

Forum:[Hack Forums](#)

Thread: [\[Release\]MW3 Tekno, Custom Classes Unlocker, No Recoil etc.](#)

MrRoot - October 23, 2014, 12:35 pm

I was so bored i made like a MultiTool/Hack for Cal of Duty: Modern Warfare 3, the point of this hack is to unlock every custom class you have in server and if custom classes are not visible this shows them, and i didn't wanted to release this on MPGH or some other shitty website cause it would ruin the game, for any suggestions PM me or Reply! :D

This is how my hack looks like:

So you can:

1. Unlock Custom Classes
2. Make AUG HBAR Class from MW2
3. Set your Field of View
4. Turn No recoil ON/OFF
- 5.Level\Experience Hack -> In Extras Tab
- 6.Prestige Hack -> Extras Tab too

Hack was made in Visual Basic 2010 Express

So Source :

Form1 :

Code Public Class Form1

```
Private Sub Button1_Click(sender As System.Object, e As System.EventArgs) Handles  
Button1.Click  
    Timer1.Enabled = True  
End Sub
```

```
Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles  
MyBase.Load  
    Timer1.Enabled = False  
    Timer2.Enabled = False  
    Timer3.Enabled = True  
    Timer4.Enabled = False
```

End Sub

```
Private Sub Timer1_Tick(sender As System.Object, e As System.EventArgs) Handles  
Timer1.Tick  
    WriteInteger("iw5mp", &H6BCF4C8, Value:=7)    <-Custom Classes Unlock  
    WriteInteger("iw5mp", &H6BCF4CC, Value:=0)  
    WriteInteger("iw5mp", &H6BCF4D0, Value:=0)  
    WriteInteger("iw5mp", &H6BCF4D8, Value:=0)  
    WriteInteger("iw5mp", &H6BCF4E8, Value:=0)  
    WriteInteger("iw5mp", &H6BCF4EC, Value:=0)  
    WriteInteger("iw5mp", &H6BCF4F0, Value:=0)
```

```
WriteInteger("iw5mp", &H6BCF4F4, Value:=0)
WriteInteger("iw5mp", &H6BCF4E4, Value:=0)
End Sub
```

```
Private Sub Button2_Click(sender As System.Object, e As System.EventArgs) Handles
Button2.Click
    WriteInteger("iw5mp", &H1CDBFE8, Value:=50331648) <- AUG HBAR
    WriteInteger("iw5mp", &H1CDBFEC, Value:=131162)
    WriteInteger("iw5mp", &H1CDBFF0, Value:=524297)
End Sub
```

```
Private Sub Timer2_Tick(sender As System.Object, e As System.EventArgs) Handles
Timer2.Tick
    WriteInteger("iw5mp", &H10056160, Value:=NumericUpDown1.Value)
End Sub
```

```
Private Sub NumericUpDown1_ValueChanged(sender As System.Object, e As
System.EventArgs) Handles NumericUpDown1.ValueChanged
    Timer2.Enabled = True
End Sub
```

```
Private Sub ExitToolStripMenuItem_Click(sender As System.Object, e As System.EventArgs)
Handles ExitToolStripMenuItem.Click
    Me.Close()
End Sub
```

```
Private Sub AboutToolStripMenuItem_Click(sender As System.Object, e As System.EventArgs)
Handles AboutToolStripMenuItem.Click
    MsgBox("Made by ~Mr.Root!", MsgBoxStyle.Information)
End Sub
```

```
Private Sub LevelHackToolStripMenuItem_Click(sender As System.Object, e As
System.EventArgs) Handles LevelHackToolStripMenuItem.Click
    Form2.Show()
End Sub
```

```
Private Sub PrestigeHackToolStripMenuItem_Click(sender As System.Object, e As
System.EventArgs) Handles PrestigeHackToolStripMenuItem.Click
    Form3.Show()
End Sub
```

```
Private Sub PictureBox1_Click(sender As System.Object, e As System.EventArgs) Handles
PictureBox1.Click
    Me.Close()
End Sub
```

```
Private Sub Timer3_Tick(sender As System.Object, e As System.EventArgs) Handles
Timer3.Tick

End Sub
```

```
Private Sub Label1_Click(sender As System.Object, e As System.EventArgs) Handles
Label1.Click
```

End Sub

Private Sub RadioButton1_CheckedChanged(sender As System.Object, e As System.EventArgs)

