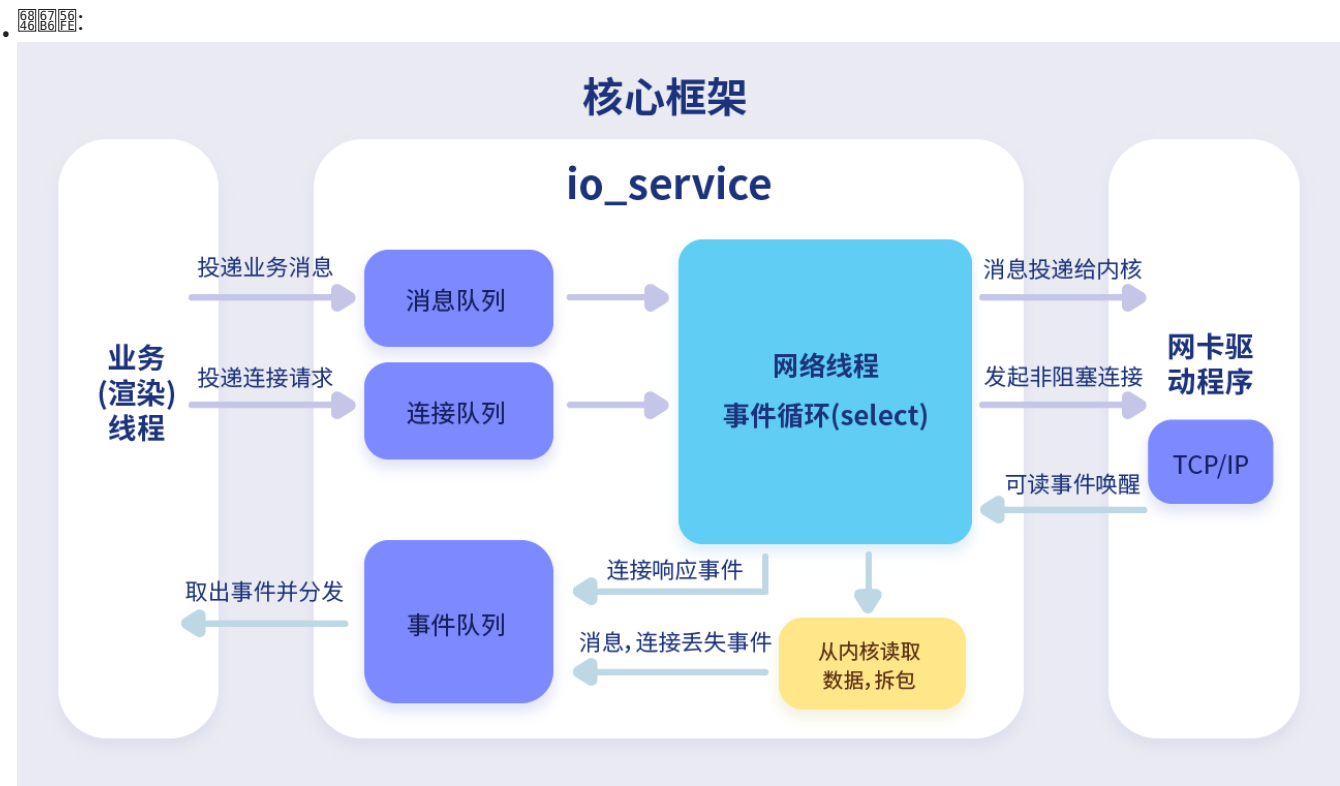


yasio `4E656568`
`2D878763`



yasio `2F4E4E8F917F8875F3765F6B` socket `5EFC13FC4E5B677A54574E54796B625E64766E625B677A7F7E6752B9`
`6B6B2A7B7FA78875F86A6265`

- `656B8B8A`:
 - [English](#)
 - `78534E65`
- `8875F367`:
 - `7E8B58`:
 - Visual Studio 2013+
 - GCC4.7+
 - xcode9+
 - `5A4E5F63` C++11,14,17 `787E8B58`
 - `8767`: x86, x64, ARM `7889`
 - `68EE7C7E`: Windows, macOS, Linux, FreeBSD, iOS, Android `7838`
- `6788`:
 - `51CE6D51`: `io_service`
 - `4E59`: `io_channel`
 - `46854A8B`: `io_transport`



