



# Interneto technologijos

---

XML Schema  
Sudėtingi tipai su  
sudėtingu turiniu



# Turinio rūšys

	Žymių/atributų <b>turinio rūšys</b>			
	<b>Mišrus (mixed)</b>	<b>Sudėtingas (complex)</b>	<b>Paprastas (simple)</b>	<b>Tuščias (empty)</b>
Gali turėti vaikinių žymių	Taip	Taip	Ne	Ne
Gali turėti tekstą	Taip	Ne	Taip	Ne



# Sudėtingo turinio struktūra

- Skirtas aprašyti vaikinių žymių eiliškumui

```
<complexContent mixed = boolean>  
  Content: (annotation?, (restriction | extension))  
</complexContent>
```

```
<restriction base = QName>  
  Content: (  
    annotation?,  
    (all | choice | sequence | group)?,  
    (  
      (attribute | attributeGroup)*  
    )  
  )  
</restriction>
```



# Sudėtingo turinio struktūra (2)

```
<extension base = QName>  
  Content: (  
    annotation?,  
    (all | choice | sequence | group)?,  
    (  
      (attribute | attributeGroup)*  
    )  
  )  
</extension>
```



Sintaksiškai tas pats  
kaip ir restriction



# Sudētingo turinio struktūra (3)

---

```
<all
  maxOccurs = 1 : 1
  minOccurs = (0 | 1) : 1>
  Content: (annotation?, element*)
</all>
<choice
  maxOccurs = (nonNegativeInteger | unbounded) : 1
  minOccurs = nonNegativeInteger : 1>
  Content: (annotation?, (element|choice|sequence)*)
</choice>
<sequence
  maxOccurs = (nonNegativeInteger | unbounded) : 1
  minOccurs = nonNegativeInteger : 1>
  Content: (annotation?, (element|choice|sequence)*)
</sequence>
```



# Sudėtingo turinio pavyzdys

---

```
<xs:element name="author">
  <xs:complexType>
    <xs:complexContent>
      <xs:restriction base="xs:anyType">
        <xs:sequence>
          <xs:element name="vardas" type="..." />
          <xs:element name="gimData" type="..." />
        </xs:sequence>
        <xs:attribute name="id" type="..." />
      </xs:restriction>
    <xs:complexContent>
  </xs:complexType>
</xs:element>
```



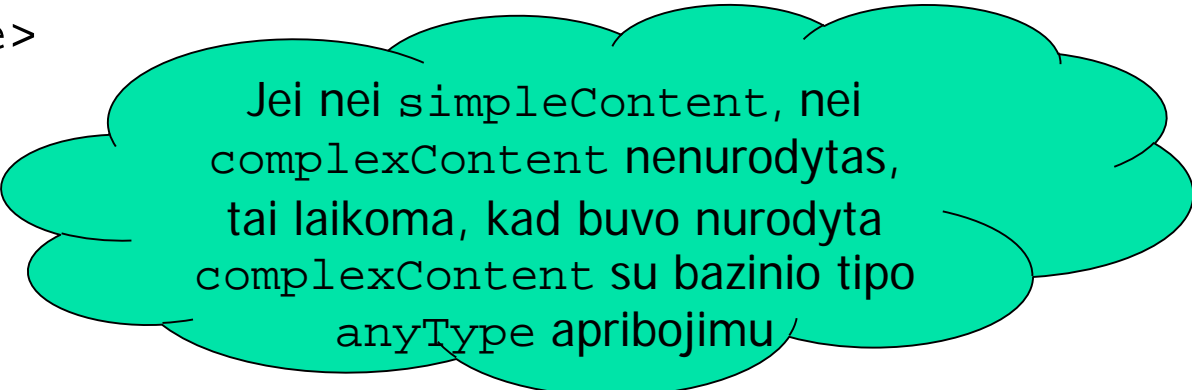
# Sutrumpintas pavyzdys

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

    <xs:sequence>
      <xs:element name="vardas" type="..." />
      <xs:element name="gimData" type="..." />
    </xs:sequence>

    <xs:attribute name="id" type="..." />

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



Jei nei simpleContent, nei  
complexContent nenurodytas,  
tai laikoma, kad buvo nurodyta  
complexContent su bazinio tipo  
anyType apribojimu

# Pavyzdys

choice ir sequence  
žymėms galima pasakyti,  
kiek kartų jos turi būti  
pakartotos dokumente

```
<xs:complexType name="tipas">
  <xs:sequence minOccurs="2" maxOccurs="3">
    <xs:element ref="a" minOccurs="2"
                  maxOccurs="3" />

    <xs:choice>
      <xs:sequence>
        <xs:element ref="b" />
        <xs:element ref="c" />
      </xs:sequence>
      <xs:element ref="d" />
    </xs:choice>

    <xs:element ref="e" />
    <xs:element ref="f" />
  </xs:sequence>
</xs:complexType>
```

choice ir sequence  
viduje galima naudoti  
ir kitas choice bei  
sequence žymes





