- È più facile modificare le esigenze in funzione del programma che viceversa.
- Non c'è linguaggio in cui sia difficile scrivere cattivi programmi.
- Le tue esigenze si espandono fino a tutte quelle che può soddisfare il tuo programma + 1.

LEGGI DEL PROGRAMMATORE
(Leggi di Murphy applicate all'informatica)

Tecniche Multimediali

Corso di Laurea in «Informatica» - aa 2010-2011

Prof. Giorgio Poletti – giorgio.poletti@unife.it

DTD – Vocabolario e Sintassi

- DTD (**D**ocument **T**ype **D**efinition)
  - Descrive
    - tag (**elementi**) utilizzabili in un file XML
    - relazioni tra tag utilizzabili in un file XML
    - definizione e descrizione degli **attributi** degli elementi del file XML
  - Elementi della sintassi
    - <!ELEMENT>
    - <!ATTLIST>
    - <!ENTITY>
    - <!NOTATION>
    - <!-- --> *commento*

DTD – Vocabolario e Sintassi

- DTD dichiarazione
  - **DTD interno**

<!DOCTYPE root-element [dichiarazione_degli_elementi]>

  - *Esempio*

```
<?xml version="1.0"?>
<!DOCTYPE __root [
      <!ELEMENT _root (from, to+, cc*)>
            <!ELEMENT from (#PCDATA)>
            <!ELEMENT to (#PCDATA)>
            <!ELEMENT cc (#PCDATA)>
]>
<_root>
      -----
</__root>
```

DTD – Vocabolario e Sintassi

- DTD dichiarazione
  - **DTD esterno SYSTEM**

<!DOCTYPE root-element SYSTEM 'DTD_location'>

(è un DTD di tipo «privato» in utilizzo a un utente o gruppo di utenti, dove DTD_location è un indirizzo relativo o un URL assoluto)

- *Esempio DTD «privato»*

```
<?xml version="1.0"?>
<!DOCTYPE __root  SYSTEM "esempio.dtd">
<__root>

  ---------------

</__root>
```

DTD – Vocabolario e Sintassi

- DTD dichiarazione
  - **DTD esterno PUBLIC**

<!DOCTYPE root-element PUBLIC 'DTD_name' 'DTD_location'>

(è un DTD di tipo «pubblico» pensato per utilizzi ampi (for broad use)
"DTD_location" è utilizzato per per trovare il public DTD se non è identificato dal "DTD_name".

"DTD_location": "prefisso//proprietario_del_DTD//descrizione_DTD//ISO 639_identificatore_lingua"

  - *Esempio DTD «pubblico»*

```
<?xml version="1.0"?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0
Transitional//EN"  "http://www.w3.org/TR/REC-tml40/loose.dtd">
<__root>  ----  </__root>
```

DTD – Vocabolario e Sintassi

- DTD dichiarazione
  - **DTD esterno PUBLIC**

Prefisso	Descrizione
ISO	DTD ISO standard. Tutti gli ISO standards sono approvati.
+	DTD non-ISO standard approvato.
-	DTD non-ISO standard e non approvato

DTD – Vocabolario e Sintassi

- DTD <!ELEMENT>
  - *Sintassi 1:* <!ELEMENT nome 'categoria'>
  - *Sintassi 2: <!ELEMENT nome* ('contenuto')>
    - Ogni elemento **può** essere dichiarato 1 volta sola
    - I nomi degli elementi sono case sensitivi, non possono contenere spazi e caratteri di controllo (. : ; ! = @ ? #) e devono iniziare con un carattere alfanumerico

Tipi	Definizione	Esempio
EMPTY	Elemento vuoto ***(categoria)***.	<!ELEMENT hl EMPTY> ***<hl />***
#PCDATA (**P**arsed **C**haracter **DATA**)	Elemento analizzato con dati solo di tipo carattere ***(categoria)***.	<!ELEMENT p (#PCDATA)>
Elementi annidati (Children Elements)	Elementi che sono contenitori per altri elementi compresenti (,) o alternativi (|)***(contenuto)***.	<!ELEMENT nodo_padre (child1,child2,(child3|child4))>
ANY	Elemento sicuramente non vuoto ma di cui non si sa il contenuto (testo, elementi o entrambi) ***(categoria)***.	<!ELEMENT nota ANY>