5F905F59
EB1F00CB

68584E7A5F785354 tool.chinaz.com 5309http88425E5378C65458586289

C++

```
#include "yasio/yasio.hpp"
#include "yasio/obstream.hpp"
using namespace yasio;
using namespace yasio::inet;
int main()
{
    io_service service({"tool.chinaz.com", 80});
    service.set_option(YOPT_S_DEFERRED_EVENT, 0); // dispatch network event on network thread
    service.start([&](event_ptr&& ev) {
        switch (ev->kind())
        {
        case YEK_ON_PACKET: {
            auto packet = std::move(ev->packet());
            fwrite(packet.data(), packet.size(), 1, stdout);
            fflush(stdout);
            break;
        }
        case YEK_ON_OPEN:
            if (ev->status() == 0)
            {
                auto transport = ev->transport();
                if (ev->cindex() == 0)
                {
                    obstream obs;
                    obs.write_bytes("GET /index.htm HTTP/1.1\r\n");

                    obs.write_bytes("Host: tool.chinaz.com\r\n");

                    obs.write_bytes("User-Agent: Mozilla/5.0 (Windows NT 10.0; "
                                   "WOW64) AppleWebKit/537.36 (KHTML, like Gecko) "
                                   "Chrome/87.0.4820.88 Safari/537.36\r\n");
                    obs.write_bytes("Accept: */*;q=0.8\r\n");
                    obs.write_bytes("Connection: Close\r\n\r\n");

                    service.write(transport, std::move(obs.buffer()));
                }
            }
            break;
        case YEK_ON_CLOSE:
            printf("The connection is lost.\n");
            break;
        }
    });
    // open channel 0 as tcp client
    service.open(0, YCK_TCP_CLIENT);
    getchar();
}
```

Lua

```
local ip138 = "tool.chinaz.com"
local service = yasio.io_service.new({host=ip138, port=80})
local respdata = ""
```

```
service:start(function(ev)
    local k = ev.kind()
    if (k == yasio.YEK_ON_PACKET) then
        respdata = respdata .. ev:packet():to_string()
    elseif k == yasio.YEK_ON_OPEN then
        if ev:status() == 0 then -- connect succeed
            local transport = ev:transport()
            local obs = yasio.obstream.new()
            obs:write_bytes("GET / HTTP/1.1\r\n")

            obs:write_bytes("Host: " .. ip138 .. "\r\n")

            obs:write_bytes("User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/
537.36 (KHTML, like Gecko) Chrome/79.0.3945.117 Safari/537.36\r\n")
            obs:write_bytes("Accept: */*;q=0.8\r\n")
            obs:write_bytes("Connection: Close\r\n\r\n")

            service:write(transport, obs)
        end
    elseif k == yasio.YEK_ON_CLOSE then
        print("request finish, respdata: " .. respdata)
    end
end)

-- Open channel 0 as tcp client and start non-blocking tcp 3 times handshake
service:open(0, yasio.YCK_TCP_CLIENT)

-- Call this function at thread which focus on the network event.
function gDispatchNetworkEvent(...)
    service:dispatch(128) -- dispatch max events is 128 per frame
end

_G.yaservice = service -- Store service to global table as a singleton instance
```

6D8B & 794F
4B05 3A8B



8F88 Lua 794F7A5E9F4F62535F5E4F69 yasio - ibstream_view::consume out of range! FF8F69794F7A5E9169615176FF8B4E5F576139
6642 3A8B6B8F6C1A53766238E16E 6C692F3A8B6B8F6C456F69846CF76D65286F62

- 6D8B:
 - echo_server: TCP/UDP/KCP 6B646D2168
 - echo_client: TCP/UDP/KCP 6B64A9627E
 - ssltest: SSL 6B89A9627E, 6969github.com 4B985B6378846E626E
 - tcp test: TCP 4B897A6F

- [speedtest](#): TCP,UDP,KCP
- [mcast](#):
- [794F3A8B](#):
- [ftp_server](#): yasio ftp
- [lua](#): Lua http TCP
- [xlua](#): xlua
- [DemoUE4](#): UE4

