EBook Gratuito

APPRENDIMENTO

xsd

Free unaffiliated eBook created from
Stack Overflow contributors.

#xsd

Sommario

Di

Capitolo 1: Iniziare con xsd

Osservazioni

XML Schema è un linguaggio e un framework per la convalida di documenti XML.

Un documento XML che è **ben formato** , nel senso che è sintatticamente conforme alla specifica XML, può essere testato per la **validità** rispetto a uno schema. La distinzione tra well-formedness, che è assoluta e validità, che è relativa a uno schema, è fondamentale.

La convalida comprende:

- Verifica se il documento XML soddisfa requisiti aggiuntivi quali gli elementi con determinati nomi, restrizioni sul contenuto degli elementi, vincoli di coerenza (chiavi primarie, unicità, ecc.), Valori di attributo o corrispondenza di determinati tipi di testo.
- In caso di successo, conversione dell'istanza del modello di dati di input (chiamata Infoset XML) in un'istanza di output (PSVI: Infoset di post-configurazione), in cui elementi e attributi sono annotati con informazioni sul tipo, dove sono stati compilati i valori predefiniti, ecc.

Lo schema XML è stato introdotto per soddisfare i requisiti che la convalida DTD non è riuscita a risolvere, tra cui un sistema di tipi più completo che include un ricco set di tipi predefiniti, limitazioni di tipo ed estensioni e un maggiore controllo sulla restrizione del layout degli elementi.

Versioni

Versione	Data di rilascio
1.0	2001/05/02
1.0, seconda edizione	2004-10-28
1.1	2012-04-05

XML Schema 1.0 è stato approvato come Raccomandazione W3C nel maggio 2001 e la seconda edizione che incorpora errata è stata pubblicata come Raccomandazione W3C alcuni anni dopo.

XML Schema 1.1 è diventato una raccomandazione W3C nel 2012, che ha corretto più bug e aggiunto altri miglioramenti, pur essendo per lo più compatibile con le versioni precedenti.

Examples

Installazione o configurazione

XSD, XML Schema Definition, è un linguaggio che descrive la struttura dei documenti XML. I file XSD possono essere utilizzati per convalidare un file XML. Il processo di fare ciò dipenderà da ciò

che scegli di implementarlo con. È necessario prestare attenzione per garantire che il motore di convalida che si utilizza sia compatibile con la versione desiderata di XSD.

Leggi Iniziare con xsd online: https://riptutorial.com/it/xsd/topic/2907/iniziare-con-xsd

Capitolo 2: xs: complexType

introduzione

A xs: complexType fornisce una descrizione del contenuto di un elemento XML nel documento di istanza. La definizione di xs: complexType può essere effettuata globalmente nel qual caso ha un nome e può essere riutilizzata all'interno dello schema, oppure può essere posizionata e utilizzata solo all'interno del contesto dichiarato.

Parametri

attributi	Descrizione
astratto	Se impostato su true, il tipo complesso non può essere utilizzato direttamente in un documento XML di istanza tramite xsi: type. Può tuttavia essere utilizzato come tipo di base per una definizione di elemento. (default false) - Valido solo per il livello root xs: complexType
bloccare	Limita i tipi che possono essere utilizzati in un documento di istanza XML (il valore predefinito è l'attributo xS: schemas blockDefault se impostato, altrimenti il valore predefinito è vuoto, valori '#all' | un elenco di ('estensione', 'lista', 'unione') separati da spazi bianchi).
finale	Limiti che derivano tipi dall'utilizzo di questo tipo in determinati modi all'interno dello schema (il valore predefinito è l'attributo xs: schemas finalDefault se impostato, altrimenti il valore predefinito è vuoto, valori '#all' | o un elenco di ('estensione', 'elenco ',' unione ') separati da spazi bianchi) - Valido solo per il livello root xs: complexType
id	L'id dell'elemento dello schema (facoltativo)
misto	Indica che l'elemento XML dell'istanza potrebbe contenere contenuto misto (predefinito su falso)
nome	Il nome di xs: complexType - Valido solo per il livello root xs: complexType
qualunque	Sono consentiti tutti gli altri attributi non presenti nello spazio dei nomi " http://www.w3.org/2001/XMLSchema ".
----------	------
Elementi	**Descrizione**
----------	------
xs: annotation	Offre la possibilità di aggiungere documentazione e dati leggibili dalla

