

Contents

Contents.....	3
FlashIMServer Overview	7
FlashIMServer Files	9
FlashIMServer Project Files by Type	10
FlashIMServer Form Files	11
FlashIMServer Module Files.....	12
FlashIMServer File Summaries.....	14
frmServer.....	15
basArc4.....	17
basCRC32.....	18
basFlashIM.....	19
basRegistry.....	20
basTools	21
FlashIMServer Procedure Summaries.....	22
frmServer.Form_Load Summary	23
frmServer.wSock_DataArrival Summary	24
frmServer.Forward_IM Summary.....	25
frmServer.Notify_StatusChange Summary.....	26
frmServer.Notify_NameChange Summary.....	27
frmServer.Notify_BuzzUser Summary.....	28
frmServer.Send_ErrorCode Summary	29
frmServer.Send_ServerKey Summary.....	30
frmServer.Send_ContactName Summary	31
frmServer.Send_DisplayName Summary	32
frmServer.Create_UserAccount Summary.....	33
frmServer.LogText Summary	34
basArc4.Arc4Init Summary	35
basArc4.Arc4 Summary	36
basArc4.encode Summary.....	37
basCRC32.AddBytes Summary	38
basCRC32.CalculateBytes Summary.....	39
basCRC32.CalculateCRC32String Summary	40
basCRC32.Clear Summary	41
basCRC32.InitializeCRC32 Summary.....	42
basFlashIM.ValidatePass Summary.....	43
basFlashIM.ValidateUser Summary	44

basFlashIM.GetPart Summary	45
basRegistry.regDoes_Key_Exist Summary	46
basRegistry.regQuery_A_Key Summary.....	47
basRegistry.regCreate_Key_Value Summary.....	48
basRegistry.regCreate_A_Key Summary	49
basRegistry.regEnumerate Summary	50
basRegistry.regDelete_Key Summary	51
basTools.Hex2Dec Summary.....	52
basTools.Dec2Hex Summary.....	53
FlashIMServer Procedure Crossreference.....	54
frmServer Crossreference.....	55
basArc4 Crossreference.....	57
basCRC32 Crossreference.....	58
basFlashIM Crossreference	59
basRegistry Crossreference.....	60
basTools Crossreference	61
Public Identifiers by Type.....	62
FlashIMServer Visible Interface	67
frmServer Visible Interface.....	68
FlashIMServer General Declarations	74
frmServer Declarations	75
basArc4 Declarations	76
basArc4 Declarations	76
basCRC32 Declarations	77
basCRC32 Declarations	77
basFlashIM Declarations.....	78
basFlashIM Declarations.....	78
basRegistry Declarations.....	80
basRegistry Declarations.....	80
basTools Declarations	82
basTools Declarations	82
FlashIMServer Source Code	83
FlashIMServer Source Code	83
frmServer.Form_Load Source Code	84
frmServer.wSock_DataArrival Source Code	85
frmServer.wSock_DataArrival Source Code	85
frmServer.Forward_IM Source Code.....	98
frmServer.Forward_IM Source Code.....	98

frmServer.Notify_StatusChange Source Code.....	100
frmServer.Notify_StatusChange Source Code.....	100
frmServer.Notify_NameChange Source Code.....	102
frmServer.Notify_NameChange Source Code.....	102
frmServer.Notify_BuzzUser Source Code.....	104
frmServer.Notify_BuzzUser Source Code.....	104
frmServer.Send_ErrorCode Source Code.....	106
frmServer.Send_ErrorCode Source Code.....	106
frmServer.Send_ServerKey Source Code.....	107
frmServer.Send_ServerKey Source Code.....	107
frmServer.Send_ContactName Source Code.....	108
frmServer.Send_ContactName Source Code.....	108
frmServer.Send_DisplayName Source Code.....	110
frmServer.Send_DisplayName Source Code.....	110
frmServer.Create_UserAccount Source Code.....	111
frmServer.Create_UserAccount Source Code.....	111
frmServer.LogText Source Code.....	112
frmServer.LogText Source Code.....	112
basArc4.Arc4Init Source Code.....	113
basArc4.Arc4Init Source Code.....	113
basArc4.Arc4 Source Code.....	114
basArc4.Arc4 Source Code.....	114
basArc4.encode Source Code.....	115
basArc4.encode Source Code.....	115
basCRC32.AddBytes Source Code.....	116
basCRC32.AddBytes Source Code.....	116
basCRC32.CalculateBytes Source Code.....	117
basCRC32.CalculateBytes Source Code.....	117
basCRC32.CalculateCRC32String Source Code.....	118
basCRC32.CalculateCRC32String Source Code.....	118
basCRC32.Clear Source Code.....	119
basCRC32.Clear Source Code.....	119
basCRC32.InitializeCRC32 Source Code.....	120
basCRC32.InitializeCRC32 Source Code.....	120
basFlashIM.ValidatePass Source Code.....	127
basFlashIM.ValidatePass Source Code.....	127
basFlashIM.ValidateUser Source Code.....	128
basFlashIM.ValidateUser Source Code.....	128

basFlashIM.GetPart Source Code	129
basFlashIM.GetPart Source Code	129
basRegistry.regDoes_Key_Exist Source Code	130
basRegistry.regDoes_Key_Exist Source Code	130
basRegistry.regQuery_A_Key Source Code.....	131
basRegistry.regQuery_A_Key Source Code.....	131
basRegistry.regCreate_Key_Value Source Code.....	133
basRegistry.regCreate_Key_Value Source Code.....	133
basRegistry.regCreate_A_Key Source Code	134
basRegistry.regCreate_A_Key Source Code	134
basRegistry.regEnumerate Source Code	135
basRegistry.regEnumerate Source Code	135
basRegistry.regDelete_Key Source Code	137
basRegistry.regDelete_Key Source Code	137
basTools.Hex2Dec Source Code.....	138
basTools.Hex2Dec Source Code.....	138
basTools.Dec2Hex Source Code.....	139
basTools.Dec2Hex Source Code.....	139

FlashIMServer Overview

FlashIMServer is a Visual Basic Version 6 Exe project.

Project Name: FlashIMServer

App Title: Flash IM Server

App Name: server.exe

File Summary

FlashIMServer includes 6 files totaling 67,970 Bytes in 1,834 Lines

6 code files

1 form file

5 module files

2 objects

1 reference

Option Settings

Option Explicit: 6/6 Files

Option Compare: All files Compare Binary

Option Base: All files Base 0

Default Data Types: None set (defaults to variant)

Statistics

3 controls in 1 control containing file.

3 unique control names

44 module-level constants

9 module-level variables

1 user defined type

8 API/DLL declarations

31 functions and subprocedures

113 local variables

1 label

589 numeric literals

89 string literals

Automation Components

Microsoft Windows Common Controls 6.0 (SP6)	C:\WINDOWS\System32\MSCOMCTL.OCX
---	----------------------------------

Microsoft Winsock Control 6.0 (SP5)	C:\WINDOWS\System32\mswinsck.ocx
-------------------------------------	----------------------------------

Automation References

OLE Automation

Other Specs & Settings

IconForm: frmServer

MajorVer: 1

RevisionVer: 2

VersionCompanyName: Dindo Liboon, Alicia Permell, and Zohair Ahmad

VersionFileDescription: Flash IM Server

VersionLegalCopyright: © 2003

VersionProductName: Flash IM

Startup: frmServer

MaxNumberOfThreads: 1

Otherspecs: AutoRefresh=1

FlashIMServer Files

Name (ClassName)	Type	Size Date	Number of Subprocedures
frmServer.frm frmServer	Form	42,617 Bytes in 1,054 Lines 27-Apr-03 11:25 PM	12
basArc4.bas basArc4	Module	1,599 Bytes in 78 Lines 27-Apr-03 11:19 PM	3
basCRC32.bas basCRC32	Module	11,928 Bytes in 351 Lines 27-Apr-03 11:19 PM	5
basFlashIM.bas basFlashIM	Module	3,870 Bytes in 132 Lines 27-Apr-03 11:19 PM	3
basRegistry.bas basRegistry	Module	7,227 Bytes in 195 Lines 27-Apr-03 11:19 PM	6
basTools.bas basTools	Module	729 Bytes in 24 Lines 27-Apr-03 11:19 PM	2

FlashIMServer Project Files by Type

The following topics contain FlashIMServer project files by type

FlashIMServer Form Files

Name (ClassName)	Size Date	Comments
frmServer.frm frmServer	42,617 Bytes in 1,054 Lines 27-Apr-03 11:25 PM	----- ----- Title: Flash IM Server Version 1.0 Course: CS-490 Senior Project Instructor: Dr. Dwight V. House Developers: Dindo Liboon, Alicia Permell, and Zohair Ahmad ----- -----

FlashIMServer Module Files

Name (ClassName)	Size Date	Comments
basArc4.bas basArc4	1,599 Bytes in 78 Lines 27-Apr-03 11:19 PM	----- ARC 4 Stream Cipher Obtained from http://www.Planet-Source-Code.com/xq/ASP/txtCodeId.1736/IngWId.1/qx/vb/scripts/ShowCode.htm -----
basCRC32.bas basCRC32	11,928 Bytes in 351 Lines 27-Apr-03 11:19 PM	CRC-32 Checksum ----- A very fast solution to calculate the CRC-32 Checksum with the help of some pre-compiled assembler code (c) 2000, Fredrik Qvarfort
basFlashIM.bas basFlashIM	3,870 Bytes in 132 Lines 27-Apr-03 11:19 PM	----- ----- Title: Flash IM Server Version 1.0 Course: CS-490 Senior Project Instructor: Dr. Dwight V. House Developers: Dindo Liboon, Alicia Permell, and Zohair Ahmad ----- ----- User status

basRegistry.bas
basRegistry

7,227 Bytes in 195 Lines
27-Apr-03 11:19 PM

Title: Flash IM Server Version 1.0
Course: CS-490 Senior Project
Instructor: Dr. Dwight V. House
Developers: Dindo Liboon, Alicia
Permell, and Zohair Ahmad

basTools.bas
basTools

729 Bytes in 24 Lines
27-Apr-03 11:19 PM

Title: Flash IM Server Version 1.0
Course: CS-490 Senior Project
Instructor: Dr. Dwight V. House
Developers: Dindo Liboon, Alicia
Permell, and Zohair Ahmad

FlashIMServer File Summaries

The following topics present information about the individual files in FlashIMServer .

frmServer

Summary

File Type: Form

File Name: frmServer.frm

Comments: ----- Title: Flash IM
Server Version 1.0 Course: CS-490 Senior Project Instructor: Dr. Dwight V.
House Developers: Dindo Liboon, Alicia Permell, and Zohair Ahmad -----

Statistics

42,617 Bytes in 1,054 Lines

3 Controls

3 Unique Control Names

12 Subprocedures

Procedures

Name (Scope/Type)	Size	Comments
Form_Load Private Event	1,199 Bytes in 42 Lines	
wSock_DataArrival Private Event	22,076 Bytes in 488 Lines	
Forward_IM Public MethodSub	2,277 Bytes in 66 Lines	
Notify_StatusChange Public MethodSub	2,430 Bytes in 66 Lines	
Notify_NameChange Public MethodSub	2,602 Bytes in 73 Lines	
Notify_BuzzUser Public MethodSub	1,707 Bytes in 59 Lines	
Send_ErrorCode Public MethodSub	140 Bytes in 4 Lines	Send msg back to user
Send_ServerKey Public MethodSub	194 Bytes in 4 Lines	Send msg back to user
Send_ContactName Public MethodSub	3,042 Bytes in 66 Lines	
Send_DisplayName Public MethodSub	203 Bytes in 4 Lines	Send msg back to user

Create_UserAccount Public MethodSub	1,229 Bytes in 30 Lines	
LogText Public MethodSub	579 Bytes in 14 Lines	

basArc4

Summary

File Type: Module

File Name: basArc4.bas

Comments: ----- ARC 4 Stream Cipher Obtained from
[http://www.Planet-Source-Code.com/xq/ASP/txtCodeId.1736/IngWId.1/
qx/vb/scripts/ShowCode.htm](http://www.Planet-Source-Code.com/xq/ASP/txtCodeId.1736/IngWId.1/qx/vb/scripts/ShowCode.htm) -----

Statistics

1,599 Bytes in 78 Lines

2 Variables

3 Subprocedures

Procedures

Name (Scope/Type)	Size	Comments
Arc4Init Private Sub	547 Bytes in 39 Lines	
Arc4 Private Function	560 Bytes in 25 Lines	
encode Public Function	122 Bytes in 4 Lines	

basCRC32

Summary

File Type: Module
File Name: basCRC32.bas
Comments: CRC-32 Checksum ----- A very fast solution to calculate the CRC-32 Checksum with the help of some pre-compiled assembler code (c) 2000, Fredrik Qvarfort

Statistics

11,928 Bytes in 351 Lines
4 Variables
1 Declaration
5 Subprocedures

Procedures

Name (Scope/Type)	Size	Comments
AddBytes Private Function	854 Bytes in 34 Lines	
CalculateBytes Private Function	261 Bytes in 8 Lines	Reset the current CRC calculation
CalculateCRC32String Public Function	422 Bytes in 13 Lines	Initialize the CRC32 checksum
Clear Private Sub	207 Bytes in 7 Lines	Here can be sloppy and reset both crc variables (this procedure will be more advanced when adding more checksums algorithms..)
InitializeCRC32 Private Sub	9,584 Bytes in 272 Lines	

basFlashIM

Summary

File Type: Module

File Name: basFlashIM.bas

Comments: ----- Title: Flash IM
Server Version 1.0 Course: CS-490 Senior Project Instructor: Dr. Dwight V.
House Developers: Dindo Liboon, Alicia Permell, and Zohair Ahmad -----
----- User status

Statistics

3,870 Bytes in 132 Lines

40 Constants

2 Variables

3 Subprocedures

Procedures

Name (Scope/Type)	Size	Comments
ValidatePass Public Function	239 Bytes in 10 Lines	Make sure password is atleast 4 characters
ValidateUser Public Function	1,077 Bytes in 36 Lines	
GetPart Public Function	586 Bytes in 25 Lines	

basRegistry

Summary

File Type: Module

File Name: basRegistry.bas

Comments: ----- Title: Flash IM
Server Version 1.0 Course: CS-490 Senior Project Instructor: Dr. Dwight V.
House Developers: Dindo Liboon, Alicia Permell, and Zohair Ahmad -----

