# 3-4. p 要素処理するテンプレート

これからはどのような順番でテンプレートを書いていってもいいのですが、ここでは元文書内に最も頻繁に登場する p 要素を処理するテンプレートを最初に書いてみることにします。

p 要素は段落を現す要素です。これを FO に変換するテンプレートを書けばいいわけですが、通常は fo:block オブジェクトに変換します。

```
<!-- p 要素を処理するテンプレート -->
<xsl:template match="p">
  <fo:block font-family="ＭＳ 明朝" font-size="10pt" space-before="0.5em"
    text-indent="1em" text-align="justify" keep-together.within-page="always">
    <xsl:apply-templates />
  </fo:block>
</xsl:template>
```

XML 文書内に p 要素を見つけるとこのテンプレートに処理が任せられます。XML 文書内の p 要素の内容を取り出し fo:block と/fo:block で囲んでいるだけです。つまり p 要素を fo:block オブジェクトに変換しているわけです。

また fo:block の属性として font-family="ＭＳ 明朝" font-size="10pt" space-before="0.5em"を指定していますが、それぞれ「フォントは MS 明朝」「フォントサイズは 9 ポイント」「直前の段落との間に 0.5 文字分の間隔をとる」ということを言っています。

同様に、text-indent="1em" はブロック（段落）の先頭行を 1 文字分インデントさせる、text-align="justify"は均等割付するための属性です。

keep-together.within-page="always"は「このブロックは必ず同じページ内に収める」と言う意味です。この属性を付けることで、ひとつのブロック（段落）が２ページにまたがるような配置が行われなくなります。

このテンプレートによりたとえば

```
<p>では、試しにある日のスポーツニュースを XML で書いてみましょう。</p>
```

という XML 文書が

```
<fo:block font-family="ＭＳ 明朝" font-size="10pt" space-before="0.5em" text-indent="1em" text-
align="justify" keep-together.within-page="always">では、試しにある日のスポーツニュースを XML で書いてみましょう。</fo:block>
```

という FO に変換されます。

# 3-5. b 要素を処理するテンプレート

b 要素は強調文字（太字）にしたいときに使う要素です。次のようなテンプレートで処理します。

```
<!-- b 要素を処理するテンプレート -->
<xsl:template match="b">
    <!-- 改行をともなわないで文字属性を変えるには fo:inline を使います -->
    <fo:inline font-weight="bold">
        <xsl:apply-templates />
    </fo:inline>
</xsl:template>
```

このテンプレートにより

```
<b>「構造化」</b>
```

という XML 文書が

```
<fo:inline font-weight="bold">「構造化」</fo:inline>
```

という FO に変換されます。*段落内の一部の文字属性を変えたいという場合、このように fo:inline を使います。*