About Fishhead Software

URL:

www.fishware.com

Email: (for fastest reponse)

support@fishware.com

Voice:

630-892-6958

Fax:

630-892-6958

Write:

Fishhead Software
912 LaFayette Street
Aurora, IL 60505

Source Code

Remember: You can not distribute the source code or use any portion of it to create commercial, shareware, or freeware ActiveX controls or similar software.

Modifying the source code to create an OCX for your company

Please do the following to the modified control:
- Change the project name and description;
- Change the class name of the control;
- Change the EXE name;
- Change version information to reflect your company;

Note: other people have bought the control and may sell their applications to the same client. Taking these precautions will save you and our clients many headaches.

System Requirements

The system requirements are as follows:

Windows 95, Windows 98, Windows NT 4.0 or Windows 2000
Visual Basic 4.0 (32 bit) or Visual Basic 5.0 or higher
486 or higher processor
8 MB RAM
4 MB disk space

Copyright and Trademarks

fsParse ActiveX Control Version 2.00 A member of fsVBActiveX

About Fishhead Software   Copyright

fsParse ActiveX Control makes it easy to extract data from a file or string.

fsParse5.ocx for use with Visual Basic 4.0 (32-bit) and 5.0
GUID = 16C97E7C-E40B-11D2-A399-000000000000
Requires MSVBVM50.DLL

fsParse6.ocx for use with Visual Basic 6.0
GUID = 363CED67-E40B-11D2-A399-000000000000
Requires MSVBVM60.DLL

Getting Started
Frequently Asked Questions
Features and Uses

Properties
Methods
Events

Visit Fishhead Software on the WEB
www.fishware.com

Getting Started

fsParse is easy to use.

Basic Setup:
Parsing a file

Basic Setup

1. Create a new project. Select the "standard exe" project type. Name this project Server.

2. From the Component dialog, select "Fishhead Software fsParse Control for VB 5" or "Fishhead Software fsParse Control for VB 6" if using Visual Basic 6. If you are using then demo version, then select "Fishhead Software fsParse Control Demo".

3. Create a form.

4. Select fsParse control from the tool bar and draw it on the form.

Parsing a file

This example will parse an ASCII file and place the results into MSFlexGrid. The record layout is as follows:

Position	Field
1 - 14	Phone number
15 - 41	Owner
42 - 91	Name of the business
92 -	Description

```
Option Explicit

Private Sub Form_Load ()

       ' ** Set the number of columns and their start positions
       fsParse1.Columns = "1, 15, 42, 92"

       ' ** OpenFile will locate one the first record if successful
       fsParse1.FileName = "example1.txt"

       fsParse1.Action = fsAMBegin

End Sub

Private Sub fsParse1_Begin()

      ' ** Place Initilize code here

      Grid1.Rows = 1
      Grid1.Redraw = False
      ProgressBar1.Value = 0

End Sub

Private Sub fsParse1_Change()

     ' ** Process the data file here

      ' ** Update the grid
      Grid1.AddItem fsParse1.Field(0) & vbTab & fsParse1.Field(1) & vbTab & fsParse1.Field(2) & vbTab &
         fsParse1.Field(3)
      ' ** Update status bar
      ProgressBar1.Value = fsParse1.PercentDone

End Sub

Private Sub fsParse1_Done()

      ' ** Place end processing here
```

```
    ProgressBar1.Value = 0
    Grid1.Redraw = True

End Sub
```

Frequently Asked Questions

- **Can I use fsParse5 with fsParse6 in the same set of applications?**

Yes.

Features and Uses

Features

- Small footprint;
- Uses standard Visual Basic runtime DLL, no additional DLLs or OCXs required;
- Safe and easy to use;

Uses

- Retrieve delimitted data out of a speadsheet file;
- Parsing grid data;
- Search for text within a file;
- Command line parsing;
- Parsing strings;

Properties

Action
Columns
CompareMode
FieldSeparator
Filename
MatchQuotes
ParseMode
Record
RemoveQuotes
SearchText

Action Property

Specifies an action to be performed.

Syntax

object.**Action** [= *fsActionMode*]

The Action property syntax has these parts:

Part	**Description**
object	An object expression that evaluates to fsParse object.
fsActionMode	(Optional) Which action to perform.

Settings

```
Public Enum fsActionMode
    fsAMStop = 0
    fsAMBegin = 1
End Enum
```

Remarks

fsAMBegin starts the parsing process, while fsAMStop will stop the process. If the process raises the end of file condition, then the action is automatically stopped. This property cannot be set at design time.

Example

```
fsParse1.Action = fsAMBegin
```

Columns Property

Sets or returns the beginnning column position for each field in a data file or string.