Statistics

7,227 Bytes in 195 Lines

4 Constants

1 Variable

1 Type

7 Declarations

6 Subprocedures

Procedures

Name (Scope/Type)	Size	Comments
regDoes_Key_Exist Public Function	459 Bytes in 21 Lines	
regQuery_A_Key Public Function	1,805 Bytes in 54 Lines	
regCreate_Key_Value Public Sub	944 Bytes in 25 Lines	
regCreate_A_Key Public Function	290 Bytes in 7 Lines	
regEnumerate Public Function	1,426 Bytes in 47 Lines	
regDelete_Key Public Function	336 Bytes in 8 Lines	

basTools

Summary

File Type: Module

File Name: basTools.bas

Comments: ----- Title: Flash IM
Server Version 1.0 Course: CS-490 Senior Project Instructor: Dr. Dwight V.
House Developers: Dindo Liboon, Alicia Permell, and Zohair Ahmad -----

Statistics

729 Bytes in 24 Lines

2 Subprocedures

Procedures

Name (Scope/Type)	Size	Comments
Hex2Dec Public Function	103 Bytes in 9 Lines	
Dec2Hex Public Function	240 Bytes in 10 Lines	

FlashIMServer Procedure Summaries

The following topics contain summaries of all procedures in FlashIMServer .

frmServer.Form_Load Summary

Name: Form_Load
File: frmServer
Scope/Type: Private Event
Declaration: Form_Load()

Statistics

1,199 Bytes in 42 Lines
4 Local Variables
10 Numeric literals
6 String literals

frmServer.wSock_DataArrival Summary

Name: wSock_DataArrival

File: frmServer

Scope/Type: Private Event

Declaration: wSock_DataArrival(ByVal bytesTotal As Long)

Statistics

22,076 Bytes in 488 Lines

22 Local Variables

1 Argument

10 Numeric literals

45 String literals

frmServer.Forward_IM Summary

Name: Forward_IM

File: frmServer

Scope/Type: Public MethodSub

Declaration: Forward_IM(ByVal sName As String, ByVal sDest As String, ByVal sMsg As String)

Statistics

2,277 Bytes in 66 Lines

9 Local Variables

3 Arguments

7 Numeric literals

4 String literals

frmServer.Notify_StatusChange

Summary

Name: Notify_StatusChange

File: frmServer

Scope/Type: Public MethodSub

Declaration: Notify_StatusChange(ByVal sName As String, ByVal sContact As String,
ByVal sStat As Long)

Statistics

2,430 Bytes in 66 Lines

8 Local Variables

3 Arguments

5 Numeric literals

3 String literals

frmServer.Notify_NameChange Summary

Name: Notify_NameChange

File: frmServer

Scope/Type: Public MethodSub

Declaration: Notify_NameChange(ByVal sName As String, ByVal sContact As String,
ByVal sDisplay As String)

Statistics

2,602 Bytes in 73 Lines

8 Local Variables

3 Arguments

6 Numeric literals

3 String literals

frmServer.Notify_BuzzUser Summary

Name: Notify_BuzzUser

File: frmServer

Scope/Type: Public MethodSub

Declaration: Notify_BuzzUser(ByVal sName As String, ByVal sContact As String)

Statistics

1,707 Bytes in 59 Lines

8 Local Variables

2 Arguments

6 Numeric literals

3 String literals

frmServer.Send_ErrorCode Summary

Name: Send_ErrorCode
File: frmServer
Scope/Type: Public MethodSub
Comments: Send msg back to user
Declaration: Send_ErrorCode(ByVal code As Long)

Statistics

140 Bytes in 4 Lines

1 Argument

frmServer.Send_ServerKey Summary

Name: Send_ServerKey
File: frmServer
Scope/Type: Public MethodSub
Comments: Send msg back to user
Declaration: Send_ServerKey(ByVal sKey As String)

Statistics

194 Bytes in 4 Lines

1 Argument

frmServer.Send_ContactName Summary

Name: Send_ContactName

File: frmServer

Scope/Type: Public MethodSub

Declaration: Send_ContactName(ByVal sMyName As String, ByVal sName As String)

Statistics

3,042 Bytes in 66 Lines

4 Local Variables

2 Arguments

4 Numeric literals

3 String literals

frmServer.Send_DisplayName Summary

Name: Send_DisplayName

File: frmServer

Scope/Type: Public MethodSub

Comments: Send msg back to user

Declaration: Send_DisplayName(ByVal sName As String)

Statistics

203 Bytes in 4 Lines

1 Argument

frmServer.Create_UserAccount Summary

Name: Create_UserAccount

File: frmServer

Scope/Type: Public MethodSub

Declaration: Create_UserAccount(ByVal sName As String, ByVal sPw As String, ByVal sDisp As String, ByVal sDate As String)

Statistics

1,229 Bytes in 30 Lines

1 Local Variable

4 Arguments

1 Numeric literal

6 String literals

frmServer.LogText Summary

Name: LogText

File: frmServer

Scope/Type: Public MethodSub

Declaration: LogText(ByVal sTitle As String, ByVal sData As String)

Statistics

579 Bytes in 14 Lines

1 Local Variable

2 Arguments

1 Numeric literal

2 String literals

basArc4.Arc4Init Summary

Name: Arc4Init

File: basArc4

Scope/Type: Private Sub

Declaration: Arc4Init(ByVal Pwd As String)

Statistics

547 Bytes in 39 Lines

3 Local Variables

1 Argument

4 Numeric literals

basArc4.Arc4 Summary

Name: Arc4

File: basArc4

Scope/Type: Private Function

Declaration: Arc4(ByVal plaintext As String) As String

Statistics

560 Bytes in 25 Lines

7 Local Variables

1 Argument

2 Numeric literals

basArc4.encode Summary

Name: encode

File: basArc4

Scope/Type: Public Function

Declaration: encode(sz As String, pw As String) As String

Statistics

122 Bytes in 4 Lines

2 Arguments

basCRC32.AddBytes Summary

Name: AddBytes

File: basCRC32

Scope/Type: Private Function

Declaration: AddBytes(ByteArray() As Byte) As Variant

Statistics

854 Bytes in 34 Lines

1 Local Variable

1 Label

1 Argument

2 Numeric literals

basCRC32.CalculateBytes Summary

Name: CalculateBytes

File: basCRC32

Scope/Type: Private Function

Comments: Reset the current CRC calculation

Declaration: CalculateBytes(ByteArray() As Byte) As Variant

Statistics

261 Bytes in 8 Lines

1 Argument

basCRC32.CalculateCRC32String Summary

Name: CalculateCRC32String

File: basCRC32

Scope/Type: Public Function

Comments: Initialize the CRC32 checksum

Declaration: CalculateCRC32String(ByVal Text As String)

Statistics

422 Bytes in 13 Lines

1 Argument

1 Numeric literal

basCRC32.Clear Summary

Name: Clear

File: basCRC32

Scope/Type: Private Sub

Comments: Here can be sloppy and reset both crc variables (this procedure will be more advanced when adding more checksums algorithms..)

Declaration: Clear()

Statistics

207 Bytes in 7 Lines

1 Numeric literal

basCRC32.InitializeCRC32 Summary

Name: InitializeCRC32

File: basCRC32

Scope/Type: Private Sub

Declaration: InitializeCRC32()

Statistics

9,584 Bytes in 272 Lines

2 Local Variables

512 Numeric literals

2 String literals

basFlashIM.ValidatePass Summary

Name: ValidatePass

File: basFlashIM

Scope/Type: Public Function

Comments: Make sure password is atleast 4 characters

Declaration: ValidatePass(ByVal pass As String) As Boolean

Statistics

239 Bytes in 10 Lines

1 Argument

1 Numeric literal

basFlashIM.ValidateUser Summary

Name: ValidateUser

File: basFlashIM

Scope/Type: Public Function

Declaration: ValidateUser(ByVal User As String) As Boolean

Statistics

1,077 Bytes in 36 Lines

5 Local Variables

1 Argument

3 Numeric literals

10 String literals

basFlashIM.GetPart Summary

Name: GetPart

File: basFlashIM

Scope/Type: Public Function

Declaration: GetPart(ByVal sMsg As String, ByVal iPart As Integer) As String

Statistics

586 Bytes in 25 Lines

5 Local Variables

2 Arguments

3 Numeric literals

basRegistry.regDoes_Key_Exist

Summary

Name: regDoes_Key_Exist

File: basRegistry

Scope/Type: Public Function

Declaration: regDoes_Key_Exist(ByVal lngRootKey As Long, ByVal strRegKeyPath As String) As Boolean

Statistics

459 Bytes in 21 Lines

1 Local Variable

2 Arguments

1 Numeric literal

basRegistry.regQuery_A_Key Summary

Name: regQuery_A_Key

File: basRegistry

Scope/Type: Public Function

Declaration: regQuery_A_Key(ByVal lngRootKey As Long, ByVal strRegKeyPath As String, ByVal strRegSubKey As String) As Variant

Statistics

1,805 Bytes in 54 Lines

6 Local Variables

3 Arguments

3 Numeric literals

basRegistry.regCreate_Key_Value

Summary

Name: regCreate_Key_Value

File: basRegistry

Scope/Type: Public Sub

Declaration: regCreate_Key_Value(ByVal lngRootKey As Long, ByVal strRegKeyPath As String, ByVal strRegSubKey As String, varRegData As Variant)

Statistics

944 Bytes in 25 Lines

4 Local Variables

4 Arguments

2 Numeric literals

basRegistry.regCreate_A_Key Summary

Name: regCreate_A_Key

File: basRegistry

Scope/Type: Public Function

Declaration: regCreate_A_Key(ByVal lngRootKey As Long, ByVal strRegKeyPath As String)

Statistics

290 Bytes in 7 Lines

1 Local Variable

2 Arguments

basRegistry.regEnumerate Summary

Name: regEnumerate

File: basRegistry

Scope/Type: Public Function

Declaration: regEnumerate(ByVal lngRootKey As Long, ByVal strRegKeyPath As String,
ByRef rvntKeys As Variant) As Long

Statistics

1,426 Bytes in 47 Lines

11 Local Variables

3 Arguments

3 Numeric literals

basRegistry.regDelete_Key Summary

Name: regDelete_Key

File: basRegistry

Scope/Type: Public Function

Declaration: regDelete_Key(ByVal Group As Long, ByVal Section As String, ByVal Key
As String) As String

Statistics

336 Bytes in 8 Lines

1 Local Variable

3 Arguments

basTools.Hex2Dec Summary

Name: Hex2Dec

File: basTools

Scope/Type: Public Function

Declaration: Hex2Dec(ByVal szHex As String) As Long

Statistics

103 Bytes in 9 Lines

1 Argument

1 String literal

basTools.Dec2Hex Summary

Name: Dec2Hex

File: basTools

Scope/Type: Public Function

Declaration: Dec2Hex(ByVal szDec As Integer) As String

Statistics

240 Bytes in 10 Lines

1 Local Variable

1 Argument

1 Numeric literal

1 String literal

FlashIMServer Procedure Crossreference

The following topics present cross reference tables for all procedures, organized by file.

frmServer Crossreference

Procedure	Refers To	Referenced By
Form_Load	basRegistry.regEnumerate basRegistry.regQuery_A_Key basArc4.encode	
wSock_DataArrival	basTools.Hex2Dec basFlashIM.GetPart basArc4.encode LogText basFlashIM.ValidateUser Send_ErrorCode basFlashIM.ValidatePass Create_UserAccount basCRC32.CalculateCRC32String Send_ServerKey Send_DisplayName basRegistry.regEnumerate Send_ContactName Notify_StatusChange Notify_NameChange basRegistry.regQuery_A_Key basRegistry.regDoes_Key_Exist basRegistry.regCreate_A_Key basRegistry.regCreate_Key_Value Notify_BuzzUser basRegistry.regDelete_Key Forward_IM	
Forward_IM	basRegistry.regQuery_A_Key basArc4.encode basTools.Dec2Hex	wSock_DataArrival
Notify_StatusChange	basArc4.encode basRegistry.regQuery_A_Key basTools.Dec2Hex	wSock_DataArrival
Notify_NameChange	basRegistry.regQuery_A_Key basArc4.encode basTools.Dec2Hex	wSock_DataArrival
Notify_BuzzUser	basRegistry.regQuery_A_Key basArc4.encode basTools.Dec2Hex	wSock_DataArrival
Send_ErrorCode	basTools.Dec2Hex	wSock_DataArrival

Send_ServerKey	basTools.Dec2Hex	wSock_DataArrival
Send_ContactName	basArc4.encode basRegistry.regQuery_A_Key basTools.Dec2Hex	wSock_DataArrival
Send_DisplayName	basTools.Dec2Hex	wSock_DataArrival
Create_UserAccount	basRegistry.regDoes_Key_Exist basRegistry.regCreate_A_Key basRegistry.regCreate_Key_Value	wSock_DataArrival
LogText		wSock_DataArrival

basArc4 Crossreference

Procedure	Refers To	Referenced By
Arc4Init		encode
Arc4		encode
encode	Arc4Init Arc4	frmServer.Form_Load frmServer.wSock_DataArrival frmServer.Forward_IM frmServer.Notify_StatusChange frmServer.Notify_NameChange frmServer.Notify_BuzzUser frmServer.Send_ContactName

basCRC32 Crossreference

Procedure	Refers To	Referenced By
AddBytes		CalculateBytes
CalculateBytes	Clear AddBytes	CalculateCRC32String
CalculateCRC32String	InitializeCRC32 CalculateBytes	frmServer.wSock_DataArrival
Clear		CalculateBytes
InitializeCRC32		CalculateCRC32String

basFlashIM Crossreference

Procedure	Refers To	Referenced By
ValidatePass		frmServer.wSock_DataArrival
ValidateUser		frmServer.wSock_DataArrival
GetPart	basTools.Hex2Dec	frmServer.wSock_DataArrival

basRegistry Crossreference

Procedure	Refers To	Referenced By
regDoes_Key_Exist		frmServer.wSock_DataArrival frmServer.Create_UserAccount
regQuery_A_Key		frmServer.Form_Load frmServer.wSock_DataArrival frmServer.Forward_IM frmServer.Notify_StatusChange frmServer.Notify_NameChange frmServer.Notify_BuzzUser frmServer.Send_ContactName
regCreate_Key_Value		frmServer.wSock_DataArrival frmServer.Create_UserAccount
regCreate_A_Key		frmServer.wSock_DataArrival frmServer.Create_UserAccount
regEnumerate		frmServer.Form_Load frmServer.wSock_DataArrival
regDelete_Key		frmServer.wSock_DataArrival

basTools Crossreference

Procedure	Refers To	Referenced By
Hex2Dec		frmServer.wSock_DataArrival basFlashIM.GetPart
Dec2Hex		frmServer.Forward_IM frmServer.Notify_StatusChange frmServer.Notify_NameChange frmServer.Notify_BuzzUser frmServer.Send_ErrorCode frmServer.Send_ServerKey frmServer.Send_ContactName frmServer.Send_DisplayName

