Rocket Uniface Library 10.4

Struct Annotations

Struct annotations are descriptive data elements (also known as *tags*) that are used to correctly interpret data in Structs and convert data to and from Uniface component data, XML, and JSON.

Each Struct node has a special child Struct to hold annotations. This Struct can be accessed using the `$tags` Struct function and is therefore known as the `$tags` Struct.

Unlike normal Struct members, annotations have no specific position inside the Struct, and are therefore not counted or treated as members of the Struct. This is in contrast to, for example, XML processing instructions, which are treated as normal Struct members because their position in an XML document is relevant, even if they are not part of the document contents.

Example: $tags Struct for XML

Consider the following XHTML code:

```
<div class="note">Text can be <b>bold</b></div>
```

When converted to a Struct, each element and attribute is converted into a Struct member, and the `xmlClass` annotation is set to indicate the original XML constructs. This can be clearly seen in the string returned by `$dbgString` Struct function, in which the `$tags` Struct (where present) is always the first child of the Struct member:

```
[]
 [div]
 [$tags]
 [xmlClass] = element
 [class] = note
 [$tags]
 [xmlClass] = attribute
 [] = Text can be
 [b] = bold
 [$tags]
 [xmlClass] = element
```

To get the value of a particular annotation you can use the `$tags` Struct function. For example, the following code assigns the value of the `xmlClass` annotation of the Struct called `div` to a variable:

```
vClass = vStruct->div->$tags->xmlClass
```

For more information, see $tags and Struct Access Operators.

Annotations for Data Conversion

The ProcScript commands for converting data to and from Structs support fixed sets of tags that are specific to the data format. The annotations typically define the data class or object type and additional metadata unique to the data format.

Because annotations always apply to a specific format, they are ignored when converting to other formats and will be lost. For example, XML annotations are ignored when converting from a Struct to a component.

Note: If annotations are supposed to affect the conversion or the data, you need to ensure that the ProcScript manipulates the Structs based upon the annotations.

When you are creating and populating a Struct in preparation for transformation to another format, you can set your own annotations.

- For `componentToStruct` and `structToComponent`, annotations are used to define the type of Uniface object and the reconnect status of occurrences. For more information, see Struct Annotations for Uniface Component Data.
- For `xmlToStruct` and `structToXml`, annotations are used to indicate the type of XML construct or instruction. For example, the `xmlClass` tag contains information about how the data was stored in the source XML (for example, attribute or element). This is useful when doing a round-trip conversion, so no information is lost. For more information, see Struct Annotations for XML.
- For `jsonToStruct` and `structToJson`, annotations indicate the JSON class and data types. For more information, see Struct Annotations for JSON.

Some data formats have their own metadata, which can be added as annotations. For example, the XML DOCTYPE specification and XML Declaration are required in XML data.

Annotations for Data Interpretation

Some annotations may help to interpret data correctly. For example, when converting from XML to Struct, the XML Schema may specify that the value was originally of type `Duration`, a type that has no equivalent in Uniface. In this case, the value must be passed as a String but the original data type is added as an annotation so that you can apply logic to handle this data type.

Annotations may also provided as read-only information to the developer. Such annotations can be ignored on the conversion back. For example, when using an XML Schema, the xmlDataType tag is added to specify the primitive type of a scalar element. When converted back using the same XML Schema, `xmlDataType` is ignored. (It is not possible to change the XML Data Type by setting this tag as the data type is determined by the XML Schema.)

User-Defined Annotations

You can also add your own annotations if you are creating customized conversion routines. In this case, you should ensure that you adopt a naming policy that prevents tag name conflicts with future versions of Uniface. For example, you could use tag names that begin with an underscore (_).

Important: Uniface guarantees that future tag names will never use the underscore as first character.

Related concepts

Struct Annotations for XML
Struct Annotations for Uniface Component Data
Structs for JSON Data
$tags
$istags
$dbgString

Related tasks

Example: Tags Inheritance