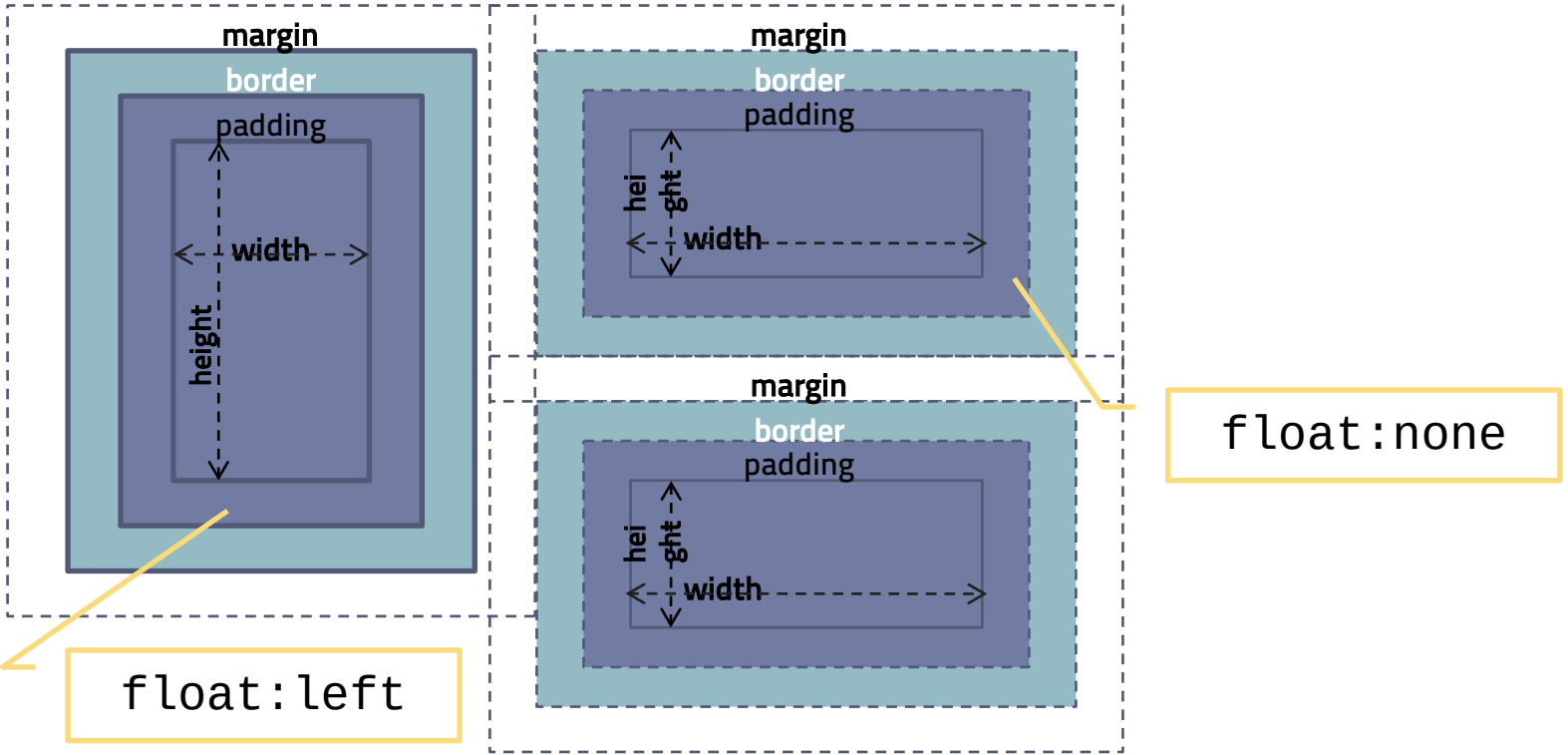
`float:none`

`float:none`

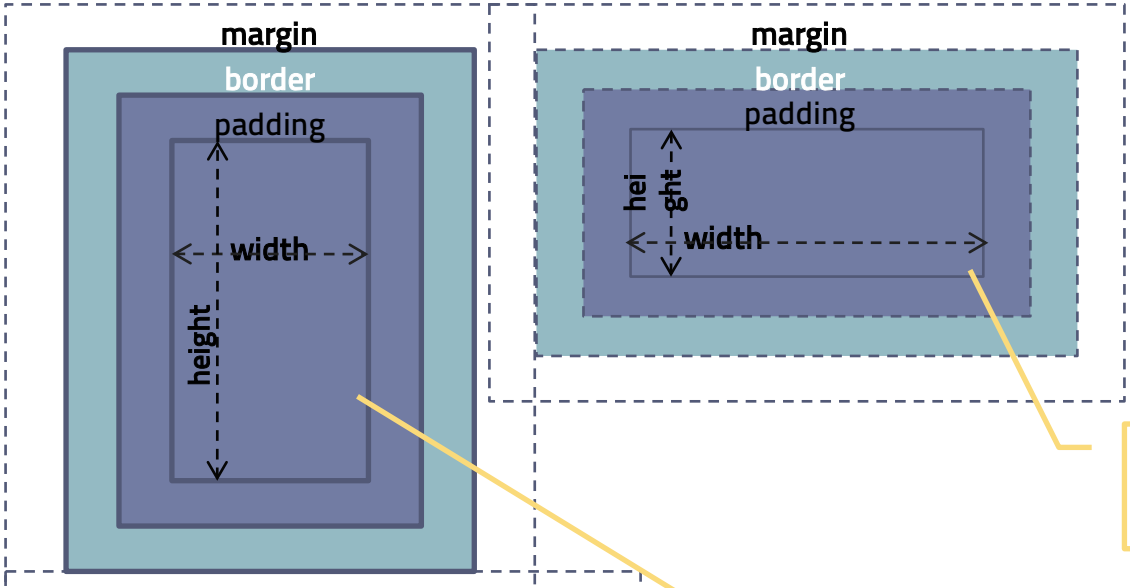
FLOAT


margin
border
padding
width
height
margin
border
padding
hei
ght
width
margin
border
padding
hei
ght
width
float:left
float:left

FLOAT


margin
border
padding
width
height
margin
border
padding
hei
ght
width
margin
border
padding
hei
ght
width
float:none
float:left

FLOAT




```
float:none;
clear:both;
```

```
float:left
```