Public Identifiers by Type

41 Public Constants

Name	Defined In	Referenced by
STATUS_ONLINE	basFlashIM	frmServer.wSock_DataArrival frmServer.Send_ContactName
STATUS_OFFLINE	basFlashIM	frmServer.wSock_DataArrival frmServer.Notify_StatusChange frmServer.Notify_NameChange frmServer.Notify_BuzzUser frmServer.Send_ContactName
STATUS_BLOCKED	basFlashIM	frmServer.wSock_DataArrival frmServer.Notify_StatusChange frmServer.Send_ContactName
CREATE_ACCOUNT	basFlashIM	frmServer.wSock_DataArrival
LOGIN	basFlashIM	frmServer.wSock_DataArrival
LOGOUT	basFlashIM	frmServer.wSock_DataArrival
CHANGE_STATUS	basFlashIM	frmServer.wSock_DataArrival
SEND_CLIENT_KEY	basFlashIM	frmServer.wSock_DataArrival
LIST_ADD_CLIENT	basFlashIM	frmServer.wSock_DataArrival
CHANGE_DISPLAY_NAME	basFlashIM	frmServer.wSock_DataArrival
CONTACT_INSTANT_MSG	basFlashIM	frmServer.wSock_DataArrival
CONTACT_BLOCK	basFlashIM	frmServer.wSock_DataArrival
CONTACT_UNBLOCK	basFlashIM	frmServer.wSock_DataArrival
CONTACT_BUZZ	basFlashIM	frmServer.wSock_DataArrival
CONTACT_DELETE	basFlashIM	frmServer.wSock_DataArrival
CREATE_ACCOUNT_GOOD	basFlashIM	frmServer.wSock_DataArrival
CREATE_ACCOUNT_INVALIDNAME	basFlashIM	frmServer.wSock_DataArrival
CREATE_ACCOUNT_INVALIDDISPLAY	basFlashIM	frmServer.wSock_DataArrival
CREATE_ACCOUNT_INVALIDPW	basFlashIM	frmServer.wSock_DataArrival
CREATE_ACCOUNT_EXISTS	basFlashIM	frmServer.wSock_DataArrival
LOGIN_GOOD	basFlashIM	frmServer.wSock_DataArrival
LOGIN_NAME_INVALID	basFlashIM	frmServer.wSock_DataArrival

LOGIN_NAME_NOTEXIS T	basFlashIM	frmServer.wSock_DataArrival
LOGIN_PW_INVALID	basFlashIM	frmServer.wSock_DataArrival
LOGIN_ALREADY_LOGG ED_IN	basFlashIM	frmServer.wSock_DataArrival
CHANGE_STATUS_GOO D	basFlashIM	frmServer.wSock_DataArrival
CHANGE_STATUS_SAM E	basFlashIM	frmServer.wSock_DataArrival
LOGIN_SERVER_KEY	basFlashIM	frmServer.Send_ServerKey
LOGOUT_FAILED	basFlashIM	frmServer.wSock_DataArrival
LOGIN_SEND_CLIENT_K EY_FAILED	basFlashIM	frmServer.wSock_DataArrival
LIST_ADD_CLIENT_GOO D	basFlashIM	frmServer.wSock_DataArrival
LIST_ADD_CLIENT_EXIS TS	basFlashIM	frmServer.wSock_DataArrival
LIST_ADD_CLIENT_ERR ORSELF	basFlashIM	frmServer.wSock_DataArrival
CONTACT_ADD	basFlashIM	frmServer.Send_ContactName
LOGIN_DISPLAY_NAME	basFlashIM	frmServer.Send_DisplayName
STATUS_CHANGED	basFlashIM	frmServer.Notify_StatusChange
DISPLAY_CHANGED	basFlashIM	frmServer.Notify_NameChange
BUZZ_USER	basFlashIM	frmServer.Notify_BuzzUser
FORWARD_INSTANT_M SG	basFlashIM	frmServer.Forward_IM
LIST_ADD_CLIENT_NOT EXIST	basFlashIM	frmServer.wSock_DataArrival
HKEY_CURRENT_USER	basRegistry	frmServer.Form_Load frmServer.wSock_DataArrival frmServer.Forward_IM frmServer.Notify_StatusChange frmServer.Notify_NameChange frmServer.Notify_BuzzUser frmServer.Send_ContactName frmServer.Create_UserAccount

2 Public Variables

Name	Defined In	Referenced By
szUser	basFlashIM	frmServer.Create_UserAccount

flashKey	basFlashIM	frmServer.Form_Load frmServer.wSock_DataArrival frmServer.Forward_IM frmServer.Notify_StatusChange frmServer.Notify_NameChange frmServer.Notify_BuzzUser frmServer.Send_ContactName
----------	------------	---

23 Public Procedures

Name	Defined In	Type	Referenced By
Forward_IM	frmServer	MethodSub	wSock_DataArrival
Notify_StatusChange	frmServer	MethodSub	wSock_DataArrival
Notify_NameChange	frmServer	MethodSub	wSock_DataArrival
Notify_BuzzUser	frmServer	MethodSub	wSock_DataArrival
Send_ErrorCode	frmServer	MethodSub	wSock_DataArrival
Send_ServerKey	frmServer	MethodSub	wSock_DataArrival
Send_ContactName	frmServer	MethodSub	wSock_DataArrival
Send_DisplayName	frmServer	MethodSub	wSock_DataArrival
Create_UserAccount	frmServer	MethodSub	wSock_DataArrival
LogText	frmServer	MethodSub	wSock_DataArrival
encode	basArc4	Function	frmServer.Form_Load frmServer.wSock_DataArrival frmServer.Forward_IM frmServer.Notify_StatusChange frmServer.Notify_NameChange frmServer.Notify_BuzzUser frmServer.Send_ContactName
CalculateCRC32String	basCRC32	Function	frmServer.wSock_DataArrival
ValidatePass	basFlashIM	Function	frmServer.wSock_DataArrival
ValidateUser	basFlashIM	Function	frmServer.wSock_DataArrival
GetPart	basFlashIM	Function	frmServer.wSock_DataArrival

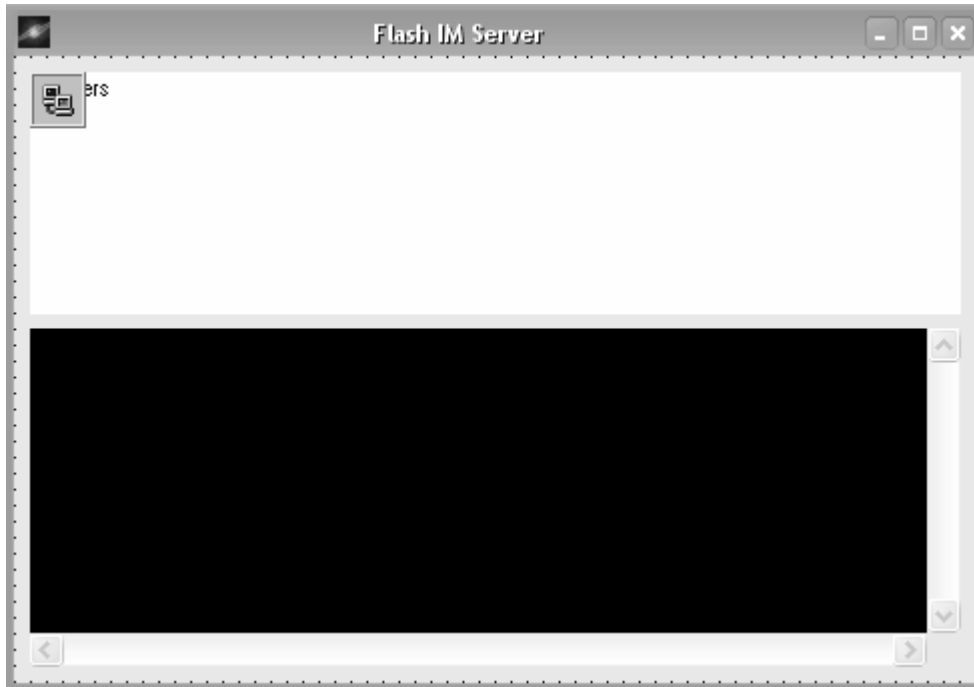
regDoes_Key_Exist	basRegistry	Function	frmServer.wSock_DataArrival frmServer.Create_UserAccount
regQuery_A_Key	basRegistry	Function	frmServer.Form_Load frmServer.wSock_DataArrival frmServer.Forward_IM frmServer.Notify_StatusChange frmServer.Notify_NameChange frmServer.Notify_BuzzUser frmServer.Send_ContactName
regCreate_Key_Value	basRegistry	Sub	frmServer.wSock_DataArrival frmServer.Create_UserAccount
regCreate_A_Key	basRegistry	Function	frmServer.wSock_DataArrival frmServer.Create_UserAccount
regEnumerate	basRegistry	Function	frmServer.Form_Load frmServer.wSock_DataArrival
regDelete_Key	basRegistry	Function	frmServer.wSock_DataArrival
Hex2Dec	basTools	Function	frmServer.wSock_DataArrival basFlashIM.GetPart
Dec2Hex	basTools	Function	frmServer.Forward_IM frmServer.Notify_StatusChange frmServer.Notify_NameChange frmServer.Notify_BuzzUser frmServer.Send_ErrorCode frmServer.Send_ServerKey frmServer.Send_ContactName frmServer.Send_DisplayName

FlashIMServer Visible Interface

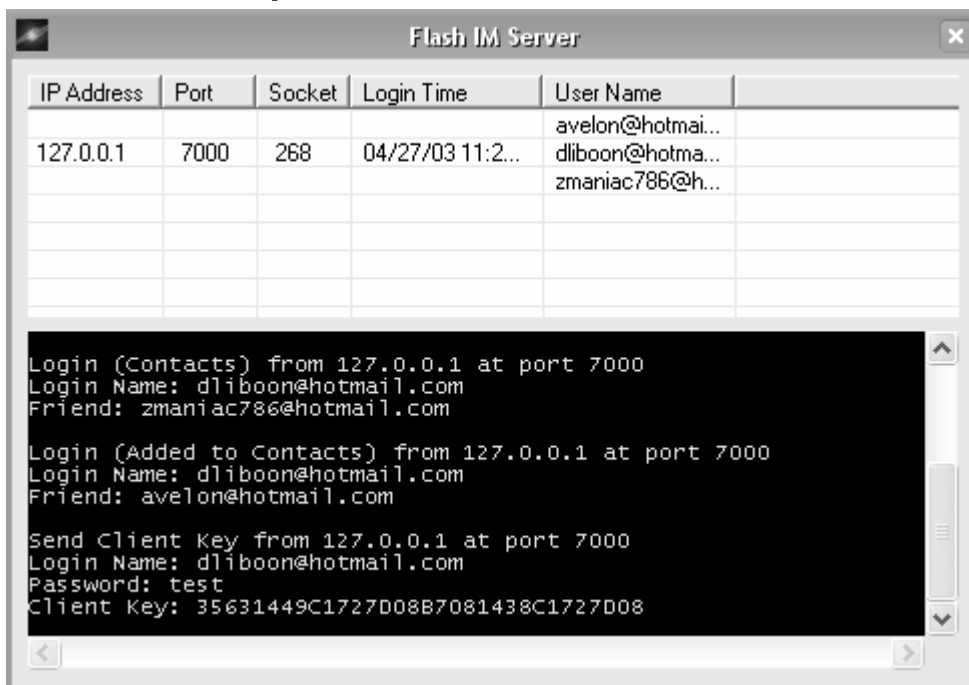
The following topics contain the project's visible interface

frmServer Visible Interface

Design Mode Bitmap



Runtime Bitmap



3 Controls

Class	Name	Instances, Events
-------	------	-------------------

MSWinsockLib.Winsock	wSock	1, 1
MSComctlLib.ListView	lstUsers	1, 0
VB.TextBox	txtTraffic	1, 0

Form Definition

VERSION 5.00

Object = "{831FDD16-0C5C-11D2-A9FC-0000F8754DA1}#2.0#0"; "MSCOMCTL.OCX"

Object = "{248DD890-BB45-11CF-9ABC-0080C7E7B78D}#1.0#0"; "mswinsck.ocx"

Begin VB.Form frmServer

```

BorderStyle      = 1  'Fixed Single
Caption          =   "Flash IM Server"
ClientHeight    =  4695
ClientLeft      =   45
ClientTop       =  435
ClientWidth     =  7230
Icon             =   "frmServer.frx":0000
LinkTopic       =   "frmServer"
MaxButton       =  0  'False
MinButton       =  0  'False
ScaleHeight     =  313
ScaleMode       =  3  'Pixel
ScaleWidth      =  482
StartupPosition =  3  'Windows Default

```

Begin MSWinsockLib.Winsock wSock

```

Left            =  120
Top             =  120
_ExtentX       =  741
_ExtentY       =  741
_Version       =  393216
Protocol        =   1
LocalPort      =  5000

```

End

Begin MSComctlLib.ListView lstUsers

```

Height          =  1815
Left            =  120
TabIndex       =   0
Top            =  120
Width          =  6975
_ExtentX       =  12303
_ExtentY       =  3201
View           =   3
LabelWrap      =  -1  'True
HideSelection  =  -1  'True
FullRowSelect  =  -1  'True
GridLines     =  -1  'True
_Version       =  393217
ForeColor      =  -2147483640
BackColor      =  -2147483643
Appearance     =   0
NumItems       =   10

```

BeginProperty ColumnHeader(1) {BDD1F052-858B-11D1-B16A-00C0F0283628}

```

Text           =   "IP Address"
Object.Width   =  1764

```

```

EndProperty
BeginProperty ColumnHeader(2) {BDD1F052-858B-11D1-B16A-00C0F0283628}
  SubItemIndex = 1
  Text = "Port"
  Object.Width = 1235
EndProperty
BeginProperty ColumnHeader(3) {BDD1F052-858B-11D1-B16A-00C0F0283628}
  SubItemIndex = 2
  Text = "Socket"
  Object.Width = 1235
EndProperty
BeginProperty ColumnHeader(4) {BDD1F052-858B-11D1-B16A-00C0F0283628}
  SubItemIndex = 3
  Text = "Login Time"
  Object.Width = 2540
EndProperty
BeginProperty ColumnHeader(5) {BDD1F052-858B-11D1-B16A-00C0F0283628}
  SubItemIndex = 4
  Text = "Alias"
  Object.Width = 0
EndProperty
BeginProperty ColumnHeader(6) {BDD1F052-858B-11D1-B16A-00C0F0283628}
  SubItemIndex = 5
  Text = "User Name"
  Object.Width = 2540
EndProperty
BeginProperty ColumnHeader(7) {BDD1F052-858B-11D1-B16A-00C0F0283628}
  SubItemIndex = 6
  Text = "Password"
  Object.Width = 0
EndProperty
BeginProperty ColumnHeader(8) {BDD1F052-858B-11D1-B16A-00C0F0283628}
  SubItemIndex = 7
  Text = "Day Created"
  Object.Width = 0
EndProperty
BeginProperty ColumnHeader(9) {BDD1F052-858B-11D1-B16A-00C0F0283628}
  SubItemIndex = 8
  Text = "Status"
  Object.Width = 0
EndProperty
BeginProperty ColumnHeader(10) {BDD1F052-858B-11D1-B16A-00C0F0283628}
  SubItemIndex = 9
  Text = "Dynamic Password"
  Object.Width = 0
EndProperty
End
Begin VB.TextBox txtTraffic
  Appearance = 0 'Flat
  BackColor = &H00000000&
  BorderStyle = 0 'None
  BeginProperty Font
    Name = "Lucida Console"
    Size = 8.25
    Charset = 0
    Weight = 400
    Underline = 0 'False

