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).

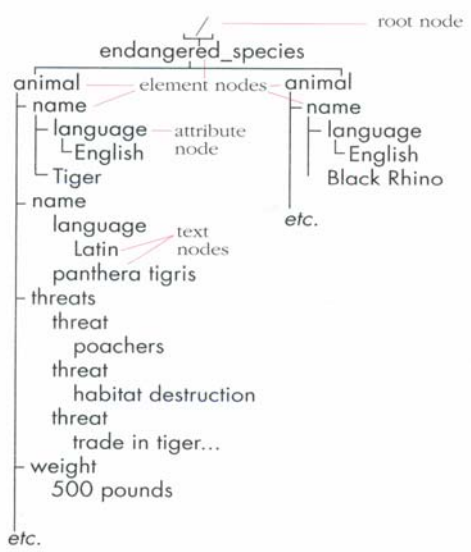
تبدیل XML با استفاده از XSLT

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.

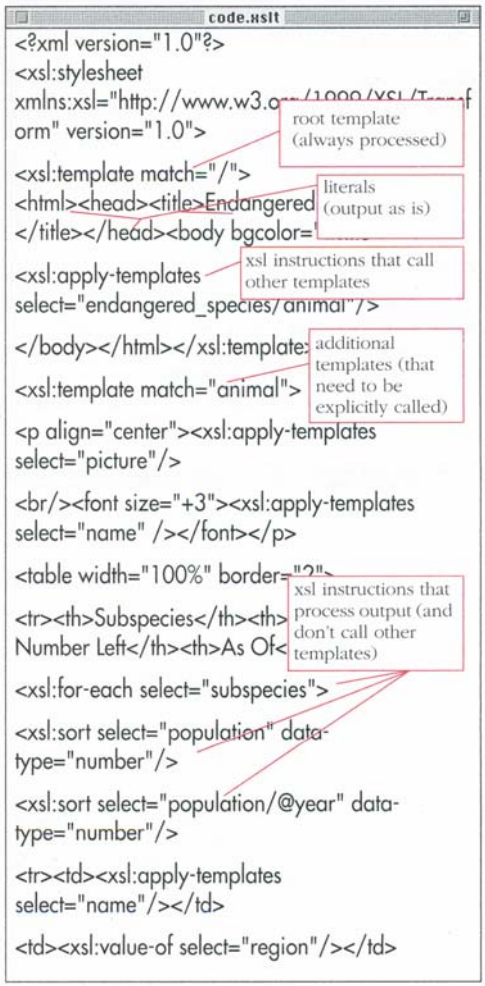
code.xml

```
<?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.



شکل ۳-۱۰ : صفحه سبک XSLT به طور کامل در سایت وب کتاب موجود است و می‌توانید آن را از این سایت دریافت کنید.

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.

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.

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

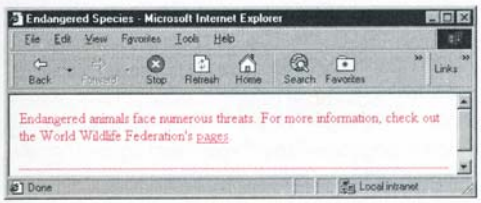
```
<xsl:template match="/">
<html><head><title>Endangered
Species</title></head><body bgcolor="white">
<p>Endangered animals face numerous threats.
For more information, check out the World Wildlife
Federation's <a
href="http://www.worldwildlife.org/
species/species.cfm?"> pages</a>.
</p><hr/></body></html>
</xsl:template>
```

شکل ۷-۱۰ : 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 ۸-۱۰ : 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



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>
```

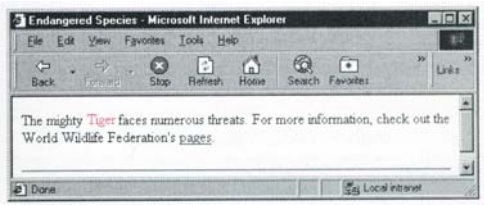
شکل ۱۱-۱۰ : به جای Endangered animals می‌خواهیم به محتویات عنصر name دسترسی پیدا کنیم (که یک عنصرanimal درون یک عنصر endangered_species با مقدار ویژگی language برابر English است).

