Технология XSD

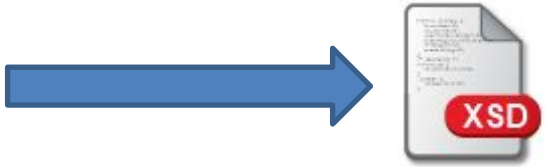
Обзор

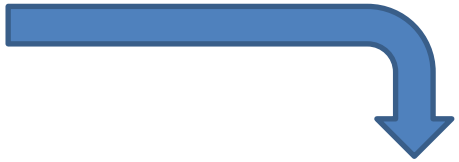
Что такое XML схема (XSD)?

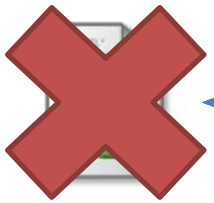
- XML Schema definition language (XSD) в настоящее время стандартный язык схем для XML документов и данных.
- 2 мая 2001, World Wide Web Consortium (W3C) опубликовал версию 1.0 стандарта XSD.
- XML Schema описывает элементы XML документа
- Описывает атрибуты в XML
- Описывает дочерние элементы их порядок и количество
- Описывает типы данных для элементов и атрибутов

Зачем нужны схемы?

Server-side programmer







Client-side programme

Создание XML Схемы

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.epam.com"
  xmlns="http://www.epam.com"
  elementFormDefault="qualified">
  ...
</xs:schema>
```

- Элемент **<schema>** - корневой элемент любой XML схемы
- **xmlns:xs="http://www.w3.org/2001/XMLSchema"** - Объявление пространства имен XML схемы с префиксом xs
- **targetNamespace="http://www.epam.com"** – для этого пространства имен применяется схема
- **xmlns="http://www.epam.com"** – пространство имен по умолчанию (без префикса)
- **elementFormDefault="qualified"** - все элементы должны быть namespace qualified

Простые элементы

- Элементы объявляются с использованием элемента `<element>`.
- Простой элемент может содержать только текст.
- Простой элемент не может содержать атрибутов.
- Объявляется простым типом (базовым типом или новым типом с расширением или ограничением базового типа с помощью элемента simpleType)

```
<xs:element name="age" type="xs:string"/>

<xs:element name="age">
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="100"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Объявление атрибутов

- Атрибуты объявляются с использованием элемента `<attribute>`
- Атрибут всегда объявляется простым типом

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

<lastname lang="EN">Smith</lastname>
```

- Для атрибута можно определить значение по умолчанию или фиксированное значение

```
<xs:attribute name="lang" type="xs:string" default="EN"/>

<xs:attribute name="lang" type="xs:string" fixed="EN"/>
```

Простой тип (Simple Type)

- Элемент `simpleType` определяет простой тип элементов или атрибутов, накладывая ограничения или расширения на базовые типы
- Может быть именованным или анонимным внутри элемента

```
<xs:element name="age" type="ageType"/>

<xs:simpleType name="ageType">
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="0"/>
    <xs:maxInclusive value="100"/>
  </xs:restriction>
</xs:simpleType>
```

- Описан элемент "age", который может иметь числовые значения от 0 до

Составной тип (Complex Type)

- Составной тип описывает элементы, содержащего другие элементы и/или атрибуты
- Составной тип описывается с помощью элемента complexType
- Может быть именованным или анонимным внутри элемента

```
<xs:element name="note">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="to" type="xs:string"/>
      <xs:element name="from" type="xs:string"/>
      <xs:element name="heading" type="xs:string"/>
      <xs:element name="body" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Элемент simpleContent

- Для расширения/ограничения составного типа, который может содержать только текст как содержимое, используется элемент `simpleContent`
- Используется для добавления атрибутов

```
<xsd:element name="shoesize">
  <xsd:complexType>
    <xsd:simpleContent>
      <xsd:extension base="xsd:integer">
        <xsd:attribute name="country" type="xsd:string"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
</xsd:element>
```

