# [MS-RGDI-Diff]:

# Remote GDI+ (RGDI) Binary Stream Format

---

## Revision Summary

| Date | Revision History | Revision Class | Comments |
|---|---|---|---|
| 8/7/2009 | 0.1 | Major | First release. |
| 11/6/2009 | 0.1.1 | Editorial | Changed language and formatting in the technical content. |
| 3/5/2010 | 0.2 | Minor | Clarified the meaning of the technical content. |
| 4/21/2010 | 0.2.1 | Editorial | Changed language and formatting in the technical content. |
| 6/4/2010 | 0.2.2 | Editorial | Changed language and formatting in the technical content. |
| 9/3/2010 | 0.2.3 | Editorial | Changed language and formatting in the technical content. |
| 2/9/2011 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 7/7/2011 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 11/3/2011 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 1/19/2012 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 2/23/2012 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 3/27/2012 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 5/24/2012 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 6/29/2012 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 7/16/2012 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 10/8/2012 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 10/23/2012 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 3/26/2013 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 6/11/2013 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 8/8/2013 | 0.2.3 | None | No changes to the meaning, language, or formatting of the technical content. |
| 12/5/2013 | 1.0 | Major | Updated and revised the technical content. |
| 2/11/2014 | 2.0 | Major | Updated and revised the technical content. |
| 5/20/2014 | 2.0 | None | No changes to the meaning, language, or formatting of the |

| Date | Revision History | Revision Class | Comments |
|---|---|---|---|
| | | | technical content. |
| 5/10/2016 | 3.0 | Major | Significantly changed the technical content. |
| 8/16/2017 | 4.0 | Major | Significantly changed the technical content. |
| 10/16/2019 | 5.0 | Major | Significantly changed the technical content. |

# Table of Contents

# 1 Introduction

The Remote GDI+ (RGDI) binary stream format is a binary format that is produced by Microsoft SQL Server Reporting Services when communicating with viewer controls to offload some of the rendering work from the server to the client viewer control. By using RGDI, the implementer can create custom report designers that will generate RGDI and can create custom report renderers that process and display RGDI to display reports.

A client control that consumes RGDI needs to translate RGDI content to GDI+ objects and calls. RGDI works with only one page of report data at a time; each page is represented by an RGDI stream.

Sections 1.7 and 2 of this specification are normative. All other sections and examples in this specification are informative.

## 1.1 Glossary

This document uses the following terms:

**action**: An interactivity event in a report, such as a hyperlink, bookmark link, or drillthrough link, that is associated with an item in a report.

**Augmented Backus-Naur Form (ABNF)**: A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [RFC5234].

**big-endian**: Multiple-byte values that are byte-ordered with the most significant byte stored in the memory location with the lowest address.

**bookmark**: An anchor that is used in a report to assist navigation, typically through the use of hyperlinks. A bookmark link in a report sends the user to another location in the report.

**brush**: An object that is used to fill the interior of graphical shapes such as rectangles, ellipses, pies, polygons, and paths. A brush specifies a color with which to fill the shape.

**fixed header**: A column header or row header of a table or a matrix that remains displayed on a page even when a user scrolls part of the table or matrix off the page.

**font**: An object that defines the graphic design, or formatting, of a collection of numbers, symbols, and letters. A font specifies the style (such as bold and strikeout), size, family (a typeface such as Times New Roman), and other qualities to describe how the collection is drawn.

**format**: A set of flags that encapsulates text layout information such as alignment, text direction, and trimming.

**hierarchy**: A logical tree structure that organizes the members of a dimension such that each member has one parent member and zero or more child members.

**interactivity block**: A structure that contains end-user interactivity data that is encoded in an XML document. The interactivity information includes the destinations of labels and bookmarks to jump to. It includes actions that specify a bookmark link, hyperlink, drillthrough, toggle, or sort. An interactivity block also includes information about how to render fixed headers for tables and matrices.

**label**: A text string that identifies a report item within the client UI in order to provide a user-friendly name for searching.

**little-endian**: Multiple-byte values that are byte-ordered with the least significant byte stored in the memory location with the lowest address.

**matrix**: A report item on a report layout that displays data in a variable columnar format.

**pen**: An object that is used to draw lines and curves. A pen has a color, a width, and a style. The style is a solid line, a dashed line, or a dotted line.

**report**: An object that is a combination of three kinds of information: data or other kinds of information about how to obtain the data (queries) as well as the structure of the data; layout or formatting information that describes how the data is presented; and properties of the report, such as author of the report, report parameters, and images included in the report.

**report item**: An object that exists on a report layout.

**RGB color**: The specification of a color in terms of the values of its red, green, and blue components.

**RGDI**: Remote GDI+ binary stream format. A structured stream specification that is used to specify the rendering of a page of a report as a collection of GDI+ function calls.

**shareable object**: A structure that contains an inline font, format, or image object that is not shared, or a structure that contains a reference to a shared font, format, or image object in a shared object structure. A shareable object that contains a reference to an object saves space compared to a shareable object that contains an inline object.

**shared object**: A structure that contains a font, format, or image object, and that specifies a unique integer identifier that can be used to refer to the object. This means the object can be defined once and used many times. This saves space.

**sort action**: An action that sorts data in ascending or descending order and that causes the page to be re-rendered.

**stream**: A sequence of bytes written to a file on the target file system. Every file stored on a volume that uses the file system contains at least one stream, which is normally used to store the primary contents of the file. Additional streams within the file can be used to store file attributes, application parameters, or other information specific to that file. Every file has a default data stream, which is unnamed by default. That data stream, and any other data stream associated with a file, can optionally be named.

**table**: A data region on a report layout that displays data in a columnar format.

**tablix**: A data region that contains rows and columns that resembles a table or matrix, possibly sharing characteristics of both.

**toggle action**: An action that results in a part of the report becoming hidden or visible. Typically, a toggle action is used to hide detail data to show only summary data.

**useable area**: The area that remains on a page after the margins, page header, and page footer are removed.

**virtual structure**: A structure that specifies the structure types that can be used to define a field in the parent structure.

**MAY, SHOULD, MUST, SHOULD NOT, MUST NOT:** These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

## 1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

### 1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[RFC2781] Hoffman, P., and Yergeau, F., "UTF-16, an encoding of ISO 10646", RFC 2781, February 2000, http://www.rfc-editor.org/rfc/rfc2781.txt

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, http://www.rfc-editor.org/rfc/rfc5234.txt

[XML10/5] Bray, T., Paoli, J., Sperberg-McQueen, C.M., et al., Eds., "Extensible Markup Language (XML) 1.0 (Fifth Edition)", W3C Recommendation, November 2008, http://www.w3.org/TR/2008/REC-xml-20081126/

### 1.2.2 (Updated Section) Informative References

[MS-RDL] Microsoft Corporation, "Report Definition Language File Format".

[MS-RSWSRE2005] Microsoft Corporation, "Report Server Web Service for Report Execution: ReportExecution2005".

[MSDN-GDI+] Microsoft Corporation, "GDI+", ~~http://msdn~~https://docs.microsoft.com/en-us/~~library/ms533798.aspx~~windows/desktop/gdiplus/-gdiplus-gdi-start

[MSDN-StreamClass] Microsoft Corporation, "Stream Class", ~~http://msdn~~https://docs.microsoft.com/en-us/~~library~~dotnet/api/system.io.stream~~.aspx~~

## 1.3 Overview

This document specifies the stream format for Remote GDI+ (RGDI), a binary stream format that is used to represent metadata for graphically rendering a single page of a report. Binary stream formats are discussed in [MSDN-StreamClass]. The stream is described by a collection of nested structures. A structure can contain information about the type of the structure as well as type-specific fields that further specify the structure content. Additionally, some structures specify information that is encoded in an XML document, as specified in [XML10/5]. The structures of these XML documents are described in this document.

For a more detailed overview of specific stream architecture and content, see section 2.

### 1.3.1 Remote GDI+ Binary Stream Format Content

The Remote GDI+ (RGDI) binary stream format consists primarily of the following kinds of information:

- **Report rendering layout**: Instructions that specify which GDI+ [MSDN-GDI+] graphics functions to call and what their arguments are. This information is used to render a page of a report. RGDI renders by using calls to six GDI+ functions. These functions are **DrawString**, **DrawRectangle**, **FillRectangle**, **DrawLine**, **FillPolygon**, and **DrawImage**.
- **End-user interactivity data**: Information about the kinds of end-user interactivity to make available in the rendered report, including label and bookmark destinations and actions that specify a bookmark link, hyperlink, drillthrough, toggle action, or sort action. Information about how to render fixed headers for a table, a matrix, or a tablix is also included in the end-user interactivity data.

### 1.3.2 Byte Ordering

Some computer architectures order bytes in a binary word from left to right, which is called big-endian. This documentation uses big-endian bit diagrams. Other architectures order bytes in a binary word from right to left, which is called little-endian. The underlying stream format structures and fields are little-endian.

Using big-endian and little-endian methods, the number 0x12345678 would be stored as shown in the following table.

| Byte order | Byte 0 | Byte 1 | Byte 2 | Byte 3 |
|---|---|---|---|---|
| Big-endian | 0x12 | 0x34 | 0x56 | 0x78 |
| Little-endian | 0x78 | 0x56 | 0x34 | 0x12 |

Unless otherwise specified, all data in files of the type that this document specifies are stored in little-endian format.

### 1.3.3 Organization of this Document

Section 2 of this document contains overviews of high-level concepts that are followed by more detailed concepts. Section 2.1, in particular, specifies higher-level concepts that are required to understand the remainder of the document. The user is advised to read this section before reading the remainder of section 2.

Section 2.1 specifies the structures and concepts that are used to organize and structure an RGDI stream.

Section 2.2 specifies the details of each RGDI structure that can be included in an RGDI stream.

Section 2.3 specifies the details of each XML element that can be included in an RGDI stream.

Section 3 provides specific examples that illustrate the concepts, RGDI structures, and XML elements of the RGDI stream format.

Section 4 does not apply to RGDI.

Section 5 is a list of version-specific behaviors. This section is not intended to be read by itself, but is to be understood in the context of the specifications in section 2. Specifications in section 2 provide links to relevant items in section 5.

## 1.4 Relationship to Protocols and Other Structures

The Remote GDI+ (RGDI) binary stream format is a stream that uses UTF-16 encoding of Unicode characters as specified in [RFC2781].

In addition to other file formats, RGDI can be used as a protocol for reports that are created in Report Definition Language (RDL), as described in [MS-RDL].

The two protocols that carry RGDI are HTTP and the Report Server ReportExecution web service [MS-RSWSRE2005].

## 1.5 Applicability Statement

The Remote GDI+ (RGDI) binary stream format is applicable for use as a protocol to allow custom report designers to create rendering instructions in RGDI and to allow custom report renderers to process and display these rendering instructions to display reports.

## 1.6 Versioning and Localization

This document describes versioning issues in the following areas:

**Structure Versions**: The following versions of Reporting Services can produce and consume the Remote GDI+ (RGDI) binary stream format:

- Microsoft SQL Server 2005 Reporting Services
- Microsoft SQL Server 2008 Reporting Services
- Microsoft SQL Server 2008 R2 Reporting Services
- Microsoft SQL Server 2012 Reporting Services
- Microsoft SQL Server 2014 Reporting Services
- Microsoft SQL Server 2016 Reporting Services
- Microsoft SQL Server 2017 Reporting Services

**Localization**: There are no localization-dependent structures in the RGDI binary stream format.

## 1.7 Vendor-Extensible Fields

None.

# 2 Structures

## 2.1 Introduction

RGDI is a binary stream format that is composed of a collection of nested RGDI structures (described in section 2.2). A structure can contain information about the type of the structure as well as type-specific fields that further specify the structure content.

This section describes each RGDI structure, its structure type, the fields contained within the structure, and the structures that are contained within those fields. Additionally, some fields contain information that is encoded in XML documents. This section describes the full structure of these XML documents.

Each RGDI binary stream that contains the report rendering information that is required to describe a page of a report MUST be written as a series of nested RGDI structures that preserve the report hierarchy. The left and top properties of an item are absolute measurements in the useable area of the page. To optimize space, some structures are shared between report items so that they are written only once.

This document uses Augmented Backus-Naur Form (ABNF) as specified in [RFC5234] to specify the sequence of fields for RGDI structures.

