۱

نوشتن XML

XML سیستم گرامری ایجاد زبانهای علامت‌گذاری دلخواه است. یعنی می‌توان آن‌را برای انواع داده‌ها از قبیل داده‌های ریاضی، شیمی، تجاری و غیره به‌کار برد.

برای اینکه بتوان به کمک XML زبان دلخواهی ایجاد کرد باید اصول گرامر XML را دانست. به همین دلیل، ما نیز بحث خود را از این نقطه شروع می‌کنیم. ابتدا اصول قوانین نوشتن اسناد XML و سپس ایجاد زبان در XML را فراخواهید گرفت. در اصطلاح به هر زبانی‌که با XML ساخته می‌شود*XML application* می‌گویند که منظور از آن کاربرد XML است نه برنامه‌های‌کاربردی. چون این اصطلاح ابهام برانگیز است و برنامه‌های کاربردی مثل فتوشاپ را تداعی می‌نماید، در این کتاب از آن استفاده نمی‌شود.

ابزارهای نوشتن XML

برای نوشتن XML نیز مانند HTML می‌توان از یک ویرایشگر یا واژه‌پرداز استفاده کرد. این‌گونه برنامه‌ها عبارتند از: TeachText یا SimpleText در مکینتاش و Notepad یا WordPad در ویندوز. ویرایشگرهای متن ویژه‌ای وجود دارند که می‌توانند هم‌زمان با نوشتن XML درستی آن‌را بررسی نمایند. همچنین برنامه‌هایی هستند که می‌توانند فایل‌های برنامه‌های دیگر مانند برنامه‌های صفحه‌آرایی، صفحه‌گسترده، بانکهای اطلاعاتی و ... را به فرمت اسناد XML تبدیل کنند.

اگر روش ایجاد، بازکردن، ویرایش و ذخیره فایلها یا اسناد را می‌دانید، برای شروع ذکر یک نکته باقی می‌ماند، آن‌هم ذخیره اسناد XML با پسوند xml. است.

عناصر، ویژگیها و مقادیر

XML نیز مانند HTML از سه جزء اصلی *عناصر*، *ویژگیها* و *مقادیر* تشکیل می‌شود. عنصر، اصلی‌ترین قسمت سند XML محسوب می‌گردد و هرچیزی می‌تواند باشد؛ از جمله، عناصر دیگر و یا متن. هر عنصر یک *برچسب شروع* و یک *برچسب پایان* دارد. برچسب شروع نامی دارد که بین علامتهای < و > قرار می‌گیرد و گاهی این نام با صفتهایی همراه است مانند **شکل ۱-۱**. نامی که در نظر می‌گیرید باید هدف عنصر و در مواردی محتویات آن را در صورت وجود معرفی نماید. محتویات عنصر بلافاصله پس از برچسب شروع آورده می‌شوند. برچسب پایان نیز نامی دارد، همان نام برچسب شروع، که در ابتدای آن علامت / قرار گرفته و در بین علامتهای < و > محصور می‌گردد.

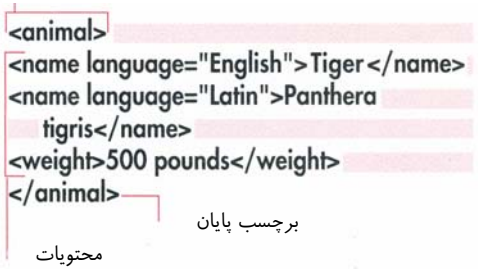
ویژگیها که در برچسب شروع آورده می‌شوند بین علامت کوتیشن قرار می‌گیرند و مقادیری را تعیین می‌کنند که درباره هدف و محتویات (در صورت وجود) عنصر خاصی توضیح بیشتری ارائه می‌دهند (**شکل ۲-۱**). یک ویژگی، اطلاعاتی درباره داده‌های موجود در سند XML می‌دهد که این اطلاعات با خود داده فرق دارد و در اصطلاح به آنها *فراداده* می‌گویند. یک عنصر می‌تواند در صورت نیاز ویژگیهای زیادی داشته باشد؛ ولی هرکدام از ویژگیها باید نام منحصر به فردی داشته باشند.در این درس با طرز نوشتن عناصر، ویژگیها و مقادیر آشنا می‌شویم.

فضای خالی

برای خواناترشدن و ویرایش آسان‌تر عناصر می‌توان در اطراف آنها فضای خالی ایجاد کرد (**شکل ۳-۱**). *تجزیه‌گر*، فضای خالی را در نظر نمی‌گیرد و برنامه‌های مرورگر وب مانندIE5 و Mozilla (نگارش بتایNetscape6) از آن صرف‌نظر می‌کنند (همان‌گونه که با HTML رفتار می‌کنند).

شکل ۱-۱. عنصری با برچسب‌های شروع و پایان که محتویات آن متنی است، نام آن name و محتویات آن Tiger می‌باشد.

شکل ۲-۱. عنصر name دارای یک *ویژگی* با نام Language است که مقدار آن English می‌باشد. توجه داشته باشید که واژه English بخشی از محتویات عنصر name نیست. *ویژگی* درباره محتویات توضیح می‌دهد و English یا English Tiger نام عنصر محسوب نمی‌گردد.



شکل ۳-۱. عنصر animal از سه عنصر دیگر تشکیل شده است که عبارتند از دو عنصر به نام name و یک عنصر به نام weight . عنصر animal متنی ندارد ولی محتویات عناصر name و weight شامل متن هستند. از طرفی عناصر name و weight از عنصر دیگری تشکیل نشده‌اند. همچنین برای خواناتر شدن کد از فضای خالی استفاده شده است.

قوانین نگارش در XML

ساختار زبان XML اصول و قوانین قابل فهمی دارد که کارآیی آن را افزایش می‌دهد و استفاده از آن را بسیار آسان می‌سازد. در این قسمت به شرح مهم‌ترین اجزای قوانین این زبان می‌پردازیم. اگر در فایلهای خود از این قوانین پیروی کنید به برنامه شما یک برنامه *خوش‌فرم* می‌گویند. یک برنامه *خوش‌فرم* می‌تواند توسط مرورگرها به نمایش درآید.

عنصر ریشه (Root element)

هر سند یا فایل XML باید یک عنصر *ریشه* داشته باشد که تمام عناصر سند را شامل گردد. تنها توضیحات و دستورات پردازشی می‌توانند خارج از عنصر ریشه باشند (**شکل ۴-۱**).

برچسب پایان

هر عنصری باید یک برچسب پایان داشته باشد. برای برچسب شروع و پایان می‌توان از یک برچسب استفاده کرد به گونه‌ای که پیش از > یک / قرار بگیرد (**شکل ۵-۱**). به این قبیل برچسبها، برچسبهای خالی می‌گویند. همچنین می‌توان از یک برچسب پایان جداگانه استفاده نمود.

عناصر تو در تو

اگر می‌خواهید عنصر A در ابتدا و عنصر B پس از آن قرار گیرد ابتدا عنصر B و سپس عنصر A را ببندید (**شکل ۵-۱**).

حروف کوچک و بزرگ

XML بین حروف کوچک و بزرگ فرق می‌گذارد. یعنی عناصرANIMAL, animal وAnimal با یکدیگر متفاوتند و می‌توانند ارتباطی با یکدیگر نداشته باشند(**شکل۶-۱**).

مقادیر باید داخل کوتیشن قرار گیرند

مقادیر ویژگیها را باید همیشه بین کوتیشن قرار داد. در این مورد ` ` و " " هر دو صحیح می‌باشند (**شکل ۷-۱**).

موجودیتها باید تعریف شوند

برخلاف HTML درXML به غیر از ۵ موجودیت داخلی (صفحه ۳۱) از هر موجودیتی می‌توان استفاده کرد. پیش از به‌کارگیری موجودیتها باید آنها را در یک DTD تعریف کرد.

```
code.xml
<?xml version="1.0" ?>
<endangered_species>
<name>Tiger</name>
</endangered_species>
```

شکل ٤-١ . در یک سند خوش‌فرم باید یک عنصر (مانند endangered_species) وجود داشته باشد که تمام *عناصر* را شامل شود. خط اول یک دستور پردازشی است و می‌تواند خارج از ریشه قرار گیرد.

```
code.xml
<?xml version="1.0" ?>
<endangered_species>
<name>Tiger</name>
<picture filename="tiger.jpg" />
</endangered_species>
```

شکل ۵-۱. هر عنصر باید قسمت پایان داشته باشد. در برچسب خالی از یک برچسب برای مشخص نمودن ابتدا و انتها استفاده می‌کنند به گونه‌ای که پیش از علامت > یک علامت / قرار می‌گیرد. همان طور که مشاهده می‌کنید از عناصر تو در تو نیز استفاده شده است.

```
code.xml
<name>Tiger</name>
<Name>Tiger</Name>
```

```
code.xml
<name>Tiger</Name>
```

شکل ۶-۱. مثال اول درست است و از قوانین پیروی می‌کند. این دو عنصر هیچ نوع وابستگی به یکدیگر ندارند. مثال دوم غلط است زیرا *برچسب شروع* و پایان آن یکسان نمی‌باشند.

```
code.xml
<picture filename="tiger.jpg"/>
```

شکل ۷-۱. وجود علامت کوتیشن الزامی است. بدین منظور می‌توان از " " یا ` ` استفاده نمود به شرطی که کوتیشن اول و دوم یکسان باشند.

اعلان نگارش XML

به طور کلی، هر سند XML با اعلان نگارش XML مورد استفاده شروع می‌گردد. به این خط از برنامه اعلان XML می‌گویند.

برای اعلان نگارش XML:

۱- در ابتدای سند پیش از هر چیزی عبارت <?xml را تایپ کنید.

۲- سپس عبارت **version="1.0"** را که در حال حاضر تنها نگارش XML است تایپ نمایید.

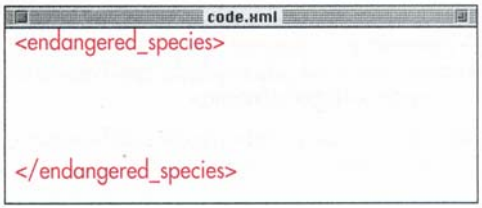
۳- برای تکمیل اعلان نگارش، علامتهای **?>** را تایپ کنید.

نکته‌ها

- به برچسبهایی که با علامتهای ?> شروع و با <? پایان می‌پذیرند دستورات پردازشی می‌گویند. علاوه بر اعلان نگارش XML ، برای تعیین صفحه‌سبک نیز از دستورات پردازشی استفاده می‌شود. درباره صفحه‌های‌سبک در صفحه ۱۷۵ توضیح خواهیم داد.
- عدد مربوط به نگارش برنامه، باید بین علامتهای کوتیشن (" " یا ' ') قرار گیرد.
- نوشتن اعلان XML اختیاری است. اگر بخواهید آن را در سند خود بیاورید باید در اولین خط نوشته شود.
- اگر سند شما به سند دیگری وابسته است باید این موضوع در برنامه قید شود (صفحه‌های ۴۰-۳۹).
- شاید برای تعیین کدگذاری کاراکتری، به دستور پردازشی اولیه XML مثل UTF-8 یا UTF-16 نیز نیاز داشته باشید.

```
code.xml
<?xml version="1.0" ?>
```

شکل ۸-۱. به دلیل اینکه خط اعلان XML یک دستور پردازشی محسوب می‌گردد و یک *عنصر* نمی‌باشد، نیازی به *برچسب پایان* ندارد.



شکل ۹-۱ . در HTML همیشه نام *عنصر* ریشه HTML است ؛ ولی در XML هر نام مجازی می‌تواند به عنوان نام *عنصر* ریشه در نظر گرفته شود. مثل endangered_species که در مثال بالا نشان داده شده است. پیش از *برچسب شروع* و پس از *برچسب پایان عنصر* ریشه نمی‌توان عنصر دیگری را قرار داد.

ایجاد عنصر ریشه

هر سند XML از عنصری تشکیل شده که شامل تمام عناصر سند می‌باشد. از آنجا که این عنصر در برگیرنده تمام عناصر سند می‌باشد به آن عنصر ریشه می‌گویند.

برای ایجاد عنصر ریشه:

۱- در ابتدای سند XML عبارت **<root>** را تایپ کنید. منظور از root نام عنصری است که بقیه عناصر سند را در برمی‌گیرد.

۲- برای اجزای دیگر برنامه (که در طول این کتاب نوشتن آن را یاد می‌گیرید) چند خط خالی ایجاد نمایید.

۳- عبارت **</root>** را تایپ کنید. این نام با نامی که در مرحله نخست تایپ کرده‌اید مطابقت کامل دارد.

نکته‌ها

- حروف کوچک و بزرگ از یکدیگر متمایز هستند. یعنی <NAME> با <Name> و <name> فرق دارد.
- نام مجاز برای عناصر و ویژگیها با یک حرف، علامت _ یا : شروع شده و پس از آن می‌توان از اعداد، حروف ، _ ، - ، . و : استفاده کرد.
- توجه داشته باشید که استفاده از علامت : فقط برای اسامی مجاز است (صفحه ۱۱۳). همچنین هر نوع ترکیبی از حروف x ، m و l چه به صورت کوچک و چه به صورت بزرگ توسط W3C رزرو شده است.
- عنصر ریشه باید برچسب پایان داشته باشد.
- خارج از برچسبهای شروع و پایان عنصر ریشه، عناصر دیگری نمی‌توان معرفی نمود. تنها دستورات پردازشی و الگوها را می‌توان پیش از برچسب شروع ریشه قرار داد (صفحه‌های ۲۴ و ۶۷).

نوشتن عناصر غیر خالی

شما می‌توانید هر عنصری را که می‌خواهید در یک سند XML ایجاد نمایید. بهتر است از اسامی و نامهایی استفاده کنید که مشخص کننده محتویات آنها باشند تا در آینده پردازش اطلاعات آسانتر و قابل فهم‌تر گردد.

برای نوشتن عنصرغیرخالی:

۱- عبارت **<name>** را تایپ کنید. نام name معرف محتویات در برگیرنده این عنصر می‌باشد.

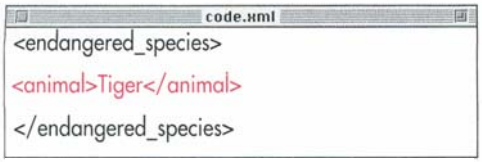
۲- محتویات عنصر را ایجاد نمایید.

۳- عبارت **</name>** را تایپ کنید. این نام باید همان نامی باشد که در مرحله نخست معرفی کرده‌اید.

نکته‌ها

- وجود برچسب پایان الزامی است (در حالی که در HTML گاهی وجود برچسب پایان اختیاری است).
- قوانین نامگذاری عناصر مانند قوانینی است که برای عنصر ریشه به کار می‌روند. همچنین علامت : بیشتر برای تعیین فضای نام به کار می‌رود و در شروع اسامی نمی‌توان از هیچ ترکیبی از حروف x، m و l (چه کوچک، چه بزرگ) استفاده نمود. زیرا این ترکیب توسط W3C رزرو شده است.
- برای نامگذاری استفاده از حروف انگلیسی یا لاتین الزامی نیست.
- اطلاعات لازم برای نوشتن ویژگیها و مقادیر آنها در صفحه ۲۸ توضیح داده شده است.
- با استفاده از یک الگو می‌توان برچسبهای مجاز سند XML را مشخص کرد. برای کسب اطلاعات بیشتر درباره الگوها به صفحه ۶۷ مراجعه نمایید.
- اگر برای عناصر خود نامهای معنی‌دار و متناسب انتخاب کنید استفاده از داده‌ها آسان‌تر می‌گردد.

شکل ۱۰-۱ . اجزای تشکیل دهنده یک *عنصر* ساده در XML عبارتند از: *برچسب شروع*، محتویات (که می‌تواند متن یا *عناصر* دیگر و یا حتی خالی باشد) و *برچسب پایان*. تنها تفاوت *برچسب پایان* با برچسب شروع وجود علامت / پیش از نام *عنصر* در *برچسب پایان* است.



شکل ۱۱-۱ . در اسناد XML تمام *عناصر* باید بین *برچسبهای شروع* و پایان *عنصر* ریشه قرار بگیرند.

ترکیب درست

```
<name>Tiger <baby>cub</baby></name>
```

```
<name>Tiger <baby>cub</name></baby>
```

ترکیب نادرست

شکل ۱۲-۱ . برای اینکه از درستی برچسبهای تو در تو مطمئن شوید هر گروه را در یک خط جداگانه قراردهید. هیچ یک از مجموعه برچسبها نباید با یکدیگر تلاقی پیدا کنند. هر مجموعه داخلی باید کاملاً توسط مجموعه بزرگتر خارجی احاطه شود.

code.xml

```
<endangered_species>
<animal>
<name>Tiger</name>
<threat>poachers</threat>
<weight>500 pounds</weight>
</animal>
</endangered_species>
```

شکل ۱۳-۱ . *عنصر* animal از سه *عنصر* دیگر تشکیل شده است که هر یک از این *عناصر* شامل اطلاعاتی هستند که می‌توانیم به آنها دسترسی داشته باشیم و از آنها استفاده کنیم.

عناصر تو در تو

گاهی یک داده بزرگ به قسمتهای کوچکتر تقسیم می‌گردد؛ بنابراین باید بتوانید هر یک از بخشهای داده را تعریف و با هر یک از آنها کار کنید.

برای ایجاد *عناصرتودرتو*:

۱- همان‌طور که در مرحله ۱ صفحه ۲۶ گفته شد برچسب شروع خارجی‌ترین عنصر را ایجاد نمایید.

۲- عبارت **<inner>** را تایپ کنید. منظور از inner نام اولین بخش از داده اصلی است.

۳- اگر برای عنصر inner محتویاتی در نظر گرفته‌اید، در قسمت مربوطه ایجاد نمایید.

۴- عبارت **</inner>** را تایپ کنید. این نام (inner) باید مطابق نامی باشد که در مرحله ۲ تعیین نموده‌اید.

۵- در صورت تمایل مراحل ۲ تا ۴ را با نامی دیگر تکرار کنید.

۶- برچسب پایان خارجی‌ترین عنصر را مطابق مرحله ۳ صفحه ۲۶ ایجاد نمایید.

نکته‌ها

- این موضوع که هر عنصر داخلی باید توسط عناصر اصلی خارجی احاطه گردد بسیار مهم است. به عبارت دیگر، برچسب پایان عنصر خارجی تا وقتی که عنصر داخلی بسته نشده است نباید نوشته شود. در غیر این صورت، سند ایجاد شده یک سند *خوش‌فرم* نخواهد بود.
- تعداد عناصر تو در تو نامحدود است و به نظر شما بستگی دارد.
- گاهی به عنصر داخلی، عنصر *فرزند* و به عنصر خارجی، عنصر والد می‌گویند.

اضافه کردن ویژگیها

بدون افزودن متن به عناصر می‌توان از طریق ویژگیها، اطلاعات اضافه‌ای ایجاد کرد.

برای اضافه‌کردن ویژگیها:

۱- پیش از علامت > در برچسب شروع عبارت **attribute=** را تایپ کنید. واژه attribute امکان ایجاد داده‌های اضافی را فراهم می‌نماید.

۲- سپس عبارت **"value"**را تایپ کنید. منظور از value همان اطلاعات اضافی می‌باشد. وجود علامت کوتیشن الزامی است.

نکته‌ها

- ویژگیها نیز از قوانین نامگذاری عناصر پیروی می‌کنند (به صفحه ۲۶ مراجعه نمایید).
- بر خلاف HTML در XML مقادیر ویژگیها را باید داخل کوتیشن قرار داد؛" " یا ` ` فرق نمی‌کند ولی برای یک ویژگی باید یکسان باشند.
- اگر مقدار ویژگی، خود شامل علایم جفت‌کوتیشن باشد آن را داخل علایم تک‌کوتیشن قرار دهید و بر عکس.
- برای یک عنصر نمی‌توان دو ویژگی با یک نام در نظر گرفت.
- یک ویژگی نمی‌تواند شامل یک عنصر خارجی باشد (صفحه ۵۸). همچنین برای ویژگی نمی‌توان از علامت > استفاده کرد. در صورت نیاز به این علامت به جای آن باید عبارت **<** را به کار برد.
- ویژگیها اطلاعاتی درباره محتویات عناصر می‌دهند و محتویات عناصر، اطلاعاتی از داده‌ها در اختیار می‌گذارند. به اطلاعات ویژگیها *فراطلاعات* می‌گویند.
- از عناصر تودرتو برای تشخیص بهتر اطلاعات استفاده می‌شود (صفحه ۲۷).

شکل ۱۴-۱. ویژگیها در قسمت برچسب شروع معرفی می‌گردند و از دو بخش تشکیل می‌شوند. مقدار ویژگیها باید بین علایم " " یا ` ` محصور گردند.

```
code.xml
<endangered_species>
<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera tigris</name>
<threat>poachers</threat>
<weight>500 pounds</weight>
</animal>
</endangered_species>
```

شکل ۱۵-۱. *ویژگیها* امکان اضافه‌کردن اطلاعاتی درباره محتویات یک *عنصر* را فراهم می‌کنند.

علامت کوچکتر از

```
<picture filename="tiger.jpg"/>
```

علامت / و علامت بزرگتر از

شکل ۱۶-۱ . عناصر خالی می‌توانند به صورت هم‌زمان برچسب شروع و پایان را داشته باشند. همان طور که در این مثال می‌بینید پس از برچسب شروع ، ویژگی، مقدار و برچسب پایان غیروابسته آورده شده است.

code.xml

```
<endangered_species>
<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera tigris</name>
<threat>poachers</threat>
<weight>500 pounds</weight>
<source sectionid="120"
newspaperid="21"></source>
<picture filename="tiger.jpg" x="200" y="197"/>
</animal>
</endangered_species>
```

شکل ۱۷-۱ . source و picture *عناصر* خالی هستند. *عنصر* خالی source *ویژگیهای* خود را دارد و *عنصر* خالی picture به داده‌های دودویی خارجی که به صورت متنی نمی‌باشند اشاره دارد.

استفاده از عناصر خالی

بعضی از عناصر فاقد محتویات می‌باشند و برای آنها نمی‌توان متنی نوشت. برای مثال عنصر picture را در نظر بگیرید که به یک تصویر اشاره می‌کند و دارای ویژگی است؛ ولی محتویات متنی ندارد.

برای نوشتن یک *عنصرخالی* که دارای یک برچسب شروع و پایان است:

۱- عبارت **name>** را تایپ کنید. منظور از name واژه‌ای است که یک عنصر خالی را مشخص می‌نماید.

۲- ویژگیهای مورد نیاز آن را مطابق مطالب صفحه ۲۸ ایجاد کنید.

۳- برای تکمیل عنصر، عبارت /> را تایپ نمایید.

برای نوشتن یک *عنصرخالی* با برچسبهای شروع و پایان جداگانه:

۱- عبارت <name را تایپ کنید. واژه name معرف یک عنصر خالی است.

۲- اگر می‌خواهید مانند آنچه که در صفحه ۲۸ توضیح داده شده ویژگیهای عنصر را ایجاد نمایید.

۳- برای تکمیل برچسب شروع، علامت > را تایپ کنید.

۴- به منظور تکمیل عنصر، عبارت </name> را تایپ نمایید.عبارت name باید منطبق بر عبارتی باشد که در مرحله ۱ تایپ کرده‌اید.

نکته‌ها

- در XML هر دو روش گفته شده معادل یکدیگر می‌باشند.
- بر خلاف HTML، در اینجا شما نمی‌توانید برچسب شروع بدون برچسب پایان داشته باشید. اگر چنین کاری انجام دهید سند شما *خوش‌فرم* نیست و *تجزیه‌گر* XML اعلام خطا خواهد کرد.

نوشتن توضیحات

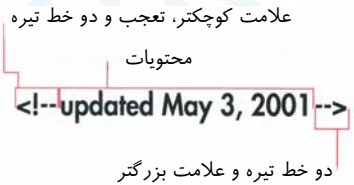
بهتر است همیشه در اسناد XML از توضیحات استفاده نمایید. بدین ترتیب همیشه دلیل استفاده از عناصر یا زمانی را که بخشی از برنامه نیازمند به‌روزرسانی باشد می‌دانید. توضیحات برنامه داخل آن نوشته می‌شوند ولی از دید بینندگان برنامه پنهان می‌مانند.

برای نوشتن توضیحات:

۱- عبارت <!-- را تایپ کنید.
۲- توضیحات لازم را بنویسید.
۳- عبارت --> را تایپ کنید.

نکته‌ها

- بین خط تیره‌هایی که در ابتدا و انتهای توضیحات قرار می‌گیرند نباید فاصله‌ای قرار بگیرد. همچنین بین توضیحات و خط تیره‌ها نباید فاصله باشد. برای مثال توضیحات را باید به صورت <!--this is a comment - -> نوشت.
- بین توضیحات نباید دو خط تیره پشت سر هم نوشت و در ضمن نباید از توضیحات تودرتو استفاده کرد.
- بخش مربوط به توضیحات را می‌توان در طول ایجاد و اشکال‌زدایی برنامه، پنهان ساخت. به چنین توضیحاتی در اصطلاح ˚Commenting out˚ می‌گویند. این توضیحات از دید تجزیه‌گر پنهان می‌مانند و مورد بررسی و غلط‌گیری قرار نمی‌گیرند.
- توضیحات برای به‌روزرسانی و بهینه سازی اسناد XML و *صفحه‌های‌سبک* نیز بسیار مفید می‌باشند.
- برنامه‌های مرورگر، توضیحات را نمایش نمی‌دهند و آنها را تنها می‌توان در کد برنامه مشاهده نمود.



شکل ۱۸-۱ . طرز نوشتن توضیحات در XML و HTML یکسان است.

code.xml

```
<endangered_species>
<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera tigris</name>
<threat>poachers</threat>
<weight>500 pounds</weight>
<!--the source tag references the corresponding
article on the World Wildlife Fund web site-->
<source sectionid="120"
newspaperid="21"></source>
<picture filename="tiger.jpg" x="200" y="197"/>
</animal>
</endangered_species>
```

شکل ۱۹-۱ . به کمک توضیحات امکان اضافه کردن اطلاعات بیشتر در برنامه به وجود می‌آید. این توضیحات به ویژه هنگامی که بخواهید با طرز کار برنامه آشنا شوید یا آن را برای شخص دیگر تشریح کنید بسیار مفید هستند.

code.xml

```
<endangered_species>
<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera tigris</name>
<threat>poachers</threat>
<weight>&lt;500 pounds</weight>
<!--the source tag references the corresponding
article on the World Wildlife Fund web site-->
<source sectionid="120"
newspaperid="21"></source>
<picture filename="tiger.jpg" x="200" y="197"/>
</animal>
</endangered_species>
```

شکل ۲۰-۱. هنگامی که این سند توسط تجزیه‌گر، تجزیه می‌گردد به جای موجودیت ;&Lt علامت < نمایش داده می‌شود.

نوشتن پنج نماد ویژه

در اسناد HTML می‌توان با نوشتن یک علامت **&** و یک نام و علامت **;** نمادهای ویژه را نمایش داد و در اسناد درج نمود. در XML تنها پنج موجودیت هستند که می‌توان از آنها به صورت پیش‌فرض استفاده کرد. بقیه *موجودیتها* پیش از آنکه بتوانند مورد استفاده قرار گیرند باید در یک DTD تعریف شوند.

برای نوشتن نمادهای‌ویژه:

۱- برای ایجاد & عبارت **&** را بنویسید.

۲- برای ایجاد علامت < عبارت **<** را تایپ کنید.

۳- به‌منظور ایجاد علامت > عبارت**>**را تایپ نمایید.

۴- برای ایجاد علامت " عبارت **"** را بنویسید.

۵- علامت ' با تایپ عبارت **'** مشخص می‌شود.

نکته‌ها

- به غیر از پنج نمادی که گفته شده بقیه نمادها و موجودیتها را ابتدا باید در یک DTD تعریف کنید (صفحه ۵۵) سپس در اسناد به کار برید.
- علامتهای < یا & را برای برچسبها یا موجودیتها می‌توان به کاربرد. در غیر این صورت اگر بخواهید به عنوان نمادها و موجودیتهای ویژه از آنها استفاده کنید باید به ترتیبی که در نکته قبلی گفته شد عمل نمایید.
- علامتهای " ، ` یا > را به صورت مستقیم در اسناد می‌توان به کار برد. برای سایر موارد نکته بعد و نکته آخر صفحه ۳۲ را بخوانید.
- یک دلیل خوب برای نوشتن " یا ' به جای علامتهای" و ` این است که شاید مقدار یک ویژگی از علامتهای " و ` تشکیل شده باشد؛ در نتیجه شما می‌توانید مقادیر را به همراه موجودیتهای داخل آنها نمایش دهید.

نمایش عناصر به صورت متن

اگر می‌خواهید در اسناد XML درباره عناصر و ویژگیها چیزی بنویسید باید کاری کنید که تجزیه‌گر از تفسیر آنها صرف‌نظر کنید و در عوض آنها را به صورت متن نمایش دهد. بدین منظور باید چنین اطلاعاتی را در داخل یک بخش CDATA محصور نمایید.

برای نمایش عناصر متنی:

۱- عبارت **<![CDATA[** را تایپ کنید.

۲- عناصر، ویژگیها و محتویاتی را که می‌خواهید نمایش داده شوند ولی تجزیه نگردند ایجاد نمایید.

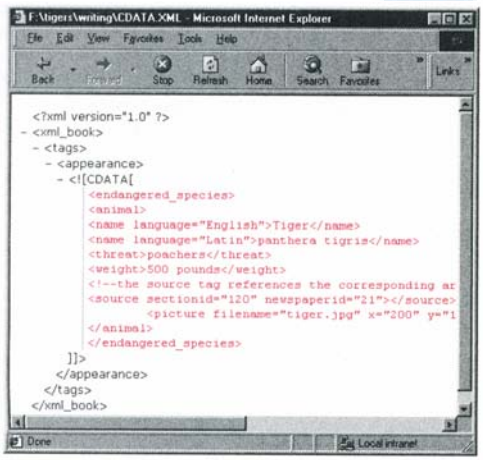
۳- علایم **]]>** را تایپ کنید.

نکته‌ها

- یکی از مزیتهای استفاده از قسمت CDATA محدودکردن *صفحه‌های‌سبک* می‌باشد (صفحه ۱۸۷).
- بهتر است از قسمتهای CDATA به صورت تو در تو استفاده نکنید.
- از آنجایی که نمادهای ویژه مفهوم خاص خود را در قسمت CDATA از دست می‌دهند، در صورتی که برنامه دارای علامت < یا & باشد، به جای **<** یا **&** می‌توانید به صورت مستقیم علامت < یا & را بنویسید.
- قسمتهای CDATA می‌توانند در هر جایی بین برچسب شروع و برچسب پایان عنصر ریشه قرار گیرند.
- اگر بنا به دلایلی بخواهید عبارت **]]>** را پیش از رسیدن به پایان قسمت CDATA تایپ کنید باید علامت > را به صورت **>** بنویسید.برای کسب اطلاعات بیشتر درباره نمادهای ویژه به صفحه ۳۱ و پیوست ۳ مراجعه نمایید.

```
code.xml
<xml_book>
<tags><appearance>
<![CDATA[<endangered_species>
<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera tigris</name>
<threat>poachers</threat>
<weight>500 pounds</weight>
<!--the source tag references the corresponding
article on the World Wildlife Fund web site-->
<source sectionid="120"
newspaperid="21"></source>
<picture filename="tiger.jpg" x="200" y="197"/>
</animal>
</endangered_species>
]]>
</appearance></tags></xml_book>
```

شکل ۲۱-۱ . در این مثال برای نمایش کد واقعی بدون اینکه مورد تجزیه قرار گیرد از CDATA استفاده شده است.



شکل ۲۲-۱ . در این مثال از برنامه Internet Explorer 5 به عنوان تجزیه‌گر استفاده شده است. ببینید که چگونه در قسمت CDATA برچسبها به صورت متن در نظر گرفته می‌شوند؛ ولی xml_book ، tags و برچسبهای appearance توسط تجزیه‌گر مورد پردازش قرار نمی‌گیرند.

۲

ایجاد یک DTD

همان‌طور که گفته شد، در XML فایلی نوشته نمی‌شود؛ بلکه XML برای ایجاد زبانهای نمادین ویژه دلخواه به کار می‌رود و سپس فایلها در آن زبانها نوشته می‌شوند.

شما می‌توانید زبانی را به همراه عناصر و ویژگیهای لازم و ضروری برای یک سند تعریف نمایید. به مجموعه قوانین به کار رفته الگو می‌گویند. برای مثال، یک محقق حیات وحش نیاز به یک EndML دارد. EndML یک واژه ساختگی برای زبان گونه‌های در معرض خطر است. این زبان به سیستمی برای گردآوری اطلاعاتی درباره گونه‌های حیوانات در معرض خطر نیاز دارد. EndML باید عناصری مانند حیوان، گونه‌ها، جمعیت و خطرات داشته باشد.

الگوها، ابزارهای مهمی برای نگهداری پیوستگی اسناد محسوب می‌گردند. با مقایسه سند معینی با الگوی آن اعتبار آن سند تعیین می‌شود (صفحه‌های ۲۴۵-۲۴۴). اگر یک سند با تمام قوانین موجود در الگوی آن مطابقت داشته باشد به آن یک سند معتبر می‌گویند. معتبر بودن یک سند نشانه مطلوب بودن داده‌های آن است.

برای نوشتن الگوها دو سیستم اصلی وجود دارد: الگوی DTDها و الگوی XML. DTD یا تعریف نوع سند ، یک روش قدیمی است، اما به جای استفاده از عبارتهای محدود به صورت گسترده‌ای از سیستم قوانین ویژه استفاده می‌کند. سه فصل بعدی به نوشتن الگوها با روش DTD اختصاص دارند. سیستم new-fangled که توسط W3C برای ایجاد الگوی XML ایجاد شده به صورت کامل در بخش ۳ شرح داده شده است.

تعریف یک DTD داخلی

برای اسناد XML ساده‌ترین راه، ایجاد DTD در داخل خود سند است.

برای تعریف DTD داخلی :

۱- بالای سند XML پس از اعلان XML (صفحه ۲۴)، عبارت **<!DOCTYPE root [** را تایپ کنید. منظور از root عنصر ریشه سند XML می‌باشد که قرار است توسط DTD به کار گرفته شود.

۲- مقداری فضای خالی برای محتویات تعریف نوع سند که توسط اطلاعات فصلهای۳ و ۴ ایجاد خواهید کرد، در نظر بگیرید.

۳- برای تکمیل DTD علامتهای **]>** را تایپ نمایید.

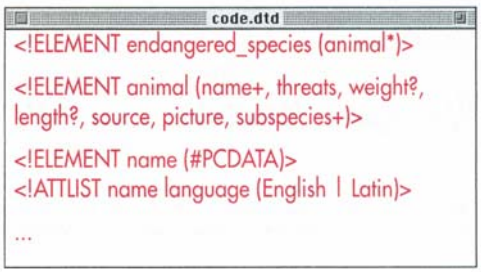
نکته‌ها

- به نکته جالبی در مورد واژه گزینی توجه کنید؛ به کدهایی که به DTD ارجاع داده می‌شوند اعلان نوع سند یا Document Type Declaration می‌گویند. از طرفی به مجموعه قوانین آنها نیز DTD می‌گویند که در این حالت منظور از DTD ، **تعریف** نوع سند یا Document Type **Definition** می‌باشد. برای تشخیص آنها از یکدیگر بدانید که اعلان نوع سند با عبارت <!DOCTYPE شروع و به علامت > ختم می‌گردد. در حالی که مجموعه قوانین یا DTD بین علایم [ ] قرار می‌گیرند. شایان ذکر است DTD می‌تواند به صورت یک فایل جدا یا خارجی نیز تعریف گردد. این موضوع را در صفحه ۳۷ بررسی خواهیم کرد.
- برای معتبر شدن یک سند، باید آن را با قوانین DTD داخلی یا خارجی مشابهی مطابقت داد.

code.xml

```
<?xml version="1.0" ?>
<!DOCTYPE endangered_species [


]>


<endangered_species>
<animal>
```

شکل ۱-۲. در اینجا ابتدای یک DTD داخلی نشان داده شده است. بلافاصله پس از اعلان XML و پیش از برچسبهای بدنه سند XML قرار می‌گیرد.



شکل ۲-۲. نگران چگونگی نوشتن تعاریف نباشید. دو فصل بعدی در این باره صحبت خواهند کرد. نکاتی که در این مرحله باید بدانید عبارتند از: ۱-قوانین یک DTD خارجی در ابتدای یک فایل متنی خالی قرار می‌گیرند. این فایل مستقل است و بخشی از سند XML محسوب نمی‌گردد. ۲- فایل DTD خارجی را باید با پسوند dtd. ذخیره نمایید.

نوشتن یک DTD خارجی

اگر مجموعه‌ای از اسناد وابسته به هم وجود داشته باشد، شاید بخواهید تمام آنها از یک DTD استفاده کنند. به جای کپی DTD در هر یک از سندها، می‌توانید یک فایل خارجی که از DTD تشکیل شده ایجاد و URL آن را به هر یک از اسناد XML که به آن نیاز دارند نسبت دهید.

برای نوشتن یک DTD خارجی :

۱- به کمک یک برنامه واژه‌پرداز یک فایل جدید ایجاد کنید.

۲- قوانین DTD را مطابق آنچه که در بخشهای ۳ و ۴ با عناوین «تعریف عناصر و ویژگیها در یک DTD» و « در DTDها» آورده شده، تعریف نمایید.

۳- فایل متنی را با پسوند dtd. ذخیره کنید.

نکته

- برای کسب اطلاعات بیشتر درباره نامگذاری و استفاده از DTD های خارجی به صفحه‌های ۳۸ تا ۴۰ مراجعه کنید.

نامگذاری یک DTD خارجی

اگر قرار است افراد دیگری از DTD شما استفاده کنند باید برای نامگذاری آن روش استاندارد را به کار برید؛ یعنی از یک شناسه عمومی رسمی یا FPI استفاده کنید؛ زیرا XML برای یافتن آخرین نگارش DTD خارجی روی یک سرور عمومی خارج از وب، ازFPI استفاده می‌کند.

برای نامگذاری DTD خارجی :

۱- اگر DTD شما توسط بدنه‌های استانداردی مثل ISO ایجاد شده علامت + را تایپ کنید.اگر DTD استاندارد نیست علامت - را تایپ نمایید.

۲- درمحلی‌که‌شناسه‌شخص‌یاسازمانی‌که‌DTDرا نوشته‌و نگهداری‌می‌کند،عبارت**Owner//DTD**//راتایپ کنید.

۳- یک فضای خالی ایجاد و سپس عبارت **label** را تایپ کنید. این قسمت برچسبی است که توضیحاتی درباره DTD می‌دهد.

۴- دو حرف اختصاری برای زبان اسناد XML که DTD به کار می‌برد به صورت //**XX**// تایپ کنید. منظور از XX همان دو حرف اختصاری است. برای مثال حروفEN که معرف English می‌باشد (برای زبانهای دیگر به نکاتی که در ادامه گفته شده نگاه کنید).

نکته‌ها

- فهرست حروف اختصاری رسمی زبانها در ISO 639 از طریق آدرس http://www.Unicode.org/unicode/onlinedat/languages.html قابل دسترسی است.
- نامگذاری DTD باعث می‌شود تا به جای استفاده از یک URL ویژه، DTD به کمک یک برچسب شناسایی گردد. درنتیجه یک برنامه‌کاربردی به‌جای جستجوی یک فایل قدیمی در یک سرور، می‌تواند به دنبال یک DTD جدید یا نسخه موجود (یا هر دو) بگردد.

شکل ۳-۲. در اینجا یک نام رسمی برای یک DTD که اسناد XML گونه‌های در معرض خطر را شرح می‌دهد نشان داده شده است.

code.xml

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE endangered_species SYSTEM
    "http://www.cookwood.com/xml/examples
    /dtd_creating/end_species.dtd">

<endangered_species>
<animal>
```

شکل ٤-۲. در این مثال، برای تعریف سند XML از یک DTD خارجی واقع در URL استفاده شده است.

code.xml

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE endangered_species SYSTEM
"http://www.cookwood.com/xml/examples
/dtd_creating/end_species.dtd"
[
<!ELEMENT continent (Asia | Europe | Africa)>
]
>

<endangered_species>
<animal>
```

شکل ۵-۲. اگر بخواهید می‌توانید در انتهای اعلان DOCTYPE ، چند DTD داخلی دیگر اعلان کنید. توجه داشته باشید DTDهای اضافی داخل علائم [ ] قرار گرفته باشند. تمام قوانین داخلی و محلی، قوانین فایلهای خارجی مرتبط را لغو می‌سازند.

