Subject: Re: Added Xlnt .xlsx library
Posted by forlano on Thu, 23 Apr 2026 11:07:10 GMT
View Forum Message <> Reply to Message

Hi Iñaki,

thanks for reply.
I had to do a fresh install of U++ and now it works. I do not know why it stopped to work.
While compiling my app I just noticed a compiler complain in package function4U (Clang under Windows)

C:\upp\UppHub\Functions4U/Functions4U/Defs.h (7): error: no matching function for call to 'isnan'

ChatGPT examined the code and said std::isnan<double>(n) does not require <double>.
I do not know if it is true (sometimes it invents things). Anyway it gives me the following version

```
#ifndef _Functions4U_Defs_h_
#define _Functions4U_Defs_h_

#include <cmath>
#include <complex>
#include <limits>

namespace Upp {

#ifdef PLATFORM_WIN32
inline bool IsNum(const double &n) { return !std::isnan(n) && !std::isinf(n) && !IsNull(n); }
inline bool IsNum(const float  &n) { return !std::isnan(n) && !std::isinf(n); }
#else
inline bool IsNum(const double &n) { return !__builtin_isnan(n) && !__builtin_isinf(n) && !IsNull(n);
}
inline bool IsNum(const float  &n) { return !__builtin_isnan(n) && !__builtin_isinf(n); }
#endif

inline bool IsNum(const int &n) { return !IsNull(n); }

template <typename T>
inline bool IsNum(const std::complex<T> &n) {
   return !(!IsNum(n.real()) || !IsNum(n.imag()));
}

template <typename T>
bool IsNull(const std::complex<T> &d) { return !IsNum(d); };

#define NaNComplex std::complex<double>(std::numeric_limits<double>::quiet_NaN(),
std::numeric_limits<double>::quiet_NaN())
#define NaNDouble  std::numeric_limits<double>::quiet_NaN()
```

```
template <typename T>
inline std::complex<T> i() { return std::complex<T>(0, 1); };

template <typename T>
inline bool IsNum(const Point_<T> &n) { return IsNum(n.x) && IsNum(n.y); }

}
#endif
```

and I was able to compile. I hope it will not backfire. Please have a look at it.

Thanks and best regards,
Luigi