# Pastabos apie `xs:all`

- `xs:all` viduje galima naudoti tik `xs:element` su atributais:
  - `minOccurs=0` arba `1`,
  - `maxOccurs=1`
- Taip negalima:

```
<xs:complexType name="TMiniHtml">
  <xs:all>
    <xs:element ref="b" minOccurs="0"
      maxOccurs="unbounded" />
    <xs:element ref="i" minOccurs="0"
      maxOccurs="unbounded" />
  </xs:all>
</xs:complexType>
```



# Pastabos apie `xs:all`

---

- Bet galima taip:

```
<xs:complexType name="TMiniHtml">
  <xs:choice minOccurs="0"
              maxOccurs="unbounded">
    <xs:element ref="b" />
    <xs:element ref="i" />
  </xs:choice>
</xs:complexType>
```

# Dviprasmiško turinio taisyklė

- Schema procesorius (validuojantis XML parseris), atlikdamas žymės validavimą, ir žiūrėdamas tik į tą žymę, niekada neturi dvejojti, kokioje schema šakoje jis yra
- Taip negalima:

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

```
<xs:choice>
```

```
<xs:sequence>
```

```
<xs:element ref="vardas" />
```

```
<xs:element ref="slapyvardis" />
```

```
</xs:sequence>
```

```
<xs:sequence>
```

```
<xs:element ref="vardas" />
```

```
<xs:element ref="antras-vardas" minOccurs="0" />
```

```
<xs:element ref="pavardė" />
```

```
</xs:sequence>
```

```
</xs:choice>
```

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

Kai sutiksim žymę vardas, nežinosim, ar po jos privalo eiti slapyvardis, ar antras-vardas



# Neprieštaringo deklaravimo taisyklė

- Draudžiama deklaruoti dvi žymes vienodais pavadinimais ir skirtingais tipais

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



- Tačiau deklaruoti dvi žymes vienodais pavadinimais ir skirtingais tipais galima, jei jos yra *skirtingų* žymių vaikinės žymės



# Naujų sudėtingo turinio tipų išvedimas praplėtimu

```
<xs:complexType name="TAsmuo">
  <xs:sequence>
    <xs:element name="vardas" type="xs:token"/>
    <xs:element name="gimData" type="xs:token"/>
  </xs:sequence>
</xs:complexType>
```

- Pridedame naują žymę (**pridėjimas galimas tik į pabaigą**):

```
<xs:complexType name="TAutorius">
  <xs:complexContent>
    <xs:extension base="TAsmuo">
      <xs:sequence>
        <xs:element name="tematika" type="xs:string"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```



# Naujų sudėtingo turinio tipų išvedimas apribojimu