اعلان یک DTD خارجی شخصی

اگر برای رسیدن به اهداف خود یک DTD شخصی ایجاد کرده‌اید، تنها راه دسترسی به آن از طریق سند XML شما، استفاده از یک URL است.

۱- در بالای سند و در قسمت اعلان XML عبارت standalone="no" را اضافه کنید.

۲- عبارت DOCTYPE root!> را تایپ کنید. Root یا ریشه نام عنصر ریشه سند XML را که این DTD به کار خواهد برد مطابقت می‌دهد.

۳- با تایپ SYSTEM مشخص می‌شود که DTD خارجی یک DTD شخصی است و استاندارد نمی‌باشد.

۴- عبارت file.dtd را تایپ کنید. فایل file.dtd یک URL (مطلق یا وابسته) است که محل DTD را مشخص می‌نماید.

۵- برای تکمیل اعلان نوع سند علامت > را تایپ نمایید.

نکته

- در صورت نیاز، می‌توانید با اضافه کردن اعلانهای دیگر DTD داخلی پس از پیوند با DTD خارجی، از DTD داخلی و خارجی استفاده نمایید. این کار پس از انجام مرحله ۴ صورت می‌پذیرد. این اعلانها باید توسط علایم [ ] محصور گردند. برای به‌دست آوردن اطلاعات بیشتر درباره DTDهای داخلی به صفحه۳۶ با عنوان «اعلان یک DTD داخلی» مراجعه نمایید. قوانین یک DTD داخلی قوانین DTD خارجی را لغو می‌سازند.

اعلان یک DTD خارجی عمومی

اگر برای مثال، DTD گونه‌های در معرض خطر به صورت عمومی مورد استفاده قرار گیرد و کپی‌های آن به صورت گسترده‌ای منتشر گردد، شاید زمانی فرا رسد که توسط شناسه رسمی عمومی خود، به خودش نسبت داده شود؛ یعنی همان نامی که در صفحه ۳۸ ایجاد کردیم. هنگامی که XML با یک شناسه عمومی رو به رو می‌شود، یک کپی DTD از بهترین منبع موجود تهیه می‌کند. شاید آن را از نزدیکترین و جدیدترین نگارش DTD تهیه نماید. اگر نتواند با استفاده از شناسه عمومی، DTD را پیدا کند، آنگاه از طریق URL اقدام خواهد کرد.

برای ارجاع دادن به یک DTD خارجی عمومی :

۱- بالای سند در قسمت اعلان XML عبارت **Standalone="no"** را تایپ کنید.

۲- عبارت **<!DOCTYPE root** را تایپ کنید. منظور از root نام عنصر ریشه سند XML است که توسط DTD مورد استفاده قرار خواهد گرفت.

۳- عبارت **PUBLIC** را تایپ نمایید. این عبارت بیان می‌کند که مجموعه قوانین نگارش عناوین در دسترس اسناد XML ، استاندارد و عمومی هستند.

۴- **"DTD_name"** را تایپ کنید. DTD_name نام رسمی DTD است که شما به آن ارجاع می‌دهید.

۵- حال **"file.dtd"** را تایپ کنید. منظور از file.dtd همان URL منسوب به DTD عمومی است و محل سرور راه دور را نشان می‌دهد.

۶- با تایپ علامت > تعریف نوع سند را کامل نمایید.

نکته

- بایکDTDداخلیمی‌توانیدDTDخارجیرا لغو سازید. جزئیات بیشتر در نکته صفحه ۳۹ گفته شده است.

```
code.xml
<?xml version="1.0" standalone="no"?>
<!DOCTYPE endangered_species PUBLIC
    "-//Liz Castro//DTD End_Species//EN//"
    "http://www.cookwood.com/xml/examples
    /dtd_creating/end_species.dtd">

<endangered_species>
<animal>
```

شکل ۶-۲. حالا تجزیه‌گر XML برای پیدا کردن DTD از شناسه عمومی استفاده می‌کند. شاید این جستجو را در یک محل عمومی انجام دهد. اگر موفق نشود، URL نسبت داده شده به DTD را به کار خواهد برد.

۳

تعریف عناصر و ویژگیها در یک DTD

در فصل دوم با عنوان "ایجاد یک DTD" طرز تنظیم یک DTD را فرا گرفتید. در این فصل با چگونگی ایجاد محتویات آن آشنا خواهید شد. در واقع می‌بینید که نوشتن قوانین تعیین عناصر و ویژگیها در اسناد XML یکسان است و برای DTD های داخلی و خارجی مشابه می‌باشد.

یک DTD باید برای تمام عناصر و ویژگیهایی که در سند XML وجود دارند، قوانینی تعریف کند. در غیر این صورت، سند XML معتبر نخواهد بود. اگر نیازی به اضافه کردن عناصری به سند XML پیدا کردید باید تعاریف آنها را نیز به DTD مربوط بیفزایید یا در صورت تمایل یک DTD جدید ایجاد نمایید.

تعریف عناصر

برای اینکه اسناد XML محتویات و ساختار مشخصی داشته باشند باید محتویات و ساختار هر یک از عناصر موجود در سند XML را تعریف کنید.

برای تعریف یک عنصر :

۱. عبارت tag **ELEMENT!>** را تایپ کنید. منظور از tag نام عنصری است که می‌خواهیم تعریف کنیم.

۲. اگر عنصر محتویاتی ندارد واژه **EMPTY** را تایپ نمایید.

۳. درغیراین‌صورت عبارت **(contents)** را تایپ کنید. منظور از contents توضیحاتی درباره عنصر یا متن محتویات آن است. پرانتزها را فراموش نکنید. گزینه‌های مجاز برای این متغیر در صفحه‌های ۴۸ و ۴۴ توضیح داده شده‌اند. اگر می‌خواهید عنصر شما شامل ترکیبهای گوناگونی از عناصر و متن باشد واژه**ANY** را تایپ کنید.

۴. در آخر برای تکمیل اعلان عنصر علامت > را تایپ نمایید.

نکته‌ها

- ویژگیها جزء محتویات محسوب نمی‌گردند. حتی عناصر خالی می‌توانند دارای ویژگیهایی باشند (صفحه ۴۹).
- هنگام استفاده از ANY باید دقت بیشتری داشته باشید. تمام قسمتهای DTD از قوانینی تشکیل شده که مشخص می‌کند هر عنصر می‌تواند چه محتویاتی داشته باشد و چه محتویاتی نداشته باشد. اگر تمام عناصر بتوانند هر نوع محتویاتی داشته باشند دیگر نیازی به استفاده ازDTD نخواهد بود. چرا که DTDها به سازگاری داده‌ها کمک می‌کنند و الزامی نیستند.
- اگر برای عنصری از واژه ANY استفاده کنید آن گاه این عنصر نمی‌تواند از عناصری تشکیل شود که در DTD تعریف نشده‌اند.

code.dtd
```
<!ELEMENT endangered_species (animal)>
```

شکل ۱-۳ . تمام عناصری که در یک سند XML ظاهر می‌شوند باید تعریف گردند. در این مثال عنصر endangered_species به صورتی تعریف شده که شامل عنصر دیگری به نام animal می‌باشد.

code.dtd
```
<!ELEMENT picture EMPTY>
```

شکل ۲-۳ . عناصری که به داده‌های دودویی اشاره دارند معمولاً به صورت EMPTY تعریف می‌گردند؛ زیرا فاقد داده‌های XML خواهند بود. گاهی با ویژگیهایی نیز همراه هستند (به صفحه ۴۹ نگاه کنید).

code.dtd
```
<!ELEMENT endangered_species ANY>
```

شکل ۳-۳ . مقدار ANY به قدری نامشخص است که بدون استفاده می‌ماند. اگر نمی‌خواهید سند XML محدود باشد، باید از DTD صرف‌نظر کنید. در این مثال عنصر endangered_species می‌تواند شامل هر چیزی مثل متن یا عناصردیگرباشد. توجه داشته باشید که عناصر تشکیل دهنده endangered_species نیز باید در DTD تعریف شده باشند.

- هر عنصر می‌تواند از عناصر دیگری تشکیل شود. با وجود این هر عنصر باید یک بار به صورت دقیق تعریف شود. هیچ عنصری تا وقتی که در DTD تعریف نشده باشد نمی‌تواند در یک سند XML معتبر ظاهر گردد.
- شما می‌توانید تعداد عناصر مشخصی را که می‌توانند در یک محل معین وجود داشته باشند کنترل کنید (به صفحه ۴۸ نگاه کنید).
- ترتیب اعلان و تعریف عناصر اهمیتی ندارد. برای مثال، می‌توانید ابتدا عنصر درون یک عنصر را اعلان کنید و سپس عنصر خارجی را تعریف نمایید بدون اینکه مشکلی به‌وجود آید.
- ترتیب ظاهر شدن عناصر یک سند XML را می‌توانید کنترل نمایید (صفحه ۴۶).
- در تمام اجزای XML بین حروف بزرگ و کوچک فرق گذاشته می‌شود. برای مثال عبارت ELEMENT!> با Element!> فرق دارد. می‌توانید از ترکیب حروف بزرگ و کوچک نیز استفاده کنید؛ ولی باید هر جایی که می‌خواهید به چیزی اشاره کنید همان ترکیب را به کار ببرید. بهتر است در نامگذاریها تمام حروف را به صورت کوچک بنویسید تا مجبور نباشید ترکیبهای مختلف را به خاطر بسپارید.
- DTD ها عناصر XML نیستند و برای بستن آنها به علامت / پیش از > نیاز نیست.

تعریف عنصری که محتویات آن به صورت متنی است

برخی از عناصر اسناد XML تنها شامل محتویات متنی می‌باشند. برای مثال عنصر Address شامل عناصر Street ، City ، State و Zip است. عنصر State تنها از دو حرف تشکیل می‌شود.

برای تعریف یک عنصر :

۱- عبارت **<!ELEMENT tag** را تایپ کنید. منظور از tag نام عنصری است که قصد تعریف آن را دارید.

۲- سپس عبارت **(#PCDTA)** را تایپ کنید (پرانتزها فراموش نشوند). این عبارت مشخص می‌کند که محتویات عنصرموردنظر تنها به صورت متنی است.

۳- برای تکمیل‌اعلان نوع‌عنصر،علامت> را تایپ نمایید.

نکته‌ها

- عبارت PCDATA از واژه‌های Parsed Character Data گرفته شده است و به هر چیزی به جز متن نشانه‌گذاری‌شده‌اشاره می‌کند. مواردی‌که PCDATA به آنها اشاره می‌نماید عبارتند از : اعداد، حروف، نمادها و موجودیتها (به صفحه ۵۵ نگاه کنید).
- عنصری که شامل PCDATA است نمی‌تواند از عنصر دیگری تشکیل گردد.
- شما می‌توانید از #PCDATA به عنوان یک سری از گزینه‌ها استفاده کنید (صفحه ۴۷) و ممکن است آن را به صورت ترتیبی مورد استفاده قرار ندهید.
- محدودیت اصلی DTDها این است که نمی‌توانید داده وارد شده را به صورت عددی، داده‌ای، متنی یا هر چیزی دیگر معین کنید. به عبارت دیگر، یک سند XML که شامل: <YEAR>dragon</YEAR> است فقط برای <YEAR>2005</YEAR> معتبر می‌باشد. به این موضوع نوع داده قابل دسترس در الگوی XML می‌گویند (صفحه ۶۷).

```
code.dtd
<!ELEMENT name (#PCDATA)>
<!ELEMENT weight (#PCDATA)>
<!ELEMENT threat (#PCDATA)>
```

شکل ٤-۳ . بیشتر DTDها شامل عناصر متنی می‌باشند.

```
code.xml
<endangered_species >
<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera tigris</name>
<threats>
<threat>poachers</threat>
<threat>habitat destruction</threat>
<threat>trade in tiger bones for traditional Chinese
    medicine (TCM)</threat>
</threats>
<weight>500 pounds</weight>
```

شکل ۵-۳ . به این مثال از یک سند معتبر XML توجه کنید. عنصر name متنی است. درباره ویژگی آن در صفحه ۵۰ صحبت خواهیم کرد. عناصر جداگانه threat نیز تنها شامل متن هستند ولی عنصر threats از عناصر threat تشکیل می‌شود و محتویات آن فقط به‌صورت متنی نیست.

تعریف عنصری که یک فرزند دارد

هنگامی که اطلاعات خود را به قسمتهای کوچکتر تقسیم می‌کنید، عناصری به وجود می‌آیند که از عناصر دیگری تشکیل می‌شوند.

برای تعریف عنصری که یک عنصر فرزند دارد :

۱. عبارت **<!ELEMENT tag** را تایپ کنید. منظور از tag عنصری است که می‌خواهیم تعریف کنیم.

۲. عبارت **(child)** را تایپ نمایید. منظور از child نام عنصری است که عنصر مورد نظر شما را تشکیل می‌دهد.

۳. برای تکمیل اعلان علامت > را تایپ کنید.

نکته‌ها

- هنگامی که اعلام می‌کنید عنصری از عناصر دیگر تشکیل شده است، آن عنصر در هر سند XML که توسط DTD شما مورد استفاده قرارگیرد شامل همان عناصر خواهد بود. درغیراین‌صورت سند شما معتبر نیست.
- اگر برچسبی به گونه‌ای تعریف شده که شامل عنصر دیگری است می‌تواند فقط شامل همان عنصر باشد نه چیز دیگر. یعنی نه متنی دارد و نه عنصر دیگری.
- ایجاد یک عنصر فرزند اختیاری است و می‌تواند چندین بار در یک سند ظاهر شود. جزئیات بیشتر در در صفحه ۴۸ گفته شده است.
- یک عنصر فرزند می‌تواند عناصر والد گوناگونی داشته باشد. عناصر چه والد باشند و چه فرزند باید فقط یک بار تعریف شوند.

code.dtd

```
<!ELEMENT endangered_species (animal)>
```

شکل ۶-۳ . عنصر endangered_species از یک عنصر به نام animal تشکیل شده است.

code.xml

```
<endangered_species>
<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera tigris</name>
<threats>
<threat>poachers</threat>
<threat>habitat destruction</threat>
<threat>trade in tiger bones for traditional Chinese
    medicine (TCM)</threat>
</threats>
<weight>500 pounds</weight>
...
</animal>
</endangered_species>
```

شکل ۷-۳ . عنصر endangered_species تنها شامل عنصر animal است. عنصر animal نیز بر اساس تعریف خود می‌تواند محتویات دیگری داشته باشد. محتویات عنصر animal به تعریف عنصر endangered_species بستگی ندارد.

تعریف عنصری با اجزای مرتب

برخی از عناصر شامل عناصر دیگری هستند که رعایت ترتیب در آنها مهم است. ترتیب عناصر فرزند تشکیل دهنده یک عنصر والد قابل تعریف است.

برای تعریف عنصری که اجزای آن ترتیب دارند :

۱. عبارت **<!ELEMENT tag** را تایپ کنید. tag نام عنصری است که قصد تعریف آن را دارید.

۲. عبارت **(child1** را تایپ کنید. منظور از child1 نخستین عنصر فرزند یک عنصر والد است.

۳. عبارت **child2 ,** را تایپ نمایید. child2 دومین عنصر فرزند همان عنصر والد می‌باشد. بقیه عناصر فرزند را به ترتیب تایپ و با یک علامت کاما و یک فضای خالی از یکدیگر جدا کنید.

۴. مرحله ۳ را برای بقیه عناصر فرزند تکرار نمایید.

۵. برای تکمیل کار، علامت (را تایپ کنید.

نکته‌ها

- مهم‌ترین نکته در نوشتن عناصر مرتب، تایپ علامت کاما است. کاما کاراکتری است که عناصر مرتب (یا گروهی از عناصر) را از یکدیگر جدا می‌سازد.
- می‌توان در تمام قسمتهای یک عنصر مرتب از عبارت #PCDATA استفاده نکرد.
- عناصر مرتب از عناصر دیگر تشکیل می‌گردند. در شکل ۹-۳ عنصر threats از عناصر جداگانه threat تشکیل شده است.
- امکان ایجاد واحدهای مرتب نیز وجود دارد. هر واحد می‌تواند یک عنصر، گزینه‌ای از عناصر (داخل پرانتز) یا ترتیبی از عناصر (داخل پرانتز) باشد.
- هر واحد تعریف شده در مجموعه واحدهای مرتب می‌تواند چندین بار ظاهر گردد (صفحه ۴۸).

code.dtd

```
<!ELEMENT animal (name, threats, weight, length,
    source, picture, subspecies)>
```

شکل ۸-۳ . عنصر مرتب animal به ترتیب شامل عناصر دیگری است که نام برده شده‌اند. شاید این عنصر فقط شامل همین عناصر باشد.

code.xml

```
<endangered_species >
<animal>
<name language="English">Tiger</name>
<threats>
    <threat>poachers</threat>
    <threat>habitat destruction</threat>
    <threat>trade in tiger bones for traditional
    Chinese medicine (TCM)</threat>
    </threats>
<weight>500 pounds</weight>
<length>3 yards from nose to tail</length>
<source sectionid="101" newspaperid="21"/>
<picture filename="tiger.jpg" x="200" y="197"/>
<subspecies>
    <name language="English">Amur or
    Siberian</name>
    <name language="Latin">P.t. altaica</name>
    <region>Far East Russia</region>
    <population year="1999">445</population>
    </subspecies>
</animal>
</endangered_species>
```

شکل ۹-۳ . به این مثال معتبر از یک سند XML توجه کنید. هر عنصر فقط یک بار تعریف شده است. عنصر name نمی‌تواند دو بار ظاهر شود و نمی‌توانیم دو عنصر subspecies داشته باشیم. در این باره در صفحه ۴۸ بیشتر توضیح خواهیم داد.

تعریف گزینه‌ها [Choice]

عناصر می‌توانند از یک یا چند چیز دیگر تشکیل شوند.

برای تعریف گزینه‌های تشکیل دهنده یک عنصر :

۱- عبارت **!ELEMENT tag**> را تایپ کنید. منظور از tag عنصری است که قصد تعریف آن را دارید.

۲- عبارت **(child1** را تایپ کنید. child1 نخستین عنصر فرزند است.

۳- علامت | را تایپ نمایید. بدین ترتیب اعلام می‌کنید که شاید عنصر اول ظاهر شود ولی عنصر دوم ظاهر نگردد و برعکس شاید عنصر اول ظاهر نشود و فقط عنصر دوم ظاهر شود.

۴- عبارت **child2** را تایپ کنید. child2 دومین عنصر فرزند است. در صورتی که عنصر دیگر دیده نشود این عنصر ظاهر می‌گردد.

۵- اگر گزینه‌های دیگری دارید مراحل۳و۴را تکرار کنید.

۶- برای تکمیل فهرست گزینه‌ها، (را تایپ کنید.

۷- به منظور تکمیل اعلان عنصر، > را تایپ نمایید.

نکته‌ها

- پس از مرحله ۶ می‌توانستید با اضافه‌کردن یک علامت * به عنصر این امکان را بدهید که شامل هر تعداد از هر گزینه‌ای باشد. این یکی از روشهای تعریف لیست نامرتب است که عنصر والد آن از چندین عنصر تشکیل شده باشد (صفحه ۴۸).
- گزینه نخست می‌تواند #PCDATA باشد تا بتوان عنصری با محتویات ترکیبی ایجاد کرد؛ اما باید یک علامت * نیز اضافه نمایید.
- می‌توان گزینه‌ها را بین واحدها نیز تعریف کرد. منظور از واحدهای عناصر، گزینه‌های بین عناصر (داخل پرانتز) و عناصر مرتب (داخل پرانتز) می‌باشد.

code.dtd
```
<!ELEMENT characteristics ((weight, length) |
    picture)>
```

شکل ۱۰-۳ . در این مثال عنصر characteristics شامل عناصر مرتب length و weight می‌باشد. از طرفی شامل عنصر picture نیز هست در صورت نمایش یکی از عناصر دیگر می‌تواند نمایش داده نشود.

code.xml
```
<characteristics>
<weight>500 pounds</weight>
<length>3 yards from nose to tail</length>
</characteristics>
```

code.xml
```
<characteristics>
<picture filename="tiger.jpg" x="200" y="197"/>
</characteristics>
```

شکل ۱۱-۳ . این دو مثال صحیح و معتبر هستند.

code.xml
```
<characteristics>
<weight>500 pounds</weight>
</characteristics>
```

code.xml
```
<characteristics>
<weight>500 pounds</weight>
<length>3 yards from nose to tail</length>
<picture filename="tiger.jpg" x="200" y="197"/>
</characteristics>
```

شکل ۱۲-۳ . هیچ یک از این مثالها معتبر نیستند. مثال اول نادرست است زیرا گزینه اول عنصر مرتبی است که ابتدا weight و سپس length را شامل می‌شود. در حالی که در اینجا فقط شامل عنصر weight شده است. مثال دوم نیز غیر معتبر است زیرا ممکن است فقط یکی از گزینه‌ها مورد استفاده قرار گیرد نه هر دو گزینه.

تعریف چند واحد

در DTDها سه نماد ویژه وجود دارد که برای تعیین تعداد واحدهای یک عنصر به کار می‌روند. یک واحد می‌تواند یک عنصر، گزینه‌ای بین دو یا چند عنصر (داخل پرانتز) یا یک عنصر مرتب (داخل پرانتز) باشد.

برای تعریف چند واحد :

۱. در بخش محتویات تعریف عنصر، عبارت **unit** را تایپ کنید. منظور از unit یکی از سه حالت: عنصرمنفرد، گزینه‌ای بین دو یا چند عنصر و یا ترتیبی از عناصر است. دو حالت اخیر باید داخل پرانتز نوشته شوند.

۲. برای اینکه واحد، حداکثر یک بار در عنصر تعریف شده ظاهر گردد علامت ؟ را تایپ کنید.

اگر علامت + را تایپ نمایید، واحد مورد نظر می‌تواند حداقل یک بار در عنصر تعریف شده ظاهر شود. در این حالت تعداد دفعات ظاهر شدن واحد نامحدود و مطابق نظر شما خواهد بود.

با تایپ علامت * واحد مورد نظر، هم می‌تواند هر چند بار که نیاز داشته باشید ظاهر شود و هم هیچ‌گاه در عنصر تعریف شده ظاهر نگردد.

نکته‌ها

- بهتر است تعداد معینی برای واحدها مشخص نکنید. برای مثال نگویید به طور حتم ۳ واحد وجود خواهد داشت. روش پیشرفته‌تر این است که با استفاده از عبارت **(unit , unit,uint+)** حداقل سه واحد در نظر بگیرید.
- علامت ستاره برای فهرستی از گزینه‌های تشکیل دهنده یک عنصر که در پرانتز نوشته می‌شوند به کار می‌رود و مشخص می کند که عنصر مذکور می‌تواند تعداد نامحدودی از گزینه‌های جداگانه با هر ترتیبی داشته باشد.

code.dtd

```
<!ELEMENT animal (name+, threats, weight?,
    length?, source, picture, subspecies*)
```

شکل ۱۳-۳ . نمادهای ویژه، انعطاف پذیری بیشتری به تعاریف می‌دهند. حالا، عنصر animal می‌تواند از یک یا چند عنصر name تشکیل گردد. عناصر weight و length می‌توانند حذف یا فقط یک بار ظاهر شوند و عناصر subspecies یا هیچ‌وقت یا چندین بار تکرار گردند. عناصر threats ، source و picture باید حتماً یک بار ظاهر شوند که این حالت پیش فرض می‌باشد.

code.dtd

```
<!ELEMENT threats (threat, threat, threat+)>
```

شکل ۱۴-۳. عنصر threats حداقل از سه عنصر threat تشکیل می‌شود. یعنی می‌تواند بیشتر از سه عنصر نیز داشته باشد.

ویژگیها

هنگامی که یک عنصر را به قطعه‌های کوچک‌تر اطلاعاتی تقسیم می‌کنید، بهتر است داده مکمل را به خود عنصر اضافه کنید نه به محتویات آن. این همان کاری است که یک ویژگی انجام می‌دهد.

ویژگیها درباره بخشی از محتویات اطلاعات نمی‌دهند. بلکه درباره محتویات صفحه XML اطلاعاتی در اختیار می‌گذارند. برای مثال، بانک اطلاعاتی Endangered Species را در نظر بگیرید. عنصر name شامل ویژگی language است. این ویژگی درباره زبان محتویات عنصر name توضیح می‌دهد.

شما می‌توانید همان اطلاعات را برای عناصر جداگانه به کار برید. یعنی عنصر name شامل یک عنصر language و یک عنصر local_name باشد. هر دو روش خوبند. عناصر، اطلاعاتی را که می‌خواهید نمایش دهید در اختیار می‌گذارند و ویژگیها درباره آن اطلاعات، اطلاعاتی را ارائه می‌دهند.

ویژگیها عناصر خالی هستند که به محتویات عناصر اشاره می‌کنند.

code.xml
```
<population year="1999">445</population>
```

code.xml
```
<population>
<year>1999</year>
<quantity>445</quantity>
</population>
```

شکل ۱۵-۳. در این دو مثال کوچک XML اطلاعات مشابهی دیده می‌شوند. یعنی هر دو مثال می‌گویند که در سال ۱۹۹۹، ۴۴۵ ببر سیبری در حیات وحش وجود داشته‌اند. تفاوت این دو مثال در چگونگی سازماندهی اطلاعات است. در مثال بالا، ۱۹۹۹ یک مقدار ویژگی است. در مثال پایین ۱۹۹۹ و ۴۴۵ محتویات عناصر جداگانه هستند. هر دو روش خوبند. انتخاب با شماست.

تعریف ویژگیهای ساده

در اسناد XML هنگامی ویژگیها ظاهر می‌شوند که در یک DTD تعریف شده باشند.

برای تعریف یک ویژگی :

۱- عبارت**<!ATTLIST tag**را تایپ کنید. منظورازtag نام عنصری است که‌به صورت‌ویژگی ظاهر خواهدشد.

۲- واژه **attribute** را تایپ نمایید. attribute نامی است که اطلاعات بیشتری درباره tag در اختیار می‌گذارد.

۳- عبارت **CDATA** را تایپ کنید. توجه داشته باشید در این حالت اثری از پرانتز و علامتهای (!P# نیست. این در صورتی است که بخواهید مقدار ویژگی را با ترکیب کاراکترها و بدون برچسب بسازید.

اگر می‌خواهید چند گزینه برای ویژگی بسازید که احتمال می‌رود فقط یکی از آنها در سند XML به کار روند، عبارت **(choice_2 | choice_1)** را تایپ کنید. هر گزینه با علامت | از گزینه قبلی جدا می‌شود و تمام گزینه‌ها باید در پرانتز قرار گیرند.

۴- سپس عبارت **"default"** را تایپ نمایید. اگر مقدار ویژگی در مجموعه مشخص نیست مقدار default برای آن در نظر گرفته می‌شود.

چنان‌چه مقدار default مقدار پیش فرض است و می‌خواهید آن‌را برای ویژگی مذکور در نظر بگیرید عبارت **#FLXED "default"** را تایپ کنید. اگر می‌خواهید ویژگی شامل مقادیر گوناگون باشد و فقط از مقدار پیش فرض تشکیل نشود عبارت **#REQUIRED** و اگر می‌خواهید ویژگی، مقدار پیش فرضی نداشته باشد و هر زمان که خواستید آن را حذف نمایید عبارت **#IMPLIED** را تایپ کنید.

۵- مراحل ۲ تا ۴ را برای تمام ویژگیهای عناصر تشکیل دهنده تکرار نمایید.

۶- با تایپ علامت > تعریف ویژگی را کامل کنید.

```
code.dtd
<!ELEMENT population (#PCDATA)>
<!ATTLIST population year CDATA #IMPLIED>
```

شکل ۱۶-۳ . عنصر population شامل ویژگی اختیاری year است(به دلیل عبارت #IMPLIED). به دلیل CDATA، محتویات این ویژگی می‌تواند ترکیبی از کاراکترهای گوناگون باشد.

```
code.xml
<population>445</population>
```

```
code.xml
<population year="1999" >445</population>
```

```
code.xml
<population year="of the
    Rabbit">445</population>
```

شکل ۱۷-۳ . براساسDTD شکل ۱۶-۳ هر سه مثال بالا معتبر هستند. ویژگی اختیاری year (به دلیل عبارت #IMPLIED) و محتویات آن که می‌توانند ترکیبی از کاراکترهای گوناگون باشند. توجه داشته باشید راهی برای اطمینان ازدرستی مقدار ویژگیyear(سال) وجوددارد. بنابراین بایداز XML Schemaاستفاده‌نمایید(صفحه۶۹).

```
code.dtd
<!ELEMENT population (#PCDATA)>
<!ATTLIST population year (1999 | 2000)
    #REQUIRED>
```

شکل ۱۸-۳ . در این مثال فقط امکان وجود دو مقدار برای ویژگی population در عنصر year فراهم شده که عبارتند از 1999 یا 2000. فهرست گزینه‌ها در بین پرانتزها قرارگرفته و با علامت | از یکدیگر جدا شده‌اند. به دلیل وجود #REQUIRED ویژگی مذکور باید محتویاتی داشته باشد.

```
code.xml
<population year="1999">445</population>
```

```
code.xml
<population>445</population>
```

```
code.xml
<population year="1998">445</population>
```

شکل ۱۹-۳. مثال اول براساس DTD شکل ۱۸-۳ معتبر و مثال دوم به دلیل اینکه ویژگی year عبارت #REQUIRED را در نظر نگرفته غیرمعتبر می‌باشد. مثال آخر نیز به دلیل اینکه 1998 از گزینه‌های مجاز محتویات ویژگی نیست غیرمعتبر است.

```
code.dtd
<!ELEMENT population (#PCDATA)>
<!ATTLIST population year CDATA "1999">
```

شکل ۲۰-۳ . در اینجا مقدار پیش فرض ۱۹۹۹ را به ویژگی year اضافه کرده‌ایم.

```
code.xml
<population year="1999">445</population>
```

```
code.xml
<population year="1998">445</population>
```

```
code.xml
<population>445</population>
```

شکل ۲۱-۳ . این سه مثال معتبر می‌باشند. ویژگی year هر مقداری می‌تواند داشته باشد و حتی می‌تواند حذف گردد. نکته جالب این است که اگر مقداری حذف شود، مانند مثال سوم، در صورت لزوم تجزیه‌گر مقدار ۱۹۹۹ را برای ویژگی year در نظر خواهد گرفت.

```
code.dtd
<!ELEMENT population (#PCDATA)>
<!ATTLIST population year CDATA #FIXED "1999">
```

شکل ۲۲-۳ . یک مقدار ثابت می‌تواند برای اطمینان از اینکه یک ویژگی مقدار دهی شده یا نشده مفید واقع شود.

```
code.xml
<population year="1999">445</population>
```

```
code.xml
<population year="1998">445</population>
```

```
code.xml
<population>445</population>
```

شکل ۲۳-۳ . این مثالها مانند مثالهای شکل ۲۱-۳ می‌باشند. در صورتی که اعتبار آنها با DTD شکل ۲۲-۳ سنجیده شود مثال دوم معتبر نخواهد بود. باید مقدار ویژگی ۱۹۹۹ باشد. نه ۱۹۹۸ و نه هیچ کاراکتر دیگری مجاز نیستند. به مثال آخر توجه کنید، تجزیه‌گر مقدار ۱۹۹۹ را به ویژگی year نسبت می‌دهد.

نکته‌ها

- هر گزینه در فهرست گزینه‌ها باید از قوانین اسامی معتبر در XML پیروی کند (صفحه ۲۶).
- می‌توان تمام ویژگیها را در قسمت اعلان یک ویژگی تعریف نمود (مرحله ۵). همچنین می‌توان هر ویژگی را به صورت جداگانه تعریف کرد.
- چند نوع ویژگی مخصوص وجود دارند که در صفحات ۵۲-۵۳ توضیح داده می‌شوند و عبارتند از: ID , IDREF و IDREFS. درباره ویژگیهای مخصوص NMTOKEN و NMTOKENS در صفحه ۵۴ و جزئیات ورودی خروجی ویژگی ENTITY در فصل ۴ توضیح خواهیم داد.
- چنانچه یک ویژگی را با یک مقدار پیش فرض تعریف کنید، تجزیه‌گر XML به صورت خودکار مقدار پیش فرض را اضافه خواهد کرد. البته این موضوع در صورتی عملی می‌گردد که ویژگی مذکور به طور صریح در سند XML ذکر نشده باشد (**شکل ۲۱-۳**).
- اگر ویژگی را به‌همراه عبارت **#FIXED "default"** تعریف کرده باشید، مقدار ویژگی در سند XML باید با مقدار پیش فرض یکسان باشند. اگر چنین نباشد، تجزیه‌گر به صورت خودکار مقدار ویژگی را با مقدار پیش فرض تنظیم می‌نماید (**شکل ۲۳-۳**).
- اگر ویژگیDTD با عبارت **#REQUIRED** تعریف شود ولی در سند XML برای ویژگی مقداری در نظرگرفته نشود، تجزیه‌گر اعلام خطا خواهد کرد.
- تجزیه‌گر، اطلاعات بازگشتی ویژگیهایی را که با عبارت **#IMPLIED** تعریف شده ولی با سند XML تنظیم نشده باشند پیش بینی می‌کند.
- توجه‌داشته‌باشید تمام بخشهای‌تعریف‌ویژگی نسبت به حروف کوچک و بزرگ حساس می‌باشند. اگر عبارتی مثل **#Required** تایپ کنید با آنچه که در DTD گفته شده فرق خواهد داشت.
- برای‌مقادیر پیش‌فرض نمی‌توان به‌صورت هم‌زمان از عبارتهای**#REQUIRED** و **#IMPLIED** استفاده کرد.

تعریف ویژگیهابامقادیر منحصربه‌فرد

در XML چند نوع ویژگی وجود دارد که به برخی از آنها ویژگی مخصوص می‌گویند. برای مثال ویژگی ID نیز یک ویژگی مخصوص است و برای داشتن مقدار منحصر به فرد تعریف شده است و نمی‌تواند در یک سند XML تکرار گردد. از ویژگی ID برای کلیدها و اطلاعات شناسایی مثل کد کالا، کد شناسه مشتری و غیره استفاده می‌شود.

برای تعریف ویژگیهای ID :

۱- برای آغاز تعریف ویژگی، مراحل ۱ تا ۲ صفحه ۵۰ را انجام دهید.

۲- اگر می‌خواهید مقدار منحصر به فردی داشته باشید که در یک سند XML قابل تکرار شدن نباشد عبارت **ID** را تایپ کنید.

۳- همان‌گونه که در مراحل ۴ تا ۶ صفحه ۵۰ گفته شد تعریف ویژگی را کامل نمایید.

نکته‌ها

- اگر دو عنصر در یک سند مقادیر ویژگی ID یکسانی داشته باشند، آن سند معتبر نخواهد بود.
- مقدار یک ویژگی ID تابع قوانین نامگذاری XML است. یعنی باید با یک حرف یا علامت ـ آغاز شود و در ادامه حروف، اعداد، ـ ، . و - به کار رود. در مواردی مثل شماره تلفن و شماره شناسایی ملی می‌توان در ابتدای مقادیر این نوع ویژگیها یک حرف یا علامت ـ قرار داد.
- کاربرد ID به عنوان یک نوع ویژگی مثل کاربرد NMTOKEN است که باید منحصر به فرد باشد (به صفحه ۵۴ مراجعه کنید).

code.dtd

```
<!ELEMENT animal (name+, threats, weight?,
    length?, source, picture, subspecies+)>
<!ATTLIST animal code ID #REQUIRED>
```

شکل ۲۴-۳ . به منظور مشخص کردن عناصر معین در سند XML خود بهتر است ویژگی ID ایجاد کنید.

code.xml

```
<animal code="T143">
<name language="English">Tiger</name>
...
</animal>
<animal code="BR45">
<name language="English">Black Rhino</name>
...
</animal>
```

code.xml

```
<animal code="T143">
<name language="English">Tiger</name>
...
</animal>
<animal code="T143">
<name language="English">Black Rhino</name>
...
</animal>
```

شکل ۲۵-۳ . با توجه به DTD شکل ۲۴-۳ ویژگی code باید در طول سند XML یک مقدار منحصر به فرد داشته باشد؛ بنابراین مثال اول معتبر ولی مثال دوم غیر معتبر است.

```
code.dtd
<!ELEMENT specialized_website (title, url)>
<!ATTLIST specialized_website animal_focus IDREF
    #REQUIRED>
```

شکل ۳-۲۶ . فرض کنید عنصری در یک سند XML مسیر سایت‌های وب متعلق به یک حیوان معین را نگه می‌دارد؛ برای نسبت دادن سایت‌های وب به کد حیوان مورد نظر از IDREF استفاده می‌شود.

```
code.xml
<resources>
<specialized_website animal_focus="T143">
    <title>Tigers in Crisis</title>
    <url>http://www.tigersincrisis.com/</url>
</specialized_website>
<specialized_website animal_focus="T143">
    <title>Tigers!</title>
    <url>http://www.geocities.com/RainForest/6
    612/</url>
</specialized_website>
<specialized_website animal_focus="O735">
    <title>International Otter Survival Fund</title>
    <url>http://www.otter.org/</url>
</specialized_website>
```

شکل ۳-۲۷ . مقدار ویژگی IDREF باید شامل یک نوع ویژگی ID از سند باشد (برای مثال شکل اول ۳-۲۵ را مشاهده کنید). همان‌طور که می‌بینید بیش از یک عنصر Specialized_website وجود دارد که می‌تواند همان مقدار ویژگی animal_focus را داشته باشد.

```
code.dtd
<!ELEMENT general_website (title, url)>
<!ATTLIST general_website contents IDREFS
    #REQUIRED>
```

شکل ۳-۲۸ . ویژگیهای IDREFS از یک سری مقادیر ID تشکیل شده‌اند.

```
code.xml
<general_website contents="T143 O735 BR45">
<title>World Wildlife Fund</title>
<url>http://www.worldwildlife.org/</url>
</general_website>
```

شکل ۳-۲۹ . یک ویژگی از نوع IDREFS باید شامل یک یا چند مقدار فضای خالی باشد که هر کدام در جایی از سند از ویژگیهای نوع ID موجود تشکیل شده‌اند.

ویژگیهایی با مقادیر منحصر به فرد

هرگاه مقدار یک ویژگی به یکی از ویژگیهای ID که در صفحه تعریف کرده‌ایم نسبت داده شود به آن ویژگی IDREF می‌گویند. یک ویژگی IDREFS شامل فهرست جداگانه‌ای از مقادیر ویژگیهای ID سند می‌باشد.

برای نسبت دادن ویژگیها با مقادیر منحصر به فرد :

۱- به منظور آغاز تعریف ویژگی مراحل ۱ تا ۲ صفحه ۵۰ را اجرا نمایید.

۲- با تایپ **IDREF** ، یک ویژگی که مقدار آن با مقدار ویژگی ID دیگری مطابقت دارد تعریف می‌شود.
تایپ **IDREFS** برای ویژگیهایی به کار می‌رود که می‌توانند از چند مقدار ID جداگانه تشکیل شوند.

۳- تعریف ویژگی را مطابق مراحل ۴ تا ۶ صفحه ۵۰ کامل کنید.

نکته‌ها

- تعریف ویژگیهای نوع IDREF تنها زمانی کاربرد دارد که DTD نیز دارای تعریفی برای ویژگیهای نوع ID که به IDREF اشاره می‌کند باشد.
- تعریف ویژگیهای نوع ID که ویژگیهای نوع IDREF به‌آنها اشاره‌می‌کنندبایددرDTD نیز وجودداشته باشد.
- شاید چند ویژگی IDREF به یک ID نسبت داده شوند(شکل۳-۲۷). این‌امرمشکلی به‌وجود نمی‌آورد فقط باید دقت کنید ID ، منحصر به یک عنصر باشد.
- هیچ راهی برای جلوگیری از تکرار اجزای ویژگی نوع IDREFS وجود ندارد. برای مثال عبارت contents = "T143T143T143" ازنظر تجزیه‌گر کاملاً معتبر است و این اعتبار به نظر شما بستگی ندارد. برای کنترل بیشتر محتویات عناصر و ویژگیها باید ازDTDهای XML Schema صرف‌نظر کرد(صفحه۶۷).