Syntax

object.**Columns** [= *string*]

The Columns property syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object.
string	(Optional) The beginning column positions. The default is a zero length string.

Remarks

Use this property when the column beginning position is known for each column in the data file or string. fsParse will always begin with position one as the start of the first column. Setting this property tells fsParse to ignore the FieldSeparator property.

Example

```
fsParse1.Columns = "10, 30, 50, 75, 90"   ' tells fsParse columns begin at 1, 10, 30, 50, 75 and 90
```

CompareMode Property

Sets or returns the comparison mode for strings.

Syntax

object.**CompareMode** [= *compare*]

The CompareMode property syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object.
compare	(Optional) If provided, compare is a value which will determine the comparison mode used by test within fsParse. The default is fsCMBinaryCompare.

Settings

```
Public Enum fsCompareMode
    fsCMBinaryCompare = vbBinaryCompare
    fsCMTextCompare = vbTextCompare
End Enum
```

Remarks

Use this property as you would with Visual Basic's CompareMode property. Setting the Visual Basic's **CompareMode** property has no effect on this property.

Example

```
fsParse1.CompareMode = fsCMTextCompare
```

FieldSeparator Property

Sets or returns the field separators for a data file or string.

Syntax

object.**FieldSeparator** [= *string*]

The FieldSeparator property syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object.
string	(Optional) When provided, changes how fsParse determines the data position. The default is tab, space and comma.

Remarks

Use this property to change how fsParse will determine the data position for each field by adding or removing seperators.

Example

```
fsParse1.FieldSeparator = ", " & Chr$(9) & "|"      ' Use comma, space, tab, and the pipe to parse the data
```

Filename Property

Sets or returns the name of the file to be parsed.

Syntax

object.**Filename** [= *string*]

The Filename property syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object;
string	A valid file name. The default is a zero length string.

Remarks

Set this property to the name of file to be parsed.

Example

```
fsParse1.Filename = "MYDATA.CSV"
```

MatchQuotes Property

Sets or returns rather quoted strings are recognized as a delimited field.

Syntax

object.**MatchQuotes** [= *Boolean*]

The MatchQuotes property syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object;
Boolean	(Optional) Determines if quoted strings should be recognized as a delimited field. Default is True.

Remarks

This property needed to help parse files or strings that use single or double quotes to mark data. This is typical of most spread sheets and databases when they output data to a text file.

fsParse recognizes single and double quote pairs, but not single and a double nor double and a single as a pair. Also, unmatched quotes will be treated as part of a field.

See Also

RemoveQuotes

Example

```
fsParse1.MatchQuotes = True
```

ParseMode Property

Specifies the type of parsing.

Syntax

object.**ParseMode** [= *fsParseMode*]

The ParseMode property syntax has these parts:

Part	**Description**
object	An object expression that evaluates to fsParse object.
fsParseMode	(Optional) Which parsing mode to perform.

Settings

```
Public Enum fsParseMode
    fsPMText = 0
    fsPMFomattedText = 1
    fsPMprn = 1
    fsPMTabDelimitedText = 2
    fsPMtxt = 2
    fsPMCommaSeparatedValue = 3
    fsPMcsv = 3
    fsPMUnknown = 32767
End Enum
```

Remarks

0 - fsPMText	standard text
1 - fsPMFomattedText fsPMprn	space delimited text
2 - fsPMTabDelimitedText fsPMtxt	tab delimited text
3 - fsPMCommaSeperatedValue fsPMcsv	comma separated value
32767 - fsPMUnknown	generic (default)

If the file type is not one of the above types, then use fsPMUnknown.

Example

```
fsParse1.ParseMode = fsPMcsv        ' Specifies comma separated value
```

Record Property

Sets or returns the value of the current record.

Syntax

object.**Record** [= *string*]

The Record property syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object;
string	A new_record to be parsed. The default is a zero length string.

Remarks

This is the default property for fsParse.   When set to a value, fsParse will reparse the data and update FieldCount, Field and FieldPos methods.

Example

fsParse1.Record = "This is a good string to parse."

fsParse1 = "This example works the same as the above example."

RemoveQuotes Property

Sets or returns rather leading and trailing qoutes should be removed.

Syntax

object.**RemoveQuotes** [= *Boolean*]

The Record property syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object;
Boolean	(Optional) Tells fsParse to remove quotes when parsing fields when MatchQuotes is True. Default is True.

Remarks

This property determines if fsParse should remove leading and trailing quotes from a field. When MatchQuotes is set to False, **RemoveQuotes** property is ignored.