attributi	Descrizione
	macchina.
xs: simpleContent	Utilizzato quando xs: complexType deriva da xs: simpleType.
xs: complexContent	Utilizzato quando xs: complexType deriva da un altro xs: complexType.
xs: Gruppo	Aggiunge gli elementi da un xs: group alla definizione xs: complexType
xs: tutto	Aggiunge gli elementi da una xs: tutti alla definizione xs: complexType
xs: scelta	Aggiunge gli elementi da una xs: scelta alla definizione xs: complexType
xs: sequence	Aggiunge gli elementi da una sequenza xs: alla definizione xs: complexType
xs: attributo	Aggiunge l'attributo xs: alla definizione xs: complexType
xs: attributeGroup	Aggiunge xs: attributeGroup alla definizione xs: complexType
xs: anyAttribute	Aggiunge xs: anyAttribute alla definizione xs: complexType

Osservazioni

Derivante da una xs: complexType

Quando un xs: complexType deriva da un altro xs: complexType può farlo tramite *estensione* o *restrizione* .

- estensione: il tipo derivante prende tutto ciò che è definito nel tipo base e lo aggiunge.
- restrizione: il tipo derivante prende solo le parti selezionate dal tipo di base, consentendo solo le parti che desidera, non è possibile aggiungere ulteriori elementi.

Derivante da una xs: simpleType

Quando xs: complexType deriva da una xs: simpleType può farlo tramite *estensione* , nel qual caso può aggiungere attributi al tipo risultante, ma non elementi.

Tipo di contenuto

Concettualmente una xs: complexType contiene contenuti *semplici* o *complessi* . Se xs: complexType deriva da un typ basato su xs: anySimpleType (xs: int, xs: string ecc) allora è *semplice* . Se deriva da un xs: complexType che contiene contenuto *complesso* , allora esso stesso è *complesso* (se xs: complexType non deriva da un tipo, allora è anche complesso).

Examples

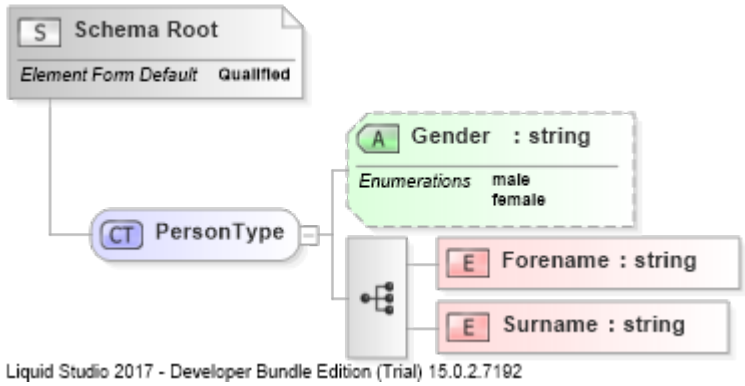
Global ComplexType con Sequence e attributi

Questo esempio mostra una semplice definizione globale di un Tipo complesso. La definizione è considerata globale in quanto è figlio dello xs: schema. I tipi definiti a livello globale possono essere utilizzati altrove nello schema.

Questa è la forma più comune per dichiarare un xs globale: complexType, definisce gli elementi figli usando xs: sequence, xs: choice o xs: all, e opzionalmente ha anche attributi.

Nota: poiché è definito globalmente, deve avere un nome univoco all'interno del set di schemi.

```
<?xml version="1.0" encoding="utf-8" ?>
<!--Created with Liquid Studio 2017 (https://www.liquid-technologies.com)-->
<xs:schema elementFormDefault="qualified"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:complexType name="PersonType">
        <xs:sequence>
            <xs:element name="Forename" type="xs:string" />
            <xs:element name="Surname" type="xs:string" />
        </xs:sequence>
        <xs:attribute name="Gender">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:enumeration value="male" />
                    <xs:enumeration value="female" />
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
    </xs:complexType>
</xs:schema>
```