Элемент complexContent

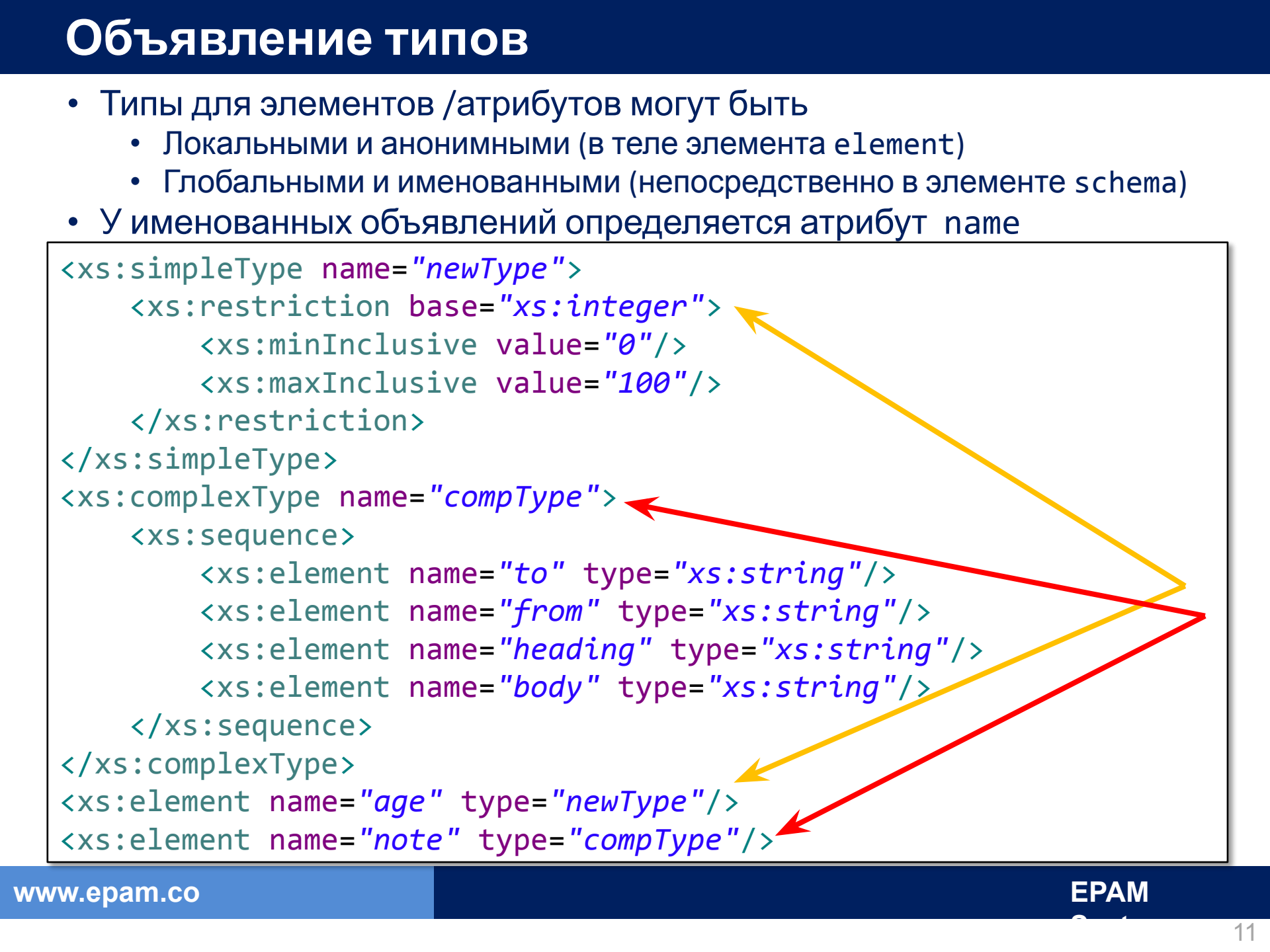
- Элемент `complexContent` служит для расширения или ограничения составных типов, объявленных ранее

```
<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>
```

Объявление типов

- Типы для элементов /атрибутов могут быть
  - Локальными и анонимными (в теле элемента `element`)
  - Глобальными и именованными (непосредственно в элементе `schema`)
- У именованных объявлений определяется атрибут `name`

```
<xs:simpleType name="newType">
    <xs:restriction base="xs:integer">
        <xs:minInclusive value="0"/>
        <xs:maxInclusive value="100"/>
    </xs:restriction>
</xs:simpleType>
<xs:complexType name="compType">
    <xs:sequence>
        <xs:element name="to" type="xs:string"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="heading" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
    </xs:sequence>
</xs:complexType>
<xs:element name="age" type="newType"/>
<xs:element name="note" type="compType"/>
```



