

Data Types and Sizes

Type	Size
char, unsigned char, signed char	1 byte
short, unsigned short	2 bytes
int, unsigned int	4 bytes
long, unsigned long	4 bytes
float	4 bytes
double	8 bytes
long double	8 bytes

```

C:\WDZLECTURE\WDZLEC_WIN32NETWORKPROGRAMMING\WDZEXAMPLE\W1\1\1\W01_SIZES_OF_TVP...
sizeof( Tv_Char ) = 1 bytes, Addr = 0x0012FF7F
sizeof( Tv_UnsignedChar ) = 1 bytes, Addr = 0x0012FF7E
sizeof( Tv_SignedChar ) = 1 bytes, Addr = 0x0012FF7D
-----
sizeof( Tv_Short ) = 2 bytes, Addr = 0x0012FF7A
sizeof( Tv_UnsignedShort ) = 2 bytes, Addr = 0x0012FF78
-----
sizeof( Tv_Int ) = 4 bytes, Addr = 0x0012FF74
sizeof( Tv_UnsignedInt ) = 4 bytes, Addr = 0x0012FF70
-----
sizeof( Tv_Long ) = 4 bytes, Addr = 0x0012FF6C
sizeof( Tv_UnsignedLong ) = 4 bytes, Addr = 0x0012FF68
-----
sizeof( Tv_Float ) = 4 bytes, Addr = 0x0012FF64
-----
sizeof( Tv_Double ) = 8 bytes, Addr = 0x0012FF5C
-----
sizeof( Tv_LongDouble ) = 8 bytes, Addr = 0x0012FF54
-----
Press any key to continue

```

```

//local -----
char Tv_Char = 1 ;
unsigned char Tv_UnsignedChar = 2 ;
signed char Tv_SignedChar = 3 ;

short Tv_Short = 4 ;
unsigned short Tv_UnsignedShort = 5 ;

int Tv_Int = 6 ;
unsigned int Tv_UnsignedInt = 7 ;

long Tv_Long = 8 ;
unsigned long Tv_UnsignedLong = 9 ;

float Tv_Float = 10 ;

double Tv_Double = 11 ;

long double Tv_LongDouble = 12 ;

```

```

Address: 0x0012FF54
0012FF4C CC CC CC CC CC CC CC CC 徽徽徽徽
0012FF54 00 00 00 00 00 00 28 40 ..... (@
0012FF5C 00 00 00 00 00 00 26 40 ..... &@
0012FF64 00 00 20 41 09 00 00 00 .. A. ...
0012FF6C 08 00 00 00 07 00 00 00 .....
0012FF74 06 00 00 00 05 00 04 00 .....
0012FF7C CC 03 02 01 C0 FF 12 00 .....
0012FF84 79 14 40 00 01 00 00 00 y.@.....
0012FF8C 40 12 37 00 18 13 37 00 @ 7 7

```

Pointer Types and Sizes

Type	Size
char*, unsigned char*, signed char*	4 bytes
short*, unsigned short*	4 bytes
int*, unsigned int*	4 bytes
long*, unsigned long*	4 bytes
float*	4 bytes
double*	4 bytes
long double*	4 bytes
user_struct*, user_class*, void*, etc*	4 bytes

```
C:\WDZLECTURE\WDZLEC_WIN32NETWORKPROGRAMMING\WDZEXAMPLE\1일차\01.SIZES OF T...
sizeof( Tv_Char )      = 1 bytes, Addr = 0x0012FF7F
sizeof( Tv_Int )      = 4 bytes, Addr = 0x0012FF78
-----
sizeof( Tv_PtrChar )  = 4 bytes, Addr = 0x0012FF74
sizeof( Tv_PtrInt )   = 4 bytes, Addr = 0x0012FF70
-----
Press any key to continue
```