محدود کردن ویژگیها به نامهای معتبر XML

در DTD ها می‌توان محدودیتهایی بر ویژگیها اعمال نمود تا یک نام معتبر XML باشد. یعنی با یک حرف یا علامت ـ شروع شده و بقیه کاراکترهای آن شامل حروف، اعداد، علامتهای - ، ـ و نقطه گردد.

برای اینکه مقادیر ویژگیها تابع قوانین نامگذاری در XML باشند :

۱. مراحل ۱ تا ۲ صفحه ۵۰ را برای تعریف ویژگی اجرا کنید.
۲. اگر می‌خواهید مقدار ویژگی یک نام معتبر در XML باشد (مانند آنچه که در صفحه ۲۶ گفته شد) عبارت **NMTOKEN** را تایپ نمایید.
۳. اگر قرار است مقدار ویژگی فهرستی از اسامی XML باشد عبارت **NMTOKENS** را تایپ کنید.
۴. تعریف ویژگی را مانند توضیحات مراحل ۴ تا ۶ صفحه ۵۰ به پایان برسانید.

نکته‌ها

- در ویژگیهای نوع NMTOKEN استفاده از فضای خالی مجاز نیست.
- اگر می‌خواهید مقدار یک ویژگی علاوه بر اینکه تابع اسامی معتبر XML است در سراسر سند XML منحصر به فرد نیز باشد به جای عبارت NMTOKEN از عبارت ID استفاده نمایید (به صفحه ۵۲ نگاه کنید).

code.dtd
```
<!ELEMENT appearance (#PCDATA)>
<!ATTLIST appearance most_prominent_color
    NMTOKEN #IMPLIED>
```

شکل ۳۰-۳. توضیحات مربوط به حیوانات به صورت خلاصه و ساده با عبارت most_prominent_color مقدار دهی شده است. چون مقدار ویژگی یک واژه است (بین حروف آن فضای خالی وجود ندارد) می‌توان آن‌را به نوع nmtoken نسبت داد.

code.xml
```
<appearance most_prominent_color="orange">
The might tiger is typically orange with black stripes
    and some white highlights.
</appearance>
```

code.xml
```
<appearance most_prominent_color="dark orange">
The might tiger is typically dark orange with black
    stripes and some white highlights.
</appearance>
```

کل ۳۱۳۱-۳. تنها مثال اول معتبر است. دلیل غیر معتبر بودن مثال دوم این است که در مقدار ویژگی most_prominent_color فضای خالی به کار رفته است. در ویژگیهای نوع NMTOKEN استفاده از فضای خالی مجاز نیست.

۴

موجودیت‌هاویادداشتها در DTDها

موجودیتها مثل سطوح اسفنجی هستند که اگر به آنها آب اضافه شود اندازه و حجم آنها افزایش می‌یابد. در نتیجه می‌توانید مرجع موجودیت را به صورت مختصر و خلاصه تعریف کنید تا هنگامی‌که در سند XML فراخوانی می‌شود داده‌های آن گسترش یابند. در حقیقت هنگامی که مرجع موجودیتی را در یک سند XML یا یک DTD معرفی و تایپ می‌کنید یک اسفنج خشک ایجاد می‌شود.

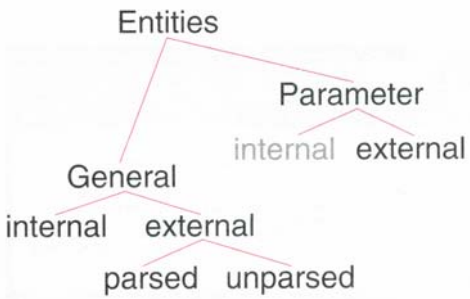
موجودیتها انواع گوناگون دارند ولی روش کار همه آنها یکسان است. تفاوت موجودیتها در محل تعریف و اطلاعات تشکیل دهنده آنهاست.

موجودیتها به دو گروه اصلی تقسیم می‌گردند. موجودیتهای عمومی و موجودیتهای پارامتری. موجودیتهای عمومی، داده‌ها را در سند XML بار (load) می‌کنند. در حالی که موجودیتهای پارامتری به داده‌هایی که بخشی از یک DTD محسوب می‌گردند اشاره می‌نمایند.

موجودیتهای عمومی به دو گروه خارجی و داخلی تقسیم می‌گردند. اگر داخل یک DTD تعریف شوند به آنها موجودیت عمومی داخلی و چنانچه خارج از یک DTD تعریف شوند موجودیت عمومی خارجی می‌گویند.

در نهایت موجودیتهای خارجی به دو گروه تجزیه شده و نشده تقسیم می‌گردند. موجودیتهای تجزیه شده توسط تجزیه‌گر XML بررسی شده و داخل سند XML قرار می‌گیرند. ولی در مورد موجودیتهای تجزیه نشده (که معمولاً به صورت باینری هستند نه به شکل متن) این اتفاق نمی‌افتد.

موجودیتهای پارامتری همیشه مورد تجزیه قرار می‌گیرند و به دو گروه خارجی و داخلی تقسیم می‌شوند. موجودیتهای پارامتری داخلی خیلی محدود می‌باشند. درباره موجودیتهای پارامتری خارجی نیز در صفحه ۶۰ صحبت خواهیم کرد.



شکل ۱-٤ . در DTDها از پنج نوع موجودیت مختلف استفاده می‌شود. موجودیت عمومی در بدنه سند XML به کار می‌رود. موجودیتهای پارامتری تنها در یک DTD مورد استفاده قرار می‌گیرند. موجودیتهای پارامتری داخلی نیز به قدری محدود هستند که بهتر است از آنها صرف‌نظر کنید.

ایجاد میانبر برای متنها

ساده‌ترین نوع موجودیتها داخل DTD تعریف می‌شوند و متنی را ارائه می‌کنند. به این نوع موجودیتها در اصطلاح برنامه نویسی موجودیتهای عمومی داخلی می‌گویند؛ ولی ما برای آنها نام میانبر متن را انتخاب کرده‌ایم.

برای ایجاد میانبرهای متن :

۱- در DTD عبارت **<!ENTITY** را تایپ کنید.

۲- واژه **abbreviation** را که بخشی از یک متن است و به یک موجودیت اشاره می‌کند تایپ کنید(معمولاً کدی است که به جای متن موجودیت تایپ می‌شود).

۳- سپس عبارت **"content"** را تایپ کنید. منظور از content متنی است که هنگام استفاده از موجودیت در سند XML ، ظاهر می‌گردد.

۴- برای تکمیل تعریف موجودیت > را تایپ نمایید.

code.dtd

```
<!ENTITY hwi "Hornocker Wildlife Institute">
```

شکل ۲-٤. از موجودیتهای عمومی داخلی برای تایپ سریع عبارتهای طولانی و دشوار استفاده می‌شود.

نکته‌ها

- جزئیات بیشتر در صفحه ۵۷گفته شده است.
- متنهای خلاصه‌ای که به یک موجودیت اشاره می‌کنند. (مرحله ۲ تمرین قبل) باید از قوانین نامهای معتبر XML پیروی نمایند (به صفحه ۲۶ نگاه کنید).
- پنج موجودیت عمومی داخلی از پیش تعریف شده وجود دارند که عبارتند از **&** ، **<**، **>** و **&apost;** **"** (صفحه۳۱). بقیه موجودیتها باید پیش از استفاده در DTD اعلان گردند.
- یک موجودیت می‌تواند از موجودیتهای دیگر تشکیل شود به شرطی که موجودیتها به صورت دوره تسلسل به یکدیگر اشاره نکنند.
- هرچند مراجع کاراکترهایی که برای اضافه کردن نمادهای ویژه به سند، مورد استفاده قرار می‌گیرند (صفحه ۲۴۷) شبیه موجودیتهای عمومی داخلی هستند ولی موجودیت محسوب نمی‌گردند و به تعریف آنها در DTD نیازی نیست.
- بسیاری از موجودیتهای عمومی و رایج تعریف شده‌اند. در صفحه‌های ۶۰ و ۶۱ جزئیات بیشتری خواهیم گفت.

استفاده از میانبرهای متنی

پس از آنکه یک موجودیت در DTD تعریف کردید، می‌توانید از آن در هر سند XML که به همان DTD مربوط می‌شود استفاده نمایید.

برای استفاده ازمیانبرهای‌متنی:

۱- در سند XML علامت **&** را تایپ کنید.

۲- سپس تایپ کنید **abbreviation** . منظور از abbreviation نام شناسه موجودیت مورد نظر شماست که درمرحله۲صفحه ۵۶یا مرحله۴صفحه ۵۸ معرفی کرده‌اید.

۳- علامت **;** را تایپ نمایید.

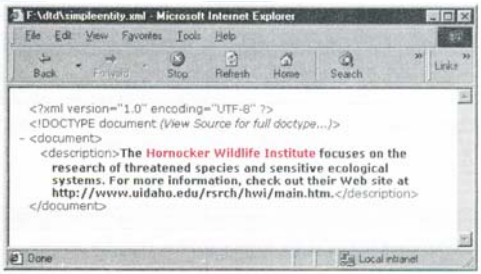
نکته‌ها

- موجودیتها را می‌توان داخل یکدیگر تعریف کرد به شرط آنکه به صورت دوره تسلسل به یکدیگر نسبت داده نشوند.
- تا هنگامی که یک موجودیت در DTD سند تعریف نشود نمی‌تواند مورد استفاده قرار گیرد. در غیر این صورت تجزیه‌گر پیغام خطا صادر خواهد کرد.
- یک موجودیت عمومی (مانند آنهایی که در صفحه‌های۵۶ و ۵۸ توضیح داده شده‌اند) در صورتی که توسط بدنه سند XML مورد استفاده قرار گیرد باید در یک DTD تعریف شود. موجودیتهایی که فقط برای توسعه به DTD اضافه می‌شوند موجودیتهای پارامتری نامیده می‌شوند که از نظر شکل ظاهری و نحوه معرفی کمی با بقیه موجودیتها متفاوت می‌باشند (به صفحه ۶۰ نگاه کنید).

code.xml

```
<description>The &hwi; focuses on the research of
threatened species and sensitive ecological systems.
For more information, check out their Web site at
http://www.uidaho.edu/rsrch/hwi/main.htm.
</description>
```

شکل ۳-٤. بهتر است به جای عبارت Hornocker Wildlife Institute عبارت &hwi را تایپ کنید.



شکل ٤-٤. هنگامی که تجزیه‌گر به بررسی سند می‌پردازد، موجودیت را به شکل گسترده‌تر می‌بینید. در محیط ویندوز از Internet Explorer 5 استفاده می‌شود. توجه داشته باشید تا هنگامی که چند سبک را به کار نگیرید این صفحه‌ها توسط برنامه مرورگر به درستی نمایش داده نخواهند شد. درباره سبکهای مذکور در صفحه ۱۷۵ صحبت خواهیم کرد.

میانبرهای متنی در فایل‌های خارجی

اگر موجودیت بزرگی دارید بهتر و آسان‌تر است که آن را در یک فایل خارجی جداگانه ذخیره نمایید. بدین ترتیب امکان به اشتراک گذاشتن موجودیتها را نیز پیدا می‌کنید.

برای ایجاد میانبر متنی در یک فایل یا سند خارجی :

۱- محتویات موجودیت را در یک فایل خارجی ایجاد کنید. فایل مذکور را به صورت متنی ذخیره نمایید. پسوند فایل مهم نیست.

۲- در فایل XML به منظور اعلان اولیه XML و برای اینکه این سند بتواند از فایل خارجی حاوی تعریف موجودیت استفاده نماید عبارت **standalone="no"** را تایپ کنید (به صفحه ۲۴ نگاه کنید).

۳- برای اینکه سند XML از موجودیت مورد نظر استفاده نماید در DTD برای آغاز تعریف موجودیت عبارت **<!ENTITY** را تایپ کنید.

۴- سپس عبارتی مانند **abbreviation** را تایپ نمایید. منظور از این عبارت نام موجودیت خارجی است.

۵- از آنجا که موجودیت در فایل دیگری تعریف می‌شود واژه **SYSTEM** را تایپ نمایید.

۶- حالا **"entity.url"** را تایپ کنید. entity.url محل فایلی را مشخص می‌کند که در مرحله ۱ ایجاد نموده‌اید.

۷- علامت > را تایپ کنید.

code.xml

```
<description>The Hornocker Wildlife Institute
focuses on the research of threatened species and
sensitive ecological systems. For more information,
check out their Web site at
http://www.uidaho.edu/rsrch/hwi/main.htm.
</description>
```

شکل ۵-٤. این بخش از کد XML شامل موجودیتی است که تعریف شده است. آن را در یک فایل متنی با نام hwi.ent ذخیره می‌کنیم.

code.dtd

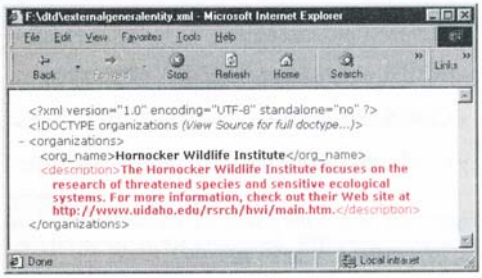
```
<!ENTITY hwi_descrip SYSTEM "hwi.ent">
```

شکل ۶-٤. موجودیت hwi_descrip به آدرس URL فایلی که شامل محتویات موجودیت است اشاره می‌کند. این موجودیت در شکل ۵-۴ تعریف شده است.

code.xml

```
<?xml version="1.0" standalone="no" ?>
...
<organizations>
<org_name>Hornocker Wildlife
Institute</org_name>
&hwi_descrip;
</organizations>
<organizations>
...
```

شکل ۷-٤. عبارت standalone="no" به اعلان XML اضافه شده است. در نتیجه می‌توانید در سند از موجودیت عمومی خارجی استفاده نمایید. همان طور که در اینجا از &hwi_descrip; استفاده شده است.



شکل ۸-٤. موجودیت خارجی شامل عناصر متنی است.

بدین صورت که شما آن را به شکل متن کوتاهی نوشته‌اید و تجزیه‌گر محتویات مرجع آن را نمایش می‌دهد. توجه داشته باشید تمام عناصر موجودیتهای خارجی باید در DTD مربوط به سند تعریف شوند تا معتبر و قابل استفاده باشند.

نکته‌ها

- جزئیات بیشتر درباره استفاده از موجودیت جدید در صفحه ۵۷ گفته شده است.
- این یکی از راههای ایجاد سند منفرد می‌باشد.
- به موجودیتهایی که شامل XML یا متن می‌شوند ولی خارج از DTD سند XML تعریف می‌گردند موجودیتهای عمومی خارجی می‌گویند.

ایجاد و استفاده از میانبرهای DTDها

تا به حال فقط درباره موجودیتهایی صحبت کردیم که مرجع متنی دارند و توسط محتویات اسناد XML مورد استفاده قرار می‌گیرند. علاوه بر این موجودیتهایی وجود دارند که به صورت میانبر بخشی از DTD به کار می‌روند. به این قبیل میانبرها ، موجودیتهای پارامتری خارجی می‌گویند.

برای ایجاد میانبرهای DTDها:

۱- محتویات یک موجودیت را در یک فایل خارجی ایجاد و آن‌را به‌صورت یک فایل متنی ذخیره نمایید. کافی است پسوند فایل یکی از انواع پسوندهای متنی باشد.

۲- در سند XML به منظور معرفی اولیه XML دستور پردازشی **standalone="no"** را تایپ کنید (صفحه ۲۴) تا سند بتواند از فایل خارجی شامل تعریف موجودیت نیز استفاده نماید.

۳- در DTD مربوط به سند XML که از موجودیت مذکور استفاده خواهد کرد، برای شروع تعریف موجودیت عبارت **<!ENTITY** را تایپ نمایید.

۴- با تایپ علامت ٪ ، موجودیت به جای بخشی از یک DTD در نظر گرفته می‌شود.

۵- یک فضای خالی تایپ کنید.

۶- واژه **abbreviation** را تایپ نمایید. منظور از این واژه نام موجودیت خارجی است.

۷- با تایپ **SYSTEM** مشخص می‌شود که موجودیت در سند دیگری تعریف شده است.

همچنین می‌توانید به جای SYSTEM عبارت **PUBLIC "name"** را تایپ کنید. در این صورت name نام فهرست موجودیتهای استاندارد و عمومی می‌باشد.

۸- محل فایلی را که در مرحله نخست ایجاد کرده‌اید با تایپ عبارتی مانند **"entity.url"** مشخص نمایید.

۹- علامت > را تایپ کنید.

code.dtd

```
<!ELEMENT picture EMPTY>
<!ATTLIST picture filename CDATA #REQUIRED
x CDATA #REQUIRED
y CDATA #REQUIRED>
```

شکل ۹-٤. در این شکل بخشی از DTD که قرار است در چند DTD دیگر به کار گرفته شود مشاهده می‌گردد. در اینجا اعلان عنصر picture و ویژگیهای آن دیده می‌شود که در فایل جداگانه‌ای به نام pic.dtd ذخیره شده‌اند.

code.dtd

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE endangered_species [
<!ELEMENT endangered_species (animal*)>
<!ELEMENT animal (name+, picture)>
<!ELEMENT name (#PCDATA)>
<!ATTLIST name language (English | Latin)
#REQUIRED>
<!ENTITY % full_pic SYSTEM "pic.dtd">
```

شکل ۱۰-٤. همان گونه که در این مثال می‌بینید اعلان عنصر picture و ویژگیهای آن در DTD به صورت تعریف موجودیت به کار رفته است.

code.dtd

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE endangered_species [
<!ELEMENT endangered_species (animal*)>
<!ELEMENT animal (name+, picture)>
<!ELEMENT name (#PCDATA)>
<!ATTLIST name language (English | Latin)
#REQUIRED>
<!ENTITY % full_pic SYSTEM "pic.dtd">
%full_pic;
]>
```

شکل ۱۱-٤ . پس از تعریف یک موجودیت با تایپ آن مرجع می‌توانید در هر جا بخواهید آن‌را مورد استفاده قرار دهید. در این مثال مرجع موجودیت با تایپ عبارت %full_pic; مشخص می‌گردد.

پس از تعریف موجودیت می‌توانید از آن استفاده نمایید.

برای استفاده از میانبر DTD :

عبارت **%abbreviation** را تایپ کنید . در اینجا منظور از abbreviation نامی است که هنگام تعریف موجودیت در مرحله ۶ صفحه ۶۰ به کار برده‌اید (**شکل ۱۱-۴**).

نکته‌ها

- موجودیتهای پارامتری باید پیش از به‌کارگیری در DTD تعریف شوند. در این مورد، ترتیبی وجود ندارد.
- می‌توان موجودیتهای پارامتری داخلی نیز ایجاد کرد (درون یک DTD داخلی) ولی در این صورت باید محتویات آنها به صورت کامل بیان گردند و نمی‌توان تنها بخشی از آنها را بیان کرد. در نتیجه چنین موجودیتهایی کاربرد زیادی ندارند.
- خیلی مواظب علایم نشانه‌گذاری باشید. این علایم عبارتند از کوتیشن، علامت بزرگ‌تر از، کوچک‌تر از، و کروشه‌ها که در صورت بی دقتی می‌توانند دردسر زیادی ایجاد کنند.
- از این روش برای اضافه کردن DTD استاندارد فرد دیگری در DTD و اسناد خود می‌توانید استفاده کنید. برای مثال، می‌توانید XHTML DTD را به کار برید و از عناصر تعریف شده آن بدون بروز هیچ مشکلی استفاده نمایید. این عناصر شبیه عناصر HTML هستند و مثل آنها عمل می‌کنند.
- می‌توانید با فهرستی از موجودیتهای استاندارد شده پیوند برقرار نمایید. مثل یکی از موجودیتهایی که در آدرس http://www.schema.net قرار دارد. در این صورت نیازی به تعریف تک تک موجودیتها ندارید و می‌توانید مراجع موجودیتهای کاراکترهای مهم صفحه‌ها را به سادگی به خاطر آورید.
- به خاطر داشته باشید که می‌توانید از هر تعداد DTDکه بخواهید استفاده نمایید.

```
code.dtd
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE endangered_species [
<!ELEMENT endangered_species (animal*)>
<!ELEMENT animal (name+, picture)>
<!ELEMENT name (#PCDATA)>
<!ATTLIST name language (English | Latin)
#REQUIRED>
<!ELEMENT picture EMPTY>
<!ATTLIST picture filename CDATA #REQUIRED
x CDATA #REQUIRED
y CDATA #REQUIRED>
]>
```

شکل ۱۲-٤. تجزیه‌گر XML مرجع موجودیت را با محتویات فایل نسبت داده شده جایگزین می‌نماید(DTD مثال فوق نشان می‌دهد که پس از استفاده از مراجع موجودیت پارامتری چه اتفاقی رخ می‌دهد. در واقع DTD از نظر فیزیکی تغییری نخواهد کرد).

ایجاد موجودیت برای محتویات تجزیه نشدنی

تا حالا فقط درباره موجودیتهایی صحبت کردیم که محتویات آنها متنی بودند. به موجودیتهایی که محتویات متنی دارند موجودیتهای تجزیه شدنی می‌گویند؛ زیرا تجزیه‌گر XML آنها را در سند XML بررسی می‌کند.در این قسمت به معرفی موجودیتهای تجزیه‌نشدنی می‌پردازیم. این موجودیتها می‌توانند محتویات متنی یا غیرمتنی داشته باشند. مهم‌ترین نکته‌ای که در مورد موجودیتهای تجزیه نشدنی وجود دارد این است که تجزیه‌گر XML از بررسی آنها صرف‌نظر می‌نماید؛ بنابراین می‌توانند به عنوان محتویات غیرمتنی و محتویاتی که تابع قوانین XML نیستند در یک سند XML جای گیرند.

برای ایجاد محتویات تجزیه‌نشدنی :

داده‌هایی را که می‌خواهید در سند XML قرار گیرند ایجاد نمایید. این داده‌ها می‌توانند هر چیزی باشند؛ مانند متن معمولی، فایل تصویری، فیلم، فایل pdf و غیره.

به منظور تعریف یک موجودیت برای محتویات تجزیه نشدنی :

۱- در DTD سندی که قصد وارد کردن داده را دارید عبارت**ENTITY!**> را به منظور تعریف موجودیت تجزیه نشدنی تایپ کنید.

۲- نام موجودیت خارجی را تایپ نمایید. این نام می‌تواند چیزی شبیه واژه **abbreviation** باشد.

۳- برای مشخص کردن اینکه موجودیت در سند دیگری تعریف شده است **SYSTEM** را تایپ نمایید.

۴- عبارت **"entity.url"** را تایپ کنید. منظور از entity.url محل فایلی است که شامل محتویات تجزیه نشدنی می‌باشد.

۵- سپس عبارت **NDATA id** را تایپ کنید. به جای id واژه‌ای نوشته می‌شودکه درباره داده تجزیه‌نشدنی توضیح می‌دهد. تمرین بعدی درباره نوشتن id صحبت می‌کند.

۶- با تایپ علامت > تعریف موجودیت را تکمیل کنید.

شکل ۱۳-٤. در اینجا یک نمونه داده تجزیه نشدنی مشاهده می‌شود. این یک فایل تصویری JPEG با نام tiger.jpg است.

code.dtd

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?>
<!DOCTYPE endangered_species [
<!ELEMENT endangered_species (animal*)>
<!ELEMENT animal (name+, photo)>
<!ELEMENT name (#PCDATA) >
<!ATTLIST name language (English | Latin)
#REQUIRED>
<!ENTITY tiger_pic SYSTEM "tiger.jpg" NDATA jpg>
```

شکل ١٤-٤. در این مثال نام موجودیت، tiger_pic است که از طریق واژه SYSTEM به یک فایل خارجی با نام tiger.jpg اشاره می‌نماید و با مراجعه به توضیحات قسمت NOTATION که با عبارت jpg مشخص شده است می‌توانیم اطلاعات بیشتری به دست آوریم (در این خصوص تمرین بعد را مطالعه نمایید).

code.dtd

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?>
<!DOCTYPE endangered_species [
<!ELEMENT endangered_species (animal*)>
<!ELEMENT animal (name+, photo)>
<!ELEMENT name (#PCDATA) >
<!ATTLIST name language (English | Latin)
#REQUIRED>
<!ENTITY tiger_pic SYSTEM "tiger.jpg" NDATA jpg>
<!NOTATION jpg SYSTEM "image/jpeg">
```

شکل ۱۵-٤. نام توضیحات عنصر باید با آنچه که پس از عبارت NDATA در قسمت تعریف موجودیت نوشته شده مطابقت داشته باشد.

برای اضافه کردن اطلاعاتی درباره نوع محتویات، :

۱- پس از تعریف موجودیت مورد نظر یک خط جدید ایجاد و عبارت **<!NOTATION id** را تایپ کنید. منظور از id نام توضیحات موجودیت است و همان واژه‌ای است که در مرحله ۵ صفحه ۶۲ به کار برده‌اید.

۲- عبارت **SYSTEM** را تایپ نمایید.

۳- عبارت **"content_information"** را تایپ کنید. عبارت content_information در واقع داده‌ای را که قرار است در سند قرار گیرد تعریف و مشخص می‌کند. برای نوشتن این قسمت فرمت خاصی وجود ندارد.

۴- در آخر با تایپ علامت > تعریف توضیحات را کامل نمایید.

نکته‌ها

- محتویات یک موجودیت تجزیه‌نشدنی می‌تواند هر چیزی باشد. بیشتر وقتها موجودیتهای تجزیه‌نشدنی شامل تصویر، صدا، فیلم و یا دیگر فایلهای مالتی مدیا می‌باشند. همچنین می‌توانند به صورت فایل متنی نیز باشند. محتویات موجودیتهای تجزیه‌نشدنی هر چیزی می‌تواند باشد؛ زیرا توسط تجزیه‌گر XML مورد بررسی قرار نمی‌گیرند.
- موجودیتهای تجزیه‌نشدنی می‌توانند بخشی از اسناد XML باشند ولی بخشی از یک DTD نمی‌توانند باشند. به عبارت دیگر تمام موجودیتهای تجزیه‌نشدنی موجودیت عمومی محسوب می‌گردند نه موجودیت پارامتری.
- محتویات قسمت Notation می‌تواند یک نوع MIME یا یک آدرس URL که مشخص کننده برنامه کاربردی نمایشگر محتویات یادداشتها یا notation است یا هر چیز دیگری باشد. فرمت خاصی وجود ندارد. هر فایل XML می‌تواند برای تعریف یادداشتها از روش مخصوص خود استفاده نماید.

جاگذاری محتویات تجزیه نشدنی

پس از تعریف یک موجودیت برای محتویات تجزیه نشدنی، همان گونه که در صفحه۶۲ توضیح داده شد، می‌توانید آن را داخل سند XML قرار دهید. موجودیتهای تجزیه نشدنی مرجع ندارند و به چیزی نسبت داده نمی‌شوند (مانند موجودیتهای تجزیه شدنی که پیشتر توضیح داده شده اند). در عوض باید آنها را توسط ویژگیهای مخصوص تعریف شده فراخوانی نمود.

برای تعریف یک ویژگی که محتویات آن به محتویات تجزیه نشدنی اشاره می‌کند :

۱- در DTD عنصری تعریف کنید. این عنصر شامل یک ویژگی است که به یک داده تجزیه نشدنی اشاره خواهد کرد (به صفحه ۴۲ نگاه کنید).

۲- عبارت **<!ATTLIST element_name** را تایپ کنید. element_name عنصری را که در مرحله ۱ تعریف کرده‌اید مشخص می‌نماید.

۳- سپس عبارت **att_name** را تایپ کنید. منظور از att_name نام همان ویژگی است که محتویات آن به یک داده تجزیه نشدنی اشاره می‌کند.

۴- برای تعریف ویژگی مذکور به صورتی که محتویات آن بتواند به داده تجزیه نشدنی اشاره کند، واژه **ENTITY** را تایپ نمایید.

اگر می‌خواهید این ویژگی شامل فهرستی از مراجعی باشد که به فایل‌های داده‌های تجزیه نشدنی اشاره کنند به جای ENTITY واژه **ENTITIES** را تایپ کنید.

۵- مقدار پیش فرض ویژگی را تایپ کنید. جزئیات بیشتر در صفحه ۵۰ گفته شده است.

۶- به منظور تکمیل اعلان ویژگی علامت > را تایپ نمایید.

code.dtd

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?>
<!DOCTYPE endangered_species [
<!ELEMENT endangered_species (animal*)>
<!ELEMENT animal (name+, photo)>
<!ELEMENT name (#PCDATA) >
<!ATTLIST name language (English | Latin)
#REQUIRED>
<!ENTITY tiger_pic SYSTEM "tiger.jpg" NDATA jpg>
<!NOTATION jpg SYSTEM "image/jpeg">
<!ELEMENT photo EMPTY>
<!ATTLIST photo source ENTITY #REQUIRED>
]>
```

شکل ۱۶-٤. ابتدا عنصر photo تعریف شده که شامل یک ویژگی است. این ویژگی به یک داده تجزیه نشدنی اشاره می‌نماید. سپس خود ویژگی موجودیت با نام source تعریف گشته است.

برای جاگذاری محتویات تجزیه نشدنی در یک سند XML :

در بدنه سند XML ، هنگام استفاده از ویژگی، عبارت **att_name="value"** را تایپ کنید. att_name مشخص کننده یک ویژگی است. این ویژگی توسط یک موجودیت تعریف شده است (به صفحه گذشته مراجعه کنید). منظور از value نیز مخفف موجودیتی است که در مرحله ۲ صفحه ۶۲ ایجاد کردید و قرار است شامل داده‌های تجزیه نشدنی باشد.

نکته

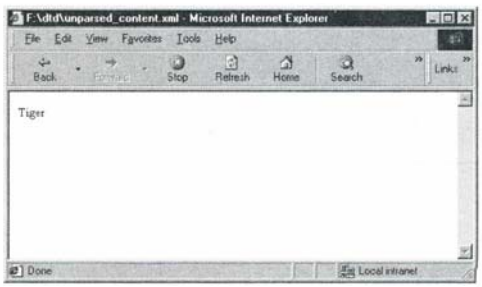
- برنامه‌های Explorer و Netscape 6 قادر به نمایش داده‌های جاگذاری شده نیستند. زیرا این برنامه‌ها نمی‌توانند محتویات تجزیه‌نشدنی را که تجزیه‌گر XML نمایش می‌دهد نشان دهند.
- به منظور نسبت‌دهی داده مشخصی به یک اعلان یادداشت، می‌توانید ویژگی نوع NOTATION بسازید (صفحه ۶۳). روش کار بدین ترتیب است:

<!ATTLIST element_name att_name NOTATION default_value>

code.xml

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?>
<!DOCTYPE endangered_species [
<!ELEMENT endangered_species (animal*)>
<!ELEMENT animal (name+, photo)>
<!ELEMENT name (#PCDATA) >
<!ATTLIST name language (English | Latin)
#REQUIRED>
<!ENTITY tiger_pic SYSTEM "tiger.jpg" NDATA jpg>
<!NOTATION jpg SYSTEM "image/jpeg">
<!ELEMENT photo EMPTY>
<!ATTLIST photo source ENTITY #REQUIRED>
]>

<endangered_species>
<animal>
<name language="English">Tiger</name>
<photo source="tiger_pic"/>
</animal>
</endangered_species>
```

شکل ۱۷-٤ . مقدار ویژگی source با نام موجودیتی که به داده تجزیه نشدنی اشاره می‌کند یکسان است.



شکل ۱۸-٤ . نتایج در برنامه‌های مرورگر مثل Explorer (در اینجا) و Mozilla (نگارش بتای Netscape b) فایل مشاهده هستند. در حال حاضر این دو برنامه تنها برنامه‌های مرورگری می‌باشند که توانایی خواندن فایلهای xml را دارند.

۵

الگوی XML

یک الگو نیز مانند DTD درباره اجزای اسناد XML صحبت می‌کند. از جمله: عناصر و ترتیب آنها، محتویات و ویژگیها (در حقیقت DTD ها نیز نوعی الگو محسوب می‌گردند. در این فصل درباره الگوهایی صحبت می‌شود که در *زبان XML Schema* نوشته شده‌اند).

DTD ها نسبت به الگوهایی که توسط XML Schema نوشته می‌شوند معایبی دارند. نخست آنکه قوانین نگارش DTD ها محدودیتهایی دارند و توسط تجزیه‌گر XML تجزیه نمی‌شوند. دوم اینکه تمام اعلانها در یک DTD به صورت سراسری می‌باشند؛ بنابراین نمی‌توانید دو عنصر گوناگون با یک نام داشته باشید حتی اگر در قسمتهای جداگانه باشند.

نکته آخر و مهمتر اینکه DTD ها نمی‌توانند نوع اطلاعات یا ویژگی یک عنصر را کنترل نمایند.

زبان XML Schema توسط W3C و با هدف برطرف کردن مشکلات موجود ایجاد شده است. XML Schema در خود XML نوشته می‌شود و امکان تعریف عناصر سراسری و محلی را فراهم می‌کند. عناصر سراسری در تمام سند و عناصر محلی در بخشی از سند XML قابل استفاده هستند. همچنین XML Schema از یک سیستم *انواع داده‌ها* تشکیل شده که امکان تعیین نوع محتویات عناصر را فراهم می‌کند. داده‌ها می‌توانند *عدد صحیح* یا *رشته‌ای* باشند. خلاصه اینکه XML Schema امکان کنترل محتویات سند را افزایش می‌دهد.

نکته مهم!

به طورحتم قوانین نگارش توسط W3C تغییر خواهند کرد. اگرتغییرخاصی‌ایجاد شود در سایت مؤلف (صفحه ۱۸) اعمال خواهد شد. همچنین آخرین تغییرات مربوط به XML Schema ازطریق آدرس زیر در دسترس قرار می‌گیرند: http://www.w3.org/xml/schema

نوعهای ساده و پیچیده

در یک الگو، محتویات یک سند می‌توانند از دو نوع تشکیل شوند؛ یکی نوع ساده و دیگری نوع پیچیده. عناصری که محتویات آنها فقط به صورت متن باشند *نوع ساده* و عناصری که از عناصر و ویژگیهای دیگر تشکیل شوند *نوعهای پیچیده* نامیده می‌شوند (ویژگیهایی که فقط محتویات متنی داشته باشند نوع ساده محسوب می‌گردند).

در یک DTD می‌توانید یک عنصر را که محتویات آن فقط به صورت متنی است با عبارت #PCDATA تعریف کنید. PCDATA می‌تواند یک نام، عدد، تاریخ یا هر چیز دیگری باشد. در XML Schema نوع دقیق متن عنصری که از نوع ساده است تعیین می‌گردد. دراین‌خصوص چند *نوع ساده* *درونی* از پیش تعریف شده مانند date (تاریخ)، integer (عدد صحیح) و string (رشته) وجود دارد که بدون هیچ تعریفی قابل استفاده می‌باشند. نوعهای ساده را بیشتر برای کنترل ظاهر محتویات عناصر می‌توان ایجاد کرد. درباره تعریف و استفاده از نوعهای ساده در فصل ۶ صحبت خواهد شد.

عناصر نوع پیچیده به تعریف ساختار یک سند کمک می‌نمایند و با محتویات کاری ندارند. چهار نوع پیچیده اصلی داریم: عناصری که شامل عناصر دیگر هستند، عناصری که از عناصر و متن تشکیل شده‌اند، عناصری که فقط شامل متن هستند و عناصری که خالی می‌باشند. هر یک از این عناصر می‌توانند ویژگیهایی نیز داشته باشند. بر حسب نیاز می‌توان نوع پیچیده خاصی را برای سند XML تعریف کرد (فصل ۷).

اگر برای نوعهای ساده و پیچیده نامی در نظر بگیرید می‌توانید آنها را در تمام قسمتهای الگو به کار برید؛ ولی اگر برای آنها نامی در نظر نگیرید فقط می‌توانید در عنصر مربوط از آنها استفاده کنید.

code.xsd

```
<xsd:element name="weight" type="xsd:string"/>
<xsd:element name="population"
type="xsd:integer"/>
```

شکل ۱-۵. این دو عنصر به کمک نوعهای ساده درونی تعریف شده‌اند. Weight از نوع رشته‌ای و population از نوع عدد صحیح است.

code.xsd

```
<xsd:simpleType name="zipcodeType">
  <xsd:restriction base="xsd:string">
  <xsd:pattern value="\d{5}(-\d{4})?"/>
  </xsd:restriction>
</xsd:simpleType>
```

شکل ۲-۵. این نوع ساده که توسط کاربر تعریف شده محتویات عناصر تعریف شده با zipcodeType را به یک رشته محدود می‌نماید. این رشته از پنج رقم به همراه یک خط‌تیره اختیاری و چهار رقم دیگر تشکیل شده‌است.

code.xsd

```
<xsd:complexType name="endType">
  <xsd:sequence>
  <xsd:element name="animal"
  type="animalType" minOccurs="1"
  maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

شکل ۳-۵. نوع پیچیده endType شامل یک عنصر دیگر به نام animal است که نوع پیچیده animalType را تعریف می‌کند. از نوع پیچیده endType برای تعریف عنصر دیگری که شامل عنصر animal باشد می‌توانید استفاده نمایید.

اعلانهای محلی و سراسری

در DTD تمام عناصر به صورت سراسری اعلان می‌گردند؛ یعنی هر عنصر نام منحصر به فردی دارد و فقط یک‌بار تعریف می‌شود. چند عنصر می‌توانند به یک عنصر نسبت داده شوند؛ بنابراین هر عنصر می‌تواند در چند جای یک سند XML ظاهر گردد و هر جا که باشد تعریف آن همیشه یکسان خواهد بود.

نگارش XML Schema بسیار مهم است. اجزای الگو، به انضمام عناصر، ویژگیها و نام نوعهای ساده و پیچیده (تحت عناوین گروهها و گروههای ویژگی که آنها را مورد بررسی قرار خواهیم داد)، باید در ابتدای یک الگو (یعنی پایین عنصر xsd:schema) اعلان گردند. در نتیجه به صورت سراسری اعلان می‌شوند و برای استفاده در بقیه قسمتهای الگو قابل دسترسی خواهند بود. توجه داشته باشید که هرگاه عنصری به صورت سراسری اعلان شود محل ظهور آن در سند XML مهم نخواهد بود؛ بلکه ظاهر عنصر تعیین می‌گردد. شما باید یک اعلان عنصر سراسری را به منظور نمایش در یک سند XML به طور صریح نسبت‌دهی کنید.

تنها استثنا در این قانون برای عنصر ریشه است که هر جا که اعلان شود به صورت خودکار نسبت‌دهی می‌گردد.

هنگامی که یک نوع پیچیده را تعریف می‌کنید می‌توانید عناصر سراسری موجود را نسبت‌دهی کنید و عناصر جدیدی را اعلان و تعریف نمایید. این عناصر محلی جدید به تعریف نوع پیچیده‌ای‌که اعلان شده‌اند و شاید در الگو مورد استفاده قرار نگرفته باشند محدود می‌باشند. همچنین باید نام منحصر به فردی در سراسر سندی که در آن ظاهر می‌شوند داشته باشند. چنین عناصر محلی به صورت خودکار به محلی که تعریف می‌شوند نسبت داده می‌شوند و محلی را که باید عنصر در سند ظاهر گردد تعیین می‌نمایند.

code.xsd

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org
    /2000/10/XMLSchema">
<xsd:element name="endangered_species"
    type="endType"/>
<xsd:element name="name" type="xsd:string"/>
<xsd:complexType name="endType">
    <xsd:sequence>
    <xsd:element name="animal">
      <xsd:complexType>
          <xsd:sequence>
          <xsd:element ref="name"
          minOccurs="2"/>
          <xsd:element name="source"
              type="sourceType"/>
    ...
    </xsd:complexType>
<xsd:complexType name="habitatType">
    <xsd:sequence>
    <xsd:element name="river">
      <xsd:complexType>
          <xsd:sequence>
          <xsd:element ref="name"
          minOccurs="1"
          maxOccurs="unbounded"/>
          <xsd:element name="source"
          type="xsd:string"/>
    ...
    </xsd:complexType>
...
```

شکل ٤-٥. در این مثال چهار جزء به صورت سراسری تعریف شده‌اند؛ زیرا پس از عنصر xsd:schema قرار گرفته‌اند. عنصر ریشه (endangered_species) به صورت خودکار نسبت‌دهی شده اما همان‌گونه که می‌بینید عنصر name (که عنصر ریشه نیست) به شکل دستی نسبت‌دهی شده است.

