TransportableItems for REALbasic

With these great classes, you can save many RB data types to strings with minimal work. This solves two common problems: saving to preferences and sending over sockets.

The TransportableItems classes all export to XML using REALbasic's built-in XML classes, so they require nothing but REALbasic.

The TransportableItem Interface

Everything that wants to be "transportable" needs to implement this interface. It's actually very simple. Because interfaces require that every class which implements them include their method declarations, I'm not going to document these methods for each and every class.

`AsXMLElement() As XMLElement`
This is the code that turns the item into an XMLElement. This element can be used with an XMLDocument via XMLDocument.ImportNode.

`Constructor(XML As XMLNode)`
This is essentially the reverse of AsXMLElement. Each class implements two constructors. This is one, and required. The second one is different for each class. They will be covered later.

`Type() As Integer`
Returns a 1-8 integer that corresponds to any of the TransportManager.kTransportable* constants. It is used to identify which type of TransportableItem we're dealing with.

So just remember, these methods are available for every TransportableItem class.

TransportManager

TransportManager is a Module that implements a few very useful Methods and Constants.

Constants

There are 8 integer Constants that match with the data from TransportableItem.Type:

kTransportableString = 1
kTransportableInteger = 2
kTransportableDouble = 3
kTransportableBoolean = 4
kTransportableArray = 5

kTransportableDictionary = 6
kTransportableColor = 7
kTransportableFolderItem = 8

Methods

TransportableItem.AsString(Extends Item As TransportableItem) As String

You may be wondering why this simply isn't implemented in the TransportableItem interface. That's because the code contained would be identical in every class, so there's no reason to make every class implement it, when this method will ensure the same thing.

This Method returns the XML data as a string, rather than as an XMLElement.

DecodeTransportableItem(XML As XMLNode) As TransportableItem

This function will turn the XMLNode back into one of the classes. It pretty much just determines what kind of class it needs to become, then creates one with it's constructor. The object returned could be any of the classes, so you'll need to use TransportableItem.Type to determine what you are working with. You'll probably want to type cast it too:

```
Dim Arr As TransportableArray
Dim Obj As TransportableItem
Dim XML As XMLNode // Assume this has been populated somewhere

Obj = DecodeTransportableItem(XML)
if Obj <> Nil and Obj.Type = kTransportableArray then
  Arr = TransportableArray(Obj)
else
  msgbox "This is NOT an array"
end
```

Save(Extends Item As TransportableItem) As String

This method is very similar to TransportableItem.AsString, except it'll return a complete XML document as a string, rather than just the element. Use this on a TransportableDictionary, and you've got a great Preferences method.

Transport(Extends Sock As TCPSocket, Item As TransportableItem)

You can send any TransportableItem over a TCPSocket by calling this Method on it:

```
Dim S As TransportableString
S = New TransportableString("Hello World")
Sock.Transport S
```

Transport(Extends TheItem As TransportableItem, ViaSocket As TCPSocket)
Maybe you want to call it backwards?

```
Dim S As TransportableString
S = New TransportableString("Hello World")
S.Transport Sock
```

The "Basic 6"

TransportableBoolean, TransportableColor, TransportableDouble, TransportableFolderItem, TransportableInteger, and TransportableString all have a Value property which can be used to get or set the class' value. The data type will match class. For example, TransportableBoolean's Value will be a Boolean.

These 6 classes also implement a second Constructor method that takes a parameter used to set their value. For example, you can use New TransportableString("Hello World") or TransportableBoolean(true)

These are the basic 6 classes. They are essentially the same. Their only differences are in their RB data types.

The TransportableArray and TransportableDictionary classes are quite different, however.

TransportableArray

Methods

Append(NewItem As TransportableItem)
Add any of the TransportableItem classes to the end of the array.

Count() As Integer
Returns a 1-based index of the number of items in the array.

HasString(S As String) As Boolean
Searches through the array for a TransportableString with the value of s. Returns true on success.

Remove(Index As Integer)
Removes an item from the array using it's 0-based index.

Properties

Item(Index As Integer) As TransportableItem
Get or set the value of an item using it's 0-based index. Just like a RB array, don't call item with an index that does not exist.

TransportableDictionary

Methods

`Count() As Integer`
Returns a 1-based integer of the number of items in the dictionary

`HasKey(KeyValue As String) As Boolean`
If there is an item with KeyValue in the dictionary, true will be returned. If not, you get false.

`Key(Index As Integer) As String`
Returns the Key for an item using it's 0-based index in the dictionary. Just like the RB dictionary, don't call it Index doesn't exist.

Properties

`Value(KeyValue As String) As TransportableItem`
Get or set an item in the dictionary using it's key. If KeyValue does not exist, Nil will be returned.

You're Done!

You're now armed to send data flying all over the place! I use these classes in just about every project. I primarily use them for Preferences, and for document creation. They are really simple to work with, once you get use to them.

Please do not distribute this file at all. If you would like to send it to a friend, please link them to the original site, as a new version may have been released.

<thom@thezaz.com>
<http://www.thezaz.com/>