Использование элементов и атрибутов

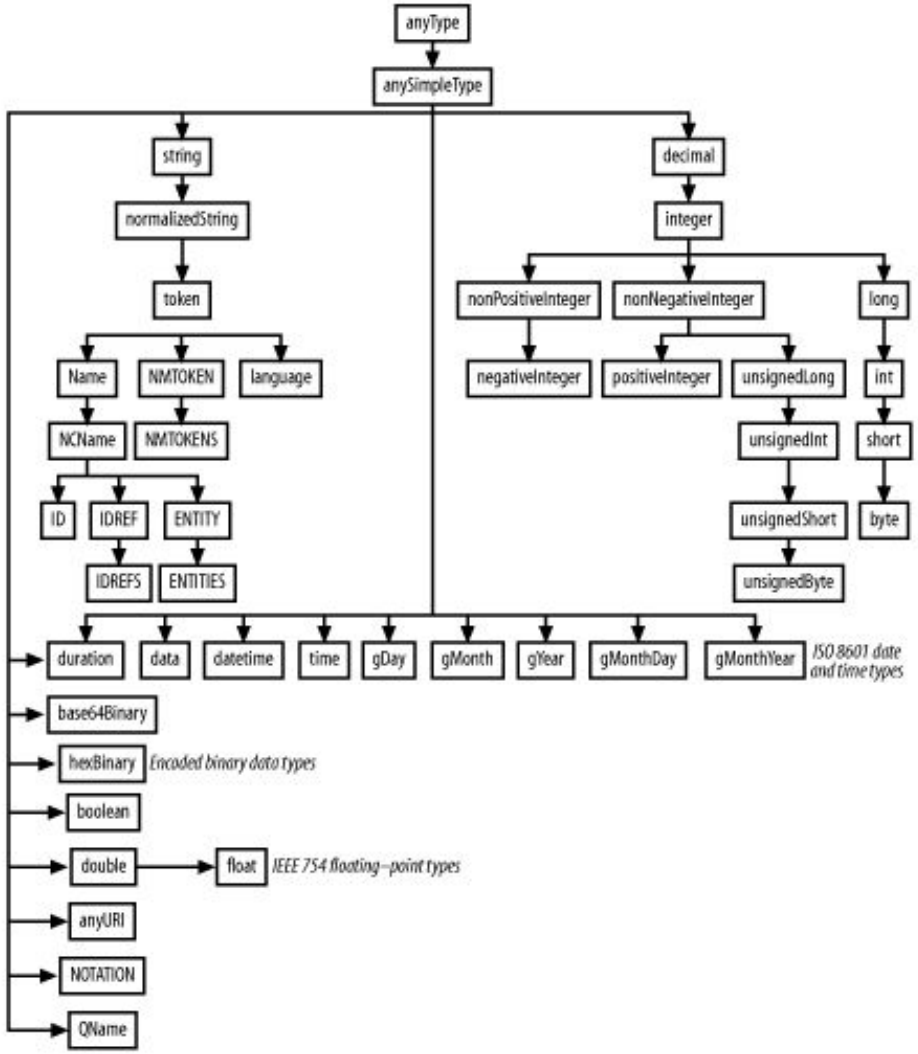
- На именованные объявления элементов и атрибутов можно ссылаться с помощью атрибута `ref`

```
<xs:attribute name="blue"/>
<xs:complexType name="eyeColorType">
    <xs:attribute ref="blue"/>
    <xs:attribute name="light"/>
</xs:complexType>

<xs:element name="eyeColor" type="eyeColorType"/>
<xs:element name="Catalog">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="eyeColor"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

Базовые типы

XML schema содержит 44 базовых типа, общие из которых:

- string
- decimal
- integer
- boolean
- date
- time



Ограничения (Restrictions)

Ограничения предназначены для контроля возможных значений элементов и атрибутов XML.

Элемент ограничения	Описание
enumeration	Определяет одно из возможных значений
fractionDigits	Задает количество знаков после запятой (>=0)
length	Задает количество символов (>=0)
maxExclusive	Определяет максимальное числовое значение.
maxInclusive	Определяет максимальное числовое значение включительно.
maxLength	Задает максимальное количество символов (>=0)
minExclusive	Определяет меньшее числовое значение.
minInclusive	Определяет меньшее числовое значение включительно.
minLength	Задает минимальное количество символов.
pattern	Определяет порядок и набор возможных символов
totalDigits	Показывает количество цифр
whiteSpace	Определяет работу с white spaces (line feeds, tabs, spaces и carriage returns)

Ограничения на значения