7F8B 6D8B & 794F
16D1 4BD5 3A8B

- C++11 msvc , gcc , clang
- git , cmake installed
- :

```
git clone https://github.com/yasio/yasio
cd yasio
git submodule update --init --recursive
cd build
# for xcode should be: cmake .. -GXcode
cmake ..
cmake --build . --config Debug
```


io_service Class

```
socket.select tcp, udp, kcp, ssl-client
```

```
EDD5
```

```
namespace yasio { namespace inet { class io_service; } }
```

```
6254  
1058
```

```
515167905165  
6C718420FD70
```

Name	Description
io_service::io_service	io_service

```
6C516985
```

Name	Description
io_service::start	
io_service::stop	
io_service::open	
io_service::close	
io_service::is_open	
io_service::dispatch	
io_service::write	
io_service::write_to	DGRAM

Name	Description
io_service::schedule	
io_service::init_globals	
io_service::cleanup_globals	
io_service::channel_at	
io_service::set_option	

object_pool

yasio.hpp

io_service::io_service

io_service

```
io_service::io_service();  
  
io_service::io_service(int channel_count);  
  
io_service::io_service(const io_hostent& channel_ep);  
  
io_service::io_service(const io_hostent* channel_eps, int channel_count);
```

channel_count

channel_ep

channel_eps

4F698F2A5757657499575789

794F
3A8B

```
#include "yasio/yasio.hpp"
int main() {
    using namespace yasio;
    using namespace yasio::inet;
    io_service s1; // s1 only support 1 channel
    io_service s2(5); // s2 support 5 channels concurrency
    io_service s3(io_hostent{"github.com", 443}); // s3 support 1 channel
    io_hostent hosts[] = {
        {"192.168.1.66", 20336},
        {"192.168.1.88", 20337},
    };
    io_service s4(hosts, YASIO_ARRAYSIZE(hosts)); // s4 support 2 channels concurrency
    return 0;
}
```

io_service::start

54527F7C67577F7A82

```
void start(io_event_cb_t cb);
```

5365
2278

cb

7F7E464E6983FF6B8B49C9534E57 io_service::dispatch 8973897F7A895E82

794F
3A8B

```
#include "yasio/yasio.hpp"
int main() {
    using namespace yasio;
    using namespace yasio::inet;
    auto service = yasio_shared_service(io_hostent{host="ip138.com", port=80});
    service->start([](event_ptr&& ev) {
        auto kind = ev->kind();
        if (kind == YEK_ON_OPEN)
        {
            if (ev->status() == 0)
                printf("[%d] connect succeed.\n", ev->cindex());
            else
                printf("[%d] connect failed!\n", ev->cindex());
        }
    });
}
```

```
    return 0;
}
```

io_service::stop

59697f7e67527f7a39

```
void stop()
```

6c61
e80f

59697f7e67527f7a69528f88, 6951654f539990514f535e795f51685e995139
64fd791a616160fae1f77649857663880fa62

io_service::open

525f4f9939

```
void open(size_t cindex, int kind);
```

5365
c270

cindex

4f53295f39

kind

4f53788939

6c61
e80f

5948 TCP, 664af9425168529997985832f14b6267fa7aef978ef382

cindex 7850c598594eio_service52c953e3794f5378cf89

kind 5f98664f4e674f504e4e:

- YCK_TCP_CLIENT
- YCK_TCP_SERVER
- YCK_UDP_CLIENT
- YCK_UDP_SERVER
- YCK_KCP_CLIENT
- YCK_KCP_SERVER

- YCK_SSL_CLIENT

io_service::close

51254F89628946854A8B89

```
void close(transport_handle_t transport);  
void close(int cindex);
```

5365
C278

transport

5C895185764F8F4A8B89

cindex

5C895185764F8F89

6C61
E88F

EB8E TCP 6F 5C4A5389409E74662C5898E6389

io_service::is_open

52A04F89624F8F4A8B8954694E525F726989

```
bool is_open(transport_handle_t transport) const;  
bool is_open(int cindex) const;
```

