

edikt

BinXed 1.0

User's Guide

BinXed: The BinX Editor User's Guide

edikt::BinXed 1.0 User's Guide
Edition 1.0
© University of Edinburgh 2005

edikt

Old College, University of Edinburgh
Edinburgh, Scotland, EH8 9YL
www.edikt.org support@edikt.org

The BinX Editor includes software developed by the Apache Software Foundation (www.apache.org).

Table of Contents

1	Installing and Running BinXed	3
1.1	BinXed Interface	4
1.2	The Menus	4
1.3	Popup Menus	5
1.4	Toolbar Buttons	5
1.5	The Property Window	6
2	BinX Document	7
2.1	Primitive Data Elements	7
2.2	Complex Data Elements	7
2.3	How to Use the BinX Document	8
3	Create a BinX Document	9
3.1	Adding Primitive Elements	9
3.2	Defining Complex Types	10
3.2.1	Defining a Struct	10
3.2.2	Defining a Union	10
3.2.3	Defining an Array	11
3.3	Adding Complex Elements	11
3.4	Save the Document	11
3.5	Editing a BinX Document	12
3.6	Opening a BinX Document	12
3.7	Error or warning messages	12
3.8	Modify Properties	13
3.9	Save as another Document	13
3.10	Browse a BinX Document	13
3.11	Validate a BinX Document	14
3.12	View BinX Schema	14



Getting Started

The BinX Editor (BinXed) is a visualisation tool to help design BinX documents. A BinX document is an XML text file based on the BinX mark-up language for the purpose of describing a binary data file. The data in the binary file can be accessed by means of the BinX library that provides basic access to the data through the BinX document.

1 Installing and Running BinXed

The BinXed is a Java application that depends on the Java virtual machine. The BinX editor has been developed on **Java version 1.4.2**. for Solaris, Linux and Windows XP.

Download the compressed Java Archive `InstallBinXed.jar` file from the Edikt Website

<http://www.edikt.org/binx>

and place the compressed it in a suitable directory.

Install BinXed by running the following command:

```
java -jar InstallBinXed.jar
```

On Window XP, double-clicking on the file will have the same effect.

Follow the on-screen instruction to install BinXed. The installation will create a short-cut file, `BinXed.sh` if you are running on Solaris or Linux; or `BinXed.bat` if you are using Windows XP.

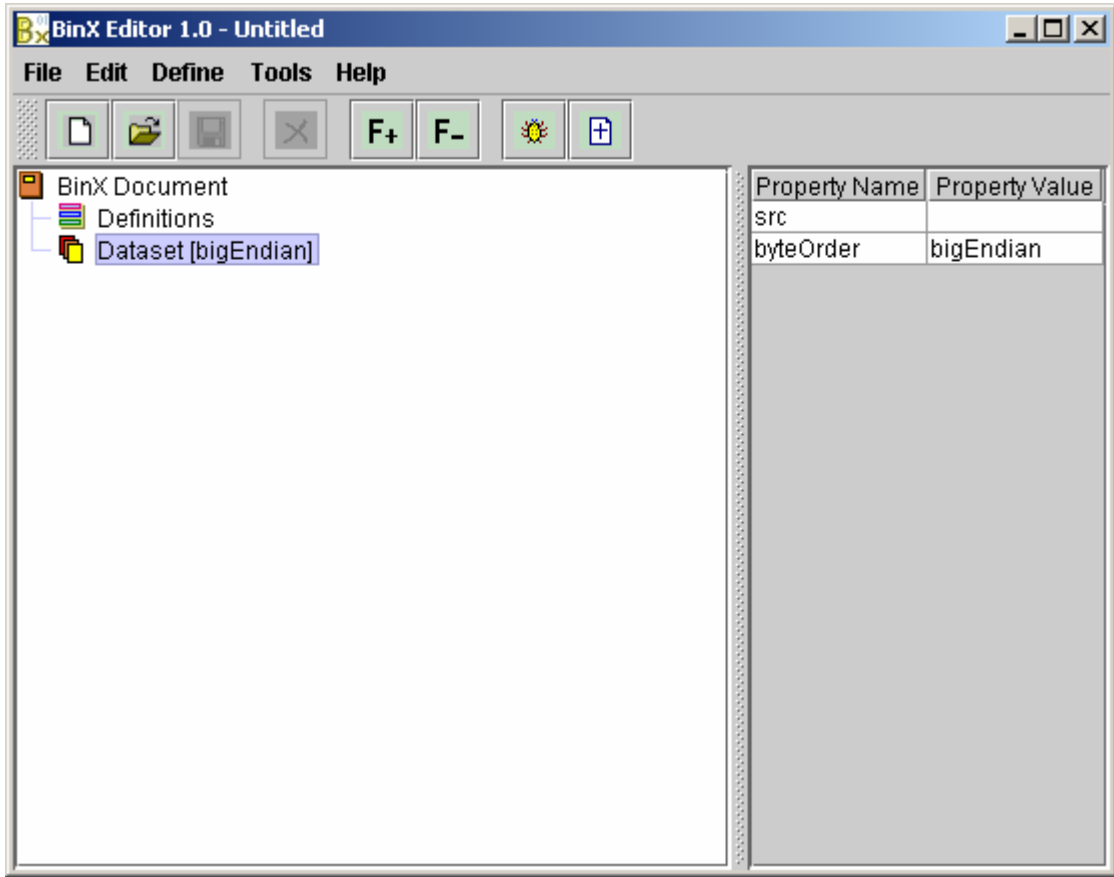
To run BinXed on Solaris and Linux you may need to make the short-cut an executable file, then invoke it using:

```
BinXed.sh
```