```
<xs:element name="age">
    <xs:simpleType>
        <xs:restriction base="xs:integer">
            <xs:minInclusive value="0"/>
            <xs:maxInclusive value="100"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<xs:element name="car">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:enumeration value="Audi"/>
            <xs:enumeration value="Golf"/>
            <xs:enumeration value="BMW"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

Ограничение на значение $0 \leq age \leq 100$

Ограничение списком возможных значений

Ограничение с помощью шаблона

```
<xs:element name="letter">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="[a-z]"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<xs:element name="initials">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="[A-Z][A-Z][A-Z]"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

Ограничение в один символ нижнего регистра от “a” до “z”

Ограничение тремя символами верхнего регистра от “a” до “z”

Ограничение с помощью шаблона

```
<xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>
```

Три символа любого регистра от “a” до “z”

```
<xs:pattern value="[xyz]"/>
```

Один символ из трех возможных (x, y или z)

```
<xs:restriction base="xs:integer">
    <xs:pattern value="[0-9][0-9][0-9][0-9][0-9]"/>
</xs:restriction>
```

Пять цифр

```
<xs:pattern value="([a-z])*"/>
```

Любое количество символов нижнего регистра от “a” до “z”

Ограничение с помощью шаблона

```
<xs:pattern value="([a-z][A-Z])+"/>
```

Несколько пар чередующихся регистрами символов

```
<xs:pattern value="male|female"/>
```

Два возможных варианта значения

```
<xs:pattern value="[a-zA-Z0-9]{8}"/>
```

Ровно восемь букв или цифр

Ограничение на длину

```
<xs:element name="password">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:length value="8"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<xs:element name="password">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:minLength value="5"/>
            <xs:maxLength value="8"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

Значение должно содержать ровно 8 символов

Задаем максимальное и минимальное количество символов

Ограничения на пробельные символы

- Ограничение `whiteSpace` может принимать три значения:
  - `preserve` – оставляет все пробелы, символы табуляции и пустой строки как они есть
  - `replace` – заменяет все такие символы на один пробел
  - `collapse` – удаляет все пробелы спереди и сзади и заменяет все промежуточные на одиночные

```
<xs:element name="address">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:whiteSpace value="preserve"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

Индикаторы (Indicators)

В XML Schema существуют 7 элементов-индикаторов которые задают поведение элементов и атрибутов в XML

Индикаторы порядка:

- `all`
- `choice`
- `sequence`

Индикаторы вхождения:

- `maxOccurs`
- `minOccurs`

Индикаторы груп:

- `group`
- `attributeGroup`

Индикаторы порядка

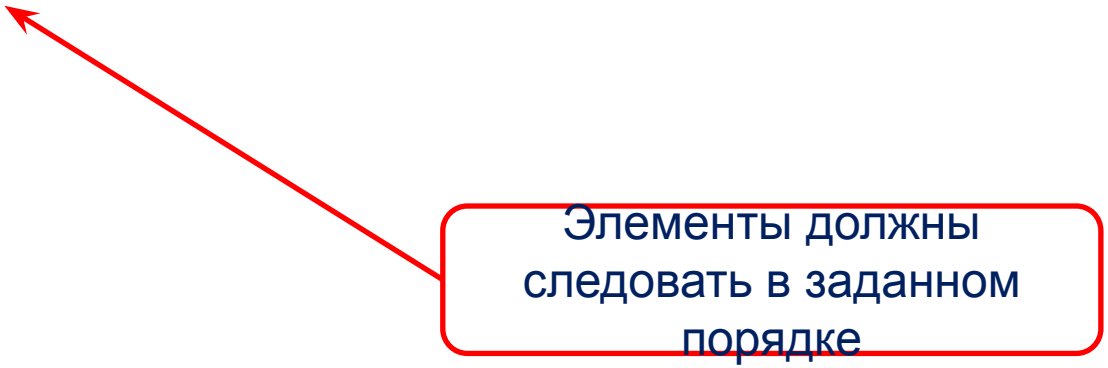
```
<xs:element name="person">
    <xs:complexType>
        <xs:all>
            <xs:element name="firstname" type="xs:string"/>
            <xs:element name="lastname" type="xs:string"/>
        </xs:all>
    </xs:complexType>
</xs:element>