به اجزای های‌لایت شده توجه کنید. دو عنصر محلی با یک نام و دو تعریف متفاوت اعلان شده‌اند. از طرفی اعلانهای عنصرهای سراسری باید نامهای منحصر به فرد داشته باشند.

آغاز یک الگوی ساده

یک الگو یک سند XML با فرمت متنی و پسوند xsd. است. این الگو با یک اعلان استاندارد XML به همراه اعلان فضای نام XML Schema آغاز می‌گردد.

برای آغاز یک الگو :

۱- درصورت تمایل در ابتدای سند الگو عبارت **<?xml version="1.0"?>** را تایپ کنید (صفحه ۲۴).

۲- عبارت **<xsd:schema** را تایپ نمایید.

۳- برای اعلان فضای نام الگوی schema عبارت **xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"** را تایپ کنید. از این پس هر عنصر یا نوعی که پیشوند:xsd داشته باشد به این فضای نام نسبت داده می‌شود.

۴- برای تکمیل عنصر الگو علامت < را تایپ نمایید.

۵- چند خط خالی برای ایجاد قوانین الگو ایجاد کنید. در فصلهای ۶ و ۷ با این قوانین آشنا خواهید شد.

۶- عبارت **</xsd:schema>** را به منظور کامل شدن سند الگو تایپ نمایید.

۷- الگو را با فرمت text only (فقط متنی) با پسوند xsd. ذخیره سازید.

نکته

- برای حفظ سادگی درس و یادگیری روش ایجاد الگو در این قسمت وارد جزئیات بیشتر نشده‌ایم. در صفحه‌های ۱۲۸-۱۲۶ با اعلان فضاهای نام اضافی و اعلان XML Schema به عنوان فضای نام پیش‌فرض آشنا خواهید شد.

```
code.xsd
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd=
"http://www.w3.org/2000/10/XMLSchema">


</xsd:schema>
```

شکل ۵-۵ . در عنصر ریشه الگو، برای الگوی Schema فضای نام اعلان شده است.

تعیین محل یک الگوی ساده

در برخی از معتبرسازهایی که برای مقایسه سندXML با الگو به کار می‌برید باید محل الگوی متناظر را در سند XML مشخص نمایید.

برای اعلان و تعیین محل یک الگو :

۱- در قسمت عنصر ریشه سند XML عبارت **xmlns:xsi="http//www.w3.org/2000/10/XMLSche ma-instance"** را تایپ کنید. در نتیجه عناصر، در محل تعیین شده الگو در دسترس خواهند بود. به این کار، *اعلان فضای‌نام* می‌گویند که در صفحه ۱۱۶ به شرح کامل آن خواهیم پرداخت.

۲- عبارت **xsi:noNamespaceSchemaLocation=** را تایپ کنید.

۳- در نهایت عبارت "file.xsd" را تایپ نمایید. file.xsd آدرس URL فایل الگویی است که در مرحله ۷ صفحه ۷۲ ایجاد کرده‌اید و می‌خواهید برای معتبرسازی این فایل XML مورد استفاده قرار دهید.

نکته‌ها

- برای تعیین اعتبار اسناد با یک الگو، برنامه‌های گوناگونی وجود دارند. از جمله برنامه XML SPY که یک ویرایشگر XML است و در آدرس http://www.xmlspy.com قابل دسترس می‌باشد. به صفحه ۲۴۵ نگاه کنید.
- هرگاه در الگو هیچ *فضای‌نام مقصدی* اعلان نشود ویژگی xsi:noNamespaceSchemaLocation در نظر گرفته می‌شود. درباره این نوع فضاهای نام در فصل ۹ صحبت خواهیم کرد.

```
code.xml
<?xml version="1.0" ?>
<endangered_species
xmlns:xsi="http://www.w3.org/2000/10/
XMLSchema-instance"
xsi:noNamespaceSchemaLocation=
"http://www.cookwood.com/ns/end_species/
end_species.xsd">
```

شکل ۶-۵ می‌توانید به فایل الگویی که سند شما را با استفاده از ویژگیxsi:schemaLocation در عنصر ریشه سند XML تعریف می‌نماید اشاره کنید.

تفسیر الگوها

می‌توانید درباره الگوها و عناصر آنها اطلاعات بیشتری اضافه کنید تا استفاده از الگوها در آینده برای شما و دیگران آسان‌تر گردد.

برای تفسیر الگوها :

۱- عبارت <**xsd:annotation**> را تایپ کنید.

۲- عبارت <**xsd:documentation**> را برای آغاز ایجاد یک یادداشت متنی که توسط انسانها (نه ماشین) خوانده خواهد شد تایپ کنید.

۳- یادداشت مورد نظر خود را تایپ کنید.

۴- با تایپ <**/xsd:documentation**> یادداشت خود را تکمیل نمایید.

۵- با تایپ <**/xsd:annotation**> اعلان تفسیر را به پایان برسانید.

نکته

- یادداشتها را می‌توان بلافاصله پس از عنصر سراسری xsd:schema یا اعلانهای جداگانه عناصر یا هردوی آنها ایجاد نمود.

code.xsd

```
<xsd:schema xmlns:xsd="http://www.w3.org/
2000/10/XMLSchema">
<xsd:annotation>
<xsd:documentation>
This schema will be used to validate the set of XML
documents for the Endangered Species
project.</xsd:documentation>
</xsd:annotation>
```

شکل ۷-۵. یک یادداشت تفسیری به درک بهتر الگو کمک می‌کند. درنتیجه به روزرسانی آن در آینده نیز آسان‌تر خواهد شد.

۶

تعریف نوعهای ساده

یک عنصر از *نوع ساده* فقط شامل متن است. این عنصر نمی‌تواند از عناصر دیگر و ویژگیها تشکیل گردد. به جای اینکه محتویات یک عنصر را به متن محدود کنید می‌توانید متن خاصی را برای آن در نظر بگیرید. چنین محدودیتهایی را می‌توانید با استفاده از تعریف نوع ساده درونی از پیش تعریف شده یا نوعهایی که خود شما تعریف کرده‌اید ایجاد نمایید.

XML Schema برای بیشتر متنهای رایج مجموعه‌ای از نوعهای ساده درونی و از پیش تعریف شده دارد که شامل مقادیر رشته‌ای، بولین، URLها، فرمتهای گوناگون تاریخ و انواع سیستمهای عددنویسی است. می‌توان با اعمال محدودیتها و شرطهایی برای یک نوع ساده درونی، نوع ساده دلخواهی ایجاد کرد. برای مثال شاید بخواهید یک عنصر به صورت رشته‌ای و فرمت خاص (مثل شماره تلفن یا کد کالا) یا عنصری که فقط شامل مجموعه‌ای از فرمتهای معین تاریخ است داشته باشید.

اعلان یک عنصر با یک نوع ساده

نوعهای ساده درونی و از پیش تعریف شده گوناگونی وجود دارند. شما نیز می‌توانید یک نوع ساده بر اساس نوعهای درونی بسازید.

برای اعلان یک نوع ساده :

۱- عبارت **xsd:element**<رابرای آغاز اعلان تایپ کنید.

۲- **name="label"** را تایپ نمایید. منظور از label نام عنصری است که قصد اعلان آن را دارید.

۳- عبارت **type="** را تایپ کنید.

سپس اگر عنصر مورد نظر شامل رشته‌ای از کاراکترها است عبارت **xsd:string** را تایپ کنید.

چنانچه عنصر از اعداد اعشاری تشکیل می‌گردد عبارت **xsd:decimal** را تایپ نمایید. برای بقیه انواع عددی به صفحه ۷۸ مراجعه کنید.

اگر عنصر شامل مقادیر درست یا نادرست است (۰یا۱) عبارت **xsd:Boolean** را تایپ کنید.

درصورتی‌که عنصر یک تاریخ است عبارت **xsd:date** را تایپ کرده و برای فرمتهای دیگر تاریخ به صفحه ۷۸ مراجعه کنید.

برای عنصری که زمانی از روز را نشان می‌دهد عبارت **xsd:time** را تایپ نموده و فرمتهای گوناگون زمان را در صفحه ۷۸ بررسی نمایید.

اگر عنصر شامل آدرس URL است، **xsd:uri-reference** را تایپ کنید (uri صحیح است نه url).

در مواردی که عنصراز دو حرف اختصاری زبانهای موجود در فهرست ISO639 تشکیل می‌شود **xsd:language** را تایپ کنید.

چنانچه قصد ایجاد نوع ساده دلخواهی دارید واژه **custom** را تایپ کنید.

۴- برای تکمیل نوع، علامت " را تایپ نمایید.

۵- به‌منظورتکمیل برچسب، علامتهای/> را تایپ کنید.

```
code.xsd
<xsd:element name="weight" type="xsd:string"/>
<xsd:element name="population"
type="xsd:integer"/>
```

شکل ۱-۶. هنگام اعلان یک عنصر، می‌توانید نام و نوع محتویات آن را انتخاب نمایید.

```
code.xml
<weight>500 pounds</weight>
<population>28</population>
```

شکل ۲-۶. منظور از string یک سری حرف،عدد یا نماد است. چون integer شامل رشته‌های عددی می‌گردد عناصر weight و population هنگام مقایسه با اعلان شکل ۱-۶ معتبر می‌باشند.

```
code.xml
<weight>500</weight>
<population>not very many</population>
```

شکل ۳-۶. در این مثال عنصر weight که در نگاه اول نادرست به نظر می‌رسد، معتبر است. زیرا به صورت رشته‌ای از اعداد ظاهر گشته است. ولی عنصر population معتبر نیست زیرا عبارت "not very many" عدد صحیح یا integer نیست.

code.xsd
```
<xsd:element name="last_modified"
type="xsd:date"/>
```

شکل ٤-۶. در این مثال عنصر last_modified شامل تاریخ است.

code.xml
```
<last_modified>1999-05-16</last_modified>
```

شکل ۵-۶. برای این‌که این عنصر معتبر باشد باید فرمت تاریخ آن به صورت CCYY-MM-DD باشد. یعنی چهار رقم برای سال، خط تیره، دو رقم برای ماه، خط تیره، دو رقم برای روز. فرمتهای دیگری نیز برای تاریخ وجود دارند که در صفحه ۷۸ به آنها اشاره شده است.

code.xml
```
<last_modified>May 16, 1999</last_modified>
```

شکل ۶-۶. عنصر last_modified معتبر نیست؛ زیرا فرمت تاریخ مناسبی ندارد.

نکته‌ها

- نوعهای ساده درونی گوناگونی وجود دارند که از طریق آدرس http://www.w3.org/TR/xmlschema-2/#built-in-datatypes در دسترس قرار می‌گیرند.
- نوعهای ساده دلخواه را می‌توان بر اساس نوعهای ساده درونی ایجاد نمود. به صفحه۸۱ نگاه کنید.
- نوعهای ساده درونی اغلب با xsd: شروع می‌شوند (مورد استثنا را در صفحه ۱۲۸ بیابید)؛ در نتیجه از نوعهای ساده‌ای که توسط کاربر ایجاد شده قابل تشخیص می‌باشند (صفحه ۸۱).
- ویژگیها اغلب به عنوان نوعهای ساده محسوب می‌گردند (زیرا نمی‌توانند شامل محتویات و ویژگیها باشند) و شبیه عناصر نوع ساده اعلان می‌گردند. اما باید فقط در یک عنصر اعلان شوند (صفحه ۱۰۸).
- اگر عنصری شامل ویژگیها و عناصر دیگر باشد یک نوع پیچیده محسوب می‌گردد (فصل ۷).
- نام یک عنصر معتبر باید با یک حرف یا علامت _ آغاز گردد و با اعداد، حروف،علامت نقطه، - و ـ ادامه یابد (صفحه ۲۶).
- عناصر نوع ساده می‌توانند به صورت سراسری نیز اعلان گردند (یعنی در ابتدای سند و پایین عنصر xsd:schema) در حالی‌که می‌توانند توسط یک نوع پیچیده فراخوانی یا نسبت‌دهی شوند. همچنین می‌توانند در تعریف یک نوع پیچیده به شکل محلی اعلان شوند که به صورت خودکار نسبت‌دهی می‌شوند (فصل ۷).

استفاده از نوعهای تاریخ و زمان

XML Schema برای تاریخ، چند نوع داده درونی دارد. یکی از فرمتهای مناسب مرحله ۳ صفحه ۷۶ یا فرمتهایی که در این قسمت گفته شده رابه‌کاربرید (صفحه ۸۱).

برای استفاده از نوعهای تاریخ و زمان :

۱- اگر می‌خواهید مقدار مشخصی از زمان، مانند ۲ ساعت و ۱۲دقیقه و ۴۵/۳ نشان را دهید عبارت **xsd:timeDuration** را تایپ کنید.

فرمت time duration (طول زمان) باید به صورت **PnYnMnDTnHnMnS** باشد.P یعنیPeriod و اجباری است. T یعنی Time section و اختیاری است. n عدد صحیح مثبت و نشانه تعداد سال(year)، ماه(month)، روز(day)، ساعت(hour)، دقیقه(minute) و ثانیه(second) است. با اضافه کردن یک خط تیره اختیاری می‌توان طول زمان را در گذشته نشان داد.

۲- اگر می‌خواهید زمان معینی از روز را که هر روز اتفاق می‌افتد نشان دهید عبارت **xsd:time** را تایپ نمایید. برای مثال ۴ و ۱۵ دقیقه بعدازظهر هر روز. فرمت کار به شکل**hh:mm:ss.sss** است. برای نشان دادن اختلاف وقت محلی با وقت مبدأ جهانی یا UTC از **Z** با فرمت **hh:mm+** یا **hh:mm** – به‌صورت اختیاری استفاده می‌شود.

۳- اگر می‌خواهید زمان خاصی را در تاریخ معینی مشخص نمایید (مانند ۴ و ۱۵ دقیقه بعدازظهر روز ششم ماه مه سال ۱۹۳۵) عبارت **xsd:timeInstant** را انتخاب کنید. فرمت کار با مشخصه اختیاری وقت محلی به‌صورت **CCYY-MM-DDThh:mm:ss.sss** است.

۴- برای تعیین روز مشخص مثل ششم ماه مه سال ۱۹۳۵ عبارت **xsd:date** با فرمت **CCYY-MM-DD** به کار می‌رود.

xsd:month برای تعیین ماه مشخص (مانند ماه مه سال ۱۹۳۵) با فرمت **CCYY-MM** به کار می‌رود.

۶- برای تعیین سال معین (مثل سال ۱۹۳۵) از **xsd:year** با فرمت **CCYY** استفاده می‌شود.

code.xsd
```
<xsd:element name="gestation"
type="xsd:timeDuration"/>
```

code.xml
```
<gestation>P3M15D</gestation>
```

شکل ۶-۷ . دوران بارداری ببر حدود سه ماه ونیم است. توجه کنید که نیازی به تعیین تمام واحدها نیست. فقط همیشه به مقدار اولیه P نیاز داریم. اگر محتویات period شامل داده زمانی باشد باید T را هم اضافه کنید.

code.xsd
```
<xsd:element name="bedtime" type="xsd:time"/>
```

code.xml
```
<bedtime>20:15-05-05:00</bedtime>
```

شکل ۶-۸ .به فرمت وقت جهانی توجه کنید. محتویات عنصر bedtime ساعت ۸ و ۱۵ دقیقه بعدازظهر را به صورت EST نشان می‌دهد.

code.xsd
```
<xsd:element name="birth"
type="xsd:timeInstant"/>
```

code.xml
```
<birth>1999-03-14T18:27:46.2398Z</birth>
```

شکل ۶-۹ . محتویات این عنصر تولد یک ببر را در ساعت ۶ و ۲۷ دقیقه و ۴۶/۲۳۹۸ ثانیه بعدازظهر روز ۱۴ ماه مارس سال ۱۹۹۹ به وقت جهانی نشان می‌دهد.

code.xsd
```
<xsd:element name="birthdate"
type="xsd:date"/>
```

code.xml
```
<birthdate>1999-03-14</birthdate>
```

شکل ۶-۱۰ . عنصر نوع xsd:date شبیه xsd:timeInstant بدون داده زمانی است.

```
code.xsd
<xsd:element name="campaign_start"
type="xsd:month"/>

code.xml
<campaign_start>1999-03</campaign_start>
```

شکل ۱۱-۶ . اگر روز مشخصی مد نظر شما نیست می‌توانید در عنصر نوع xsd:month فقط سال و ماه را تعیین نمایید.

```
code.xsd
<xsd:element name="last_seen"
type="xsd:year"/>

code.xml
<last_seen>1950</last_seen>
```

شکل ۱۲-۶ . آخرین ببر کاسپین در سال ۱۹۹۵ دیده شده است. روز و ماه آن مشخص نیست. در این‌گونه مواقع فقط به نمایش سال عنصر نیاز داریم.

```
code.xsd
<xsd:element name="greatest_loss"
type="xsd:century"/>

code.xml
<greatest_loss>19</greatest_loss>
```

شکل ۱۳-۶ . توجه داشته باشید که عدد ۱۹ به قرن بیستم اشاره می‌کند. یعنی از سال ۱۹۰۰ تا ۱۹۹۹ . در این مدت ٪۹۶ ببرهای جهان از بین رفته‌اند.

```
code.xsd
<xsd:element name="birthday"
type="xsd:recurringDate"/>

code.xml
<birthday>--03-14</birthday>
```

شکل ۱۴-۶ . اگر بیان سال اهمیتی نداشته باشد و فقط به ماه و روز نیاز باشد می‌توان از xsd:recurringDate استفاده کرد.

```
code.xsd
<xsd:element name="payday"
type="xsd:recurringDay"/>

code.xml
<payday>---30</payday>
```

شکل ۱۵-۶ . برای رویدادی که در روز مشخصی از هر ماه اتفاق می‌افتد از xsd:recurringDay استفاده می‌شود.

۷- اگر می‌خواهید قرن خاصی را مشخص نمایید (مانند سده ۱۹۰۰) باید **xsd:century** را با فرمت **CC** به کار برید (قرن بیستم با عدد ۱۹ مشخص می‌گردد).

۸- به منظور نمایش روز خاصی از ماه معینی بدون بیان سال (مانند اول ماه ژانویه) باید از **xsd:recurringDate** با فرمت **--MM-DD** استفاده کنید. دو خط تیره اول به جای عدد سال قرار می‌گیرند و خط تیره بعدی برای جدا کردن ماه و روز به کار می‌رود.

۹- برای تعیین روز معینی از ماه (مانند روز ششم هر ماه) **xsd:recurringDay** با فرمت **---DD** به کار می‌رود. دو خط تیره اول به جای حذف مقادیر سال و ماه و خط تیره سوم برای جداکردن گذاشته شده است.

نکته

- منظور ازUTC(Coordinated Universal Time) وقت مبدأ جهانی است که توسط انجمن بین‌المللی وقت (International Time Bureau) تنظیم می‌گردد و همان زمان گرینویچ است.EST یا Eastern Standard Time در ایالات متحده به صورت UTC-5 بیان می‌گردد.PST یا Pacific Standard Time نیز UTC-8 است.UTC+1 مربوط به نقاط غربی اروپا و UTC+3 روسیه،UTC+6 برای ژاپن و UTC+10 برای استرالیا به‌کار می‌رود. جدول زمانهای جهانی در سایت وب مربوط (صفحه ۱۸) قرار دارد.

نوعهای عددی

XML Schema چند نوع عددی داخلی دارد (شماره ۳ صفحه ۷۶) که برای محدود کردن محتویات عناصر و ویژگیها به کار می‌روند. از این نوعها برای ایجاد نوعهای عددی دلخواه نیز می‌توان استفاده نمود (صفحه ۸۱).

برای استفاده ازنوعهای عددی:

۱- برای محتویات عددی مثبت و منفی که ارقام اعشاری محدودی دارند (مثل ۴/۲۶، ۱۰۰-، یا ۰) نوع **xsd:decimal** را انتخاب کنید.

۲- از **xsd:integer** برای اعداد مثبت یا منفی بدون اعشار (مانند ۵۴۲ یا ۷-) استفاده نمایید.

۳- **xsd:positiveInteger** برای {۱، ۲ ، ...}، **xsd:negativeInteger** برای {۱- ،۲- ، ...}، **xsd:nonPositiveInteger** برای {۰، ۱- ، ۲- ، ...} و **xsd:nonNegativeInteger** برای {۰ ، ۱، ۲ ، ...} به کار می‌رود.

۴- اعداد اعشاری ۳۲ بیتی مثل 43E2 ، صفر، مثبت و منفی، منفی و مثبت بینهایت (INF , -INF) و فقدان عدد (not a number) یا NaN از نوع **xsd:float** استفاده می‌کنند.

۵- محتویاتی که شامل اعداد اعشاری ۶۴ بیتی هستند **xsd:double** را به کار می‌برند.

نکته

- در آدرس http://www.w3.org/TR/xmlschema-2 نوعهای عددی دیگری با جزئیات بیشتر قرار دارند.

```
code.xsd
<xsd:element name="population"
type="xsd:nonNegativeInteger"/>
<xsd:element name="density"
type="xsd:decimal"/>
```

شکل ۱۶-۶ . برای population (جمعیت) به اعداد منفی نیاز نداریم. ولی چون xsd:positiveInteger شامل عدد صفر است نمی‌توانیم آن را به کار بریم. عنصر density که به تعداد حیوانات در کیلومتر مربع اشاره دارد با اعداد اعشاری بیان می‌گردد.

```
code.xml
<population>342</population>
<density>4.2</density>
```

شکل ۱۷-۶ . چون population می‌تواند مقدار صفر یا بیشتر داشته باشد معتبر است. عنصر density (پراکندگی) نیز می‌تواند عدد صحیح یا اعشاری باشد.

```
code.xml
<population>112.5</population>
<density>7</population>
```

شکل ۱۸-۶ . عنصر population معتبر نیست چون عدد صحیح است و نمی‌تواند مقدار اعشار داشته باشد.

نوعهای ساده دلخواه

زبان XML Schema نوعهای ساده درونی از پیش تعریف شده زیادی دارد که می‌توانند بر حسب نیاز شما تغییر کنند.

برای ایجاد نوع ساده دلخواه :

۱- برای آغاز تعریف نوع ساده عبارت **<xsd:simpleType** را تایپ کنید.

۲- عبارت **name="label"** را تایپ نمایید. label نوع ساده دلخواه شماست و عنصر نیست؛ زیرا یک نوع می‌تواند برای یک یا چند عنصر به کار گرفته شود.

۳- عبارت **<xsd:restriction base="foundation">** را تایپ کنید. منظور از foundation نوع ساده‌ای است که نوع دلخواه بر اساس آن تعریف می‌شود.

۴- برای تعریف نوع دلخواه جدید محدودیتهای لازم را اعمال کنید (صفحه ۸۴ تا ۹۰).

۵- عبارت **<xsd:restriction>** را تایپ کنید.

۶- برای تکمیل تعریف نوع ساده دلخواه عبارت **</xsd:simpleType>** را تایپ نمایید.

نکته‌ها

- پس از تعریف نام نوع ساده دلخواه می‌توان به صورتی که در پاراگراف پس از مرحله ۳ صفحه ۷۶ گفته شد، از آن استفاده نمود. توجه داشته باشید که به آن به صورت xsd:label اشاره نمی‌کنید. بلکه مانند label در مرحله ۲ با آن برخورد می‌شود.
- در اعلان یک عنصر منفرد می‌توان نوع ساده بدون نام نیز ایجاد کرد.
- فهرستی ازنوعهای‌ساده‌نیزمی‌توان ایجاد نمود(صفحه۹۰).

code.xsd

```
<xsd:simpleType name="zipcodeType">
<xsd:restriction base="xsd:string">
<xsd:pattern value="\d{5}(-\d{4})?"/>
</xsd:restriction>
</xsd:simpleType>
```

شکل ۱۹-۶ . در اینجا یک نوع جدید با نام zipcodeType بر اساس نوع xsd:string ایجاد شده و الگویی دارد که محتویات عناصر این نوع را به پنج رقم و خط تیره اختیاری و ۴ رقم دیگر محدود می‌نماید.

code.xsd

```
<xsd:element name="zipcode"
type="zipcodeType"/ minOccurs="1">
```

شکل ۲۰-۶ . از نوع جدید zipcodeType در اعلانهای عناصر نیز می‌توانید استفاده نمایید (صفحه ۷۶). ویژگی minOccurs در صفحه ۱۰۱ توضیح داده شده است. از این ویژگی برای تعیین تعداد دفعات ظهور یک عنصر استفاده می‌شود.

code.xml

```
<zipcode>45632</zipcode>
```

code.xml

```
<zipcode>42398-0987</zipcode>
```

شکل ۲۱-۶ . هر دو عنصر zipcode معتبر هستند.

code.xml

```
<zipcode>4398-12349</zipcode>
```

code.xml

```
<zipcode>781001</zipcode>
```

شکل ۲۲-۶ . هر دو عنصر zipcode غیر معتبر هستند. در مثال اول شامل ۴ رقم با یک خط تیره و پنج رقم است. در مثال دوم نیز الگویی ندارد و مانند کدپستی محل مورد نظر نیست. اغلب کد پستیهای خارج از ایالات متحده به فرمت و الگوی خاصی نیاز ندارند.

استفاده از نوعهای دلخواه بی‌نام

برای تمام نوعهایی که به دلخواه ایجاد می‌کنید نیازی نیست نامی در نظر بگیرید. اگر نوعی را یک بار برای عنصر مشخصی به کار می‌برید می‌توانید تداخل ارجاع بین عنصر و نوع را حذف نمایید.

برای تعریف و استفاده از نوع دلخواه بی‌نام :

۱- با تایپ عبارت **<xsd:element name="label">** اعلان عنصر را آغاز نمایید. منظور از label نام عنصری است که قصد اعلان آن را دارید. این نام در سند XML شبیه عبارت <label> ظاهر خواهد شد.

۲- عبارت **<xsd:simpleType>** را تایپ کنید.

۳- **<xsd:restriction base="foundation">** را تایپ کنید. Foundation نوع ساده‌ای است که نوع دلخواه خود را بر اساس آن می‌سازید.

۴- برای نوع جدید محدودیتهای لازم را تعیین نمایید (صفحه‌های ۸۴ تا ۹۰).

۵- **</xsd:restriction>** را تایپ کنید.

۶- برای تکمیل تعریف نوع ساده جدید عبارت **</xsd:simpleType>** را تایپ کنید.

۷- به منظور کامل شدن اعلان عنصر نوع ساده بی‌نام عبارت **</xsd:element>** را تایپ کنید.

نکته‌ها

- کافی است نوعهای درونی را به ویژگی type نسبت دهید.
- تنها تفاوت بین نوع بی‌نام و نامگذاری شده این است که نوع نامگذاری شده می‌تواند بیش از یک بار مورد استفاده قرار گیرد (با تنظیم ویژگی type و نام). در حالی که نوع بی‌نام تنها برای عنصری که شامل آن باشد قابل استفاده است.

```
code.xsd
<xsd:element name="zipcode">
    <xsd:simpleType>
    <xsd:restriction base="xsd:string">
    <xsd:pattern value="\d{5}(-\d{4})?"/>
    </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
```

شکل ۲۳-۶ . این اعلان عنصر را با تعریف نوع دلخواه پیچیده و اعلان عنصر شکل ۱۹-۶ و ۲۰-۶ مقایسه کنید. تعریف عنصر zipcode در هر دو مثال به کار رفته است. تفاوت مهم این است که نوع دلخواه این مثال برای عناصر دیگر قابل استفاده نیست. عنصر xsd:simpleType نه نامی دارد و نه چیزی به آن اشاره می‌کند.

تعیین مجموعه‌ای از مقادیر قابل قبول

برای حفظ پیوستگی و منطق سند XML می‌توان محتویات یک عنصر یا ویژگی را به مجموعه‌ای از مقادیر، محدود نمود.

برای تعیین مجموعه مقادیر قابل قبول :

۱- در قسمت اعلان نوع دلخواه (شماره ۴ صفحه ۸۱) عبارت **xsd:enumeration>** را تایپ کنید.

۲- **value="choice"** را تایپ نمایید. choice مقدار قابل قبولی برای محتویات عنصر یا ویژگی است.

۳- با تایپ /> عنصر xsd:enumeration کامل می‌گردد.

۴- برای هر مقدار اضافه‌ای که عنصر یا ویژگی می‌تواند داشته باشد مراحل ۱ تا ۳ را تکرار کنید.

نکته‌ها

- برای همه نوعهای ساده به غیر از بولین می‌توانید xsd:enumeration را به کار برید.
- مقادیر تعیین شده باید منحصر به فرد باشند.
- مجموعه مقادیر می‌تواند از فضای خالی نیز تشکیل شود.

code.xsd

```
<xsd:element name="continent">
<xsd:simpleType>
   <xsd:restriction base="xsd:string">
   <xsd:enumeration value="Asia"/>
   <xsd:enumeration value="Africa"/>
   <xsd:enumeration value="Australia"/>
   <xsd:enumeration value="Europe"/>
   <xsd:enumeration value="North America"/>
   <xsd:enumeration value="South America"/>
   <xsd:enumeration value="Antartica"/>
   </xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

شکل ۲۴-۶ . عنصر continent می‌تواند شامل هریک از مقادیر باشد.

code.xml

```
<continent>Asia</continent>
```

شکل ۲۵-۶ . عنصر continent معتبر است؛ زیرا با یکی از مقادیر مطابقت دارد.

code.xml

```
<continent>Asia Europe</continent>
```

code.xml

```
<continent>America</continent>
```

شکل ۲۶-۶ . هیچ یک ازعناصر continent معتبر نیستند. اولی شامل دو مقدار است. در حالی که فقط می‌تواند یک مقدار داشته باشد. دومی شامل بخشی از یک مقدار است. در فهرست مقادیر، آمریکا از دو بخش شمالی و جنوبی تشکیل شده در حالی که در این مثال فقط به آمریکا اشاره شده است.

تعیین طرحی برای یک نوع ساده

برای اینکه طرحی ایجاد شود تا با مقایسه محتویات با آن طرح، اعتبار محتویات تعیین گردد، می‌توان از زبان ویژه regex (regular expression) استفاده نمود. اساس زبان regex در XML Schema زبان عبارتهای معمولی Perl است. برای توضیح این زبان می‌توان در یک فصل جداگانه صحبت کرد؛ ولی در اینجا اشاره‌ای به آن می‌نماییم.

به منظور تعیین طرحی برای یک نوع ساده :

۱- در قسمت اعلان نوع دلخواه (شماره ۴ صفحه ۸۱) عبارت **xsd:pattern>** را تایپ کنید.

۲- سپس **value="regex"** را تایپ کنید. regex یک عبارت معمولی است که باید با ظاهر محتویات، مطابقت داشته باشد و به صورت زیر ایجاد می‌گردد:

حروف، اعداد و نمادهایی را که در محتویات ظاهر خواهند شد، تعیین کنید.

(.) علامت نقطه به جای هر کاراکتری قرار می‌گیرد.

\d برای ارقام و **\D** برای non-digit استفاده می‌شود.

\s به جای فضاهای خالی (که با کلیدهای Enter ، Tab و Spacebar ایجاد می‌شوند) و **\S** برای کاراکترهای غیر خالی به کار می‌رود.

X* به جای هیچ یا چند x می‌نشیند؛ **(XY)*** نشانه هیچ یا چند xy است.

X? به جای یک یا هیچ x و **(XY)?** به جای یک یا هیچ xy به کار می‌رود.

X+ معادل یک یا چند x و **(XY)+** نیز معادل یک یا چند xy می‌باشد.

[abc] شامل یک یا گروهی از مقادیر a، b یا c می‌گردد.

[0-9] مقادیر 0 تا 9 را نشان می‌دهد.

```
code.xsd
<xsd:element name="invoice_number">
<xsd:simpleType>
   <xsd:restriction base="xsd:string">
   <xsd:pattern value="INV #99\d{3}"/>
   </xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

شکل ۶-۲۷ . این طرح محتویات عنصر invoice_number را به رشته‌ای که با INV #99 شروع می‌شود و سه رقم در ادامه آن دارد محدود می‌نماید. هر کاراکتر که در regex ظاهر می‌شود باید در همان موقعیتی از محتویات‌که‌برای عنصر معتبر است ظاهر شود.

```
code.xml
<invoice_number>INV #99426</invoice_number>
```

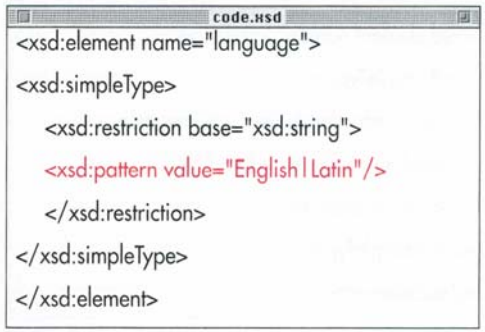
شکل ۶-۲۸ . در اینجا مثال معتبر الگوی شکل ۶-۲۷ را می‌بینید: کاراکترهای INV #99 به همراه سه رقم.

```
code.xsd
<xsd:element name="gestation">
<xsd:simpleType>
   <xsd:restriction base="xsd:timeDuration">
   <xsd:pattern value="P\d+D"/>
   </xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

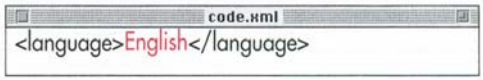
شکل ۶-۲۹ . طرحها را برای کنترل محتویات عناصری که بر اساس نوعهای دیگر ایجاد شده‌اند نیز می‌توانید به کار برید. برای مثال اگر می‌خواهید عنصر gestation (بارداری) شامل تعداد روزهای دوره بارداری ببر باشد باید طرحی را که در این مثال می‌بینید پیاده سازید. توجه داشته باشید این طرح با نوع کاری ندارد و شامل حرف بزرگ P به همراه یک یا چند رقم می‌شود که با حرف D پایان می‌یابد.

```
code.xml
<gestation>P108D</gestation>
```

شکل ۶-۳۰ . مثالی برای الگوی شکل ۶-۲۹ .



شکل ۳۱-۶ . می‌توان از طرحها برای ارائه گزینه‌های گوناگون محتویات یک عنصر استفاده کرد. راه استاندارد برای ارائه گزینه‌های گوناگون در صفحه ۸۳ گفته شده است.



شکل ۳۲-۶ . مثالی برای الگوی شکل ۳۱-۶ .

برای اینکه در محتویات، واژه this یا that وجود داشته باشد **this | that** به کار می‌رود. واژه‌های اختیاری دیگری را نیز به کمک خطوط عمودی دیگر می‌توانید اضافه نمایید.

x{5} یعنی پنج x در یک ردیف.

x{5,} یعنی حداقل پنج x در یک ردیف.

x{5,8} معرف حداقل پنج و حداکثر هشت x در یک ردیف است.

(xyz){2} به معنی دو xyz در یک ردیف می‌باشد.

۳- برای تکمیل عنصر xsd:pattern علامتهای /> را تایپ کنید.

نکته‌ها

- یکی از تفاوتهای مهم بین عبارتهای معمولی XML Schema و عبارتهای معمولی Perl این است که همیشه عبارت معمولی و محتویات عنصر با یکدیگر تطابق دارند و مانند Perl برای ابتدا یا پایان یک خط از کاراکترهای ^ یا $ استفاده نمی‌شود.
- برای کسب اطلاعات بیشتر درباره Perl به آدرس http://www.perl.com/pub/doc/manual/html/pod/perlre.html یا کتاب Perl و CGI از همین مؤلف مراجعه کنید.

تعیین مقادیر قابل قبول

برای محدود کردن محتویات یک عنصر یا ویژگی می‌توان بیشترین و کمترین مقدار آن را مشخص نمود.

برای تعیین بیشترین مقدار ممکن :

۱- در قسمت اعلان نوع دلخواه (شماره ۴ صفحه ۸۱) عبارت **xsd:maxInclusive>** را تایپ کنید. به حرف I که به صورت حرف بزرگ در عبارت Inclusive (جامع) قرار گرفته توجه نمایید.

۲- عبارت **value="n"** را تایپ کنید. برای اینکه محتویات معتبر باشند باید کوچکتر یا برابر n باشند.

۳- برای تکمیل عنصر xsd:maxInclusive علامتهای /> را تایپ کنید.

روش دیگر برای تعیین بیشترین مقدار ممکن:

۱- در قسمت اعلان نوع دلخواه (شماره ۴ صفحه ۸۱) عبارت **xsd:maxExclusive>** را تایپ کنید. به حرف بزرگ E در عبارت Exclusive (انحصاری) توجه نمایید.

۲- عبارت **value="n"** را تایپ کنید. محتویات باید کوچکتر n باشد تا معتبر محسوب گردد.

۳- برای تکمیل عنصر xsd:maxExclusive علامتهای /> را تایپ نمایید.

code.xsd

```
<xsd:element name="population">
<xsd:simpleType>
   <xsd:restriction base="xsd:integer">
   <xsd:maxInclusive value="5000"/>
   </xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

شکل ۳۳-۶ . طرح xsd:maxExclusive بیشترین مقدار ممکن برای یک عنصر را تعیین می‌کند.

code.xml

```
<population>5000</population>
```

code.xml

```
<population>4999</population>
```

شکل ۳٤-۶ . عناصر population معتبر هستند؛ زیرا در مثال اول برابر و در مثال دوم کوچک‌تر از xsd:maxInclusive است.

code.xsd

```
<xsd:element name="population">
<xsd:simpleType>
   <xsd:restriction base="xsd:integer">
   <xsd:maxExclusive value="5000"/>
   </xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

شکل ۳۵-۶ . در طرح xsd:maxInclusive محتویات باید کمتر از مقدار xsd:maxExclusive باشد.

code.xml

```
<population>5000</population>
```

code.xml

```
<population>4999</population>
```

شکل۳۶-۶ . عنصر population اول غیر معتبر ولی دومی معتبر است.

```
code.xsd
<xsd:element name="start_date">
<xsd:simpleType>
  <xsd:restriction base="xsd:date">
  <xsd:minInclusive value="1999-07-25"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

شکل ۳۷-۶ . طرح xsd:minInclusive کمترین مقدار ممکن را برای یک عنصر تعیین می‌کند. یعنی عنصر start_date باید بیست‌وپنجم جولای ۱۹۹۹ به‌بعد باشد.

```
code.xml
<start_date>1999-07-25</start_date>
```

```
code.xml
<start_date>1999-07-26</start_date>
```

شکل ۳۸-۶ . عناصر start_date معتبر هستند؛ زیرا اولی برابر و دومی بزرگتر از مقدار xsd:minInclusive است.

```
code.xsd
<xsd:element name="start_date">
<xsd:simpleType>
  <xsd:restriction base="xsd:date">
  <xsd:minExclusive value="1999-07-25"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

شکل ۳۹-۶ . در طرح xsd:minInclusive مقدار عنصر باید بیشتر از مقدار xsd:minInclusive باشد.