```

```
        Italic          = 0   'False
        Strikethrough   = 0   'False
    EndProperty
    ForeColor          = &H00E0E0E0&
    Height             = 2535
    Left               = 120
    Locked             = -1   'True
    MultiLine          = -1   'True
    ScrollBars         = 3    'Both
    TabIndex           = 1
    Top                = 2040
    Width              = 6975
End
End
Attribute VB_Name = "frmServer"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
```


FlashIMServer General Declarations

The following topics present the general declarations in each file.

frmServer Declarations

```
-----  
'      Title: Flash IM Server Version 1.0  
'      Course: CS-490 Senior Project  
' Instructor: Dr. Dwight V. House  
' Developers: Dindo Liboon, Alicia Permell, and Zohair Ahmad  
-----
```

Option Explicit

basArc4 Declarations

```
-----  
' ARC 4 Stream Cipher  
'  
' Obtained from  
' http://www.Planet-Source-Code.com  
'   /xq/ASP/txtCodeId.1736/lngWId.1/  
'   qx/vb/scripts/ShowCode.htm  
-----
```

Option Explicit

```
Private s(0 To 255) As Integer  
Private kep(0 To 255) As Integer
```

basCRC32 Declarations

```
'CRC-32 Checksum
```

```
'-----  
'
```

```
'A very fast solution to calculate the  
'CRC-32 Checksum with the help of some  
'pre-compiled assembler code
```

```
'
```

```
'(c) 2000, Fredrik Qvarfort
```

```
'
```

```
Option Explicit
```

```
Private m_CRC32 As Long
```

```
Private m_CRC32Asm() As Byte
```

```
Private m_CRC32Init As Boolean
```

```
Private m_CRC32Table(0 To 255) As Long
```

```
Private Declare Function CallWindowProc Lib "user32" Alias "CallWindowProcA" (ByVal lpPrevWndFunc As Long,  
ByVal hWnd As Long, ByVal msg As Long, ByVal wParam As Long, ByVal lParam As Long) As Long
```

basFlashIM Declarations

```
-----  
'      Title: Flash IM Server Version 1.0  
'      Course: CS-490 Senior Project  
' Instructor: Dr. Dwight V. House  
' Developers: Dindo Liboon, Alicia Permell, and Zohair Ahmad  
-----
```

Option Explicit

```
' User status  
Public Const STATUS_ONLINE = 1  
Public Const STATUS_OFFLINE = 7  
Public Const STATUS_BLOCKED = 8  
  
' To-Server messages  
Public Const CREATE_ACCOUNT = 1  
Public Const LOGIN = 2  
Public Const LOGOUT = 3  
Public Const CHANGE_STATUS = 4  
Public Const SEND_CLIENT_KEY = 5  
Public Const LIST_ADD_CLIENT = 6  
Public Const CHANGE_DISPLAY_NAME = 7  
Public Const CONTACT_INSTANT_MSG = 8  
Public Const CONTACT_BLOCK = 9  
Public Const CONTACT_UNBLOCK = 10  
Public Const CONTACT_BUZZ = 11  
Public Const CONTACT_DELETE = 12  
  
' From-Server messages  
Public Const CREATE_ACCOUNT_GOOD = 1  
Public Const CREATE_ACCOUNT_INVALIDNAME = 2  
Public Const CREATE_ACCOUNT_INVALIDDISPLAY = 3  
Public Const CREATE_ACCOUNT_INVALIDPW = 4  
Public Const CREATE_ACCOUNT_EXISTS = 5  
Public Const LOGIN_GOOD = 6  
Public Const LOGIN_NAME_INVALID = 7  
Public Const LOGIN_NAME_NOTEXIST = 8  
Public Const LOGIN_PW_INVALID = 9
```

```
Public Const LOGIN_ALREADY_LOGGED_IN = 10
Public Const CHANGE_STATUS_GOOD = 12
Public Const CHANGE_STATUS_SAME = 14
Public Const LOGIN_SERVER_KEY = 15
Public Const LOGOUT_FAILED = 16
Public Const LOGIN_SEND_CLIENT_KEY_FAILED = 17
Public Const LIST_ADD_CLIENT_GOOD = 18
Public Const LIST_ADD_CLIENT_EXISTS = 19
Public Const LIST_ADD_CLIENT_ERRORSELF = 20
Public Const CONTACT_ADD = 21
Public Const LOGIN_DISPLAY_NAME = 22
Public Const STATUS_CHANGED = 23
Public Const DISPLAY_CHANGED = 24
Public Const BUZZ_USER = 25
Public Const FORWARD_INSTANT_MSG = 26
Public Const LIST_ADD_CLIENT_NOTEXIST = 27
```

```
Global szUser As String
Global flashKey As String
```

basRegistry Declarations

```
-----  
'      Title: Flash IM Server Version 1.0  
'      Course: CS-490 Senior Project  
'      Instructor: Dr. Dwight V. House  
'      Developers: Dindo Liboon, Alicia Perrell, and Zohair Ahmad  
-----
```

Option Explicit

```
Public Const HKEY_CURRENT_USER As Long = &H80000001
```

```
Private Const REG_SZ As Long = 1  
Private Const REG_DWORD As Long = 4  
Private Const ERROR_SUCCESS As Long = 0
```

```
Private Type FILETIME  
    dwLowDateTime As Long  
    dwHighDateTime As Long  
End Type
```

```
Private Declare Function RegCloseKey Lib "advapi32.dll" (ByVal lngRootKey As Long) As Long  
Private Declare Function RegCreateKey Lib "advapi32.dll" Alias "RegCreateKeyA" (ByVal lngRootKey As Long,  
ByVal lpSubKey As String, phkResult As Long) As Long  
Private Declare Function RegOpenKey Lib "advapi32.dll" Alias "RegOpenKeyA" (ByVal lngRootKey As Long,  
ByVal lpSubKey As String, phkResult As Long) As Long  
Private Declare Function RegQueryValueEx Lib "advapi32.dll" Alias "RegQueryValueExA" (ByVal lngRootKey As  
Long, ByVal lpValueName As String, ByVal lpReserved As Long, lpType As Long, lpData As Any, lpcbData As  
Long) As Long  
Private Declare Function RegDeleteKey Lib "advapi32.dll" Alias "RegDeleteKeyA" (ByVal hKey As Long, ByVal  
lpSubKey As String) As Long  
Private Declare Function RegSetValueEx Lib "advapi32.dll" Alias "RegSetValueExA" (ByVal lngRootKey As  
Long, ByVal lpValueName As String, ByVal Reserved As Long, ByVal dwType As Long, lpData As Any, ByVal  
cbData As Long) As Long  
Private Declare Function RegEnumKeyEx Lib "advapi32.dll" Alias "RegEnumKeyExA" (ByVal hKey As Long, ByVal  
dwIndex As Long, ByVal lpName As String, lpcbName As Long, ByVal lpReserved As Long, ByVal lpClass As  
String, lpcbClass As Long, lpftLastWriteTime As FILETIME) As Long
```

```
Private m_lngRetVal As Long
```


basTools Declarations

```
-----  
'      Title: Flash IM Server Version 1.0  
'      Course: CS-490 Senior Project  
' Instructor: Dr. Dwight V. House  
' Developers: Dindo Liboon, Alicia Permell, and Zohair Ahmad  
'-----
```

Option Explicit

FlashIMServer Source Code

The following topics contain the code of all procedures

frmServer.Form_Load Source Code

```
Private Sub Form_Load()  
    Dim noUsers As Long  
    Dim idx As Long  
    Dim itm As ListItem  
    Dim szLogins() As String  
  
    On Error Resume Next  
  
    ' Set default decryption key  
    flashKey = "Flash IM Client"  
  
    ' Obtain all Flash IM users  
    noUsers = regEnumerate(HKEY_CURRENT_USER, "Software\Flash IM Server\Users", szLogins)  
  
    For idx = 0 To noUsers - 1  
        Set itm = lstUsers.ListItems.Add(, , "")  
        itm.SubItems(0) = ""  
        itm.SubItems(1) = ""  
        itm.SubItems(2) = ""  
        itm.SubItems(3) = ""  
        itm.SubItems(5) = Trim$(szLogins(idx))  
        itm.SubItems(4) = Trim$(regQuery_A_Key(HKEY_CURRENT_USER, _  
            "Software\Flash IM Server\Users\" & itm.SubItems(5), _  
            "DisplayName"))  
        itm.SubItems(6) = encode(regQuery_A_Key(HKEY_CURRENT_USER, _  
            "Software\Flash IM Server\Users\" & itm.SubItems(5), _  
            "Password"), flashKey)  
        itm.SubItems(7) = regQuery_A_Key(HKEY_CURRENT_USER, _  
            "Software\Flash IM Server\Users\" & itm.SubItems(5), _  
            "CreationDate")  
        itm.SubItems(8) = ""  
        itm.SubItems(9) = ""  
    Next  
  
    wSock.Bind  
End Sub
```

frmServer.wSock_DataArrival Source Code

```
Private Sub wSock_DataArrival(ByVal bytesTotal As Long)
    Dim szbuffer As String
    Dim szSn As String
    Dim szDis As String
    Dim szPw As String
    Dim lCmd As Long
    Dim lVer As Long
    Dim idx As ListItem
    Dim namFind As Long
    Dim i As Long
    Dim lstName As String
    Dim lstPw As String
    Dim ip As String
    Dim szKey As String
    Dim tmpIP As String
    Dim tmpPort As Long
    Dim lngRootKey As Long
    Dim noUsers As Long
    Dim szContacts() As String
    Dim j As Long
    Dim szMsg As String
    Dim szDes As String
    Dim lStat As Long

    On Error Resume Next

    ' Save settings into the registry
    lngRootKey = HKEY_CURRENT_USER

    ' Obtain data from client
    wSock.GetData szbuffer, vbString

    ' Determine command and real data
    lCmd = Val(Mid(szbuffer, 5, Hex2Dec(Left(szbuffer, 4))))

    ' Strip command
    szbuffer = Right(szbuffer, Len(szbuffer) - (4 + Hex2Dec(Left(szbuffer$, 4))))
```

```

Select Case lCmd
Case CREATE_ACCOUNT
    lVer = Val(GetPart(szbuffer, 1))           ' Version
    szSn = encode(GetPart(szbuffer, 2), flashKey) ' Login Name
    szDis = encode(GetPart(szbuffer, 3), flashKey) ' Display Name
    szPw = encode(GetPart(szbuffer, 4), flashKey) ' Password

    Call LogText("Create Account", "Version: " & Str(lVer) & vbCrLf & "Login Name: " & szSn & vbCrLf &
"Display Name: " & szDis & vbCrLf & "Password: " & szPw)

    ' Check is login name is valid
    If ValidateUser(szSn) = False Then
        Call Send_ErrorCode(CREATE_ACCOUNT_INVALIDNAME)
        Exit Sub
    End If

    ' Check if display name is valid
    If Len(Trim$(szDis)) < 1 Then
        Call Send_ErrorCode(CREATE_ACCOUNT_INVALIDDISPLAY)
        Exit Sub
    End If

    ' Check if password is valid
    If ValidatePass(szPw) = False Or szPw = szSn Then
        Call Send_ErrorCode(CREATE_ACCOUNT_INVALIDPW)
        Exit Sub
    End If

    ' Look for user
    namFind = 0
    For i = 1 To lstUsers.ListItems.Count
        lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)

        ' Check PW's if names match
        If LCase$(lstName) = LCase$(szSn) And lstUsers.ListItems.Item(i).ListSubItems.Item(7) = ""
Then
            lstUsers.ListItems.Item(i).ListSubItems.Item(1) = ""
            lstUsers.ListItems.Item(i).ListSubItems.Item(2) = ""
            lstUsers.ListItems.Item(i).ListSubItems.Item(3) = ""
            lstUsers.ListItems.Item(i).ListSubItems.Item(4) = szDis
            lstUsers.ListItems.Item(i).ListSubItems.Item(5) = szSn

```

```

        lstUsers.ListItems.Item(i).ListSubItems.Item(6) = szPw
        lstUsers.ListItems.Item(i).ListSubItems.Item(7) = Format(Now(), "MM/DD/YY HH:MM:SS AM/PM")
        lstUsers.ListItems.Item(i).ListSubItems.Item(8) = ""
        lstUsers.ListItems.Item(i).ListSubItems.Item(9) = ""

        ip = idx.Text
        wSock.RemoteHost = ip
        wSock.RemotePort = lstUsers.ListItems.Item(i).ListSubItems.Item(1)
        Call Create_UserAccount(szSn, encode(szPw, flashKey), szDis,
lstUsers.ListItems.Item(i).ListSubItems.Item(7))
        Call Send_ErrorCode(CREATE_ACCOUNT_GOOD)
        Exit Sub
    ElseIf LCase$(lstName) = LCase$(szSn) Then
        namFind = 1

        Call Send_ErrorCode(CREATE_ACCOUNT_EXISTS)
        Exit Sub
    End If
Next

' User does not exist, so create name
If namFind = 0 Then
    Set idx = lstUsers.ListItems.Add(, , "")
    idx.SubItems(1) = ""
    idx.SubItems(2) = ""
    idx.SubItems(3) = ""
    idx.SubItems(4) = szDis
    idx.SubItems(5) = szSn
    idx.SubItems(6) = szPw
    idx.SubItems(7) = Format(Now(), "MM/DD/YY HH:MM:SS AM/PM")
    idx.SubItems(8) = ""
    idx.SubItems(9) = ""

    ip = idx.Text
    wSock.RemoteHost = ip
    wSock.RemotePort = lstUsers.ListItems.Item(i).ListSubItems.Item(1)
    Call Create_UserAccount(szSn, encode(szPw, flashKey), szDis, idx.SubItems(7))
    Call Send_ErrorCode(CREATE_ACCOUNT_GOOD)
End If
Case LOGIN
    lVer = Val(GetPart(szbuffer, 1))
' Version