Timer1.Enabled = True

End Sub

Private Sub Timer4_Tick(sender As System.Object, e As System.EventArgs) Handles Timer4.Tick

WriteInteger("iw5mp", &H9761A4, Value:=0)

WriteInteger("iw5mp", &H96EE14, Value:=0)

WriteInteger("iw5mp", &H976198, Value:=0)

WriteInteger("iw5mp", &H97619C, Value:=0)

WriteInteger("iw5mp", &H9761A8, Value:=0)

WriteInteger("iw5mp", &H1063800, Value:=0)

WriteInteger("iw5mp", &H1063804, Value:=0)

End Sub

Private Sub RadioButton1_CheckedChanged_1(sender As System.Object, e As System.EventArgs) Handles RadioButton1.CheckedChanged

Timer4.Enabled = True

End Sub

Private Sub RadioButton2_CheckedChanged(sender As System.Object, e As System.EventArgs) Handles RadioButton2.CheckedChanged

Timer4.Enabled = False

End Sub

End Class

Form2[LEVEL HACK]:

Code Public Class Form2

Private Sub PictureBox1_Click(sender As System.Object, e As System.EventArgs) Handles PictureBox1.Click

Me.Close()

End Sub

Private Sub Button1_Click(sender As System.Object, e As System.EventArgs) Handles Button1.Click

WriteInteger("iw5mp", &H1CDBA54, Value:=TextBox1.Text)

End Sub

End Class

Form3[PRESTIGE HACK]:

Code Public Class Form3

Private Sub PictureBox1_Click(sender As System.Object, e As System.EventArgs) Handles PictureBox1.Click

Me.Close()

End Sub

Private Sub ComboBox1_SelectedIndexChanged(sender As System.Object, e As System.EventArgs) Handles ComboBox1.SelectedIndexChanged

WriteInteger("iw5mp", &H1CDBC64, Value:=ComboBox1.Text)

```

End Sub
End ClassReadWriteMemory:
Code Module ReadWritingMemory
    Private Declare Function OpenProcess Lib "kernel32" (ByVal dwDesiredAccess As Integer,
ByVal bInheritHandle As Integer, ByVal dwProcessId As Integer) As Integer

    Private Declare Function WriteProcessMemory1 Lib "kernel32" Alias "WriteProcessMemory"
(ByVal hProcess As Integer, ByVal lpBaseAddress As Integer, ByRef lpBuffer As Integer, ByVal
nSize As Integer, ByRef lpNumberOfBytesWritten As Integer) As Integer
    Private Declare Function WriteProcessMemory2 Lib "kernel32" Alias "WriteProcessMemory"
(ByVal hProcess As Integer, ByVal lpBaseAddress As Integer, ByRef lpBuffer As Single, ByVal
nSize As Integer, ByRef lpNumberOfBytesWritten As Integer) As Single
    Private Declare Function WriteProcessMemory3 Lib "kernel32" Alias "WriteProcessMemory"
(ByVal hProcess As Integer, ByVal lpBaseAddress As Integer, ByRef lpBuffer As Long, ByVal nSize
As Integer, ByRef lpNumberOfBytesWritten As Integer) As Long

    Private Declare Function ReadProcessMemory1 Lib "kernel32" Alias "ReadProcessMemory"
(ByVal hProcess As Integer, ByVal lpBaseAddress As Integer, ByRef lpBuffer As Integer, ByVal
nSize As Integer, ByRef lpNumberOfBytesWritten As Integer) As Integer
    Private Declare Function ReadProcessMemory2 Lib "kernel32" Alias "ReadProcessMemory"
(ByVal hProcess As Integer, ByVal lpBaseAddress As Integer, ByRef lpBuffer As Single, ByVal
nSize As Integer, ByRef lpNumberOfBytesWritten As Integer) As Single
    Private Declare Function ReadProcessMemory3 Lib "kernel32" Alias "ReadProcessMemory"
(ByVal hProcess As Integer, ByVal lpBaseAddress As Integer, ByRef lpBuffer As Long, ByVal nSize
As Integer, ByRef lpNumberOfBytesWritten As Integer) As Long

    Const PROCESS_ALL_ACCESS = &H1F0FF

    Public Function WriteDMAInteger(ByVal Process As String, ByVal Address As Integer, ByVal
Offsets As Integer(), ByVal Value As Integer, ByVal Level As Integer, Optional ByVal nsize As
Integer = 4) As Boolean
        Try
            Dim lvl As Integer = Address
            For i As Integer = 1 To Level
                lvl = ReadInteger(Process, lvl, nsize) + Offsets(i - 1)
            Next
            WriteInteger(Process, lvl, Value, nsize)
            Return True
        Catch ex As Exception
            Return False
        End Try
    End Function

    Public Function ReadDMAInteger(ByVal Process As String, ByVal Address As Integer, ByVal
Offsets As Integer(), ByVal Level As Integer, Optional ByVal nsize As Integer = 4) As Integer
        Try
            Dim lvl As Integer = Address
            For i As Integer = 1 To Level
                lvl = ReadInteger(Process, lvl, nsize) + Offsets(i - 1)
            Next
            Dim vBuffer As Integer
            vBuffer = ReadInteger(Process, lvl, nsize)
            Return vBuffer
        End Try
    End Function