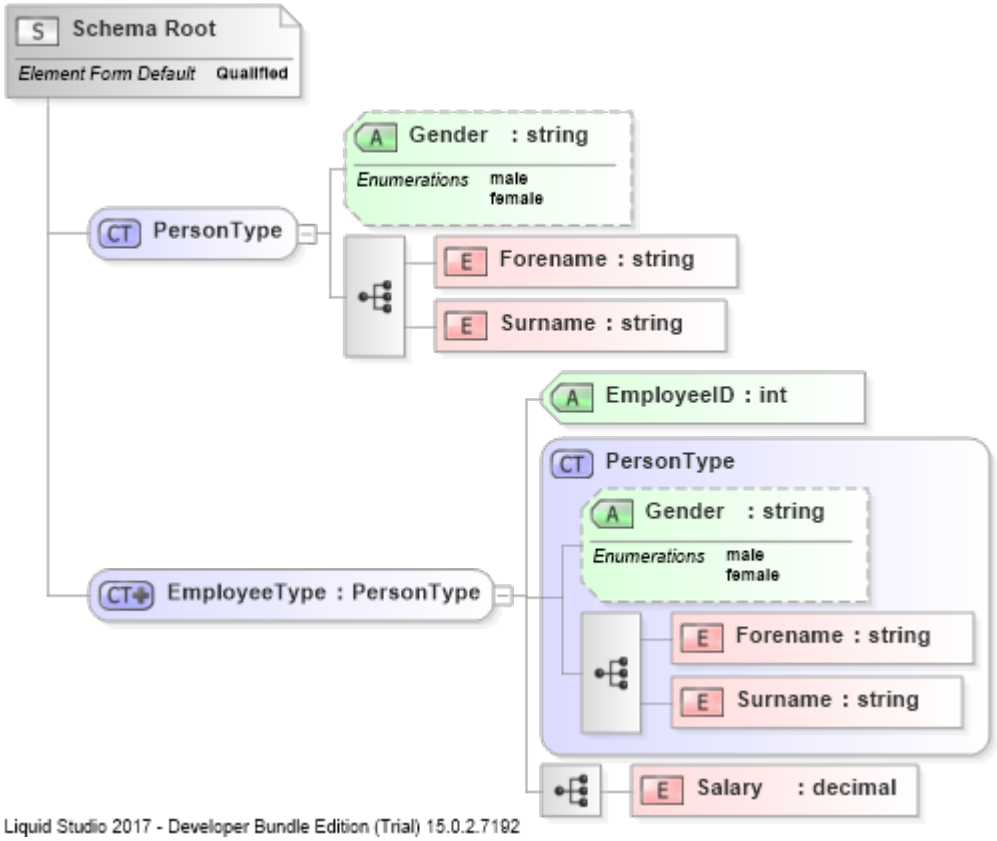
Creazione di un xs globale: complexType estendendo un xs esistente: complexType

In questo esempio stiamo creando un nuovo xs: complexType (EmployeeType) basato su un xs esistente: complexType (PersonType).

La costruzione di questo è leggermente più complicata. Poiché la base xs: complexType (PersonType) è considerata *complessa* (di seguito più avanti) aggiungiamo l'elemento <xs:

complexContent>. Quindi, poiché stiamo estendendo il PersonType, aggiungiamo l'elemento <xs: extension base = "PersonType">. All'interno del tag xs: extension possiamo aggiungere un compositore (xs: all / xs: choice / xs: sequence) e qualsiasi altro attributo.

```
<?xml version="1.0" encoding="utf-8" ?>
<!--Created with Liquid Studio 2017 (https://www.liquid-technologies.com)-->
<xs:schema elementFormDefault="qualified"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:complexType name="PersonType">
        <xs:sequence>
            <xs:element name="Forename" type="xs:string" />
            <xs:element name="Surname" type="xs:string" />
        </xs:sequence>
        <xs:attribute name="Gender">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:enumeration value="male" />
                    <xs:enumeration value="female" />
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
    </xs:complexType>
    <xs:complexType name="EmployeeType">
        <xs:complexContent>
            <xs:extension base="PersonType">
                <xs:sequence>
                    <xs:element name="Salary" type="xs:decimal" />
                </xs:sequence>
                <xs:attribute name="EmployeeID" type="xs:int" use="required" />
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
</xs:schema>
```