### 2.1.1 Simple Data Type Structures

| Data type | Description format |
|---|---|
| Byte | Represents an 8-bit (1-byte) byte. |
| UInt16 | Represents a 16-bit (2-byte) unsigned short integer. |
| Int32 | Represents a 32-bit (4-byte) signed integer. |
| Float | Represents a 32-bit (4-byte) single-precision floating point value. |
| Boolean | Represents an 8-bit (1-byte) logical **Boolean** type value. The valid values are true (1) and false (0). |
| String | All strings in the protocol MUST be UNICODE UTF-16. By default, all strings start with the length of the string. Strings are represented in the protocol as an array of bytes, and the number of bytes MUST be equal to the number of characters in the string multiplied by two. |

#### 2.1.1.1 Writing Strings to an RGDI Stream

Strings are written as a length-prefixed string to the RGDI stream by using UTF-16 encoding as specified in [RFC2781]. After the length of the strings is written, the current position of the stream is advanced in accordance with the encoding and with the specific characters that are being written to the stream.

#### 2.1.1.2 Writing the Length Value of a String to an RGDI Stream

The length of a string is written to the RGDI stream seven bits at a time, starting with the seven least-significant bits. The high bit of a byte indicates whether there are more bytes to be written after the current byte.

If a value fits in seven bits, it takes only one byte of space. If a value does not fit in seven bits, the high bit is set on the first byte and is written out. The value is then shifted by seven bits and the next byte is written.

The process of writing to the RGDI stream seven bits at a time is repeated until the entire integer has been written.

#### 2.1.1.3 Reading Strings from an RGDI stream

Strings are prefixed by their length, which is an integer that is encoded by using the rules that are specified in section 2.1.1.2.

## 2.2 RGDI Structures

This section defines RGDI structures and their associated grammar. Every structure includes the following items:

- A definition of the structure.
- The ABNF grammar for the structure.
- A bit diagram of the structure.
- Definitions of all fields contained within the structure, including the sizes of all related structures. Fields following a variable length field can be unaligned; that is, these fields can start on an even or an odd byte boundary from the beginning of the structure.

An entire RGDI stream is represented by the top-level (root) structure, which is the **Stream** structure. This structure contains a sequence of **Structure** structures that correspond to the top-level report items in a report. Within these **Structure** structures are the GDI+ function calls and arguments that specify how the corresponding report item of the report is to be rendered.

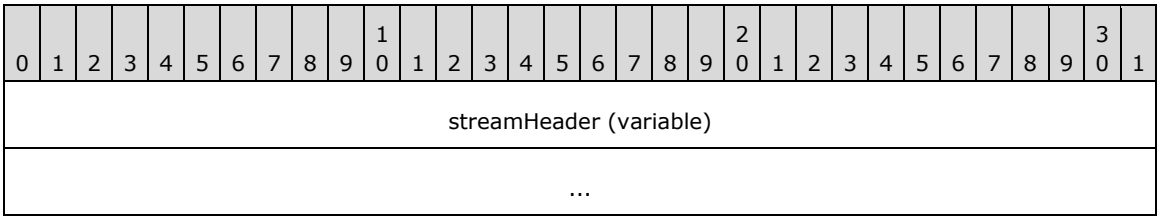
In addition to GDI+ function calls, the **Structure** structures can contain shared objects, which serve as arguments to the GDI+ functions. **Structure** structures can also contain nested **Structure** structures, which correspond to rendering instructions for child report items of a parent report item within the report item hierarchy.

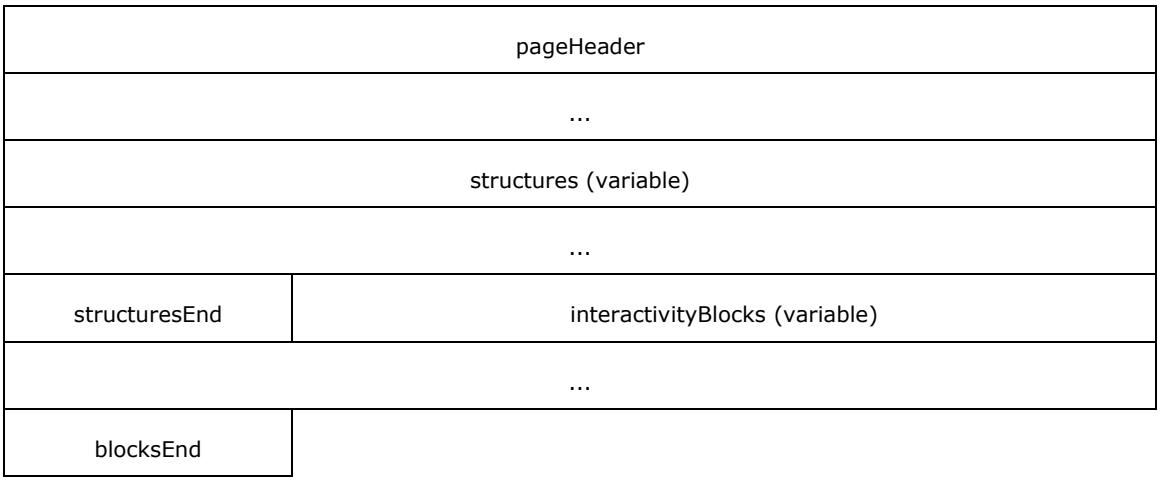
Following the sequence of **Structure** structures are zero or more **InteractivityBlock** structures. The interactivity blocks describe the end-user interactivity data in the report such as hyperlinks, bookmark links, drillthrough links, sort actions, toggle actions, or fixed header information that is associated with an item in a report. The content of the interactivity blocks is in a byte array, which contains an XML document. The structures of these XML documents are defined in section 2.3.

### 2.2.1 Stream

The **Stream** structure is the root structure of the Remote GDI+ (RGDI) stream, and it specifies the contents of the stream in its entirety. There MUST be only one **Stream** structure in the stream.

```
Stream = StreamHeader PageHeader *Structure structuresEnd *InteractivityBlock blocksEnd
```






**streamHeader (variable)**: A StreamHeader structure that specifies the type of the stream and version information.

**pageHeader (8 bytes)**: A PageHeader structure that specifies the width and height of a page of the report.

**structures (variable)**: An array of Structure structures that specifies the objects of the report to be rendered.

**structuresEnd (1 byte)**: A byte field that specifies a delimiter to mark the end of the structures part of the **Stream** structure and to mark the beginning of the interactivity blocks. The value of this field MUST be "0xFF".

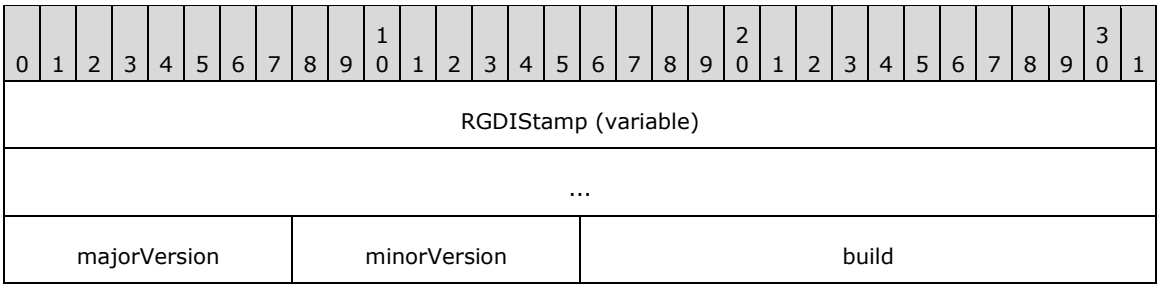
**interactivityBlocks (variable)**: An array of InteractivityBlock structures that specify the actions, bookmarks, labels, and fixed headers on a report page. There MUST be no more than one **InteractivityBlock** structure for each of the four types of interactivity blocks: actions, bookmarks, labels, and fixed headers.<1>

**blocksEnd (1 byte)**: A byte field that specifies a delimiter to mark the end of the interactivity blocks part of the **Stream** structure and to mark the end of the RGDI stream. The value of this field MUST be "0xFF".

### 2.2.2 StreamHeader

The **StreamHeader** structure is the header structure of the Stream structure. The **StreamHeader** structure specifies the type of the stream and RGDI version information.

```
StreamHeader = RGDIStamp majorVersion minorVersion build
```



| |
|---|
| ... |

**RGDIStamp (variable)**: A field of type **String** that specifies the type of the stream. The value of this field MUST be "RGDI".

**majorVersion (1 byte)**: A byte field that represents the major version of RGDI that is specified in the stream. The value of this field MUST be "0x0A".

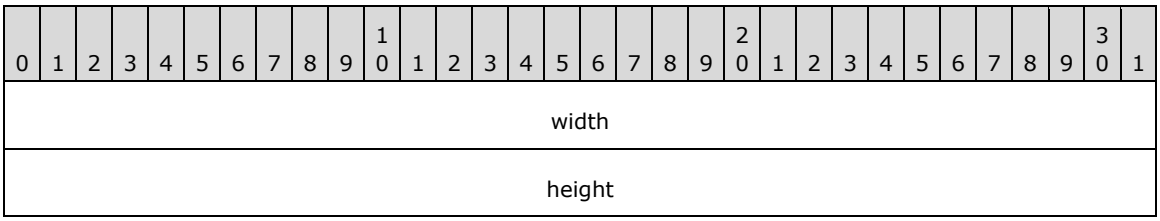
**minorVersion (1 byte)**: A byte field that represents the minor version of RGDI that is specified in the stream. The value of this field MUST be "0x00".

**build (4 bytes)**: An **Int32** field that represents the build version of RGDI that is specified in the stream. The value of this field MUST be "0x00000001".

### 2.2.3 PageHeader

The **PageHeader** structure specifies the width and height of a page of a report.

```
PageHeader = width height
```



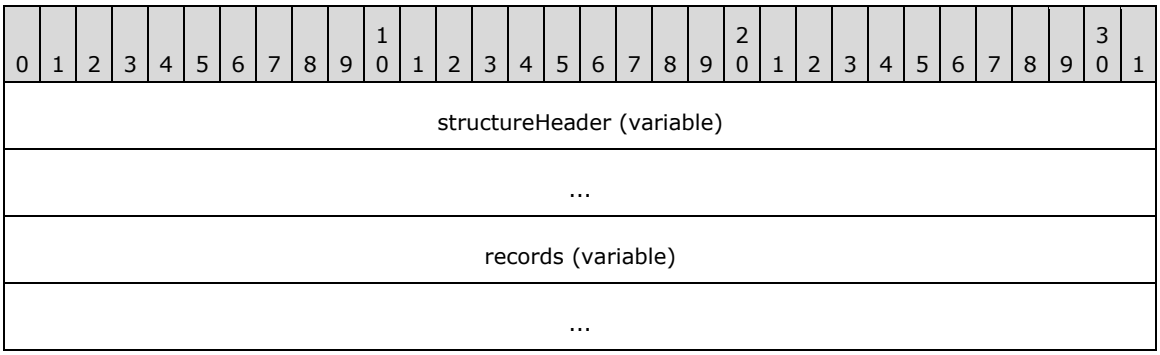

**width (4 bytes)**: A **Float** value field that specifies the width, in millimeters, of a physical page of the report, including the margins. The value of this field MUST be a non-negative **Float**.<2>

**height (4 bytes)**: A **Float** value field that specifies the height, in millimeters, of a physical page of the report, including the margins. The value of this field MUST be a non-negative **Float**.<3>

### 2.2.4 Structure

The **Structure** structure specifies a rendered report item in the stream. This structure consists of a structure header, followed by zero or more Record structures, followed by a structure end tag.

```
Structure = StructureHeader *Record structureEnd
```



| structureEnd |
|---|

**structureHeader (variable)**: A StructureHeader structure that specifies the structure type that corresponds to the following items:

- Report item types.
- A unique name for the structure.
- The location of the structure on the page.
- The size of the structure.

