TNMUUProcessor component

Unit
NMUUE

Description
The TNMUUProcessor component MIME encodes or UUEncodes files and decodes MIME encoded or UUEncoded files.

TNMUUProcessor component

TNMUUProcessor Properties

In TNMUUProcessor

InputStream
Method
OutputStream

Derived from TComponent

ComObject
ComponentCount
ComponentIndex
Components
ComponentState
ComponentStyle
DesignInfo
Owner
Tag
VCLComObject

TNMUUProcessor Methods
TNMUUE
Legend

In TNMUUProcessor
🔑 Decode
    🔑   Encode

Derived from TComponent
DestroyComponents
Destroying
FindComponent
FreeNotification
FreeOnRelease
GetParentComponent
HasParent
InsertComponent
RemoveComponent
SafeCallException

Derived from TPersistent
Assign
GetNamePath

Derived from TObject
ClassInfo
ClassName
ClassNameIs
ClassParent
ClassType
CleanupInstance
DefaultHandler
Dispatch
FieldAddress
Free
FreeInstance
GetInterface
GetInterfaceEntry
GetInterfaceTable
InheritsFrom
InitInstance
InstanceSize
MethodAddress
MethodName
NewInstance

TNMUUE Events

In TNMUUProcessor

About the TNMUUProcessor component

Purpose
The purpose of the NMUUProcessor is to encode files for transmission across the internet or an intranet, and to decode them so that they can be used in their original form.

Tasks
By simply setting the InputFile property to the file to be processed, the Method property to the style of encoding/decoding to be used, the OutputFile property for the resulting file, and finally either calling the Encode or Decode methods, a file can be encoded or decoded as needed.

InputStream property

Applies to
TNMUUProcessor component

Declaration
```
property InputStream: TStream;
```

Description
The InputStream property specifies the stream that will be encoded or decoded.

Scope: Published
Accesability: RunTime

Note: This property must be set before the **Encode** or **Decode** methods are called.

See Also

[Decode](#) method
[Encode](#) method
[OutputStream](#) property

Example

To recreate this example, you will need to create a new blank Delphi application.

Place a TMemo, 2 TEdits, 3 TButtons, a TRadioGroup, a TOpenDialog, and a TNMUUProcessor on the form.

Memo1: Status window
Edit1: Name of file to encode/decode (for InputStream)
Edit2: Name of file to write encoded/decoded data to (for OutputStream)
Button1: Browse for a file to encode/decode
Button2: Encode selected file
Button3: Decode selected file
RadioGroup1: Encode/Decode methods (fille **Items** property of RadioGroup 1 with MIME and UUEncode/Decode (2 items) <u>in that order</u>)
OpenDialog1: For selecting the file to encode/decode

Insert the following code into Button1's OnClick event:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  if OpenDialog1.Execute then
    Edit1.Text := OpenDialog1.FileName;
end;
```

When Button1 is clicked, the OpenDialog is displayed, allowing the user to select a file. If the Open Button is clicked, the filename is placed into Edit1.

Insert the following code into Button2's OnClick event:

```
procedure TForm1.Button2Click(Sender: TObject);
var
  FS1,
  FS2: TFileStream;
begin
  // Encode
  FS1 := TFileStream.Create(Edit1.Text, fmOpenRead);
  FS2 := TFileStream.Create(Edit2.Text, fmCreate);
  try
    case RadioGroup1.ItemIndex of
      0: NMUUProcessor1.Method := uuMime;
      1: NMUUProcessor1.Method := uuCode;
    end;
    NMUUProcessor1.InputStream := FS1;
    NMUUProcessor1.OutputStream := FS2;
    NMUUProcessor1.Encode;
  finally
    FS1.Free;
    FS2.Free;
  end;
end;
```

When Button2 is clicked, 2 filestreams are created. Depending on which item in RadioGroup1 is selected, the **Method** property is set to either uuMime for MIME encoding, or uuCode for UUEncoding. The

InputStream property is set to FS1, which contains the file that will be encoded. The **OutputStream** property is set to FS2, which will contain the encoded file data. The **Encode** method encodes the data in FS1 and stores it in FS2. Note that FS1 and FS2 are freed in the **finally** section of a **try**. . .**finally** block to ensure their resources are freed.

Insert the following code into Button3's OnClick event:

```
procedure TForm1.Button3Click(Sender: TObject);
var
  FS1,
  FS2: TFileStream;
begin
  // Decode
  FS1 := TFileStream.Create(Edit1.Text, fmOpenRead);
  FS2 := TFileStream.Create(Edit2.Text, fmCreate);
  try
    case RadioGroup1.ItemIndex of
      0: NMUUProcessor1.Method := uuMime;
      1: NMUUProcessor1.Method := uuCode;
    end;
    NMUUProcessor1.InputStream := FS1;
    NMUUProcessor1.OutputStream := FS2;
    NMUUProcessor1.Decode;
  finally
    FS1.Free;
    FS2.Free;
  end;
end;
```

When Button3 is clicked, 2 filestreams are created. Depending on which item in RadioGroup1 is selected, the **Method** property is set to either uuMime for MIME decoding, or uuCode for UUDecoding. The **InputStream** property is set to FS1, which contains the file that will be decoded. The **OutputStream** property is set to FS2, which will contain the decoded file data. The **Decode** method decodes the data in FS1 and stores it in FS2. Note that FS1 and FS2 are freed in the **finally** section of a **try**. . .**finally** block to ensure their resources are freed.

Insert the following code into NMUUProcessor1's OnBeginDecode event:

```
procedure TForm1.NMUUProcessor1BeginDecode(Sender: TObject);
begin
  Memo1.Lines.Add('Decoding. . .');
end;
```

When a decoding operation begins, the **OnBeginDecode** event is called. In this instance, Memo1 is updated to indicate that the **InputStream** is being decoded.

Insert the following code into NMUUProcessor1's OnEndDecode event:

```
procedure TForm1.NMUUProcessor1EndDecode(Sender: TObject);
begin
  Memo1.Lines.Add('Decode finished');
end;
```

When a decoding operation completes, the **OnEndDecode** event is called. In this instance, Memo1 is updated to indicate that the **InputStream** property has been decoded and the results are stored in the **OutputStream** property.

Insert the following code into NMUUProcessor1's OnBeginEncode event:

```
procedure TForm1.NMUUProcessor1BeginEncode(Sender: TObject);
begin
  Memo1.Lines.Add('Encoding. . .');
end;
```

When an encoding operation begins, the **OnBeginEncode** event is called. In this instance, Memo1 is updated to indicate that the **InputStream** property is being encoded.

Insert the following code into NMUUProcessor1's OnEndEncode event:

```
procedure TForm1.NMUUProcessor1EndEncode(Sender: TObject);
begin
  Memo1.Lines.Add('Encode finished');
end;
```

When an encoding operation completes, the **OnEndEncode** event is called. In this case, Memo1 is updated to indicate that the **InputStream** property has completed being encoded and the results are stored in the **OutputStream** property.

Method property

Applies to
TNMUUProcessor component

Declaration
`property Method: UUMethods;`

Description
The Method property determines the method in which InputStream is encoded or decoded. The possible values are uuMime for MIME encoding/decoding, and uuCode for UUEncoding/UUDecoding.

Default: uuMime

Scope: Published
Accessibility: DesignTime, RunTime

Range:
    uuMime - MIME/Base 64 encoding/decoding
    uuCode - UUEncoding/Decoding

See Also

Decode method
Encode method

OutputStream property

Applies to
TNMUUProcessor component

Declaration
`property OutputStream: TFileStream;`

Description
The OutputStream property specifies the stream that will hold the encoded data if the InputStream is processed using the Encode method, or it will contain the decoded data if the Decode method is called.

Scope: Published
Accessibility: RunTime

Note: This property must be set before the **Encode** or **Decode** methods are called.

See Also

Decode method
Encode method
InputStream property

Decode method

Applies to
TNMUUProcessor component

Declaration
```
procedure Decode;
```

Description
The Decode method decodes the stream data specified by the InputStream property using the method determined by the Method property, and saves the data into the stream specified by the OutputStream property.

See Also

Encode method
InputStream property
OnBeginDecode event
OnEndDecode event
OutputStream property

Encode method

Applies to
TNMUUProcessor component

Declaration
```
procedure Encode;
```

Description
The Encode method encodes the data in the stream specified by the InputStream property using the method determined by the Method property, and saves it into the stream specified by the OutputStream property.

See Also

Decode method
InputStream property
OnBeginEncode event
OnEndEncode event
OutputStream property

OnBeginEncode event

Applies to
component

Declaration
procedure OnBeginEncode: TNotifyEvent*;*

Description
The OnBeginEncode event is called when a stream begins encoding.
The **Encode** method initiates this event

See Also

Encode method
OnEndEncode event

OnEndEncode event

Applies to
TNMUUProcessor component

Declaration
procedure OnEndEncode: TNotifyEvent;

Description
The OnEndEncode event is called when a stream ends encoding.
This event is called at the conclusion of the **Encode** method.

See Also

Encode method
OnBeginEncode event

OnBeginDecode event

Applies to
TNMUUProcessor component

Declaration
procedure OnBeginDecode: TNotifyEvent;

Description
The OnBeginDecode event is called when a stream begins decoding.
This event is initiated by the **Decode** method.

See Also

Decode method
OnEndDecode event

OnEndDecode event

Applies to
TNMUUProcessor component

Declaration
procedure OnEndDecode: TNotifyEvent;

Description
The OnEndDecode event is called when a stream finishes decoding.
This event is called at the conclusion of the **Decode** method.

See Also

Decode method
OnBeginDecode event

UUMethods type

Declaration
type
  UUMethods = (uuMime, uuCode);

Description
The UUMethods type defines the types of encoding and decoding available to the TNMUUProcessor
component.

uuMime uses the MIME specification for encoding/decoding files
uuCode uses UUEncode to encode files and UUDecode to decode files

Heirarchy

TObject
|
TPersistent
|
TComponent

Legend

- Run-time only
- Read-Only
- Published
- Protected
- Key item