DTD – Vocabolario e Sintassi

- DTD <!ELEMENT>
  - *Sintassi 1:* <!ELEMENT nome 'categoria'>
  - *Sintassi 2: <!ELEMENT nome* ('contenuto')>
    - Ogni elemento **può** essere dichiarato 1 volta sola
    - I nomi degli elementi sono case sensitivi, non possono contenere spazi e caratteri di controllo (. : ; ! = @ ? #) e devono iniziare con un carattere alfanumerico

Tipi	Definizione	Esempio
Elemento misto (Mixed Element)	elemento che può contenere testo o altri elementi in alternativa ***(categoria)***.	<!ELEMENT MixElement (#PCDATA | child1|chil2|...)>

DTD – Vocabolario e Sintassi

- DTD <!ELEMENT> occorrenza
  OCCURRENCE OF AN ELEMENTS

Dichiarazione	Definizione	Esempio
BLANK	L'elemento child **deve** comparire una e una sola volta all'interno dell'elemento padre **(1)**	<!ELEMENT padre (child)>
+	L'elemento child **deve** comparire una più volte all'interno dell'elemento padre **(1-N)**	<!ELEMENT padre (child**+**)>
?	L'elemento child **può** comparire al massimo una volta all'interno dell'elemento padre **(0-1)**	<!ELEMENT padre (child**?**)>
*****	L'elemento child **può** più volte all'interno dell'elemento padre **(0-N)**	<!ELEMENT padre (child*****)>

DTD – Vocabolario e Sintassi

- DTD <!ATTLIST>

  ATTRIBUTI: TIPI E OCCORRENZE

- *Sintassi*

  <**!ATTLIST** ***nome_elemento*** *nome_attributo (*tipo*) valori_predefiniti*
  *nome_attributo (*tipo*) valori_predefiniti*>

TIPO	Definizione
Stringa	Testo che **NON** viene "processato" dall'analizzatore sintattico (parola chiave **CDATA**)
Token	Le opzioni *tokenizzate* permettono di limitare i valori permessi per gli attributi
Enumerativo	Permettono di definire un elenco di valori possibili per l'attributo definito

OCCORRENZA	Definizione
#REQUIRED	Identifica un attributo obbligatorio
#FIXED *value*	Dichiara costante il valore di un attributo; se il valore è diverso da quello dichiarato, il documento non è valido
#IMPLIED	Identifica un attributo facoltativo; se l'attributo non è presente nell'elemento, il motore di parsing può usare qualsiasi valore (se necessario)

DTD – Vocabolario e Sintassi

- DTD <!ATTLIST>
  STRINGA

TIPO	Definizione
Stringa	Parola chiave **CDATA**: la stringa può contenere tutti i caratteri tranne **< > &** **<!ATTLIST mail data CDATA #REQUIRED>** **<!ATTLIST mail criptata CDATA #IMPLIED>** **<!ATTLIST mail oggetto CDATA 'N/A'>**
Esempio in XML	
• <mail data='20/04/2011' criptata='YBBX' oggetto='N+/A'> … </mail> • <mail data='20/04/2011' oggetto='N/A'> … </mail> • <mail data='20/04/2011' oggetto='Mail di prova'> … </mail>	

DTD – Vocabolario e Sintassi

- DTD <!ATTLIST>
  TOKEN

TIPO	Definizione	
Token	ID	*valore univoco*
	IDREF	*valore punta ad un elemento con attributo **ID***
	IDREFS	*valore punta ad una serie di elementi con attributo **ID** (separati da uno spazio)*
	NMTOKEN	*valore è un nome **XML valido***
	NMTOKENS	*valore è una lista di nomi **XML validi***
	ENTITY	*valore è un entità (**ENTITY**)*
	ENTITIES	*valore è una lista di entità (**ENTITY**)*