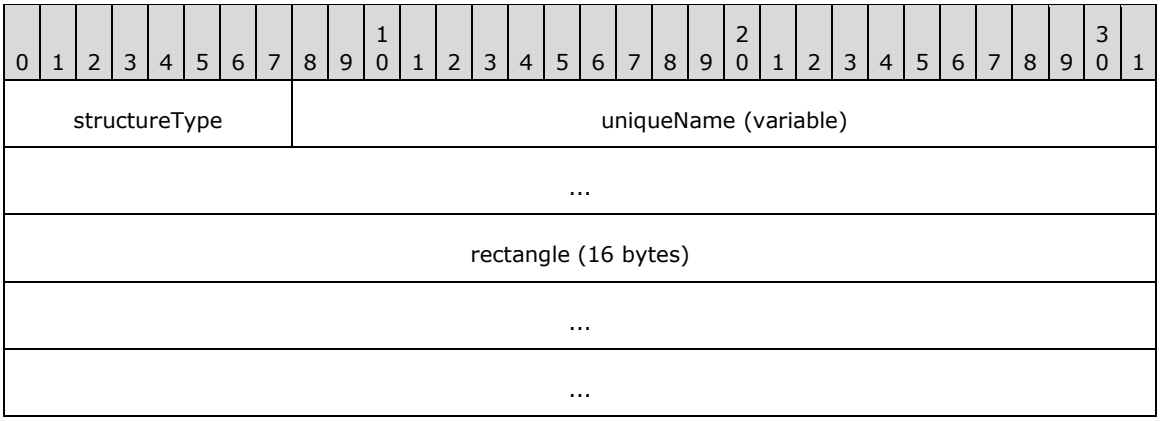
**records (variable)**: A collection of **Record** structures that specify the rendered objects for the report item.

**structureEnd (1 byte)**: A delimiter byte that specifies the end of the structure. The value of this field MUST be "0xFF".

### 2.2.5 StructureHeader

The **StructureHeader** structure specifies the structure type corresponding to a report item type, a unique name for the structure, and a rectangle that specifies the position and size of the report item on the page.

```
StructureHeader = structureType uniqueName Rectangle
```




**structureType (1 byte)**: A byte field that specifies the type of the structure. The value of this field MUST be one of the report item types in the following table.

| Value | Report item type |
|---|---|
| 0x00 | Textbox |
| 0x01 | Line |
| 0x02 | Image |
| 0x03 | Rectangle |
| 0x04 | Chart, GaugePanel, or Map<4> |

| Value | Report item type |
|---|---|
| 0x05 | List |
| 0x06 | Table |
| 0x07 | Matrix or Tablix<5> |
| 0x08 | Subreport |

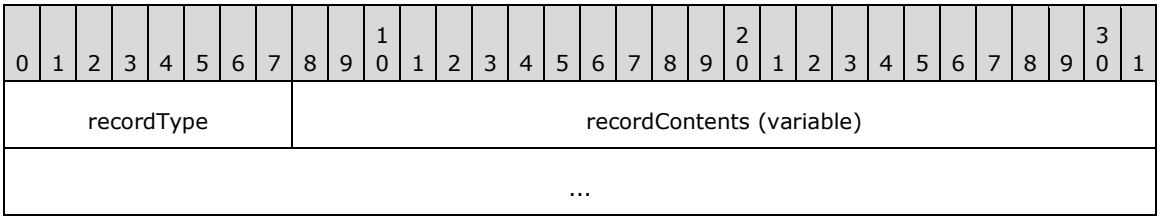
**uniqueName (variable)**: A field of type **String** that specifies the unique name of the specified report item.

**rectangle (16 bytes)**: A Rectangle structure that specifies the position and size of the report item on the page. The rectangle is measured in millimeters.<6>

### 2.2.6 Record

The **Record** structure specifies nested Structure structures for child report items of the report item of the parent **Structure** structure that contains the **Record**, GDI+ drawing functions, and shared objects that are used as arguments for the GDI+ drawing functions.

```
Record = recordType RecordContents
```




**recordType (1 byte):** A byte field that specifies the type of the **Record** structure. The type specified in this field MUST match the type of the RecordContents structure. The value of **recordType** MUST be one of the **Record** structure types in the following table.

| Value | Record Type | RecordContents type |
|---|---|---|
| 0x00 | Structure | Structure |
| 0x01 | Function | Function |
| 0x02 | Shared Object | SharedObject |

**recordContents (variable):** A **RecordContents** structure that specifies a nested structure, a GDI+ function call with arguments, or a shared object.

### 2.2.7 RecordContents

The **RecordContents** structure specifies a nested structure, a GDI+ function call with arguments, or a shared object. This is a virtual structure that does not appear in the stream. Instead, a Structure, Function, or SharedObject structure appears in its place.

```
RecordContents = Structure / Function / SharedObject
```

### 2.2.8 Function

The **Function** structure specifies a call to a GDI+ function. This structure includes the name of the function to call and the associated arguments to pass to the GDI+ function. This is a virtual structure that does not appear in the stream. Instead, a DrawString, DrawRectangle, FillRectangle, FillPolygon, or DrawImage structure appears in its place.

The arguments to these GDI+ functions are inline structures, fields, or a reference to a shared object.

```
Function = DrawString / DrawRectangle / FillRectangle / DrawLine / FillPolygon / DrawImage
```

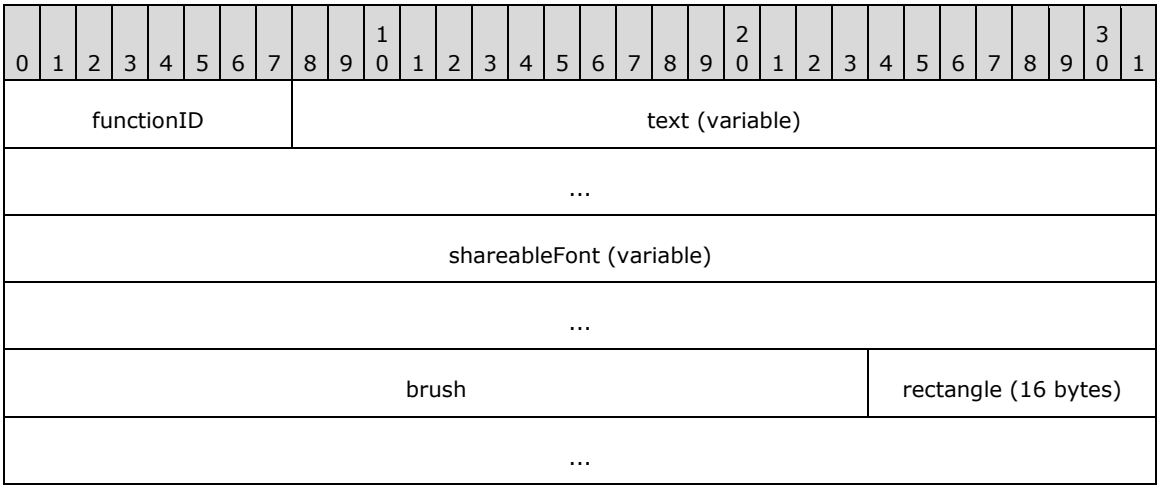
The name of the function to call is specified by the value of the leading byte in the **Function** structure as specified in the following table. The arguments to pass to the GDI+ function are specified in the individual function structure specifications.<7>

| Value | Function name |
|---|---|
| 0x00 | DrawString |
| 0x01 | DrawRectangle |
| 0x02 | FillRectangle |
| 0x03 | DrawLine |
| 0x04 | FillPolygon |
| 0x05 | DrawImage |

### 2.2.9 DrawString

The **DrawString** structure specifies the arguments for the GDI+ function **DrawString**. The arguments are a string, a font, a brush, a rectangle, and a format. The **DrawString** function draws the specified text string in the specified rectangle by using the specified **Brush** and **Font** objects, applying the formatting attributes of the specified string format object.

```
DrawString = functionID text shareableFont Brush Rectangle shareableFormat
```



| ... | |
|---|---|
| ... | shareableFormat (variable) |
| ... | |

**functionID (1 byte):** A byte field that specifies that the type of the function is **DrawString**. The value of this field MUST be "0x00".

**text (variable):** A field of type **String** that specifies the text to be drawn.

**shareableFont (variable):** A ShareableObject structure that specifies either a NonSharedObject structure that contains a Font structure or a UseSharedObject structure that contains a reference to a **Font** structure.

**brush (3 bytes):** A Brush structure that specifies the brush to use to draw the text string.

**rectangle (16 bytes):** A Rectangle structure that specifies the rectangle in which to draw the text string. The rectangle is measured in millimeters.<8>

**shareableFormat (variable):** A **ShareableObject** structure that specifies either a **NonSharedObject** structure that contains a Format structure or a **UseSharedObject** structure that contains a reference to a **Format** structure.

### 2.2.10 DrawRectangle

The **DrawRectangle** structure specifies the arguments for the GDI+ function **DrawRectangle**. The arguments are a pen and a rectangle. The **DrawRectangle** function draws a rectangle specified by a coordinate pair, a width, and a height that are specified by the **rectangle** field.

```
DrawRectangle = functionID Pen Rectangle
```

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 20 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 30 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| functionID | | | | | | | | pen | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | rectangle (16 bytes) | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**functionID (1 byte):** A byte field that specifies that the type of the function is **DrawRectangle**. The value of this field MUST be "0x01".

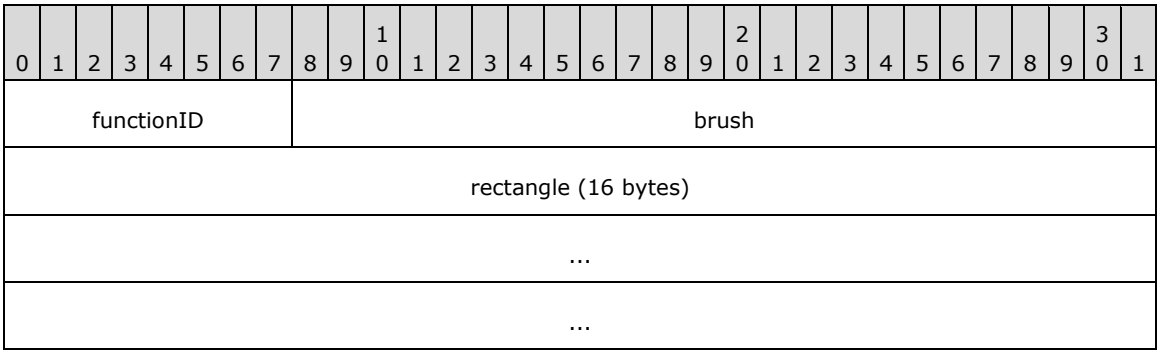
**pen (8 bytes):** A Pen structure that specifies the pen to use to draw the rectangle.

**rectangle (16 bytes):** A Rectangle structure that specifies the rectangle to be drawn. The rectangle is measured in millimeters.<9>

### 2.2.11 FillRectangle

The **FillRectangle** structure specifies the arguments for the GDI+ function **FillRectangle**. The arguments are a brush and a rectangle. The **FillRectangle** function fills the entire interior area of a rectangle as specified by a pair of coordinates, a width, and a height, all of which are specified by the **rectangle** field.

```
FillRectangle = functionID Brush Rectangle
```




**functionID (1 byte):** A byte field that specifies that the type of the function is **FillRectangle**. The value of this field MUST be "0x02".

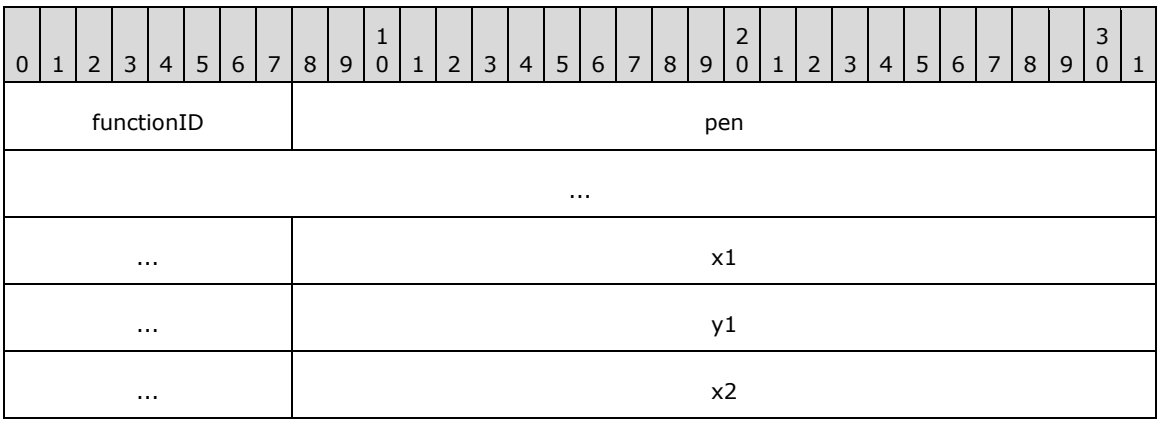
**brush (3 bytes):** A Brush structure that specifies the brush to use to fill the rectangle.