<xs:element name="person">
    <xs:complexType>
        <xs:choice>
            <xs:element name="employee" type="employee"/>
            <xs:element name="member" type="member"/>
        </xs:choice>
    </xs:complexType>
</xs:element>
```

Элементы могут следовать в любом порядке

Только один из представленных элементов может присутствовать

Индикаторы порядка

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



Индикаторы вхождения

- Индикаторы вхождения показывают количество вхождений элемента
- Если индикатор не указан, то по умолчанию должен быть только один элемент. По умолчанию:
  - minOccurs=1
  - maxOccurs=1

```
<xs:element name="person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="full_name" type="xs:string"/>
      <xs:element name="child_name" type="xs:string"
                  maxOccurs="10"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Элемент <child_name> должен следовать за <full_name> и может быть представлен минимум 1 и максимум 10 раз

Индикаторы вхождения

- Индикатор `maxOccurs` может иметь минимальное значение 1, а `minOccurs` значение 0
- Для указания любого количества вхождений ставим `maxOccurs="unbounded"`

```
<xs:element name="person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="full_name" type="xs:string"/>
      <xs:element name="child_name" type="xs:string"
                  minOccurs="0" maxOccurs="10"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Элемент <child_name> может отсутствовать либо должен следовать за <full_name> в количестве максимум 10 раз

Индикаторы групп (Элемент group)

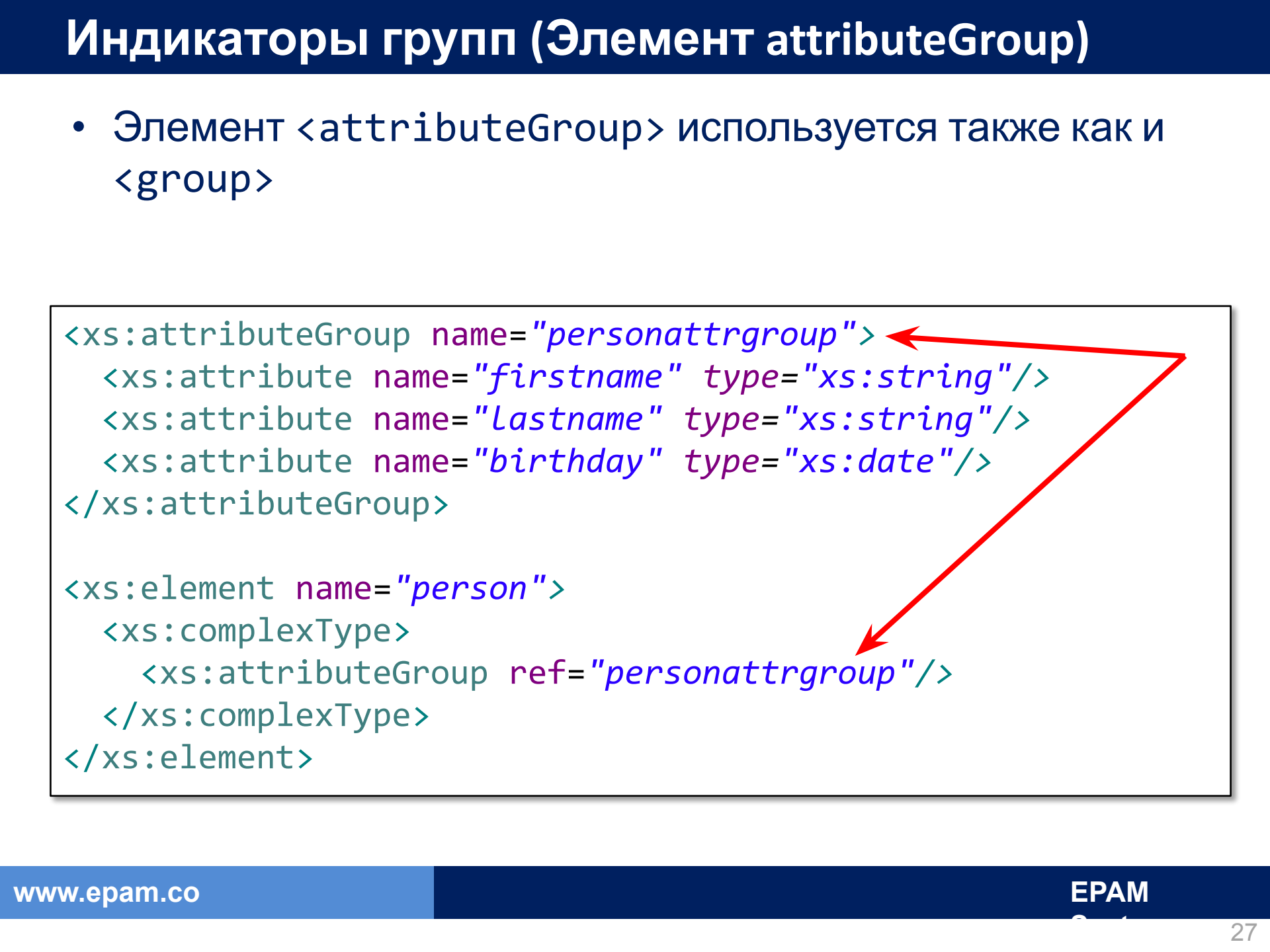
- Индикаторы групп объявляют связанные наборы элементов или атрибутов для последующей ссылки на них

