BIRT Report Object Model – Base Elements

Functional Specification

Draft 2: April 26, 2005

Abstract

Describes the base elements for many of the BIRT components. Report Element is the base element for all elements, while Report Item is the base element for all visual report items.

Document Revisions

Version	Date	Description of Changes
Draft 2	04/26/2005	Properties changes for Report element: Extends moved to "After the first release". Comments moved to "First Release"
Draft 1	11/29/2004	First BIRT release

Contents

1. Introduction

This specification defines the two elements at the root of the report element hierarchy. The Report Element is the base element for anything that major components that make up a report. A report item has a name, properties and so on. The report item element is the base element for all visual report items and includes data and style information.

This specification also defines a number of structures used by these two base items.

2. Design Element

2.1 Property Element Types

ROM defines a number of property elements:

`property`

The value of most simple parameters. ROM determines the type of the value by consulting a set of property meta-data.

```
<property name="propName">value</property>
```

`xml-property`

The value of a property that contains custom XML. This property is often used by extensions to store its information.

```
<xml-property name="propName">value</xml-property>
```

`property-list`

For any property that has a list of properties such as custom colors, bindings, and so on.

```
<property-list name="propName">
   [ <structure>…</structure> ] *
</property-list>
```

`expression`

For any property defined by an expression. (Intended for future use.)

```
<expression name="propName">value</expression>
```

`ex-property`

An extended property mostly for use with extensions that don't have well-formed property names:

```
<ex-property
   <name>propName</name>
   <value>value</value>
</ex-property>
```

`structure`

For a property defined by a structure (a collection of two or more properties.)

```
<structure name="propName">
   <property name="member1">value1</property>
   <property name="member2">value2</property>
</structure>
```

2.2 Design Element

Properties

`methods[ ]`

An associative array of method names. The method name is the key into the array. Returns a String that holds the method text, or null if no method is defined.

`propName`

Provides the value of a property given a property name. Design Element properties appear directly as properties on the DesignElementDefn JavaScript object. Used to access all properties: those defined by ROM, those added though extensions, and user-defined properties. The type of the returned value depends on the property definition. Returns null if the property is not set.

2.2.1 `methods` Property

List of methods.

Synopsis

```
DesignElementDefn.methods[ ]
```

Runtime Scripting

`element.methodName`

Summary

Display Name: Methods

JavaScript Type: Array of String

Default value: None

Inherited: Yes

Settable at runtime: No

Availability: First release

XML Summary

Methods are defined in XML using the method element:

<method name="*methodName*">*code*</method>

The method name must match one of the methods defined in this specification. The code can be any valid JavaScript script.

Description

List of methods defined on the element. The array is indexed by method name. The method code appears inside the array.

At design time, the methods are simply a list of JavaScript scripts. At design time, the methods are actual JavaScript methods on the runtime object.

Methods are designed to be called by the BIRT Report Engine at interesting events. They are usually not meant to be called by user-defined scripts.

The methods available to an element are described on the derived elements.

3. Element Definition Structures

The following property structures are used by Design Element.

3.1 User Property Definition Structure

Properties

`choices`

A list of available choices if the property type is Choice.

3.1.1 `choices` Property

List of choices for a user-defined property.

Synopsis

```
UserProperty.choices[]
```

Summary

Display Name: Choices

ROM Type: List of Choice Structures

JavaScript Type: Array of `ChoiceDefn`

Availability: First release

Description

Provides a list of choices for the property as a list of Value Choice elements. The list provides a set of choices from which the user can select, or the user can enter some other value not on the list. Required if the property is a choice, optional otherwise.

Choices give the end user a set of pre-defined value choices. Choices can be open or close-ended. Open-ended choices are suggestions: the user can choose one of the choices, or can enter some other value. Closed-ended choices are the only legal values. The developer indicates a set of closed-ended choices by setting the property type to “choice.” For all other property types, the choice list is open-ended.

3.2 Property Choice Structure

Defines a choice for a user-defined property.

Properties

`displayName`

The value to display to the developer in the UI. Can be externalized. If omitted, then the value is displayed.

`value`

The value to use internally to represent this choice.

Description

User-defined properties can provide the user with a list of choices. The choices can be externalized and translated. The value is required; the display name is optional.