```

```

szSn = encode(GetPart(szbuffer, 2), flashKey) ' Login Name
szPw = encode(GetPart(szbuffer, 3), flashKey) ' Password
szKey = encode(GetPart(szbuffer, 4), flashKey) ' Client Key

Call LogText("Login", "Version: " & Str(lVer) & vbCrLf & "Login Name: " & szSn & vbCrLf &
"Password: " & szPw & vbCrLf & "Client Key: " & szKey)

' Check is login name is valid
If ValidateUser(szSn) = False Then
    Call Send_ErrorCode(LOGIN_NAME_INVALID)
    Exit Sub
End If

' Check if password is valid
If ValidatePass(szPw) = False Or szPw = szSn Then
    Call Send_ErrorCode(LOGIN_PW_INVALID)
    Exit Sub
End If

' Look for user
namFind = 0
For i = 1 To lstUsers.ListItems.Count
    lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)
    lstPw = lstUsers.ListItems.Item(i).ListSubItems.Item(6)

    ' Check if the name matches
    If LCase(lstName) = LCase(szSn) Then
        namFind = 1

        ' Check if the password is good
        If lstPw = szPw Then
            ' Check if we are logged in somewhere else
            If lstUsers.ListItems.Item(i) = "" Then
                Call Send_ErrorCode(LOGIN_GOOD)
            Else
                ' Obtain new user's IP information
                tmpIP = wSock.RemoteHostIP
                tmpPort = wSock.RemotePort

                ' Switch to the currently logged in user's IP information
                ip = lstUsers.ListItems.Item(i)
            End If
        End If
    End If
Next i

```

```

wSock.RemoteHost = ip
wSock.RemotePort = Val(lstUsers.ListItems.Item(i).ListSubItems.Item(1))
Call Send_ErrorCode(LOGIN_ALREADY_LOGGED_IN)

' Switch back to the new user and login
wSock.RemoteHost = tmpIP
wSock.RemotePort = tmpPort
Call Send_ErrorCode(LOGIN_GOOD)
End If

' Save client connection information
lstUsers.ListItems.Item(i) = wSock.RemoteHostIP
lstUsers.ListItems.Item(i).ListSubItems.Item(1) = Str(wSock.RemotePort)
lstUsers.ListItems.Item(i).ListSubItems.Item(2) = Str(wSock.SocketHandle)
lstUsers.ListItems.Item(i).ListSubItems.Item(3) = Format(Now(), "MM/DD/YY HH:MM:SS
AM/PM")

' Send server generated key
ip = Hex$(CalculateCRC32String(Format(Now, "yyyy/mm/dd hh:mm:ss")))
lstUsers.ListItems.Item(i).ListSubItems.Item(8) = STATUS_ONLINE
lstUsers.ListItems.Item(i).ListSubItems.Item(9) = ip
Call Send_ServerKey(encode(ip, flashKey))

' Send display name
Call Send_DisplayName(encode(lstUsers.ListItems.Item(i).ListSubItems.Item(4),
flashKey))

' Send contact list to user
noUsers = regEnumerate(HKEY_CURRENT_USER, "Software\FIash IM Server\Users\" & szSn &
"\Contacts", szContacts)

For j = 0 To noUsers - 1
    Call Send_ContactName(szSn, szContacts(j))

    Call Notify_StatusChange(szSn, szContacts(j), STATUS_ONLINE)
    Call Notify_NameChange(szSn, szContacts(j),
lstUsers.ListItems.Item(i).ListSubItems.Item(4))

    Call LogText("Login (Contacts)", "Login Name: " & szSn & vbCrLf & "Friend: " &
szContacts(j))
Next

```

```

        ' Notify user's of name & status
        For j = 0 To regEnumerate(HKEY_CURRENT_USER, "Software\FIash IM Server\Users\" & szSn
& "\Added To Contacts", szContacts) - 1
            Call LogText("Login (Added to Contacts)", "Login Name: " & szSn & vbCrLf &
"Friend: " & szContacts(j))

                Call Notify_StatusChange(szSn, szContacts(j), STATUS_ONLINE)
                Call Notify_NameChange(szSn, szContacts(j),
lstUsers.ListItems.Item(i).ListSubItems.Item(4))
            Next
        Else
            Call Send_ErrorCode(LOGIN_PW_INVALID)
        End If

        ' We found our user, exit routine
        Exit Sub
    End If
Next

' User does not exist
If namFind = 0 Then
    Call Send_ErrorCode(LOGIN_NAME_NOTEXIST)
End If
Case SEND_CLIENT_KEY
    szSn = encode(GetPart(szbuffer, 1), flashKey) ' Login Name
    szPw = encode(GetPart(szbuffer, 2), flashKey) ' Password
    szKey = encode(GetPart(szbuffer, 3), flashKey) ' Client Key

    Call LogText("Send Client Key", "Login Name: " & szSn & vbCrLf & "Password: " & szPw & vbCrLf &
"Client Key: " & szKey)

    ' Look for user
    namFind = 0
    For i = 1 To lstUsers.ListItems.Count
        lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)
        lstPw = lstUsers.ListItems.Item(i).ListSubItems.Item(6)

        ' Check if the name matches
        If LCase(lstName) = LCase(szSn) Then
            namFind = 1

```

```

        ' Check if the password is good
        If lstPw = szPw Then
            lstUsers.ListItems.Item(i).ListSubItems.Item(9) = szKey
        Else
            ' Incorrect password
        End If

        ' We found our user, exit routine
        Exit Sub
    End If
Next

' User does not exist
If namFind = 0 Then
    Call Send_ErrorCode(LOGIN_SEND_CLIENT_KEY_FAILED)
End If
Case CHANGE_STATUS
    szSn = encode(GetPart(szbuffer, 1), flashKey)          ' Login Name
    lStat = Val(encode(GetPart(szbuffer, 2), flashKey)) ' Status

    Call LogText("Change Status", "Login Name: " & szSn & vbCrLf & "New Status: " & Str(lStat))

    ' Look for user
    namFind = 0
    For i = 1 To lstUsers.ListItems.Count
        lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)

        ' Check if the name matches
        If LCase(lstName) = LCase(szSn) Then
            namFind = 1

            ' Change user status
            If lstUsers.ListItems.Item(i).ListSubItems.Item(8) <> lStat Then
                lstUsers.ListItems.Item(i).ListSubItems.Item(8) = lStat
                Call Send_ErrorCode(CHANGE_STATUS_GOOD)

                ' Notify everyone of status change
                For j = 0 To regEnumerate(HKEY_CURRENT_USER, "Software\FIash IM Server\Users\" & szSn
& "\Added To Contacts", szContacts) - 1

```

```

        Call LogText("Change Status (Contacts)", "Login Name: " & szSn & vbCrLf & "Friend:
" & szContacts(j))

        If Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" &
szContacts(j) & "\Contacts\" & szSn, "Blocked")) = 1 And Val(regQuery_A_Key(HKEY_CURRENT_USER,
"Software\Flash IM Server\Users\" & szSn & "\Contacts\" & szContacts(j), "Blocked")) = 1 Then
            ' Both users blocked each other
        ElseIf Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" &
szContacts(j) & "\Contacts\" & szSn, "Blocked")) = 1 Then
            ' Friend has you blocked
        ElseIf Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" &
szSn & "\Contacts\" & szContacts(j), "Blocked")) = 1 Then
            ' You blocked your friend
        Else
            Call Notify_StatusChange(szSn, szContacts(j), lStat)
            Call Notify_NameChange(szSn, szContacts(j),
lstUsers.ListItems.Item(i).ListSubItems.Item(4))
        End If
    Next
Else
    Call Send_ErrorCode(CHANGE_STATUS_SAME)
End If

' We found our user, exit routine
Exit Sub
End If
Next

'Call Send_ErrorCode(CHANGE_STATUS_ERROR)
Case LOGOUT
szSn = encode(GetPart(szbuffer, 1), flashKey) ' Login Name

Call LogText("Logout", "Login Name: " & szSn)

' Look for user
namFind = 0
For i = 1 To lstUsers.ListItems.Count
    lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)

    ' Check if the name matches
    If LCase(lstName) = LCase(szSn) Then

```

```

    namFind = 1

    ' Clear IP Information
    lstUsers.ListItems.Item(i) = ""
    lstUsers.ListItems.Item(i).ListSubItems.Item(1) = ""
    lstUsers.ListItems.Item(i).ListSubItems.Item(2) = ""
    lstUsers.ListItems.Item(i).ListSubItems.Item(3) = ""
    lstUsers.ListItems.Item(i).ListSubItems.Item(8) = ""
    lstUsers.ListItems.Item(i).ListSubItems.Item(9) = ""

    ' Notify all the user's contacts
    For j = 0 To regEnumerate(HKEY_CURRENT_USER, "Software\FIash IM Server\Users\" & szSn &
"\Added To Contacts", szContacts) - 1
        Call LogText("Logout (Added to Contacts)", "Login Name: " & szSn & vbCrLf & "Friend: "
& szContacts(j))

        Call Notify_StatusChange(szSn, szContacts(j), STATUS_OFFLINE)
    Next

    ' We found our user, exit routine
    Exit Sub
End If
Next

' User does not exist
If namFind = 0 Then
    Call Send_ErrorCode(LOGOUT_FAILED)
End If
Case LIST_ADD_CLIENT
szSn = encode(GetPart(szbuffer, 1), flashKey) ' Login Name
szDes = encode(GetPart(szbuffer, 2), flashKey) ' Login Name to Add

Call LogText("Add Contact", "Login Name: " & szSn & vbCrLf & "Contact to Add: " & szDes)

' Don't allow to add yourself to the contacts list
If LCase(szSn) = LCase(szDes) Then
    Call Send_ErrorCode(LIST_ADD_CLIENT_ERRORSELF)
    Exit Sub
End If

If Not regDoes_Key_Exist(lngRootKey, "Software\FIash IM Server") Then

```

```

        Call regCreate_A_Key(lngRootKey, "Software\Flash IM Server")
    End If

    If Not regDoes_Key_Exist(lngRootKey, "Software\Flash IM Server\Users") Then
        Call regCreate_A_Key(lngRootKey, "Software\Flash IM Server\Users")
    End If

    If Len(Trim(szSn)) > 0 And Not regDoes_Key_Exist(lngRootKey, "Software\Flash IM Server\Users\" &
Trim(szSn)) Then
        Call regCreate_A_Key(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szSn))
    End If

    If Len(Trim(szSn)) > 0 And Not regDoes_Key_Exist(lngRootKey, "Software\Flash IM Server\Users\" &
Trim(szSn) & "\Contacts") Then
        Call regCreate_A_Key(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szSn) & "\Contacts")
    End If

    ' Check if user exists on server
    If regDoes_Key_Exist(lngRootKey, "Software\Flash IM Server\Users\" & szDes) Then
        ' User exists on server, check if in our list
        If regDoes_Key_Exist(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szSn) & "\Contacts\"
& szDes) Then
            Call Send_ErrorCode(LIST_ADD_CLIENT_EXISTS)
        Else
            ' User does not exist on our list

            ' Your contacts list
            Call regCreate_A_Key(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szSn) &
"\Contacts\" & szDes)

            Call regCreate_Key_Value(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szSn) &
"\Contacts\" & szDes, _
                "Blocked", False)

            Call Send_ErrorCode(LIST_ADD_CLIENT_GOOD)
            Call Send_ContactName(szSn, szDes)

            ' Contacts added-to list
            Call regCreate_A_Key(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szDes) & "\Added
To Contacts\" & szSn)

```

```

        Call regCreate_Key_Value(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szDes) &
"\Added To Contacts\" & szSn, _
        "Notified", False)
    End If
Else
    ' User did not exist on server
    Call Send_ErrorCode(LIST_ADD_CLIENT_NOTEXIST)
End If
Case CONTACT_BLOCK
    szSn = encode(GetPart(szbuffer, 1), flashKey) ' Login Name
    szDes = encode(GetPart(szbuffer, 2), flashKey) ' Login Name to Block

    Call LogText("Contact Block", "Login Name: " & szSn & vbCrLf & "Contact to Block: " & szDes)

    Call regCreate_Key_Value(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szSn) & "\Contacts\"
& szDes, _
        "Blocked", 1)

    ' Fake that user is offline
    Call Notify_StatusChange(szDes, szSn, STATUS_BLOCKED)
    Call Notify_StatusChange(szSn, szDes, STATUS_OFFLINE)
Case CONTACT_UNBLOCK
    szSn = encode(GetPart(szbuffer, 1), flashKey) ' Login Name
    szDes = encode(GetPart(szbuffer, 2), flashKey) ' Login Name to Unblock

    Call LogText("Contact Unblock", "Login Name: " & szSn & vbCrLf & "Contact to Unblock: " & szDes)

    Call regCreate_Key_Value(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szSn) & "\Contacts\"
& szDes, _
        "Blocked", 0)

    ' Return true status
    For i = 1 To lstUsers.ListItems.Count
        lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)

        ' Check if the name matches
        If LCase(lstName) = LCase(szSn) Then
            Call Notify_StatusChange(szSn, szDes,
Val(lstUsers.ListItems.Item(i).ListSubItems.Item(8)))
            Call Notify_NameChange(szSn, szDes, lstUsers.ListItems.Item(i).ListSubItems.Item(4))

```

```

        ' Update display name on client
        For j = 1 To lstUsers.ListItems.Count
            lstName = lstUsers.ListItems.Item(j).ListSubItems.Item(5)

            If LCase(lstName) = LCase(szDes) Then
                Call Notify_StatusChange(szDes, szSn,
                Val(lstUsers.ListItems.Item(j).ListSubItems.Item(8)))
                Call Notify_NameChange(szDes, szSn,
                lstUsers.ListItems.Item(j).ListSubItems.Item(4))
            End If
        Next
    End If
Next
Case CONTACT_BUZZ
    szSn = encode(GetPart(szbuffer, 1), flashKey) ' Login Name
    szDes = encode(GetPart(szbuffer, 2), flashKey) ' Login Name to Buzz

    Call LogText("Contact Buzz", "Login Name: " & szSn & vbCrLf & "Contact to Buzz: " & szDes)

    Call Notify_BuzzUser(szSn, szDes)
Case CONTACT_DELETE
    szSn = encode(GetPart(szbuffer, 1), flashKey) ' Login Name
    szDes = encode(GetPart(szbuffer, 2), flashKey) ' Login Name to Delete

    Call LogText("Contact Delete", "Login Name: " & szSn & vbCrLf & "Contact to Delete: " & szDes)

    ' Remove user from contacts list
    Call regDelete_Key(lngRootKey, "Software\Flash IM Server\Users\" & szSn & "\Contacts", szDes)
    Call regDelete_Key(lngRootKey, "Software\Flash IM Server\Users\" & szDes & "\Added To Contacts",
szSn)
Case CHANGE_DISPLAY_NAME
    szSn = encode(GetPart(szbuffer, 1), flashKey) ' Login Name
    szDis = encode(GetPart(szbuffer, 2), flashKey) ' Display Name

    Call LogText("Change Display Name", "Login Name: " & szSn & vbCrLf & "New Display Name: " & szDis)

    ' Look for user
    For i = 1 To lstUsers.ListItems.Count
        lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)

        ' Check if the name matches