**rectangle (16 bytes):** A Rectangle structure that specifies the rectangle to fill. The rectangle is measured in millimeters.<10>

### 2.2.12 DrawLine

The **DrawLine** structure specifies the arguments for the GDI+ function **DrawLine**. The arguments are a pen and the coordinates of both endpoints of a line. The **DrawLine** function draws a line connecting the two points specified by coordinate pairs.

```
DrawLine = functionID Pen x1 y1 x2 y2
```



| | |
|---|---|
| ... | y2 |
| ... | |

**functionID (1 byte):** A byte field that specifies that the type of the function is **DrawLine**. The value of this field MUST be "0x03".

**pen (8 bytes):** A Pen structure that specifies the pen to use to draw the line.

**x1 (4 bytes):** A **Float** value field that specifies the x-coordinate, or horizontal position, of the first endpoint of the line, expressed in millimeters, relative to the left edge of the page. The value of this field MUST be a non-negative **Float**.<11>

**y1 (4 bytes):** A **Float** value field that specifies the y-coordinate, or vertical position, of the first endpoint of the line, expressed in millimeters, relative to the top edge of the page. The value of this field MUST be a non-negative **Float**.<12>

**x2 (4 bytes):** A **Float** value field that specifies the x-coordinate, or horizontal position, of the second endpoint of the line, expressed in millimeters, relative to the left edge of the page. The value of this field MUST be a non-negative **Float**.<13>

**y2 (4 bytes):** A **Float** value field that specifies the y-coordinate, or vertical position, of the second endpoint of the line, expressed in millimeters, relative to the top edge of the page. The value of this field MUST be a non-negative **Float**.<14>

### 2.2.13 FillPolygon

The **FillPolygon** structure specifies the arguments for the GDI+ function **FillPolygon**. The arguments are a brush and an array of points that specify the vertices. The **FillPolygon** function fills the interior of a polygon that is defined by an array of points specified by Point structures.

```
FillPolygon = functionID Brush PointArray
```

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 20 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 30 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| functionID | | | | | | | | brush | | | | | | | | | | | | | | | | | | | | | | | |
| pointArray (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**functionID (1 byte):** A byte field that specifies that the type of the function is **FillPolygon**. The value of this field MUST be "0x04".

**brush (3 bytes):** A Brush structure that specifies the brush to use to fill the polygon.

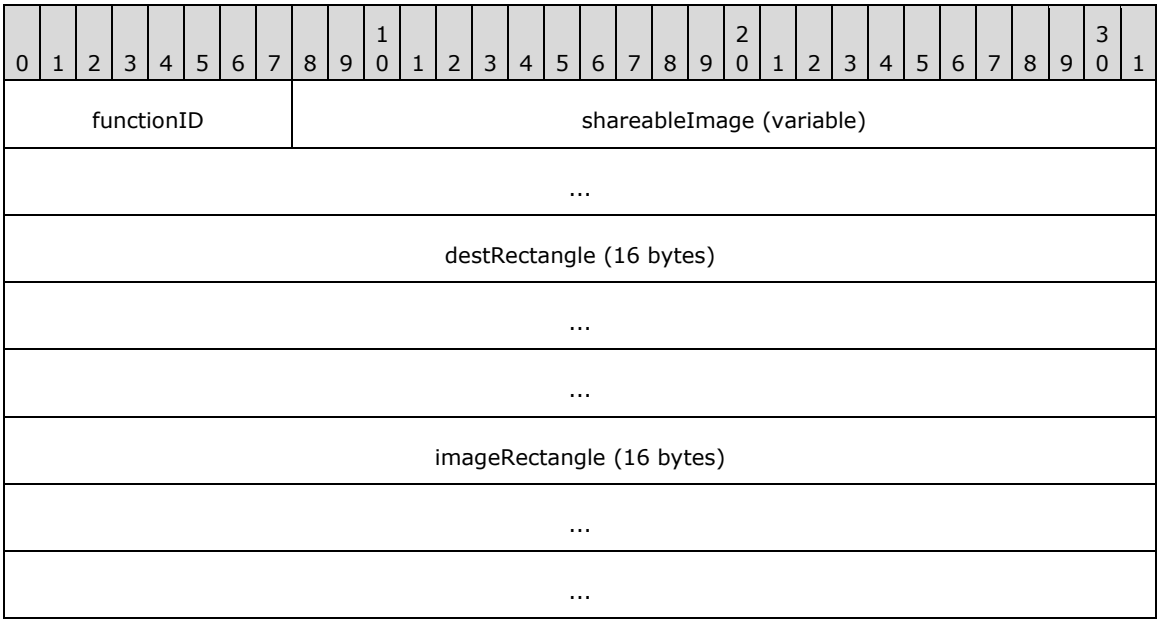
**pointArray (variable):** A PointArray structure that specifies a collection of **Point** structures and the number of items in the collection. The collection specifies the vertices of a polygon.

### 2.2.14 DrawImage

The **DrawImage** structure specifies the arguments for the GDI+ function **DrawImage**. The arguments are an image, a destination rectangle to draw the image into, and an image rectangle from

which to take the image. The **DrawImage** function draws the specified portion of the specified **Image** object at the specified location and with the specified size.

```
DrawImage = functionID shareableImage Rectangle Rectangle
```




**functionID (1 byte):** A byte field that specifies that the type of the function is **DrawImage**. The value of this field MUST be "0x05".

**shareableImage (variable):** A ShareableObject structure that specifies either a NonSharedObject structure that contains an Image structure or a UseSharedObject structure that contains a reference to an **Image** structure.

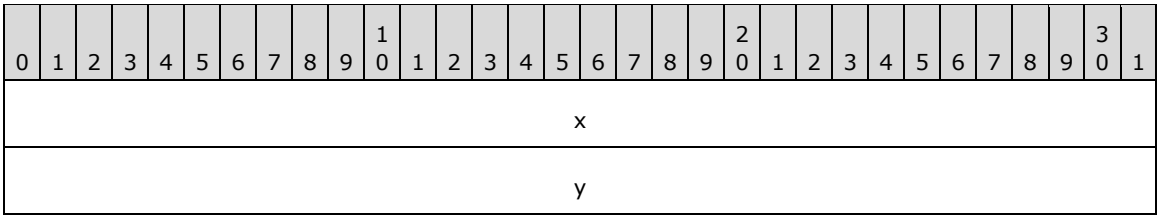
**destRectangle (16 bytes):** A Rectangle structure that specifies the destination rectangle on the page on which to draw the image. The destination rectangle has been clipped to fit the useable area on the page. This rectangle is measured in millimeters.<15>

**imageRectangle (16 bytes):** A **Rectangle** structure that specifies the boundaries of the image, clipped to fit into the usable area on the page. This rectangle is measured in pixels.<16>

### 2.2.15 Point

The **Point** structure specifies the location of a point by using x-coordinates and y-coordinates. The **Point** structure is used to specify a vertex of a polygon for the **FillPolygon** GDI+ function.

```
Point = x y
```



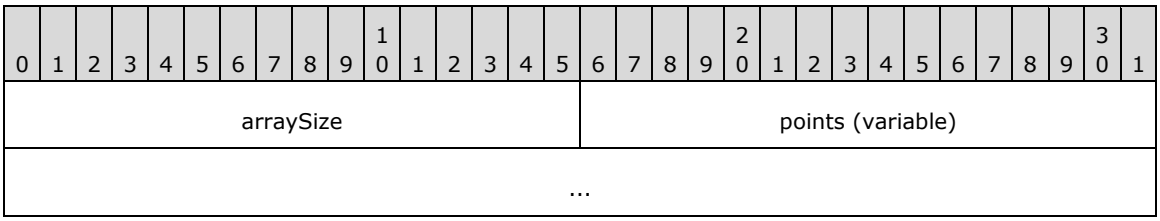
**x (4 bytes):** A **Float** value field that specifies the left position of the point, expressed in millimeters, relative to the left edge of the page. The value of this field MUST be a non-negative **Float**.<17>

**y (4 bytes):** A **Float** value field that specifies the top position of the point, expressed in millimeters, relative to the top edge of the page. The value of this field MUST be a non-negative **Float**.<18>

### 2.2.16 PointArray

The **PointArray** structure specifies an array of points with x- and y-location coordinates. This structure is used to specify the vertices of a polygon for the **FillPolygon** GDI+ function.

```
PointArray = arraySize *Point
```



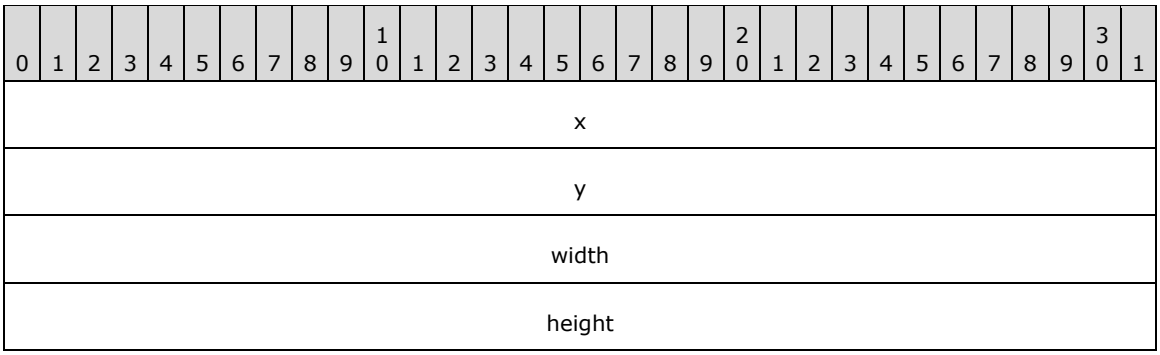

**arraySize (2 bytes):** A **UInt16** value that specifies the number of Point structures in the array.

**points (variable):** An array of **Point** structures. The number of items in the array MUST equal the value of **arraySize**.

### 2.2.17 Rectangle

The **Rectangle** structure specifies the position and size of a rectangle. When this structure is used to specify a rectangle on a page, the fields of the rectangle are expressed in millimeters. When it is used to specify a part of an image that has been clipped, the fields of the rectangle are expressed in pixels. The parent function structure that contains the **Rectangle** structure specifies whether the fields of the rectangle are expressed in millimeters or in pixels.

```
Rectangle = x y width height
```




**x (4 bytes):** A **Float** value field that specifies the x-coordinate of the upper-left corner of the rectangle, expressed in millimeters or pixels, relative to the left edge of the page or image. The value of this field MUST be a non-negative **Float**.<19>

**y (4 bytes):** A **Float** value field that specifies the y-coordinate of the upper-left corner of the rectangle, expressed in millimeters or pixels, relative to the top edge of the page or image. The value of this field MUST be a non-negative **Float**.<20>

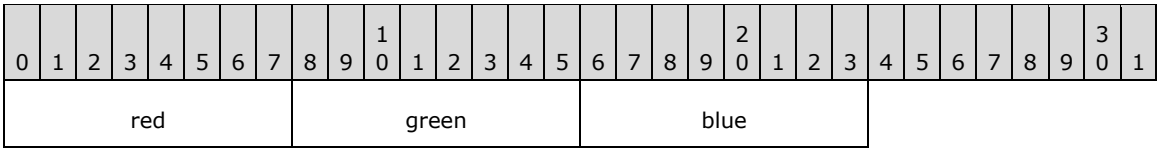
**width (4 bytes):** A **Float** value field that specifies the width of the rectangle, expressed in millimeters or pixels. The value of this field MUST be a non-negative **Float**.<21>

**height (4 bytes):** A **Float** value field that specifies the height of the rectangle, expressed in millimeters or pixels. The value of this field MUST be a non-negative **Float**.<22>

### 2.2.18 Brush

The **Brush** structure specifies the color of a brush that is used to draw GDI+ objects.

```
Brush = red green blue
```




**red (1 byte):** A byte field that specifies the value of the red component of an RGB color.

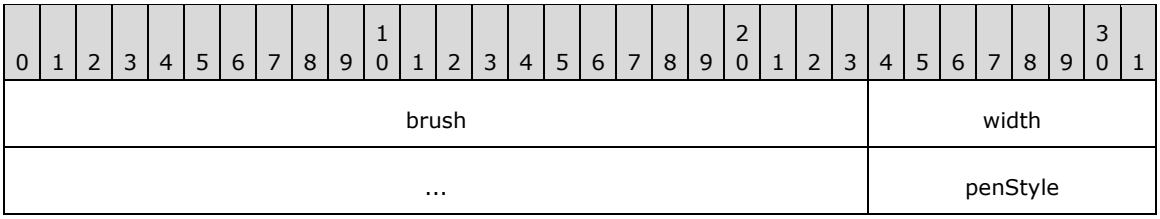
**green (1 byte):** A byte field that specifies the value of the green component of an RGB color.