```
code.xml
<start_date>1999-07-25</start_date>
```

```
code.xml
<start_date>1999-07-26</start_date>
```

شکل ۴۰-۶ . عنصر start_date در مثال اول غیر معتبر است؛ زیرا نمی‌تواند با مقدار xsd:minInclusive برابر باشد. عنصر مثال دوم به دلیل اینکه بزرگتر از مقدار تعیین شده است معتبر می‌باشد.

برای تعیین کمترین مقدار ممکن :

۱- در قسمت اعلان نوع دلخواه (شماره ۴ صفحه ۸۱) عبارت **xsd:minInclusive>** را تایپ کنید. حرف I باید در Inclusive بزرگ باشد.

۲- عبارت **value="n"** را تایپ نمایید. محتویات باید کمتر یا برابر مقدار n باشند.

۳- برای تکمیل عنصر xsd:minInclusive علامتهای /> را تایپ نمایید.

روش دیگر برای تعیین کمترین مقدار ممکن:

۱- در قسمت اعلان نوع دلخواه (شماره ۴ صفحه ۸۱) عبارت **xsd:minInclusive>** را تایپ کنید. حرف I باید در Inclusive بزرگ باشد.

۲- عبارت **value="n"** را تایپ کنید. محتویات معتبر باید کوچکتر از n باشند.

۳- با تایپ علامتهای /> عنصر xsd:Exclusive را تکمیل کنید.

نکته‌ها

- برای یک نوع نمی‌توانید به صورت هم‌زمان دو مقدار حداقل یا دو مقدار حداکثر در نظر بگیرید. مقادیر حداقل و حداکثر را می‌توانید ترکیب کنید یا فقط یکی از مقادیر را به کار برید.
- مفاهیم کوچکتر از و بزرگتر از برای مقادیر، مشخص است. مقدار تاریخ یا زمان بزرگتر، برای آینده و مقدار تاریخ یا زمان کوچکتر برای گذشته استفاده می‌شود.

محدود کردن طول یک نوع ساده

یک راه برای تعریف عنصر رشته‌ای یا نوع ساده URL ،تعیین محدودیت طول آن است.

برای تعیین طول دقیق یک عنصر:

عبارت <xsd:lengthvalue="x"/> را در قسمت اعلان نوع دلخواه تایپ کنید. منظور از x تعداد کاراکترهایی است که عنصر می‌تواند داشته باشد.

برای تعیین حداقل طول یک عنصر:

عبارت <xsd:minLengthvalue="m"/> را در قسمت اعلان نوع دلخواه تایپ نمایید. منظور از m حداقل کاراکترهای یک عنصر است.

برای تعیین حداکثر طول یک عنصر:

عبارت <xsd:maxLengthvalue="n"/> را در قسمت اعلان نوع دلخواه تایپ کنید. n حداکثر کاراکترهای عنصر را مشخص می‌نماید.

نکته‌ها

- اگر طول را تعیین کنید مقدار حداقل یا حداکثر را نمی‌توانید معین نمایید و برعکس اگر مقدار حداقل یا حداکثر را تعیین کردید امکان تعیین طول وجود ندارد.
- مقدارهایی که برای xsd:length ، xsd:minLength و xsd:maxLength درنظر گرفته می‌شوند باید اعداد صحیح مثبت باشند.
- اگر عنصر بر اساس نوع دودویی تعریف شده باشد طول، اعداد داده‌های دودویی را مشخص می‌کند. اگر عنصر به صورت فهرست باشد(صفحه ۹۰) طول، تعداد اعضای فهرست را معین می‌نماید.

code.xsd

```
<xsd:element name="animal_code">
<xsd:simpleType>
   <xsd:restriction base="xsd:string">
   <xsd:length value="4"/>
   </xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

شکل ۴۱-۶ . طول یک رشته(string) را می‌توان تنظیم کرد.

code.xml

```
<animal_code>TIGR</animal_code>
```

شکل ۴۲-۶ . عنصرanimal_code به دلیل اینکه شامل چهار کاراکتر است (شکل ۴۱-۶)، معتبر می‌باشد.

code.xsd

```
<xsd:element name="description">
<xsd:simpleType>
   <xsd:restriction base="xsd:string">
   <xsd:maxLength value="200"/>
   </xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

شکل ۴۳-۶ . طول عنصر string را می‌توان محدود کرد.

code.xml

```
<description> The tiger (panthera tigris), largest of
all cats, is one of the biggest and most fearsome
predators in the world.</description>
```

شکل ۴۴-۶ . عنصر description معتبر است؛ زیرا می‌تواند از ۲۰۰ کاراکتر تشکیل شود در حالی‌که شامل ۱۱۳ کاراکتر می‌باشد.

code.xml

```
<description> The tiger (panthera tigris), largest of
all cats, is one of the biggest and most fearsome
predators in the world.Powerfully built with fierce
retractile claws (they can be pulled into the paw, like
a house cat's), the tiger's distinctive gold coloring
with black stripes allows it to melt unseen into its
environment. </description>
```

شکل ۴۵-۶ . به دلیل اینکه عنصر description از ۳۱۷ کاراکتر تشکیل شده است معتبر نیست.

محدود کردن رقمهای اعداد

تعداد رقمهای قسمت صحیح و اعشار اعداد را می‌توان محدود کرد.

برای تعیین تعداد رقمهای اعداد :

۱- در قسمت اعلان نوع دلخواه (شماره ۴ صفحه ۸۱) عبارت **xsd:precision>** را تایپ کنید.

۲- عبارت **value="n"** را تایپ نمایید. منظور از n حداکثر تعداد رقمهای یک عدد است.

۳- با تایپ علامتهای /> این مرحله را تکمیل کنید.

برای تعیین تعداد رقمهای سمت راست ممیز :

۱- در قسمت اعلان نوع دلخواه، عبارت **xsd:scale>** را تایپ نمایید.

۲- عبارت **value="n"** را تایپ نمایید. n حداکثر تعداد رقمهای سمت راست ممیز را مشخص می‌کند.

۳- علامتهای /> را تایپ نمایید تا این مرحله پایان پذیرد.

نکته‌ها

- طرح xsd:precision باید از اعداد صحیح مثبت یعنی {۱،۲،۳،...} تشکیل گردد. مقدار این طرح نمی‌تواند صفر یا کمتر از xsd:scale باشد.
- xsd:scale باید از مجموعه اعداد صحیح {۰،۱،۲،۳،...} تشکیل شود.
- xsd:precision و xsd:scale مقدار حداکثر را مشخص می‌نمایند یعنی تعداد ارقام کمتر نیز قابل قبول است.
- xsd:precision و xsd:scale برای هر نوع عددی معتبر می‌باشند؛ ولی برای نوعهای رشته‌ای، تاریخ و غیره قابل قبول نیستند.

code.xsd

```
<xsd:element name="numbers">
<xsd:simpleType>
   <xsd:restriction base="xsd:decimal">
   <xsd:precision value="5"/>
   <xsd:scale value="2"/>
   </xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

شکل ۴۶-۶. مقدار precision تعداد کل رقمهای عدد را تعیین می‌کند. scale تعداد رقمهای اعشار عدد را که در سمت راست ممیز قرار می‌گیرند مشخص می‌نماید.

code.xml

```
<numbers>564.59</numbers>
```

code.xml

```
<numbers>34.5</numbers>
```

شکل ۴۷-۶. عناصر هر دو مثال معتبرند؛ زیرا می‌توانند حداکثر پنج رقم و حداکثر دو رقم اعشار داشته باشند.

code.xml

```
<numbers>1476.32</numbers>
```

code.xml

```
<numbers>4.3987</numbers>
```

شکل ۴۸-۶. این دو عنصر عددی غیر معتبر هستند؛ زیرا مثال اول شش رقم دارد و مثال دوم از چهار رقم اعشار تشکیل شده است.

ایجاد نوعهای فهرستی

تا به حال درباره عناصری صحبت کردیم که از یک بخش تشکیل می‌شدند. اگر عنصری را به صورت تاریخ تعریف کنید فقط می‌تواند شامل یک تاریخ باشد. ولی اگر به عنصری نیاز داشته باشید که فهرستی از تاریخ داشته باشد می‌توانید نوع تاریخ خود را به صورت نوع فهرستی تعریف کنید.

برای ایجاد نوع فهرستی :

۱- عبارت **<xsd:simpleType>** را برای آغاز تعریف نوع فهرستی تایپ کنید.

۲- سپس **<name="label">** را تایپ کنید. label نام عنصری است که می‌خواهید تعریف کنید و در سند XML به صورت <label> ظاهر می‌گردد.

۳- عبارت **<xsd:list base="individual">** را تایپ کنید. منظور از individual نام نوع ساده‌ای است که مقدار هر یک از اعضای فهرست را تعریف می‌کند.

۴- فهرستها را می‌توان با طرحهای xsd:length، xsd:minLength و xsd:maxLength (صفحه ۸۸) و xsd:enumeration (صفحه ۸۳) محدود نمود.

۵- برای تکمیل تعریف فهرست عبارت**</xsd:list>** را تایپ کنید.

۶- برای تکمیل تعریف نوع ساده، عبارت **</xsd:simpleType>** را تایپ نمایید.

نکته‌ها

- اگر به ایجاد محدودیت در فهرست نیازی ندارید می‌توانید علامتهای /> را تایپ و عنصر xsd:list را تکمیل کنید (**شکل ۴۹-۶**).
- نوع فهرستی را می‌توان تنها براساس نوع ساده ایجاد کرد (به فصل بعد مراجعه کنید).
- فضای خالی، تعداد اعضای فهرست را مشخص می‌کند. سپس باید هنگام تعیین طول فهرست رشته‌ای بیشتر دقت کنید.

code.xsd

```
<xsd:simpleType name="datelist">
    <xsd:list base="xsd:date"/>
</xsd:simpleType>
<xsd:element name="list_of_birthdays"
type="datelist"/>
```

شکل ۴۹-۶ . یک فهرست باید بر اساس یک نوع ساده دلخواه یا درونی باشد. نوع فهرستی datelist بر اساس نوع xsd:date می‌باشد. در قسمت اعلان نوع ساده دلخواه، عنصر list_of_birthdays تعریف شده است.

code.xml

```
<list_of_birthdays>1893-04-20 1904-05-11
</list_of_birthdays>
```

شکل ۵۰-۶ . یک فهرست می‌تواند از هیچ یا چند مقدار که بر اساس ویژگی تعیین و با فضای خالی جدا شده است تشکیل گردد.

code.xsd

```
<xsd:simpleType name="three_datelist">
    <xsd:list base="xsd:date">
        <xsd:length value="3"/>
    </xsd:list>
</xsd:simpleType>
<xsd:element name="list_of_3_birthdays"
type="three_datelist"/>
```

شکل ۵۱-۶ . با طرحهای xsd:length، xsd:minLength و xsd:maxLength می‌توان محدودیت بیشتری برای نوع فهرستی ایجاد کرد.

code.xml

```
<list_of_3_birthdays>1893-04-20 1904-05-11
1852-06-25</list_of_3_birthdays>
```

شکل ۵۲-۶ . نوع فهرستی list_of_3_birthdays باید از سه تاریخ تشکیل گردد تا معتبر باشد.

تعریف محتویات یک عنصر

به منظور استفاده از یک الگو برای از پیش تعریف کردن محتویات یک عنصر دو روش وجود دارد.می‌توان محتویات و مقدار عنصری را که در XML به صورت عنصر خالی ظاهر می‌شود تعریف و تعیین نمود. به حالت اول مقدار ثابت و به حالت دوم مقدار پیش فرض می‌گویند.

برای تعیین محتویات یک عنصر :

۱- در برچسب عنصر عبارت **fixed=** را تایپ کنید.

۲- سپس **"value"** را تایپ کنید.value محتویات عنصر را تعیین می‌نماید و نباید خالی باشد.

برای تعیین مقدار اولیه یک عنصر :

۱- در برچسب عنصر عبارت **default=** را تایپ نمایید.

۲- سپس **"value"** را تایپ کنید. value محتویات عنصر را مشخص می‌کند که می‌تواند خالی باشد و یا حذف شود.

نکته‌ها

- ویژگی fixed محتویات عنصری را تعیین می‌کند که در XML ظاهر می‌شود. اگر حذف شود محتویاتی نیز برای آن در نظر گرفته نمی‌شود.
- اگر ویژگی fixed تنظیم شود و عنصر خالی باشد مقدار عنصر به صورت خودکار با مقدار ثابت تنظیم می‌شود.
- چنانچه ویژگی default تنظیم گردد ولی عنصر از XML حذف شود مقدار عنصر به صورت خودکار با مقدار پیش فرض تنظیم می‌گردد.
- در صورتی که ویژگی default در نظر گرفته شود و عنصر در XML ظاهر نگردد محتویات آن به ویژگی fixed بستگی خواهد داشت.
- ویژگی default و fixed را نمی‌توان به صورت هم‌زمان داشت.

code.xsd
```
<xsd:element name="status" type="xsd:string"
fixed="endangered"/>
```

شکل ۵۳-۶ . عنصر status در سند XML ظاهر شده و شامل رشته endangered می‌باشد. اگر خالی باشد رشته endangered برای آن در نظر گرفته خواهد شد.

code.xml
```
<status>vulnerable</status>
```

شکل ٥٤-۶ . عنصرstatus براساس الگوی شکل ۵۳-۶ معتبر است.

code.xml
```
<status>vulnerable</status>
```

شکل ۵۵-۶ . براساس الگوی شکل ۵۳-۶ عنصرstatus معتبر نیست.

code.xsd
```
<xsd:element name="status" type="xsd:string"
default="endangered"/>
```

شکل ۵۶-۶ . اگر عنصرstatus را با یک مقدار پیش فرض تعریف کنیم در صورت ظهور یا عدم نمایش عنصر status در سند XML برای آن، محتویات پیش فرض در نظر گرفته خواهد شد.

code.xml
```
<status>endangered</status>
```

code.xml
```
<status>vulnerable</status>
```

شکل ۵۷-۶ . هر دو عنصرstatus معتبرند. ویژگی پیش فرض تنها یک مقدار اولیه را تنظیم می‌کند و مقادیر دیگر نیز قابل قبول هستند.

۷

تعریف نوعهای پیچیده

به عنصری که می‌تواند از عناصر دیگر و ویژگیها تشکیل شود نوع پیچیده می‌گویند. از آنجا که بیشتر اسناد XML عناصری دارند که از عناصر دیگر تشکیل شده‌اند، ایجاد نوعهای پیچیده در الگو کار بعیدی به نظر نمی‌رسد. چهار عنصر نوع پیچیده داریم که عبارتند از:

عناصر «فقط عنصر» که از عناصر یا ویژگیها تشکیل شده‌اند و متنی ندارند (**شکل ۱-۷**). عناصر «خالی» عنصر و متنی ندارند ولی شاید از ویژگیهایی تشکیل شوند (**شکل ۲-۷**). عناصر «مختلط» که شامل ترکیبی از عناصر، ویژگیها و یا متن می‌باشند. بیشتر، ترکیب عناصر و متن به کار می‌رود (**شکل ۳-۷**). عناصر «متنی» که فقط شامل متن هستند و شاید ویژگیهایی نیز داشته باشند (**شکل ۴-۷**).

code.xml

```
<subspecies>
<name language="English">Amur</name>
<name language="Latin">P.t. altaica</name>
<region>Far East Russia</region>
<population year="1999">445</population>
</subspecies>
```

شکل ۱-۷. عنصر subspecies از عناصر دیگر تشکیل می‌شود ولی متنی ندارد. هرچند که این عنصر بدون ویژگی است، می‌تواند جزء عناصر «فقط عنصر» باشد.

code.xml

```
<source sectionid="101" newpaperid="21"/>
<picture filename="tiger.jpg" x="200" y="197"/>
```

شکل ۲-۷. عناصر source و picture عناصر «خالی» هستند زیرا محتویاتی ندارند. عناصر این مثال ویژگیهایی دارند که این امر برای تمام عناصر خالی الزامی نیست.

code.xml

```
<description length="short">The <name
language="English">tiger</name> (panthera
tigris), largest of all cats, is one of the biggest and
most fearsome predators in the
world.</description>
```

شکل ۳-۷. عنصر description شامل متن و عنصر name و ویژگی است. بنابراین به دلیل داشتن محتویات گوناگون عنصر «مختلط» محسوب می‌گردد.

code.xml

```
<name language="Latin">panthera tigris</name>
```

شکل ٤-۷. عنصر name فقط شامل متن و یک ویژگی است؛ بنابراین یک عنصر «فقط متنی» می‌باشد.

تعریف عناصر فقط عنصری

به عنصر نوع پیچیده‌ای که می‌تواند از عناصر و ویژگیها تشکیل شود ولی متنی نداشته باشد عنصر فقط عنصری می‌گویند.

برای ایجاد نوع پیچیده‌ای که فقط شامل عناصر است :

۱- عبارت **<xsd:complexType>** را تایپ کنید.

۲- سپس عبارت **name="label"** را تایپ نمایید. label مشخص کننده نوع پیچیده است نه نام عنصر.

۳- برای تکمیل برچسب علامت > را تایپ کنید.

۴- به منظور تعیین عناصر تشکیل دهنده نوع پیچیده، یک دنباله (صفحه ۹۵)، گزینه (صفحه ۹۶) یا گروه نامنظم (صفحه ۹۷) اعلان کنید یا به یک گروه نامگذاری شده (صفحه ۹۹) اشاره نمایید.

۵- سپس ویژگیها (صفحه ۱۰۷) یا گروه ویژگیها (صفحه ۱۱۱) را که در عناصر این نوع ظاهر خواهند شد اعلان کرده یا به آنها اشاره کنید.

۶- سپس **</xsd:complexType>** را تایپ کنید.

نکته‌ها

- پس از ایجاد نوع پیچیده باید عنصر یا عناصر تشکیل دهنده آن را اعلان کنید (صفحه ۱۰۶).
- عناصر تشکیل دهنده یک نوع پیچیده باید بخشی از یک دنباله، گزینه، گروه نامنظم یا گروه نامگذاری شده باشند.

code.xsd

```
<xsd:complexType name="endspeciesType">
  <xsd:sequence>
  <xsd:element name="animal"
  type="animalType"/>
  </xsd:sequence>
</xsd:complexType>
```

شکل ۵-۷ . در اینجا تعریف comlexType (که برای تعریف عنصر endangered_species به کار می‌رود) دیده می‌شود که از عنصر animal با نوع animalType تعریف شده تشکیل می‌گردد. نوعهای پیچیده می‌توانند شامل عناصر نوع پیچیده دیگر باشند.

```
code.xsd
<xsd:complexType name="animalType">
<xsd:sequence>
<xsd:element ref="name" minOccurs="2"/>
<xsd:element name="threats"
    type="threatsType"/>
<xsd:element name="weight" type="xsd:string"/>
<xsd:element name="length" type="xsd:string"/>
<xsd:element name="source"
    type="sourceType"/>
<xsd:element name="picture"
    type="pictureType"/>
<xsd:element name="subspecies"
    type="subspeciesType"/>
</xsd:sequence>
</xsd:complexType>
```

شکل ۶-۷ . هرعنصر تعریف شده با نوع animalType به ترتیب شامل عناصر name، threats، weight، length، source، picture و subspecies است.

عناصری که در یک دنباله ظاهر می‌گردند

اگر عنصر نوع پیچیده‌ای از عناصر دیگر تشکیل می‌شود باید برای عناصر آن دنباله‌ای تعریف کرد.

برای تعریف دنباله‌ای از عناصر :

۱- عبارت **xsd:sequence>** را تایپ کنید.

۲- اگر بخواهید می‌توانید تعداد دفعات ظاهر شدن دنباله عناصر را با تنظیم ویژگیهای minOccurs و maxOccurs تعیین کنید (صفحه ۱۰۱).

۳- برای تکمیل کار، علامت < را تایپ کنید.

۴- هریک از اجزای دنباله را اعلان کنید (صفحه ۷۶ و ۱۰۶) یا ارجاع دهید (صفحه ۱۰۰).

۵- عبارت <**xsd:sequence**> را تایپ کنید.

نکته‌ها

- یک دنباله که ترتیب ظاهر شدن عناصر تشکیل دهنده را در سند XML تعیین می‌نماید تعریف کنید.
- یک دنباله می‌تواند شامل دنباله‌ها، گزینه‌ها (صفحه ۹۶) یا مراجع گروههای نامگذاری شده (صفحه ۹۹) باشد.
- یک دنباله می‌تواند شامل تعریف نوع‌پیچیده (صفحه۹۴)، دنباله‌های دیگر یا مجموعه‌ای از گزینه‌ها (صفحه ۹۶) یا تعاریف گروههای نامگذاری شده (صفحه ۹۸) باشد.
- عنصر**xsd:sequence** معادل علامت کاما (,) در DTDها است.
- یک دنباله می‌تواند فقط شامل یک عنصر باشد.

ایجاد مجموعه‌ای از گزینه‌ها

گاهی لازم می‌شود عنصری از یک عنصر، گروهی از عناصر یا چیزهای دیگر تشکیل شود. این کار با ایجاد یک گزینه امکانپذیر می‌گردد.

برای ارائه یک گزینه :

۱- عبارت **xsd:choice>** را تایپ کنید.

۲- با ویژگیهای minOccurs و maxOccurs می‌توانید تعداد دفعات ظاهر شدن مجموعه گزینه‌ها را تنظیم نمایید (صفحه ۱۰۱).

۳- علامت < را تایپ کنید.

۴- عناصر تشکیل دهنده مجموعه گزینه‌ها را اعلان کنید (صفحه ۷۶ و ۱۰۶) و یا ارجاع دهید (صفحه ۱۰۰).

۵- عبارت **<xsd:choice/>** را تایپ کنید.

نکته‌ها

- مقدار پیش فرض ویژگیهای minOccurs و maxOccurs برابر یک است. در این صورت عناصر مجموعه گزینه‌ها فقط یک بار می‌توانند در سند XML ظاهر گردند. اگر مقدار ویژگی maxOccurs بزرگ‌تر از یک باشد تعداد دفعات ظهور عناصر بیشتر خواهد شد. همچنین عبارت maxOccurs="unbounded" معادل علامت * در مجموعه گزینه‌های DTD است (صفحه ۴۷).
- از طرفی مجموعه گزینه‌ها می‌تواند شامل دنباله‌های تودرتو، مجموعه گزینه‌های دیگر یا ارجاع به گروههای نامگذاری شده (صفحه ۹۹) باشد.
- مجموعه گزینه‌ها می‌تواند شامل تعریف نوع پیچیده (صفحه۹۴)، دنباله، مجموعه گزینه‌های دیگر یا تعاریف گروههای نامگذاری شده (صفحه ۹۸) باشد.
- عنصر :xsd choice معادل خط عمودی در DTDها است (صفحه ۴۷).

code.xsd

```
<xsd:complexType name="animalType">
    ...
<xsd:choice>
    <xsd:element name="subspecies"
    type="subspeciesType"/>

    <xsd:sequence>
    <xsd:element name="region"
    type="xsd:string"/>
    <xsd:element name="population"
    type="popType"/></xsd:sequence>
</xsd:choice> ...
```

شکل ۷-۷ . برخی از گونه‌های در معرض خطر (endangered species)، زیرگونه (subspecies) ندارند. درچنین مواقعی تهیه فهرستی از زیرگونه‌ها مفید نیست. بلکه بهتر است فهرستی از منطقه (region) و جمعیت (population) گونه‌ها تهیه کرد.

code.xml

```
<animal><name language="English">
    Tiger</name>
...
<subspecies>
<name language="English">Amur</name>
<name language="Latin">P.t. altaica</name>
<region>Far East Russia</region>
<population year="1999">445</population>
</subspecies>
<subspecies>
<name language="English">Balian</name>
...
</animal>
```

code.xml

```
<animal><name language="English">Great River
    Otter</name>
...
<region>Peru, Northern Argentina</region>
<population year="2000">5000</population>
</animal>
```

شکل ۷-۸ . عنصرهای animal در هر دو مثال معتبرند. در اولی اختصاص هر تعداد عنصر subspecies مجاز است و در دومی به جای subspecies دنباله‌ای از عناصر region و population داریم.

ایجاد عناصر نامرتب

اگر عنصری شامل عناصر دیگری باشد که رعایت ترتیب در آنها مهم نیست می‌توان عناصر داخلی را به صورت گروه all معرفی نمود.

برای ایجاد ترتیبهای گوناگون در عناصر :

۱- برای آغاز گروه عناصر نامرتب عبارت <**xsd:all**> را تایپ کنید.

۲- اگر بخواهید می‌توانید به کمک تنظیم ویژگیهای minOccurs و maxOccurs تعداد دفعات ظاهر شدن گروه نامرتب را تعیین نمایید (صفحه ۱۰۱).

۳- علامت > را تایپ کنید.

۴- براساس مطالب صفحات ۷۶،۱۰۶ و ۱۰۰ می‌توانید عناصر گروه را اعلان کنید یا ارجاع دهید.

۵- برای تکمیل گروه، <**xsd:all**> را تایپ کنید.

نکته‌ها

- اعضای گروه all می‌توانند با توجه به مقادیر minOccurs یا maxOccurs یک یا هیچ دفعه با هر ترتیبی ظاهر شوند.
- مقادیر قابل قبول ویژگیهای minOccurs وmaxOccurs صفر و یک است.
- در یک گروه all اعلانها یا ارجاعهای عناصر، به صورت جداگانه تعریف می‌شوندنه گروهی. درضمن هیچ عنصری نمی‌تواند بیش از یک بار ظاهر شود.
- گروه all تنها می‌تواند شامل تعریف نوع پیچیده (صفحه۹۴) یا گروه نامگذاری شده (صفحه ۹۸) باشد.

code.xsd

```
<xsd:complexType name="subspeciesType">
<xsd:all>
<xsd:element ref="name"/>
<xsd:element name="region"
type="xsd:string" minOccurs="0"/>
<xsd:element name="population">
    <xsd:complexType base="xsd:integer"
    derivedBy="extension">
    <xsd:attribute name="year"
    type="xsd:year"/>
    </xsd:complexType></xsd:element>
</xsd:all>
</xsd:complexType>
```

شکل ۹-۷ . در گروه all عناصر name، region و population با هر ترتیبی می‌توانند در عناصر subspeciesType ظاهر شوند. همچنین عنصر region اختیاری است؛ زیرا برای ویژگی minOccurs مربوط به آن عدد صفر در نظر گرفته شده است (صفحه ۱۰۱). حذف این عنصر مشکلی به‌وجود نمی‌آورد.

code.xml

```
<subspecies>
<name language="English">Bengal</name>
<population year="1999">3159</population>
<region>India</region>
</subspecies>

<subspecies>
<population year="1998">1227</population>
<name language="English">Amoy</region>
</subspecies>
```

شکل ۱۰-۷ . عناصر subspecies در هر دو مثال معتبرند؛ زیرا region اختیاری است و ترتیب عناصر اهمیتی ندارد.

نامگذاری گروهها

اگر مجموعه‌ای از عناصر در قسمتهای گوناگون سند XML ظاهر گردند می‌توان آن عناصر را به صورت یک گروه در نظر گرفت تا فراخوانی آنها آسانتر گردد.

برای تعریف نام گروه :

۱- عبارت **<xsd:group** را تایپ کنید.

۲- سپس **name="label"** را تایپ نمایید. منظور از label واژه‌ای است که این گروه را معرفی می‌کند.

۳- علامت > را تایپ کنید.

۴- دنباله (صفحه ۹۵)، مجموعه گزینه‌ها (صفحه ۹۶) یا گروه نامرتب (صفحه ۹۷) تشکیل دهنده گروه را اعلان نمایید.

۵- برای تکمیل تعریف گروه عبارت **</xsd:group>** را تایپ کنید.

نکته‌ها

- یک گروه معادل پارامتر موجودیت در DTDهاست (صفحه ۶۰).
- در حالی که گروه در بالای الگو (و پایین xsd:schema) تعریف می‌شود امکان ارجاع به آن در تمام قسمتها وجود دارد (صفحه ۹۹).
- در XML Schema به دنباله‌ها، مجموعه گزینه‌ها، گروههای نامرتب و گروههای نامگذاری شده، گروهها یا گروههای مدل می‌گویند. در این کتاب به مجموعه اجزایی که با عنصر xsd:group تعریف می‌شوند و نامی دارند گروههای نامگذاری شده می‌گوییم.

code.xsd

```
<xsd:schema>
<xsd:group name="physical_traits">
   <xsd:sequence>
   <xsd:element name="weight"
   type="xsd:string"/>
   <xsd:element name="length"
   type="xsd:string"/>
   <xsd:element name="gestation"
   type="xsd:timeDuration"/>
   <xsd:element name="distinguishing"
   type="xsd:string"/>
   </xsd:sequence>
</xsd:group>
...
```

شکل ۱۱-۷ . یک گروه، فهرستی از عناصر مرتبط را که در یک یا چند عنصر با یکدیگر مورد استفاده قرار می‌گیرند تعریف می‌کند.

ارجاع به گروه نامگذاری شده

پس از ایجاد یک گروه می‌توانید از گروههای دیگر و یا در تعریف نوعهای پیچیده به آن ارجاع کنید.

برای ارجاع به یک گروه :

۱- در بخشی از الگو، جایی که می‌خواهید عناصر گروه ظاهر شوند عبارت **<xsd:group** را تایپ کنید.

۲- سپس **ref="label"** را تایپ نمایید. label واژه‌ای است که برای مشخص کردن گروه، هنگام ایجاد آن (مرحله ۲ صفحه قبل) به کار می‌برید.

۳- برای تکمیل ارجاع، علامتهای /> را تایپ کنید.

نکته

- می‌توان به یک گروه در اعلان نوع پیچیده (صفحه۹۴)، یک دنباله (صفحه ۹۵)، مجموعه‌ای از گزینه‌ها (صفحه ۹۶)، یک گروه نامرتب (صفحه ۹۷) یا گروههای نامگذاری شده دیگر ارجاع داد.

code.xsd

```
<xsd:element name="animal">
<xsd:complexType>
   <xsd:sequence>
   <xsd:element ref="name"/>
   <xsd:group ref="physical_traits"/>
   <xsd:element name="subspecies"
   type="subspeciesType"/>
   </xsd:sequence>
</xsd:complexType>
</xsd:element>

<xsd:element name="individual">
<xsd:complexType>
   <xsd:group ref="physical_traits"/>
   <xsd:attribute name="birthdate"
   type="xsd:date"/>
   <xsd:attribute name="nickname"
   type="xsd:string"/>
</xsd:complexType>
</xsd:element>
```

شکل ۱۲-۷ . عناصر animal و individual شامل فهرست عناصر گروه physical_traits می‌باشند (شکل ۱۱-۷). همچنین از عناصر و ویژگیهای جداگانه دیگری نیز تشکیل شده‌اند.

code.xml

```
<animal><name language="English">Tiger
   </name><weight>500
   pounds</weight><length>3 yards from nose
   to tail</length>...<subspecies>...</animal>

<individual birthdate="1999-06-10"
   nickname="Zoe"><weight>268
   pounds</weight><length>2.5
   yards</length>...</individual>
```

شکل ۱۳-۷ . عناصر می‌توانند به روشهای گوناگون در محتویات عناصر گوناگون به کار روند.

ارجاع به عناصر تعریف شده

عناصر نوعهای ساده و پیچیده که به صورت سراسری اعلان می‌گردند (داخل عنصر xsd:schema) می‌توانند در سند XML فراخوانی شده یا ارجاع داده شوند.

برای ارجاع به عنصر سراسری :

۱- در دنباله (صفحه۹۵)، مجموعه گزینه‌ها (صفحه ۹۶)، گروه نامرتب (صفحه ۹۷) یا تعریف گروه نامگذاری شده (صفحه ۹۸) که عناصر ظاهر می‌شوند, عبارت **<xsd:element** را تایپ کنید.

۲- سپس **ref="label"** را تایپ کنید. label نام عنصر سراسری است.

۳- تعداد دفعات ظاهر شدن عنصر را در این محل تعیین نمایید (صفحه ۱۰۱).

۴- برای تکمیل ارجاع عنصر، علامتهای /> را تایپ کنید.

نکته‌ها

- به اعلان عنصر فقط در تعاریف دنباله‌ها، مجموعه گزینه‌ها، گروههای نامرتب و گروههای نامگذاری شده می‌توان ارجاع داد.
- به عنصر سراسری در هر قسمت از سند XML می‌توان ارجاع داد. هر ارجاع باید شامل مقدار دقیق minOccurs و maxOccurs باشد.
- به عناصر محلی در اجزای تعاریفی که ظاهر می‌شوند به صورت خودکار ارجاع داده می‌شود. به این عناصر در قسمتهای دیگر نمی‌توان رجوع کرد.
- در صفحه ۷۱ درباره اعلانهای محلی و سراسری صحبت شده است.

code.xsd

```
<xsd:element name="name" type="nameType"/>
<xsd:complexType name="animalType">
    <xsd:sequence>
    <xsd:element ref="name" minOccurs="2"/>
    <xsd:element name="threats"
    type="threatsType"/>
    ...
    </xsd:complexType>
<xsd:complexType name="subspeciesType"/>
    <xsd:sequence>
    <xsd:element ref="name" minOccurs="1"/>
    <xsd:element name="region"
    type="xsd:string"/>
    ...
    </xsd:complexType>
```

شکل ۷-۱۴ . عنصر name در خط اول به صورت سراسری اعلان شده و می‌تواند در هر جای تعریف نوع پیچیده به آن رجوع کرد.

code.xml

```
<animal>
<name language="English">Tiger</name>
<name language="Latin">Panthera tigris</name>
<threats><threat>poachers</threat>
...
<subspecies>
<name language="English">Amur</name>
...
</subspecies>
</animal>
```

شکل ۷-۱۵ . براساس الگوی شکل ۷-۱۴، عنصر name باید حداقل دوبار در عنصر animal ظاهر شود؛ ولی هنگامی‌که شامل عنصر subspecies باشد کافی است یک بار ظاهر گردد.

کنترل تعداد دفعات ظاهر شدن

تعداد دفعات ظاهر شدن یک عنصر، دنباله، مجموعه گزینه‌ها، گروههای نامرتب و نامگذاری شده قابل کنترل می‌باشند.

برای تعیین حداقل تعداد ظاهر شدن یک عنصر یا گروه:

در برچسب شروع عنصر یا گروه عبارت **minOccurs="n"** را تایپ کنید. منظور از n مقدار حداقل تعداد دفعات مجاز ظاهر شدن عنصر یا گروه در سند است.

برای تعیین حداکثر تعداد ظاهر شدن یک عنصر یا گروه:

در برچسب شروع عنصر یا گروه عبارت **maxOccurs="n"** را تایپ کنید. منظور از n مقدار حداکثر تعداد دفعات مجاز ظاهر شدن عنصر یا گروه در سند است.

نکته‌ها

- ویژگی minOccurs می‌تواند مقادیر صحیح غیر منفی ({۰,۱,۲,...}) داشته باشد.
- ویژگی maxOccurs می‌تواند هر عدد صحیح مثبتی باشد. همچنین می‌توان از واژه unbounded استفاده کرد تا محدودیتی برای تعداد دفعات ظاهر شدن عنصر وجود نداشته باشد.
- مقدار پیش فرض برای minOccurs و maxOccurs عدد یک است.
- ویژگیهای minOccurs و maxOccurs نمی‌توانند برای عناصر سراسری (که در پایین عنصر xsd:schema اعلان می‌شوند) به کار روند. آنها تنها برای عناصر محلی قابل استفاده هستند و می‌توانند به عناصر سراسری ارجاع داده شوند.
- این ویژگیها می‌توانند در xsd:sequence، xsd:choice، xsd:all و ارجاع به گروههای نامگذاری شده استفاده گردند.

```
code.xsd
<xsd:element name="threat" type="xsd:string"
minOccurs="2" maxOccurs="5"/>
<xsd:element name="population"
type="xsd:integer"/>
```

شکل ۱۶-۷ . ویژگیهای minOccurs و maxOccurs تعداد دفعات ظاهر شدن یک عنصر را کنترل می‌کنند.

```
code.xml
<threat>poachers</threat>
<threat>habitat destruction</threat>
<threat>trade in tiger bones for traditional Chinese
medicine (TCM)</threat>
<population>28</population>
```

شکل ۱۷-۷ . عنصر threat می‌تواند حداقل ۲ و حداکثر ۵ بار استفاده شود. چون در اینجا ۳ بار ظاهر شده، معتبر می‌باشد. هرگاه برای عنصری نامی از ویژگیهای minOccurs و maxOccurs برده نشود (مانند عنصر population) مقدار پیش فرض (یعنی یک) برای آنها در نظر گرفته می‌شود و عنصر مربوطه باید یک بار در سند XML ظاهر گردد.

```
code.xsd
<xsd:choice minOccurs="0"
maxOccurs="unbounded">
<xsd:element name="sister_name"
type="xsd:string"/>
<xsd:element name="brother_name"
type="xsd:string"/>
</xsd:choice>
```

شکل ۱۸-۷ . ویژگیهای minOccurs و maxOccurs را می‌توان برای دنباله‌ها، مجموعه گزینه‌ها، گروههای نامرتب یا ارجاع به گروههای نامگذاری شده به کار برد. در این مثال امکان ظاهر شدن گزینه‌های گروه به هر تعداد وجود دارد. یعنی عناصر sister_name و brother_name می‌توانند هیچ بار یا بیشتر نمایش داده شوند. مانند اضافه کردن علامت * به گزینه در یک DTD (صفحه ۹۶).

تعریف عناصر متنی

اگر به نوع ساده‌ای نیاز دارید که از متن خاصی تشکیل شده و شامل ویژگیهایی نیز می‌باشد می‌توانید از یک نوع پیچیده فقط متنی استفاده نمایید.

برای تعریف نوع پیچیده فقط متنی :

۱- عبارت **<xsd:complexType>** را تایپ کنید.

۲- سپس **name="label"** را تایپ کنید. label معرف نوع پیچیده است نه نام عنصری که با استفاده از تعریف نوع پیچیده در اعلانهای چند عنصر به کار می‌برید.

۳- برای تکمیل برچسب شروع علامت < را تایپ کنید.

۴- **<xsd:simpleContent>** را تایپ نمایید.

۵- اگر می‌خواهید برای نوع ساده محدودیتهایی به‌وجود آورید عبارت **<xsd:restriction>** را تایپ کنید.

اگر می‌خواهید نوع ساده را گسترش دهید **<xsd:extension>** را تایپ نمایید.

۶- سپس **base="foundation">** را تایپ کنید. foundation تعریف نوع ساده‌ای است که نوع جدیدی روی آن ایجاد خواهید کرد.

۷- اگر دستورالعمل ۵ را اجرا کرده‌اید محدودیتهایی را که باعث محدود شدن تعریف نوع پیچیده می‌گردد اعلان نمایید (صفحه‌های ۹۰-۸۳).

۸- ویژگیها (صفحه ۱۰۸) یا گروه ویژگیهایی (صفحه ۱۱۱) را که در عناصر این نوع، ظاهر خواهند شد اعلان کنید.

۹- با توجه به عملی که در شماره ۵ انجام داده‌اید یکی از عبارتهای **</xsd:restriction>** یا **</xsd:extension>** را تایپ نمایید.

۱۰- عبارت **</xsd:simpleContent>** را تایپ کنید.

۱۱- عبارت **</xsd:complexType>** را برای تکمیل اعلان تایپ کنید.

code.xsd
```
<xsd:complexType name="popType">
  <xsd:simpleContent>
  <xsd:extension base="xsd:integer">
  <xsd:attribute name="year" type="xsd:year"/>
  </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
```

شکل ۷-۱۹ . عنصر simpleContent نشان می‌دهد که عناصر تعریف شده با تعریف این نوع پیچیده شامل متن خاصی (بر اساس نوع ساده) می‌باشند؛ ولی از عنصر دیگری تشکیل نمی‌گردند و در ضمن شاید ویژگیهایی نیز داشته باشند.

code.xsd
```
<xsd:element name="population"
  type="popType"/>
```

شکل ۷-۲۰ . همیشه باید عنصر مورد استفاده تعریف نوع پیچیده را اعلان نمود (صفحه ۱۰۶).

code.xml
```
<population year="1999">445</population>
```

شکل ۷-۲۱ . عنصر population با توجه به تعریف نوع پیچیده شکل ۷-۱۹ معتبر است؛ زیرا شامل یک عدد صحیح و ویژگی year می‌باشد.

code.xml
```
<population year="1999">Less than
500</population>
```

شکل ۷-۲۲ . عنصر population غیر معتبر است؛ زیرا عبارت "Less than 500" یک عدد صحیح نیست. در حالی که نوع ساده‌ای که نوع پیچیده بر اساس آن ایجاد شده به‌صورت عدد صحیح می‌باشد.

تعریف عناصر خالی

به عناصری که از ویژگیها تشکیل شده ولی بین برچسب شروع و پایان محتویاتی ندارند عناصر خالی می‌گویند.

به منظور تعریف نوعهای پیچیده برای عناصر خالی :

۱- **<xsd:complexType>** را تایپ کنید.

۲- **سپس <name="label">** را تایپ نمایید. label نوع پیچیده را مشخص می‌کند و نام عنصری که نوع پیچیده در اعلانهای چند عنصر به کار می‌برد نیست.

۳- **<xsd:copmplexContent>** را تایپ کنید.

۴- **<xsd:extension base="xsd:anyType"/>** را تایپ نمایید. در نتیجه مشخص می‌شود که نوع پیچیده براساس نوع خاصی نیست و محتویاتی نخواهد داشت.

۵- ویژگیهای عناصر این نوع را در صورت وجود اعلان نمایید (صفحه ۱۰۸).

۶- **</xsd:complexContent>** را تایپ کنید.

۷- **</xsd:complexType>** را برای تکمیل اعلان تایپ نمایید.