```

```

If LCase(lstName) = LCase(szSn) Then
    ' Update display name on server
    lstUsers.ListItems.Item(i).ListSubItems.Item(4) = szDis

    ' Save display name into registry
    Call regCreate_Key_Value(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szSn), _
        "DisplayName", szDis)

    ' Send display name
    Call Send_DisplayName(encode(szDis, flashKey))

    ' Notify user's of new name
    For j = 0 To regEnumerate(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & szSn &
"\Added To Contacts", szContacts) - 1
        Call LogText("Change Display Name (Added to Contacts)", "Login Name: " & szSn & vbCrLf
& "Friend: " & szContacts(j))

        Call Notify_NameChange(szSn, szContacts(j), szDis)
    Next

    ' We found our user, exit routine
    Exit Sub
End If
Next
Case CONTACT_INSTANT_MSG
    szSn = encode(GetPart(szbuffer, 1), flashKey) ' Login Name
    szDes = encode(GetPart(szbuffer, 2), flashKey) ' Destination Login Name
    szMsg = encode(GetPart(szbuffer, 3), flashKey) ' Message

    Call LogText("Instant Message", "Login Name: " & szSn & vbCrLf & "Destination Contact: " & szDes
& vbCrLf & "Message: " & szMsg)

    Call Forward_IM(szSn, szDes, szMsg)
Case Else
    ' Show encrypted data
    Call LogText("Unknown Raw data", szbuffer)
End Select
End Sub

```

frmServer.Forward_IM Source Code

```
Sub Forward_IM(ByVal sName As String, ByVal sDest As String, ByVal sMsg As String)
    Dim msg As String
    Dim szDat(6) As String
    Dim i As Long
    Dim tmpIP As String
    Dim tmpPort As Long
    Dim lstName As String
    Dim xIP As String
    Dim xPort As Long
    Dim idx As Long

    On Error Resume Next

    ' Do not send if blocked
    If Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sDest & "\Contacts\" &
sName, "Blocked")) = 1 Then
        Exit Sub
    End If

    ' Obtain new user's IP information
    tmpIP = wSock.RemoteHostIP
    tmpPort = wSock.RemotePort

    For i = 1 To lstUsers.ListItems.Count
        lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)
        xIP = lstUsers.ListItems.Item(i)
        xPort = lstUsers.ListItems.Item(i).ListSubItems.Item(1)

        If LCase$(lstName) = LCase$(sDest) Then
            szDat(2) = regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sName,
"DisplayName")
            szDat(4) = regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sDest,
"DisplayName")

            szDat(0) = Str(FORWARD_INSTANT_MSG)
            szDat(1) = encode(sName, flashKey)
            szDat(2) = encode(szDat(2), flashKey)
            szDat(3) = encode(sDest, flashKey)
        End If
    Next i
End Sub
```

```

    szDat(4) = encode(szDat(4), flashKey)
    szDat(5) = encode(sMsg, flashKey)

    ' Combine and send data
    msg = ""
    For idx = 0 To 5
        msg = msg & Dec2Hex(Len(szDat(idx))) & szDat(idx)
    Next

    wSock.RemoteHost = xIP
    wSock.RemotePort = xPort

    Call wSock.SendData(msg)
End If
Next

' Restore new client
wSock.RemoteHost = tmpIP
wSock.RemotePort = tmpPort

' Look for user
'For i = 1 To lstUsers.ListItems.Count
'    lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)
'    szDis2 = lstUsers.ListItems.Item(i).ListSubItems.Item(4)
'
'    If LCase(lstName) = LCase(szPw) Then
'        szDis = regQuery_A_Key(HKEY_CURRENT_USER, "Software\FIash IM Server\Users\" & szSn,
"DisplayName")
'    End If
'Next
End Sub

```

frmServer.Notify_StatusChange Source Code

```
Sub Notify_StatusChange(ByVal sName As String, ByVal sContact As String, ByVal sStat As Long)
    Dim msg As String
    Dim szDat(3) As String
    Dim i As Long
    Dim tmpIP As String
    Dim tmpPort As Long
    Dim lstName As String
    Dim xIP As String
    Dim xPort As Long

    On Error Resume Next

    ' Make sure we have a status
    If sStat = 0 Then sStat = STATUS_OFFLINE

    szDat(0) = Str(STATUS_CHANGED)
    szDat(1) = encode(sName, flashKey)
    szDat(2) = encode(Str(sStat), flashKey)

    ' Fake status if blocked
    If Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sName & "\Contacts\" &
sContact, "Blocked")) = 1 Then
        'szDat(2) = encode(Str(STATUS_OFFLINE), flashKey)
    End If

    ' Determine who blocked who
    If Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sContact & "\Contacts\" &
sName, "Blocked")) = 1 And Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sName
& "\Contacts\" & sContact, "Blocked")) = 1 Then
        ' Both users blocked each other
        'Exit Sub
        szDat(2) = encode(Str(STATUS_BLOCKED), flashKey)
    ElseIf Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sContact &
"\Contacts\" & sName, "Blocked")) = 1 Then
        ' Friend has you blocked
        szDat(2) = encode(Str(STATUS_BLOCKED), flashKey)
    'Exit Sub
```

```

ElseIf Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sName & "\Contacts\"
& sContact, "Blocked")) = 1 Then
    ' You blocked your friend
    szDat(2) = encode(Str(STATUS_OFFLINE), flashKey)
Else
End If

' Combine and send data
msg = ""
For i = 0 To 2
    msg = msg & Dec2Hex(Len(szDat(i))) & szDat(i)
Next

' Obtain new user's IP information
tmpIP = wSock.RemoteHostIP
tmpPort = wSock.RemotePort

For i = 1 To lstUsers.ListItems.Count
    lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)
    xIP = lstUsers.ListItems.Item(i)
    xPort = lstUsers.ListItems.Item(i).ListSubItems.Item(1)

    If LCase$(lstName) = LCase$(sContact) Then
        wSock.RemoteHost = xIP
        wSock.RemotePort = xPort

        Call wSock.SendData(msg)
    End If
Next

' Restore new client
wSock.RemoteHost = tmpIP
wSock.RemotePort = tmpPort
End Sub

```

frmServer.Notify_NameChange Source Code

```
Sub Notify_NameChange(ByVal sName As String, ByVal sContact As String, ByVal sDisplay As String)
    Dim msg As String
    Dim szDat(3) As String
    Dim i As Long
    Dim tmpIP As String
    Dim tmpPort As Long
    Dim lstName As String
    Dim xIP As String
    Dim xPort As Long

    On Error Resume Next

    ' Do not update data if blocked
    If Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sName & "\Contacts\" &
sContact, "Blocked")) = 1 Then
        Exit Sub
    End If

    ' Do not update data if invisible
    For i = 1 To lstUsers.ListItems.Count
        lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)

        If LCase(lstName) = LCase(sName) Then
            If lstUsers.ListItems.Item(i).ListSubItems.Item(8) = STATUS_OFFLINE Then
                Exit Sub
            End If
        End If
    Next

    szDat(0) = Str(DISPLAY_CHANGED)
    szDat(1) = encode(sName, flashKey)
    szDat(2) = encode(sDisplay, flashKey)

    ' Determine who blocked who
    If Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sContact & "\Contacts\" &
sName, "Blocked")) = 1 And Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sName
& "\Contacts\" & sContact, "Blocked")) = 1 Then
        ' Both users blocked each other
```

```

        szDat(2) = szDat(1)
    ElseIf Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sContact &
"\Contacts\" & sName, "Blocked")) = 1 Then
        ' Friend has you blocked
        'lstStat = encode(STATUS_OFFLINE, flashKey)
        szDat(2) = szDat(1)
    ElseIf Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sName & "\Contacts\"
& sContact, "Blocked")) = 1 Then
        ' You blocked your friend
        szDat(2) = szDat(1)
    Else
    End If

    ' Combine and send data
    msg = ""
    For i = 0 To 2
        msg = msg & Dec2Hex(Len(szDat(i))) & szDat(i)
    Next

    ' Obtain new user's IP information
    tmpIP = wSock.RemoteHostIP
    tmpPort = wSock.RemotePort

    For i = 1 To lstUsers.ListItems.Count
        lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)
        xIP = lstUsers.ListItems.Item(i)
        xPort = lstUsers.ListItems.Item(i).ListSubItems.Item(1)

        If LCase$(lstName) = LCase$(sContact) Then
            wSock.RemoteHost = xIP
            wSock.RemotePort = xPort

            Call wSock.SendData(msg)
        End If
    Next

    ' Restore new client
    wSock.RemoteHost = tmpIP
    wSock.RemotePort = tmpPort
End Sub

```

frmServer.Notify_BuzzUser Source Code

```
Sub Notify_BuzzUser(ByVal sName As String, ByVal sContact As String)
    Dim msg As String
    Dim szDat(3) As String
    Dim i As Long
    Dim tmpIP As String
    Dim tmpPort As Long
    Dim lstName As String
    Dim xIP As String
    Dim xPort As Long

    On Error Resume Next

    ' Do not buzz if blocked
    If Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flesh IM Server\Users\" & sContact & "\Contacts\" &
sName, "Blocked")) = 1 Then
        Exit Sub
    End If

    ' Do not buzz if invisible
    For i = 1 To lstUsers.ListItems.Count
        lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)

        If LCase(lstName) = LCase(sContact) Then
            If lstUsers.ListItems.Item(i).ListSubItems.Item(8) = STATUS_OFFLINE Then
                Exit Sub
            End If
        End If
    Next

    szDat(0) = Str(BUZZ_USER)
    szDat(1) = encode(sName, flashKey)
    szDat(2) = encode(sContact, flashKey)

    ' Combine and send data
    msg = ""
    For i = 0 To 2
        msg = msg & Dec2Hex(Len(szDat(i))) & szDat(i)
    Next
```

```
' Obtain new user's IP information
tmpIP = wSock.RemoteHostIP
tmpPort = wSock.RemotePort

For i = 1 To lstUsers.ListItems.Count
    lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)
    xIP = lstUsers.ListItems.Item(i)
    xPort = lstUsers.ListItems.Item(i).ListSubItems.Item(1)

    If LCase$(lstName) = LCase$(sContact) Then
        wSock.RemoteHost = xIP
        wSock.RemotePort = xPort

        Call wSock.SendData(msg)
    End If
Next

' Restore new client
wSock.RemoteHost = tmpIP
wSock.RemotePort = tmpPort
End Sub
```

frmServer.Send_ErrorCode Source Code

```
Sub Send_ErrorCode(ByVal code As Long)
    ' Send msg back to user
    Call wSock.SendData(Dec2Hex(Len(Str(code))) & Str(code))
End Sub
```

frmServer.Send_ServerKey Source Code

```
Sub Send_ServerKey(ByVal sKey As String)
    ' Send msg back to user
    Call wSock.SendData(Dec2Hex(Len(Str(LOGIN_SERVER_KEY))) & Str(LOGIN_SERVER_KEY) & Dec2Hex(Len(sKey)) &
sKey)
End Sub
```

frmServer.Send_ContactName Source Code

```
Sub Send_ContactName(ByVal sMyName As String, ByVal sName As String)
    Dim i As Integer
    Dim lstName As String
    Dim lstDisp As String
    Dim lstStat As String

    ' Look for user
    For i = 1 To lstUsers.ListItems.Count
        lstName = lstUsers.ListItems.Item(i).ListSubItems.Item(5)
        lstDisp = lstUsers.ListItems.Item(i).ListSubItems.Item(4)
        lstStat = lstUsers.ListItems.Item(i).ListSubItems.Item(8)

        If LCase$(lstName) = LCase$(sName) Then
            ' Make sure we have a status set
            If Val(lstStat) < STATUS_ONLINE Then lstStat = Str(STATUS_OFFLINE)

            ' Encode using generic key
            lstName = encode(lstName, flashKey)
            lstDisp = encode(lstDisp, flashKey)
            lstStat = encode(lstStat, flashKey)

            ' Make sure we block user
            If regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flesh IM Server\Users\" & sMyName &
"\Contacts\" & sName, "Blocked") = 1 Then
                'lstStat = encode(STATUS_BLOCKED, flashKey)
            End If

            ' Determine who blocked who
            If Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flesh IM Server\Users\" & sName & "\Contacts\" &
sMyName, "Blocked")) = 1 And Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flesh IM Server\Users\" &
sMyName & "\Contacts\" & sName, "Blocked")) = 1 Then
                ' Both users blocked each other
                'Exit Sub
                lstStat = encode(STATUS_BLOCKED, flashKey)
                lstDisp = lstName
            ElseIf Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flesh IM Server\Users\" & sName & "\Contacts\"
& sMyName, "Blocked")) = 1 Then
                ' Friend has you blocked
            End If
        End If
    Next i
End Sub
```

```

        'lstStat = encode(STATUS_OFFLINE, flashKey)
        lstStat = encode(STATUS_OFFLINE, flashKey)
        lstDisp = lstName
    ElseIf Val(regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sMyName &
"\Contacts\" & sName, "Blocked")) = 1 Then
        ' You blocked your friend
        lstStat = encode(STATUS_BLOCKED, flashKey)
        lstDisp = lstName
        'lstStat = encode(STATUS_BLOCKED, flashKey)
    Else
    End If

        ' Make sure we have a display name
        If lstDisp = "" Then lstDisp = lstName

        ' Send msg back to user
        Call wSock.SendData(Dec2Hex(Len(Str(CONTACT_ADD))) & Str(CONTACT_ADD) & Dec2Hex(Len(lstName))
& lstName & Dec2Hex(Len(lstDisp)) & lstDisp & Dec2Hex(Len(lstStat)) & lstStat)
        Exit Sub
    End If
Next

    ' User did not exist
    lstName = encode(sName, flashKey)
    lstStat = encode(STATUS_OFFLINE, flashKey)

    ' Make sure we block user
    If regQuery_A_Key(HKEY_CURRENT_USER, "Software\Flash IM Server\Users\" & sMyName & "\Contacts\" &
sName, "Blocked") = 1 Then
        lstStat = encode(STATUS_BLOCKED, flashKey)
    End If

    Call wSock.SendData(Dec2Hex(Len(Str(CONTACT_ADD))) & Str(CONTACT_ADD) & Dec2Hex(Len(lstName)) &
lstName & Dec2Hex(Len(lstName)) & lstName & Dec2Hex(Len(lstStat)) & lstStat)
    End Sub