Creazione di un xs globale: complexType limitando un xs esistente: complexType

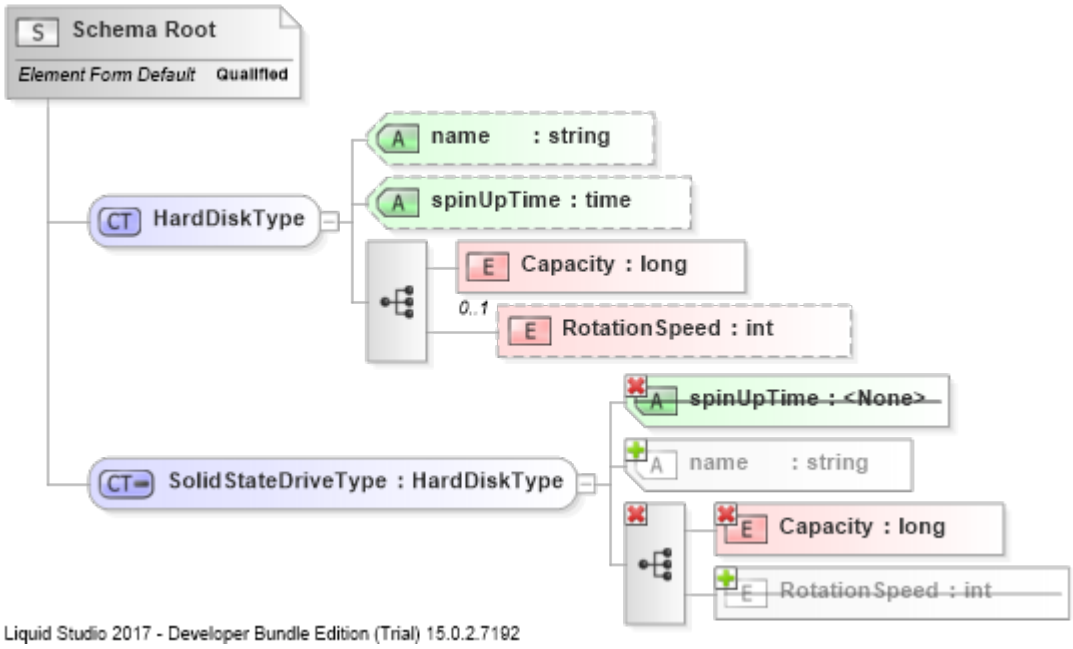
Questo è dove le cose diventano un po 'complicate. Ora stiamo limitando una xs esistente: complexType. Il nostro SolidStateDriveType deriva da HardDiskType ma rimuove l'attributo spinUpTime e l'elemento RotationSpeed.

Si noti che l'approccio per trattare attributi ed elementi è diverso. Per rimuovere un attributo devi dichiararlo nuovamente e impostare il suo *utilizzo* su *proibito* . Per gli elementi semplicemente non la loro re-dichiarazione li farà escludere, infatti è necessario ri-dichiarare qualsiasi elemento che si desidera mantenere nel nuovo tipo.

Concetto chiave per tipi ristretti *: deve essere possibile caricare un elemento dell'istanza XML derivante dal tipo limitato nel tipo base, in altri termini il tipo limitato deve essere in grado di "adattarsi" al tipo base. Quindi non è possibile escludere un attributo o elemento obbligatorio, per poterlo escludere nel tipo limitato deve essere facoltativo nel tipo base. Se si modificano le regole type / facet di un elemento o attributo nel tipo limitato, le nuove regole type / facet devono essere compatibili con il tipo base, quindi se il tipo base era short, il tipo restricted poteva essere un byte, ma non molto tempo*

```
<?xml version="1.0" encoding="utf-8" ?>
<!--Created with Liquid Studio 2017 - Developer Bundle Edition (Trial) 15.0.2.7192
(https://www.liquid-technologies.com)-->
<xs:schema elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:complexType name="HardDiskType">
        <xs:sequence>
            <xs:element name="Capacity" type="xs:long" />
            <xs:element name="RotationSpeed" type="xs:int" minOccurs="0" />
```

```
        </xs:sequence>
        <xs:attribute name="name" type="xs:string" />
        <xs:attribute name="spinUpTime" type="xs:time" />
    </xs:complexType>
    <xs:complexType name="SolidStateDrive">
        <xs:complexContent>
            <xs:restriction base="HardDiskType">
                <xs:sequence>
                    <xs:element name="Capacity" type="xs:long" />
                </xs:sequence>
                <xs:attribute name="spinUpTime" use="prohibited" />
            </xs:restriction>
        </xs:complexContent>
    </xs:complexType>
</xs:schema>
```



Leggi xs: complexType online: https://riptutorial.com/it/xsd/topic/9047/xs--complextype

Capitolo 3: xs: schema

introduzione

Descrive elementi, attributi e tipi validi in un documento di istanza XML. Uno schema XML (XSD) deve contenere un singolo livello radice xs: elemento dello schema.