Esempio in XML

- **(ID)** <studente matricola="51947">Giorgio Poletti</studente>
- **(IDREF)** <corso>
  <docente docId="D12345">Giorgio Poletti</docente>
  <studente matricola="099999" doc1="D12345">Mario Rossi</studente>
  </corso>

DTD – Vocabolario e Sintassi

- DTD <!ATTLIST>
  TOKEN

TIPO	Definizione	
Token	ID	*valore univoco*
	IDREF	*valore punta ad un elemento con attributo **ID***
	IDREFS	*valore punta ad una serie di elementi con attributo **ID** (separati da uno spazio)*
	NMTOKEN	*valore è un nome **XML valido***
	NMTOKENS	*valore è una lista di nomi **XML validi***
	ENTITY	*valore è un entità (**ENTITY**)*
	ENTITIES	*valore è una lista di entità (**ENTITY**)*
Esempio in XML		

- **(IDREFS)**

```
<corso>
  <docente  docId="D12345">Giorgio Poletti</docente>
  <docente  docId="D12346">Ugo Bianchi</docente>
  <studente matricola="099999" doc1="D12345 D12346">Mario Rossi</studente>
</corso>
```

DTD – Vocabolario e Sintassi

DTD <!ATTLIST>

TOKEN

TIPO	Definizione
Token	ID *valore univoco* IDREF *valore punta ad un elemento con attributo **ID*** IDREFS *valore punta ad una serie di elementi con attributo **ID** (separati da uno spazio)* NMTOKEN *valore è un nome **XML valido*** NMTOKENS *valore è una lista di nomi **XML validi*** ENTITY *valore è un entità (**ENTITY**)* ENTITIES *valore è una lista di entità (**ENTITY**)*

Esempio in XML

- **(NMTOKEN)** *attributio con valori che possono contenere solo lettere, cifre, punti (.), dash (-), underscore (_) e punti e virgola (;) , contrariamente a CDATA che accetta tutti i caratteri*

```
<corso>
  <docente  docId="D12345">Giorgio Poletti</docente>
  <docente  docId="D12345 A">Mario Rossi</docente>  valore errato!
</corso>
```

DTD – Vocabolario e Sintassi

DTD <!ATTLIST>

TOKEN

TIPO	Definizione
Token	ID *valore univoco* IDREF *valore punta ad un elemento con attributo **ID*** IDREFS *valore punta ad una serie di elementi con attributo **ID** (separati da uno spazio)* NMTOKEN *valore è un nome **XML valido*** NMTOKENS *valore è una lista di nomi **XML validi*** ENTITY *valore è un entità (**ENTITY**)* ENTITIES *valore è una lista di entità (**ENTITY**)* NOTATION *valore è il nome di una notazione (NOTATION)* xml: valore è un valore **XML predefinito**

Esempio in XML

- **(NMTOKENS)** *attributi con valori che possono contenere solo lettere, cifre, punti (.), dash (-), underscore (_) e punti e virgola (;) separati da spazio*

```
<corso>
  <docente  docId="D12345">Giorgio Poletti</docente>
  <docente  docId="D12345 A">Mario Rossi</docente>  valore corretto!
</corso>
```

DTD – Vocabolario e Sintassi

DTD <!ATTLIST>

TOKEN

TIPO	Definizione
Token	ID *valore univoco* IDREF *valore punta ad un elemento con attributo **ID*** IDREFS *valore punta ad una serie di elementi con attributo **ID** (separati da uno spazio)* NMTOKEN *valore è un nome **XML valido*** NMTOKENS *valore è una lista di nomi **XML validi*** ENTITY *valore è un entità (**ENTITY**)* ENTITIES *valore è una lista di entità (**ENTITY**)* NOTATION *valore è il nome di una notazione (NOTATION)* xml: valore è un valore **XML predefinito**
Esempio in XML	