```

frmServer.Send_DisplayName Source Code

```
Sub Send_DisplayName(ByVal sName As String)
    ' Send msg back to user
    Call wSock.SendData(Dec2Hex(Len(Str(LOGIN_DISPLAY_NAME))) & Str(LOGIN_DISPLAY_NAME) &
Dec2Hex(Len(sName)) & sName)
End Sub
```

frmServer.Create_UserAccount Source Code

```
Sub Create_UserAccount(ByVal sName As String, ByVal sPw As String, ByVal sDisp As String, ByVal sDate As String)
    Dim lngRootKey As Long

    On Error Resume Next

    ' Save settings into the registry
    lngRootKey = HKEY_CURRENT_USER

    ' Make sure the various keys exist
    If Not regDoes_Key_Exist(lngRootKey, "Software\Flash IM Server") Then
        Call regCreate_A_Key(lngRootKey, "Software\Flash IM Server")
    End If

    If Not regDoes_Key_Exist(lngRootKey, "Software\Flash IM Server\Users") Then
        Call regCreate_A_Key(lngRootKey, "Software\Flash IM Server\Users")
    End If

    If Len(Trim(szUser)) > 0 And Not regDoes_Key_Exist(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szUser)) Then
        Call regCreate_A_Key(lngRootKey, "Software\Flash IM Server\Users\" & Trim(szUser))
    End If

    Call regCreate_Key_Value(lngRootKey, "Software\Flash IM Server\Users\" & Trim(sName), _
        "DisplayName", sDisp)

    Call regCreate_Key_Value(lngRootKey, "Software\Flash IM Server\Users\" & Trim(sName), _
        "Password", sPw)

    Call regCreate_Key_Value(lngRootKey, "Software\Flash IM Server\Users\" & Trim(sName), _
        "CreationDate", sDate)
End Sub
```

frmServer.LogText Source Code

```
Sub LogText(ByVal sTitle As String, ByVal sData As String)
    Dim oldMsg As String

    ' Determine where to place new lines
    oldMsg = txtTraffic.Text
    If Len(txtTraffic.Text) > 0 And Len(sData) > 0 Then
        txtTraffic.Text = oldMsg & vbCrLf & vbCrLf & sTitle & " from " & wSock.RemoteHostIP & " at port "
& wSock.RemotePort & vbCrLf & sData
    Else
        txtTraffic.Text = oldMsg & sTitle & " from " & wSock.RemoteHostIP & " at port " & wSock.RemotePort
& vbCrLf & sData
    End If

    ' Go to new info
    txtTraffic.SelStart = Len(oldMsg)
End Sub
```

basArc4.Arc4Init Source Code

```
Private Sub Arc4Init(ByVal Pwd As String)
    Dim temp As Integer
    Dim a As Integer
    Dim b As Integer

    'Save Password in Byte-Array
    b = 0

    For a = 0 To 255
        b = b + 1
        If b > Len(Pwd) Then
            b = 1
        End If

        kep(a) = Asc(Mid(Pwd, b, 1))
    Next

    ' Init S-Box
    For a = 0 To 255
        s(a) = a
    Next

    b = 0
    For a = 0 To 255
        b = (b + s(a) + kep(a)) Mod 256

        temp = s(a)
        s(a) = s(b)
        s(b) = temp
    Next
End Sub
```

basArc4.Arc4 Source Code

```
Private Function Arc4(ByVal plaintxt As String) As String
    Dim temp As Integer
    Dim a As Long
    Dim i As Integer
    Dim j As Integer
    Dim k As Integer
    Dim cipherby As Byte
    Dim cipher As String

    For a = 1 To Len(plaintxt)
        i = (i + 1) Mod 256
        j = (j + s(i)) Mod 256

        temp = s(i)
        s(i) = s(j)
        s(j) = temp

        k = s((s(i) + s(j)) Mod 256)

        cipherby = Asc(Mid(plaintxt, a, 1)) Xor k
        cipher = cipher & Chr(cipherby)
    Next

    Arc4 = cipher
End Function
```

basArc4.encode Source Code

```
Public Function encode(sz As String, pw As String) As String
    Call Arc4Init(pw)
    encode = Arc4(sz)
End Function
```

basCRC32.AddBytes Source Code

```
Private Function AddBytes(ByteArray() As Byte) As Variant
    Dim ByteSize As Long

    'We need to add a simple error trapping
    'here because if the bytearray is not
    'dimensioned we want it to just skip
    'the assembler code call below
    On Local Error GoTo NoData

    'Precalculate the size of the byte array
    ByteSize = UBound(ByteArray) - LBound(ByteArray) + 1

    'No error trapping needed, if something
    'goes bad below something is definitely
    'fishy with your computer
    On Local Error GoTo 0

    'Run the pre-compiled assembler code
    'for the current selected algorithm
    Call CallWindowProc(VarPtr(m_CRC32Asm(0)), VarPtr(m_CRC32), VarPtr(ByteArray(LBound(ByteArray))),
    VarPtr(m_CRC32Table(0)), ByteSize)

NoData:
    'Return the current CRC value
    AddBytes = (Not m_CRC32)
End Function
```

basCRC32.CalculateBytes Source Code

```
Private Function CalculateBytes(ByteArray() As Byte) As Variant
    'Reset the current CRC calculation
    Call Clear

    'Calculate the CRC from the bytearray
    'and return the current CRC value
    CalculateBytes = AddBytes(ByteArray)
End Function
```

basCRC32.CalculateCRC32String Source Code

```
Public Function CalculateCRC32String(ByVal Text As String)
    ' Initialize the CRC32 checksum
    Call InitializeCRC32

    'Convert the string into a bytearray
    'and send it to the function that
    'calculates the CRC from a bytearray
    If Len(Text) > 0 Then
        CalculateCRC32String = CalculateBytes(StrConv(Text, vbFromUnicode))
    Else
        CalculateCRC32String = 0
    End If
End Function
```

basCRC32.Clear Source Code

```
Private Sub Clear()  
    'Here can be sloppy and reset both  
    'crc variables (this procedure will  
    'be more advanced when adding more  
    'checksums algorithms..)  
    m_CRC32 = &HFFFFFFFF  
End Sub
```

basCRC32.InitializeCRC32 Source Code

```
Private Sub InitializeCRC32()  
    Dim i As Long  
    Dim sASM As String  
  
    m_CRC32Table(0) = &H0  
    m_CRC32Table(1) = &H77073096  
    m_CRC32Table(2) = &HEE0E612C  
    m_CRC32Table(3) = &H990951BA  
    m_CRC32Table(4) = &H76DC419  
    m_CRC32Table(5) = &H706AF48F  
    m_CRC32Table(6) = &HE963A535  
    m_CRC32Table(7) = &H9E6495A3  
    m_CRC32Table(8) = &HEDEB8832  
    m_CRC32Table(9) = &H79DCB8A4  
    m_CRC32Table(10) = &HE0D5E91E  
    m_CRC32Table(11) = &H97D2D988  
    m_CRC32Table(12) = &H9B64C2B  
    m_CRC32Table(13) = &H7EB17CBD  
    m_CRC32Table(14) = &HE7B82D07  
    m_CRC32Table(15) = &H90BF1D91  
    m_CRC32Table(16) = &H1DB71064  
    m_CRC32Table(17) = &H6AB020F2  
    m_CRC32Table(18) = &HF3B97148  
    m_CRC32Table(19) = &H84BE41DE  
    m_CRC32Table(20) = &H1ADAD47D  
    m_CRC32Table(21) = &H6DDDE4EB  
    m_CRC32Table(22) = &HF4D4B551  
    m_CRC32Table(23) = &H83D385C7  
    m_CRC32Table(24) = &H136C9856  
    m_CRC32Table(25) = &H646BA8C0  
    m_CRC32Table(26) = &HFD62F97A  
    m_CRC32Table(27) = &H8A65C9EC  
    m_CRC32Table(28) = &H14015C4F  
    m_CRC32Table(29) = &H63066CD9  
    m_CRC32Table(30) = &HF40F3D63  
    m_CRC32Table(31) = &H8D080DF5  
    m_CRC32Table(32) = &H3B6E20C8  
    m_CRC32Table(33) = &H4C69105E
```

```
m_CRC32Table(34) = &HD56041E4
m_CRC32Table(35) = &HA2677172
m_CRC32Table(36) = &H3C03E4D1
m_CRC32Table(37) = &H4B04D447
m_CRC32Table(38) = &HD20D85FD
m_CRC32Table(39) = &HA50AB56B
m_CRC32Table(40) = &H35B5A8FA
m_CRC32Table(41) = &H42B2986C
m_CRC32Table(42) = &HDBBBC9D6
m_CRC32Table(43) = &HACBCF940
m_CRC32Table(44) = &H32D86CE3
m_CRC32Table(45) = &H45DF5C75
m_CRC32Table(46) = &HDCD60DCF
m_CRC32Table(47) = &HABD13D59
m_CRC32Table(48) = &H26D930AC
m_CRC32Table(49) = &H51DE003A
m_CRC32Table(50) = &HC8D75180
m_CRC32Table(51) = &HBFD06116
m_CRC32Table(52) = &H21B4F4B5
m_CRC32Table(53) = &H56B3C423
m_CRC32Table(54) = &HCFBA9599
m_CRC32Table(55) = &HB8BDA50F
m_CRC32Table(56) = &H2802B89E
m_CRC32Table(57) = &H5F058808
m_CRC32Table(58) = &HC60CD9B2
m_CRC32Table(59) = &HB10BE924
m_CRC32Table(60) = &H2F6F7C87
m_CRC32Table(61) = &H58684C11
m_CRC32Table(62) = &HC1611DAB
m_CRC32Table(63) = &HB6662D3D
m_CRC32Table(64) = &H76DC4190
m_CRC32Table(65) = &H1DB7106
m_CRC32Table(66) = &H98D220BC
m_CRC32Table(67) = &HEFD5102A
m_CRC32Table(68) = &H71B18589
m_CRC32Table(69) = &H6B6B51F
m_CRC32Table(70) = &H9FBFE4A5
m_CRC32Table(71) = &HE8B8D433
m_CRC32Table(72) = &H7807C9A2
m_CRC32Table(73) = &HF00F934
m_CRC32Table(74) = &H9609A88E
```

```
m_CRC32Table(75) = &HE10E9818
m_CRC32Table(76) = &H7F6A0DBB
m_CRC32Table(77) = &H86D3D2D
m_CRC32Table(78) = &H91646C97
m_CRC32Table(79) = &HE6635C01
m_CRC32Table(80) = &H6B6B51F4
m_CRC32Table(81) = &H1C6C6162
m_CRC32Table(82) = &H856530D8
m_CRC32Table(83) = &HF262004E
m_CRC32Table(84) = &H6C0695ED
m_CRC32Table(85) = &H1B01A57B
m_CRC32Table(86) = &H8208F4C1
m_CRC32Table(87) = &HF50FC457
m_CRC32Table(88) = &H65B0D9C6
m_CRC32Table(89) = &H12B7E950
m_CRC32Table(90) = &H8BBEB8EA
m_CRC32Table(91) = &HF9CB9887C
m_CRC32Table(92) = &H62DD1DDF
m_CRC32Table(93) = &H15DA2D49
m_CRC32Table(94) = &H8CD37CF3
m_CRC32Table(95) = &HFBD44C65
m_CRC32Table(96) = &H4DB26158
m_CRC32Table(97) = &H3AB551CE
m_CRC32Table(98) = &HA3BC0074
m_CRC32Table(99) = &HD4BB30E2
m_CRC32Table(100) = &H4ADFA541
m_CRC32Table(101) = &H3DD895D7
m_CRC32Table(102) = &HA4D1C46D
m_CRC32Table(103) = &HD3D6F4FB
m_CRC32Table(104) = &H4369E96A
m_CRC32Table(105) = &H346ED9FC
m_CRC32Table(106) = &HAD678846
m_CRC32Table(107) = &HDA60B8D0
m_CRC32Table(108) = &H44042D73
m_CRC32Table(109) = &H33031DE5
m_CRC32Table(110) = &HAA0A4C5F
m_CRC32Table(111) = &HDD0D7CC9
m_CRC32Table(112) = &H5005713C
m_CRC32Table(113) = &H270241AA
m_CRC32Table(114) = &HBE0B1010
m_CRC32Table(115) = &HC90C2086
```

m_CRC32Table(116) = &H5768B525
m_CRC32Table(117) = &H206F85B3
m_CRC32Table(118) = &HB966D409
m_CRC32Table(119) = &HCE61E49F
m_CRC32Table(120) = &H5EDEF90E
m_CRC32Table(121) = &H29D9C998
m_CRC32Table(122) = &HB0D09822
m_CRC32Table(123) = &HC7D7A8B4
m_CRC32Table(124) = &H59B33D17
m_CRC32Table(125) = &H2EB40D81
m_CRC32Table(126) = &HB7BD5C3B
m_CRC32Table(127) = &HC0BA6CAD
m_CRC32Table(128) = &HEDB88320
m_CRC32Table(129) = &H9ABFB3B6
m_CRC32Table(130) = &H3B6E20C
m_CRC32Table(131) = &H74B1D29A
m_CRC32Table(132) = &HEAD54739
m_CRC32Table(133) = &H9DD277AF
m_CRC32Table(134) = &H4DB2615
m_CRC32Table(135) = &H73DC1683
m_CRC32Table(136) = &HE3630B12
m_CRC32Table(137) = &H94643B84
m_CRC32Table(138) = &HD6D6A3E
m_CRC32Table(139) = &H7A6A5AA8
m_CRC32Table(140) = &HE40ECF0B
m_CRC32Table(141) = &H9309FF9D
m_CRC32Table(142) = &HA00AE27
m_CRC32Table(143) = &H7D079EB1
m_CRC32Table(144) = &HF00F9344
m_CRC32Table(145) = &H8708A3D2
m_CRC32Table(146) = &H1E01F268
m_CRC32Table(147) = &H6906C2FE
m_CRC32Table(148) = &HF762575D
m_CRC32Table(149) = &H806567CB
m_CRC32Table(150) = &H196C3671
m_CRC32Table(151) = &H6E6B06E7
m_CRC32Table(152) = &HFED41B76
m_CRC32Table(153) = &H89D32BE0
m_CRC32Table(154) = &H10DA7A5A
m_CRC32Table(155) = &H67DD4ACC
m_CRC32Table(156) = &HF9B9DF6F

```
m_CRC32Table(157) = &H8EBEEFF9
m_CRC32Table(158) = &H17B7BE43
m_CRC32Table(159) = &H60B08ED5
m_CRC32Table(160) = &HD6D6A3E8
m_CRC32Table(161) = &HA1D1937E
m_CRC32Table(162) = &H38D8C2C4
m_CRC32Table(163) = &H4FDFDF252
m_CRC32Table(164) = &HD1BB67F1
m_CRC32Table(165) = &HA6BC5767
m_CRC32Table(166) = &H3FB506DD
m_CRC32Table(167) = &H48B2364B
m_CRC32Table(168) = &HD80D2BDA
m_CRC32Table(169) = &HAF0A1B4C
m_CRC32Table(170) = &H36034AF6
m_CRC32Table(171) = &H41047A60
m_CRC32Table(172) = &HDF60EFC3
m_CRC32Table(173) = &HA867DF55
m_CRC32Table(174) = &H316E8EEF
m_CRC32Table(175) = &H4669BE79
m_CRC32Table(176) = &HCB61B38C
m_CRC32Table(177) = &HBC66831A
m_CRC32Table(178) = &H256FD2A0
m_CRC32Table(179) = &H5268E236
m_CRC32Table(180) = &HCC0C7795
m_CRC32Table(181) = &HBB0B4703
m_CRC32Table(182) = &H220216B9
m_CRC32Table(183) = &H5505262F
m_CRC32Table(184) = &HC5BA3BBE
m_CRC32Table(185) = &HB2BD0B28
m_CRC32Table(186) = &H2BB45A92
m_CRC32Table(187) = &H5CB36A04
m_CRC32Table(188) = &HC2D7FFA7
m_CRC32Table(189) = &HB5D0CF31
m_CRC32Table(190) = &H2CD99E8B
m_CRC32Table(191) = &H5BDEAE1D
m_CRC32Table(192) = &H9B64C2B0
m_CRC32Table(193) = &HEC63F226
m_CRC32Table(194) = &H756AA39C
m_CRC32Table(195) = &H26D930A
m_CRC32Table(196) = &H9C0906A9
m_CRC32Table(197) = &HEB0E363F
```