```

Catch ex As Exception

End Try

End Function

Public Function WriteDMAFloat(ByVal Process As String, ByVal Address As Integer, ByVal Offsets As Integer(), ByVal Value As Single, ByVal Level As Integer, Optional ByVal nsize As Integer = 4) As Boolean

Try

Dim lvl As Integer = Address

For i As Integer = 1 To Level

lvl = ReadFloat(Process, lvl, nsize) + Offsets(i - 1)

Next

WriteFloat(Process, lvl, Value, nsize)

Return True

Catch ex As Exception

Return False

End Try

End Function

Public Function ReadDMAFloat(ByVal Process As String, ByVal Address As Integer, ByVal Offsets As Integer(), ByVal Level As Integer, Optional ByVal nsize As Integer = 4) As Single

Try

Dim lvl As Integer = Address

For i As Integer = 1 To Level

lvl = ReadFloat(Process, lvl, nsize) + Offsets(i - 1)

Next

Dim vBuffer As Single

vBuffer = ReadFloat(Process, lvl, nsize)

Return vBuffer

Catch ex As Exception

End Try

End Function

Public Function WriteDMALong(ByVal Process As String, ByVal Address As Integer, ByVal Offsets As Integer(), ByVal Value As Long, ByVal Level As Integer, Optional ByVal nsize As Integer = 4) As Boolean

Try

Dim lvl As Integer = Address

For i As Integer = 1 To Level

lvl = ReadLong(Process, lvl, nsize) + Offsets(i - 1)

Next

WriteLong(Process, lvl, Value, nsize)

Return True

Catch ex As Exception

Return False

End Try

End Function

Public Function ReadDMALong(ByVal Process As String, ByVal Address As Integer, ByVal Offsets As Integer(), ByVal Level As Integer, Optional ByVal nsize As Integer = 4) As Long

Try

```

Dim lvl As Integer = Address
For i As Integer = 1 To Level
lvl = ReadLong(Process, lvl, nsize) + Offsets(i - 1)
Next
Dim vBuffer As Long
vBuffer = ReadLong(Process, lvl, nsize)
Return vBuffer
Catch ex As Exception

```

```

End Try
End Function

```

```

Public Sub WriteNOPs(ByVal ProcessName As String, ByVal Address As Long, ByVal NOPNum
As Integer)

```

```

    Dim C As Integer
    Dim B As Integer
    If ProcessName.EndsWith(".exe") Then
        ProcessName = ProcessName.Replace(".exe", "")
    End If
    Dim MyP As Process() = Process.GetProcessesByName(ProcessName)
    If MyP.Length = 0 Then
        MessageBox.Show(ProcessName & " isn't open!")
        Exit Sub
    End If
    Dim hProcess As IntPtr = OpenProcess(PROCESS_ALL_ACCESS, 0, MyP(0).Id)
    If hProcess = IntPtr.Zero Then
        MessageBox.Show("Failed to open " & ProcessName & "!")
        Exit Sub
    End If

```

```

    B = 0
    For C = 1 To NOPNum
        Call WriteProcessMemory1(hProcess, Address + B, &H90, 1, 0&)
        B = B + 1
    Next C
End Sub

```

```

Public Sub WriteXBytes(ByVal ProcessName As String, ByVal Address As Long, ByVal Value As
String)