On Windows XP, simply double-click on the BinXed icon.

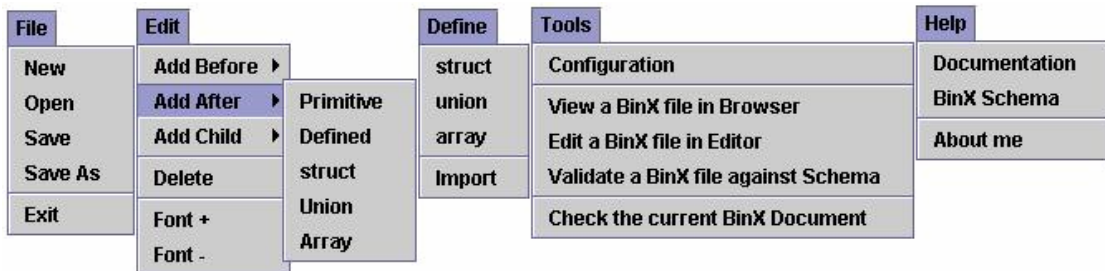
1.1 BinXed Interface

There are two panes in the main window, the tree view on the left pane shows all the data elements in a BinX document as a hierarchical structure. The right-hand side pane contains a list of properties for a selected data element in the document tree.



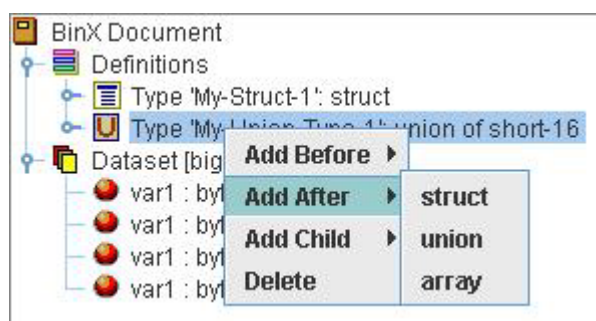
1.2 The Menus

This is the main menu showing the sub menu items. Some of the functions can be done either through selecting the buttons on the toolbar, or by triggering a popup menu in the document tree view via the right mouse button.











1.3 Popup Menus

Popup menus are context-sensitive (i.e. the menu items change depending on what part of the tree is selected by the mouse). The example below shows a popup menu triggered on a defined type element. In this case, it is possible to add a sibling either before this element or after it, but only the three complex types are allowed for user-defined type. As a union-based type, it is also possible to add additional union cases, so Add Child->Case will also appear on the menu. To show a popup menu, trigger it with the mouse cursor over the target tree node (data element). This is done by right-clicking the mouse button while the cursor is over the required item.