نکته

- برای به‌دست آوردن اطلاعات بیشتر درباره تعریف نوعهای پیچیده به صفحه‌های ۹۴ و ۱۰۷ مراجعه کنید.

code.xsd

```
<xsd:complexType name="sourceType">
  <xsd:complexContent>
  <xsd:extension base="xsd:anyType"/>
  <xsd:attribute name="sectionid"
  type="xsd:integer"/>
  <xsd:attribute name="newspaperid"
  type="xsd:integer"/>
  </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

شکل ۲۳-۷ . عنصر complexContent هنگامی که به اعلان ویژگیها نیاز دارید به کار می‌رود؛ ولی از عنصر دیگری تشکیل شده است.

code.xsd

```
<xsd:element name="source"
  type="sourceType"/>
```

شکل ۲۴-۷ . عنصر را در محلی که باید ظاهر شود اعلان کنید.

code.xml

```
<source sectionid="101" newspaperid="21"/>
```

شکل ۲۵-۷ . پس از اعلان عنصر source می‌توان آن را در سند XML به کار برید.

تعریف عناصر با محتویات ترکیبی

اگر یک نوع پیچیده، محتویات مختلط دارد باید این موضوع را هنگام اعلان نوع پیچیده اعلان نمود. یک نوع پیچیده مانند یک عنصر تعریف می‌شود.

برای ایجاد نوعهای پیچیده با محتویات مختلط :

۱- **xsd:complexType>** را تایپ کنید.

۲- سپس عبارت **name="label"** را تایپ کنید. label مشخصه نوع پیچیده است. چون label نام عنصر نیست می‌توان تعریف این نوع پیچیده را در اعلان چند عنصر به کار برد.

۳- برای اینکه عنصر بتواند شامل عناصر، ویژگیها و یا حتی متن باشد عبارت **mixed="true"** را تایپ نمایید.

۴- برای تکمیل برچسب شروع، علامت > را تایپ کنید.

۵- یک دنباله (صفحه ۹۵)، گزینه (صفحه ۹۶)، گروه نامرتب (صفحه ۹۷) یا ارجاع به یک گروه نامگذاری شده را (که می‌توان شامل هر یک ازموارد مذکور باشد) اعلان کنید. در این صورت مشخص می‌گردد که نوع پیچیده شامل چه عناصری می‌تواند باشد.

۶- سپس ویژگیها (صفحه ۱۰۸) یا گروه ویژگیهایی (صفحه ۱۱۱) را که ممکن است در عناصر این نوع، درصورت وجود، ظاهر شوند اعلان کنید یا ارجاع دهید.

۷- با تایپ عبارت **xsd:complextype/>** اعلان را تکمیل نمایید.

نکته

- محتویات مختلط عناصر برای توصیف اطلاعاتی که بخشی از آن مبتنی بر متن است بسیار مناسب می‌باشد.

code.xsd

```
<xsd:complexType name="paragraph"
    mixed="true">
    <xsd:sequence>
    <xsd:element name="name"
    type="nameType">
    </xsd:sequence>
    <xsd:attribute name="length"
    type="xsd:string"/>
</xsd:complexType>
```

شکل ۲۶-۷ . تعریف paragraph شامل یک عنصر و یک ویژگی است. ویژگی mixed="true" امکان وجود هر نوع متنی را فراهم می‌نماید.

code.xsd

```
<xsd:element name="description"
    type="paragraph"/>
```

شکل ۲۷-۷ . باید عنصر را در محلی که قرار است ظاهر شود اعلان نمایید.

code.xml

```
<description length="short">The <name
language="English">tiger</name>
(panthera tigris), largest of all cats, is one of the
biggest and most fearsome predators in the
world.</description>
```

شکل ۲۸-۷ . عنصر description شامل متن دلخواه (های‌لایت شده) و عناصر (های‌لایت و پررنگ) می‌باشد.

تعریف نوعهای پیچیده بر اساس نوعهای پیچیده موجود

نوعهای پیچیده را می‌توان بر اساس نوعهای پیچیده موجود ایجاد کرد. نوع پیچیده جدید با تمام اطلاعات نوع موجودآغاز می‌گردد و سپس جزئیاتی به آن اضافه یا از آن حذف می‌گردد.

برای ایجاد نوعهای پیچیده بر اساس نوعهای موجود :

۱- برای آغاز تعریف نوع پیچیده جدید عبارت **<xsd:complexType** را تایپ کنید.

۲- **name="label>"** را تایپ نمایید. label مشخصه نوع پیچیده‌ای است که در حال ایجاد آن هستید.

۳- **<xsd:complexContent>** را تایپ نمایید.

۴- برای اضافه کردن اجزای نوع پیچیده جدید به نوع پیچیده موجود عبارت **<xsd:extension** را تایپ کنید. یا اگر اجزای نوع پیچیده جدید محدودتر از نوع پیچیده موجود است عبارت **<xsd:restriction** را تایپ نمایید.

۵- **base="existing"** را تایپ کنید. existing نام نوعی است که نوع جدید بر اساس آن ایجاد خواهد شد.

۶- علامت > را تایپ کنید.

۷- دنباله‌ها، گزینه‌ها یا ارجاع به گروههای نامگذاری شده‌ای را که ممکن است بخشی از نوع جدید باشند، اعلان نمایید.

۸- ویژگیهای نوع جدید را اعلان کنید یا ارجاع دهید.

۹- بر چسب پایان مناسب با شماره ۴ را تایپ کنید.

۱۰- **</xsd:complexContent>** را تایپ نمایید.

۱۱- با تایپ **</xsd:complexType>** تعریف نوع پیچیده تکمیل می‌گردد.

code.xsd
```
<xsd:complexType name="characteristicsType">
    <xsd:sequence>
    <xsd:element name="weight"
    type="xsd:string"/>
    <xsd:element name="length"
    type="xsd:string"/>
    <xsd:attribute name="kind"
    type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
```

شکل ۲۹-۷ . تعریف characteristicsType به دنباله عناصر weight و length و ویژگی kind نیاز دارد.

code.xsd
```
<xsd:complexType name="birthType">
    <xsd:complexContent>
    <xsd:extension base="characteristicsType">
    <xsd:sequence>
    <xsd:element name="mother"
    type="xsd:string"/>
    <xsd:element name="birthdate"
    type="xsd:date"/>
    </xsd:sequence>
    </xsd:extension></xsd:complexContent>
    </xsd:complexType>
```

شکل ۳۰-۷ . نوع جدید birthType فقط عناصر اضافه را نشان می‌دهد.

code.xsd
```
<xsd:element name="birth_characteristics"
    type="birthType"/>
```

شکل ۳۱-۷ . همیشه باید عنصری را که از تعریف نوع پیچیده استفاده می‌کند اعلان نمود (صفحه ۱۰۶).

code.xml
```
<birth_characteristics kind="normal">
<weight>2-3 pounds</weight>
<length>18-24 inches</length>
<mother>Danai</mother>
<birthdate>1999-06-10</birthdate>
</birth_characteristics>
```

شکل ۳۲-۷ . عنصر birth_characteristics باید شامل اجزایی از characteristicsType (عناصر weight و length و ویژگی kind) و اجزای جدید (mother و birthdate) باشد.

اعلان عنصر نوع پیچیده

پس از تعریف یک نوع پیچیده، می‌توان آن را به عنصری که در سند XML به کار خواهد رفت نسبت داد.

برای اعلان عنصر نوع پیچیده :

۱- برای شروع اعلان عنصر، عبارت **xsd:element>** را تایپ کنید (**شکل ۳۳-۷**).

۲- **type="lable"** را تایپ کنید. label مشخصه‌ای است که برای تعریف نوع پیچیده به کار برده‌اید (مرحله ۲ صفحه ۹۴).

۳- علامتهای /> را برای تکمیل اعلان عنصر تایپ نمایید.

نکته‌ها

- عناصر را می‌توان به صورت *سراسری* (در سطوح اولیه یک الگو، پایین عنصر xsd:schema) یا *محلی*، در قسمتهای گوناگون اعلان نمود. این قسمتها عبارتند از: در یک تعریف نوع پیچیده (صفحه ۹۴)، یک دنباله (صفحه ۹۵)، مجموعه‌ای از گزینه‌ها (صفحه ۹۶)، یک گروه نامرتب (صفحه ۹۷) یا تعریف گروه نامگذاری شده (صفحه ۹۸).
- اگر عنصری را به صورت محلی اعلان نمایید (صفحه ۷۱) می‌توانید تعداد صفحه‌های ظهور عنصر را توسط ویژگیهای maxOccurs, minOccurs کنترل کنید (صفحه ۱۰۱).

```
code.xsd
<xsd:complexType name="characteristicsType">
    <xsd:sequence>
    <xsd:element name="weight"
    type="xsd:string"/>
    <xsd:element name="length"
    type="xsd:string"/>
    <xsd:attribute name="kind"
    type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
```

شکل ۳۳-۷ . در این مثال تعریف نوع پیچیده‌ای که در شکل ۲۹-۷ وجود داشت مشاهده می‌شود.

```
code.xsd
<xsd:element name="characteristics"
    type="characteristicsType"/>

<xsd:complexType name="animalType"/>
    <xsd:sequence>
    <xsd:element name="name" type="nameType"
    minOccurs="2"/>
    <xsd:element ref="characteristics"
    minOccurs="1"/>
    ...
```

شکل ۳٤-۷ . عنصر را می‌توان به صورت سراسری در بالای سند الگو اعلان کرد. بنابراین برای استفاده از این عنصر می‌توانید در هر قسمت از الگو به آن ارجاع دهید.

```
code.xsd
<xsd:complexType name="animalType"/>
    <xsd:sequence>
    <xsd:element name="name" type="nameType"
    minOccurs="2"/>
    <xsd:element name="characteristics"
    type="characteristicsType" minOccurs="1"/>
    ...
```

شکل ۳۵-۷ . با اعلان محلی یک عنصر در اجزای دیگر (مانند تعریف یک نوع پیچیده یا گروه نامگذاری شده) به صورت خودکار به آن عنصر اشاره خواهد شد.

عناصری با نوعهای پیچیده بی نام

اگر به استفاده مجدد از نوع پیچیده‌ای نیاز ندارید بهتر است در قسمت اعلان یک عنصر، نوع پیچیده بدون نامی ایجاد نمایید.

برای اعلان عنصری با نوع پیچیده بی نام :

۱- عبارت **<xsd:element** را تایپ کنید.

۲- **name="label">** را تایپ نمایید. منظور از label نام عنصری است که اعلان کرده‌اید و در سند XML به صورت <label> ظاهر خواهد شد.

۳- با تایپ **<xsd:complexType>** نوع پیچیده بی نام را اعلان کنید.

۴- برای تعیین عناصر تشکیل دهنده نوع پیچیده، یک دنباله (صفحه ۹۵)، گزینه (صفحه ۹۶)، گروه نامرتب (صفحه ۹۷) یا ارجاعی به یک گروه نامگذاری شده که می‌تواند شامل هر یک از موارد مذکور باشد (صفحه ۹۹) اعلان نمایید.

۵- سپس ویژگیها (صفحه ۱۵۸) یا گروه ویژگیهایی (صفحه ۱۱۱) را که در عناصر این نوع ظاهر خواهند شد، اعلان کنید یا ارجاع دهید.

۶- برای تکمیل تعریف نوع پیچیده بی نام عبارت **</xsd:complexType>** را تایپ کنید.

۷- با تایپ عبارت **</xsd:element>** اعلان عنصر نوع پیچیده را کامل نمایید.

نکته

- تنها تفاوت بین نوعهای پیچیده بی‌نام و نامگذاری شده این است که نوع پیچیده نامگذاری شده را می‌توان برای اعلان عناصر گوناگون دلخواه و بر اساس نوعهای پیچیده دیگر به دفعات مورد استفاده قرار داد, در حالی که نوع پیچیده بی نام فقط عنصر داخل خود را تعریف می‌کند.

code.xsd

```
<xsd:element name="characteristics">
    <xsd:complexType>
    <xsd:sequence>
    <xsd:element name="weight"
    type="xsd:string"/>
    <xsd:element name="length"
    type="xsd:string"/>
    <xsd:attribute name="kind"
    type="xsd:string"/>
    </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

شکل ۳۶-۷ . این بار نوع characteristics به صورت نوع بی نام تعریف شده است. حالا فقط می‌توان تعریف این نوع پیچیده را برای عنصری که شامل characterstics است به کار برد. عنصر نیز به دو صورت سراسری و محلی قابل تعریف می‌باشد.

code.xml

```
<characteristics kind="physical">
<weight>500 pounds</weight>
<length>3 yards from nose to tail</length>
</characteristics>
```

شکل ۳۷-۷ . این یک مثال معتبر است. نوع پیچیده نامگذاری شده و بی‌نام, یکسان به نظر می‌رسند.

اعلان ویژگیها

با توجه به اینکه یک ویژگی از عناصر و ویژگیهای دیگر تشکیل نمی‌گردد، همیشه یک نوع ساده محسوب می‌شود. یک ویژگی همیشه در عنصر یک نوع پیچیده ظاهر می‌شود. در این فصل درباره ایجاد ویژگیها بیش از پیش صحبت خواهیم کرد.

برای اعلان یک ویژگی :

۱- عبارت **<xsd:attribute>** را در قسمت تعریف نوع پیچیده تایپ کنید.

۲- سپس عبارت **name="label"** را تایپ نمایید. label نامی است که برای ویژگی سند XML به کار خواهید برد.

۳- سپس **type="simple"** را تایپ کنید. منظور از simple نوع ساده‌ای است که ویژگی به آن تعلق دارد. برای به دست آوردن اطلاعات بیشتر درباره نوعهای ساده به صفحه ۷۶ مراجعه نمایید.

همچنین می‌توانید عبارت **ref="label"** را تایپ کنید. label مشخصه تعریف یک ویژگی است که در مراحل ۱ و ۲ به صورت سراسری اعلان نموده‌اید.

۴- برای تکمیل برچسب شروع علامت > را تایپ نمایید.

۵- هرگونه محدودیت و طرحی را که می‌خواهید اضافه کنید (صفحه‌های ۸۳ تا ۹۰).

۶- عبارت **</xsd:attribute>** را تایپ کنید.

نکته‌ها

- اگر قصد اضافه کردن طرح و محدودیتی را ندارید به جای انجام مراحل ۴ تا ۶، علامتهای /> را تایپ کنید.
- برای ویژگیها، نوعهای ساده درونی بیشتری وجود دارد که در آدرس http://www.w3.org/TR/xmlschema-2 قابل دسترسی هستند.
- امکان تعریف یک ویژگی نوع ساده بی‌نام نیز وجود دارد.
- محل اعلان یک ویژگی در نوع پیچیده‌ای که به آن تعلق دارد پس از اعلان تمام اجزای نوع پیچیده می‌باشد.

code.xsd

```
<xsd:element name="source">
    <xsd:complexType>
    <xsd:complexContent>
    <xsd:extension base="xsd:anyType">
    <xsd:attribute name="sectionid"
    type="xsd:string"/>
    <xsd:attribute name="newspaperid"
    type="xsd:string"/>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
</xsd:element>
```

شکل ۳۸-۷. اعلان یک ویژگی شبیه اعلان عنصر نوع ساده است.

code.xml

```
<source sectionid="101" newspaperid="21"/>
```

شکل ۳۹-۷ . ویژگیها همیشه در برچسب شروع محتویات عنصر ظاهر می‌گردند.

تعیین لزوم وجود ویژگی

تعریف ویژگی اختیاری است. یعنی بودن یا نبودن ویژگی هیچ مشکلی در سند XML به وجود نمی‌آورد. از طرفی می‌توانید لزوم ظهور یا عدم ظهور ویژگی تعریف شده‌ای را تعیین کنید تا هنگام بررسی اعتبار سند XML در نظر گرفته شود.

برای تعیین لزوم ظهور یک ویژگی :

۱- هنگام تعریف یک ویژگی عبارت **use=** را تایپ کنید.

۲- سپس **"required"** را تایپ نمایید. در نتیجه برای معتبر بودن سند، ظهور ویژگی الزامی می‌گردد.

۳- با تایپ عبارت **value="must"** می‌توانید تنها مقدار قابل قبول ویژگی (must) را تعیین نمایید.

برای تعیین عدم لزوم ظهور یک ویژگی:

هنگام تعریف ویژگی عبارت **use="prohibited"** را تایپ کنید تا شرط اعتبار سند عدم ظهور ویژگی گردد.

نکته

- تایپ عبارت **use="optional"** باعث می‌شود تا ظهور ویژگی اختیاری باشد. از آنجایی که این مقدار پیش فرض است تایپ آن ضرورتی ندارد و تنها باعث شلوغی کار می‌گردد.

code.xsd

```
<xsd:element name="source">
    <xsd:complexType>
    <xsd:complexContent>
    <xsd:extension base="xsd:anyType">
    <xsd:attribute name="sectionid"
    type="xsd:string" use="required"/>
    <xsd:attribute name="newspaperid"
    type="xsd:string"/>
    </xsd:extension>
    </xsd:complexContent>
    </xsd:complexType>
</xsd:element>
```

شکل ۴۰- ۷. چون ظهور ویژگی اختیاری است الزام ظهور آن باید تعیین شود.

code.xml

```
<source sectionid="101"/>
```

code.xml

```
<source sectionid="101" newspaperid="21"/>
```

شکل ۴۱-۷ . هر دو عنصر source معتبر هستند؛ زیرا تنها ظهور ویژگی sectionid اجباری است.

تعریف دوباره محتویات ویژگی

برای تعریف دوباره محتویات یک ویژگی، به دو روش می‌توان از یک الگو استفاده کرد. یک راه، نوشتن محتویات ویژگی هنگام ظاهر شدن در سند XML است. راه دیگر، در نظر گرفتن مقدار اولیه برای ویژگی است و این مقدار به محل ظهور ویژگی در سند بستگی ندارد. به مقدار روش اول مقدار ثابت و در روش دوم مقدار پیش فرض می‌گویند.

برای تعیین مقدار ثابت ویژگی :

۱- در برچسب ویژگی عبارت **use="fixed"** را تایپ کنید.

۲- سپس عبارت **value="content"** را تایپ نمایید. منظور از content مقدار ویژگی است که هنگام ظهور در سند معتبر XML خواهد داشت.

برای تنظیم مقدار اولیه یک ویژگی :

۱- در برچسب شروع ویژگی عبارت **use="default"** را تایپ نمایید.

۲- سپس عبارت **value="content"** را تایپ کنید. content مقدار ویژگی است که در صورت ظاهر شدن ویژگی در سند، باز هم برای آن در نظر گرفته خواهد شد.

نکته‌ها

- مقادیر ثابت ویژگیها تنها در صورتی برای آنها در نظر گرفته می‌شوند که ویژگی مربوطه در سند ظاهر گردد. در غیر این صورت مقداری برای آنها وجود نخواهد داشت.
- مقادیر پیش فرض در صورت عدم ظهور ویژگی در سند باز هم برای آنها در نظر گرفته می‌شوند.
- اگر یک ویژگی مقدار پیش فرض داشته باشد و در سند XML ظاهر گردد امکان تعیین مقادیر ثابت برای آن وجود دارد و به مقدار پیش فرض خود محدود نمی‌گردد.

code.xsd
```
<xsd:attribute name="sectionid" type= "xsd:string"
    use="fixed" value="101"/>
<xsd:attribute name="newspaperid"
    type="xsd:string"/>
```

شکل ۷-۴۲ . مقدار ویژگی sectionid هنگام ظاهر شدن در سند XML باید مقدار ۱۰۱ داشته باشد. این ویژگی می‌تواند حذف گردد. این مثال از تعریف نوع پیچیده شکل ۷-۴۰ برگزیده شده است.

code.xml
```
<source sectionid="101" newspaperid="21"/>
```

شکل ۷-۴۳ . عنصر source در مقایسه با الگوی داده شکل ۷-۴۲ معتبر است. در صورت حذف ویژگی sectionid نیز مشکلی پیش نخواهد آمد.

code.xml
```
<source sectionid="456" newspaperid="21"/>
```

شکل ۷-۴۴ . عنصر source این مثال با توجه به اعلان شکل ۷-۴۲ معتبر نمی‌باشد.

code.xsd
```
<xsd:attribute name="sectionid" type= "xsd:string"
    use="fixed" value="101"/>
<xsd:attribute name="newspaperid" type=
    "xsd:string" use="default" value="21"/>
```

شکل ۷-۴۵ . اگر ویژگی newspaperid با مقدار پیش‌فرض تعریف شود این مقدار بدون توجه به ظهور یا عدم ظهور ویژگی برای آن در نظر گرفته خواهد شد. از تعریف نوع پیچیده شکل ۷-۴۰ اقتباس شده است.

code.xml
```
<source sectionid="101" newspaperid="21"/>
```

code.xml
```
<source sectionid="101" newspaperid="25"/>
```

شکل ۷-۴۶ . عناصر source هر دو مثال معتبر هستند. مقدار پیش‌فرض تنها به‌عنوان مقدار اولیه در نظر گرفته می‌شود و مقادیر دیگر نیز قابل قبول خواهند بود.

تعریف گروه ویژگی

اگر برای چند عنصر به مجموعه‌ای از ویژگیها نیاز دارید، بهتر است یک گروه ویژگی تعریف کنید. سپس از داخل نوع پیچیده هر عنصر به آن گروه ویژگی ارجاع دهید.

برای تعریف گروه ویژگی :

۱- عبارت **<xsd:attributeGroupname="label">** را تایپ کنید. label مشخصه گروه ویژگی است.

۲- هر یک از ویژگیهای گروه را اعلان کنید یا به هر یک از آنها ارجاع دهید.

۳- در آخر به منظور تکمیل تعریف گروه ویژگی، عبارت **</xsd:attributeGroup>** را تایپ نمایید.

نکته‌ها

- یک گروه ویژگی باید به صورت سراسری اعلان گردد. یعنی در ابتدای الگو و پایین عنصر xsd:schema.
- پس از اعلان یک گروه ویژگی باید مانند توضیحات صفحه ۱۱۲ به آن ارجاع دهید.
- فقط می‌توان به ویژگیهای سراسری ارجاع نمود. یعنی آنهایی که در ابتدای الگو اعلان شده‌اند (صفحه ۷۱).
- اگر ویژگیها را داخل گروه ویژگی به صورت محلی اعلان کنید تنها برای گروه ویژگی مربوطه قابل دسترس خواهند بود. گاهی این حد از دسترسی کافی است.
- یک گروه ویژگی می‌تواند به گروههای ویژگی دیگر نیز ارجاع داده شود.

code.xsd

```
<xsd:attributeGroup name="imageAtts">
    <xsd:attribute name="filename" type="xsd:uri-reference"/>
    <xsd:attribute name="x" type="xsd:integer"/>
    <xsd:attribute name="y" type="xsd:integer"/>
</xsd:attributeGroup>
```

شکل ۷-۴۷ . با نامگذاری یک گروه ویژگی، استفاده از ویژگیهای گروه در تعاریف نوعهای گوناگون آسانتر خواهد شد.

ارجاع به گروههای ویژگی

پس از تعریف یک گروه ویژگی می‌توان از هر محلی به ویژگیهای آن گروه اشاره نمود.

برای ارجاع یک گروه ویژگی :

۱- در تعریف یک نوع پیچیده، پس از اعلان تمام عناصر تشکیل دهنده آن، عبارت **<xsd:attributeGroup** را تایپ کنید.

۲- سپس عبارت **ref="label"/>** را تایپ نمایید. label نام گروه ویژگی مورد نظر است که در مرحله ۱ صفحه ۱۱۱ به کار بردید.

نکته‌ها

- گروه ویژگیها مانند پارامتر موجودیتها در DTDها هستند (صفحه ۶۰) ولی البته برای ارائه ویژگیها محدودیتهایی دارند.
- ویژگیها (و گروه ویژگیها) همیشه در انتهای اجزا و پس از اعلان تمام عناصر اعلان می‌گردند.

code.xsd

```
<xsd:element name="picture">
   <xsd:complexType>
   <xsd:complexContent>
   <xsd:extension base="xsd:anyType">
   <xsd:attributeGroup ref="imageAtts"/>
   </xsd:extension>
   </xsd:complexContent>
   </xsd:complexType>
</xsd:element>

<xsd:element name="video">
   <xsd:complexType>
   <xsd:complexContent>
   <xsd:extension base="xsd:anyType">
   <xsd:element name="description"
   type="xsd:string"/>
   <xsd:element name="running_time"
   type="xsd:timeDuration"/>
   <xsd:attributeGroup ref="imageAtts"/>
   <xsd:attribute name="format"
   type="xsd:string"/>
   </xsd:extension>
   </xsd:complexContent>
   </xsd:complexType>
</xsd:element>
```

شکل ۷-۴۸ . تایپ ref="imageAtts" از تعریف سه ویژگی جداگانه بسیار آسان‌تر و سریعتر می‌باشد. عنصر Video یک ویژگی اضافی به نام format دارد که پس از گروه ویژگی اعلان شده است.

code.xml

```
<picture filename="tiger.jpg" x="200" y="197"/>

<video filename="tiger.ram" x="100" y="100"
format="realplayer">
<description>A tiger moves gracefully through the
dappled light of the forest.</description>
<running_time>PT3M43S</running_time>
</video>
```

شکل ۷-۴۹ . عناصر picture و video ویژگیهای filename ، x و y دارند ولی هر یک از آنها یک بار تعریف شده‌اند.

۸

استفاده از فضاهای نام در XML

در دو فصل گذشته چگونگی تعریف الگو، مجموعه‌ای از عناصر و ویژگیهای تشکیل دهنده اسناد XML را یاد گرفتید. حالا فرض کنید که می‌خواهید اسناد خود را با اسناد شخص دیگری ترکیب نمایید. ولی آن شخص در الگوهای خود از نامهایی استفاده کرده که شما برای عناصر سراسری خود در نظر گرفته‌اید. برای مثال، سند Endangered Specise مطرح شده در این کتاب از "source" برای نشان دادن منبع داده‌های سایت World Wildlife Fund's به کار رفته است. در حالی که شاید شما از Source برای نمایش منابع آب رودخانه‌ها استفاده کرده باشید. اگر بخواهیم این اسناد را با یکدیگر ترکیب کنیم داده عنصر Source بی معنی خواهد شد.

راه حل این مشکل ایجاد یک superlabel است که قادر به تشخیص عناصر افراد مختلف باشد. برای مثال می‌توان عبارت "Liz:" را برای تمام عناصر یک سند در نظر گرفت. بنابراین عنصر Liz: Source از یک سند با عنصر your:source از سند دیگر اشتباه نمی‌شود. البته Liz حالت انحصاری ندارد. نام superlaber از یک آدرس URL بر اساس نام دامنه مناسب انتخاب می‌شود تا منحصر به فرد باشد. زیرا اسامی دامنه‌ها انحصاری می‌باشند.

به superlabel در اصطلاح، نام فضای نام می‌گویند. فضای نام مجموعه‌ای از نامهای عناصر مرتبط است.

به دلیل اینکه عناصری که به صورت محلی اعلان می‌شوند منحصر به فرد می‌باشند (صفحه ۷۱) کمتر به استفاده از فضای نام نیاز پیدا می‌کنند.

code.xsd

```
<xsd:schema xmlns:xsd="http://www.w3.org/
    2000/10/XMLSchema">
<xsd:element name="name" type="xsd:string"/>
<xsd:element name="source" type="xsd:string"/>
<xsd:element name="river">
    <xsd:complexType>
    <xsd:sequence>
    <xsd:element ref="name"/>
    <xsd:element ref="source"/>
    ...
    </xsd:element>
...
```

شکل ۸-۱ . در الگوی فرضی "RiversML" تمام عناصر به صورت سراسری در بالای سند اعلان شده‌اند و سپس در تعریف نوع پیچیده به آنها ارجاع داده شده است.

code.xsd

```
<xsd:schema xmlns:xsd="http://www.w3.org/
    2000/10/XMLSchema">
<xsd:element name="name" type="xsd:string"/>
<xsd:element name="source" type="xsd:string"/>
<xsd:element name="animal">
    <xsd:complexType>
    <xsd:sequence>
    <xsd:element ref="name"/>
    <xsd:element ref="source"/>
    ...
    </xsd:element>
...
```

شکل ۸-۲ . در این الگو تمام عناصر به صورت سراسری اعلان گشته‌اند. اگر اسناد ایجاد شده توسط این الگو را با اسناد ایجاد شده توسط شکل ۸-۱ ترکیب کنیم عناصر name و source دو گروه با یکدیگر تداخل نخواهند داشت.

طراحی یک نام فضای نام

به دلیل اینکه فضاهای نام بخشهای مشابه عناوین عناصر را از یکدیگر تشخیص می‌دهند باید نام منحصر به فردی داشته باشند. در XML، نامهای فضاهای نام شکلی از URL هستند.

برای طراحی یک نام فضای نام :

۱- کار نامگذاری را با نام دامنه خود آغاز نمایید.

۲- برای منحصر به فرد بودن نام فضای نام، اطلاعات توضیحی بیشتری (مانند یک مسیر URL) اضافه کنید.

نکته‌ها

- عملی‌ترین روش برای ایجاد نام منحصر به فرد فضای نام، یک آدرس URL است.
- اگر بخواهید می‌توانید به فضای نام، اطلاعات توضیحی اضافه کنید.
- استفاده از دامنه شخصی به عنوان اساس نامگذاری فضاهای نام راه مطمئنی برای ایجاد نامهای منحصر به فرد است؛ زیرا نام دامنه‌ها انحصاری هستند.
- در ابتدای نام فضای نام از نام دامنه هیچ شخص دیگری استفاده نکنید.
- دلیل استفاده از آدرسهای URL برای نامهای فضاهای نام، انحصاری بودن آنهاست. از آنجا که یک URL ممکن است به یک DTD یا یک الگو نیز اشاره کند این روش از طرف W3C روش تأیید شما و قطعی نیست. بهتر است از آدرسی استفاده کنید که هیچ‌گاه تغییر نکند و به هیچ فایلی ارجاع داده نشود.

شکل ۳-۸ . یک فضای نام می‌تواند شکلی از یک آدرس URL باشد. برای مثال با پروتکل استاندارد HTTP آغاز شود و در ادامه آن نام دامنه شما قرار گیرد. می‌توان یک توضیح کوتاه و یک شماره نگارش نیز به طور اختیاری به آن اضافه نمود. آدرس URL نباید به یک فایل اشاره کند.

اعلان فضاهای نام پیش فرض

پس از طراحی یک نام فضای نام می‌توان آن را برای تمام سند یا هر بخش از آن به صورت پیش فرض اعلان کرد.

به منظور اعلان یک فضای نام پیش فرض برای یک عنصر و تمام فرزندان آن، .

۱- در برچسب شروع عنصری که می‌خواهید فضای نام داشته باشد عبارت **xmlns=** را تایپ کنید.

۲- سپس **"URL"** را تایپ کنید. URL نام فضای نام شماست (صفحه ۱۱۴).

نکته‌ها

- اختصاص دادن فضای نام برای یک عنصر به صورتی که در بالا گفته شد نه تنها بر آن عنصر بلکه بر تمام عناصر تشکیل دهنده آن نیز اثر می‌گذارد. البته در صورتی که به این عناصر، فضاهای نام دیگری تخصیص داده نشده باشد.
- اگر یک فضای نام پیش فرض برای عنصر ریشه یک سند در نظر بگیرید، تمام عناصر آن سند تحت پوشش فضای نام مذکور قرار می‌گیرند. البته به شرطی که به عناصر دیگر فضای نام دیگری اختصاص دهید (صفحه‌های ۱۱۶ و ۱۱۷).
- فضای نامی که برای تمام (یا بخشی از) سند به کار رود فضای نام پیش فرض نامیده می‌شود؛ زیرا بر تمام عناصر سند (یا بخشی از آن) اثر می‌گذارد. مگر آنکه برای برخی از عناصر، فضای نام دیگری در نظر گرفته شود.
- با تعیین فضای نام پیشوندی برای یک عنصر می‌توانید از فضای نام پیش فرض صرف‌نظر کنید (صفحه‌های ۱۱۶ و ۱۱۷). این عمل بر عناصر فرزند اثری نمی‌گذارد.

code.xml

```
<endangered_species>
<animal>
<name language="English">Giant River
    Otter</name>
...
<source sectionid="122" newspaperid="21"
    contentid="630"
    xmlns="http://www.cookwood.com/ns/
    end_species1"/>
<picture filename="otter.jpg" x="200" y="197"/>
```

شکل ٤-٨ . عنصر Source جزئی از فضای نام http://www.cookwood.com/ns/end_species1 شده است. برای عناصر دیگر هیچ فضای نامی در نظر گرفته نشده است.

code.xml

```
<endangered_species
    xmlns="http://www.cookwood.com/ns/
    end_species1">
<animal>
<name language="English">Giant River
    Otter</name>
<name language="Latin">pteronura
    brasiliensis</name>
<threats><threat>habitat destruction</threat>
<threat>hunting</threat>
<threat>mercury poisoning from gold
    mining</threat>
<threat>pollution from fossil fuel
    extraction</threat>
```

شکل ٥-٨ . در این مثال، با اعلان فضای‌نام در عنصر ریشه،تمام عناصر سند به فضای‌نام زیر اختصاص یافته‌اند: http://www.cookwood.com/ns/end_species1 در صفحات ۱۱۶ و ۱۱۷ درباره صرف‌نظر کردن از فضای نام پیش فرض صحبت شده است.

فضاهای نام برای عناصر جداگانه

اگر می‌خواهید در سند خود به عناصر جداگانه، فضاهای نام جداگانه اختصاص دهید به گونه‌ای که روی عناصر فرزند آنها اثر نگذارد باید ابتدا برای فضای نام یک پیشوند در نظر بگیرید و سپس آن پیشوند را برای عناصر جداگانه به کار برید.

به منظور اعلان یک پیشوند برای نام فضای نام :

۱- در عنصر ریشه سند، عبارت **xmlns:prefix** را تایپ کنید. منظور از prefix پیشوند فضای نام است.

۲- سپس عبارت **"URL"=** را تایپ نمایید. URL فضای نامی است که پیشوند را برای آن در نظر گرفته‌اید.

نکته‌ها

- در پیشوند نمی‌توان از حروف کوچک و بزرگ x ، m و l استفاده کرد.
- می‌توانید یک پیشوند را برای هر یک از عناصر تشکیل دهنده عنصری که پیشوند را برای آن اعلان کرده‌اید به کار برید. یعنی اگر پیشوندی را برای عنصر ریشه در نظر بگیرید آن پیشوند برای تمام عناصر و ویژگیهای سند که از آن عنصر ریشه مشتق شده‌اند قابل استفاده است. اگر پیشوندی را در عنصر دیگری اعلان کنید تنها برای همان عنصر و عناصر فرزند آن قابل استفاده است.
- تعداد فضاهای نام پیشوندی در هر عنصر به نیاز شما بستگی دارد.

code.xml

```
<endangered_species xmlns="http://www.
cookwood.com/ns/ end_species1"
xmlns:rivers="http://www.cookwood.com/ns/
rivers1">
<animal>
<name language="English">Giant River
Otter</name>...
<source sectionid="122" newspaperid="21"
```

شکل ۶-۸ . با اعلان فضای نام rivers1 در عنصر ریشه سند و نسبت دادن آن به یک پیشوند می‌توان عناصر متعلق به فضای نام revers1 را برای عناصر جداگانه در نظر گرفت (شکل ۸-۸). اعلان فضای نام rivers1 در عنصر endangered_species بر فضای نام عنصر endangered_species اثری نمی‌گذارد.

code.xml

```
<endangered_species xmlns="http://www.
cookwood.com/ns/ end_species1">
<animal>
<name language="English">Giant River
Otter</name>...
<source sectionid="122" newspaperid="21"
contentid="630"/>
...
<rivers:habitat xmlns:rivers=
"http://www.cookwood.com/ns/rivers1">
<rivers:river><rivers:name>Amazon</rivers:name
>
<rivers:source>Andes Mountain in
Peru</rivers:source>...
```

شکل ۷-۸ . چون فضای نام rivers1 در عنصر rivers:habitat اعلان شده است تنها برای عنصر و عناصر تشکیل دهنده آن قابل استفاده است و بر عناصر خارج از عنصر rivers:habitat اثر نمی‌گذارد.

```
code.xml
<endangered_species
    xmlns="http://www.cookwood.com/ns/
    end_species/1.0"
xmlns:rivers="http://www.cookwood.com/ns/
    rivers/1.0">
<animal>
<name language="English">Giant River
    Otter</name>
<name language="Latin">pteronura
    brasiliensis</name>
<threats><threat>habitat destruction
<rivers:name>Amazon</rivers:name>
<rivers:source>Andes Mountains</rivers:source>
</threat>
<threat>hunting</threat>
<threat>overfishing</threat>
<threat>infection by canine distemper
    virus</threat>
</threats>
<source sectionid="122" newspaperid="21"
    contentid="630"/>
<picture filename="otter.jpg" x="200" y="197"/>
<rivers:picture file="amazon.jpg" x="200"
    y="197"/>
<weight>60 pounds</weight>
<length>8 feet long</length>
<diet><rivers:fauna>fish,
    crustaceans</rivers:fauna></diet>
...
</animal>
```

شکل ۸-۸ . هر عنصر متعلق به فضای نام rivers1 با rivers: مشخص می‌گردد. تایپ مکرر rivers: گاهی وقت‌گیر و خسته کننده می‌شود و در مورد تایپ مکرر عبارتهایی شبیه عبارت زیر وضع از این هم بدتر می‌شود: http://www.cookwood.com/ns/rivers1 .

پس از اعلان یک پیشوند و فضای نام مربوط به آن می‌توان آن پیشوند را برای عناصر جداگانه با فضاهای نام گوناگون در سند XML مورد استفاده قرار داد. این امر اثری بر عناصر فرزند عناصر مذکور نخواهد گذاشت.

به منظور تخصیص فضاهای نام گوناگون برای عناصر جداگانه :

۱- برای آغاز اعلان عنصر علامت < را تایپ کنید.

۲- سپس عبارت **prefix:** را تایپ نمایید. prefix مشخصه فضای نام عنصری است که اعلان خواهید کرد.

۳- واژه **element** را تایپ کنید. element نام عنصری است که می‌خواهید به کار برید.

۴- اعلان عنصر را کامل نمایید. با توجه به وجود یا فقدان ویژگی از برچسب پایان < یا /> استفاده کنید (فصل اول).

نکته‌ها

- عنصری به یک فضای نام مربوط می‌گردد که پیشوند آن با پیشوند فضای نام یکسان باشد.
- اگر فضای نام پیش فرض (صفحه ۱۱۵) وجود داشته باشد، پردازشگر XML، عناصر بدون پیشوند را به آن نسبت می‌دهد.
- یک پردازشگر XML پیشوند را به عنوان بخشی از نام عنصر در نظر می‌گیرد. بنابراین باید چسبهای شروع و پایان یکسان باشند. یعنی اگر برچسب شروع <rivers:source> است برای برچسب پایان نیز از </rivers:source> استفاده نمود.
- اگر عناصر و ویژگیهای یک قسمت از یک فضای نام استفاده می‌کنند بهتر است برای بالاترین عنصر آن قسمت یک فضای نام پیش فرض تعریف کرد (صفحه ۱۱۵).

تأثیر فضاهای نام بر ویژگیها

یک ویژگی را به یک فضای نام خاص با پیشوندی مناسب می‌توان نسبت داد ولی ضرورتی ندارد؛ زیرا ویژگیها توسط عناصری که شامل آنها هستند منحصر به فرد می‌شوند.

برای مثال هنگامی که ویژگی sectionid در عنصر source دیده می‌شود یعنی به عنصر source که متعلق به فضای نام end_species1 است تعلق دارد. اگر در عنصر orange نیز ویژگی sectionid ببینید بی هیچ تردیدی و بدون نیاز به توضیح بیشتر در می‌یابید که این ویژگی به عنصر orange تعلق دارد؛ زیرا در عنصر orange قرار دارد.

فضاهای نام پیش فرض برای ویژگیها کاربردی ندارند. به ندرت اتفاق می‌افتد که ویژگیها پیشوند داشته باشند. اگر ویژگی پیشوند نداشته باشد فضای نامی برای آن در نظر گرفته نمی‌شود و به صورت محلی توسط عناصر در بر گیرنده معرفی می‌گردد.

در حقیقت موقعیت ویژگیها در عناصر محلی تفاوت زیادی ندارد. بیشتر توسط عناصری که شامل آنهاست و گاهی با یک فضای نام معرفی می‌گردند.

code.xml

```
<source sectionid="122" newspaperid="21"
contentid="630"/>

<orange sectionid="3A"/>
```

شکل۹-۸ . عناصر source و orange هر دو دارای ویژگی sectionid هستند؛ ولی هیچ مشکل و تداخلی به‌وجود نمی‌آید؛ زیرا هریک توسط عنصر جداگانه‌ای معرفی شده‌اند. در نتیجه معرفی آنها به فضای نام نیاز ندارد.