```
<xs:group name="persongroup">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
    <xs:element name="birthday" type="xs:date"/>
  </xs:sequence>
</xs:group>
<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:group ref="persongroup"/>
    <xs:element name="country" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="person" type="personinfo"/>
```

Индикаторы групп (Элемент attributeGroup)

- Элемент <attributeGroup> используется также как и <group>

```
<xs:attributeGroup name="personattrgroup">
  <xs:attribute name="firstname" type="xs:string"/>
  <xs:attribute name="lastname" type="xs:string"/>
  <xs:attribute name="birthday" type="xs:date"/>
</xs:attributeGroup>

<xs:element name="person">
  <xs:complexType>
    <xs:attributeGroup ref="personattrgroup"/>
  </xs:complexType>
</xs:element>
```



Уникальность (Элемент unique)

- Указывает, что значение атрибута или элемента (либо их сочетание) должно быть уникально в данной области видимости

```
<unique id=ID name=NCName
  {any attributes with non-schema Namespace}...>

  Content: (annotation?, selector, field+)
</unique>
```

Уникальность – Пример

```xml
<?xml version="1.0"?>
<cities ... >
  <city name="Milan"
    country="Italy"/>
  <city name="Paris"
    country="France"/>
  <city name="Munich"
    country="Germany"/>
  <city name="Lyon"
    country="France"/>
  <city name="Venice"
    country="Italy"/>
  <city name="Venice"
    country="Italy"/>
</cities>
```

```xml
<xs:element name="cities">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="e:city"
        maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:unique name="cityName">
    <xs:selector xpath="e:city"/>
    <xs:field xpath="@name"/>
  </xs:unique>
</xs:element>
```

Нарушение уникальности

Ключи (Элемент key)

- Указывает, что значение атрибута или элемента (либо набор значений) должно быть ключом в данной области видимости

- Ключ должен быть уникальным в пределах конкретной области действия, не нулевым и постоянно доступным

```
<key id=ID name=NCName
  {any attributes with non-schema Namespace}...>
  Content: (annotation?, selector, field+)
</key>
```

Ссылки на ключи (Элемент keyref)

- Указывает, что значение атрибута или элемента (или набор значений) соответствует значению указанного элемента key или unique

