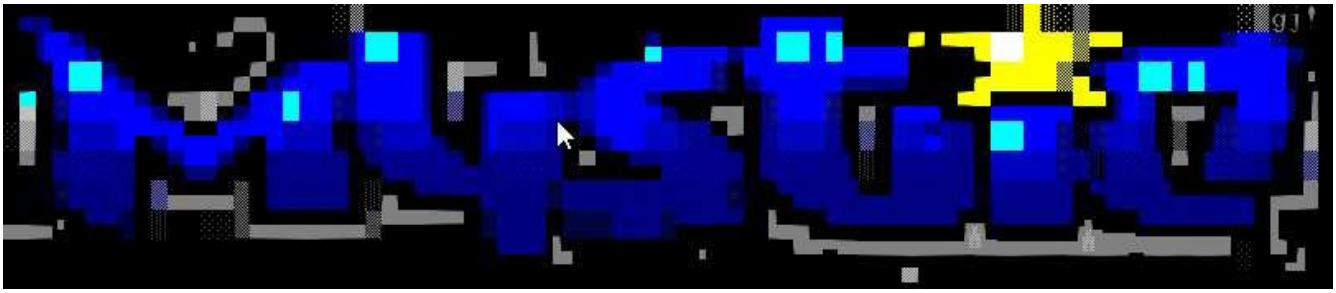

MPL Training
Lesson 19
Black Panther

-=-=-=-=-=-=-=-=-

Let’s have a little fun in this lesson. ;)

In this lesson, let’s take a look at how we can work with strings. Let’s say you have a string, something like ‘Mystic BBS Rocks!’, and you want to do something fancy when it displays on the users screen.

Well, using MPL, we can do many different things to this string as it draws on the screen. Let’s start with something simple.

```
Uses
  Cfg

Var
  Str       : String
  Count     : Byte
  Count1 : Byte

Begin
  Str:=’Mystic BBS Rocks!’
  Count:=Length(Str)
  ClrScr
  For Count1:=Count downto 1 Do
  Begin
    GotoXY(Count1,WhereY)
    Write(Str[Count1])
    Delay(100)
  End
  WriteLn(‘’)
End
```

If you copy/paste this into your favorite text editor, and save it as a .mps file and compile it, you will see a simple, fancy way to display text to the screen. Basically, it will start at the end of the string, and start printing it in reverse from the ‘!’ all the way to the ‘M’.

How does this work? Correctly. (sorry, I’m a bit of a smart-a$$ tonight)

Well, we can look at the string ‘Mystic BBS Rocks!’, as an array of characters. Each character has it’s own address where it can be accessed. The entire string is located in the variable ‘Str’.

Let’s look at it this way. I know I’m more of a visual learner, so this really helped me.

Str = Mystic BBS Rocks!

Str[1] = M
Str[2] = y
Str[3] = s
Str[4] = t
Str[5] = i
Str[6] = c
...etc
Str[16] = s
Str[17] = !

I should have picked a shorter string… ;)

So, if we want to access any of the letters by themselves, we can use the Str[x], where x is the location within the string.

If you notice in the code above, we used the line Count:=Length(Str). What this did, is count the number of characters, or length, of what was being held in Str. In our case, it came back as 17.

We then have a For/Do loop, that is starting at the end of the string, and working it’s way back to the beginning. Within the loop, we are setting the XY coordinates, printing one character, and then delaying so the user can see what’s happening.

Kinda cool, right? ;)

Now, let’s change it up just a bit… ;)

```
Uses
  Cfg

Const
  DelayTime=70

Var
  Str    : String
  Count  : Byte
  Count1 : Byte

Begin
  Str:='Mystic BBS Rocks!'
  Count:=Length(Str)
  ClrScr
```

```
  For Count1:=Count downto 1 Do
  Begin
    GotoXY(Count1,WhereY)
    Write('|15'+Str[Count1])
    Delay(DelayTime)
    GotoXY(Count1,WhereY)
    Write('|11'+Str[Count1])
    Delay(DelayTime)
    GotoXY(Count1,WhereY)
    Write('|09'+Str[Count1])
    Delay(DelayTime)
    GotoXY(Count1,WhereY)
    Write('|11'+Str[Count1])
    Delay(DelayTime)
    GotoXY(Count1,WhereY)
    Write('|15'+Str[Count1])
    Delay(DelayTime)
  End
  WriteLn('')
End
```

Now, this time, it will still display the string backwards, as it did before, but it will change the colors of each letter as it's displaying them.

There is shorter ways of doing this, but we'll save that for another lesson.

Let's look at how we could display the characters at random onto the screen.

```
Uses
  Cfg

Const
  DelayTime=100

Var
  Str    : String
  Count  : Byte
  Count1 : Byte

Begin
  Str:='Mystic BBS Rocks!'
  Count:=Length(Str)
  ClrScr
  Repeat
    Count1:=Random(Count)+1
    GotoXY(Count1,WhereY)
    Write(Str[Count1])
    Delay(DelayTime)
  Until Keypressed
```

```
  WriteLn('')
End
```

With this one, the MPL will generate a random number between 1 and the length of the string. In this case, 17. It will then display that character to the screen, in the right location due to using the location within the string as the x coordinate.

This could also be combined with the previous example, where we were changing the colors of the characters as they displayed to the screen.

Or, we could get fancy, like xqtr has in the past. :) This is borrowed from one of his MPLs, and slightly modified to fit our use here.

```
Uses
  Cfg

Const
  DelayTime=150

Var
  Str    : String
  Count  : Byte
  Count1 : Byte
  Y      : Byte

Begin
  Str:='Mystic BBS Rocks!'
  Y:=WhereY
  Count:=Length(Str)
  ClrScr
  Repeat
    TextColor(8)
    GotoXY(Count1,Y)
    Write('.')
    Delay(DelayTime/10)
    GotoXY(Count1,Y)
    Write('o')
    Delay(DelayTime/10)
    GotoXY(Count1,Y)
    Write(Str[Count1])
    Delay(DelayTime/3)
    GotoXY(Count1,Y)
    TextColor(9)
    Write(Str[Count1])
    Delay(DelayTime/3)
    TextColor(11)
    GotoXY(Count1,Y)
    Write(Str[Count1])
    Delay(DelayTime/10)
```

```
    Count1:=Count1+1
  Until Count1=Count+1
  WriteLn('')
End
```

This writes the string the correct way on the screen, but gives it a cool effect while it’s being drawn to the screen.

Basically, what I’m trying to show you here, is there is no limitations as to what you can do with your programming. If you can think of a cool effect that would amaze people, and get their attention. DO IT! Why should all text just appear in the proper order, one letter at a time, and always look boring. Come up with your own ways to make some Really Crazy Sh!t! ;)