
Subject: Re: Value with type float
Posted by [mirek](#) on Tue, 10 May 2022 13:44:16 GMT
[View Forum Message](#) <> [Reply to Message](#)

Tom1 wrote on Tue, 10 May 2022 13:21: Hi Mirek,

Sorry to return to this subject, but could you consider adding Null support for float?

I had something like this in mind... (in Core/Defs.h):

```
const int   INT_NULL      = INT_MIN;
const int64 INT64_NULL    = INT64_MIN;
```

```
constexpr double DOUBLE_NULL = -std::numeric_limits<double>::infinity();
constexpr float  FLOAT_NULL  = -std::numeric_limits<float>::infinity();
```

```
class Nuller {
public:
    operator int() const          { return INT_NULL; }
    operator int64() const        { return INT64_NULL; }
    operator double() const       { return DOUBLE_NULL; }
    operator float() const        { return FLOAT_NULL; }
    operator bool() const         { return false; }
```

```
    Nuller() {}
};
```

```
extern const Nuller Null;
```

```
template <class T> void SetNull(T& x) { x = Null; }
```

```
template <class T> bool IsNull(const T& x)    { return x.IsNullInstance(); }
```

```
template<> inline bool IsNull(const int& i)    { return i == INT_NULL; }
template<> inline bool IsNull(const int64& i)  { return i == INT64_NULL; }
template<> inline bool IsNull(const double& r) { return !(std::abs(r) <
std::numeric_limits<double>::infinity()); }
template<> inline bool IsNull(const float& r)  { return !(std::abs(r) <
std::numeric_limits<float>::infinity()); }
template<> inline bool IsNull(const bool& r )  { return false; }
```

Although, I'm not entirely sure, if this is completely correct way to do it.

Best regards,

Tom

PS. EDIT: I think the above should work mostly. Only the "Cout() << FLOAT_NULL;" prints -inf

instead of empty field, which is the default for "Cout() << DOUBLE_NULL;":

```
CONSOLE_APP_MAIN{
double d=NULL;
float f=NULL;
double a=f;
float b=d;
Cout() << "d = " << d << "\n";
Cout() << "f = " << f << "\n";
Cout() << "a = " << a << "\n";
Cout() << "b = " << b << "\n";
Cout() << "a == f : " << (bool)(a==f) << "\n";
Cout() << "b == d : " << (bool)(b==d) << "\n";
Cout() << "IsNull(d) = " << IsNull(d) << "\n";
Cout() << "IsNull(f) = " << IsNull(f) << "\n";
Cout() << "IsNull(a) = " << IsNull(a) << "\n";
Cout() << "IsNull(b) = " << IsNull(b) << "\n";
}
```

IDK. Does it solve any real problem?

Up until this year, I have considered double-float relation to be similar to int-int16. You use float to reduce memory consumption or for special things (GPU), but you really do not need to support it in Value or widgets. Served me well for many many years. What has changed? :)

Mirek