- **(ENTITY)** *valore che viene utilizzato per fare riferimento a dati e si usano come abbreviazione o per dati presenti in loacazioni esterne; il primo carattere del **valore** di entità deve essere una lettera, underscore (_) o due punti (:)*

```
<!ATTLIST docente  image ENTITY #REQUIRED>
<!ENTITY  face SYSTEM "http://www.unife.it/images/face.gif">
        <docente  image="face">Giorgio Poletti</docente>
```

DTD – Vocabolario e Sintassi

DTD <!ATTLIST>

TOKEN

TIPO	Definizione
Token	ID *valore univoco* IDREF *valore punta ad un elemento con attributo **ID*** IDREFS *valore punta ad una serie di elementi con attributo **ID** (separati da uno spazio)* NMTOKEN *valore è un nome **XML valido*** NMTOKENS *valore è una lista di nomi **XML validi*** ENTITY *valore è un entità (**ENTITY**)* ENTITIES *valore è una lista di entità (**ENTITY**)*

Esempio in XML

- **(ENTITIES)** *consente di specificare più entità separate da uno spazio*

```
<!ATTLIST docente  image ENTITIES #REQUIRED>
<!ENTITY  face SYSTEM "http://www.unife.it/images/face.gif">
<!ENTITY  uniLogo SYSTEM "http://www.unife.it/images/logo.gif">
        <docente  image="face uniLogo">Giorgio Poletti</docente>
```

DTD – Vocabolario e Sintassi

DTD <!ATTLIST>

ENUMERATIVO

TIPO	Definizione
ENUMERATIVO	NOTATION *valore è il nome di una notazione (NOTATION)* Enumerated

Esempio in XML

- **(NOTATION)** *utili quando il testo deve essere interpretato in modo particolare, per esempio, da un'altra applicazione; il primo carattere del nome deve essere una lettera, underscore (_) o due punti (:)*

```
<!NOTATION xml PUBLIC «XML 1.0">
<!ATTLIST esercizio lang NOTATION (xml) #REQUIRED>
<esercizio lang=«xml">Alcune istruzioni XML</esercizio>
```

DTD – Vocabolario e Sintassi

DTD <!ATTLIST>

ENUMERATIVO

TIPO	Definizione
ENUMERATIVO	NOTATION *valore è il nome di una notazione (NOTATION)* Enumerated

Esempio in XML

- **(Enmerated)** *permettono fare una scelta tra una lista di valori; il primo carattere di ogni valore deve essere una lettera, underscore (_) o due punti (:)*

```
<!ATTLIST prova giudizio  (insuff|suff| buono|ottimo) #REQUIRED>
<prova giudizio='suff'> Prova di Maggio </prova>
```

DTD – Vocabolario e Sintassi

- DTD <!NOTATION>
  - *Sintassi possibili*
    - *<!NOTATION nome SYSTEM 'URI'>*
    - *<!NOTATION nome PUBLIC 'public_ID'>*
    - *<!NOTATION nome PUBLIC 'public_ID' 'URI'>*
  - *sono utilizzati per identificare il formato di entità non analizzate (dati non-XML), gli elementi con un attributo di notazione, o specifiche istruzioni di elaborazione*
  - ***URI** è un URL in cui si trova la notazione esterna*
  - ***public_ID** può essere usato dal processor XML come alternativa al URI. Se non si trova il public:id viene utilizzata l'URI*

DTD – Vocabolario e Sintassi

- DTD <!ENTITY>
  - *Sintassi possibili*
    - *<!ENTITY nome 'entity value'>*
    - *<!ENTITY nome SYSTEM 'URI'>*
    - *<!ENTITY nome PUBLIC 'public_ID'>*
  - *dati di riferimento a entità utiklizzate come abbreviazione o che si possono trovato in una locazione esterna; servono a ridurre l'immissione di informazioni ripetitive e anche per consentire un editing più facile (riducendo il numero di occorrenze di dati da modificare). Ci sono due tipi di dichiarazioni di entità: dichiarazioni di entità **GENERALE**, e le dichiarazioni **PARAMETRICHE***

