Subject: Re: RPC_METHOD how to "define"
Posted by dolik.rce on Thu, 02 Feb 2017 16:14:15 GMT
View Forum Message <> Reply to Message

Well, the problem is that THISBACK can't be converted to function pointer as expected by Register. This can be worked around, but I had to made an one assumption: Only one instance of the class exists at a time. This is quite reasonable in RPC server, since otherwise you wouldn't know which instance to use when client calls one of the methods.

```
Here is a working (tested :)) code:#include <Core/Core.h>
#include <Core/Rpc/Rpc.h>
using namespace Upp;
class MyRpcClass {
  Time last;
public:
  typedef MyRpcClass CLASSNAME;
  void Status(RpcData& rpc) {
     rpc << last;
  void Ping(RpcData& rpc) {
    last = GetSysTime();
    rpc << last;
  }
  static void RegisterMethods() {
    // we use lambda to get a singleton instance of the class and call its method
    Register("Status", [](RpcData& rpc){Single<MyRpcClass>().Status(rpc);});
    Register("Ping", [](RpcData& rpc){Single<MyRpcClass>().Ping(rpc);});
};
INITBLOCK {
// calls all the registrations before main executes
MyRpcClass::RegisterMethods();
}
CONSOLE APP MAIN
Cout() << "Server...\n";
LogRpcRequests();
RpcServerLoop(1234);
```

It could be made slightly prettier with macros, but I didn't want to obfuscate the logic too much...

Honza