Parametri

attributi	Descrizione
attributeFormDefault	Indica se gli attributi nel documento di istanza XML devono essere qualificati con uno spazio dei nomi (predefinito non qualificato)
blockDefault	Il valore predefinito dell'attributo di *blocco* applicato a xs: complexType e xs: element. Definisce le regole per il blocco della derivazione / sostituzione nel documento di istanza (predefinito vuoto, ovvero non blocca nulla)
defaultAttributes	**(XSD 1.1)** Specifica un xs: attributeGroup che sarà associato a tutti x: complexType e xs: elemento all'interno dello schema (opzionale).
elementFormDefault	Indica se l'elemento nel documento di istanza XML deve essere qualificato con uno spazio dei nomi (predefinito non qualificato). **Nota** : quasi senza eccezioni tutti gli schemi lo impostano su ' *qualificato* '.
finalDefault	Il valore di attributo *finale* predefinito utilizzato in xs: complexType e xs: element. Definisce le regole per il blocco della derivazione / sostituzione nello schema (default vuoto, ie block nothing)
id	L'id dell'elemento dello schema (facoltativo)
targetNamespace	Qualifica tutti gli elementi e gli attributi (e i componenti definiti a livello globale) definiti all'interno di questo schema.
versione	La versione dello schema, questa è la versione del documento, non questa versione XSD (cioè Soap 1.2, FpML 4.2 ecc.)
xpathDefaultNamespace	**(XSD 1.1)** Il valore predefinito per l'attributo *xpathDefaultNamespace* utilizzato in xs: selector, xs: field, xs: alternative & xs: assert (specifica lo spazio dei nomi predefinito da utilizzare nelle espressioni XPath)
qualunque	Sono consentiti tutti gli altri attributi non presenti nello spazio dei

attributi	Descrizione
	nomi " http://www.w3.org/2001/XMLSchema ".
Elementi	**Descrizione**
xs: annotation	Offre la possibilità di aggiungere documentazione e dati leggibili dalla macchina.
xs: includono	Utilizzato per includere uno schema con lo stesso spazio dei nomi target o nessuno spazio dei nomi target (vedere gli schemi dei camaleonti).
xs: import	Utilizzato per includere uno schema con uno spazio targetName diverso da quello principale.
xs: ridefiniscono	Utilizzato per includere uno schema con lo stesso targetNamespace (o nessun targetNamespace) e modificare le definizioni xs: simpleType, xs: complexType, xs: group o xs: attributeGroup contenute al suo interno (qui si trovano i draghi ....)
xs: simpleType	Definisce un tipo semplice (denominato) globale che può quindi essere referenziato e riutilizzato.
xs: complexType	Definisce un tipo globale (denominato) complesso che può quindi essere referenziato e riutilizzato.
xs: Gruppo	Definisce un gruppo globale (denominato) di elementi che possono quindi essere referenziati e riutilizzati.
xs: attributeGroup	Definisce un gruppo globale (denominato) di attributi che può quindi essere referenziato e riutilizzato.
xs: attributo	Definisce un attributo globale (denominato) che può quindi essere referenziato e riutilizzato.
xs: element	Definisce un elemento globale (denominato) che può quindi essere referenziato e riutilizzato o utilizzato come base per un documento di istanza XML.
xs: la notazione	-
xs: defaultOpenContent	**(XSD 1.1)** Specifica le regole per consentire l'autorizzazione di ulteriori elementi all'interno di ogni xs: complexType e xs: elemento all'interno dello schema.

Examples

Xs di base: schema

Mostra uno schema molto semplice.

