Бесплатная электронная книга

учусь

xsd

Free unaffiliated eBook created from
Stack Overflow contributors.

#xsd

1
1: xsd 2
2
2
Examples 3
3
2: : ComplexType 4
4
4
5
Examples 6
6
xs: complexType xs: complexType 7
xs: complexType xs: complexType 8
3: : 10
10
10
Examples 12
xs: 12
xs: schema elementFormDefault 12
15

Около

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: xsd

It is an unofficial and free xsd ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official xsd.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

глава 1: Начало работы с xsd

замечания

XML Schema - это язык и структура для проверки XML-документов.

XML-документ, который **хорошо сформирован** , в том смысле, что он синтаксически соответствует спецификации XML, может быть проверен на предмет **достоверности в** отношении схемы. Различие между хорошо сформированной, абсолютной и достоверной, которая относится к схеме, имеет первостепенное значение.

Валидация охватывает:

- Проверка того, выполняет ли XML-документ дополнительные требования, такие как элементы, имеющие определенные имена, ограничения на содержимое элементов, ограничения последовательности (первичные ключи, уникальность и т. Д.), Значения атрибутов или текст, соответствующий определенным типам.
- После успеха преобразование экземпляра модели входных данных (называемого XML Infoset) в выходной экземпляр (PSVI: Post-Schema-Validation Infoset), где элементы и атрибуты аннотируются информацией о типе, где значения по умолчанию заполнены и т. Д.

XML-схема была введена для удовлетворения требований, которые не удалось устранить при проверке DTD, среди прочего, более полной системы типов, включая богатый набор встроенных типов, ограничений типа и расширения, а также большего контроля над ограничением компоновки элементов.

Версии

Версия	Дата выхода
1,0	2001-05-02
1.0, второе издание	2004-10-28
1,1	2012-04-05

XML-схема 1.0 была одобрена в качестве рекомендации W3C в мае 2001 года, а второе издание, включающее ошибки, было опубликовано в качестве рекомендации W3C несколькими годами позже.

XML-схема 1.1 стала Рекомендацией W3C в 2012 году, в которой исправлено больше ошибок и добавлены другие улучшения, хотя они в основном совместимы с более ранними

версиями.

Examples

Установка или настройка

XSD, XML Schema Definition - это язык, который описывает структуру XML-документов. Файлы XSD могут использоваться для проверки XML-файла. Процесс выполнения этого будет зависеть от того, что вы хотите реализовать. Необходимо следить за тем, чтобы механизм проверки, который вы используете, совместим с желаемой версией XSD.

Прочитайте Начало работы с xsd онлайн: https://riptutorial.com/ru/xsd/topic/2907/начало-работы-с-xsd

глава 2: xs: ComplexType

Вступление

Xs: complexType предоставляет описание содержимого элемента XML в документе экземпляра. Определение xs: complexType может быть сделано глобально, и в этом случае оно имеет имя и может быть повторно использовано внутри схемы, или оно может быть на месте и использоваться только в контексте, который объявлен.

параметры

Атрибуты	Описание
Аннотация	Если установлено значение true, сложный тип нельзя использовать непосредственно в XML-документе экземпляра с помощью xsi: type. Однако он может использоваться как базовый тип для определения элемента. (по умолчанию false) - допустимо только для корневого уровня xs: complexType
блок	Ограничивает типы, которые могут быть использованы в документе экземпляра XML (по умолчанию используется значение атрибута blocksDefault атрибутов xs: schemas, в противном случае по умолчанию пустым, значения «#all» | список («расширение», «список», 'union'), разделенные пробелами).
окончательный	Пределы, получающие типы от использования этого типа определенными способами в рамках схемы (по умолчанию используется значение атрибута finalsefs xs: schemas finalDefault, если по умолчанию установлено значение empty, значения '#all' | или список ('extension', 'list ',' union '), разделенные пробелами) - Только для корневого уровня xs: complexType
Я бы	Идентификатор элемента схемы (необязательно)
смешанный	Указывает, что элемент XML экземпляра может содержать смешанный контент (по умолчанию - false)
название	Имя xs: complexType - допустимо только для корневого уровня xs: complexType
любой	Разрешены любые другие атрибуты не в пространстве имен «