Dichiarazione	Tipo	Esempio
GENERALE	INTERNAL (*PARSED*) *(generalmente si riferisce a testo)*	<!ENTITY nome "entity_value"> (***entity value**: qualsiasi carattere che NON sia '&', '%' o ' " ', una referenza a una entità ('%nome') o a un carattere Unicode*) **<!ENTITY gp "Giorgio Poletti"> <docente>&gp</docente>**

DTD – Vocabolario e Sintassi

- DTD <!ENTITY>
  - *Sintassi possibili*
    - *<!ENTITY nome 'entity value'>*
    - *<!ENTITY nome SYSTEM 'URI'>*
    - *<!ENTITY nome PUBLIC 'public_ID'>*
  - *dati di riferimento a entità utiklizzate come abbreviazione o che si possono trovato in una locazione esterna; servono a ridurre l'immissione di informazioni ripetitive e anche per consentire un editing più facile (riducendo il numero di occorrenze di dati da modificare). Ci sono due tipi di dichiarazioni di entità: dichiarazioni di entità **GENERALE**, e le dichiarazioni **PARAMETRICHE***

Dichiarazione	Tipo	Esempio
GENERALE	EXTERNAL (*PARSED*) *(generalmente si riferisce a testo)*	<!ENTITY nome SYSTEM 'URI'> **<!ENTITY c SYSTEM "http://www.unife.it/corsi.xml">** **<!ENTITY cext PUBLIC "-//W3C//TEXT copyright//EN"** **"http://www.miur.it/corsiExt.xml">** **<docente>&c</docente>** **<docente>&cext</docente>**

DTD – Vocabolario e Sintassi

- DTD <!ENTITY>
  - *Sintassi possibili*
    - *<!ENTITY nome 'entity value'>*
    - *<!ENTITY nome SYSTEM 'URI'>*
    - *<!ENTITY nome PUBLIC 'public_ID'>*

Dichiarazione	Tipo	Esempio
GENERALE	EXTERNAL (*PARSED*) *(generalmente si riferisce a testo)*	<!ENTITY nome SYSTEM 'URI'> **<!ENTITY c SYSTEM "http://www.unife.it/corsi.xml">** **<!ENTITY cext PUBLIC "-//W3C//TEXT copyright//EN"** **"http://www.miur.it/corsiExt.xml">** **<docente>&c</docente>** **<docente>&cext</docente>**

DTD – Vocabolario e Sintassi

- DTD <!ENTITY>
  - *Sintassi possibili*
    - *<!ENTITY nome 'entity value'>*
    - *<!ENTITY nome SYSTEM 'URI'>*
    - *<!ENTITY nome PUBLIC 'public_ID'>*

NDATA = Notation DATA

Dichiarazione	Tipo	Esempio
GENERALE	EXTERNAL (*UNPARSED*) *(generalmente si riferisce a dati non-XML)*	<!ENTITY nome SYSTEM "URI" NDATA name> <!ENTITY nome PUBLIC "public_ID" "URI" NDATA nome1> **<!ENTITY logoA SYSTEM "http://www.mysite.net/logo.gif" NDATA gif> <!NOTATION gif PUBLIC "gif viewer"> <!ENTITY logoB PUBLIC "-//W3C//GIF logo//EN" "http://www.mysite.org/logo.gif" NDATA gif> <!NOTATION gif PUBLIC "gif viewer"> <img src="logoB">**

DTD – Vocabolario e Sintassi

- DTD <!ENTITY>
  - *Sintassi possibili*
    - *<!ENTITY nome 'entity value'>*
    - *<!ENTITY nome SYSTEM 'URI'>*
    - *<!ENTITY nome PUBLIC 'public_ID'>*