1.4 Toolbar Buttons

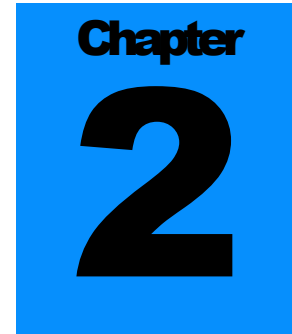
The toolbar in the main window provide users with a short-cut to menu item, as follows:

-  New Document
-  Open Document
-  Save Document
-  Delete
-  Increase Font size
-  Decrease font size
-  Validate Document
-  Import type definitions from another Document.

1.5 The Property Window

The property window on the right-hand side of the main window shows the properties of the currently selected element. The example shown below is the properties of a union type. The `typeName` is the name given in the define type dialog, the `byteOrder` is the default, `blockSize` is also the default value, and the discriminant type is selected as `short-16`.

Property Name	Property Value
<code>typeName</code>	My-Union-Typ...
<code>byteOrder</code>	
<code>blockSize</code>	0
<code>discriminant</code>	short-16



An introduction to BinX

BinX (Binary in XML) is a mark-up language to describe the format of the elements in a binary data file. It is a simple XML document based on the XML schema for BinX. A single BinX document maps to a particular binary data file, and can describe complex data structures. The BinX document can be written manually, or created using the BinX document editor – BinXed.

2 BinX Document

A BinX document contains a standard XML instruction line `<?xml version="1.0">`, and a `<binx>` root element followed by two child elements, `<definitions>` and `<dataset>`. The `<definitions>` element is an optional section for defining abstract data types. The `<dataset>` section contains the description of the data elements in a binary file in sequential order.

2.1 Primitive Data Elements

The supported primitive data elements in the BinX library 1.1 include characters, shorts, integers and floating point numbers. The integers (16, 32 and 64-bits) can be signed or unsigned.

2.2 Complex Data Elements

There are three types of complex data elements; structures, unions and arrays. The struct is a record that contains a limited number of elements grouped together. The member element can also be a struct so that a structure can be embedded in another structure. The union type is a data structure to allow a choice of element based on a given discriminant value. The data type of the elements in the union, can be either primitive type or any of the complex ones. The array structure describes a repetitive list of any type of data element that can be described by the BinX mark-up language. There are three types of arrays, a fixed array contains a fixed number of elements with the number given in the description file (the BinX document), a variable-length array has the number of elements stored in the binary file and the type and location of this number is given in the description file, and the third type of array is a plain stream of any data elements.

2.3 How to Use the BinX Document

A well-designed BinX document can be used by the BinX library to read the binary data on a platform-independent fashion. For example, if the binary data file was produced on a big-endian machine, the data file can still be read on a little-endian machine using BinX. To read the data file without conversion, mark it up as big-endian in the BinX document and read the data using the BinX library routines.

Chapter 3

Using BinXed

3 Create a BinX Document

It is simple to create a BinX document with BinXed. When BinXed is started, a default empty document named 'untitled' is ready for edit. To create a blank new document while editing a document, simply select menu File->New, or click the "New" button on the toolbar. To create a BinX document add elements to the tree structure in the left pane which represents the data elements in the binary file. The binary data file is assumed to have some structure. An arbitrary file such as a text document is unsuitable for the BinX library. Structured data elements in a binary file are normally grouped into chunks. The chunks may be flat with primitive data elements, or hierarchical with embedded smaller data chunks. We use primitive types to represent a primitive data value, and complex types to show the chunks and relationship between the chunks. A data chunk can be represented by the struct complex type. A list of data chunks that share the same structure but have different values can be represented as an array type. When chunk structure is conditional, it can be represented by union type.

3.1 Adding Primitive Elements

Primitive elements can be added to the dataset, which is a structure data chunk. To add a primitive element to the dataset, select the dataset tree node in the tree view, then select menu Edit->Add Child->Primitive. Another way is to trigger a popup menu on the dataset tree node, right-click the mouse button with the cursor floating over the dataset tree node. The popup menu contains Add Child->Primitive. In the dialog window that appears, type in a variable name (optional), select a primitive data type, and then click the OK button.



