Linguaggi

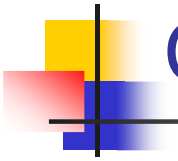
Corso di Laurea Ingegneria Informatica (M-Z)
A.A. 2006-2007

Alessandro Longheu
http://www.diit.unict.it/users/alongheu
alessandro.longheu@diit.unict.it

Cenni su DHTML e XML

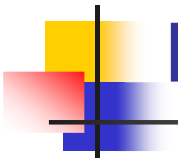
DHTML

- DHTML=DOM+Javascript+CSS
- DOM = Document Object Model (modello ad oggetti della pagina WEB)
- Javascript (EcmaScript) linguaggio di programmazione per la scrittura degli script
- CSS = Cascading Style Sheet (fogli di stile a cascata)



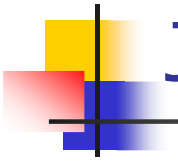
Cos'è il DOM

- Il DOM definisce gli oggetti di una pagina WEB
- Ogni oggetto ha delle proprietà che lo descrivono.
- Ogni oggetto ha dei metodi che permettono di inviare dei messaggi allo stesso per comandargli di fare qualcosa
- notazione
- standardizzazione



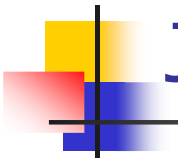
DOM – Oggetti predefiniti

- Window (parent, top, self, opener)
  - frames[i]
    - document
      - images[i]
      - links[i]
      - forms[i]
        - elements[i]
        - button
        - checkbox
        - image
        - radio
        - reset
        - select
        - options[i]
        - submit
        - text/textarea
      - location



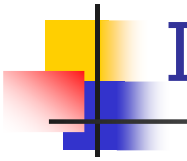
Javascript

- Ha la funzione di definire funzioni che agiscono sugli oggetti che compongono la pagina WEB (DOM)
- Javascript è un linguaggio di programmazione, ideato da Netscape (non da SUN); non ha in tal senso niente in comune con Java se non il nome
- In pratica, è un linguaggio interpretato, object-based piuttosto che object-oriented, nel senso che non permette una completa definizione di nuovi tipi di dato (classi, ereditarietà ecc.), piuttosto opera sull'insieme limitato di oggetti predefiniti che rappresentano le diverse componenti della pagina



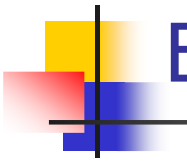
Javascript

- Il comportamento dinamico degli oggetti nella pagina non e' altro che una lista di istruzioni del tipo proprieta'=valore e/o messaggi del tipo metodo() e puo' essere predefinito da chi ha scritto la pagina in modo che avvenga quando la pagina e' caricata, oppure iniziato da qualcosa che l'utente compie.
- In questo caso si dice che si e' avuto un certo evento e chi scrive la pagina deve indicare cosa deve succedere quando l'evento accade



Il programma più semplice!!

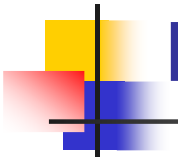
```
<title> Benvenuti nel mondo di Javascript
  </title>
<SCRIPT language=JavaScript>
  document.write(Benvenuti in Javascript );
</SCRIPT>
```

Eventi

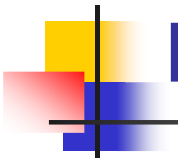
- Oggetti di tipo text per leggere i dati e per scrivere il risultato

```
<title>Somma di due numeri</title>
<form name="somma">
    <input name="operando1" > +
    <input name="operando2" > =
    <input name="risultato" value="0"> <p>
    <input type="button" value="Fai la somma!" onClick=
      "somma.risultato.value=Number(somma.operando1.value)
      + Number(somma.operando2.value)"> </form>
```



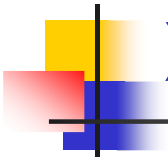
Esempio funzione

```
<head>
   <title>Calcola la media di una serie di numeri</title>
   <script language=javascript>
        function calcolamedia(lista){
                dati = lista.split(" ")
                if (dati.length > 0) {
                        var somma = 0;
                        for (var i = 0; i < dati.length; i++){
                                somma = somma+Number(dati[i]) }
                        return (somma / dati.length);
                }
        }
   </script>
</head>
```

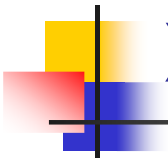
Esempio funzione

```
<body>
   <form name="media">
        Lista di valori <input name="valori"> <br>
        Risultato <input name="risultato"> <p>
        <input type=button value="Calcola la media!"
                onClick = "media.risultato.value =
                        calcolamedia(media.valori.value)">
</form>
</body>
```



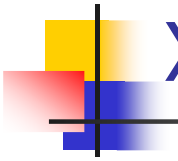
XML

- L'eXtensible Markup Language (XML) è un linguaggio nato nel 1998 per offrire un'alternativa più robusta e più structure oriented dell'HTML.
- A differenza dell'HTML, è possibile definire un proprio insieme di tag (DTD) ammessi per i propri documenti XML.
- I tag XML dovrebbero consentire di descrivere la struttura dei dati, e non la loro rappresentazione su browser (come avviene attualmente con l'HTML).
- Un documento XML è composto da un insieme di entità, porzioni fisiche di un unico documento logico. Le entità possono essere:
  - interne di tipo testo, quando ad esempio una porzione di testo si ripete nello stesso file;
  - esterne di tipo testo, quando su altri file si trovano porzioni logiche dello stesso documento;
  - esterne di tipo binario, ad esempio immagini.