NDATA = Notation DATA

Dichiarazione	Tipo	Esempio
PARAMETRICHE	INTERNAL (*PARSED*) *(riferimenti ad entità utilizzati per dichiarare entità esistenti solo nella DTD)*	**<!ENTITY % p "(#PCDATA)">** **<!ELEMENT student (id,cognome)>** **<!ELEMENT id %p;>**

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Descrive
    - tag (**elementi**) utilizzabili in un file XML
    - relazioni tra tag utilizzabili in un file XML
    - definizione e descrizione degli **attributi** degli elementi del file XML
    - usa le convenzioni del linguaggio XML
  - Struttura generale di uno SCHEMA
    - tipicizzazione dei dati
    - divisione in 2 parti
      - *Structures (descrizione struttura analoga a DTD)*
      - *Datatype (tipicizzazione e definizione di strutture dati*

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
        <!--  Definizione della grammatica-->
</xs:schema>
```

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Elemento <schema>
    - *Elemento radice del file*
    - *Sintassi <schema> </schema>*
    - *Attributi principali* (*funzionamento correlato ai* **namespace**)

<schema>	
xmlns	xml namespace – indirizzo del namespace di riferimento
targetNamespace	indica da quale namespace provengono gli elementi definiti nello schema e può avere valore NULL

```
<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com">
...
</xs:schema>
```

namespace: «raccolta di nomi identificata da un riferimento URI; tali nomi vengono utilizzati nei documenti XML come tipi di elementi e nomi di attributi» (Namespace in XML – W3C)

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - XMLSchema: Structures
    - *Dichiarazioni di elementi*
    - *Particelle*
    - *Definizioni di tipi complessi*
    - *Definizioni di tipi complessi con* **simpleContent**
    - *Dichiarazioni di attributi*
    - *Gruppi di attributi*

XMLSchema: Structures	
Componenti primari	• *Definizioni: tipi semplici e tipi complessi* • *Dichiarazioni: di elementi e attributi*
Componenti secondari	• *Definizioni: gruppi di attributi, vincoli di identità e gruppi di modelli* • *Dichiarazioni: notazioni*
Helper	• *Annotazioni* • *Gruppi di modelli* • *Particelle* • *Wildcard* • *Uso degli attributi*

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - XMLSchema: Nomi
    - *Iniziano con carattere alfanumerico (A-Z; a-z) e possono contenere cifre (0-9), underscore (_), dash (-), punto (.) e due punti (:)*
    - *Due punti riservati ai **namespace***
    - *I nomi NON possono iniziare con «xml» (né UPPER né LOW CASE)*

XMLSchema: Nomi	
NCName (Non-Colon Name)	• *Non hanno prefissi qualificanti **<tagname />***
Nome qualificato	• *Qualificati da namespase **<prefisso:elementName />***

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Tipi semplici: <simpleType />
    - *Sono sempre «restrizioni»*
    - *Sono restrizioni del* ***tipo primario*** **SimpleType**

<simpleType *id=ID name=NCName any attributes*>
(annotation?,(restriction|list|union))
</simpleType>

Tipi semplici *(esempi)*

```
<simpleType name='key'>
  <restriction base='xs:string'>
    <minLenght value='5' />
    <maxLenght value='15' />
  </restriction>
</simpleType>
```

```
<xs:element name="age">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="120"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Tipi complessi: <complexType />
    - *Sono restrizioni del* ***tipo primario*** **anyType**
    - *Sono restrizioni di qualche tipo complessio*
    - *Sono estensioni di altro tipo complessi*
    - *Sono estensioni di un tipo semplice*

<complexType id=ID name=NCName abstract=true|false mixed=true|false
block=(#all|list of (extension|restriction))
final=(#all|list of (extension|restriction))
any attributes>
(annotation?,(simpleContent|complexContent|((group|all|
choice|sequence)?,((attribute|attributeGroup)*,anyAttribute?))))
</complexType>