3.2.1 `displayName` Property

The value to display to the developer in the UI.

Summary

Display Name: Display Name

ROM Type: Text structure

Default: value

Availability: First release

Description

Provides a display name to appear for the choice in the GUI. The display name can be externalized and translated. The display name is optional. If omitted, the value is displayed instead.

See Also

Static Text ROM data type

3.2.2 `value` Property

The value to use internally to represent this choice.

Summary

Display Name: Value

ROM Type: Any. (Must match the type of the user-defined property.)

JavaScript Type: any

Required.

Availability: First release

Description

Each choice represents a discrete value. For example, if the property represents a list of customer types, the display names may be “Commercial”, “Residential”, and “Government”, while the values may be “C”, “R” and “G”.

4. Report Element

4.1 Report Element

Summary

JavaScript Design-time Object: `ReportElementDefn (extends DesignElementDefn)`

JavaScript Runtime Object: `ReportElement (extends DesignElement)`

4.1.1 `displayName` Property

Summary

ROM Type: Text structure

5. Report Item

5.1 Report Items

ROM reports contain *report items* that display specific data values, images, labels, and so on. ROM defines the following report items defined in later sections:

- **Label:** displays static text that can be translated into multiple languages.
- **Data item:** displays data from database columns, expressions and so on.
- **Text item:** displays text with embedded formatting directives, and that typically includes placeholders to indicate where to merge in database data. (Similar to an e.RD-Pro rich text control.)
- **Image:** displays a variety of formats.
- **Line and rectangle:** provides graphics within a report. (Not in the first release.)
- **Chart:** display business graphics such as line charts, pie charts, and so on.
- **Matrix:** present a tabular view of data. (Not in the first release.)
- **Table-of-contents item:** lets the user insert a table of contents directly into the report. (Not in the first release.)
- **Browser control:** allows the developer to create custom web-page content. (Not in the first release.)
- **List:** provides a typical banded report.
- **Table:** banded report organized into rows and columns.
- **Free-form:** holds an arbitrary collection of report items at arbitrary (x, y) positions. (Not in the first release.)
- **Grid:** provide a static, table-oriented layout of report items.
- **Include:** imports another design file into the report. (Not in the first release.)
- **Extended item:** allows the developer to specify his own custom item.

5.2 Report Item Element

Summary

Design object: `ReportItemDesign`

Runtime object: ReportItem

5.2.1 `dataSet` Property

Data Set Usage

The following table shows the use of the data set name for each type of section.

Report Item	Data Set Name Usage
List	The data set that provides the data for the groups and detail sections.
Table	

Report Item	Data Set Name Usage
Free-form Grid	Provides data to the contents of the container. If the contents work with a set of rows (chart, list, etc.), then each item reads the entire data set. If the item works with a single row (data item), then the item displays the first (or only) row from the data set.
Text	Provides data for any data expressions in the text. The expressions show values from the first (or only) row in the data set.
Matrix	The matrix displays data from the data set.
Chart	The chart displays data from the data set.
TOC Include Label	Ignored.
Image	Used if the image is obtained from the data set, otherwise ignored.
Browser Control	TBD
Extended Item	Depends on the implementation of the item.

The meaning for list, table, matrix and chart is clear, because each of these works with a set of rows. The meaning for Free-form, Grid and Text requires a bit more explanation.

A text section can display static text and/or data expressions. The data expression can refer to data in the data set. Since the text item appears only once, it can display data from only one row in the data set. BIRT defines that row to be the first row of the data set. A common usage is to display data looked up from a database. For example, suppose that the report is an invoice. The header of the invoice might display the invoice number and customer information. The report may take a parameter that provides the invoice number. The data set associated with the title would look up the invoice record, and join it to the customer table. The result should be a single row, and the data from this row is what appears in the report title.

The free-form and grid items both contain other report items. If those items are simple items such as a data item or text item, then these items follow the same rule discussed above: they display the first row from the data set. However, items that can display multiple rows (such as a chart, list, matrix, etc.) display all rows from the data set. The effect is as though each item makes a separate pass over the data set. This structure is useful for a section that displays the same data multiple ways. For example, a section may show the top 10 sales as a list then show them again as a pie chart.