فضاهای نام، DTDها و سندهای معتبر

اگر چه شما یک عنصر را تنها با نام آن تصور می‌کنید؛ ولی پردازشگر XML عناصر را به صورت prefix:element در نظر می‌گیرد. برای مقایسه سند با یک DTD باید تمام عناصر پیشوندی را در DTD اعلان کنید.

همچنین اگر یک ویژگی با یک فضای نام معرفی شده باید آن را با عبارت xmlns یا xmlns:prefix مشخص نمود.

به دلیل عدم پشتیبانی مستقیم از فضاهای نام، DTDها به الگوهایی که در XML Schema نوشته می‌شوند ترجیح داده می‌شوند.

برای به‌دست آوردن اطلاعات بیشتر درباره DTDها به فصل دوم (صفحه ۳۳) مراجعه کنید.

۹

فضاهای نام، الگوها و معتبرسازی

در فصل ۶ (*تعریف نوعهای ساده*) و فصل ۷ (*تعریف نوعهای پیچیده*) آموختید که چگونه الگویی تهیه کنید که عنصرها و ویژگیهای سند XML را مشخص کند. با این روش یک رده از سندهای XML را می‌توان تهیه نمود.

هر یک از این سندهای XML را - که نمونه یا Instance نامیده می‌شوند - می‌توان با الگو مقایسه نمود تا درستی آنها تایید شود (صفحه ۷۶). یکپارچه بودن اطلاعات با این کار تضمین می‌گردد.

در صورتی که سند XML از فضاهای‌نام مختلفی ساخته شده باشد عمل معتبرسازی آن پیچیده‌تر می‌گردد. جزئیات این مطلب در فصل ۸ (*استفاده ازفضاهای نام در XML*) توضیح داده شده است.

در این فصل به تشریح عملیات معتبرسازی می‌پردازیم (برای توضیحات بیشتر در مورد نحوه استفاده از معتبرساز الگو به *مبحث معتبرسازی XML با استفاده از الگو* در صفحه ۲۴۵ رجوع کنید).

الگوها و فضاهای‌نام

همان‌گونه که گفته شد، فضای‌نام مجموعه‌ای از عنصرها و ویژگیهای مرتبط با یکدیگر است که به وسیله یک نام مشخص می‌گردد. این نام شبیه به URL می‌باشد. معمولا فضاهای‌نام برای مشخص کردن عنصرهایی به کار می‌روند که به صورت سراسری تعریف شده‌اند و علاوه بر آن نامهایی شبیه به یکدیگر نیز دارند (به طور کلی اجزایی که به صورت محلی تعریف می‌شوند به وسیله محتوای خود از یکدیگر متمایز می‌شوند). نام فضای‌نام منحصر به فرد است، نام فضای‌نام و نام عنصر مورد نظر نیز باید منحصر به فرد باشند.

هنگامی که سندی را که عنصرهایش با یک فضای‌نام مشخص شده‌اند معتبرسازی می‌کنید، باید بدانید که یک عنصر از آن منحصر به فرد است و یا خیر. هم‌چنین باید از نحوه تعریف آن، این که در چه زمانی و چند بار باید تکرار شود و دارای چه عنصرهای دیگری باید باشد و. . . اطلاع داشته باشید. تمامی این اطلاعات در یک الگو موجود می‌باشند، عنصرهای فضای‌نام نیز در این الگو تعریف شده‌اند.

الگو می‌تواند تعریف کند که یک سند XML چگونه باید باشد. هم‌چنین می‌تواند به طور هم‌زمان با عنصرهای خود (که به وسیله این الگو تعریف می‌شوند) یک فضای‌نام ایجاد کند و یا آن را توسعه دهد. بعد از اینکه عنصرهای یک الگو به یک فضای‌نام خاص مرتبط شدند، عنصرهایی با پیشوند آن فضای‌نام در سند قابل استفاده‌اند. این سندها به وسیله الگوی مربوط قابل معتبرسازی هستند.

توسعه یک فضای‌نام

می‌توان اجزایی از الگو را که به صورت سراسری تعریف شده‌اند (یا به عبارت دیگر در بالاترین مرتبه می‌باشند) با یک فضای‌نام مرتبط کرد. بدین وسیله می‌توان از این اجزاء در سندهای الگوهای دیگر استفاده نمود. جزئی که به صورت سراسری تعریف شده باشد فرزند مستقیمی از عنصر xsd:schema است. این جزء می‌تواند تعریفهای عنصرها، تعریفهای ویژگیها، تعریفهای نوعهای ساده و یا پیچیده، تعریفهای گروههای نام‌گذاری شده و یا تعریفهای گروههای ویژگی باشد.

با این که احتیاجی به انجام کار خاصی در قسمت تعریفها نمی‌باشد اما باید فضای‌نام هدف را که اجزا با آن مرتبط می‌شوند مشخص نمود.

برای مشخص کردن فضای‌نام هدف :

در عنصر ریشه سند الگوی خود عبارت targetNamespace="URL" را تایپ کنید. URL آدرس فضای‌نامی است که می‌خواهید اجزای تعریف شده در این الگو را با آن مرتبط کنید. به این کار توسعه یک فضای‌نام می‌گویند.

نکته

- تنها اجزایی که به صورت سراسری تعریف شده باشند (به عبارت دیگر در بالاترین مرتبه باشند) با فضای‌نام مرتبط می‌شوند. البته این امر به آن معنی نیست که شما نمی‌توانید از قسمتهایی که به صورت محلی تعریف شده‌اند (و از این رو با فضای‌نام مرتبط نشده‌اند) استفاده کنید و یا آنها را معتبرسازی کنید (مانند عنصر animal در **شکل ۱-۹**). هنگامی که ما یک نمونه سند XML را با الگویی که در شکل نشان داده شده معتبرسازی می‌کنیم، پردازشگر به طور خودکار می‌داند در کجا به دنبال تعریف عنصر endangered_species بگردد. هم‌چنین می‌داند که این عنصر دارای عنصر تایید نشده animal است (عنصری که با یک فضای‌نام تایید نشده است).

code.xsd

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/
    2000/10/XMLSchema"
targetNamespace="http://www.cookwood.com/
    ns/end_species1">
<xsd:element name="endangered_species">
    <xsd:complexType>
    <xsd:sequence>
    <xsd:element name="animal"
    type="animalType"
    maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
    </xsd:element>
<xsd:complexType name="animalType">
    <xsd:sequence>
    <xsd:element name="name" type="nameType"
    minOccurs="2"/>
    <xsd:element name="threats"
    type="threatsType"/>
    <xsd:element name="weight"
    type="xsd:string" minOccurs="0"
    maxOccurs="1"/>
    ...
    </xsd:sequence>
    </xsd:complexType>
...
```

شکل ۱-۹ :در این مثال ، عنصر endangered_species و نوع پیچیده animalType با فضای‌نام http://www.cookwood.com/ns/end_species1 مشخص شده‌اند.عنصرهای animal ، threats و weight با فضای‌نام مرتبط نشده‌اند زیرا در بالاترین مرتبه نیستند.

اضافه کردن تمامی عنصرهایی که به صورت محلی تعریف شده‌اند

به صورت پیش‌فرض، تنها عنصرهایی که به صورت سراسری تعریف شده باشند (و در بالاترین مرتبه باشند) با فضای‌نام هدف مرتبط می‌شوند. در صورتی که بخواهید تعریفهای عنصرهای محلی را (آنهایی که در یک یا چند مرتبه پایینتر قرار دارند) اضافه کنید می‌توانید تمام عنصرها را یکباره اضافه کنید، یا تمام تعریفهای ویژگیها را یکباره اضافه کنید (همان‌گونه که در پایین نشان داده شده) و یا عنصرها و ویژگیها را (همان‌گونه که در صفحه ۱۲۵ نشان داده شده است) به صورت جدا از یکدیگر اضافه کنید.

برای این که تمامی عنصرهایی را که به صورت محلی تعریف شده‌اند به فضای‌نام اضافه کنید :

عبارت **elementFormDefault="qualified"** را در عنصر xsd:schema تایپ کنید.

برای این که تمامی ویژگیهایی را که به صورت محلی تعریف شده‌اند به فضای‌نام اضافه کنید :

عبارت **attributeFormDefault="qualified"** را در عنصر xsd:schema را تایپ کنید.

نکته

- همان‌گونه که مشاهده شد، مقدار پیش‌فرض هر یک از این ویژگیها unqualified است. این به آن معنی است که تنها اجزایی که به صورت سراسری (بالاترین مرتبه) تعریف شده باشند با فضای‌نام مرتبط می‌شوند مگر آنکه شما به صورت دیگری این انتخاب را تعیین کنید.

code.xsd

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/
    2000/10/XMLSchema"
targetNamespace="http://www.cookwood.com
    /ns/end_species1"
elementFormDefault="qualified">
<xsd:element name="endangered_species">
    <xsd:complexType>
    <xsd:sequence>
    <xsd:element name="animal"
    type="animalType"
    maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
    </xsd:element>
<xsd:complexType name="animalType">
    <xsd:sequence>
    <xsd:element name="name" type="nameType"
    minOccurs="2"/><xsd:element
    name="threats" type="threatsType"/>
    <xsd:element name="weight"
    type="xsd:string" minOccurs="0"
    maxOccurs="1"/>
    ...
    </xsd:sequence>
    </xsd:complexType>
...
```

شکل ۲-۹ : اکنون عنصرهای animal، threats و weight ،که به صورت محلی تعریف شده‌اند، با فضای‌نام مرتبط شده‌اند. این عنصرها در سند XML قابل استفاده‌اند.

اضافه‌کردن عنصرهای خاصی که به صورت محلی تعریف شده‌اند

ویژگی form برای مشخص کردن تعیین ارتباط یک عنصر خاص، محلی و تک با فضای‌نام هدف کاربرد دارد. این کار بستگی به مقدار پیش فرض ندارد.

برای این که عنصرهای خاصی که به صورت محلی تعریف شده‌اند به فضای‌نام اضافه شوند :

در تعریف عنصر عبارت **form="qualified"** را تایپ کنید.

عنصر بدون توجه به این که در کجا تعریف شده است، با فضای‌نام هدف مرتبط می‌شود.

اگر از ویژگی elementFormDefault="qualified" در عنصر xsd:schema استفاده کرده باشید، می‌توانید از ویژگی form برای جلوگیری از مرتبط شدن یک عنصر خاص، که به صورت محلی تعریف شده است، با فضای‌نام هدف استفاده کنید.

برای جلوگیری از مرتبط شدن یک عنصر خاص ،که به صورت محلی تعریف شده است، با فضای‌نام هدف (بدون توجه به پیش‌فرض) :

در تعریف عنصر عبارت **form="unqualified"** را تایپ کنید.

code.xsd

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/
    2000/10/XMLSchema"
targetNamespace="http://www.cookwood.com
    /ns/end_species1"
elementFormDefault="qualified">
<xsd:element name="endangered_species">
    <xsd:complexType>
    <xsd:sequence>
    <xsd:element name="animal"
    type="animalType"
    maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
    </xsd:element>
<xsd:complexType name="animalType">
    <xsd:sequence>
    <xsd:element name="name" type="nameType"
    minOccurs="2"/>
    <xsd:element name="threats"
    type="threatsType" form="unqualified"/>
    <xsd:element name="weight"
    type="xsd:string" minOccurs="0"
    maxOccurs="1" form="unqualified"/>
    ...
    </xsd:sequence>
    </xsd:complexType>
...
```

شکل ۳-۹ : بدون توجه به تنظیم elementFormDefault در عنصر xsd:schema هیچ‌یک از عناصر weight و threats با فضای‌نام هدف مرتبط نخواهند شد. این کار به دلیل آن است که ویژگی form آنها بر مقدار elementFormDefault برتری دارد.

ارجاع اجزا به فضاهای‌نام

بعد از این که اجزای الگو (نوعهای ساده و یا پیچیده، عنصرها، ویژگیها، گروههای ویژگی و گروههای نام) با یک فضای‌نام مرتبط شدند می‌توان درون همان الگو و یا در الگوهای دیگر به آنها ارجاع کرد. به دلیل این که این اجزا با یک فضای‌نام مرتبط شده‌اند، برای ارجاع به آنان باید فضای‌نام نیز مشخص گردد.

در این کتاب، از پسوند xsd برای تمامی عنصرها XML Schema استفاده شده است. همچنین الگوی XML Schema (که الگوی الگوها نامیده می‌شود) در عنصر ریشه الگو تعریف شده است. به همین دلیل وقتی به نوعهای موجود (نوعهایی که در الگوی الگوها موجودند) ارجاع داده می‌شود، باید از پسوند xsd استفاده شود. به این طریق، الگو متوجه می‌شود که تعاریف را در کجا بیابد.

ارجاع به نوعهای تعریف شده توسط کاربر(ساده و یا پیچیده)، عنصرها، ویژگیها و یا گروهها کمی پیچیده‌تر است. ارجاع به هر یک از موارد فوق باید حاوی اطلاعاتی درباره فضای‌نام مربوط باشد.

برای مشخص کردن یک فضای‌نام پیش‌فرض برای اجزای مورد ارجاع و ارجاع کردن به آنها در الگو :

۱- در عنصر ریشه سند الگو عبارت **xmlns="URL"** را تایپ کنید. URL آدرس فضای‌نامی است که اجزای مورد ارجاع با آن مرتبط شده‌اند.

۲- در قسمت مقدار ویژگیهای type و ref مقدار reference (بدون پیشوند) را قرار دهید. reference نام جزئی است که با فضای‌نام پیش‌فرض مرتبط شده است.

code.xsd

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/
    2000/10/XMLSchema"
targetNamespace="http://www.cookwood.com/
    ns/end_species1"
xmlns="http://www.cookwood.com/
    ns/end_species1">
<xsd:element name="endangered_species">
    <xsd:complexType>
    <xsd:sequence>
    <xsd:element name="animal"
    type="animalType"
    maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
    </xsd:element>
<xsd:complexType name="animalType">
    <xsd:sequence>
    <xsd:element name="name" type="nameType"
    minOccurs="2"/>
    <xsd:element name="threats"
    type="threatsType"/>
    <xsd:element name="weight"
    type="xsd:string" minOccurs="0"
    maxOccurs="1"/>
    ...
    </xsd:sequence>
    </xsd:complexType>
...
```

شکل ٤-۹ : برای این سند الگو، فضای‌نام پیش‌فرض با نام end_species1 را در عنصر پایه الگو تعریف کرده‌ایم. این مطلب به معنی آن است که تعریفهای مربوط به نوعهای بدون پیشوند (animalType ، nameType و threatsType) در الگوی مرتبط با فضای‌نام end_species1 قرار دارند (تعریفهای مربوط به نوعهای با پیشوند در الگوی مرتبط با فضای‌نام XMLSchema هستند).

```
code.xsd
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/
    2000/10/XMLSchema"
targetNamespace="http://www.cookwood.com/
    ns/end_species1"
xmlns:end="http://www.cookwood.com/
    ns/end_species1">
<xsd:element name="endangered_species">
    <xsd:complexType>
    <xsd:sequence>
    <xsd:element name="animal"
    type="end:animalType"
    maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
    </xsd:element>
<xsd:complexType name="animalType">
    <xsd:sequence>
    <xsd:element name="name"
    type="end:nameType" minOccurs="2"/>
    <xsd:element name="threats"
    type="end:threatsType"/>
    <xsd:element name="weight"
    type="xsd:string" minOccurs="0"
    maxOccurs="1"/>
    ...
    </xsd:sequence>
    </xsd:complexType>
...
```

شکل ۵-۹: این سند با شکل ۴-۹ معادل است. در اینجا، از پیشوند end برای فضای‌نام end_species استفاده شده است. پس باید از پیشوند استفاده شود تا مشخص گردد که end:animalType ، end:nameType و end:threatsType به فضای‌نام end_species1 تعلق دارند. با وجود اینکه عنصرها به همان فضای‌نام تعلق دارند (که توسط ویژگی targetNamespace تعیین شده است) نحوه نمایش آنها تغییر کرده است.

برای تعریف یک فضای‌نام با یک پیشوند و استفاده از آن پیشوند برای مشخص کردن اجزای مورد ارجاع در الگو:

۱- در عنصر اصلی سند XML خود عبارت **xmlns:prefix="URL"** را تایپ کنید. prefix نامی است که برای مشخص کردن تعاریف در این الگو استفاده می‌شود. این تعاریف به فضای‌نامی که توسط URL مشخص شده است تعلق دارند.

۲- در قسمت مقدار ویژگیهای type و ref مقدار prefix:reference را قرار دهید که در آن reference نام جزئی است که با فضای‌نام مربوط به آن پیشوند مرتبط شده است (این پیشوند همان پیشوند مرحله ۱ است).

نکته‌ها

- شما می‌توانید به هر تعداد که لازم دارید فضای‌نام تعریف کنید و از تعریفهای درون آنها به وسیله پیشوندهای مربوط استفاده کنید.
- مقدار یک ویژگی باید با یک فضای‌نام مرتبط باشد (و یا با هیچ فضای‌نامی مرتبط نباشد).
- این تفاوت مهم را به خاطر داشته باشید : فضای‌نام هدف، فضای‌نامی است که به طور فعال و واقعی با یک جزء از فضای‌نام مرتبط می‌شود. یک تعریف فضای‌نام مانند xmlns="URL" یا xmlns:end="URL" فضای‌نامی را مشخص می‌کند که یک جزء با آن مرتبط شده است.

الگوی الگوها به عنوان پیش‌فرض

اجباری به استفاده از پیشوند xsd برای عنصرها و نوعهای الگوی الگوها وجود ندارد بلکه از هر پیشوند دیگری نیز می‌توان استفاده نمود. اگر الگو بیشتر از نوعهای پیش‌ساخته استفاده می‌کند، استفاده از الگوی الگوها به عنوان فضای‌نام پیش‌فرض مناسبتر خواهد بود؛ با این کار احتیاجی به استفاده از پیشوند برای فضای‌نام مربوط نمی‌باشد.

برای تعریف الگوی الگوها به عنوان فضای‌نام پیش‌فرض یک الگو:

۱- برای شروع الگو، بعد از تعریف XML عبارت schema>را تایپ کنید (دقت کنید که از xsd:schema استفاده نشد).

۲- الگوی الگوها به عنوان فضای‌نام پیش‌فرض الگوی خود را با این دستور تعریف کنید :
xmlns="http://www.w3-org/2000/10/XMLSchema"

۳- عبارت **targetNamespace="URL"** را تایپ کنید. URL فضای‌نامی است که مایلید تعریفهای نوعها و عنصرهای موجود در بالاترین مرتبه را با آن مرتبط کنید (این قسمت مانند قسمتهای گذشته می‌باشد).

۴- عبارت **xmlns:prefix="URL"** را تایپ کنید. prefix نامی است که برای مشخص کردن تعاریفی در این الگو استفاده می‌شود که به فضای‌نامی که توسط URL مشخص شده است تعلق دارند.

۵- مرحله ۴ را برای تمامی فضای‌نامهایی که دارای تعریفهایی هستند که در این الگو استفاده شده است تکرار کنید.

۶- برای کامل شدن این قسمت از الگو یک< تایپ کنید.

نکته

- معمولا نیاز به تخصیص یک پیشوند برای فضای‌نام هدف وجود دارد (مگر اینکه این تعریفها را از یک فضای‌نام دیگر استخراج کنید (صفحه ۱۳۲).

code.xsd

```
<?xml version="1.0" ?>
<schema xmlns="http://www.w3.org/
    2000/10/XMLSchema"
targetNamespace="http://www.cookwood.com/
    ns/end_species1"
xmlns:end="http://www.cookwood.com/
    ns/end_species1">
<element name="endangered_species">
    <complexType>
    <sequence>
    <element name="animal"
    type="end:animalType"
    maxOccurs="unbounded"/>
    </sequence>
    </complexType>
    </element>
<complexType name="animalType">
    <sequence>
    <element name="name"
    type="end:nameType" minOccurs="2"/>
    <element name="threats"
    type="end:threatsType"/>
    <element name="weight" type="string"
    minOccurs="0" maxOccurs="1"/>
    ...
    </sequence>
    </complexType>
...
```

شکل ۶-۹ : این سند کاملا معادل شکل ۴-۹ است. تنها تفاوت در نشانه‌گذاری است. در اینجا الگوی الگوها به عنوان فضای‌نام پیش‌فرض تعیین شده است و از این رو احتیاجی به استفاده از پیشوند در مورد نامهای عنصرها (schema ، element ، complexType و غیره) و نوعهای پیش‌ساخته شده (رشته‌ها) توسط xsd وجود ندارد. اما باید برای هر نوعی که مربوط به فضای‌نام دیگری باشد از پیشوند استفاده کرد (مانند end:animalType ، end:nameType و end:threatsType).

فضاهای نام و معتبرسازی XML

تا این مرحله شما تعدادی اجزا را با استفاده از یک فضای‌نام کنترل کردید و اکنون برای معتبرسازی سندی که از این اجزاء استفاده می‌کند آماده هستید. در فصلهای ۶ و ۷ در هنگام معتبرسازی احتیاجی به توجه به فضاهای‌نام نبود زیرا در آن سندها هیچ یک از اجزای تعریف شده را با فضای‌نامی مرتبط نکردید. اکنون که تمامی و یا قسمتی از اجزای شما به یک فضای‌نام تعلق دارند، باید در هنگام معتبرسازی سندهای خود آن فضای‌نام را نیز مشخص کنید.

ممکن است نیاز باشد محلی را مشخص کنید که در آن پردازشگر، الگوی حاوی تعریفهای اجزای مرتبط با فضای‌نام را بیابد.

نوشتن سندهای معتبر XML با استفاده از اجزای کنترل شده:

۱- ابتدا باید فضای‌نام اجزای مورد نظر مشخص شوند. این کار را می‌توان به یکی از طرق زیر انجام داد : با تعریف یک فضای‌نام پیش‌فرض (xmlns="URL") و استفاده نکردن از پیشوند برای اجزا یا با تعریف یک فضای‌نام با پیشوند (xmlns:prefix="URL") و استفاده از پیشوند برای اجزای مورد نظر. توضیحات بیشتر در فصل ۸ (*استفاده از فضاهای‌نام در XML*) آمده است.

۲- گاهی اوقات لازم است آدرس الگویی را تعیین کنیم که فضای‌نام مورد استفاده برای کنترل کردن عنصرها را تعریف و یا توسعه می‌دهد.

نکته

- به خاطر داشته باشید تنها عنصرهایی را که با فضای‌نام مرتبط شده‌اند کنترل کنید.

code.xml

```
<?xml version="1.0" ?>
<end:endangered_species
    xmlns:end="http://www.cookwood.com/ns/
    end_species1">
<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera tigris</name>
<threats>
    <threat>poachers</threat>
    <threat>habitat destruction</threat>
    <threat>trade in tiger bones for traditional
    Chinese medicine (TCM)</threat>
    </threats>
<weight>500 pounds</weight>
<length>3 yards from nose to tail</length>
...
</end:endangered_species>
```

شکل ۷-۹ : در الگوی نشان داده شده در **شکل ۴-۹** عنصر endangered_species با فضای‌نام http://www.cookwood.com/ns/end_species1 مرتبط شده بود (به دلیل targetNamespace موجود در الگو). در صورتی که بخواهیم سندی که حاوی این عنصر باشد معتبرسازی کنیم، باید endangered_species را کنترل کنیم. از آنجا که animal با فضای‌نام هدف مرتبط نیست، به این کار احتیاج ندارد.

مشخص کردن آدرس الگو

در فصلهای گذشته با چگونگی مشخص کردن آدرس یک الگو زمانی که عنصرهای کنترل شده در سند XML وجود نداشته باشد آشنا شدید (صفحه ۷۳). مشخص کردن آدرس یک الگو برای سندی از XML که دارای عنصرهای کنترل شده باشد نیز کاملا مشابه است.

برای مشخص کردن آدرس الگو برای عنصرهای کنترل شده :

۱- در عنصر ریشه سند XML و بعد از تعریف فضای‌نامی که با عنصرهای کنترل شده مرتبط شده است عبارت

xmlns:xsi="http://www.w3-org/2000/10/XMLSchema-instance"

را تایپ کنید. این کار برای تعریف فضای‌نامی است که به وسیله آن آدرس الگو مشخص می‌شود.

۲- سپس عبارت **xsi:schemaLocation="URL** را تایپ کنید. URL نام فضای‌نامی است که شما در مرحله بعد آدرس الگوی آن را می‌دهید (دقت کنید که کوتیشن بسته وجود ندارد).

۳- یک فاصله و یا return تایپ کنید.

۴- سپس عبارت **file.xsd"** را تایپ کنید. file.xsd آدرس URL یک فایل اصلی می‌باشد که حاوی الگوی تعریف‌کننده فضای‌نام مورد استفاده در این سند است (دقت کنید که کوتیشن باز وجود ندارد).

۵- برای هر تعداد الگوی به کار رفته در سند مراحل ۲ تا ۴ را تکرار کنید.

نکته

- بیشتر پردازشگرهای سندهای XML برای پیدا کردن فایل الگو از ویژگی xsi:schemaLocation استفاده می‌کنند. برای مطمئن شدن از این موضوع به قسمت راهنمای پردازشگر خود مراجعه کنید.

code.xml

```
<?xml version="1.0" ?>
<end:endangered_species
    xmlns:end="http://www.cookwood.com/ns/
    end_species1"
xmlns:xsi="http://www.w3.org/2000/10/
    XMLSchema-instance"
xsi:schemaLocation="http://www.cookwood.com/
    ns/end_species1
http://www.cookwood.com/xml/schemas/
    end_species1.xsd">
<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera tigris</name>
<threats>
    <threat>poachers</threat>
    <threat>habitat destruction</threat>
    <threat>trade in tiger bones for traditional
    Chinese medicine (TCM)</threat>
    </threats>
<weight>500 pounds</weight>
<length>3 yards from nose to tail</length>
...
```

شکل ۸-۹: اولین سطری که مشخص شده است فضای‌نام مربوط به اشیایی را که از پیشوند xsi استفاده می‌کنند تعریف می‌کند. سطر سوم قسمت دومی که مشخص شده است فایلی بر روی سرور را مشخص می‌کند که فضای‌نام درون آن تعریف شده است.

الگوها در فایلهای مختلف

برای استفاده از اجزای یک الگو در چندین الگو یا سند و یا برای استفاده بهتر از یک الگوی بزرگ می‌توان اجزای آن را به چند فایل مختلف تقسیم کرد.

برای اضافه کردن اجزای الگو :

۱- اجزای الگو را در چند فایل قسمت کنید. هر فایل باید به شکل متنی و دارای پسوند xsd. باشد.

۲- در الگویی که می‌خواهید اجزایی را اضافه کنید، عبارت

<xsd:include schemaLocation="included-file.xsd"/> را تایپ کنید. includedfile.xsd آدرس URL سند الگویی است که می‌خواهید اجزایی از آن را اضافه کنید.

نکته‌ها

- ویژگی targetNamespace فایل الگویی که اضافه می‌شود باید با فایل الگویی که آن را دریافت می‌کند یکسان باشد. برای اضافه‌کردن اجزای الگویی که فضاهای‌نام هدف متفاوتی داشته باشند به قسمت واردکردن اجزا در صفحه ۱۳۲ مراجعه کنید.
- در صورتی که فضای‌نام اضافه‌شده دارای فضای‌نام هدف نباشد، فرض می‌شود که فضای‌نام هدف آن با فضای‌نام سندی که به آن اضافه می‌شود یکسان است.

code.xsd

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/
    2000/10/XMLSchema"
targetNamespace="http://www.cookwood.com/
    ns/end_species1">
<xsd:complexType name="threatsType">
    <xsd:sequence>
    <xsd:element name="threat"
    maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
```

شکل ۹-۹ : ابتدا یک فایل جدید با تعریف نوع پیچیده threatsType ایجاد می‌کنیم(threats.xsd که در شکل نشان داده شده است) . حال می‌توانیم از این تعریف در الگوهای دیگر استفاده کنیم.

code.xsd

```
<?xml version="1.0" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/
    2000/10/XMLSchema"
targetNamespace="http://www.cookwood.com/
    ns/end_species1"
xmlns="http://www.cookwood.com/
    ns/end_species1">
<xsd:include schemaLocation=
    "http://www.cookwood.com/xml/schemas/
    threats.xsd"/>
<xsd:element name="endangered_species">
    <xsd:complexType><xsd:sequence>
    <xsd:element name="animal"
    type="animalType"
    maxOccurs="unbounded"/>
    </xsd:sequence></xsd:complexType>
    </xsd:element>
<xsd:complexType name="animalType">
    <xsd:sequence>
    <xsd:element name="name" type="nameType"
    minOccurs="2"/>
    <xsd:element name="threats"
    type="threatsType"/>
...</xsd:complexType>
```

شکل ۱۰-۹ : با استفاده از عنصر xsd:include می‌توان اجزای الگو را از الگوهای دیگر با فضای‌نام یکسان وارد کرد.

واردکردن اجزا

وقتی بخواهید سندهایی از XML را که عنصرهایشان با چندین فضای‌نام مرتبط شده باشد معتبرسازی کنید می‌توانید اجزایی از الگو را که در بالاترین مرتبه باشند از الگوهای دیگر با فضاهای‌نام هدف متفاوت وارد کنید.

برای وارد کردن اجزا از الگوهایی که دارای فضای‌نام هدف متفاوت باشند :

۱- بلافاصله بعد از عنصر xsd:schema در سند الگویی که الگوهای دیگر را به آن وارد می‌کنید عبارت **<xsd:import** را تایپ کنید.

۲- سپس عبارت **namespace="URL"** را تایپ کنید که در آن URL مشخص‌کننده نام فضای‌نامی است که اجزای الگوی واردشده با آن مرتبط شده‌اند.

۳- سپس در صورت لزوم، آدرس فایل حاوی الگوی تعریف کننده فضای‌نام مرحله ۲ را با تایپ کردن عبارت **xsi:schemaLocation="URL"** وارد کنید. URL نام فضای‌نامی است که شما می‌خواهید سندهای الگوی آن را آدرس بدهید.

۴- یک فاصله و یا return تایپ کنید.

۵- اکنون عبارت **file.xsd"** را تایپ کنید.file.xsd آدرس URL فایلی است که حاوی الگوی مورد نظر برای وارد کردن اجزا از آن است.

۶- برای کامل کردن برچسب xsd:import یک /> تایپ کنید.

۷- همان‌گونه که در صفحه ۱۲۶ توضیح داده شد، یک پیشوند برای فضای‌نام واردشده تعیین کنید تا بتوانید به اجزای وارد شده در الگوی خود ارجاع کنید.

نکته

- در صورتی که تا این مرحله تعریف نمونه الگوی XML را انجام نداده‌اید، باید این کار را همان‌گونه که در مرحله ۱ صفحه ۱۳۰ توضیح داده شده است انجام دهید.

code.xsd

```
<?xml version="1.0" ?>

<xsd:schema xmlns:xsd="http://www.w3.org/
    2000/10/XMLSchema"
    xmlns:xsi="http://www.w3.org/2000/10/
    XMLSchema-instance"
    targetNamespace="http://www.cookwood.
    com/ns/end_species1"
    xmlns="http://www.cookwood.com/
    ns/end_species1"
    xmlns:rivers="http://www.cookwood.com/
    ns/rivers1">

<xsd:import namespace=
    "http://www.cookwood.com/ns/rivers1"
    xsi:schemaLocation=
    "http://www.cookwood.com/ns/rivers1
    http://www.cookwood.com/xml/schemas/
    rivers.xsd"/>

<xsd:element name="endangered_species">
    <xsd:complexType><xsd:sequence>
    <xsd:element name="animal"
    type="animalType"
    maxOccurs="unbounded"/>
    </xsd:sequence></xsd:complexType>
    </xsd:element>

<xsd:complexType name="animalType">
    <xsd:sequence><xsd:element name="name"
    type="nameType" minOccurs="2"/>
    <xsd:element name="threats"
    type="threatsType"/>
    ...
    <xsd:element name="habitat"
```

شکل ۱۱-۹ : با استفاده از وارد کردن یک الگو در الگوی دیگر می‌توان از اجزای الگوی اول برای تعریفهای اجزای الگوی دوم استفاده کرد.

۱۰

XSLT

متن کامل و رسمی برای تبدیل و قالب‌بندی سندهای XML ابتدا در تعریفی به‌نام XSL که مخفف Extensible Style Language است عرضه شد. به دلیل این که برای کامل شدن XSL وقت زیادی گرفته می‌شد، W3C، XSL را به دو بخش تقسیم کرد : XSLT (برای تبدیل‌کردن) و XSL-FO (برای قالب‌بندی اشیا).

این فصل و دو فصل بعد از آن استفاده از XSLT و تبدیل کردن سندهای XML را توضیح می‌دهند. نتیجه نهایی، یک سند XML دیگر و یا به طور عمومی‌تر یک سند HTML خواهد بود که در مرورگرهای قدیمی و جدید قابل دیدن باشد. تبدیل‌کردن یک سند XML به معنی آن است که محتوای فایل بررسی شود و با توجه به عنصرهای موجود در آن اعمال خاصی انجام شود. از XSLT در زمینه‌هایی نظیر مرتب کردن خروجی بر اساس موضوع و یا نشان‌دادن تنها قسمتهای خاصی از اطلاعات و بسیاری از موارد دیگر استفاده می‌شود.

به دلیل این که XSL-FO هنوز به طور کامل رسمی نشده است، در این کتاب مورد بررسی قرار نگرفته است. با رفتن به آدرس http://www.w3-org/Style/XSL می‌توانید اطلاعات بیشتری در مورد وضعیت فعلی XSL-FO کسب کنید. در حال حاضرXSLT معمولا همراه با CSS که مخفف Cascading Style Sheets است و عمل قالب‌بندی را انجام می‌دهد، استفاده می‌شود. CSS به صورت گسترده‌تری مورد استفاده قرار می‌گیرد. برای اطلاعات بیشتر در مورد آن می‌توانید به قسمت ۵ در صفحه ۱۷۵ رجوع کنید.

مثالهای این قسمت از کتاب بر پایه یک فایل XML و یک دسته فایلهای XSLT ترتیبی است که هریک فایل قبلی را می‌سازد. پیشنهاد می‌شود حداقل فایلهای XML را از اینترنت دریافت و استفاده کنید (به صفحه ۱۸ مراجعه کنید).

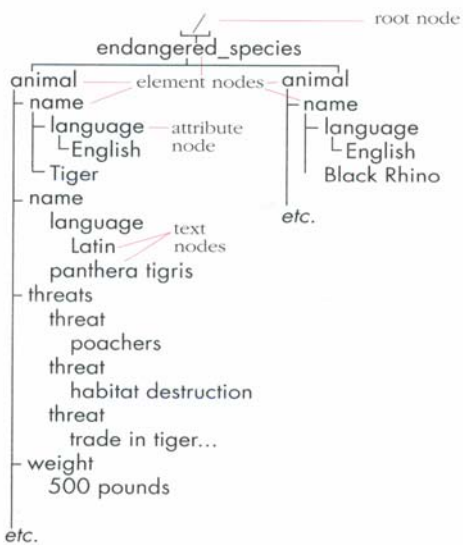
تبدیل XML با استفاده از XSLT

این قسمت با دورنمایی از کل چرخه تبدیل شروع می‌شود.
برای انجام تبدیل، شما به یک پردازشگر XSLT احتیاج دارید. بر روی اینترنت برنامه‌های زیادی برای این کار وجود دارد. یکی از این برنامه‌ها Instant Saxon (نوشته مایکل کی) است. برای اطلاعات بیشتر به بخش تبدیل‌کردن XML با استفاده از یک پردازشگر XSLT در صفحه ۲۴۶ رجوع کنید.
اولین کاری که یک پردازشگر XSLT می‌کند بررسی سند XML (**شکل ۱-۱۰**) و سپس تبدیل آن به یک درخت گره است(**شکل ۲-۱۰**). یک گره قسمتی از سند XML است که می‌تواند یک عنصر، یک ویژگی و یا مقداری متن باشد. یک درخت گره نمودار سلسله‌وار تمامی سند XML است.
بعد از این که پردازشگر گره‌های یک سند XML را شناسایی کرد، به صفحه سبک XSLT مراجعه می‌کند تا تصمیم بگیرد با این گره‌ها چگونه برخورد کند. این دستورالعملها در قالبهای مختلفی هستند. هر قالب دارای دو قسمت است : اول، یک نوع برچسب که مشخص کند این قالب بر روی چه نوع گره‌هایی اعمال می‌شود، و دوم، دستورالعملهایی درباره چگونگی انجام تبدیل.
پردازشگر به طور خودکار به دنبال یک قالب ریشه می‌گردد که بر روی گره ریشه سند XML (گرهی که خارجی‌ترین عنصر را دارد) اعمال می‌شود. قالب ریشه دارای ترکیبی از عنصرهای لفظی (literal) است که باید به همان صورتی که هستند در خروجی ظاهر شوند. هم‌چنین این قالب دارای دستورالعملهایی ازXSLT است که گره‌های سند اصلی را به خروجی می‌دهند و یا پردازش می‌کنند.

code.xml

```
<?xml version="1.0"?>
<endangered_species>

<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera
tigris</name>
<threats><threat>poachers</threat>
<threat>habitat destruction</threat>
<threat>trade in tiger bones for traditional
Chinese medicine (TCM)</threat>
</threats>

...
```

شکل ۱-۱۰ : می‌توانید سند کامل XML مثالهای این فصل را از سایت وب این کتاب بگیرید(صفحه ۱۸ را ببینید). توصیه می‌شود یک نسخه از این مثالها را دریافت و استفاده کنید.



شکل ۲-۱۰ : در این شکل قسمتی از درخت گره سند نشان داده شده در **شکل ۱-۱۰** نمایش داده شده است.

xsl:apply-templates یکی از دستورالعملهای خاص XSLT است که تعدادی از گره‌ها را مشخص می‌کند (که به آنها یک دسته گره می‌گویند) و تعیین می‌کند که این گره‌ها در آن نقطه با مناسبترین قالبهای موجود پردازش شوند. هر یک از این زیرقالبها می‌توانند دستورهای xsl:apply-templates دیگری نیز داشته باشند که به زیرقالبهای دیگری اشاره کنند. این قابلیت اجازه می‌دهد ترتیب و حالتی که بر اساس آن محتوای سند اصلی پردازش می‌شوند و به خروجی می‌روند کنترل شود.

انتخاب و مشخص کردن دسته‌های گره به وسیله عبارتها و طرحها صورت می‌گیرد. این انتخابها در دستور XPath صورت می‌گیرند و به دلیل طولانی‌بودن مبحث در فصلهای آینده : XPath : عبارتها و طرحها، که در صفحه ۱۵۳ آغاز می‌شود، و عبارتهای آزمون و تابعها، که در صفحه ۱۶۳ آغاز می‌شود مورد بررسی قرار می‌گیرند.

اطلاعاتی که تبدیل شده‌اند، سپس نمایش داده می‌شوند و یا بر روی یک فایل دیگر ذخیره می‌شوند.

اگرچه با استفاده از XSLT می‌توان هر نوع سندی را به هر نوع دیگری تبدیل کرد، در این کتاب تنها به استفاده از XSLT برای تبدیل سندهای XML به HTML می‌پردازیم. این عمل اجازه استفاده همزمان از قابلیتها و انعطاف‌پذیری XML و سازگاری بالای HTML را می‌دهد.

code.xslt

```
<?xml version="1.0"?>
<xsl:stylesheet
xmlns:xsl="http://www.w3.o...
orm" version="1.0">

<xsl:template match="/">
<html><head><title>Endangered
</title></head><body bgcolor="

<xsl:apply-templates
select="endangered_species/animal"/>

</body></html></xsl:template>

<xsl:template match="animal">
<p align="center"><xsl:apply-templates
select="picture"/>

<br/><font size="+3"><xsl:apply-templates
select="name" /></font></p>

<table width="100%" border="2">
<tr><th>Subspecies</th><th>
Number Left</th><th>As Of<

<xsl:for-each select="subspecies">

<xsl:sort select="population" data-
type="number"/>

<xsl:sort select="population/@year" data-
type="number"/>

<tr><td><xsl:apply-templates
select="name"/></td>

<td><xsl:value-of select="region"/></td>
```

root template (always processed)

literals (output as is)

xsl instructions that call other templates

additional templates (that need to be explicitly called)

xsl instructions that process output (and don't call other templates)

شکل ۳-۱۰ : صفحه سبک XSLT به طور کامل در سایت وب کتاب موجود است و می‌توانید آن را از این سایت دریافت کنید.

شروع یک صفحه سبک XSLT

هر صفحه‌سبک XSLT یک سند XML است و از این رو باید با تعریف استاندارد XML آغاز گردد. بعد از این کار باید یک فضای‌نام برای این صفحه‌سبک تعریف کنید.

برای آغاز یک صفحه سبک XSLT :

۱- برای تعسیین این که صفحه‌سبک XSLT شما یک سند XML است **<?xml version="1.0"?>** را تایپ کنید. برای اطلاعات بیشتر به قسمت تعیین نسخه XML در صفحه ۲۴ مراجعه کنید.

۲- سپس برای مشخص‌کردن فضای‌نام برای صفحه‌سبک و تعیین یک پیشوند (xsl) عبارت **<xsl:stylesheet xmlns:xsl="http://www.w3-org/1999/XSL/Transform" version="1- 0">** را تایپ کنید.

۳- چندین سطر خالی برای ساختن صفحه‌سبک (با استفاده از دستورالعملهای این فصل و فصلهای آتی) قرار دهید.

۴- برای کامل‌کردن این صفحه‌سبک عبارت **</xsl:stylesheet>** را تایپ کنید.

نکته‌ها

- دقت داشته باشید در دستور xsl:stylesheet دو لغت style و sheet به یکدیگر متصل می‌باشند.
- در صورتی که برای پردازش XSLT از Internet Explorer نسخه ۵ استفاده می‌کنید، باید برای تعریف فضای‌نام از دستور زیر استفاده کنید

<xsl:stylesheet xmlns:xsl="http://www.w3-org/TR/WD-xsl">

برای اطلاعات بیشتر در مورد تعریف فضاهای‌نام به فصل ۸، استفاده از فضاهای‌نام در XML مراجعه کنید.

