
Subject: Re: About Nuller and Null

Posted by [Tom1](#) on Sat, 10 Oct 2020 22:02:45 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi,

How about this? I did not benchmark the performance, but at least this is not relying on testing for a Null reference. As you can see, the 'Optional' is named in the foot steps of std::optional which is available in C++17 for the same purpose. (However, std::optional does not seem to support passing reference variables.)

```
#include <Core/Core.h>
```

```
using namespace Upp;
```

```
template <typename T>
struct Optional : public Tuple2<bool, T>{
    typedef Tuple2<bool, T> Base;
    Optional(T data) : Base(true, data) { }
    Optional() : Base(false, (T)Null) { }
    inline bool IsOK(){ return (bool)Base::a; }
    inline T Get(){ return (T)Base::b; }
};
```

```
class A{
public:
    int a;
    int b;
```

```
    A(){
        a=1;
        b=2;
    }
```

```
    String ToString() const { return String("A[") << a << ", " << b << "]" ; }
};
```

```
// Testing:
```

```
Array<A> av;
```

```
Optional<A&> GetA2(int x){
    if((x<0)|| (x>=av.GetCount())) return Optional<A&>();
    return Optional<A&>(av[x]);
}
```

```
CONSOLE_APP_MAIN{
    av.Add().a=1;
```

```
av.Add().a=2;  
av.Add().a=3;  
av.Add().a=4;  
  
for(int i=-1;i<6;i++){  
    Optional<A&> result=GetA2(i);  
    Cout() << (result.IsOK() ? AsString(result.Get()) : "Null") << "\n";  
}  
}
```

Best regards,

Tom