Атрибуты	Описание
	http://www.w3.org/2001/XMLSchema ».
----------	------
элементы	**Описание**
----------	------
xs: аннотация	Обеспечивает возможность добавления документации и машиносчитываемых данных.
xs: simpleContent	Используется, когда xs: complexType происходит от xs: simpleType.
xs: complexContent	Используется, когда xs: complexType происходит от другого xs: complexType.
xs: группа	Добавляет элементы из xs: group в определение xs: complexType
xs: все	Добавляет элементы из xs: all в определение xs: complexType
xs: выбор	Добавляет элементы из xs: choice в определение xs: complexType
xs: последовательность	Добавляет элементы из последовательности xs: в определение xs: complexType
xs: атрибут	Добавляет атрибут xs: в определение xs: complexType
xs: attributeGroup	Добавляет xs: attributeGroup в определение xs: complexType
xs: anyAttribute	Добавляет xs: anyAttribute в определение xs: complexType

замечания

Вывод из xs: complexType

Когда xs: complexType происходит от другого xs: complexType может делать это через *расширение* или *ограничение* .

- extension - тип вывода принимает все, что определено в базовом типе и добавляет к нему.
- ограничение - тип получения принимает только выбранные части из базового типа, только разрешая части, которые он хочет, никакие дополнительные элементы не

могут быть добавлены.

Получение из xs: simpleType

Когда xs: complexType происходит от xs: simpleType может делать это через *расширение* , и в этом случае он может добавлять атрибуты к результирующему типу, но не к элементам.

Тип содержимого

Концептуально xs: complexType содержит либо *простой,* либо *сложный* контент. Если xs: complexType происходит от типизированного на основе xs: anySimpleType (xs: int, xs: string и т. Д.), То это *просто* . Если это происходит из xs: complexType, который содержит *сложное* содержимое, то он сам является *сложным* (если xs: complexType не является производным от типа, то он также является сложным).

Examples

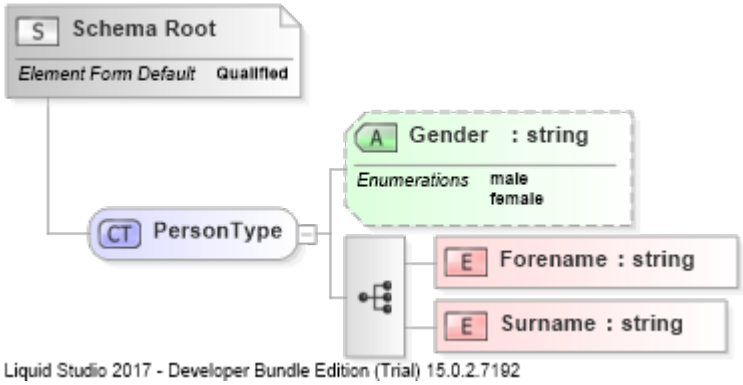
Глобальный комплексный тип с последовательностью и атрибутами

В этом примере показано простое глобальное определение complexType. Определение считается глобальным, поскольку оно является дочерним элементом схемы xs:. Определенные глобально типы могут использоваться в другом месте схемы.

Это наиболее распространенная форма для объявления глобального xs: complexType, он определяет дочерние элементы, используя последовательность xs: xs: choice или xs: all и, возможно, также имеет атрибуты.

Примечание: поскольку он определяется глобально, он должен иметь уникальное имя в наборе схем.

```
<?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>
```