5365
C278

transport

4F8F4F8B536789
20831A8B53C482

cindex

4F89205F89

8F5650
B4DE3C

true: 5355FF false: 2A5355

io_service::dispatch

52	6D	7F	7E	7E	7A	4E	75	76	4E	4E	30
06	3E	51	DC	BF	0B	A7	1F	84	8B	F6	02

```
void dispatch(int max_count);
```

53	65
C2	70

max_count

67	6B	67	59	52	6D	4E	4E	65	30
2C	21	00	27	06	3E	8B	F6	70	02

6C	61
E8	0F

1A5E68656C5E5F5751537F7E4E4E764E52908F7E7A8C75, 4F59Cocos2d-x766E677E7A8C4E53514E6E675F64(Unity,UE4)764E908F7E7A39

6B	65	6C	5B	4E	5B	51	57	66	65	6E	62	75	97	97	5E	67	75	30
64	B9	D5	F9	8E	89	68	30	F4	B0	38	0F	4C	62	5E	38	09	28	02

794F
3A8B

```
yasio_shared_service()->dispatch(128);
```

io_service::write

54	4F	8F	4F	88	8F	7A	53	90	65	63	30
11	20	93	1A	DD	DC	EF	D1	01	70	6E	02

```
int write(
    transport_handle_t thandle,
    std::vector<char> buffer,
    io_completion_cb_t completion_handler = nullptr
);
```

53	65
C2	70

thandle

4F	8F	4F	8B	53	67	30
20	93	1A	DD	E5	C4	02

buffer

89	53	90	76	4E	8F	52	7F	51	53	30
81	D1	01	84	8C	DB	36	13	82	3A	02

```
completion_handler
```

53	90	5B	62	56	8C	30
D1	01	8C	10	DE	03	02

8F5650
04DE3C

645F53897662598276, < 0 : F86F637F198F89

6C61
E88F

completion_handler 4F2F63 KCP 39

7Abuffer4A796388F5756F4E464A8A63 completion_handler 89

io_service::write_to

54UDP28834A886389766289

```
int write_to(  
    transport_handle_t thandle,  
    std::vector<char> buffer,  
    const ip::endpoint& to,  
    io_completion_cb_t completion_handler = nullptr  
);
```

5365
2270

thandle

4F834A88E3C289

buffer

89538976buffer89

to

895389768F7A575789

completion_handler

618988166E8389

8F5650
04DE3C

765F538976598276, 5F < 0 F86F6F637F198F8F1A582F26834A88F273E5(TCP8F63A86678)89

6C61
E88F

63F07845E7Z84E DGRAM 28834A886F73 UDP, KCP 39

538988166E89 completion_handler 4F2F63 KCP 39

buffer completion_handler

io_service::schedule

```
highp_timer_ptr schedule(  
    const std::chrono::microseconds& duration,  
    timer_cb_t cb  
);
```

duration

cb

std::shared_ptr

```
// Register a once timer, timeout is 3 seconds.  
yasio_shared_service()->schedule(std::chrono::seconds(3), []()->bool{  
    printf("time called!\n");  
    return true;  
});  
  
// Register a loop timer, interval is 5 seconds.  
auto loopTimer = yasio_shared_service()->schedule(std::chrono::seconds(5), []()->bool{  
    printf("time called!\n");  
    return false;  
});
```

io_service::init_globals

```
static void init_globals(print_fn2_t print_fn);
```

```
print_fn
```

81	5B	4E	7F	7E	65	5F	62	53	51	65	30
FA	9A	49	51	DC	E5	D7	53	70	FD	70	02

6C
E8 61
OF

64 5D 68 6E 6F 6F 68 63 68 63 78 6C 4F 6E 53 78 62 67 67 68 61 66 57 6C 63 57 61 5B 5A 51 53 2E 5A 4A 65 67 7B 7D 6E 6C 6C
52 97 5E 5B 69 78 6C 6B 6E 62 91 5A 5A 54 38 UE4 54 U3D 64 65 6F 6B 51 69