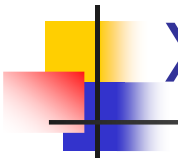
XML – DTD

- Il DTD stabilisce le regole formali per la struttura del documento, contiene infatti non solo l'elenco dei tag ammessi ed i loro attributi, ma anche la loro organizzazione gerarchica ammessa nel documento, ad esempio sequence (,), choice (|), quantity control (*,+);
- Il DTD non è obbligatorio, ma averlo consente di usare un XML processor per verificare se un dato documento è o no conforme alle regole definite nel DTD stesso.
- Un XML processor verifica successivamente:
  - se un documento è well-formed (es. tag di chiusura presenti);
  - se un documento è valido (conforme ad un DTD);
- Un DTD può essere interno o esterno ad un documento. Se esterno, può essere usato per più documenti.
- Sono stati scritti DTD per la rappresentazione e gestione delle espressioni matematiche (MathML) e chimiche (ChemML).



XML – DTD

```
<!  Element  InstanceTechnology  (  ListAttrTechnology,  Rules,
Hierarchy) >
<! Element ListAttrTechnology ( Name, UserDefFields ) >
<! Element Name ( #PCDATA ) >
<! Element Rules ( Rule* ) >
<! Element Rule empty >
<! Attlist Rule     href CDATA #REQUIRED
                    xml:link CDATA #FIXED “simple”
                    show ( embed | replace | new ) embed
       actuate ( user | auto ) auto >
<! Element Hierarchy (Process) >
<! Attlist Hierarchy  xml:link CDATA #FIXED “simple” >
<! Element Processes (locator+) >
<! Attlist Processes xml:link CDATA #FIXED “extended” >
<! Element locator empty >
<! Attlist locator href CDATA #REQUIRED
show (embed | replace | new ) embed
actuate ( user | auto ) auto >
```

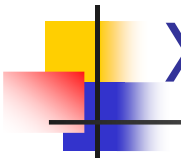
XML – uso di DTD

Esempio di file XML conforme al DTD precedente:

```
<IstanceTechnology>
       <ListAttribTechnology>
       Name=FS551
       ...
       </ListAttribTechnology>
<Rules>
          <Rule  xml:link=”simple”  href=//…#id(R1)  show=embed
actuate=auto />
</Rules>
<Hierarchy xml:link=”simple” href=http://…#id(SubT) show=embed
actuate=auto >
<Process xml:link=”extended”>
<locator href=”http://...#id(……)show=”embed” actuate=”auto”/>
</Process>
</Hierarchy>
```

XML – XLL

- L'XML fornisce un complesso meccanismo per la gestione dei link, definiti dallo standard XML Linking Language (XLL), tramite cui:
- specificare quando un link deve attivarsi (manualmente o meno);
  - specificare cosa effettuare a seguito dell'attivazione (aprire una nuova finestra, sostituire la precedente, o sostituire l'elemento iniziale;
  - collegare l'elemento iniziale a più risorse (link semplici o estesi), specificando la regola per ogni singolo locator;
  - definire una gerarchia di documenti tramite l'X-pointer, così da poter supportare link assoluti (es puntare alla radice, puntare al terzo figlio), o relativi alla gerarchia (es puntare al precedente documento), eventualmente completandoli con string matching, ad esempio:
  - ROOT.CHILD(3,CHAP).STRING(7,"Napoleone")
    trova la settima occorrenza della stringa "Napoleone" all'interno del terzo capitolo del documento puntato.



XML – XSL

- L'XML fornisce un complesso meccanismo per la rappresentazione dei documenti, l'eXtensible Stylesheet Language (XSL).
- l'XSL si basa sulle template rules, regole che consentono di intercettare una sequenza richiesta di tag all'interno del documento XML da formattare e di associarvi un opportuno oggetto di formattazione.
- L'XSL processor prende in ingresso il documento XML da rappresentare (source tree), vi applica le template rules e genera il documento con i soli oggetti di formattazione (output tree). Esso viene dato ad un formatter in cui viene dettagliatamente descritto come rappresentare ogni oggetto di formattazione. Il formatter deve essere incluso nei browser.