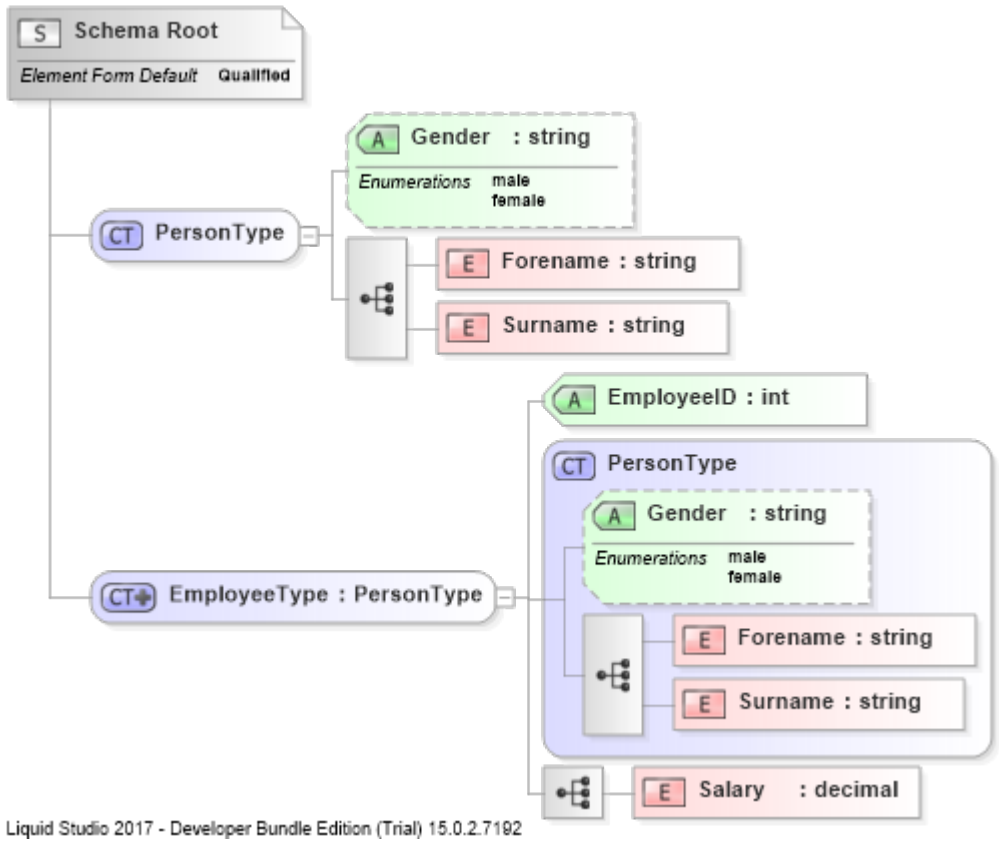


Создание глобального xs: complexType путем расширения существующего xs: complexType

В этом примере мы создаем новый xs: complexType (EmployeeType) на основе существующего xs: complexType (PersonType).

Построение этого немного сложнее. Поскольку базовый xs: complexType (PersonType) считается *сложным* (подробнее об этом ниже), мы добавляем элемент <xs: complexContent>. Затем, поскольку мы расширяем PersonType, мы добавляем элемент <xs: extension base = "PersonType">. Внутри тега xs: extension мы можем добавить композитор (xs: all / xs: choice / xs: sequence) и любые дополнительные атрибуты.

```
<?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>
```



Создание глобального xs: complexType путем ограничения существующего xs: complexType