```
code.xslt
<?xml version="1.0"?>
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/
Transform" version="1.0">


</xsl:stylesheet>
```

شکل ٤-١٠ : قسمت header یک صفحه سبک معمولا یکسان است.می‌توانید این اطلاعات را از یک صفحه سبک copy و در یک صفحه دیگر paste کنید.

XSLT

ساخت قالب ریشه

اولین چیزی که پردازشگر XSLT در یک صفحه‌سبک به دنبال آن می‌گردد قالبی است که بتواند آن را به گره ریشه سند XML نسبت دهد. این قالب را قالب ریشه می‌نامیم.

برای ساخت قالب ریشه :

۱- عبارت **<xsl:template** را تایپ کنید.

۲- عبارت **match="/"** را تایپ کنید. کاراکتر slash یک طرح است که با گره ریشه سند XML همخوانی دارد.

۳- یک > تایپ کنید.

۴- برای اعمالی که باید در سند XML شما انجام شود چندین سطر فاصله قرار دهید (این قسمت در صفحات ۱۴۰ – ۱۵۱ مورد بررسی قرار می‌گیرد).

۵- برای کامل کردن قالب ریشه عبارت **</xsl:template>** را تایپ کنید.

نکته‌ها

- با این که محل قرار گرفتن این قالب در صفحه‌سبک XSLT شما برای پردازشگر اهمیتی ندارد، در صورتی که آن را در بالای صفحه خود قرار دهید، کارکردن با سند، برای شما و افراد دیگر ساده‌تر خواهد بود.
- در صورتی که در سند خود قالبی که با گره ریشه تطبیق داشته باشد تعیین نکنید، از یک قالب پیش‌ساخته استفاده می‌شود که برای هر یک از گره‌های فرزند گره ریشه به دنبال یک قالب مناسب می‌گردد (قالب ریشه پیش‌ساخته با <xsl:template><xsl:applytemplates/></xsl:template> معادل است). برای اطلاعات بیشتر درباره xsl:apply-templates به قسمت ساخت و اجرای قوانین قالب در صفحه ۱۴۴ مراجعه کنید.

code.xslt

```
<?xml version="1.0"?>
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/
Transform" version="1.0">
<xsl:template match="/">


</xsl:template>
</xsl:stylesheet>
```

شکل ۵-۱۰ : قالب پایه ، تنها قالبی است که به طور خودکار توسط پردازشگر XSLT صدا می‌شود.

code.xml

شکل ۶-۱۰ : در صورتی که یک سند را با این صفحه سبک و یک قالب پایه معمولی پردازش کنید، یک سند خالی بدست خواهید آورد.این چیزی است که قالب **شکل ۵-۱۰** بیان می‌کند : دو خط خالی در خروجی.

کد HTML خروجی

به طور کلی دو نوع جزء در یک صفحه‌سبک XSLT وجود دارد: دستورالعملها و عبارتهای لفظی. دستورالعملهای XSLT تعیین می‌کنند سند XML اصلی چگونه تبدیل شود. عبارتهای لفظی (که به طور معمول عبارات متنی و کدهای HTML هستند) همان‌گونه که در صفحه‌سبک می‌باشند در خروجی ظاهر می‌گردند.

ساختار سند نهایی در قالب ریشه ایجاد می‌شود. برای ایجاد خروجی HTML، باید ساختارهای HTML مانند head و body به سند اضافه‌گردند. در صورت تمایل می‌توان قالب‌بندیهای دیگر HTML را نیز اضافه نمود.

در قالبهایی غیر از قالب ریشه، می‌توان هر نوع قالب‌بندی HTML را که لازم باشد پیاده نمود اما از عنصرهای html و body و head نمی‌توان استفاده نمود.

برای اضافه نمودن کد HTML :

در قسمت دستورهای قاعده قالب (که بین دستور <xsl:template match="..."> و </xsl:template> می‌باشد)، هر دستوری از HTML را که مایلید در هنگام اجرای آن قالب در خروجی ببینید وارد کنید.

code.xslt

```
<xsl:template match="/">
<html><head><title>Endangered
Species</title></head><body bgcolor="white">
<p>Endangered animals face numerous threats.
For more information, check out the World Wildlife
Federation's <a
href="http://www.worldwildlife.org/
species/species.cfm?"> pages</a>.
</p><hr/></body></html>
</xsl:template>
```

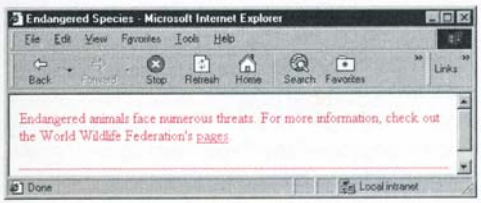
شکل ۱۰-۷ : هر چیزی که جزء دستورالعملهای xsl نباشد به همان صورت در خروجی ظاهر خواهد شد.با این روش می‌توان سادگی کدهای HTML و متنها را اضافه نمود.HTML باید از قوانین خوش فرمی پیروی کند؛ به طور مثال برچسب باید یک برچسب متناظر نیز داشته باشد و همچنین در اینجا از <hr/> به جای <hr> استفاده شده است.

code.html

```
<html><head>
<meta http-equiv="Content-Type"
content="text/html; charset=utf-8">
<title>Endangered Species</title></head>
<body bgcolor="white">
<p>Endangered animals face numerous threats.
For more information, check out the World Wildlife
Federation's <a
href="http://www.worldwildlife.org/species/speci
es.cfm?"> pages</a>.
</p><hr></body></html>
```

شکل ۱۰-۸ : پردازشگر XSLT (در اینجا SAXON) تمامی کدهای HTML و متنها را در خروجی قرار داده است.دقت کنید که SAXON در هنگامی که به برچسب <html> برخورد کند و متوجه شود که خروجی کد HTML است ، به طور خودکار برچسب <meta> را اضافه می‌کند.همچنین <hr/> را به <hr> که شناخته‌شده‌تر است تبدیل می‌کند

XSLT



شکل ۹-۱۰ : در اینجا می‌توانید ببینید تا این مرحله صفحه شما در مرورگر چگونه خواهد بود. چندان جالب به نظر نمی‌رسد ولیکن امیدوارکننده است.

نکته‌ها

- کدهای HTML نوشته شده باید از قوانین قالب‌بندی XML پیروی کند. با وجود این که نیازی به وجود فایلی با فرمت XHTML (که تنها فرق آن در این است که باید تمامی نامهای عنصرها و ویژگیها باحروف کوچک نوشته شود) نمی‌باشد، ولیکن نوشتن با این قالب به صحیح بودن سند شما کمک می‌نماید. برای اطلاعات بیشتر در مورد نوشتن HTML با توجه به قواعد XML به فصل ۱ (نوشتن XML) و صفحه ۲۳ آن (قواعد نوشتن XML) مراجعه کنید. نگاه‌کردن به پیوست A با عنوان XHTML خالی از فایده نیست.
- در مثال مقدار ویژگی تطابق تعیین نشد زیرا خروجی HTML را می‌توان به هر قالبی اضافه کرد.
- به روش فوق می‌توان هرنوع گرهی را ایجاد کرد. هر چیزی که دستور XSL نباشد به همان صورت در سند نهایی ظاهر می‌گردد.
- با استفاده از دستورهای xsl:element, xsl:attribute و xsl:text نیز می‌توان عنصرها (و ویژگیها و عبارات سندی) را ایجاد کرد اما این دستورها مقداری پیچیده هستند و برای موارد خاص می‌باشند.
- برای اطلاعات بیشتر در مورد نحوه نوشتن HTML می‌توانید به کتاب"HTML برای وب " از همین مولف مراجعه کنید. آدرس http://www.cookwood.com/html4_4e نیز دارای اطلاعات مفیدی می‌باشد.

محتویات گره در خروجی

بعد از این که کدی از HTML که محتویات یک گره را قالب‌بندی می‌کند ایجاد کردید، می‌خواهید این محتوا را (که مقدار رشته نامیده می‌شود) در خروجی داشته باشید. ساده‌ترین راه برای اضافه‌کردن محتویات یک گره (مانند یک عنصر) به خروجی نوشتن آنها به همان صورتی که هستند می‌باشد.

برای استفاده از محتویات گره در خروجی :

۱- در صورت تمایل کدی از HTML بنویسید که محتوای مورد نظر را قالب‌بندی کند (صفحه ۱۴۰).

۲- عبارت **<xsl:value-of** را تایپ کنید.

۳- عبارت **select="expression"** را تایپ کنید. expression مشخص‌کننده دسته‌گرهی از سند XML است که محتوای آن باید در خروجی باشد.

۴- برای کامل شدن این قسمت یک /> تایپ کنید.

نکته‌ها

- برای استفاده از محتوای گره فعلی در خروجی از عبارت select="." استفاده کنید. برای اطلاعات بیشتر در مورد نوشتن عبارات، به فصل ۱۱ (XPath : عبارتها و طرحها) مراجعه کنید.
- در صورتی که دسته گره مشخص شده توسط عبارت فوق بیشتر از یک گره داشته باشد، تنها مقدار رشته اولین گره در خروجی خواهد بود. در مثال **شکل ۱۱-۱۰** یک دسته گره با دو گره که هر دو در شرط عبارت می‌گنجند یافت می‌شود، ولیکن تنها مقدار اولی ("Tiger") خروجی خواهد بود.
- به طور کلی مقدار رشته یک گره سندی است که در آن گره وجود دارد. در صورتی که گره دارای عنصرهای فرزند باشد، مقدار رشته شامل سندهایی که در این عنصرهای فرزند وجود دارند نیز می‌شود.
- در صورتی که دسته‌گره خالی باشد، چیزی در خروجی نخواهد بود.

code.xml

```
<?xml version="1.0"?>
<endangered_species>
<animal>
<name language="English">Tiger</name>
<name language="Latin">panthera tigris</name>
<threats><threat>poachers</threat>
```

شکل ۱۰-۱۰ : در این قسمت از سند XML عنصر نام و محتویات آن دیده می‌شود.

code.xslt

```
<xsl:template match="/">
<html><head><title>Endangered Species
</title></head><body bgcolor="white">
<p>The mighty <xsl:value-of
select="endangered_species/animal/name
[@language='English']"/> faces numerous threats.
For more information, check out the World Wildlife
Federation's <a
href="http://www.worldwildlife.org/
species/species.cfm?"> pages</a>.
</p><hr/>
</body>
</html>
</xsl:template>
```

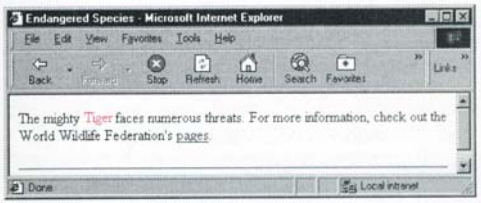
شکل ۱۱-۱۰ : به جای Endangered animals می‌خواهیم به محتویات عنصر name دسترسی پیدا کنیم (که یک عنصرanimal درون یک عنصر endangered_species با مقدار ویژگی language برابر English است).

- در بعضی از نسخه‌های برنامه پردازشگر XSLT احتیاجی به ویژگی select نمی‌باشد (و فرض می‌شود که گره فعلی مفقود شده است). ولیکن به طور رسمی این ویژگی باید وجود داشته باشد.
- از آنجا که عنصر <xsl:value-of> هیچ‌گاه دارای محتوایی نمی‌باشد، همیشه می‌توان قسمتهای باز و بسته کردن دستور را همان‌طور که در مرحله ۴ نشان داده شد ترکیب کرد.
- در صورتی که عبارت دارای مقدار بولی باشد، خروجی درست (True) و یا غلط (False) خواهد بود. در صورتی که عبارت به صورت عددی باشد، این عدد به رشته تبدیل می‌گردد.
- می‌توان عبارتی ساخت که مقداری را با استفاده از توابع محاسبه کند. برای اطلاعات بیشتر به فصل ۱۲ (عبارتهای شرطی و توابع) مراجعه کنید.

code.html

```
<html><head><meta http-equiv="Content-Type"
content="text/html; charset=utf-8"><title>
Endangered Species</title></head><body
bgcolor="white">

<p>The mighty Tiger faces numerous threats.
For more information, check out the World Wildlife
Federation's <a href=
"http://www.worldwildlife.org/species/species.cf
m?"> pages</a>.
    </p>
    <hr></body></html>
```

کل ۱۲۱۲–۱۰ : هنگامی که پردازشگر XSLT قالب پایه را اعمال می‌کند، ابتدا قسمت HTML header را در خروجی قرار می‌دهد.سپس هنگامی که به عنصر xsl:value-of رسید یک دسته گره حاوی دو گره پیدا می‌کند. این دسته گره محتویات عنصرهای animal و name است که language آنها برابر English می‌باشد. سپس مقدار اولین گره را که Tiger است در خروجی قرار می‌دهد.



شکل ۱۳–۱۰ : اکنون خروجی شما واقعا از ورودی سند XML استفاده می‌کند.

ساخت و اجرای قواعد قالبها

قواعد قالب چگونگی ظاهرشدن یک قطعه از سند XML اصلی را در خروجی تعیین می‌کنند. هر قاعده شامل سه قسمت است : قسمت آغازین دستور که محلی را که قالب باید برروی آن اجرا شود مشخص می‌کند، قسمت میانی تعیین می‌کند بعد از این که این قسمت پیدا شد چه عملی انجام گردد و قسمت پایانی که برای کامل شدن این قطعه می‌باشد.

برای ساخت یک قاعده قالب :

۱- برای شروع عبارت **<xsl:template>** را تایپ کنید.

۲- سپس عبارت **match="pattern"** را تایپ کنید. در این عبارت pattern قسمتهایی از سند XML که این قالب باید بر روی آن اجرا شود مشخص می‌نماید. مثال این مورد در صفحات ۱۶۱-۱۵۴ موجود است.

۳- برای کامل کردن دستور یک > تایپ کنید.

۴- اعمالی را که باید در صورت یافتن گرهی که در شرط ۲ صدق کند انجام گردد، مشخص کنید. دستورات این قسمت در ادامه این فصل مورد بررسی قرار گرفته است.

۵- برای تکمیل، **</xsl:template>** را تایپ کنید.

نکته‌ها

- قالب ریشه که در صفحه ۱۳۹ مشاهده گردید در واقع یک قالب می‌باشد که با گره ریشه مطابقت می‌کند.
- قالب ریشه تنها قالبی است که به طور خودکار صدا می‌شود. تمامی قالبهای دیگر باید در سند صدا زده شوند. در غیر این صورت این قالبها مورد بررسی قرار نمی‌گیرند.
- ترتیب قالبها در یک سند دارای اهمیت خاصی نیست. ترتیب و محل دستور xsl:apply-templates است که ترتیب پردازش قالبها را مشخص می‌کند.

code.xslt

```
<xsl:template match="/"><html><head>
<title>Endangered Species</title></head> <body
bgcolor="white">

<xsl:apply-templates
select="endangered_species/animal"/>
</body></html>
</xsl:template>

<xsl:template match="animal">
<p align="center">
<br/><font size="+3"><xsl:apply-templates
select="name" /></font></p>
<p>The mighty <xsl:value-of
select="name[@language='English']"/> faces
numerous threats. For more information, check out
the World Wildlife Federation's <a
href="http://www.worldwildlife.org/species/speci
es.cfm?"> pages</a>. </p><hr/>
</xsl:template>

<xsl:template match="name[@language=
'English']"><nobr><b><xsl:value-of select="."/>:
</b> </nobr></xsl:template>

<xsl:template match="name[@language=
'Latin']"><nobr><i><xsl:value-of select="."/></i>
</nobr></xsl:template>
```

شکل ۱۴-۱۰ : در این قسمت چهار قاعده قالب وجود دارد: قالب پایه، قالب برای گره‌های animal، قالب برای گره‌هایی از name که language آنها دارای مقدار English باشد ویک قالب برای گره‌هایی از name که language آنها دارای مقدار Latin باشد.

code.xslt

```
<xsl:template match="/"><html><head>
<title>Endangered Species</title></head> <body
bgcolor="white">
<xsl:apply-templates
select="endangered_species/animal"/>
</body></html>
</xsl:template>

<xsl:template match="animal">
<p align="center">
<br/><font size="+3"><xsl:apply-templates
select="name" /></font></p>
```

شکل ۱۵-۱۰ : این قسمتی از صفحه سبکی است که در **شکل ۱۴-۱۰** نشان داده شده است؛ با این تفاوت که عنصرهای xsl:apply-templates مشابه مشخص

برای استفاده مناسب از یک قاعده قالب، باید آن را در محلهای خاصی به کار برد.

برای اعمال یک قاعده قالب :

۱- در سند یک قاعده قالب عبارت **<xsl:apply-templates** را تایپ کنید.

۲- در صورت تمایل، عبارت **select="expression"** را تایپ کنید. expression عنصرهایی از سند XML را که قاعده‌های آنها باید اجرا شوند، مشخص می‌کند.

۳- برای کامل شدن این دستور یک /> تایپ کنید.

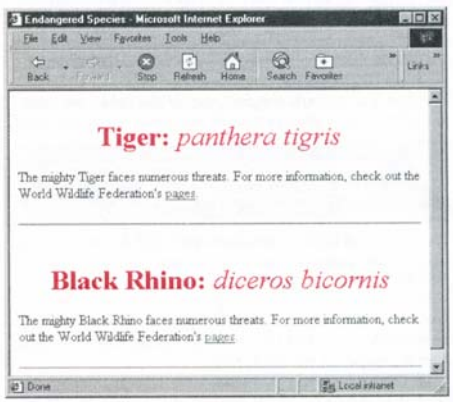
نکته‌ها

- در صورتی که در مرحله ۲ ویژگی select مشخص نگردد، پردازشگر به طور خودکار بر روی تمامی فرزندان گره فعلی یک قاعده را اجرا می‌کند.
- عنصر xsl:apply-templates به دنبال مناسبترین قاعده قالب برای هر گره پردازش شده می‌گردد. تصمیم‌گیری در مورد این که چه گره‌هایی باید پردازش شوند با استفاده از عبارت به کار رفته برای این عنصر معین می‌گردد. تصمیم‌گیری در مورد این که چه قالبهایی استفاده گردند با نگاه‌کردن به طرحهای این قالبها و پیداکردن مناسبترین قالبی که با هر گره در دسته‌گره تطبیق داشته باشد صورت می‌گیرد. با این وجود امکان انتخابهای دیگری برای هر گره نیز وجود دارد.
- در صورتی که پردازشگر هیچ قالب مناسبی پیدا نکند، یک قالب پیش‌ساخته مورد استفاده قرار می‌گیرد. در صورتی که گره مورد بحث، گره ریشه یا گره عنصر باشد، این عمل به معنی پیداکردن قالب مناسب برای تمام گره‌های فرزند است. برای یک گره سندی، این عمل به معنی قرارگرفتن این سند به صورت موجود در خروجی است. برای یک گره ویژگی نیز این عمل به معنی قرارگرفتن این ویژگی به صورت متنی در خروجی است.

code.html

```
<body bgcolor="white">

<p align="center"><br><font size="+3">
<nobr><b>Tiger: </b></nobr><nobr>
<i>panthera tigris</i></nobr>
</font></p>

<p>The mighty Tiger faces numerous threats. For
more information, check out the World Wildlife
Federation's <a
href="http://www.worldwildlife.org/species/
species.cfm?"> pages</a>. </p>

<hr>

<p align="center"><br><font size="+3">
<nobr><b>Black Rhino: </b></nobr>
<nobr><i>diceros bicornis</i></nobr>
</font></p>

<p>The mighty Black Rhino faces numerous threats.
For more information, check out the World Wildlife
Federation's <a
```

شکل ۱۶-۱۰ : قسمت کم‌رنگ قسمتی است که برای هر گره animal ایجاد شده است(در این سند XML دو گره animal وجود دارد). قسمت پررنگتر نتیجه اثر قالبهای name است.



شکل ۱۷-۱۰ : دقت کنید که دسته گره endangered_species/animal شامل هر دو گره animal است (Tiger و Black Rhino). از این رو بر خلاف **شکل ۱۳-۱۰** هر دو دسته در خروجی خواهند بود.قالبهای name در هنگام پردازش یک گره animal صدا می‌شوند(بدین وسیله دو عنوان بالا ساخته شده‌اند).

پردازش دسته‌ای گره‌ها

عنصر xsl:for-each تمامی گره‌های موجود در یک دسته خاص را به ترتیب پردازش می‌کند. مهمترین تفاوت آن با xsl:apply-templates در نحوه عمل آن است. نتیجه استفاده از هر دو یکسان است.

برای پردازش دسته‌ای گره‌ها :

۱- درون یک قاعده قالب عبارت **<xsl:for-each** را تایپ کنید.

۲- عبارت **select="expression"** را تایپ کنید. expression گره‌هایی را که باید مورد پردازش قرار گیرند مشخص می‌کند.

۳- برای کامل شدن این دستور > را تایپ کنید.

۴- پردازشی که باید انجام گردد مشخص کنید.

۵- عبارت **</xsl:for-each>** را تایپ کنید.

code.xslt

```
<xsl:template match="animal">
<p align="center"><br/><font
size="+3"><xsl:apply-templates select="name"
/></font></p>
<table width="100%" border="2">
<tr><th>Subspecies</th><th>Region</th><th>Nu
mber Left</th><th>As Of</th></tr>
<xsl:for-each select="subspecies">
<tr><td><xsl:apply-templates
select="name"/></td>
<td><xsl:value-of select="region"/></td>
<td><xsl:value-of select="population"/></td>
<td><xsl:value-of
select="population/@year"/></td></tr>
</xsl:for-each>
</table>
<p>The mighty <xsl:value-of
```

شکل ۱۸-۱۰ : دقت کنید که <table> و اولین سطر آن قبل از دستورالعمل xsl:for-each و </table> بعد از آن قرار گرفته است.در دستورالعمل xsl:for-each تمامی کارهایی که باید برای هر گره در دسته مورد نظر انجام شود قرار دارند(در این مثال هر subspecies برای animal فعلی).

اولین خط، اولین خانه اولین سطر جدول را در خروجی قرار می‌دهد.سپس دسته‌گره name را پردازش می‌کند و در پایان این خانه را می‌بندد.

خط دوم، یک خانه جدول دیگر ایجاد می‌کند، مقدار گره region را در آن قرار می‌دهد و این خانه را می‌بندد.

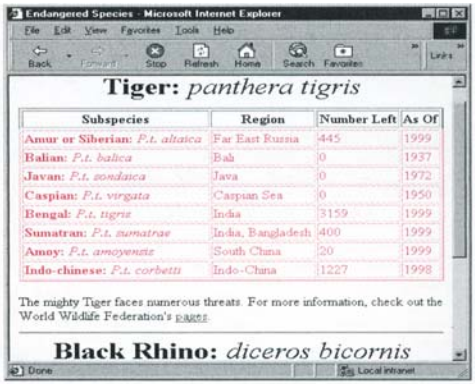
خط سوم، یک خانه جدول دیگر ایجاد می‌کند، مقدار گره population را در آن قرار می‌دهد و این خانه را می‌بندد.

خط چهارم، یک خانه جدول دیگر ایجاد می‌کند، مقدار ویژگی year مربوط به عنصر population را در آن قرار می‌دهد و سپس این سطر جدول را پایان می‌دهد.

هر خط برای هر گره در دسته‌گره انتخاب شده پردازش می‌شود(در اینجا subspecies). از این رو هر خط یک <tr> برای animal subspecies ایجاد می‌کند.

```
code.html
<i>panthera tigris</i></nobr></font></p>
<table width="100%" border="2">
<tr><th>Subspecies</th>
<th>Region</th>
<th>Number Left</th>
<th>As Of</th></tr>
<tr><td><nobr><b>Amur or Siberian: </b>
</nobr><nobr><i>P.t. altaica</i></nobr></td>
<td>Far East Russia</td>
<td>445</td>
<td>1999</td>
</tr>
<tr><td><nobr><b>Balian: </b></nobr>
<nobr><i>P.t. balica</i></nobr></td>
<td>Bali</td>
<td>0</td>
<td>1937</td></tr>
...
</table>
  <p>The mighty Tiger faces numerous threats. For
```

شکل ۱۹-۱۰ : دستورالعمل xsl:for-each برای هر subspecies در animal فعلی یک سطر جدید ایجاد می‌کند(به دلیل محدودیت فضا همه آنها نشان داده نشده‌اند). هنگامی که همه گره‌های موجود در دسته انتخاب‌شده پردازش شدند، بقیه قالب ادامه می‌یابد.



شکل ۲۰-۱۰ : یک جدول دیگر حاوی اطلاعات در مورد Black Rhino زیر نام آن وجود دارد که در شکل نشان داده نشده است

نکته‌ها

- معمولا عنصر xsl:for-each برای ساختن جدولهای HTML به کار می‌رود. برچسبهای باز و بسته کردن جدول قبل و بعد از آن عنصر قرار می‌گیرند و دستورات td و tr به عنوان قسمتی از عملیات پردازش می‌باشند.
- عنصر xsl:for-each قبل از قاعده‌هایی قرار می‌گیرد که باید برای گره‌ها تکرار شوند. در صورت تمایل می‌توانید قبل و بعد از برچسبهای باز و بسته کردن، یک چهارچوب (مثلا یک جدول) قرار دهید.
- به دلیل این که جدول HTML قسمتی از یک فایل XSLT می‌باشد، باید از قوانین نوشتن XML پیروی کند. از این رو باید به‌خاطر داشته باشید که هر برچسب بازکردن باید یک برچسب بستن داشته باشد و هم‌چنین عنصرها نباید با یکدیگر همپوشانی داشته باشند. برای اطلاعات بیشتر به مبحث قوانین نوشتن XML در صفحه ۲۳ مراجعه کنید.

پردازش گره‌ها به صورت مشروط

گاهی در نوشتن سندهای XML، پردازش یک گره و یا یک دسته‌گره در صورت برقرار بودن یک شرط مورد نیاز می‌باشد. این شرط به صورت یک عبارت نوشته می‌شود. برای مثال، در صورت خالی بودن محتوای یک گره و یا برابر بودن محتوای یک گره با یک رشته متنی عمل خاصی انجام گردد.

برای پردازش مشروط گره‌ها :

۱- در داخل یک قاعده قالب، عبارت <xsl:if را تایپ کنید.

۲- عبارت **test="expression"** را تایپ کنید. در این عبارت expression یک دسته‌گره، رشته و یا یک عدد را مشخص می‌کند. برای اطلاعات بیشتر در مورد نوشتن این‌گونه عبارات به فصل ۱۱ (XPath : عبارتها و طرحها) مراجعه کنید.

۳- برای کامل شدن این قطعه، یک > تایپ کنید.

۴- در این مرحله مشخص کنید در صورتی که دسته‌گره، رشته و یا عدد مشخص شده در مرحله ۲ خالی نباشد (و یا در مورد عدد برابر صفر نباشد) چه عملی انجام شود.

۵- عبارت **</xsl:if>** را تایپ کنید.

نکته‌ها

- یک دسته‌گره در صورتی صحیح (True) فرض می‌شود که خالی نبوده، حاوی چندین گره باشد.
- در صورتی که می‌خواهید هنگام غلط (False) بودن شرط نیز عملی انجام گردد (مانند یک شرط else) از xsl:choose استفاده کنید (به صفحه ۱۴۹ مراجعه کنید).
- تمامی انواع شرطها قابل بررسی هستند. برای اطلاعات بیشتر به فصل ۱۱ (XPath : عبارتها و طرحها) مراجعه کنید.

```
code.xslt
<td><xsl:apply-templates
select="population"/></td>
...
<xsl:template match="population">
<xsl:value-of select="."/>
<xsl:if test=". = 0">
<font color="red" title="that means there are no
more left"> --&gt;Extinct!!</font>
</xsl:if>
```

شکل ۲۱-۱۰ : اکنون به جای قراردادن مقدار population در خروجی، یک قالب اعمال می‌کنیم.در این قالب ابتدا مقدار را به خروجی می‌دهیم.سپس آزمایش می‌کنیم تا مشخص شود که مقدار population صفر است یا نه.در صورت صفر بودن "-- >Extinct" را با رنگ قرمز اضافه می‌کنیم تا این صفر بیشتر به نظر آید.دقت کنید که علامت کوچکتر باید به صورت &gt نوشته شود.

```
code.html
<tr><td><nobr><b>Amur or Siberian: </b>
</nobr><nobr><i>P.t. altaica</i></nobr></td>
<td>Far East Russia</td>
<td>445</td>
<td>1999</td></tr>
<tr><td><nobr><b>Balian: </b></nobr>
<nobr><i>P.t. balica</i></nobr></td>
<td>Bali</td>
<td>0<font color="red" title="that means there are
no more left"> -->Extinct!!</font></td>
<td>1937</td></tr>
```

شکل ۲۲-۱۰ : متن اضافی با رنگ قرمز تنها در صورت صفر بودن population اضافه می‌شود.

شکل ۲۳-۱۰ : در صورت صفر بودن population متنی اضافه می‌شود تا اطلاعات مشخص‌تر شوند.

اضافه‌کردن انتخابهای شرطی

با استفاده از دستور xsl:if که در صفحه قبل مورد بررسی قرار گرفت، تنها می‌توان یک شرط را مورد بررسی قرار داد و تنها می‌توان یک عمل را انجام داد. از xsl:choose در زمانهایی استفاده می‌شود که بررسی چندین شرط و عمل با توجه به برقرار بودن هریک مورد نظر باشد.

برای اضافه‌کردن انتخابهای شرطی :

۱- در داخل یک قاعده قالب، عبارت <**xsl:choose**> را تایپ کنید.

۲- برای شروع اولین شرط عبارت <**xsl:when** را تایپ کنید.

۳- عبارت **test="expression"** را تایپ کنید. expression یک دسته‌گره، رشته و یا عدد را مشخص می‌کند. برای اطلاعات بیشتر در این مورد به فصل ۱۱ مراجعه کنید.

۴- برای تکمیل عنصر xsl:when یک > تایپ کنید.

۵- مشخص کنید در صورت خالی نبودن (و یا صفر نبودن) دسته‌گره، رشته و یا عدد مرحله ۳، چه پردازشهایی باید انجام گیرد.

۶- دستور <**/xsl:when**> را تایپ کنید.

۷- برای هر شرط موجود مراحل ۲ تا ۶ را تکرار کنید.

۸- در صورت تمایل، عبارت <**xsl:otherwise**> را تایپ کنید. این عبارت مشخص می‌کند اگر هیچ یک از شروط تعیین شده توسط xsl:when برقرار نبود چه عملی صورت گیرد.

۹- برای تکمیل عبارت <**/xsl:choose**> را تایپ کنید.

نکته

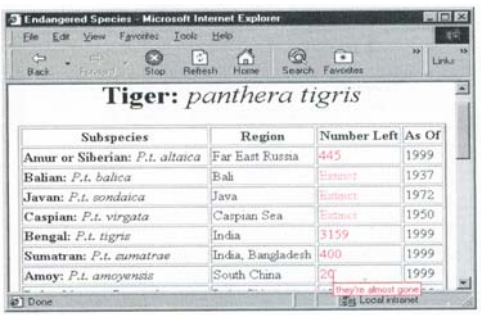
- عمل تعیین شده در اولین شرطی که برقرار باشد انجام می‌شود و بعد از آن، شرطهای دیگر مورد بررسی قرار نمی‌گیرند.

```
code.xslt
<xsl:template match="population">
<xsl:choose>
<xsl:when test=". = 0">
<font color="red" title="that means there are
no more left">Extinct</font>
</xsl:when>
<xsl:when test=". &gt; 0 and . &lt; 50">
<font title="they're almost gone"><xsl:value
of select="."/></font>
</xsl:when>
<xsl:otherwise>
<xsl:value-of select="."/>
</xsl:otherwise></xsl:choose>
</xsl:template>
```

شکل ۲۴-۱۰ : ابتدا پردازشگر XSLT آزمایش می‌کند که مقدار population برابر صفر است و یا خیر. در صورت صفر بودن، Extinct را در خروجی قرار می‌دهد. در صورت صفر نبودن آزمایش می‌کند تا ببیند که بیشتر از ۰ و کمتر از ۵۰ است.در صورت برقراری این شرط یک tooltip احتیاطی کنار مقدار قرار می‌گیرد. در تمامی حالات دیگر پردازشگر تنها مقدار را در خروجی قرار می‌دهد.

```
code.html
<tr><td><nobr><b>Balian: </b></nobr>
<nobr><i>P.t. balica</i></nobr></td>
<td>Bali</td>
<td><font color="red" title="that means there are
no more left">Extinct</font></td>
<td>1937</td></tr>
```

شکل ۲۵-۱۰ : در این قسمت می‌بینید که به جای اضافه‌کردن Extinct بعد از مقدار، فقط کلمه Extinct در هنگام صفر بودن population نمایش داده می‌شود.



شکل ۲۶-۱۰ : اکنون با توجه به مقدار population سه کار انجام می‌شود.برای مشخص شدن مقدارهای کم population یک tooltip اضافه کردیم.

مرتب کردن گره‌ها قبل از پردازش

پردازشگر به طور پیش‌فرض، گره‌ها را به همان ترتیب موجود در سند پردازش می‌کند. برای پردازش به ترتیب دیگر، عنصر xsl:sort به عنصرهای xsl:apply-templates و یا xsl:for-each اضافه می‌شود.

برای مرتب کردن گره‌ها قبل از پردازش :

۱- بلافاصله بعد از عنصر xsl:apply-templates یا xsl:for-each و یا xsl:sort دیگر، عبارت **xsl:sort>** را تایپ کنید.

۲- عبارت **select="criteria"** را تایپ کنید. criteria عبارتی است که پردازش گره‌ها باید با استفاده از ترتیبی که آن مشخص می‌کند انجام گیرد.

۳- در صورت تمایل عبارت **order="descending"** را نیز اضافه کنید. ترتیب مرتب‌سازی به طور پیش‌فرض صعودی می‌باشد.

۴- در صورت تمایل عبارت **data-type="text"** و یا **data-type="number"** را با توجه به نوع داده‌های خود تایپ کنید.

۵- برای تکمیل عنصر xsl:sort یک /> تایپ کنید.

۶- برای هر تعداد مورد نیاز، مراحل ۱تا۵ را تکرار کنید.

نکته‌ها

- دقت داشته باشید که در مرحله ۴ نوع داده را به طور صحیح انتخاب کنید. در صورتی که اعداد به صورت متنی فرض شوند نتیجه مرتب‌سازی درست نخواهد بود. همچنین مرتب‌سازی متن به صورت عددی نیز نتیجه صحیحی نخواهد داشت.
- نزولی (Descending) به معنی آن است که از اعداد بزرگ به اعداد کوچک و از حرف Z به حرف A مرتب گردد. صعودی به معنی آن است که اعداد کوچکتر (و حروف ابتدایی) در آغاز قرار گیرند.

code.xslt

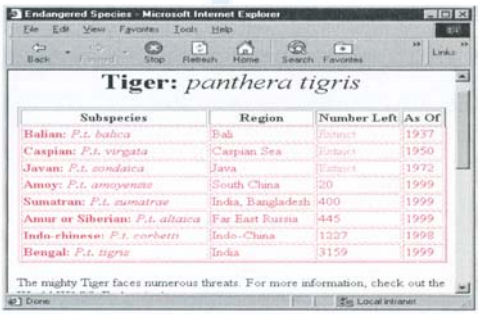
```
<table width="100%" border="2">
<tr><th>Subspecies</th><th>Region</th><th>Number Left</th><th>As Of</th></tr>
<xsl:for-each select="subspecies">
<xsl:sort select="population" data-type="number"/>
<xsl:sort select="population/@year" data-type="number"/>
<tr><td><xsl:apply-templates select="name"/></td>
```

شکل ۲۷-۱۰ : در قسمت بالای صفحه سبک XSLT، بعد از عنصر xsl:for-each و قبل از تولید خروجی دو دستورالعمل xsl:sort اضافه کردیم.

code.html

```
<table width="100%" border="2">
<tr><th>Subspecies</th><th>Region</th>
<th>Number Left</th><th>As Of</th></tr>
<tr><td><nobr><b>Balian: </b></nobr>
<nobr><i>P.t. balica</i></nobr></td>
<td>Bali</td>
<td><font color="red" title="that means there are no more left">Extinct</font></td>
<td>1937</td>
```

شکل ۲۸-۱۰ : اکنون Bali tiger اولین گرهی خواهد بود که پردازش می‌شود و در بالای جدول نشان داده می‌شود.



شکل ۲۹-۱۰ : در این شکل ببرهای جهان به ترتیب جمعیت و بعد به ترتیب سال انقراض مشخص شده‌اند.

ساخت ویژگیها

هدف از این قسمت اضافه‌کردن ویژگی (و مقدار آن) به یک عنصر خاص است.

برای ساخت ویژگی :

۱- بلافاصله بعد از دستور باز کردن عنصری که ویژگی باید به آن اضافه شود عبارت **<xsl:attribute>** را تایپ کنید.

۲- عبارت **name="att_name"** را تایپ کنید. att_name نامی است که ویژگی باید در عنصر داشته باشد.

۳- یک > تایپ کنید.

۴- سپس مقدار ویژگی جدید را تایپ کنید و یا آن را در خروجی قرار دهید.

۵- در پایان **</xsl:attribute>** را تایپ کنید.

نکته‌ها

- با استفاده از عنصر xsl:value-of می‌توان یک ویژگی را به وسیله اطلاعات موجود در سند XML مقداردهی کرد.
- این روش برای تبدیل عنصرهای تصویری دلخواه به دستورات img که استاندارد HTML می‌باشند، مناسب است.

code.xslt

```
<xsl:template match="picture">
    <img>
    <xsl:attribute name="src"><xsl:value-of
    select="./@filename"/></xsl:attribute>
    <xsl:attribute name="width"><xsl:value-of
    select="./@x"/></xsl:attribute>
    <xsl:attribute name="height"><xsl:value-of
    select="./@y"/></xsl:attribute>
    </img>
    </xsl:template>
```

شکل ۳۰-۱۰ : در اینجا قالبی که عنصرهای picture را به برچسبهای استاندارد img مربوط به HTML تبدیل می‌کند ساخته‌شده‌است.عبارت "./@filename" به معنای آن است که ویژگی filename گره فعلی را انتخاب کن.

code.html

```
<body bgcolor="white">
<p align="center">
<img src="tiger.jpg" width="200" height="197">
<br><font size="+3">
<nobr><b>Tiger: </b></nobr>
```

شکل ۳۱-۱۰ : یک برچسب کامل img مربوط به HTML !

شکل ۳۲-۱۰ : عکس majestic tiger در شکل بالا مشاهده می‌شود.