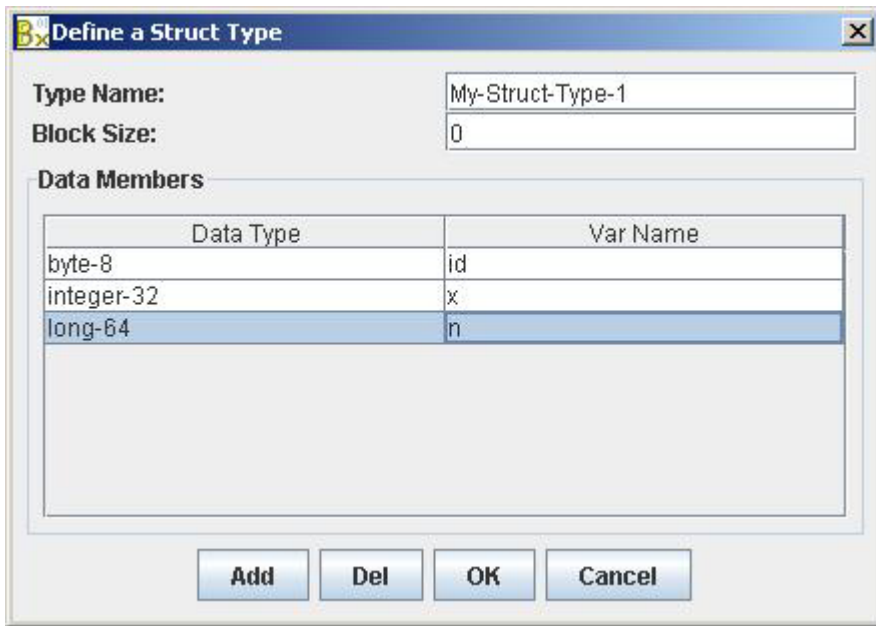
If the "more" button is selected, then the window appears again for another primitive element. This can be helpful when adding a list of primitive elements. The "Cancel"

button closes the dialog box without adding another element. Note that after selecting the “More” button, the data elements are actually inserted into the document tree even though they are not immediately visible. So the final Cancel click only cancels the last data element and not all of the elements added.

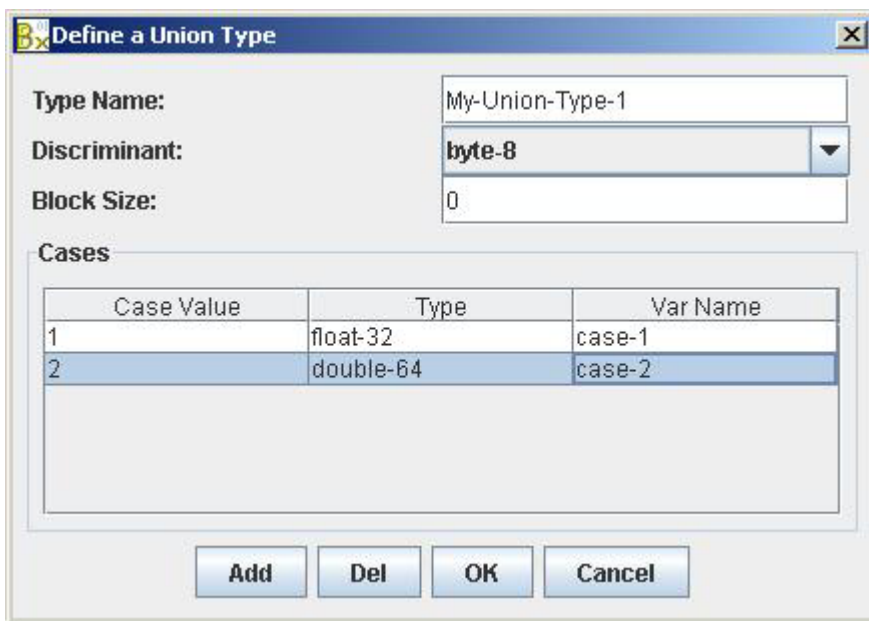
3.2 Defining Complex Types

To define a struct, union or array type that can be used later in the document with the useType tag to reference the type. Defining a complex type helps minimising the work of adding multiple chunks of data of the same structure.