```

```

    If ProcessName.EndsWith(".exe") Then
        ProcessName = ProcessName.Replace(".exe", "")
    End If
    Dim MyP As Process() = Process.GetProcessesByName(ProcessName)
    If MyP.Length = 0 Then
        MessageBox.Show(ProcessName & " isn't open!")
        Exit Sub
    End If
    Dim hProcess As IntPtr = OpenProcess(PROCESS_ALL_ACCESS, 0, MyP(0).Id)
    If hProcess = IntPtr.Zero Then
        MessageBox.Show("Failed to open " & ProcessName & "!")
        Exit Sub
    End If

```

```
Dim C As Integer
Dim B As Integer
Dim D As Integer
Dim V As Byte
```

```
B = 0
D = 1
For C = 1 To Math.Round((Len(Value) / 2))
V = Val("&H" & Mid$(Value, D, 2))
Call WriteProcessMemory1(hProcess, Address + B, V, 1, 0&)
B = B + 1
D = D + 2
Next C
```

```
End Sub
```

```
Public Sub WriteInteger(ByVal ProcessName As String, ByVal Address As Integer, ByVal Value
As Integer, Optional ByVal nsize As Integer = 4)
```

```
    If ProcessName.EndsWith(".exe") Then
        ProcessName = ProcessName.Replace(".exe", "")
    End If
    Dim MyP As Process() = Process.GetProcessesByName(ProcessName)
    If MyP.Length = 0 Then
        MessageBox.Show(ProcessName & " isn't open!")
        Exit Sub
    End If
    Dim hProcess As IntPtr = OpenProcess(PROCESS_ALL_ACCESS, 0, MyP(0).Id)
    If hProcess = IntPtr.Zero Then
        MessageBox.Show("Failed to open " & ProcessName & "!")
        Exit Sub
    End If
```

```
    Dim hAddress, vBuffer As Integer
    hAddress = Address
    vBuffer = Value
    WriteProcessMemory1(hProcess, hAddress, CInt(vBuffer), nsize, 0)
End Sub
```

```
Public Sub WriteFloat(ByVal ProcessName As String, ByVal Address As Integer, ByVal Value As
Single, Optional ByVal nsize As Integer = 4)
```

```
    If ProcessName.EndsWith(".exe") Then
        ProcessName = ProcessName.Replace(".exe", "")
    End If
    Dim MyP As Process() = Process.GetProcessesByName(ProcessName)
    If MyP.Length = 0 Then
        MessageBox.Show(ProcessName & " isn't open!")
        Exit Sub
    End If
    Dim hProcess As IntPtr = OpenProcess(PROCESS_ALL_ACCESS, 0, MyP(0).Id)
    If hProcess = IntPtr.Zero Then
        MessageBox.Show("Failed to open " & ProcessName & "!")
        Exit Sub
    End If
```

```
Dim hAddress As Integer
Dim vBuffer As Single
```

```
hAddress = Address
vBuffer = Value
WriteProcessMemory2(hProcess, hAddress, vBuffer, nsize, 0)
End Sub
```

```
Public Sub WriteLong(ByVal ProcessName As String, ByVal Address As Integer, ByVal Value As
Long, Optional ByVal nsize As Integer = 4)
    If ProcessName.EndsWith(".exe") Then
        ProcessName = ProcessName.Replace(".exe", "")
    End If
    Dim MyP As Process() = Process.GetProcessesByName(ProcessName)
    If MyP.Length = 0 Then
        MessageBox.Show(ProcessName & " isn't open!")
        Exit Sub
    End If
    Dim hProcess As IntPtr = OpenProcess(PROCESS_ALL_ACCESS, 0, MyP(0).Id)
    If hProcess = IntPtr.Zero Then
        MessageBox.Show("Failed to open " & ProcessName & "!")
        Exit Sub
    End If
```

```
Dim hAddress As Integer
Dim vBuffer As Long
```

```
hAddress = Address
vBuffer = Value
WriteProcessMemory3(hProcess, hAddress, vBuffer, nsize, 0)
End Sub
```

```
Public Function ReadInteger(ByVal ProcessName As String, ByVal Address As Integer, Optional
ByVal nsize As Integer = 4) As Integer
    If ProcessName.EndsWith(".exe") Then
        ProcessName = ProcessName.Replace(".exe", "")
    End If
    Dim MyP As Process() = Process.GetProcessesByName(ProcessName)
    If MyP.Length = 0 Then
        MessageBox.Show(ProcessName & " isn't open!")
        Exit Function
    End If
    Dim hProcess As IntPtr = OpenProcess(PROCESS_ALL_ACCESS, 0, MyP(0).Id)
    If hProcess = IntPtr.Zero Then
        MessageBox.Show("Failed to open " & ProcessName & "!")
        Exit Function
    End If
```

```
Dim hAddress, vBuffer As Integer
hAddress = Address
ReadProcessMemory1(hProcess, hAddress, vBuffer, nsize, 0)
Return vBuffer
```

