Структура шаблона

Report (парный)

<report></report> - корневой элемент отчета. Имеет обязательный атрибут — name (название отчета)
Список необязательных атрибутов:
- pageWidth - ширина страницы в пикселях (default 595)
- pageHeight - высоту страницы в пикселях (default 842)
- orientation — ориентация страницы. (Portrait | Landscape)

Дочерние элементы:
<style>
<title>
<detail>

Style (одиночный)

<style> - стили позволяют единожды определить некоторый набор свойств элементов, а затем использовать этот набор в любом блоке отчета. Стиль применятеся для элемента <reportElement> путем указания имени стиля в качестве атрибута style="", в противном случае применяется стиль по умолчанию.

Пример:
1) Объявление стилей

```
     <style name="Regular" isDefault="true" fontSize="12"  />
     <style name="Emphasis" fontSize="12" isBold="true"   />
```

2) Применение

```
     <reportElement x="180" y="0" width="200" height="20" />
            <text><![CDATA[стиль по умолчанию]]></text>

     <reportElement x="180" y="20" width="200" height="20"
style="Emphasis" />
            <text><![CDATA[стиль «Emphasis»]]></text>
```

Атрибуты (необязательны для заполнения):
- name - уникальное имя стиля
- isDefault - этот стиль будет использован по умолчанию (true | false). Опционально стоит false.
- fontSize - размер шрифта
- forecolor - цвет шрифта (black|blue|gray|green|red| yellow|white)
- fontName - название шрифта
- isBold - определяет, будет ли текст “жирным” (true | false)
- isItalic - определяет, будет ли текст курсивом (true | false)

Title (парный)

<title> - Название отчета - отображается один раз в начале отчета.
Дочерние элементы:
<band>

Пример:

```
<title>
      <band height="50">
         <staticText>
            <reportElement x="100" y="16" width="100"
height="20"/>
            <textElement verticalAlignment="Top"
textAlignment="Right" />
            <text><![CDATA[Title]]></text>
         </staticText>
      </band>
   </title>
```

StaticText (парный)

<staticText> - Постоянный текст, который не зависит от каких-либо источников данных.
Дочерние элементы:

```
<reportElement />
<textElement />
<text></text>
```

Атрибуты <textElement>:
- textAlignment - выравнивание текста (Left | Center | Right | Justified) - необязателен
- verticalAlignment - вертикальное выранивание (Top | Middle | Bottom) - необязателен

Пример использования:

```
<staticText>
  <reportElement x="0" y="40" width="500" height="20"/>
  <textElement verticalAlignment="Top" textAlignment="Right"/>
  <text><![CDATA[Это пример форматирования с помощью
      textElement]]></text>
</staticText>
```

Detail (парный)

<detail> - ключевая и самая важная часть отчета. Можно назвать этот блок “телом” отчета. В блоке detail содержится основная информация. Для каждой записи в источнике данных. Может содержать несколько полос - элементов <band>
Дочерние элементы:
<band>

Band (парный)

<band> - блок, в котором перечисляются элементы отчета.
Атрибуты:
- height — высота полосы (блока) - обязателен
Дочерние элементы:
<staticText>

ReportElement (одиночный)

<reportElement> - Первый элемент каждого из подтегов тега <band>. Определяет, как данные размещаются для этого конкретного элемента (задает положение и размер элемента, перед которым указан).
Атрибуты (все обязательны для заполнения):
- x - координата x
- y - координата y
- width - ширина элемента
- height - высота элемента

CDATA

CDATA - это часть содержания элемента, которая помечена для парсера как содержащая только символьные данные, а не разметку.
Начинается последовательностью символов <![CDATA[
Заканчивается символами]]>

Например, в этой строке:

```
<sender>John Smith</sender>
```

открывающий и закрывающий теги «sender» будут интерпретированы как разметка. Однако, если записать её вот так:

```
<![CDATA[<sender>John Smith</sender>]]>
```

то этот код будет интерпретирован так же, как если бы было записано:

```
&lt;sender&gt;John Smith&lt;/sender&gt;
```

Таким образом, теги sender будут восприниматься так же, как «John Smith», то есть как текст.

Parameter (одиночный)

Параметр – это ссылка на объекты, которые передаются при процессе заполнения отчёта к движку генератора отчёта. Не являются обязательными элементами отчёта.

Атрибуты (обязательны для заполнения):
- name – имя параметра;
- class – класс(тип) значений параметра;

Пример объявление параметра:

```
<parameter name="name_parameter" class="name_class" />
```

Чтобы использовать параметр, используется конструкция: $P{name_parameter}, где name_parameter – имя объявленного ранее параметра.

