Subject: Re: Make THISFN simpler and more powerful
Posted by Lance on Fri, 11 Nov 2022 01:50:42 GMT
View Forum Message <> Reply to Message

Problem solved with a hackish modification to the definition of the Moveable template in <Core/Topt.h>

```
template <class T, class B = EmptyClass>
struct Moveable : public B {
 typedef B MoveableBase; // Add this line
 friend void AssertMoveable0(T *) {}
};
```

Now it works like a charm :)

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

template <class T>
concept UppMoveable = std::is_trivial_v<T> || requires{
 typename T::MoveableBase;
 requires std::derived_from<T,
          Upp::Moveable<T, typename T::MoveableBase>
        >;
};

CONSOLE_APP_MAIN
{
 using namespace Upp;

 struct D{
  D(){}
 };

 struct F{
  F() = default;
 };

 struct E : Moveable<E, D>{
  E(){}
 };


 DUMP(UppMoveable<String>);//true
 DUMP(UppMoveable<D>);//false
 DUMP(UppMoveable<F>);//true, note the difference between D and F
```

```
 DUMP(UppMoveable<E>);//true
 DUMP(UppMoveable<int64>);//true
}
```

This is a hack. I am still interested to know if there is a way to do it without resorting to touch <Core/Topt.h>.
