# DCMTK - Bug #1131

## Definition of ssize_t on Windows is incorrect

2024-07-22 12:25 - Marco Eichelberg

| | | | |
|---|---|---|---|
| **Status:** | Closed | **Start date:** | 2024-07-22 |
| **Priority:** | Normal | **Due date:** | |
| **Assignee:** | Marco Eichelberg | **% Done:** | 100% |
| **Category:** | | **Estimated time:** | 0:00 hour |
| **Target version:** | | | |
| **Module:** | | **Compiler:** | |
| **Operating System:** | | | |

**Description**

Currently, DCMTK defines ssize_t on platforms that do not define it themselves, such as Windows, like this:

```
/* Define `ssize_t' to `long' if <sys/types.h> does not define. */
#define HAVE_NO_TYPEDEF_SSIZE_T
#ifdef HAVE_NO_TYPEDEF_SSIZE_T
#define ssize_t long
#endif
```

On 64-Bit Windows, however, size_t is a 64 bit integer, so ssize_t should also be 64 bit.
Windows actually has a typedef for SSIZE_T (in uppercase), which is defined as:

```
#if defined(_WIN64)
 typedef __int64 ssize_t;
#else
 typedef long ssize_t;
#endif
```

A more comprehensive solution might be desirable to cover possible other 64-bit platforms where no ssize_t is defined. This may follow the definition of Sint64 in dcmtk/ofstd/oftypes.h, which however is also rather complex.

Reported 2024-07-15 by Helmut Steiner <helmut@shl.at>.

### History

**#1 - 2024-08-09 18:09 - Marco Eichelberg**

*- Status changed from New to Closed*

*- Assignee set to Marco Eichelberg*

*- % Done changed from 0 to 100*

Closed by commit #5ca58cb6e.