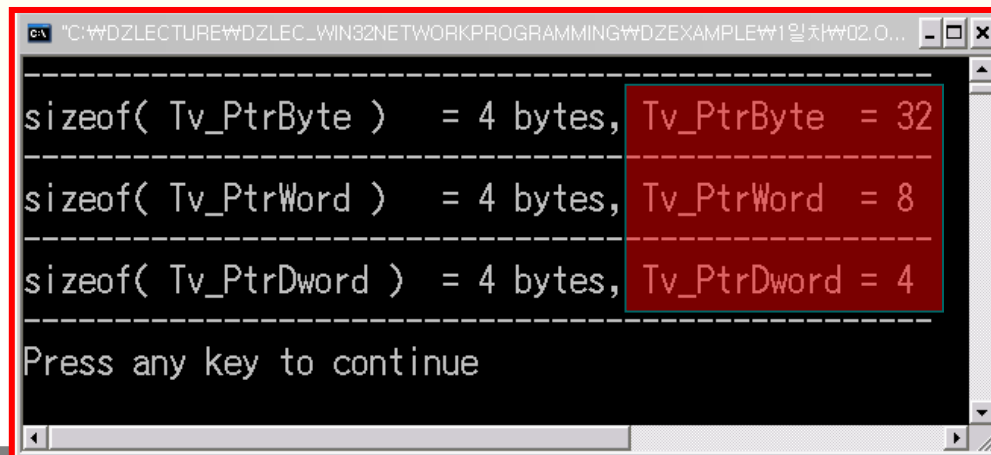
Address: 0x0012FF70

0012FF6C	CC	CC	CC	CC	徹徹
0012FF70	78	FF	12	00	x...
0012FF74	7F	FF	12	00	...
0012FF78	06	00	00	00	...
0012FF7C	CC	CC	CC	01	徹..
0012FF80	CC	FF	12	00	...
0012FF84	B9	14	40	00	@

```
//local -----
char      Tv_Char      = 1 ;
int       Tv_Int       = 6 ;
char*     Tv_PtrChar   = &Tv_Char ;
int*      Tv_PtrInt    = &Tv_Int ;
```

Pointer 연산

```
//local -----  
BYTE*      Tv_PtrByte      = 0 ;  
WORD*      Tv_PtrWord     = 0 ;  
DWORD*     Tv_PtrDword    = 0 ;  
  
//code -----  
Tv_PtrByte = Tv_PtrByte + 2 * 16 ;  
Tv_PtrWord = Tv_PtrWord + 4 ;  
Tv_PtrDword = Tv_PtrDword + 1 ;
```



The screenshot shows a Windows command prompt window with the following output:

```
sizeof( Tv_PtrByte ) = 4 bytes, Tv_PtrByte = 32  
-----  
sizeof( Tv_PtrWord ) = 4 bytes, Tv_PtrWord = 8  
-----  
sizeof( Tv_PtrDword ) = 4 bytes, Tv_PtrDword = 4  
-----  
Press any key to continue
```

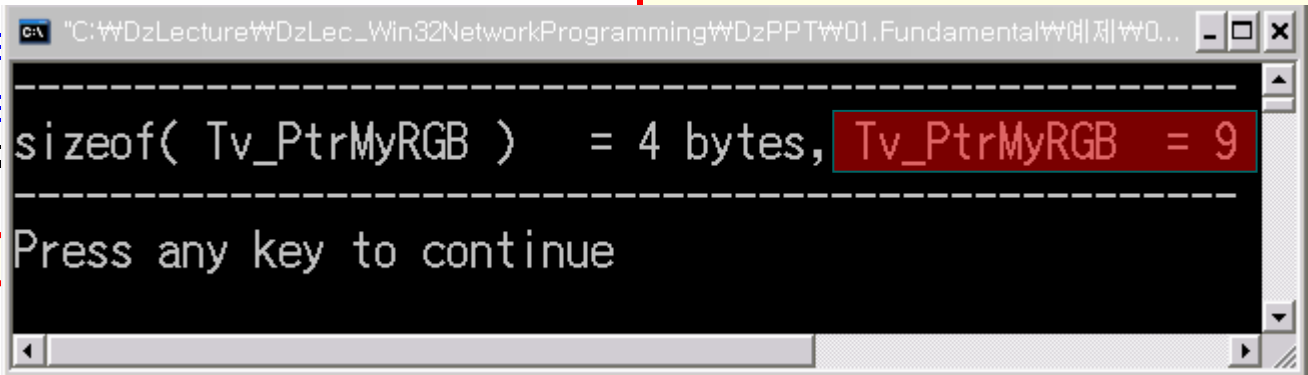
The values for `Tv_PtrByte`, `Tv_PtrWord`, and `Tv_PtrDword` are highlighted in red in the original image.

Structure Pointer

```
// structure -----  
typedef struct St_myRGB  
{  
    unsigned char    Blue ;  
    unsigned char    Green ;  
    unsigned char    Red ;  
} *Ptr_myRGB;
```

St_myRGB는 char형 이 3개로 구성
즉, 3바이트 의 구조체
위의 구조체에 +3을 해주면
3+3+3 = 9가되는것이다

```
// local -  
Ptr_myRGB    Tv_PtrMyRGB    = 0 ;  
  
//code -----  
Tv_PtrMyRGB = Tv_PtrMyRGB + 3 ;
```



```
C:\WDzLecture\WDzLec_Win32NetworkProgramming\WDzPPT\01.Fundamental\예제\0... - _ _ x  
-----  
sizeof( Tv_PtrMyRGB ) = 4 bytes, Tv_PtrMyRGB = 9  
-----  
Press any key to continue
```