**blue (1 byte):** A byte field that specifies the value of the blue component of an RGB color.

### 2.2.19 Pen

The **Pen** structure specifies the color, width, and style of a pen that is used to draw GDI+ objects.

```
Pen = Brush width penStyle
```




**brush (3 bytes):** A Brush structure that specifies the color of the pen.

**width (4 bytes):** A **Float** value field that specifies the width, expressed in millimeters, of the pen. The value of this field MUST be a non-negative **Float**.<23>

**penStyle (1 byte):** A byte field that specifies the style of the pen. The value of this field MUST be one of the line styles in the following table.<24>

| Value | Line style |
|---|---|
| 0x00 | Solid |
| 0x01 | Dashed |
| 0x02 | Dotted |

### 2.2.20 ShareableObject

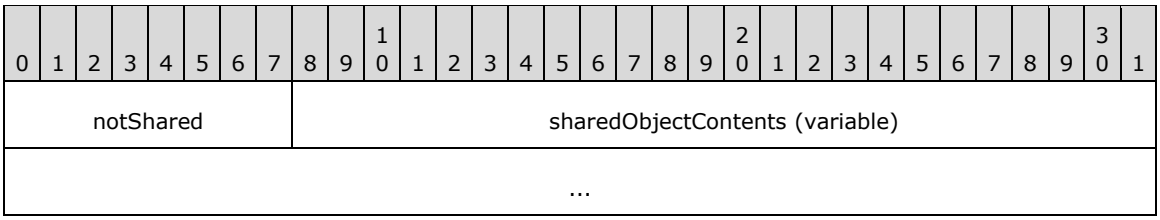
The **ShareableObject** structure specifies an object that can be passed as an argument to GDI+ functions. The shareable objects are the Font, Format, and Image objects. A **ShareableObject** structure can contain either the object, in which case the object cannot be shared, or a reference to a shared objects that was specified previously in the stream.

```
ShareableObject = NonSharedObject / UseSharedObject
```

### 2.2.21 NonSharedObject

The **NonSharedObject** structure specifies an object that can be passed as an argument to GDI+ functions. The objects are the Font, Format, and Image objects; these are specified in the **sharedObjectContents** field.

```
NonSharedObject = notShared SharedObjectContents
```




**notShared (1 byte):** A byte field that specifies that the type of the ShareableObject structure is a **NonSharedObject** structure. The value of this field MUST be "0x00".

**sharedObjectContents (variable):** A structure of type SharedObjectContents that specifies a **Font**, **Format**, or **Image** object.

### 2.2.22 SharedObjectContents

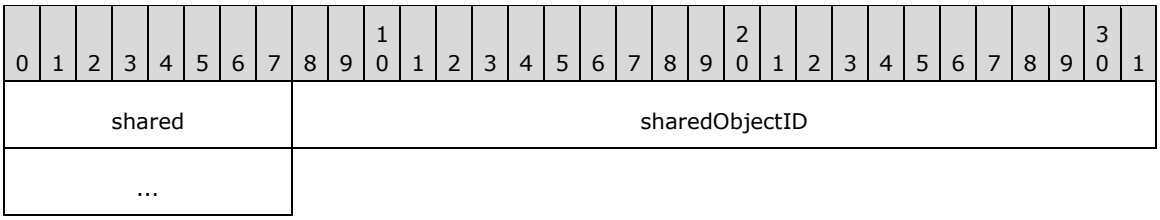
The **SharedObjectContents** structure specifies the content of a SharedObject, which is an object that can be shared by reference and passed as an argument to multiple GDI+ function calls. The **SharedObjectContents** structure also specifies the content of a NonSharedObject, which is an object that can be passed only once as an argument to a GDI+ function. The content of the **SharedObjectContents** structure is one of the Font, Format, or Image structures.

```
SharedObjectContents = Font / Format / Image
```

### 2.2.23 UseSharedObject

The **UseSharedObject** structure specifies an object that can be passed as an argument to GDI+ functions. The **UseSharedObject** structure contains a reference to a shared object that was specified previously in the stream. The shareable objects are the Font, Format, and Image objects.

```
UseSharedObject = shared SharedObjectID
```



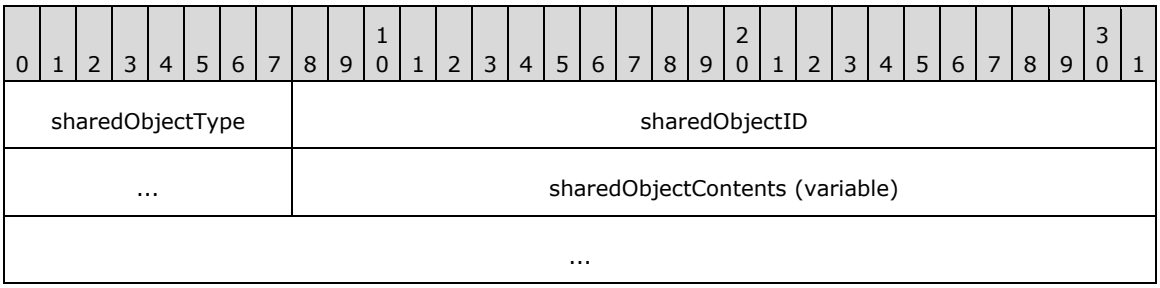

**shared (1 byte):** A byte field that specifies that the type of the ShareableObject structure is a **UseSharedObject** structure. The value of this field MUST be "0x01".

**sharedObjectID (4 bytes):** An **Int32** identifier that specifies the unique identifier of a shared object previously defined in the stream.

### 2.2.24 SharedObject

The **SharedObject** structure specifies the definition of an object that can be shared. This structure specifies the type of the object and associates an identifier with the object. The identifier MUST uniquely identify the object. The object MUST be one of the Font, Format, or Image structures.

```
SharedObject = sharedObjectType sharedObjectID SharedObjectContents
```




**sharedObjectType (1 byte):** A byte field that specifies the type of the shared object. The value of this field MUST be one of the values listed in the following table.

| Value | Object type |
|---|---|
| 0x00 | Font |
| 0x01 | Format |
| 0x02 | Image |

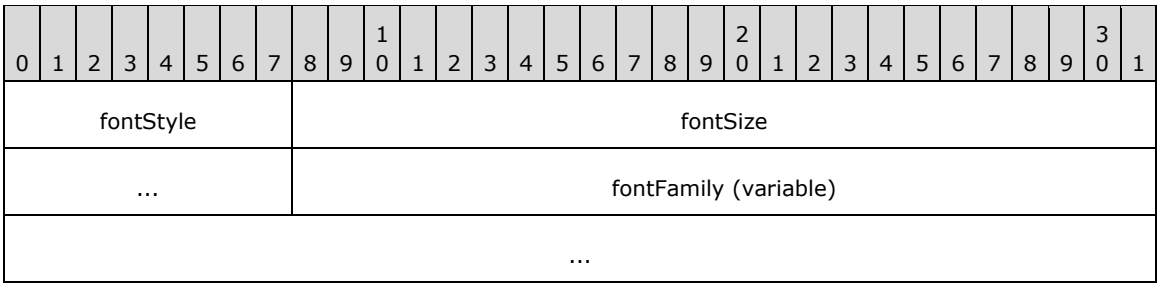
**sharedObjectID (4 bytes):** An **Int32** identifier that uniquely identifies the shared object that is being defined in the stream.

**sharedObjectContents (variable):** A structure of type SharedObjectContents that specifies the object being shared. The type of the structure contained herein MUST match the object type specified by the value of the **sharedObjectType** field.

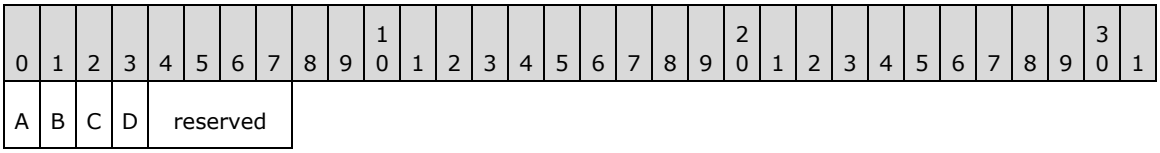
### 2.2.25 Font

The **Font** structure specifies a font that is used as an argument to a GDI+ function.

```
Font = fontStyle fontSize fontFamily
```




**fontStyle (1 byte):** A byte field that specifies the style of the font. The font style is a combination of the following basic font styles: **Strikeout**, **Underline**, **Bold**, and **Italic**. This field is a bitmask and is specified by the following bit fields.




**A – Italic (1 bit):** A bit that specifies that the style of the font is **Italic**.

**B – Bold (1 bit):** A bit that specifies that the thickness of the font is **Bold**.

**C – Underline (1 bit):** A bit that specifies that the text formatting of the font is **Underline**. **Strikeout** and **Underline** MUST NOT be specified together at the same time.<25>

**D – Strikeout (1 bit):** A bit that specifies that the text formatting of the font is **Strikeout**. **Strikeout** and **Underline** MUST NOT be specified together at the same time.<26>

**reserved (4 bits):** This value is ignored.

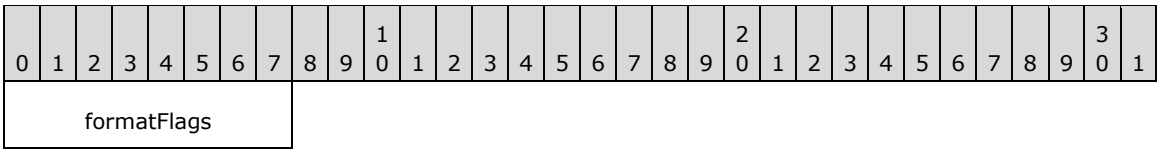
**fontSize (4 bytes):** A 4-byte **Float** value field that specifies the em size of the font, measured in points.

**fontFamily (variable):** A field of type **String** that specifies the name of the font family that this font belongs to.

### 2.2.26 Format

The **Format** structure specifies a format that is used as an argument to a GDI+ function.

```
Format = formatFlags
```




**formatFlags (1 byte)**: A byte field that specifies the formatting to be applied when drawing a string. The format flags are a combination of the following basic formatting properties: **Vertical Writing Mode**, **RTL Direction**, **Character Trim**, **Bottom Alignment**, **Top Alignment**, **Right Alignment**, and **Left Alignment**. This field is a bitmask and is specified by the following bit fields:

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 20 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 30 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | E | F | G | R | | | | | | | | | | | | | | | | | | | | | | | | |

**A – VerticalWritingMode (1 bit)**: A bit that specifies that the string is to be written vertically.

**B – DirectionRightToLeft (1 bit)**: A bit that specifies that the string is to be written in reverse order right-to-left.

**C – CharTrim (1 bit)**: A bit that specifies that the string is to be trimmed to the nearest character to fit where it is drawn.

**D – AlignBottom (1 bit)**: A bit that specifies that the string is to be aligned vertically at the bottom. **AlignTop** and **AlignBottom** MUST NOT be specified together at the same time.<27>

**E – AlignTop (1 bit)**: A bit that specifies that the string is to be aligned vertically at the top. **AlignTop** and **AlignBottom** MUST NOT be specified together at the same time.<28>

**F – AlignRight (1 bit)**: A bit that specifies that the string is to be aligned horizontally at the right. **AlignLeft** and **AlignRight** MUST NOT be specified together at the same time.<29>

**G – AlignLeft (1 bit)**: A bit that specifies that the string is to be aligned horizontally at the left. **AlignLeft** and **AlignRight** MUST NOT be specified together at the same time.<30>

**R (1 bit)**: **Reserved** - The value is ignored.

### 2.2.27 Image

The **Image** structure specifies an image that is used as an argument to a GDI+ function.

```
Image = imageFlags length imageContents
```

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 20 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 30 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| imageFlags | | | | | | | | length | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | imageContents (variable) | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**imageFlags (1 byte)**: A byte field that specifies the image properties used to draw the image. This field is a bitmask and is specified by the following bit fields.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 20 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 30 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**A – Smoothing (1 bit)**: A bit that specifies that the image is smoothed when scaling.