79	4F
3A	8B

```
// yasio_uelua.cpp
// compile with: /EHsc
#include "yasio_uelua.h"
#include "yasio/platform/yasio_ue4.hpp"
#include "lua.hpp"
#ifdef(NS_SLUA)
using namespace NS_SLUA;
#endif
#include "yasio/bindings/lyasio.cpp"

DECLARE_LOG_CATEGORY_EXTERN(yasio_ue4, Log, All);
DEFINE_LOG_CATEGORY(yasio_ue4);

void yasio_uelua_init(void* L)
{
    auto Ls = (lua_State*)L;
    print_fn2_t log_cb = [] (int level, const char* msg) {
        FString text(msg);
        const TCHAR* tstr = *text;
        UE_LOG(yasio_ue4, Log, L"%s", tstr);
    };
    io_service::init_globals(log_cb);

    luaregister_yasio(Ls);
}

void yasio_uelua_cleanup()
{
    io_service::cleanup_globals();
}
```


6C61
E80F

5378578081788F6354E15845E55F63786E8369A86958(.dll,.so)52E598837869F8656E8C9964787A5FE2A8869

io_service::channel_at

9A87EF83725F8762E159E36482

```
io_channel* channel_at(size_t cindex) const;
```

5365
C270

cindex

4F537265

8F5650
04DE3C

4F53E3646788, 55725F5880FA83F4F86C845E nullptr 82

io_service::set_option

887E889882

```
void set_option(int opt, ...);
```

5365
C270

opt

8898674E, 89E768 YOPT_X_XXX.

794F
3A8B

```
#include "yasio/yasio.hpp"

int main(){
    using namespace yasio;
    using namespace yasio::inet;
    io_hostent hosts[] = {
        {"192.168.1.66", 20336},
        {"192.168.1.88", 20337},
    };
    auto service = std::make_shared<io_service>(hosts, YASIO_ARRAYSIZE(hosts));
```

```
// for application protocol with length field, you just needs set this option.
// it's similar to java netty length frame based decode.
// such as when your protocol define as following
//     packet.header: (header.len=12bytes)
//         code:int16_t
//         datalen:int32_t (not contains packet.header.len)
//         timestamp:int32_t
//         crc16:int16_t
//     packet.data
service->set_option(YOPT_C_LFBFD_PARAMS,
    0,      // channelId, the channel index
    65535, // maxFrameLength, max packet size
    2,      // lenghtFieldOffset, the offset of length field
    4,      // lengthFieldLength, the size of length field, can be 1,2,4
    12,     // lengthAdjustment, if the value of length feild ==
packet.header.len + packet.data.len, this parameter should be 0, otherwise should be
sizeof(packet.header)
);

// for application protocol without length field, just sets length field size to -1.
// then io_service will dispatch any packet received from server immediately,
// such as http request, this is default behavior of channel.
service->set_option(YOPT_C_LFBFD_PARAMS, 1, 65535, -1, 0, 0);
return 0;
}
```

8B5396
F7C205

[io_event Class](#)

[io_channel Class](#)

[io_service Options](#)

[xxsocket Class](#)

[obstream Class](#)

[ibstream_view Class](#)

[ibstream Class](#)

io_channel Class

TCP/SSL/UDP/KCP

```
namespace yasio { namespace inet { class io_channel; } }
```

Name	Description
io_channel::get_service	io_service
io_channel::index	
io_channel::remote_port	
io_channel::bytes_transferred	
io_channel::connect_id	ID

io_service
io_service::channel_at

io_channel::get_service

io_service

```
io_service& get_service()
```

io_channel::index

87624f998f7a7ae3

```
int index() const
```

io_channel::remote_port

87624f998f7a7ae3.

```
u_short remote_port() const;
```

8f5650
04de3c

8f564f998f7a7ae3f7

- f98e2377a4f9988799041768f7a7ae3
- f98e6757a4f99887976547ae3

io_channel::bytes_transferred

87624f998f7a7ae3565982f8

```
long long bytes_transferred() const;
```

8f5650
04de3c

4e8e63fa7a5f58565520ff6336585982f86c784ef88a2377a636641c7

io_channel::connect_id

87624f998f7a7ae3565982f8id

```
unsigned int connect_id() const;
```

8f5650
04de3c

58677a4f9955208f63id



[io_service Class](#)

