

---

Subject: Re: Flatbuffers package

Posted by [Xemuth](#) on Tue, 15 Dec 2020 11:50:10 GMT

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

---

Server flatbuffers example:

```
#include <Core/Core.h>
#include "flatbuffer-command/command_generated.h"

using namespace Upp;

//a simple Command struct
struct Command{
public:
    int id;
    String name;

    Command(int _id, String _name){id = _id; name = _name;}

    String ToString()const{
        String toStr;
        toStr << "Id: " << id << ", Name: " << name;
        return toStr;
    }
};

struct CommandSerializer{
public:
    CommandSerializer(const Command& cmd){
        flatbuffers::Offset<flatbuffers::String> str1 = builder.CreateString(cmd.name.ToStd()); //First, we
create a special flatbuffers object that handle string
        TcpIpServer::CommandBuilder commandBuilder(builder); //Then we create a commandBuilder
(a special object that will construct our Command (the fbs file))
        commandBuilder.add_id(cmd.id); //We define value of each parameters of our command
        commandBuilder.add_name(str1); //We define value of each parameters of our command
        flatbuffers::Offset<TcpIpServer::Command> freshCommand = commandBuilder.Finish(); //we
retrieve our command by telling our commandBuilder we are done with it
        builder.Finish(freshCommand); //we end the builder of our newly command (named
freshCommand)
    }

    int GetSize(){return builder.GetSize();} //Return a ptr to our serialized object
    uint8_t* GetDataptr(){return builder.GetBufferPointer();} //Return size of our serialized object

private:
    flatbuffers::FlatBufferBuilder builder; //Builder is used to create everything we want with
flatbuffers
};
```

```

Command CommandDeserializer(const uint8_t* datas, int size){
    flatbuffers::Verifier verifier(datas,size); //Verifier is used to verify our buffer of data is a valid
command
    if(TcpIpServer::VerifyCommandBuffer(verfier)){
        const TcpIpServer::Command* serializedCmd = TcpIpServer::GetCommand(datas); //If our
buffer is valid then we retrieve a ptr to the command within this buffer
        return Command(serializedCmd->id(), String(serializedCmd->name()->c_str())); //then we use
our buffer to build our command
    }
    throw Exc("Invalide command");
}

```

CONSOLE\_APP\_MAIN

```

{
    StdLogSetup(LOG_COUT|LOG_FILE);
    TcpSocket server;
    if(!server.Listen(3214, 5)) {
        LOG("Unable to initialize server socket!\n");
        SetExitCode(1);
        return;
    }
    LOG("Waiting for requests..");
    for(;;) {
        TcpSocket s;
        if(s.Accept(server)){

            String datas;
            dword sizeReceived;
            if(s.Get(&sizeReceived, sizeof(sizeReceived)))
                datas = s.Timeout(5000).Get(sizeReceived);
            try{
                LOG(CommandDeserializer((uint8_t*)datas.ToStd().c_str(),datas.GetLength()));

                CommandSerializer cmdValide(Command(0,"Valide"));
                dword sizeToSend = cmdValide.GetSize();
                if(s.Put(&sizeToSend,sizeof(sizeToSend)))
                    s.Timeout(5000).Put(cmdValide.GetDataspPtr(),cmdValide.GetSize());
            }catch(Exc& exception){
                CommandSerializer cmdInvalide(Command(-1,"Invalide"));
                dword sizeToSend = cmdInvalide.GetSize();
                if(s.Put(&sizeToSend,sizeof(sizeToSend)))
                    s.Timeout(5000).Put(cmdInvalide.GetDataspPtr(),cmdInvalide.GetSize());
            }
        }
    }
}

```

## File Attachments

---

1) [ServerFlatbuffers.7z](#), downloaded 359 times

---