**reserved (7 bits)**: The value MUST be zero.

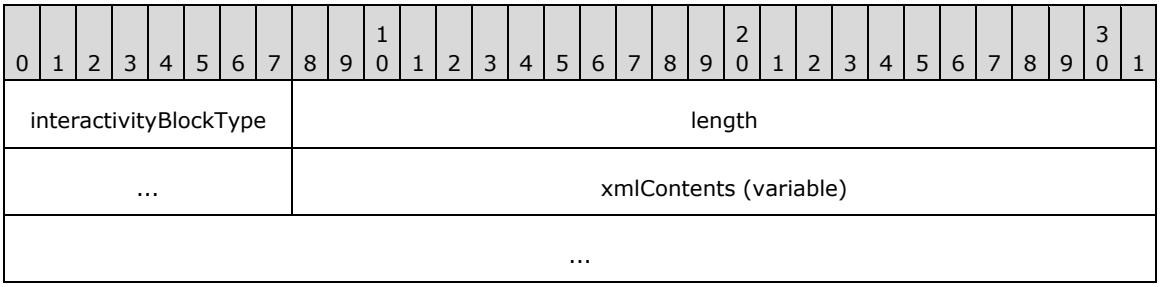
**length (4 bytes)**: An **Int32** value that specifies the length, in bytes, of the image (specified by the **imageContents** field) in this structure.

**imageContents (variable)**: A byte array that specifies the content of the image. The number of bytes MUST match the value of the **length** field.

### 2.2.28 InteractivityBlock

The **InteractivityBlock** structure specifies an XML document that describes properties of the bookmarks, labels, actions, and fixed headers of tables and matrices. There are four types of **InteractivityBlock** structures, each with its own XML document type. Each type of **InteractivityBlock** can occur no more than once in the stream.

```
InteractivityBlock  = interactivityBlockType length xmlContents
```




**interactivityBlockType (1 byte):** A byte field that specifies the type of the structure contents. The type is one of **Bookmarks**, **Labels**, **Actions**, or **FixedHeaders**. The value of this field MUST be one of the values associated with an interactivity block type listed in the following table.

| Value | Interactivity block type |
|---|---|
| 0x00 | Bookmarks |
| 0x01 | Labels |
| 0x02 | Actions |
| 0x04 | FixedHeaders |

**length (4 bytes):** An **Int32** that specifies the length in bytes of the XML document contained in the **xmlContents** field.

**xmlContents (variable):** A byte array that specifies the properties and property values of the interactivity block as an XML document. For each of the four types of interactivity blocks, the root XML element is specified in the following table. The entire XML specification of the interactivity blocks is in section 2.3.

| Interactivity block type | XML root element |
|---|---|
| Bookmarks | BOOKMARKS |
| Labels | LABELS |
| Actions | INTERACTION |
| FixedHeaders | FIXEDHEADERS |

## 2.3 RGDI XML Elements

This section specifies the definitions and the valid values of XML elements that are used in the RGDI stream to represent the content of interactivity blocks. The types of interactivity blocks are as follows:

- Interaction
- Bookmarks
- Labels
- FixedHeader

Each interactivity block type contains a single XML document in a byte array field, which is preceded by the type of the block and the number of bytes in the byte array. (For more information, see the InteractivityBlock structure.) The root XML elements for these four XML documents are the **INTERACTION**, **BOOKMARKS**, **LABELS**, and **FIXEDHEADERS** elements.

These interactivity blocks are all optional. Each block can be specified no more than once in an RGDI stream. These blocks can appear in any order, as in the following examples.

**Example 1**

```
<BOOKMARKS>
  <Item Left="3.175" Top="6.35">BID42</Item>
</BOOKMARKS>
<INTERACTION>
  <Item
    Id="47"
    Label="Bookmark"
    Type="BookmarkLink"
    Left="25.40" Top="38.1" Width="76.2" Height="50.8"
    Shape="R"
    >
    <Action Page="39">BID42</Action>
  </Item>
</INTERACTION>
<LABELS>
  <Item Left="3.175" Top="6.35">LID76</Item>
</LABELS>
<FIXEDHEADERS>
  <FH
    ID="21"
    HHB="19.05"
    >
  </FH>
  <FH
    ID="36"
    VHL="38.1" VHR="76.2"
    >
  </FH>
  <FH
    ID="54"
    HHB="19.05"
    VHL="38.1" VHR="76.2"
    >
  </FH>
</FIXEDHEADERS>
```

**Example 2**

```
<FIXEDHEADERS>
  <FH
    ID="21"
    HHB="19.05"
    >
  </FH>
  <FH
    ID="36"
    VHL="38.1" VHR="76.2"
    >
  </FH>
  <FH
    ID="54"
    HHB="19.05"
    VHL="38.1" VHR="76.2"
    >
  </FH>
</FIXEDHEADERS>
<INTERACTION>
  <Item
    Id="47"
    Label="Bookmark"
    Type="BookmarkLink"
    Left="25.40" Top="38.1" Width="76.2" Height="50.8"
    Shape="R"
    >
    <Action Page="39">BID42</Action>
  </Item>
</INTERACTION>
```

**Example 3**

```
<LABELS>
  <Item Left="3.175" Top="6.35">LID76</Item>
</LABELS>
<BOOKMARKS>
  <Item Left="3.175" Top="6.35">BID42</Item>
</BOOKMARKS>
```

### 2.3.1 INTERACTION

The **INTERACTION** element is the root element of the **Actions** block and specifies a collection of INTERACTION.Item elements. The child elements of the **INTERACTION** element specify which actions (links, toggle actions, and sort actions) are present.

If the **Actions** block is present in the RGDI stream, the **INTERACTION** element MUST be specified. If the **INTERACTION** element is specified, there MUST be at least one and there MAY<31> be more than one **INTERACTION.Item** element in the **INTERACTION** collection.

Following is the child element of the **INTERACTION** element.

| Child elements |
| --- |
| INTERACTION.Item |

#### 2.3.1.1 INTERACTION.Item

The **INTERACTION.Item** element specifies the actions (links, toggle actions, and sort actions) that are present. This element MUST be specified. There MUST be at least one and there MAY<32> be more than one **INTERACTION.Item** element in the INTERACTION collection.

The **INTERACTION.Item** element is of type Item.

Following is the parent element of the **INTERACTION.Item** element.

| Parent elements |
| --- |
| INTERACTION |

### 2.3.2 Item (INTERACTION)

***Applies to the INTERACTION block only***

The **Item** element specifies the actions (links, toggle actions, and sort actions) that are present in a report. The location, size, and shape of an area on the page that can be clicked to activate the action are specified along with an identifier and an optional label that provides a user-friendly label for the action. The type of the action is also specified. The action type is one of the following:

- HyperLink
- DrillThrough
- BookmarkLink
- Toggle
- Sort

The following is the parent element of the **INTERACTION.Item** element.

| Parent elements |
| --- |
| INTERACTION |

The following are the attributes of the **INTERACTION.Item** element.

| Attributes |
| --- |
| Item.Height |
| Item.Id |
| Item.Label |
| Item.Left |
| Item.Shape |
| Item.Top |
| Item.Type |
| Item.Width |

The following are the child elements of the **INTERACTION.Item** element.

| Child elements |
| --- |
| Item.Action |
| Item.Vertices |

### 2.3.2.1 Item.Height

***Applies to the INTERACTION block only***

The **Item.Height** attribute specifies the height of an area on a page that can be clicked to activate the action that is associated with an Item. This height is measured in millimeters. The **Item.Height** attribute MUST be specified. The value of this attribute MUST be a non-negative **Float**.<33>

If the shape of the area that is specified by the Item.Shape element is a rectangle or circle, the **Item.Height** attribute specifies the height of the rectangle or circle. If the shape of the area that is specified by the **Item.Shape** element is a polygon, the **Item.Height** attribute specifies the height of the image that contains the polygon.

Following is the parent element of the **Item.Height** attribute.

| Parent elements |
| --- |
| Item |

### 2.3.2.2 Item.Id

***Applies to the INTERACTION block only***

The **Item.Id** attribute specifies a unique name for an action. This attribute MUST be specified and is of type **String**.

Following is the parent element of the **Item.Id** attribute.

| Parent elements |
| --- |
| Item |

### 2.3.2.3 Item.Label

***Applies to the INTERACTION block only***

The **Item.Label** attribute specifies a user-friendly label for the action that is associated with an Item. This attribute is optional and is of type **String**.

Following is the parent element of the **Item.Label** attribute.

| Parent elements |
| --- |
| Item |

### 2.3.2.4 Item.Left

***Applies to the INTERACTION block only***

The **Item.Left** attribute specifies the distance from the left edge of the page of an area on the page that can be clicked to activate the action that is associated with an Item. This distance is measured in millimeters. The **Item.Left** attribute MUST be specified. The value of this attribute MUST be a non-negative **Float**.<34>

If the shape of the area that is specified by the Item.Shape element is a rectangle or circle, the **Item.Left** attribute specifies the distance of the left edge of the rectangle or circle from the left edge of the page. If the shape of the area that is specified by the **Item.Shape** element is a polygon, the **Item.Left** attribute specifies the distance of the left edge of the image that contains the polygon from the left edge of the page.

Following is the parent element of the **Item.Left** attribute.

| Parent elements |
| --- |
| Item |

### 2.3.2.5 Item.Shape

***Applies to the INTERACTION block only***

The **Item.Shape** attribute specifies the shape of an area on a page that can be clicked to activate the action that is associated with an Item. This shape can be a rectangle, a polygon, or a circle. The **Item.Shape** attribute MUST be specified, and its value MUST be a **String**. The value of the **Item.Shape** attribute MUST be one of the following:<35>

**R**: Specifies that the shape of the area is a rectangle.

**P**: Specifies that the shape of the area is a polygon.

**C**: Specifies that the shape of the area is a circle.

Following is the parent element of the **Item.Shape** attribute.

| Parent elements |
| --- |
| Item |

### 2.3.2.6 Item.Top

***Applies to the INTERACTION block only***

The **Item.Top** attribute specifies the distance from the top edge of the page of an area on the page that can be clicked to activate the action that is associated with an Item. This distance is measured in millimeters. The **Item.Top** attribute MUST be specified. The value of this attribute MUST be a non-negative **Float**.<36>

If the shape of the area that is specified by the Item.Shape element is a rectangle or a circle, the **Item.Top** attribute specifies the distance of the top edge of the rectangle or circle from the top edge of the page. If the shape of the area that is specified by the **Item.Shape** element is a polygon, the **Item.Top** attribute specifies the distance of the top edge of the image that contains the polygon from the top edge of the page.

Following is the parent element of the **Item.Top** attribute.

| Parent elements |
| --- |
| Item |

### 2.3.2.7 Item.Type

***Applies to the INTERACTION block only***

The **Item.Type** attribute specifies the action type of the action that is associated with an Item. This attribute MUST be specified, and its value MUST be a **String**. The value of the **Item.Type** attribute MUST be one of the following:

**HyperLink**: Specifies that the action is to follow a hyperlink, and then display the target of the hyperlink.

**DrillThrough**: Specifies that the action is to display a drillthrough report.

**BookmarkLink**: Specifies that the action is to follow a hyperlink, and then display the target of the hyperlink.

**Toggle**: Specifies that the action is to toggle an item.

**Sort**: Specifies that the action is to sort an item.

Following is the parent element of the **Item.Type** attribute.

| Parent elements |
| --- |
| Item |

### 2.3.2.8 Item.Width

***Applies to the INTERACTION block only***

The **Item.Width** attribute specifies the width of an area on a page that can be clicked to activate the action that is associated with an Item. This distance is measured in millimeters. The **Item.Width** attribute MUST be specified. The value of this attribute MUST be a non-negative **Float**.<37>

If the shape of the area that is specified by Item.Shape is a rectangle or circle, the **Item.Width** attribute specifies the width of the rectangle or circle. If the shape of the area that is specified by **Item.Shape** is a polygon, the **Item.Width** attribute specifies the width of the image that contains the polygon.

Following is the parent element of the **Item.Width** attribute.

| Parent elements |
| --- |
| Item |

### 2.3.2.9 Item.Action

***Applies to the INTERACTION block only***

The **Item.Action** element specifies additional and specific information for the particular action type that is specified by the sibling Item.Type attribute. The **Item.Action** element MUST be specified and is of type Action.<38>

Following is the parent element of the **Item.Action** element.

| Parent elements |
| --- |
| Item |

### 2.3.2.10 Item.Vertices

***Applies to the INTERACTION block only***

The **Item.Vertices** element specifies the collection of Point elements that define the vertices of a polygon that is the boundary of the area on a page that can be clicked to activate the associated action. This element is optional and is of type Vertices. However, if the value of the sibling Item.Type attribute is "P", which means that the shape is a polygon, the **Item.Vertices** element MUST be specified. Otherwise, this element is ignored.