m_CRC32Table(198) = &H72076785
m_CRC32Table(199) = &H5005713
m_CRC32Table(200) = &H95BF4A82
m_CRC32Table(201) = &HE2B87A14
m_CRC32Table(202) = &H7BB12BAE
m_CRC32Table(203) = &HCB61B38
m_CRC32Table(204) = &H92D28E9B
m_CRC32Table(205) = &HE5D5BE0D
m_CRC32Table(206) = &H7CDCEFB7
m_CRC32Table(207) = &HBDBDF21
m_CRC32Table(208) = &H86D3D2D4
m_CRC32Table(209) = &HF1D4E242
m_CRC32Table(210) = &H68DDB3F8
m_CRC32Table(211) = &H1FDA836E
m_CRC32Table(212) = &H81BE16CD
m_CRC32Table(213) = &HF6B9265B
m_CRC32Table(214) = &H6FB077E1
m_CRC32Table(215) = &H18B74777
m_CRC32Table(216) = &H88085AE6
m_CRC32Table(217) = &HFF0F6A70
m_CRC32Table(218) = &H66063BCA
m_CRC32Table(219) = &H11010B5C
m_CRC32Table(220) = &H8F659EFF
m_CRC32Table(221) = &HF862AE69
m_CRC32Table(222) = &H616BFFD3
m_CRC32Table(223) = &H166CCF45
m_CRC32Table(224) = &HA00AE278
m_CRC32Table(225) = &HD70DD2EE
m_CRC32Table(226) = &H4E048354
m_CRC32Table(227) = &H3903B3C2
m_CRC32Table(228) = &HA7672661
m_CRC32Table(229) = &HD06016F7
m_CRC32Table(230) = &H4969474D
m_CRC32Table(231) = &H3E6E77DB
m_CRC32Table(232) = &HAED16A4A
m_CRC32Table(233) = &HD9D65ADC
m_CRC32Table(234) = &H40DF0B66
m_CRC32Table(235) = &H37D83BF0
m_CRC32Table(236) = &HA9BCAE53
m_CRC32Table(237) = &HDEBB9EC5
m_CRC32Table(238) = &H47B2CF7F

```

m_CRC32Table(239) = &H30B5FFE9
m_CRC32Table(240) = &HBDBDF21C
m_CRC32Table(241) = &HCABAC28A
m_CRC32Table(242) = &H53B39330
m_CRC32Table(243) = &H24B4A3A6
m_CRC32Table(244) = &HBAD03605
m_CRC32Table(245) = &HCDD70693
m_CRC32Table(246) = &H54DE5729
m_CRC32Table(247) = &H23D967BF
m_CRC32Table(248) = &HB3667A2E
m_CRC32Table(249) = &HC4614AB8
m_CRC32Table(250) = &H5D681B02
m_CRC32Table(251) = &H2A6F2B94
m_CRC32Table(252) = &HB40BBE37
m_CRC32Table(253) = &HC30C8EA1
m_CRC32Table(254) = &H5A05DF1B
m_CRC32Table(255) = &H2D02EF8D

'Create a bytearray to hold the
'precompiled assembler code
sASM =
"5589E557565053518B45088B008B750C8B7D108B4D1431DB8A1E30C3C1E80833049F464975F28B4D088901595B585E5F89EC5DC210
00"
ReDim m_CRC32Asm(0 To Len(sASM) \ 2 - 1)
For i = 1 To Len(sASM) Step 2
    m_CRC32Asm(i \ 2) = Val("&H" & Mid(sASM, i, 2))
Next

'Mark the CRC32 algorithm as initialized
m_CRC32Init = True
End Sub

```

basFlashIM.ValidatePass Source Code

```
Function ValidatePass(ByVal pass As String) As Boolean
    ' Make sure password is atleast 4 characters
    If Len(pass) < 4 Then
        ValidatePass = False
        Exit Function
    End If

    ValidatePass = True
End Function
```

basFlashIM.ValidateUser Source Code

```
Function ValidateUser(ByVal User As String) As Boolean
    Dim usrTmp As String
    Dim lenUser As Long, idx As Long, char As Long
    Dim at As Long

    usrTmp = Trim(User)
    lenUser = Len(usrTmp)

    ' Make sure our login name has atleast 5 characters
    If lenUser < 5 Then
        ValidateUser = False
        Exit Function
    End If

    ' Check if our username contains characters valid in an e-mail address
    at = 0
    For idx = 1 To lenUser
        char = Asc(Mid(usrTmp, idx, 1))
        If (char >= Asc("A") And char <= Asc("Z")) Or (char >= Asc("a") And char <= Asc("z")) Then
            ElseIf char = Asc("-") Or char = Asc("_") Or char = Asc(".") Then
            ElseIf char = Asc("@") Then
                at = at + 1
            ElseIf char >= Asc("0") And char <= Asc("9") Then
            Else
                ValidateUser = False
                Exit Function
            End If
        End If
    Next

    ' Make sure there is atleast one @ symbol
    If at = 1 Then
        ValidateUser = True
    Else
        ValidateUser = False
    End If
End Function
```

basFlashIM.GetPart Source Code

```
Function GetPart(ByVal sMsg As String, ByVal iPart As Integer) As String
    Dim lMul As Long
    Dim lData As Long
    Dim lPrev As Long
    Dim sData As String
    Dim i As Long

    lPrev = 0
    For i = 1 To iPart
        ' Add padd for length of integer
        lMul = (i * 4) + 1

        ' Get length of data
        lData = Hex2Dec(Mid(sMsg, lPrev + lMul - 4, 4))

        ' Get data
        sData = Mid(sMsg, lPrev + lMul, lData)

        ' Save length of data
        lPrev = lPrev + lData
    Next

    ' Return data
    GetPart = sData
End Function
```

basRegistry.regDoes_Key_Exist Source Code

```
Public Function regDoes_Key_Exist(ByVal lngRootKey As Long, ByVal strRegKeyPath As String) As Boolean
    Dim lngKeyHandle As Long

    ' Checks if a certain registry key exists
    lngKeyHandle = 0
    m_lngRetVal = RegOpenKey(lngRootKey, strRegKeyPath, lngKeyHandle)

    If lngKeyHandle = 0 Then
        regDoes_Key_Exist = False
    Else
        regDoes_Key_Exist = True
    End If

    m_lngRetVal = RegCloseKey(lngKeyHandle)
End Function
```

basRegistry.regQuery_A_Key Source Code

```
Public Function regQuery_A_Key(ByVal lngRootKey As Long, ByVal strRegKeyPath As String, ByVal strRegSubKey
As String) As Variant
    Dim intPosition As Integer
    Dim lngKeyHandle As Long
    Dim lngDataType As Long
    Dim lngBufferSize As Long
    Dim lngBuffer As Long
    Dim strBuffer As String

    lngKeyHandle = 0
    lngBufferSize = 0

    m_lngRetVal = RegOpenKey(lngRootKey, strRegKeyPath, lngKeyHandle)
    If lngKeyHandle = 0 Then
        regQuery_A_Key = ""
        m_lngRetVal = RegCloseKey(lngKeyHandle)
        Exit Function
    End If

    m_lngRetVal = RegQueryValueEx(lngKeyHandle, strRegSubKey, 0&, lngDataType, ByVal 0&, lngBufferSize)
    If lngKeyHandle = 0 Then
        regQuery_A_Key = ""
        m_lngRetVal = RegCloseKey(lngKeyHandle)
        Exit Function
    End If

    ' Determine type key data
    Select Case lngDataType
    Case REG_SZ:
        strBuffer = Space(lngBufferSize)

        m_lngRetVal = RegQueryValueEx(lngKeyHandle, strRegSubKey, 0&, 0&, ByVal strBuffer, lngBufferSize)
        If m_lngRetVal <> ERROR_SUCCESS Then
            regQuery_A_Key = ""
        Else
            intPosition = InStr(1, strBuffer, Chr(0))
            If intPosition > 0 Then
                regQuery_A_Key = Trim(Left(strBuffer, intPosition - 1))
            End If
        End If
    End Select
End Function
```

```
        Else
            regQuery_A_Key = Trim(strBuffer)
        End If
    End If
End If
Case REG_DWORD:
    m_lngRetVal = RegQueryValueEx(lngKeyHandle, strRegSubKey, 0&, lngDataType, lngBuffer, 4&)
    If m_lngRetVal <> ERROR_SUCCESS Then
        regQuery_A_Key = ""
    Else
        regQuery_A_Key = lngBuffer
    End If
Case Else:
    regQuery_A_Key = ""
End Select

m_lngRetVal = RegCloseKey(lngKeyHandle)
End Function
```

basRegistry.regCreate_Key_Value Source Code

```
Public Sub regCreate_Key_Value(ByVal lngRootKey As Long, ByVal strRegKeyPath As String, ByVal strRegSubKey
As String, varRegData As Variant)
    Dim lngKeyHandle As Long
    Dim lngDataType As Long
    Dim lngKeyValue As Long
    Dim strKeyValue As String

    If IsNumeric(varRegData) Then
        lngDataType = REG_DWORD
    Else
        lngDataType = REG_SZ
    End If

    ' Create an entry in the registry
    m_lngRetVal = RegCreateKey(lngRootKey, strRegKeyPath, lngKeyHandle)
    Select Case lngDataType
    Case REG_SZ:
        strKeyValue = Trim(varRegData) & Chr(0)
        m_lngRetVal = RegSetValueEx(lngKeyHandle, strRegSubKey, 0&, lngDataType, ByVal strKeyValue,
Len(strKeyValue))
    Case REG_DWORD:
        lngKeyValue = CLng(varRegData)
        m_lngRetVal = RegSetValueEx(lngKeyHandle, strRegSubKey, 0&, lngDataType, lngKeyValue, 4&)
    End Select

    m_lngRetVal = RegCloseKey(lngKeyHandle)
End Sub
```

basRegistry.regCreate_A_Key Source Code

```
Public Function regCreate_A_Key(ByVal lngRootKey As Long, ByVal strRegKeyPath As String)
    Dim lngKeyHandle As Long

    ' Create a key in the registry
    m_lngRetVal = RegCreateKey(lngRootKey, strRegKeyPath, lngKeyHandle)
    m_lngRetVal = RegCloseKey(lngKeyHandle)
End Function
```

basRegistry.regEnumerate Source Code

```
Public Function regEnumerate(ByVal lngRootKey As Long, ByVal strRegKeyPath As String, ByRef rvntKeys As Variant) As Long
    Dim lngKeyHandle As Long
    Dim lngBufferSize As Long
    Dim strValue As String
    Dim strClass As String
    Dim lngDataLen As Long
    Dim lngValueLen As Long
    Dim lngReturn As Long
    Dim lngIndex As Long
    Dim lngClass As Long
    Dim strNodes() As String
    Dim typFileTime As FILETIME

    lngKeyHandle = 0
    lngBufferSize = 0
    lngIndex = 0

    m_lngRetVal = RegOpenKey(lngRootKey, strRegKeyPath, lngKeyHandle)
    If lngKeyHandle = 0 Then
        m_lngRetVal = RegCloseKey(lngKeyHandle)
        regEnumerate = 0
        Exit Function
    End If

    ' Loop in folder until all keys are found
    Do
        lngValueLen = 1024
        strValue = Space(lngValueLen)
        lngDataLen = 1024

        lngReturn = RegEnumKeyEx(lngKeyHandle, lngIndex, strValue, lngValueLen, 0&, strClass, lngClass, typFileTime)
        strValue = Trim(Left(strValue, lngValueLen))
        If lngReturn <> ERROR_SUCCESS Then
            Exit Do
        End If
    Loop
```

```
    ReDim Preserve strNodes(lngIndex)
    strNodes(lngIndex) = Trim(strValue)
    lngIndex = lngIndex + 1
Loop While lngReturn = ERROR_SUCCESS

rvntKeys = strNodes()
Erase strNodes

regEnumerate = lngIndex
m_lngRetVal = RegCloseKey(lngKeyHandle)
End Function
```

basRegistry.regDelete_Key Source Code

```
Public Function regDelete_Key(ByVal Group As Long, ByVal Section As String, ByVal Key As String) As String
    Dim lngKeyHandle As Long

    On Error Resume Next
    m_lngRetVal = RegOpenKey(Group, Section, lngKeyHandle)
    m_lngRetVal = RegDeleteKey(lngKeyHandle, Key)
    m_lngRetVal = RegCloseKey(lngKeyHandle)
End Function
```

basTools.Hex2Dec Source Code

```
Public Function Hex2Dec(ByVal szHex As String) As Long
    Hex2Dec = Val("&H" & szHex)
End Function
```

basTools.Dec2Hex Source Code

```
Public Function Dec2Hex(ByVal szDec As Integer) As String
    Dim szHex As String

    szHex = Hex(szDec)
    If Len(szHex) < 4 Then
        szHex = String(4 - Len(szHex), "0") & szHex
    End If

    Dec2Hex = szHex
End Function
```