#For Developers who want to custom serialize/deserialize | protocol,or intergrate with other network framework.

Custom Serialize/Deserialize

- Implement Decoder & Encoder interface;
- Register Your Decoder & Encoder implemenation to Coders;

```
Codecs.addEncoder(4,// your encoder class);
Codecs.addDecoder(4,// your decoder class);
```

- When u call AbstractClient.invokeSync,u can set codectype to 4,then framework will use your encoder & decoder.

Custom Protocol

- Implement Protocol interface,u can see RPCProtocol or SimpleProcessorProtocol to learn how to implement it;
- Implement ServerHandler interface,u can see RPCServerHandler or SimpleProcessorServerHandler to learn how to implement it;
- Register your protocol

```
ProtocolFactory.registerProtocol(// protocol type, // your protocol class, // your serverhandler class);
```

- ps: when u register server handler,u must set protocoltype,so the server handler will only service for the protocol.
- When u call AbstractClient.invokeSync or ClientInvocationHandler,u can set protocoltype to your type,then framework will use your protocol to handle.

Intergrate with other network framework

Server

- Implement Server Interface;
- When your server receive message,you can call ProtocolFactory.getServerHandler(protocolType).handleRequest(request) to handle the requestWrapper,pls see MinaServerHandler to learn more;

Serialize/Deserialize

- Implement ByteBufferWrapper Interface,pls see MinaByteBufferWrapper to learn more;
- In where u need serialize,u can call ProtocolUtils.encode(// receive object, // your ByteBufferWrapper),pls see MinaProtocolEncoder to learn more;
- In where u need deserialize,u can call ProtocolUtils.decode(// your

ByteBufferWrapper, // return value),pls see MinaProtocolDecoder to learn more;

Client

- Extend AbstractClient to implement sendRequest method,pls see MinaClient to learn more;
- Extend AbstractClientFactory to implement createClient method,pls see MinaClientFactory to learn more;
- If u need use rpc based on reflection,extend AbstractInvocationHandler to implement getClientFactory method,pls see MinaClientInvocationHandler to learn more;
- When client receive response,u only need to call client.putResponse;
- If client receive response more than one,u can call client.putResponses;

Benchmak

For Reflection RPC

- Extend AbstractRPCBenchmarkClient & AbstractBenchmarkServer;
- client startup args: -Dwrite.statistics=false serverIP serverPort concurrents timeout codectype requestSize runtime(seconds) clientNums
- server startup args: listenPort maxThreads responseSize
- welcome submit your code or benchmark results.

For Direct Call RPC

- Extend AbstractSimpleProcessorBenchmarkClient & AbstractBenchmarkServer;
- welcome submit your code or benchmark results.