Following is the parent element of the **Item.Vertices** element.

| Parent elements |
| --- |
| Item |

## 2.3.3 Action

The **Action** element specifies additional and specific information for the particular action type that is specified by the sibling Item.Type attribute. The value of the **Action** element is a **String**. This value depends on the value of the sibling **Item.Type** attribute. The value of the **Action** element MUST be one of the values in the following table.

| Item.Type | Value of Item.Action string |
| --- | --- |
| HyperLink | The URL of the hyperlink. |
| DrillThrough | The unique identifier of the drillthrough report to be displayed. |
| BookmarkLink | The unique identifier of the bookmark link. |

| Item.Type | Value of Item.Action string |
|---|---|
| Toggle | True or false depending on whether the item is toggled. |
| Sort | "Ascending" or "Descending" if the current sort state is descending or ascending. |

Following is the parent element of the **Action** element.

| Parent elements |
|---|
| Item |

Following is the attribute of the **Action** element.

| Attributes |
|---|
| Action.Page |

### 2.3.3.1 Action.Page

The **Action.Page** attribute specifies the number of the current physical page. This attribute is optional and is of type **Int32**. The value of **Action.Page** MUST be a non-zero positive **Int32**.<39>

If the value of the parent sibling Item.Type attribute is "BookmarkLink", the **Action.Page** attribute MUST be specified, and it represents the page that contains the target **Bookmark** of the **BookmarkLink** action. Otherwise, the **Action.Page** attribute is ignored.

Following is the parent element of the **Action.Page** attribute.

| Parent elements |
|---|
| Action |

## 2.3.4 Vertices

The **Vertices** element specifies a collection of Point elements that define the vertices of a polygon.

Following is the parent element of the **Vertices** element.

| Parent elements |
|---|
| Action |

Following is the child element of the **Vertices** element.

| Child elements |
|---|
| Vertices.Point |

### 2.3.4.1 Vertices.Point

The **Vertices.Point** attribute specifies the location of a vertex of a polygon. This element MUST be specified. There MUST be at least one and there MAY<40> be more than one **Vertices.Point** element in the Vertices collection. This element is of type Point.

Following is the parent element of the **Vertices.Point** attribute.

| Parent elements |
| --- |
| Vertices |

## 2.3.5 Point

The **Point** element specifies the position of a vertex of a polygon. This element has two attributes that specify the x-coordinate and y-coordinate of the point relative to the top and left edge of the page.

Following is the parent element of the **Point** element.

| Parent elements |
| --- |
| Vertices |

The following are the attributes of the **Point** element.

| Attributes |
| --- |
| Point.X |
| Point.Y |

### 2.3.5.1 Point.X

The **Point.X** attribute specifies the horizontal x-coordinate location of a Point relative to the left edge of the page. This distance is measured in millimeters. The **Point.X** attribute MUST be specified. The value of this attribute MUST be a non-negative **Float**.<41>

Following is the parent element of the **Point.X** attribute.

| Parent elements |
| --- |
| Point |

### 2.3.5.2 Point.Y

The **Point.Y** attribute specifies the vertical y-coordinate location of a Point relative to the top edge of the page. This distance is measured in millimeters. The **Point.Y** attribute MUST be specified. The value of this attribute MUST be a non-negative **Float**.<42>

Following is the parent element of the **Point.Y** attribute.

| Parent elements |
| --- |
| Point |

### 2.3.6 BOOKMARKS

The **BOOKMARKS** element is the root element of the **Bookmarks** block and specifies a collection of BOOKMARKS.Item elements. Each **BOOKMARKS.Item** child element specifies the location on the page where a bookmark resides by mapping the unique name of the bookmarked item to a particular point.

If the **Bookmarks** block is present in the RGDI stream, the **BOOKMARKS** element MUST be specified. If the **BOOKMARKS** element is specified, there MUST be at least one and there can be more than one **BOOKMARKS.Item** element in the **BOOKMARKS** collection.<43>

Following is the child element of the **BOOKMARKS** element.

| Child elements |
| --- |
| BOOKMARKS.Item |

#### 2.3.6.1 BOOKMARKS.Item

The **BOOKMARKS.Item** element specifies the unique name of a bookmarked item and associates the item with a particular point on a page. This element MUST be specified and is of type Item. There MUST be at least one and there can be more than one **BOOKMARKS.Item** element in the BOOKMARKS collection.<44>

Following is the parent element of the **BOOKMARKS.Item** attribute.

| Parent elements |
| --- |
| BOOKMARKS |

### 2.3.7 LABELS

The **LABELS** element is the root element of the **Labels** block and specifies a collection of LABELS.Item elements. Each **LABELS.Item** child element specifies the location on a page where a label resides by mapping the unique name of the labeled item with a particular point. This point is specified in the **LABELS.Item** element.

If the **Labels** block is present in the RGDI stream, the **LABELS** element MUST be specified. If the **LABELS** element is specified, there MUST be at least one and there can be more than one **LABELS.Item** element in the **LABELS** collection.<45>

Following is the child element of the **LABELS** element.

| Child elements |
| --- |
| LABELS.Item |

### 2.3.7.1 LABELS.Item

The **LABELS.Item** element specifies the unique name of a labeled item and associates it with a particular point on a page. This element MUST be specified and is of type Item. There MUST be at least one and there can be more than one **LABELS.Item** element in the LABELS collection.<46>

Following is the parent element of the **LABELS.Item** attribute.

| Parent elements |
| --- |
| LABELS |

## 2.3.8 Item (BOOKMARKS and LABELS)

***Applies to the BOOKMARKS and LABELS blocks only***

The **Item** element specifies the unique name of a bookmarked or labeled item and associates the item with a particular point on a page. This element MUST be specified and its value MUST be a **String**. The location of the point on the page is specified as the values of the **Item.Left** and **Item.Top** attributes.

The following are the parent elements of the **Item** element.

| Parent elements |
| --- |
| BOOKMARKS |
| LABELS |

The following are the attributes of the **Item** element.

| Attributes |
| --- |
| Item.Left |
| Item.Top |

### 2.3.8.1 Item.Left

***Applies to the BOOKMARKS and LABELS blocks only***

The **Item.Left** attribute specifies the horizontal position of a point that is associated with a bookmark or label relative to the left edge of the page. The distance is measured in millimeters. The **Item.Left** attribute MUST be specified. The value of this attribute MUST be a non-negative **Float**.<47>

Following is the parent element of the **Item.Left** attribute.

| Parent elements |
| --- |
| Item |

### 2.3.8.2 Item.Top

***Applies to the BOOKMARKS and LABELS blocks only***

The **Item.Top** attribute specifies the vertical position of a point that is associated with a bookmark or label relative to the top edge of the page. This distance is measured in millimeters. The **Item.Top** attribute MUST be specified. The value of this attribute MUST be a non-negative **Float**.<48>

Following is the parent element of the **Item.Top** attribute.

| Parent elements |
| --- |
| Item |

### 2.3.9 FIXEDHEADERS

The **FIXEDHEADERS** element is the root element of the **FixedHeaders** block and specifies a collection of FIXEDHEADERS.FH elements. The child elements of the **FIXEDHEADERS** element specify the information that is required to render floating headers.

If the **FixedHeaders** block is present in the RGDI stream, the **FIXEDHEADERS** element MUST be specified. If this element is specified, there MUST be at least one and there can be more than one **FIXEDHEADERS.FH** element in the **FIXEDHEADERS** collection.<49>

Following is the child element of the **FIXEDHEADERS** element.

| Child elements |
| --- |
| FIXEDHEADERS.FH |

#### 2.3.9.1 FIXEDHEADERS.FH

The **FIXEDHEADERS.FH** element specifies the information that is required to render a floating header. This element MUST be specified and is of type FH. There MUST be at least one and there MAY<50> be more than one **FIXEDHEADERS.FH** element in the FIXEDHEADERS collection.

Following is the parent element of the **FIXEDHEADERS.FH** element.

| Parent elements |
| --- |
| FIXEDHEADERS |

### 2.3.10 FH

The **FH** element specifies an individual fixed header region for a table or matrix. The **ID** attribute refers to the unique name of the table or matrix for which the header region is specified. There can be multiple **FH** elements per table or matrix. An element can contain an **HHB** attribute, or the **VHL** and **VHR** attributes, or all three.<51>

The following is the parent element of the **FH** element.

| Parent element |
| --- |
| FIXEDHEADERS |

The following are the attributes of the **FH** element.

| Attributes |
| --- |
| FH.HHB |
| FH.ID |
| FH.VHL |
| FH.VHR |

### 2.3.10.1 FH.HHB

The **FH.HHB** attribute specifies the bottom-most position, in millimeters, of a horizontal fixed header. This attribute specifies the distance from the top of the page to the bottom position of the bottom-most horizontal header. This distance is measured in millimeters. This attribute is optional. If this attribute is present, its value MUST be a non-negative **Float**.

If the **FH.HHB** attribute is not specified, the FH.VHL and FH.VHR attributes MUST be specified.<52>

Following is the parent element of the **FH.HHB** attribute.

| Parent elements |
| --- |
| FH |

### 2.3.10.2 FH.ID

The **FH.ID** attribute specifies the unique name of the table or matrix to which the fixed header refers. This attribute MUST be specified, and its value MUST be a **String**.

Following is the parent element of the **FH.ID** attribute.

| Parent elements |
| --- |
| FH |

### 2.3.10.3 FH.VHL

The **FH.VHL** attribute specifies the left-most position, in millimeters, of a vertical fixed header. The **FH.VHL** attribute specifies the distance from the left edge of the page to the left edge of the vertical header in millimeters.

This attribute is optional. If it is present, its value MUST be a non-negative **Float**. If the FH.HHB attribute is not specified or if the FH.VHR attribute is specified, the **FH.VHL** attribute MUST be specified.<53>

Following is the parent element of the **FH.VHL** attribute.

| Parent elements |
| --- |
| FH |

### 2.3.10.4 FH.VHR

The **FH.VHR** attribute specifies the right-most position, in millimeters, of a vertical fixed header. The **FH.VHR** attribute specifies the distance from the right edge of the page to the right edge of the vertical header in millimeters.

This attribute is optional. If it is present, its value MUST be a non-negative **Float**. If FH.HHB is not specified or if FH.VHL is specified, the **FH.VHR** attribute MUST be specified.<54>

Following is the parent element of the **FH.VHR** attribute.

| Parent elements |
| --- |
| FH |

# 3 Structure Examples

This section contains examples of some of the more commonly used data structures in an RGDI stream. These examples are meant to be a starting point for an implementer who is learning the stream format. They do not cover all structures in the stream format.

The following conventions are followed for examples, unless noted otherwise:

- The order of the structures and fields within the example match their corresponding order in the stream format.
- The examples begin with the first structure that is relevant to the example and end with the last structure that is relevant to the example.
- The examples are self-contained and contiguous; no required structures are omitted within an example.
- Undefined and ignored fields are not included in the field explanations.

## 3.1 Record

The following example illustrates a Record structure with a DrawRectangle function structure in an RGDI stream. The **DrawRectangle** function structure specifies a rectangle that has a solid-line SlateBlue-colored border that is one pixel thick. The size of the rectangle is 2" x 3" (height x width). Its upper-left corner is located 1.5 inches below the top of the page and 1 inch to the right of the left edge of the page.

| Offset | Length | Field | Value |
|---|---|---|---|
| 0000 | 001A | **Record** | |
| 0000 | 0001 | BYTE - **recordType** | 0x01 |
| 0001 | 0019 | **RecordContents** = **Function** = **DrawRectangle** | |
| 0001 | 0001 | BYTE - **functionID** | 0x01 |
| 0002 | 0008 | **Pen** | |
| 0002 | 0003 | **Brush** | |
| 0002 | 0001 | BYTE – **red** | 0x6A |
| 0003 | 0001 | BYTE – **green** | 0x5A |
| 0004 | 0001 | BYTE – **blue** | 0xCD |

| Offset | Length | Field | Value |
|---|---|---|---|
| 0005 | 0004 | FLOAT – **width** | 0.2645838 |
| 0009 | 0001 | BYTE – **penStyle** | 0x00 |
| 000A | 0010 | **Rectangle** | |
| 000A | 0004 | FLOAT – **x** | 25.4 |
| 000E | 0004 | FLOAT - **y** | 38.1 |
| 0012 | 0004 | FLOAT – **width** | 76.2 |
| 0016 | 0004 | FLOAT – **height** | 50.8 |

**Record**: A **Record** structure that specifies nested structures, GDI+ drawing functions, and shared objects to render a report item.

**recordType**: "0x01" specifies that the type of the **Record** structure is a Function and that the contents of the **RecordContents** field will be a **Function**.

