Maya – JavaServer Templates –

chapter.1 エレメント出力

WEB ページ出力時、HTML テンプレート上の「テンプレート」部分も明示的にインジェクションする場合と同様に、テンプレートプロセッサに置き換えられて出力されます。テンプレートのエレメントは ElementProcessor に、テンプレートのテキスト部は CharacterProcessor に置き換えられます。

ElementProcessor や CharacterProcessor は明示的に利用することも可能です。明示的にこれらのプロセッサを利用することで、より柔軟なテンプレート記述が可能になります。

1.1. 暗黙的インジェクションによるエレメント出力

1.1.1 テンプレート上のエレメント

HTML テンプレート上のエレメントは、テンプレートビルダによって処理され、テンプレートプロセッサに変換されますが、テンプレートプロセッサのうち、ElementProcessor と、CharacterProcessor は何もインジェクションされなかった HTML テンプレート上エレメントの記述内容をそのまま出力する、Maya デフォルト動作を行うプロセッサです。

```
<div class="box">こんにちは</div>
```

と記述した場合、当然そのまま、出力されます。しかし、div エレメント部と「こんにちは」と書かれたテキスト部は異なるテンプレートプロセッサによって出力されます。

- → ElementProcessor による出力。
- こんにちは → CharacterProcessor による出力。
- → 同じ ElementProcessor による出力

1.1.2 m:rendered 属性によるコントロール

```
<div class="box" m:inject="c:out" c:value="こんにちは" m:rendered="true">
    ダミー文字列
</div>
```

上記の場合、c:out が JSTL Core の out タグ、m:が Maya の URI をプレフィックスマッピングしているとすると、下記のような出力となります。

```
<div class="box">こんにちは</div>
```

一方で、m:rendered="false"と、属性値を変更すると、下記の通りです。m:rendered 属性省略時に適用されるデフォルト値は false です。

```
こんにちは
```

この出力の違いは、テンプレートビルド時のインジェクションの動作の違いより生じます。Maya はテンプレートのノードを m:inject 属性によるインジェクション・設定 XML と id 属性でマッチングするインジェクション・XPath によりマッチングするインジェクションと多様なインジェクション動作を行いますが、その際に、m:rendered=true の際に、インジェクション結果をテンプレート上のオリジナルなエレメントの情報をコピーしたノードを生成し、包みます。

1.2. 明示的なインジェクションによるエレメント出力

1.1にて説明した暗黙的にインジェクションされる場合と異なり、明示的にElementProcessorをインジェクションするように記述を行うことができます。m:elementおよび、m:attributeの二つのQNameによって指示を行います。

List. 1-1　明示的な ElementProcessor インジェクション例

```
<?xml version="1.0" encoding="UTF-8"?>
<m:maya xmlns:m="http://maya.seasar.org"
        xmlns:c="http://java.sun.com/jstl/core_rt">
    <m:element m:id="foo" m:qName="span">
        <m:attribute m:qName="id" m:value="id_01"/>
        <c:if c:test="${ model.attrOutput }">
            <m:attribute m:qName="class" m:value="${ model.attrValue }"/>
        </c:if>
    </m:element>
</m:maya>
```

1.2.1 m:element ノード

m:element ノードによって、明示的な ElementProcessor のインジェクションを行います。

```
<m:element m:id="foo" m:qName="span">
```

例では、テンプレート上の id が「foo」のエレメントに対して、必須属性の m:qName に指示された「span」を QName としたエレメント出力を行うように ElementProcessor を追加します。

1.2.2 m:attribute ノード

出力エレメントの属性は、m:attribute ノードによって指示します。必須の m:qName 属性に指示された QName、やはり必須の m:value 属性に指示される値をエレメントの属性として出力します。

```
<m:attribute m:qName="id" m:value="id_01"/>
```

上記例では、「id="id_01"」と、親エレメントに追加します。

1.2.3 動的なエレメントおよび属性出力制御

List. 1-1において、属性の出力をc:ifを用いて条件分岐を行っています。このように、他のプロセッサを組み合わせることで、動的なエレメントおよび属性出力制御が行えます。

また、式言語を組み合わせることによって、属性値も動的に決定しています。m:value の値は常に動的に評価されます。

1.2.4 m:inject 属性によるインジェクションでの副作用

```
<span m:inject="m:element" class="box">
    <span m:inject="m:attribute" m:qName="id" m:value="id_01"/>
</span>
```

上記のように、テンプレート上において m:inject 属性を用いて直接 m:element をインジェクションした場合、テンプレートのインジェクションホストエレメント内に記述された HTML 属性がそのまま出力されます。

```
<span class="box" id="id_01"></span>
```

これは、暗黙のインジェクションによるテンプレート上のエレメント出力を行う機能の副作用です。特別害をなさないので、この副作用を抑制する仕様とはしていません。