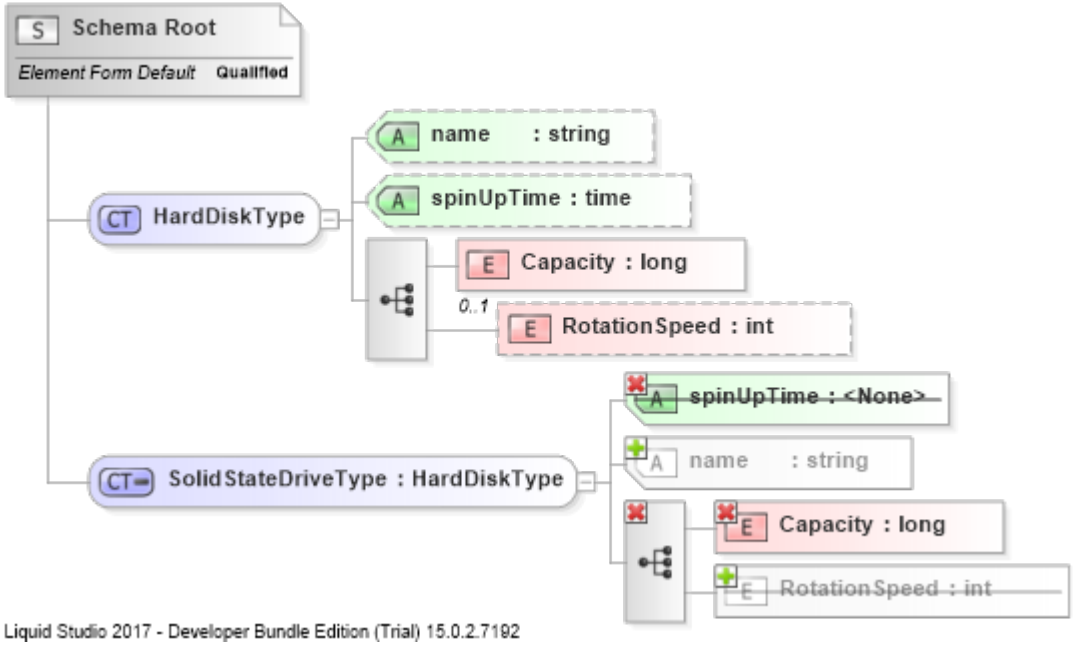
Здесь все становится немного сложнее. Теперь мы ограничиваем существующий xs: complexType. Наш SolidStateDriveType происходит из HardDiskType, но удаляет атрибут spinUpTime и элемент RotationSpeed.

Обратите внимание, что подход к работе с атрибутами и элементами отличается. Чтобы удалить атрибут, вам необходимо повторно объявить его и *запретить* его *использование* . Для элементов просто не повторное объявление их приведет к их исключению, на самом деле вам нужно повторно объявить любые элементы, которые вы хотите сохранить в новом типе.

Ключевая концепция для ограниченных типов *: должна быть возможность загрузить элемент экземпляра XML, полученный из ограниченного типа, в базовый тип, иначе говоря, ограниченный тип должен иметь возможность «вписываться» в базовый тип. Таким образом, вы не можете исключить обязательный атрибут или элемент, чтобы исключить его в ограниченном типе, он должен быть необязательным в базовом типе. Если вы изменяете правила типа / аспект элемента или атрибута в ограниченном типе, новые правила типа / грани должны быть совместимы с базовым типом, поэтому, если базовый тип был коротким, ограниченный тип может быть байтом, но недолго.*

```
<?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>
```



Прочитайте xs: ComplexType онлайн: https://riptutorial.com/ru/xsd/topic/9047/xs--complextype

глава 3: xs: схема

Вступление

Описывает элементы, атрибуты и типы, которые действительны в документе экземпляра XML. XML-схема (XSD) должна содержать один элемент xs: schema с одним корневым уровнем.

параметры

Атрибуты	Описание
attributeFormDefault	Указывает, должны ли атрибуты в экземпляре экземпляра XML быть квалифицированы с пространством имен (по умолчанию неквалифицировано)
blockDefault	Значение по умолчанию для атрибута *block* , применяемого к xs: complexType и xs: element. Определяет правила блокировки вывода / подстановки в документе экземпляра (по умолчанию пуст, т.е. ничего не блокирует)
defaultAttributes	**(XSD 1.1)** Определяет группу xs: attributeGroup, которая будет связана со всеми элементами xs: complexType и xs: внутри схемы (необязательно).
elementFormDefault	Указывает, должен ли элемент в документе экземпляра XML быть квалифицированным с пространством имен (по умолчанию неквалифицирован). **Примечание** . Почти все исключения все схемы заданы для « *квалифицированных* ».
finalDefault	Значение *конечного* атрибута по умолчанию, используемое в xs: complexType и xs: element. Определяет правила блокировки деривации / замены в схеме (по умолчанию пустой, т.е. ничего не блокирует)
Я бы	Идентификатор элемента схемы (необязательно)
целевое пространство	Квалифицирует все элементы и атрибуты (и глобально определенные компоненты), определенные в этой схеме.
версия	Версия схемы, это версия документа, а не версия XSD (т.е. Soap 1.2, FpML 4.2 и т. Д.),

