__ptr32, __ptr64

Article • 08/03/2021

Microsoft Specific

`__ptr32` represents a native pointer on a 32-bit system, while `__ptr64` represents a native pointer on a 64-bit system.

The following example shows how to declare each of these pointer types:

C++

```
int * __ptr32 p32;
int * __ptr64 p64;
```

On a 32-bit system, a pointer declared with `__ptr64` is truncated to a 32-bit pointer. On a 64-bit system, a pointer declared with `__ptr32` is coerced to a 64-bit pointer.

> **Note**
>
> You cannot use `__ptr32` or `__ptr64` when compiling with **/clr:pure**. Otherwise, Compiler Error C2472 will be generated. The **/clr:pure** and **/clr:safe** compiler options are deprecated in Visual Studio 2015 and unsupported in Visual Studio 2017.

For compatibility with previous versions, **_ptr32** and **_ptr64** are synonyms for `__ptr32` and `__ptr64` unless compiler option /Za (Disable language extensions) is specified.

Example

The following example shows how to declare and allocate pointers with the `__ptr32` and `__ptr64` keywords.

C++

```
#include <cstdlib>
#include <iostream>

int main()
{
    using namespace std;

    int * __ptr32 p32;
    int * __ptr64 p64;

    p32 = (int * __ptr32)malloc(4);
    *p32 = 32;
    cout << *p32 << endl;

    p64 = (int * __ptr64)malloc(4);
    *p64 = 64;
    cout << *p64 << endl;
}
```

Output

```
32
64
```

END Microsoft Specific

See also

Built-in types

Feedback

Was this page helpful? Yes No

Provide product feedback | Get help at Microsoft Q&A