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Tipi complessi: <complexType />

Attributo	Descrizione
id	Specifica un identificativo unico per il tipo (Opzionale)
name	Specifica il nome per il tipo
abstract	Specifica se il tipo può essere usato in un documento. **True** indica che il tipo non può essere utilizzato direttamente, ma deve utilizzare un tipo complesso derivato da questo tipo complesso. Default è **false** (Opzionale)
mixed	Specifica se possono esserci dati tra gli elementi figlio di questo tipo; default è false. Se un elemento **simpleContent** è un elemento figlio, l'attributo mixed NON è permesso (Opzionale)
block	Identifica un tipo complesso che ha un tipo specificato di derivazione da essere utilizzati al posto di questo tipo complesso. Questo valore può contenere # all o un elenco che è un sottoinsieme di estensione o restrizione (Opzionale): • **extension** – identifica i tipi complessi derivati per estensione • **restriction** – identifica i tipi complessi derivati per restrizione • **#all** – identifica i tipi complessi derivati
final	Identifica un tipo specificato di derivazione di tipo complesso. Può contenere # all o un elenco che è un sottoinsieme di estensione o restrizione. • **extension** – identifica derivazioni per estensione • **restriction** – identifica derivazioni per restrizione • **#all** – identifica tutte le derivazioni

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Tipi complessi: <complexType />

Esempio

```
<xs:element name="note">
  <xs:complexType>

     <xs:sequence>
        <xs:element name="to" type="xs:string"/>
         <xs:element name="from" type="xs:string"/>
    </xs:sequence>

  </xs:complexType>
</xs:element>
```

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Tipi complessi: <complexType />

Esempio

```
<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="fullpersoninfo">
  <xs:complexContent>
    <xs:extension base="personinfo">
      <xs:sequence>
        <xs:element name="address" type="xs:string"/>
        <xs:element name="city" type="xs:string"/>
        <xs:element name="country" type="xs:string"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>

</xs:complexType>
```

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Elementi semplici: <xs:element />
    - *Un elemento semplice è un elemento XML che può contenere solo testo. Non può contenere altri elementi o attributi.*
    - Sintassi **<xs:element name="element_namen" type="element_type**"/>

Alcuni tipi principali

- xs:string
- xs:decimal
- xs:integer
- xs:boolean
- xs:date
- xs:time

Esempio

```
<xs:element name="lastname" type="xs:string"/>
<xs:element name="age" type="xs:integer"/>
<xs:element name="dateborn" type="xs:date"/>
```

```
<lastname>Refsnes</lastname>
<age>36</age>
<dateborn>1970-03-27</dateborn>
```

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Elementi complessi: <xs:element />
    - *Un elemento semplice è un elemento XML che può contenere solo testo. Non può contenere altri elementi o attributi.*
    - Sintassi **<xs:element name="element_namen">**
      <xs:complexType>

      <xs:complex>
      </xs:element>

Tipi di elementi complessi

- empty elements
- elements che contengono solo altri elements
- elements che contengono solo testo
- elements che contengono sia testo che altri elementi

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Elementi semplici: <xs:attribute />
    - *Un elemento semplice che non può avere attributi.*
    - Sintassi **<xs:attribute name=«att_namen" type=«att_type**"/>

Alcuni tipi principali

- xs:string
- xs:decimal
- xs:integer
- xs:boolean
- xs:date
- xs:time

Esempio

<xs:attribute name="lang" type="xs:string" use="required"/>

Attributo use="required|optional"

SCHEMA – Vocabolario e Sintassi

- SCHEMA (struttura del documento XML)
  - Elementi complessi: <xs:element />

Esempio

```
<xs:element name="employee">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Esempio

```
<xsd:element name="_root">
   <xsd:complexType>
     <xsd:sequence>
      <xsd:element ref="mail" minOccurs="1“ maxOccurs="unbounded"/>
     </xsd:sequence>
</xsd:complexType>
```