End Function

```
Public Function ReadFloat(ByVal ProcessName As String, ByVal Address As Integer, Optional
ByVal nsize As Integer = 4) As Single
    If ProcessName.EndsWith(".exe") Then
        ProcessName = ProcessName.Replace(".exe", "")
    End If
    Dim MyP As Process() = Process.GetProcessesByName(ProcessName)
    If MyP.Length = 0 Then
        MessageBox.Show(ProcessName & " isn't open!")
        Exit Function
    End If
    Dim hProcess As IntPtr = OpenProcess(PROCESS_ALL_ACCESS, 0, MyP(0).Id)
    If hProcess = IntPtr.Zero Then
        MessageBox.Show("Failed to open " & ProcessName & "!")
        Exit Function
    End If

    Dim hAddress As Integer
    Dim vBuffer As Single

    hAddress = Address
    ReadProcessMemory2(hProcess, hAddress, vBuffer, nsize, 0)
    Return vBuffer
End Function
```

```
Public Function ReadLong(ByVal ProcessName As String, ByVal Address As Integer, Optional
ByVal nsize As Integer = 4) As Long
    If ProcessName.EndsWith(".exe") Then
        ProcessName = ProcessName.Replace(".exe", "")
    End If
    Dim MyP As Process() = Process.GetProcessesByName(ProcessName)
    If MyP.Length = 0 Then
        MessageBox.Show(ProcessName & " isn't open!")
        Exit Function
    End If
    Dim hProcess As IntPtr = OpenProcess(PROCESS_ALL_ACCESS, 0, MyP(0).Id)
    If hProcess = IntPtr.Zero Then
        MessageBox.Show("Failed to open " & ProcessName & "!")
        Exit Function
    End If

    Dim hAddress As Integer
    Dim vBuffer As Long

    hAddress = Address
    ReadProcessMemory3(hProcess, hAddress, vBuffer, nsize, 0)
    Return vBuffer
End Function
```

End Module So if you think you can make this better or add some more better features pls PM me or leave a reply :D

\$lick - October 23, 2014, 12:49 pm

Please use the Code tags.

That way the thread will look nicer.

ElementalTree - October 23, 2014, 2:19 pm

You should've posted there if you don't even know what code tags are .-.

Hex Hooves - October 23, 2014, 3:47 pm

This isn't even all that impressive. All of the [relevant] addresses in Call of Dooty are static, and the values have been known for years.

Also you posting this wouldn't affect the game's quality, it's already ruined, and things like this have been around for years.

denn1s - October 24, 2014, 2:43 am

May can u help me to install teko mw3 ? it didn't worked for me

MrRoot - October 24, 2014, 7:40 am

(Oct 23, 2014 03:47 PM)Hex Hooves Wrote: [\[CITE\]](#) This isn't even all that impressive. All of the [relevant] addresses in Call of Dooty are static, and the values have been known for years.

Also you posting this wouldn't affect the game's quality, it's already ruined, and things like this have been around for years.

Yes but the addresses aren't public, you have to search for them in memory browser or find search for static ones and those have 2-3-4 layers

frawg - October 24, 2014, 7:44 am

Interesting... thanks for sharing mate.

MrRoot - October 24, 2014, 7:44 am

(Oct 23, 2014 12:49 PM)Lazy. Wrote: [\[CITE\]](#) Please use the Code tags.

That way the thread will look nicer.

Took your advice, it does look nicer, thank you :D

r00t_z - October 24, 2014, 8:25 am

Looks sweet! Great share, bro. Keep it up!

FreeAnneFrank - October 30, 2014, 7:12 am

Nice man, will post to bookmark it :) Thanks

MrRoot - November 4, 2014, 7:30 am

(Oct 30, 2014 07:12 AM).RKO Wrote: [\[CITE\]](#) Nice man, will post to bookmark it :) Thanks
ye i might release it, cuz i added sum more features :D