Пример обращения и получения значения к конкретному параметру:

```
<![CDATA[$P{ name_parameter }]]>
```

queryString

<queryString> - шаблон, необходимый для определения SQL-запроса для данных отчета, если эти данные расположены в реляционных базах данных.
Существует три возможных способа использовать параметры запроса:

$P{paramName} Syntax
Эти параметры используются как обычные параметры, используя следующий синтаксис:

```
<queryString>
 <![CDATA[
  SELECT * FROM Orders WHERE OrderID <= $P{MaxOrderID} ORDER BY
ShipCountry
 ]]>
</queryString>
```

$P!{paramName} Syntax
Иногда бывает полезно использовать параметры для динамического изменения части SQL-запроса или для передачи всего SQL-запроса в качестве параметра для отчета наполнения процедур. В таких случаях синтаксис отличается немного.

```
<queryString>
 <![CDATA[
   SELECT * FROM $P!{MyTable} ORDER BY $P!{OrderByClause}
 ]]>
</queryString>
```

$X{functionName, param1, param2,...} Syntax
Есть также случаи, когда часть запроса должна быть динамически построена исходя из значения параметра отчета, с частью запроса, содержащего как текст запроса и связывания параметров. Такие сложные элементы запроса вводятся в запрос с использованием синтаксиса $ X {}.

```
<queryString><![CDATA[
SELECT * FROM Orders WHERE $X{IN, ShipCountry, CountryList}
]]></queryString>
```

Variable

<variable> - переменные упрощают шаблон отчета путем выделения в одной части выражения, которое широко используется во всем шаблоне отчета. Они могут выполнять расчеты, основанные на соответствующей формуле (выражении). В своем выражении (или формуле) переменная может использовать другие переменные, поля или параметры. Переменные вычисляются/инкрементируются с каждой записью из источника данных в том порядке, в котором они объявлены.

Атрибуты:
- `name` - Имя переменной. Является обязательным атрибутом и позволяет ссылаться на переменную по этому имени.
- `class` - тип данных, которому принадлежит значение переменной (`java.lang.string - by default`).
- `resetType` - периодичность установки исходного значения (`None | Report | Page | Column | Group`). По умолчанию стоит “Report”.

Варианты значений:
- `None` - никогда не инициализируется начальным значением. Содержит значение полученное путем вычисления выражения переменной.
- `Report` — инициализируется начальным значением (`<initialValueExpression>`) один раз в начале заполнения отчета (`by default`).
- Page — инициализируется заново вначале каждой страницы.
- `Column` - инициализируется заново вначале каждого нового столбца.
- `Group` - инициализируется заново каждый раз, когда задается новая группа.

- resetGroup — Если resetGroup используется, то он содержит имя группы и работает только в сочетании с атрибутом resetType, значение которого будет resetType=Group. Необязателен.
- incrementType — периодичность приращения переменной. (None | Report | Page | Column | Group). По умолчанию переменная инкрементируется с каждой записью из испточника данных. Стоит значение “None” by default.

Варианты значений:
- None — переменная инкрементируется с каждой записью (by default).
- Report — переменная никогда не инкрементируется.
- Page — переменная инкрементируется с каждой новой страницей.
- Column - переменная инкрементируется с каждым новым столбцом.
- Group - инкрементируется каждый раз, когда задается новая группа..
- incrementGroup - Если incrementGroup используется, то он содержит имя группы и работает только в сочетании с атрибутом incrementType, значение которого будет incrementType=Group.
- calculation - агрегатная функция (Count | Sum | Average | Lowest | Highest). “Nothing” по умолчанию.

Выражение, задающее значение переменной.
```
<variableExpression><![CDATA[$F{absence}]]></variableExpression>
```

Тег, задающий исходное значение. (может не использоваться)
```
<initialValueExpression><![CDATA[0]]></initialValueExpression>
```

Встроенные переменные
- PAGE_NUMBER — содержит номер текущей страницы. В конце заполнения отчета содержит общее число страниц в документе. Может использоваться для отображения текущей страницы и количества страниц одновременно.
- COLUMN_NUMBER — содержит номер текущей колонки. Переменная подсчитывает количество колонок для каждой страницы.
- REPORT_COUNT — содержит общее число обработанных записей в отчете.
- PAGE_COUNT — содержит количество записей обработанных в процессе заполнения текущей страницы.
- COLUMN_COUNT — число записей, обработанных при генерации текущей колонки.