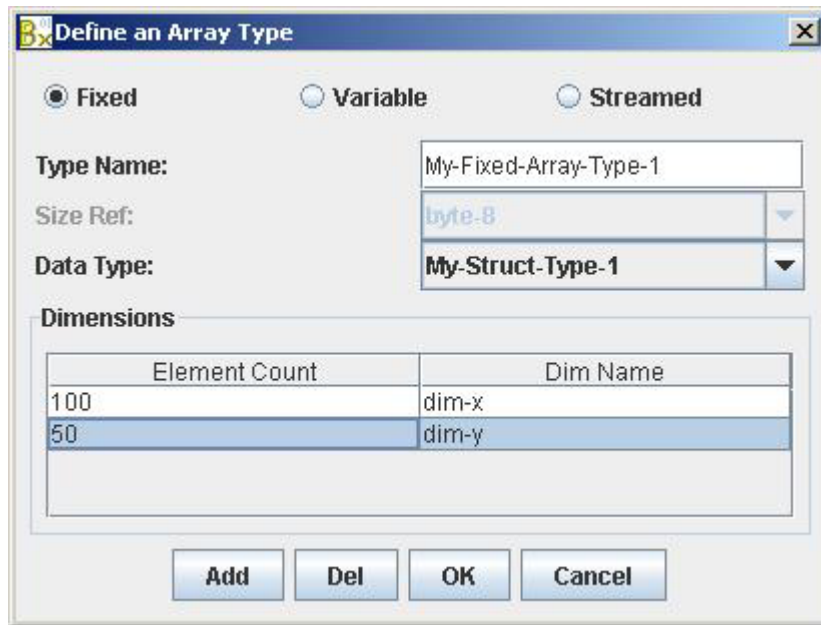
3.2.1 Defining a Struct



3.2.2 Defining a Union

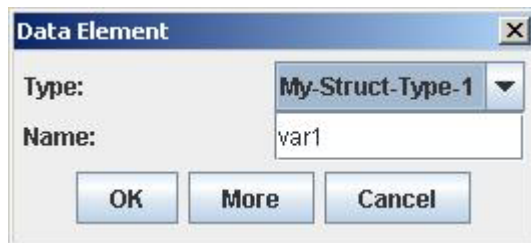


3.2.3 Defining an Array



3.3 Adding Complex Elements

A complex data element can be a struct, a union, or an array. Such an element can be added to the dataset, or defined as a complex type beforehand, and added as a reference to the dataset later. To add a complex type directly, follow the menu for struct, union and array. To add a reference, select menu Add Defined.