[io_event Class](#)

io_event Class

io_service

```
namespace yasio { namespace inet { class io_event; } }
```

Name	Description
io_event::kind	
io_event::status	
io_event::passive	
io_event::packet	
io_event::timestamp	
io_event::transport	
io_event::transport_id	ID
io_event::transport_udata	

io_event::kind

```
int kind() const;
```


8F	56	50
D4	DE	3C

```
4E 4E 7C 57 FF 53 4E 66 4E 4E 50
8B F6 7B 8B 0C EF E5 2F E5 0B 3C.
```

- YEK_ON_PACKET : 6D694E4E
886E687E
- YEK_ON_OPEN : 67686B6E6E6E686E69677A6E696E6E686E637665
- YEK_ON_CLOSE : 7372686E6E6E6E686E69677A6E696E6E686E637575

io_event::status

83	53	4E	4E	72	60	30
B7	D6	8B	F6	B6	01	02

```
int status() const;
```

8F	56	50
D4	DE	3C

- 0: $\begin{bmatrix} 68 & 5F \\ 63 & 38 \end{bmatrix}$
- $\begin{bmatrix} 97 & \\ 5E & \end{bmatrix}$ 0: $\begin{bmatrix} 51 & 95 \\ FA & 19 \end{bmatrix}, \begin{bmatrix} 75 & 62 & 53 & 97 & 89 & 78 & 53 & 62 & 53 & 53 & 30 \\ 28 & 37 & FA & 00 & 81 & 80 & 55 & 53 & 76 & 73 & EF & 02 \end{bmatrix}$

io_event::passive

68	67	66	54	66	88	52	4E	4E
C0	E5	2F	26	2F	AB	A8	8B	F6

```
int passive() const
```

8F	56	50
D4	DE	3C

- ```

* 0: 0218852484666632526774f1997968633635f19173e088f6542283f1a887968686f68f682
* 1: open52close6752584f59f3477f6f45553667669107f YASIO_ENABLE_PASSIVE_EVENT 624f4f7389
4f4f4f4f53515f8f6f6884f6886f51957889
14340d817cf898f63484f480a9f7f7f8d82

```

```
io_event::packet
```

|    |    |    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|----|----|
| 83 | 53 | 4E | 4E | 64 | 5E | 76 | 6D | 60 | 53 |
| 87 | D6 | 8B | F6 | 3A | 26 | 84 | 88 | 6F | 05 |

```
std::vector<char>& packet()
```

8F5650  
D4DE3C

886963241578, 286727454F73 std::move GC696F4E4E4E639080696389

## io\_event::timestamp

8363464E4F79785F7978F8F43380

```
highp_time_t timestamp() const;
```

8F5650  
D4DE3C

547C7E65956551795F7978F8F43380  
8C1E8D6F6F44E87384AE8D2A7F6F43382

## io\_event::transport

8363464E4F784F8F4A8B53C482

```
transport_handle_t transport() const;
```

8F5650  
D4DE3C

8F565367FF5F6552655F4E4E65FF4F8F4F8B5367505065FF4E53754F575750688F38  
D4DEE5C46C533638AD808B8BF6F60C20931A0D25C4F231486C5FF285C38403C048382

## io\_event::transport\_id

8363464E4F784F8F4A8B53C482ID80

```
unsigned int transport_id() const;
```

8F5650  
D4DE3C

324F4B482867747883F48565785F4EID80

## io\_event::transport\_udata

887E128363288F4A8B2857656280

```
template<typename _Uty>
_Uty io_event::transport_udata();
```

```
template<typename _Uty>
void io_event::transport_udata(_Uty uservalue);
```

6C61  
E80F

73576789E1F97874 userdata 7951E8, 6F59:

- 55506FA3FA7A62524E4E6558A8 userdata
- 52536FA325574E4E656574 userdata

8B5396  
F7C205

[io\\_service Class](#)

[io\\_channel Class](#)



# ostream Class

634f4f8b525f175252f0b9

- 3.35.0 basic\_ostream C++
- ostream int16~int64 float/double, ,
- fast\_ostream

8b6c  
ed05

```
namespace yasio {
using ostream = basic_ostream<endian::network_convert_tag>;
using fast_ostream = basic_ostream<endian::host_convert_tag>;
}
```