See Also

MatchQuotes

Example

fsParse1.RemoveQuotes = False

SearchText Property

Sets or returns a string to be search for in a file.

Syntax

object.**SearchText** [= *string*]

The SearchText property syntax has these parts:

Part	**Description**
object	An object expression that evaluates to fsParse object;
string	The text string to search for in a file. The default is a zero length string.

Remarks

Example

fsParse1.SearchText = "fish"

Methods

Field Method

Returns a varaint value of a field within the current record.

Syntax

object.**Field**

The Field method syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object.

Remarks

The **Field** method numbering is from 0 to FieldCount. When MatchQuotes and RemoveQuotes properties are set to True, the **Field** method will remove the qoutes that my be around the data.

See Also

FieldPos

Example

```
For i = 1 To fsParse1.FieldCount
        Debug.Print fsParse1.Field(i-1)
Next
```

FieldCount Method

Returns a long integer representing the number of fields parsed for a record.

Syntax

object.**FieldCount**

The FieldCount method syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object.

Remarks

Use **FieldCount** method when the number of fields is unkown for a given record.

Example

```
For i = 1 To fsParse1.FieldCount
        Debug.Print fsParse1.Field(i-1)
Next
```

FieldPos Method

Returns a long integer representing the start position of a field within a record or file.

Syntax

object.**FieldPos**

The FieldPos method syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object.

Remarks

The **FieldPos** method numbering is from 0 to FieldCount.   Use this method to get the field position within a record.   When searching for text in a file, the **FieldPos** method will return the field position within the file for each occurrence found.

Example

```
For i = 1 To fsParse1.FieldCount
        Debug.Print fsParse1.FieldPos(i-1)
Next
```

PercentDone Method

Returns the percentage of how much of the file has been parsed.

Syntax

object.**PercentDone**

The PercentDone method syntax has these parts:

Part	**Description**
object	An object expression that evaluates to fsParse object.

Remarks

Use this method to update a progress bar while parsing or searching takes place.
This method will return a value in the range of 0 to 100. Zero indicating the file has not been parsed and 100 indicating parsing was completed.

Example

```
ProgressBar1.Value = fsParse1.PercentDone
```

RecordCount Method

Returns the current record count when parsing a file.

Syntax

object.**RecordCount**

The RecordCount method syntax has these parts:

Part	**Description**
object	An object expression that evaluates to fsParse object.

Remarks

Use this method to get the number of records in a file.

Example

```
Debug.Print fsParse1.RecordCount
```

Value Method

Converts a formatted string to a numeric value in the form of a variant.

Syntax

object.**Value** (*formattedstring*)

The Value method syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object.
formattedstring	An accounting formatted string to be converted to a number.

Remarks

When parsing a file that contains accounting formatted data, this method will remove the special formatting and return the numeric representation of the field value. For example, if the passed in string contains "($1,234,567.89)", then this method will return -1234567.89.

Example

```
Debug.Print fsParse1.Value("456.89-")    '   displays -456.89

v$ = "$1,.34w0e+03"
Debug.Print fsParse1.Value(v)       ' displays 1.34
Debug.Print Val(v)                        ' displays 0
```

Version Method

Returns the version number for fsParse.

Syntax

object.**Version** () As String

The Version method syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object;

Remarks

The Version method will return a the saved version information in the form of major.minor.revision.

Example

```
MsgBox fsParse1.Version    ' Will display "1.00.0000" for the 1.00 release
```

Events

Begin
Change
Done
Error

Begin Event

The Begin event gets raised when parsing begins.

Syntax

Private Sub object_**Begin** ()

The Begin event syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object.

Remarks

This event gets fired when the Action property is set to fsAMBegin.

Change Event

The Change event gets raised whenever a new value gets found.

Syntax

Private Sub object_**Change** ()

The Change event syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object.

Remarks

Use this event to get the return fields from the parsed file.

Done Event

The Done event gets raised when parsing ends.

Syntax

Private Sub object_**Done** ()

The Done event syntax has these parts:

Part	**Description**
object	An object expression that evaluates to fsParse object.

Error Event

The Error event gets raised when an error occurs.

Syntax

Private Sub object_**Error** (ByVal nError As Long, ByVal Description As String, bCancel As Boolean)

The Error event syntax has these parts:

Part	Description
object	An object expression that evaluates to fsParse object.
nError	Visual Basic Runtime error number;
Description	A string value representing the generated error.
bCancel	Determines if an error message dialog displays. When set to False, the default, fsParse will display an error message. When set to True, no message will be displayed by fsParse.

Remarks

This event is useful if you want to centralize your error handling or prevent fsParse from displaying an error (set bCancel = True).