```
<xs:complexType name="TAsmuo">
  <xs:sequence>
    <xs:element name="vardas" type="xs:token" />
    <xs:element name="gimData" type="xs:token"
                minOccurs="0" />
    <xs:element name="asmensKodas" type="xs:token"
                minOccurs="0" />
  </xs:sequence></xs:complexType>
```

- Uždraudžiam žymę qualification, žymę default padarome privaloma:

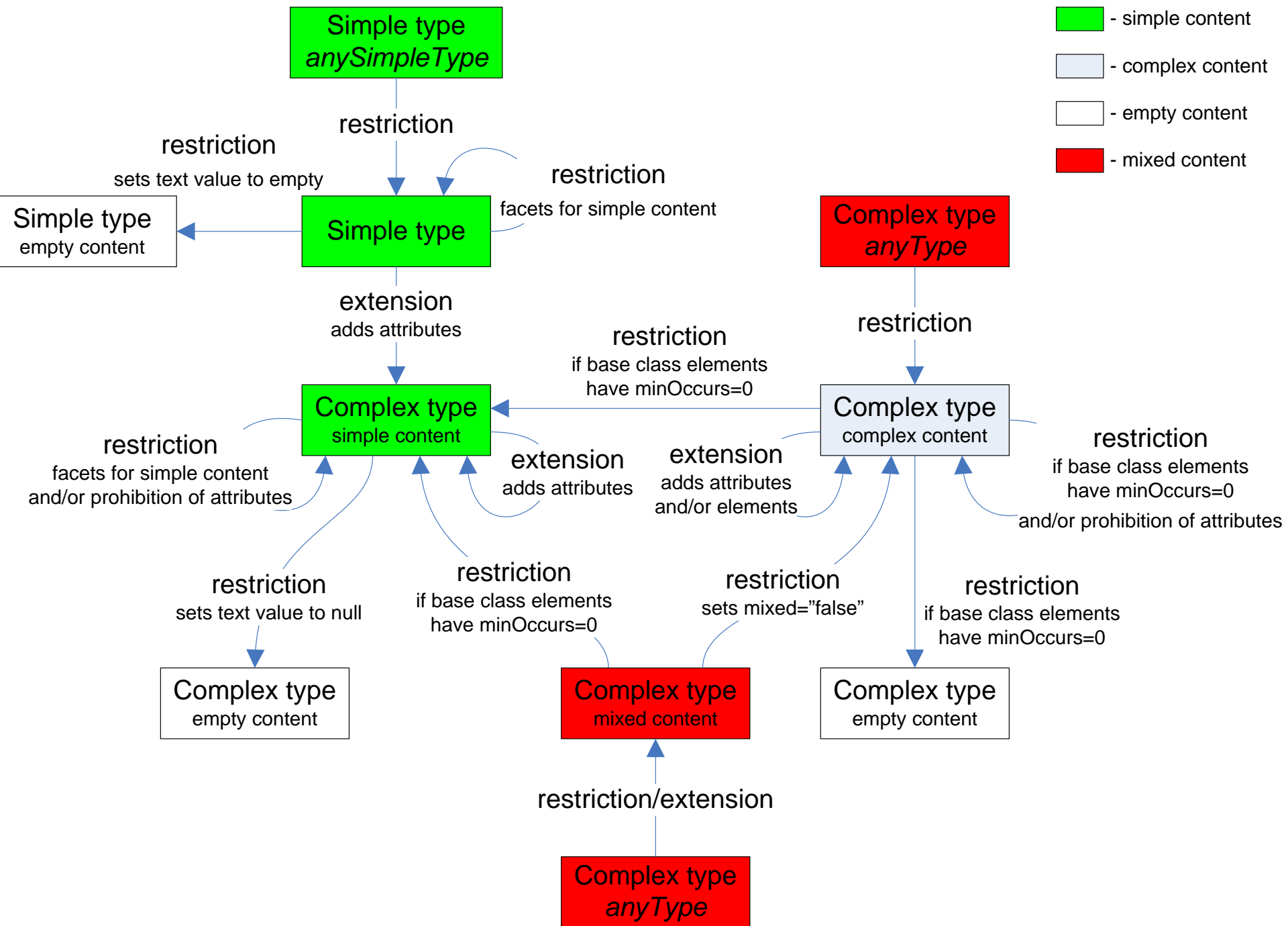
```
<xs:complexType name="TAsmuoLegalus">
  <xs:complexContent>
    <xs:restriction base="TAsmuo">
      <xs:sequence>
        <xs:element name="vardas" type="xs:token" />
        <xs:element name="gimData" type="xs:token" />
      </xs:sequence>
    </xs:restriction>
  </xs:complexContent></xs:complexType>
```



# Naujų sudėtingo turinio tipų išvedimas apribojimu

- Negalima pažeisti apribojimo principo!
- Jei tėvinis tipas paskelbė žymę kaip būtiną (minOccurs  $\geq$  1), jos išmesti negalima.
- Taip negalima:

```
<xs:complexType name="TAsmuoBeDatos">
<xs:complexContent>
  <xs:restriction base="TAsmuo">
    <xs:sequence>
      <xs:element name="vardas" type="xs:token" />
    </xs:sequence>
  </xs:restriction>
</xs:complexContent>
</xs:complexType>
```







# Tuščias turinys

---

- Tuščio turinio pavyzdys (per sudėtingą tipą su sudėtingu turiniu):

```
<xs:element name="book">  
  <xs:complexType>  
    <xs:attribute name="isbn" type="isbnType"/>  
  </xs:complexType>  
</xs:element>
```

- Žymės "book" viduje gali būti tik atributas. Kitos žymės ir tekstas – draudžiami.
- Kitas būdas sukonstruoti tipą su tuščiu turiniu: paskelbti paprastą turinį, ir jį apriboti iki tuščios eilutės.

# Mišraus turinio pavyzdys

- Čia ne mišrus, o sudėtingas turinys:

```
<kažkas>
```

```
<aaa>kva</aaa>
```

```
<bbb>miau</bbb>
```

```
</kažkas>
```

Žymė "kažkas"  
tekstinių **vaikų**  
neturi! Turi tik  
vaikines žymes.

- O va čia – mišrus:

```
<kažkas>
```

```
<aaa>kva</aaa>
```

```
<bbb>miau</bbb>
```

```
kuku
```

```
</kažkas>
```

O čia turi ir  
vaikines žymes, ir  
vaikinį tekstinį  
mazgą "kuku".



# Mišrus turinys

---

- Pavyzdys:

```
<xs:element name="book">
  <xs:complexType mixed="true">
    <xs:all>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="author" type="xs:string"/>
    </xs:all>
    <xs:attribute name="isbn" type="xs:string"/>
  </xs:complexType>
</xs:element>
```

- Šią schemą atitinkantis XML fragmentas:

```
<book isbn="0836217462">
  Funny book by <author>Charles M. Schulz</author>.
  Its title <title>Being a Dog Is a Full-Time
  Job</title> says it all !
</book>
```



# Žodynėlis

---

- Žymė - element
- Paprastas turinys – simple content
- Sudėtingas turinys – complex content
- Paprastas tipas – simple type
- Sudėtingas tipas – complex type
- Apribojimas – restriction
- Išplėtimas – extension