POLITÉCNICO DE LEIRIA
ESCOLA SUPERIOR
DE TECNOLOGIA E GESTÃO

XML Schemas

Systems Integration course

Simple elements

- XML example:

```
<address>Leiria</address>
<age>36</age>
```

- Simple element definitions:

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

XSD attribute

- XML element with an attribute:

```
<contact mode="email">aa@gmail.com</contact>
```

- Attribute definition in XSD

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

or

```
<xs:attribute name="mode" type="xs:string" default="phone"/>
```

XSD restrictions

Restrictions on values

```
<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>
```

Restrictions on a set of values

```
<xs:element name="country" type="countryType"/>

<xs:simpleType name="countryType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Portugal"/>
    <xs:enumeration value="Spain"/>
    <xs:enumeration value="France"/>
  </xs:restriction>
</xs:simpleType>
```

XSD restrictions

Restrictions on a set of values

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

Exactly eight characters that must be lowercase or uppercase letters from a to z, or a number from 0 to 9

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z0-9]{8}"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

XSD restrictions

Restrictions on length

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

Examples of restrictions for data types:

- enumeration, length, maxExclusive, minExclusive, maxInclusive, minInclusive, minLength, pattern, totalDigits

XSD complex types

```
<employee>
  <firstname>John</firstname>
  <lastname>Smith</lastname>
</employee>
```

Alternative 1: The "employee" element can be declared directly by naming the element

```
<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>
```

Alternative 2: The "employee" element can have a type attribute that refers to the name of the complex type to use

```
<xs:element name="employee" type="personinfo"/>

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

Note: you may use the <xs:sequence> or <xs:choice> or <xs:all>

XSD complex types

Several elements can refer to the same complex type

```
<xs:element name="employee" type="personinfo"/>
<xs:element name="student" type="personinfo"/>
<xs:element name="member" type="personinfo"/>

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

Base complex element on an existing complex element and add some elements

```
<xs:element name="employee" type="fullpersoninfo"/>

<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>
```

XSD complex types

To declare the "product" element more compactly

```
<xs:element name="product">
  <xs:complexType>
    <xs:attribute name="prodid" type="xs:positiveInteger"/>
  </xs:complexType>
</xs:element>
```

Give the complexType element a name, and let the "product" element have a type attribute that refers to the name of the complexType (if you use this method, several elements can refer to the same complex type)

```
<xs:element name="product" type="prodtype"/>

<xs:complexType name="prodtype">
  <xs:attribute name="prodid" type="xs:positiveInteger"/>
</xs:complexType>
```

Occurance indicators

Occurrence indicators: maxOccurs or minOccurs

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

You can use only one of the indicators or both

More information: Understanding XML Schemas, MSDN,

http://msdn.microsoft.com/en-us/library/aa468557.aspx