Nota: per convenzione elementFormDefault è impostato su ' *qualificato* ', nel mondo reale ti verrà difficile trovare uno schema che non lo imposti (quindi includilo nei tuoi schemi!).

```
<?xml version="1.0" encoding="utf-8" ?>
<!--Created with Liquid Studio 2017 - (https://www.liquid-technologies.com)-->
<xs:schema elementFormDefault="qualified"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="Person">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Forename" type="xs:string" />
                <xs:element name="Surname" type="xs:string" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

xs: schema elementFormDefault attributo

Per convenzione elementFormDefault è sempre impostato su *qualificato* , ma permette di vedere cosa effettivamente fa.

Innanzitutto con elementFormDefault impostato su qualificato.

```
<?xml version="1.0" encoding="utf-8" ?>
<!--Created with Liquid Studio 2017 (https://www.liquid-technologies.com)-->
<xs:schema elementFormDefault="qualified"
           targetNamespace="http://base.com"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="MyBaseElement">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="ChildA" type="xs:string" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Esempio di documento XML

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Created with Liquid Studio 2017 (https://www.liquid-technologies.com) -->
<b:MyBaseElement xmlns:b="http://base.com"
                 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:schemaLocation="http://base.com ElementFormDefault_qualified.xsd">
    <b:ChildA>string</b:ChildA>
</b:MyBaseElement>
```

Si noti che l'elemento ChildA deve anche essere qualificato con lo spazio dei nomi 'b'.

Ora guardiamo con elementFormDefault impostato su non qualificato.

```
<?xml version="1.0" encoding="utf-8" ?>
<!--Created with Liquid Studio 2017 (https://www.liquid-technologies.com)-->
<xs:schema elementFormDefault="unqualified"
           targetNamespace="http://base.com"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="MyBaseElement">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="ChildA" type="xs:string" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Esempio di documento XML

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Created with Liquid Studio 2017 (https://www.liquid-technologies.com) -->
<b:MyBaseElement xmlns:b="http://base.com"
                 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:schemaLocation="http://base.com ElementFormDefault_unqualified.xsd">
    <ChildA>string</ChildA>
</b:MyBaseElement>
```

Nota che questa volta solo l'elemento globalmente definito MyBaseElement è qualificato con lo spazio dei nomi 'b', l'elemento interno ChildA (che è definito all'interno dello schema) non è qualificato.

Nell'ultimo esempio abbiamo visto che gli elementi definiti a livello globale devono essere qualificati nel documento di istanza XML, ma gli elementi definiti inplace no. Ma questo non significa solo l'elemento radice, se si dispone di elementi definiti a livello globale a cui si fa riferimento, quindi devono anche essere qualificati.

```
<?xml version="1.0" encoding="utf-8" ?>
<!--Created with Liquid Studio 2017 (https://www.liquid-technologies.com)-->
<xs:schema elementFormDefault="unqualified"
           targetNamespace="http://base.com"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="MyBaseElement">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="ChildA" type="xs:string" />
                <xs:element xmlns:q1="http://base.com" ref="q1:MyElement" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="MyElement" type="xs:string" />
</xs:schema>
```

Esempio di documento XML

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<!-- Created with Liquid Studio 2017 (https://www.liquid-technologies.com) -->
<b:MyBaseElement xmlns:b="http://base.com"
                 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:schemaLocation="http://base.com ElementFormDefault_unqualified.xsd">
    <ChildA>string</ChildA>
    <b:MyElement>string</b:MyElement>
</b:MyBaseElement>
```

Si noti che anche MyElement necessita di una qualifica come è definita globalmente.

In conclusione, se avete impostato elementFormDefault su qualificato, allora tutto deve essere qualificato con uno spazio dei nomi (tramite un alias di namespace o impostando il default namesapce xmlns = "..."). Comunque elementFormDefault è impostato su non qualificato, le cose si complicano e devi fare un esame approfondito degli schemi per capire se le cose dovrebbero essere qualificate o no.

Suppongo che sia per questo che elementFormDefault è sempre impostato su qualificato!

Leggi xs: schema online: https://riptutorial.com/it/xsd/topic/9052/xs--schema

Titoli di coda

S. No	Capitoli	Contributors
1	Iniziare con xsd	Beth Whitezel, Community, Ghislain Fourny, whrrgarbl, Wolfgang Schindler
2	xs: complexType	Sprotty
3	xs: schema	Sprotty