Простейший пример использования:

```
<variable name="sumAbsenceByDiscipline" class="java.lang.Integer"
resetType="Group" resetGroup="Discipline" calculation="Sum">

<variableExpression><![CDATA[$F{absence}]]></variableExpression>
    <initialValueExpression><![CDATA[0]]></initialValueExpression>
</variable>

<variable name="sumAbsenceByName" class="java.lang.Integer"
resetType="Group" resetGroup="Student name" calculation="Sum">

<variableExpression><![CDATA[$F{absence}]]></variableExpression>
    <initialValueExpression><![CDATA[0]]></initialValueExpression>
</variable>
```

```
 <textField>
    <reportElement x="396" y="0" width="159" height="34"/>
    <textElement verticalAlignment="Middle"/>

<textFieldExpression><![CDATA[$V{sumAbsenceByDiscipline}]]></textF
ieldExpression>
</textField>
```

field

<field > - поле отчета представляет собой единственный способ отображения данных из источника в отчет шаблона, и использовать эти данные в выражениях отчетов, чтобы получить желаемый результат.

При использовании запроса SQL в отчете, необходимо убедиться, что есть столбец для каждого поля, полученный после выполнения запроса. Соответствующий столбец должен нести то же самое имя и имеют один и тот же тип данных, что и поле, которое отображает его.

Пример:

```
EmployeeID int 4
LastName varchar 50
FirstName varchar 50
HireDate datetime 8
```

Поля отчета следует указать следующим образом:

```
<field name="EmployeeID" class="java.lang.Integer"/>
<field name="LastName" class="java.lang.String"/>
<field name="FirstName" class="java.lang.String"/>
<field name="HireDate" class="java.util.Date"/>
```

`name` - атрибут имени элемента `<field >` является обязательным. Это позволяет ссылаться на поле в отчете по названию.

`class` - второй атрибут `<field >` определяет имя класса для значений полей. По умолчанию является `java.lang.String`, но оно может быть изменено на любой класс доступный во время выполнения. То есть для заполнения является необязательным.

`<fieldDesciption>` - это дополнительный текстовый фрагмент может оказаться очень полезным при реализации пользовательских данных. Например, Вы можете хранить в нем ключ, или любую информацию, которая может понадобиться для того, чтобы восстановить значение поля из источника пользовательских данных во время выполнения.

```
<field name="PersonName" class="java.lang.String"
isForPrompting="true">
<fieldDesciption>PERSON NAME</fieldDesciption>
</field>
```

Image Attributes

- scaleImage – необязательный атрибут, указывает на то, как должно быть вынесено изображение, если его фактический размер не подходит под размер элемента отчета изображения. Это происходит потому, что много изображений загружаются во время выполнения, и нет никакого способа узнать их точный размер при создании шаблона отчета. Возможные значения для этого атрибута:
    - Отсечение изображения: Если фактическое изображение больше, чем размер элемента image, то он будет отрезан так чтобы она сохранила своё первоначальное разрешение, и только область, которой соответствует указанный размер будет отображаться (scaleImage="Clip").
    - Принудительный размер изображения: Если размеры фактического изображения не соответствуют указанным для элемента image, который отображает его, изображение будет растягиваться так, что оно будет подходить в обозначенную область вывода. Изображение будет искаженно при неообходимости (scaleImage="FillFrame").
    - Сохранение пропорций изображения: Если фактическое изображение не помещается в элемент image , оно может быть адаптирован к этим размерам, сохраняя при этом свои первоначальные недеформированные пропорции (scaleImage="RetainShape").
    - Растягивания изображения, сохраняя ширину: Изображение может быть вытянуто по вертикали, чтобы соответствовать фактической высоте изображения, сохраняя при этом заявленную ширину элемента image (scaleImage="RealHeight").
    - Растяжение изображения,регулировка ширины: Изображение может быть вытянуто по вертикали, чтобы соответствовать фактической высоте изображения, в то время как регулируя ширину элемента изображения, чтобы соответствовать фактической ширине изображения (scaleImage="RealSize").
- Если тип scaleImage - Clip или RetainShape, и фактическое изображение меньше его заданного размера в шаблоне отчета или не имеют те же пропорции, изображение не может занимать все пространство, выделенное ему в шаблоне отчета. В таких случаях, вы можете выровнять изображение внутри заранее заданного пространства отчетов с помощью Align и атрибута VALIGN, которые определяют выравнивание изображения по горизонтальной оси (Left, Center, Right) и вертикальной (Top, Middle, Bottom). По умолчанию изображения выравниваются по верхней и левой внутренней границе. Являются необязательными.
- Все элементы изображения имеют динамическое содержимое. Однако изображения в отчете статическое и не обязательно исходит от источника данных или из параметров. Как правило, они загружаются из файлов на диске и представляют собой логотипы и другие статические ресурсы. Чтобы отобразить одно изображение несколько раз в отчете (например, логотип появляется в заголовке страницы), можно кэшировать изображение для лучшей производительности. Когда будет установлен атрибут isUsingCache (необязателен) как TRUE, обработчик отчетности будет пытаться распознать ранее загруженные изображения, используя их указанный источник. Эта функция кэширования для элементов изображения, чьи выражения возвращают объекты любого типа в качестве источника изображения. Флаг isUsingCache