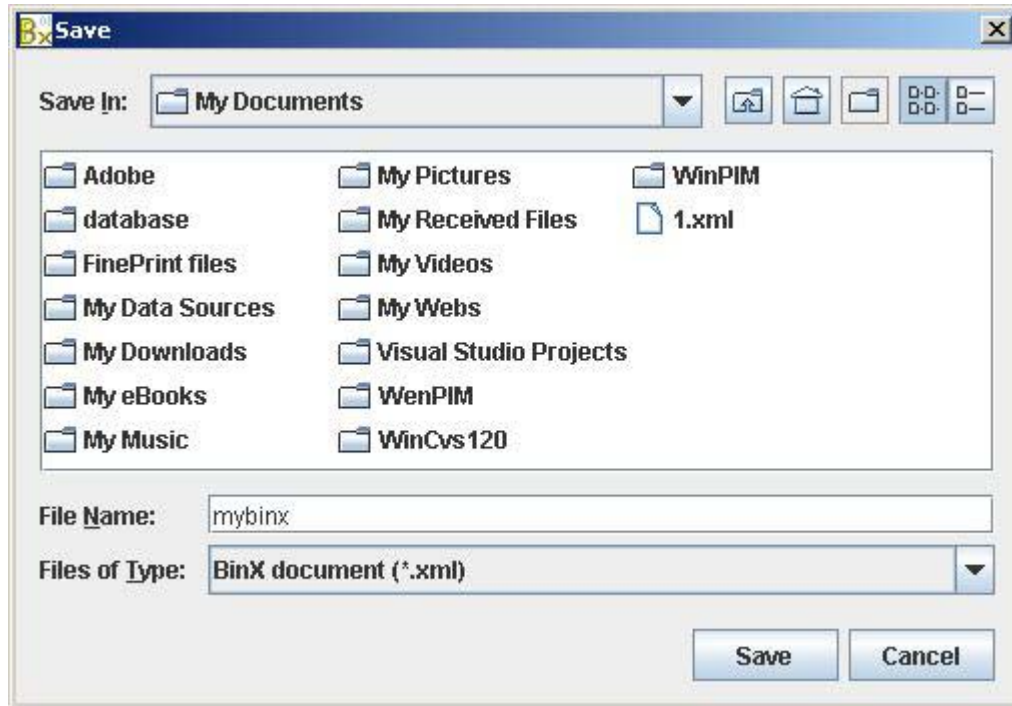


3.4 Save the Document

To save the document, select menu File->Save. If the document has been modified and either the menu File->Exit or the close button on the window is selected, it prompts for saving the document before quitting the application.



Choosing Yes will save the document, and if there is no file name associated with the current document, a file chooser appears to ask for a file name.



3.5 Editing a BinX Document

A BinX document can be edited in the BinXed editor, an XML editor, or even a text editor. A BinX document produced with any editor can all be loaded into BinXed and edited with it. After the document is loaded, a validator is executed to check the syntax of the elements and shows the error or warning messages inside a popup window. Validation is also done with saving the document.

3.6 Opening a BinX Document

To open a document, select menu File->Open, or select button Open on the toolbar. A file chooser dialog window appears to help select a BinX file for editing.

3.7 Error or warning messages

After a BinX document is loaded or saved, it is validated to see whether there are any mistakes that violate the BinX grammar. Some are errors that must be corrected, and others are warnings. The example below shows that the document contains a dataset without a binary file attached.



Possible error messages are:-

Error: duplicate type name. – User-defined types must have unique type names.

Error: duplicate case value.

Error: Array has no dimension element.

Error: array dimension is of invalid size.

Error: invalid type name. – referenced type wrong.

Error: recursively referencing the type. – use itself.

Error: type used before defined.

Possible error warnings are:-

Warning: dataset has no binary file attached.

Warning: dataset has no element.

Warning: struct contains no member element.

Warning: union contains no case element.

Warning: Streamed array contains more than one dimension.

Warning: first dimension count of a variable-sized array ignored.

Warning: duplicate variable name.

Warning: BinX library version 1.x does not support arrays containing union.

Warning: BinX library version 1.x does not support arrays containing variable-sized arrays.

3.8 Modify Properties

There are two types of property values, ones that must be typed either as text or as numbers and others that must be selected from a list of names. The binary file name, for example, is a property of the text value for the dataset element. The property of the blockSize is an input value for only positive integers. Entering a non-digit text will be ignored. The byteOrder is a selectable property, clicking the property value of byteOrder will invoke a selection among default, big-endian, and little-endian. Once the property values are changed, they are reflected in the document, and may be shown in the left tree view.

3.9 Save as another Document

Select menu File->Save As will prompt for another file name and save the document by that name.

3.10 Browse a BinX Document

Since BinX documents are XML files, they can be viewed with a browser. The BinX editor can be configured to invoke a specific browser to view a particular BinX document. On Windows, it is usually Internet Explorer. To specify a browser for BinXed, start the application on the command line with a `-D` parameter like this:

```
java -DBROWSER=C:\Program Files\Internet Explorer\iexplore.exe  
org.edikt.binx.tools.binxed.BinXed
```

Another method is to put the browser path in the configuration file.

3.11 Validate a BinX Document

It is sometimes necessary to validate a BinX document against the correct schema. To do so, select the menu Utility->Validate a BinX document. The BinX schema can be configured in the configuration file, by default the schema is referred to the one on the edikt web site. This validation only works based on the BinX schema, it may not be able to find out the warnings that are reported by the internal validator when opening or saving a document.

3.12 View BinX Schema

The BinX schema is a XSD file that is also be viewed with the configured browser.