- 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 ۲ ۱۲۱–۱۰ : 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

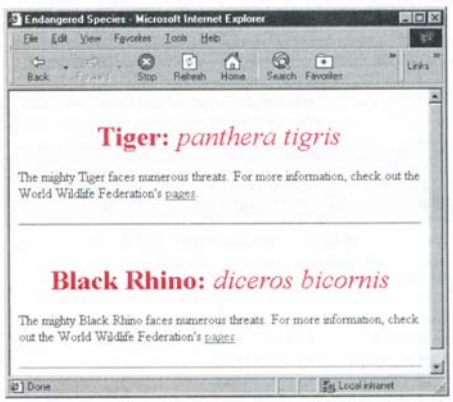
```
<xsl:template match="/"><html><head>
<title>Endangered Species</title></head> <body
bgcolor="white">
<xsl:apply-templates
select="endangered_species/animal"/>
</body></html>
</xsl:template>

<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 شده‌اند.

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).

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.

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.

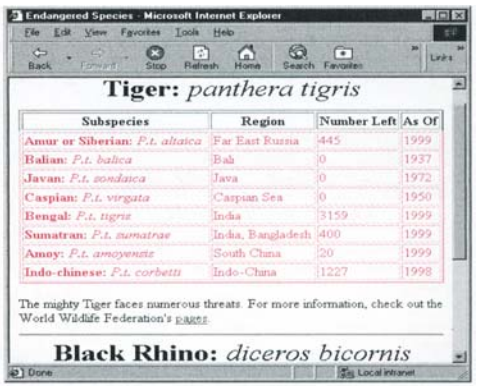
خط سوم، یک خانه جدول دیگر ایجاد می‌کند، مقدار گره population را در آن قرار می‌دهد و این خانه را می‌بندد.

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 سطر جدید 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

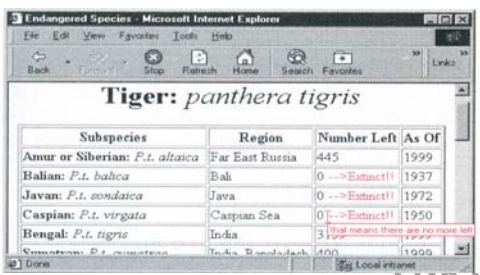
- یک دسته‌گره در صورتی صحیح (True) فرض می‌شود که خالی نبوده، حاوی چندین گره باشد.
- در صورتی که می‌خواهید هنگام غلط (False) بودن شرط نیز عملی انجام گردد (مانند یک شرط else) از xsl:choose استفاده کنید (به صفحه ۱۴۹ مراجعه کنید).
- تمامی انواع شرطها قابل بررسی هستند. برای اطلاعات بیشتر به فصل ۱۱ (XPath : 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

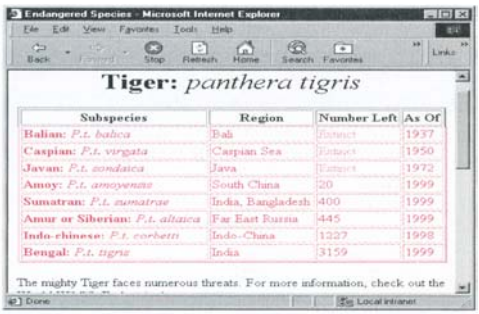
- دقت داشته باشید که در مرحله ۴ نوع داده را به طور صحیح انتخاب کنید. در صورتی که اعداد به صورت متنی فرض شوند نتیجه مرتب‌سازی درست نخواهد بود. هم‌چنین مرتب‌سازی متن به صورت عددی نیز نتیجه صحیحی نخواهد داشت.
- نزولی (Descending) به معنی آن است که از اعداد بزرگ به اعداد کوچک و از حرف Z به حرف A مرتب گردد. 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.