устанавливается как TRUE по умолчанию для изображений, имеющих java.lang.String выражения и как FALSE для всех остальных типов. Ключ, используемый для кэша, является значением выражения исходного изображения; ключевые сравнения выполняются с использованием стандартного метода EQUALS. Как следствие, для изображений, имеющих источник java.io.InputStream с включенным кэшированием входной поток считывается только один раз, а затем изображение будет браться из кэша.Флаг isUsingCache не следует устанавливать в тех случаях, когда изображение имеет динамический источник (например, изображение загружается из бинарного поля базы данных для каждой строки), потому что изображения будут накапливаться в кэше и заполнение прекратится из-за ошибки, связанной с отсутствием памяти. Очевидно, что флаг не должен также быть установлен, когда один источник используется для получения различных изображений (например, URL-адрес, который будет возвращать другое изображение каждый раз, когда это доступо).

- isLazy - Флаг, который определяет, должно ли быть загружено изображение и обрабатывается при заполнении отчета или во время экспорта, в случае, если изображение не доступно во время заполнения. По умолчанию этот флаг установлен в false. Если установлено значение true, изображение сохраняется во время заполнения вместо самого изображения, и в процессе экспортирования изображение будет загружено с места для чтения с пути.
- По разным причинам, изображение может быть недоступно, когда обработчик пытается загрузить его либо в отчете заполнения или во время экспорта, особенно если изображение загружается из какого-то публичного URL. По этой причине вы можете настроить обработчик, обрабатывающий отсутствующие изображения во время создания отчета. Атрибут OnErrorType для изображений позволяет это (необязателен). Он может принимать следующие значения:
  - Ошибка: Возникает исключение, если обработчик не может загрузить изображение (OnError TypeError ").
  - Бланк: Любое исключение image-loading игнорируется, и ничего не будет отображаться в
    созданном документе (onErrorType="Blank").
  - Значок: Если изображение не загружается успешно, то обработчик поставит небольшой значок в документе, чтобы указать, что фактическое изображение отсутствует (onErrorType="Icon").
- Как и в случае с текстовыми полями, вы можете отложить вычисление выражения изображения, которое по умолчанию выполняется немедленно. Это позволяет отображать в документе изображения, которые будут построены или выбраны позднее в процессе заполнения отчета. Атрибут evaluationTime (по умолчанию – “Now”) может принимать следующие значения:
    - Непосредственной оценки: изображения выражение вычисляется, когда заполняется текущий диапазон (evaluationTime="Now").
    - Конец отчета оценки: изображения выражение вычисляется при достижении конца отчета (evaluationTime="Report").
    - Конец страницы оценка: изображения выражение вычисляется при достижении конца текущей страницы (evaluationTime="Page").
    - Конца столбца оценки: по достижении конца текущего столбца вычисляется выражение изображения (evaluationTime="Column").

    - Конец группы оценки: изображения выражение вычисляется при группе путем изменения атрибута evaluationGroup (evaluationTime="Group").
    - Авто оценки: каждой переменной, участвующих в выражении изображения оценивается в то время, соответствующий типу его сброс. В настоящее время оцениваются поля (evaluationTime = «Auto»).

- evaluationGroup - группа участвует в процессе оценки изображения, когда атрибут оценки времени устанавливается в группе. Необязателен для заполнения.

Image Expression

Значение, возвращаемое image expression, является источником для изображения, которое будет отображаться. Image expression вводится с помощью элемента `<imageExpression />` и могут возвращать значения только из ограниченного диапазона классов, перечисленных следующим образом:

- java.lang.String
- java.io.File
- java.net.URL
- java.io.InputStream
- java.awt.Image
- net.sf.jasperreports.engine.JRRenderable

```
Image Examples

  <image scaleImage="Clip" onErrorType="Icon" isLazy="true">
    <reportElement x="0" y="0" width="150" height="40"/>
    <imageExpression
class="java.lang.String"><![CDATA["http://jasperreports.sourceforg
e.net/jasperreports.png"]]></imageExpression>
  </image>
```

Все атрибуты, напротив которых указано #REQUIRED, являются обязательными для заполнения