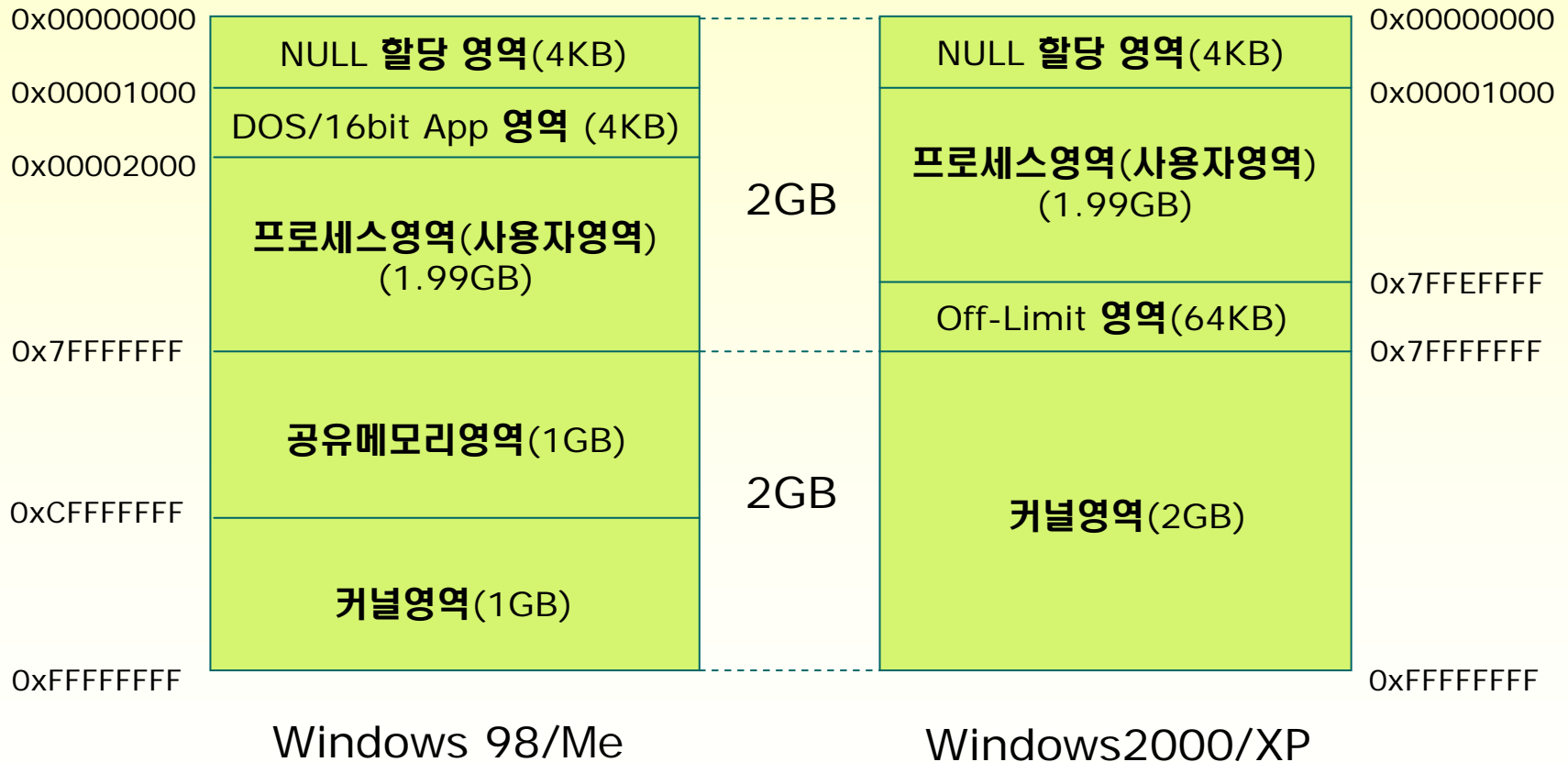
DEBUGGING

- Debugging == JOB of Finding BUGs!
- in MSVC++
 - F5 – Start Debug
 - F10 – Step Over
 - F11 – Step Into
- Debug Window
 - watch, variables, registers, memory, call stack, etc
- Debugging Tools
 - DEBUGVIEW (App)
 - OutputDebugString Win32API

예제

- DebuggingTest

32비트 윈도우 가상메모리 구조



4GB, Multi-Tasking Enable?

- PHYSICAL MEMORY/VIRTUAL MEMORY
- Virtual Memory Paging
- Memory-Mapping
- Scheduler
- Round-Robbin
- Context Switching(or Preemption)

The background features a central yellow rectangle with a thin black border. Surrounding this central area are several other colored rectangular blocks: a light green block at the top left, a light brown block at the top right, a dark green block on the right side, a light purple block at the bottom left, and a blue block at the bottom right. A thin black line runs horizontally across the top of the page, and another runs horizontally across the bottom, intersecting with vertical lines that define the layout of the colored blocks.

Fundamental

argc, argv

```
void main (int argc, char *argv[])  
{  
    for (int i= 0 ; i < argc ; ++i)  
    {  
        printf (“명령줄 %d = %s\n”, i,  
argv[i]) ;  
    }  
}
```

argc, argv 예제 in Linux

```
■ #include <stdio.h>
■ #include <stdlib.h>

■ int main(int argc, char** argv)
■ {
■     int i;
■     for(i=0; i<argc ; ++i)
■     {
■
■ printf("Commandline %d=%s\n",i,argv[i]);
■     }
■ }
```