**RecordContents**: A structure that specifies the contents of the **Record** structure.

**Function**: A structure that specifies a GDI+ drawing function and its arguments.

**DrawRectangle**: A structure that specifies the GDI+ **DrawRectangle** drawing function and its arguments.

**functionID**: "0x01" specifies that the GDI+ drawing function is **DrawRectangle** and that the content of the RecordContents structure is the **DrawRectangle** structure.

**Pen**: A structure that specifies the contents of the Pen structure.

**Brush**: A structure that specifies the contents of the Brush structure, which specifies the red, green, and blue components of the color that is used to draw the rectangle.

**red**: "0x6A" specifies the value of the red component of the System.Drawing.Color.SlateBlue color.

**green**: "0x5A" specifies the value of the green component of the System.Drawing.Color.SlateBlue color.

**blue**: "0xCD" specifies the value of the blue component of the System.Drawing.Color.SlateBlue color.

**width**: "0.2645838" specifies the width of the pen in millimeters. This is the width of a line that is one pixel thick at 96 dpi.

**penStyle**: "0x00" specifies that a solid line is to be drawn for the boundaries of the rectangle.

**Rectangle**: A structure that specifies the position and size of the rectangle to be drawn.

**x**: "25.4" specifies the distance of the upper-left corner of the rectangle from the left edge of the page in millimeters. This is equivalent to 1 inch.

**y**: "38.1" specifies the distance of the upper-left corner of the rectangle from the top edge of the page in millimeters. This is equivalent to 1.5 inches.

**width**: "76.2" specifies the width of the rectangle in millimeters. This is equivalent to 3 inches.

**height**: "50.8" specifies the height of the rectangle in millimeters. This is equivalent to 2 inches.

## 3.2 Actions Block

The following example illustrates the XML content of the **Actions** block. This example describes a bookmark action. This action specifies an active area in the document that, when clicked, advances the renderer to display the page of the report that contains a specified bookmark.

The root element of the **Actions** block is the INTERACTION element. The **INTERACTION** collection contains only one child Item element, which specifies the clickable area and the action to take when a user clicks in this region of the rendered report page. The unique identifier of the area is the string "47", specified by the Item.Id attribute. The Item.Label attribute specifies the string "Bookmark". Item.Type specifies that the type of this action is a **BookmarkLink**.

The Item.Shape attribute that has a value of "R" specifies that the area has a rectangular shape and that the position and size of this rectangle are specified by the **Left**, **Top**, **Width**, and **Height** attributes of the **Item** element and are measured in millimeters.

The Action subelement of the **Item** element specifies that the bookmark to jump to is on page 39 and that this is the bookmark whose unique identifier is "BID42".

```
<INTERACTION>
  <Item
    Id="47"
    Label="Bookmark"
    Type="BookmarkLink"
    Left="25.40" Top="38.1" Width="76.2" Height="50.8"
    Shape="R"
    >
    <Action Page="39">BID42</Action>
  </Item>
</INTERACTION>
```

## 3.3 Bookmarks Block

The following example illustrates the use of the XML content of the **Bookmarks** block. This example describes a bookmark that specifies a point in the report to which an action can jump.

The root element of the **Bookmarks** block is the BOOKMARKS element. The **BOOKMARKS** collection contains a single Item element. This element specifies, in millimeters, the x-coordinate and y-coordinate of a point on the page as the values of the **Left** and **Top** attributes. The unique identifier of this bookmark is specified by the value of the **Item** element, which is "BID42" in this example.

```
<BOOKMARKS>
  <Item Left="3.175" Top="6.35">BID42</Item>
</BOOKMARKS>
```

## 3.4 Labels Block

The following example illustrates the use of the XML content of the **Labels** block. This example describes a label that specifies a point in a report to which an action can jump.

The root element of the **Labels** block is the LABELS element. The **LABELS** collection contains a single Item element. This element specifies, in millimeters, the x-coordinate and y-coordinate of a point on the page as the values of the **Left** and **Top** attributes. The unique identifier of this bookmark is specified by the value of the **Item** element, which is "LID76" in this example.

```
<LABELS>
  <Item Left="3.175" Top="6.35">LID76</Item>
</LABELS>
```

## 3.5 FixedHeaders Block

The following example illustrates the use of the XML content of the **FixedHeaders** block. This example describes three different fixed headers: a horizontal column, a vertical column, and a corner region of a matrix. The root element of the **FixedHeaders** block is the FIXEDHEADERS element. The **FIXEDHEADERS** collection contains three FH elements. Each of these elements has an **ID** attribute that specifies a unique identifier that is associated with the header.

The first **FH** element has only an FH.HHB attribute, which specifies that this fixed header is for a fixed row header and that this header extends to 19.05 millimeters from the top of the page.

The second **FH** element has only a FH.VHL and a FH.VHR attribute, which specify that this fixed header is for a fixed column row header that begins 38.1 millimeters from the left edge of the page and ends at 76.3 millimeters from the left edge of the page.

The third **FH** element has an **FH.HHB** attribute and both **FH.VHL** and **FH.VHR** attributes, which specify that this fixed header is for a fixed corner header that extends to 19.05 millimeters from the top of the page, begins 38.1 millimeters from the left edge of the page, and ends at 76.3 millimeters from the left edge of the page.

```
<FIXEDHEADERS>
  <FH
    ID="21"
    HHB="19.05"
    >
  </FH>
  <FH
    ID="36"
    VHL="38.1" VHR="76.2"
    >
  </FH>
  <FH
    ID="54"
    HHB="19.05"
    VHL="38.1" VHR="76.2"
    >
  </FH>
</FIXEDHEADERS>
```

# 4 Security Considerations

None.

# 5 (Updated Section) Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include ~~released service packs~~updates to those products.

- Microsoft SQL Server 2005
- Microsoft SQL Server 2008
- Microsoft SQL Server 2008 R2
- Microsoft SQL Server 2012
- Microsoft SQL Server 2014
- Microsoft SQL Server 2016
- Microsoft SQL Server 2017
- Microsoft SQL Server 2019
- Microsoft Visual Studio 2005
- Microsoft Visual Studio 2008
- Microsoft Visual Studio 2010
- Microsoft Visual Studio 2012
- Microsoft Visual Studio 2013
- Microsoft Visual Studio 2015
- Microsoft Visual Studio 2017
- Microsoft Visual Studio 2019

Exceptions, if any, are noted ~~below.~~in this section. If ~~a~~ an update version, service pack or ~~Quick Fix Engineering (QFE~~Knowledge Base (KB) number appears with ~~the~~a product ~~version,~~name, the behavior changed in that ~~service pack or QFE.~~update. The new behavior also applies to subsequent ~~service packs of the product~~updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms "SHOULD" or "SHOULD NOT" implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term "MAY" implies that the product does not follow the prescription.

<1> Section 2.2.1: The number of **InteractivityBlock** structures is not validated by the ReportViewer Windows Forms control that ships with Microsoft Visual Studio.

<2> Section 2.2.3: The **width** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<3> Section 2.2.3: The **height** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<4> Section 2.2.5: Report items **GaugePanel** and **Map** are not supported by Microsoft SQL Server 2005 Reporting Services.

<5> Section 2.2.5: Report item **Tablix** is not supported by SQL Server 2005 Reporting Services.

<6> Section 2.2.5: The position and size of the rectangle is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<7> Section 2.2.8: The **functionID** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<8> Section 2.2.9: The position and size of the rectangle is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<9> Section 2.2.10: The position and size of the rectangle is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<10> Section 2.2.11: The position and size of the rectangle is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<11> Section 2.2.12: The **x1** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<12> Section 2.2.12: The **y1** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<13> Section 2.2.12: The **x2** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<14> Section 2.2.12: The **y2** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<15> Section 2.2.14: The position and size of the destination rectangle is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<16> Section 2.2.14: The position and size of the image rectangle is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<17> Section 2.2.15: The **x** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<18> Section 2.2.15: The **y** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<19> Section 2.2.17: The **x** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<20> Section 2.2.17: The **y** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<21> Section 2.2.17: The **width** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<22> Section 2.2.17: The **height** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<23> Section 2.2.19: The **width** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<24> Section 2.2.19: The **penStyle** value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio. If it is not one of the values in the table, it will be treated as if it were pen style **Dotted**.

<25> Section 2.2.25: **Strikeout** and **Underline** being set at the same time is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<26> Section 2.2.25: **Strikeout** and **Underline** being set at the same time is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<27> Section 2.2.26: **AlignTop** and **AlignBottom** being set at the same time is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<28> Section 2.2.26: **AlignTop** and **AlignBottom** being set at the same time is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<29> Section 2.2.26: **AlignLeft** and **AlignRight** being set at the same time is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<30> Section 2.2.26: **AlignLeft** and **AlignRight** being set at the same time is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<31> Section 2.3.1: The number of **INTERACTION.Item** elements in the **INTERACTION** collection is not validated by the ReportViewer Windows Forms server control that ships with Visual Studio.

<32> Section 2.3.1.1: The number of **INTERACTION.Item** elements in the **INTERACTION** collection is not validated by ReportViewer Windows Forms server control that ships with Visual Studio.

<33> Section 2.3.2.1: The **Item.Height** attribute value is not currently validated by the ReportViewer Windows Forms control that ships with Visual Studio. This value can be written to the RGDI stream with double precision by the ReportViewer control.

<34> Section 2.3.2.4: The **Item.Left** attribute value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio. This value can be written to the RGDI stream with double precision by the ReportViewer control.

<35> Section 2.3.2.5: The **Item.Shape** attribute value is not currently validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<36> Section 2.3.2.6: The **Item.Top** attribute value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio. This value can be written to the RGDI stream with double precision by the ReportViewer control.

<37> Section 2.3.2.8: The **Item.Width** attribute value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio. This value can be written to the RGDI stream with double precision by the ReportViewer control.

<38> Section 2.3.2.9: The presence of the **Item.Action** element is not currently validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<39> Section 2.3.3.1: The **Action.Page** attribute value is not validated by the ReportViewer Windows Forms server control that ships with Visual Studio.

<40> Section 2.3.4.1: The existence of at least one **Vertices.Point** element in the Vertices collection is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<41> Section 2.3.5.1: The **Point.X** attribute value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio. This value can be written to the RGDI stream with double precision by the ReportViewer control.

<42> Section 2.3.5.2: The **Point.Y** attribute value is not validated by the ReportViewer Windows Forms control that ships with Visual Studio. This value can be written to the RGDI stream with double precision by the ReportViewer control.

<43> Section 2.3.6: The existence of at least one **BOOKMARKS.Item** element in the **BOOKMARKS** collection is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<44> Section 2.3.6.1: The existence of at least one **BOOKMARKS.Item** element in the **BOOKMARKS** collection is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<45> Section 2.3.7: The existence of at least one **LABELS.Item** element in the **LABELS** collection is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<46> Section 2.3.7.1: The existence of at least one **LABELS.Item** element in the **LABELS** collection is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<47> Section 2.3.8.1: The **Item.Left** attribute value is not currently validated to be non-negative by the ReportViewer Windows Forms control that ships with Visual Studio. This value can be written to the RGDI stream with double precision by the ReportViewer control.

<48> Section 2.3.8.2: The **Item.Top** attribute value is not validated to be non-negative ReportViewer Windows Forms control that ships with Visual Studio. This value can be written to the RGDI stream with double precision by the ReportViewer control.

<49> Section 2.3.9: The existence of at least one **FIXEDHEADERS.FH** element in the **FIXEDHEADERS** collection is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<50> Section 2.3.9.1: The existence of at least one **FIXEDHEADERS.FH** element in the **FIXEDHEADERS** collection is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<51> Section 2.3.10: The existence of either **FH.HHB** or both **FH.VHR** and **FH.VHL** attributes in the **FH** element is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<52> Section 2.3.10.1: The existence of either **FH.HHB** or both **FH.VHR** and **FH.VHL** attributes in the **FH** element is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<53> Section 2.3.10.3: The existence of either **FH.HHB** or both **FH.VHR** and **FH.VHL** attributes in the FH element is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

<54> Section 2.3.10.4: The existence of either **FH.HHB** or both **FH.VHR** and **FH.VHL** attributes in the FH element is not validated by the ReportViewer Windows Forms control that ships with Visual Studio.

# 6 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

| Section | Description | Revision class |
|---|---|---|
| 5 Appendix A: Product Behavior | Added SQL Server 2019 and Visual Studio 2019 to the product applicability list. | Major |

# 7 Index

**L**



**N**



**O**



**P**



**R**

## S