Атрибуты	Описание
xpathDefaultNamespace	**(XSD 1.1)** Значение по умолчанию для атрибута *xpathDefaultNamespace,* которое используется в xs: selector, xs: field, xs: alternative & xs: assert (указывает пространство имен по умолчанию, которое будет использоваться в выражениях XPath)
любой	Разрешены любые другие атрибуты не в пространстве имен « http://www.w3.org/2001/XMLSchema ».
элементы	**Описание**
xs: аннотация	Обеспечивает возможность добавления документации и машиносчитываемых данных.
xs: включить	Используется для включения схемы с тем же целевым пространством имен или без targetNamespace (см. Схемы хамелеона).
xs: импорт	Используется для включения схемы с целевым пространством имен, отличным от родителя.
xs: переопределять	Используется для включения схемы с тем же самым целевым пространством имен (или без targetNamespace) и изменения xs: simpleType, xs: complexType, xs: group или xs: attributeGroup определения, содержащиеся в нем (вот драконы ....)
xs: simpleType	Определяет глобальный (именованный) простой тип, который затем можно ссылаться и повторно использовать.
xs: ComplexType	Определяет глобальный (именуемый) сложный тип, который затем можно ссылаться и повторно использовать.
xs: группа	Определяет глобальную (именованную) группу элементов, которая затем может быть указана и повторно использована.
xs: attributeGroup	Определяет глобальную (именованную) группу атрибутов, которая затем может быть указана и повторно использована.
xs: атрибут	Определяет глобальный (именованный) атрибут, который затем можно ссылаться и повторно использовать.
xs: элемент	Определяет глобальный (именованный) элемент, который затем можно ссылаться и повторно использовать, или использоваться как основа документа экземпляра XML.

Атрибуты	Описание
xs: обозначение	-
xs: defaultOpenContent	**(XSD 1.1)** Определяет правила разрешения дополнительных элементов в каждом элементе xs: complexType и xs: внутри схемы.

Examples

Основная xs: схема

Показывает очень основную схему.

Примечание: по соглашению elementFormDefault устанавливается в « *qualified* », в реальном мире вам будет трудно найти схему, которая не устанавливает это (так что просто включите ее в свои схемы!).

```
<?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>
```

Элемент xs: schema elementFormDefault

По соглашению elementFormDefault всегда настроен на *квалифицированный* , но позволяет посмотреть, что он на самом деле делает.

Сначала элемент elementFormDefault установлен на квалифицированный.

```
<?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>
```

Пример XML-документа

```
<?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>
```

Обратите внимание, что элемент ChildA также должен быть квалифицирован в пространстве имен «b».

Теперь давайте посмотрим на него с установкой elementFormDefault на неквалифицированный.

```
<?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>
```

Пример XML-документа

```
<?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>
```

Обратите внимание на то, что только глобально определенный элемент MyBaseElement имеет квалификацию с пространством имен «b», внутренний элемент ChildA (который определен на месте внутри схемы) не является квалифицированным.

В последнем примере мы увидели, что глобально определенные элементы должны быть квалифицированы в документе экземпляра XML, но элементы, определенные на месте, этого не делают. Но это не просто означает корневой элемент, если у вас есть глобально

определенные элементы, на которые ссылаются, то они также нуждаются в квалификации.

```
<?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>
```

Пример XML-документа

```
<?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>
```

Обратите внимание, что MyElement также нуждается в квалификации, поскольку он определен глобально.

В заключение, если у вас установлен элемент elementFormDefault, тогда все должно быть квалифицировано с пространством имен (либо через псевдоним пространства имен, либо путем установки по умолчанию namesapce xmlns = "..."). Однако elementFormDefault настроен на неквалифицированный, все усложняется, и вам нужно сделать некоторое окончательное рассмотрение схем для разработки, если что-то должно быть квалифицировано или нет.

Я предполагаю, что именно поэтому elementFormDefault всегда имеет квалификацию!

Прочитайте xs: схема онлайн: https://riptutorial.com/ru/xsd/topic/9052/xs--схема

кредиты

S. No	Главы	Contributors
1	Начало работы с xsd	Beth Whitezel, Community, Ghislain Fourny, whrrgarbl, Wolfgang Schindler
2	xs: ComplexType	Sprotty
3	xs: схема	Sprotty