```
<keyref id = ID name = NCName refer = QName
  {any attributes with non-schema Namespace}...>
  Content: (annotation?, selector, field+)
</keyref>
```

Ключи – Пример

```
<?xml version="1.0"?>
<data ... >
 <cities>
  <city name="Milan"
    country="Italy"/>
  <city name="Paris"
    country="France"/>
  <city name="Munich"
    country="Germany"/>
 </cities>
 <persons>
  <person name="Paul"
    city="Paris"/>
  <person name="Jack"
    city="London"/>
  </persons>
</data>
```

```
<xs:element name="data">
 <xs:complexType>
  ...
 </xs:complexType>
 <xs:key name="cityName">
  <xs:selector
    xpath="e:cities/e:city"/>
  <xs:field xpath="@name"/>
 </xs:key>
 <xs:keyref name="personCity"
   refer="e:cityName">
  <xs:selector
    xpath="e:persons/e:person"/>
  <xs:field xpath="@city"/>
 </xs:keyref>
</xs:element>
```

Ключевое значение отсутствует

Использование содержимого других схем

- Элемент <any> используется для добавления элемента из другого пространства имен
- Элемент <any> может быть дочерним для <group>, <sequence>, <all> или <choice>.

```
<xs:element name="catalog">
  <xs:complexType>
    <xs:sequence>
      <xs:any namespace="http://www.fabrikam.com/catalog"
              minOccurs="0" maxOccurs="unbounded"
              processContents="skip"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Документирование схем

- Можно использовать обычные XML комментарии

```
<!-- This is a comment -->
```

- Аннотации могут появляться везде в схеме
  - <annotation> - родительский для <appinfo> и <documentation>
  - <appinfo> - предоставляет информацию для внешних приложений
  - <documentation> - позволяет размещать комментарии для разработчиков

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:annotation>
    <xs:appinfo>Schema for processing submitted applications</xs:appinfo>
    <xs:documentation>Submitted for Human Resources</xs:documentation>
  </xs:annotation>
  <xs:element name="relocationStatus">
    <xs:complexType>
      <xs:annotation>
        <xs:appinfo>No further processing is required.</xs:appinfo>
        <xs:documentation>This element is for determining relocation
status to determine help determine costs.</xs:documentation>
      </xs:annotation>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Описание ссылки на XSD в XML

```
<?xml version="1.0"?>
<note xmlns="http://www.epam.com"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.epam.com note.xsd">
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Местоположение схемы

Пространство имен

- **xmlns="http://www.epam.com"** - пространство имен по умолчанию
- **xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"** - пространство имен экземпляра схемы одним из имен в котором является атрибут schemaLocation
- **xsi:schemaLocation="http://www.epam.com note.xsd"** - определяет какой схеме (пространству имен) соответствует данный XML и где находится схема для валидации данного XML

Описание ссылки на XSD в XML

- xsi:noNamespaceSchemaLocation используется, когда не используются пространства имен

```
<book xsi:noNamespaceSchemaLocation="http://www.epam.com/epam.xsd">
  <title>Presenting XML</title>
  <author>Richard Light</author>
</book>
```

- Можно использовать URL файловой системы

```
xsi:noNamespaceSchemaLocation="file:///d:/xml/schemas/epam.xsd"
```

- xsi:schemaLocation используется, когда префиксы пространств имен явно определены и используются

```
<data:catalog xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
  xmlns:data="http://www.epam.com"
  xsi:schemaLocation="http://www.epam.com
    http://www.epam.com/epam.xsd">
  ...
</data:catalog>
```

Задание

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE Data>
<Data>
  <Customers>
    <Customer>John</Customer>
  </Customers>
  <Items>
    <Item>Item1</Item>
    <Item>Item2</Item>
  </Items>
  <Orders>
    <Order>
      <Customer>John</Customer>
      <Item count="5">Item1</Item>
      <Item count="2">Item2</Item>
    </Order>
  </Orders>
</Data>
```

Создать схему, описывающую подобную структуру данных

Спасибо за внимание!

Контактная информация:
Денис Мурашев
Инструктор
EPAM Systems, Inc.
Адрес: Саратов, Рахова, 181
Email: Denis_Murashev@epam.com
http://www.epam.com