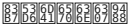


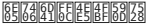
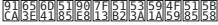
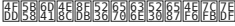
6254  
1058

515167905165  
6c718420fd70


| Name             | Description |
|------------------|-------------|
| ostream::ostream | ostream     |

5151656c  
6c718905

| Name                | Description              |
|---------------------|--------------------------|
| ostream::write      |                          |
| ostream::write_ix   | (7bit Encoded Int/Int64) |
| ostream::write_v    | (7bit Encoded Int)       |
| ostream::write_byte |                          |

| Name                                    | Description                                                                       |
|-----------------------------------------|-----------------------------------------------------------------------------------|
| <a href="#">obstream::write_bytes</a>   |  |
| <a href="#">obstream::empty</a>         |  |
| <a href="#">obstream::data</a>          |  |
| <a href="#">obstream::length</a>        |  |
| <a href="#">obstream::buffer</a>        |  |
| <a href="#">obstream::clear</a>         |  |
| <a href="#">obstream::shrink_to_fit</a> |  |
| <a href="#">obstream::save</a>          |  |



: obstream.hpp

## obstream::obstream

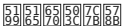
 obstream 

```
obstream(size_t capacity = 128);

obstream(const obstream& rhs);

obstream(obstream&& rhs);
```

## obstream::write



```
template<typename _Nty>
void obstream::write(_Nty value);
```

5365  
C270

*value*

8951517959  
8159636439

6C61  
E88F

\_Nty 5849795757456948671~8598263637957626979795739

## ostream::write\_ix

5632/644F6968594E7Bit Encoded Int656F887F6451516989

```
template<typename _Intty>
void ostream::write_ix(_Intty value);
```

5365  
C270

*value*

895151795989  
815963643982

6C61  
E88F

\_Intty 795753896959487957

- int32\_t
- int64\_t

64F0788879767869696F7C68AE8Fdotnet594E6165

- [BinaryWriter.Write7BitEncodedInt](#)
- [BinaryWriter.Write7BitEncodedInt64](#)

## ostream::write\_v

51514E8F5268626C6C63529F5E5788(7Bit Encoded Int).

```
void write_v(cxx17::string_view sv);
```

5365  
C270

sv

89515176596238  
81996364786282

6C61  
E88F

69F17851497Bit Encoded7678636F51516378627F26, 608972 write\_bytes 515159826363.

## ostream::write\_byte

515114A598238  
99632A578282

```
void write_byte(uint8_t value);
```

5365  
C270

value

895151765938  
819963643C82

6C61  
E88F

69F1785289784F4E ostream::write

## ostream::write\_bytes

51515982637C38  
99635982787C82

```
void write_bytes(cxx17::string_view sv);

void write_bytes(const void* data, int length);

void write_bytes(std::streamoff offset, const void* data, int length);
```

5365  
C270

sv

5151string\_view5388685982637C.

data

8951515982637C7474985757.  
8199635982787C7474985757.



*length*

8951515982657E7E79955E.  
8169655782787C647F8A.

*offset*

8951515982657E7E79685979.  
8169655782787C647E674FF8.

6C61  
E88F

offset + length 78505F985C4E  
645C5F786F6E ostream::length

ostream::empty

5268406F524E7A82  
24AD49263A7A82

```
bool empty() const;
```

8F5650  
04DE3C

true 7A; false 927A82

6C61  
E88F

695168784F4E length == 082  
F07849F76E

ostream::data

83536563639439  
87D8786E878882

```
const char* data() const;

char* data();
```

8F5650  
04DE3C

598260656388575739  
878261786388384082

ostream::length

8353609F5E82  
87D8417FA682

```
size_t length() const;
```

8F5650  
D4DE3C

8F56604F532476685982F539  
64664126632864685782F539

## ostream::buffer

8F56604F532476685982F539  
64664126632864685782F539

```
const std::vector<char>& buffer() const;

std::vector<char>& buffer();
```

8F5650  
D4DE3C

69642875632456236F22457F78 std::move 629069

794F  
3A8B

```
// ostream_buffer.cpp
// compile with: /EHsc
#include "yasio/ostream.hpp"

int main()
{
 using namespace yasio;
 using namespace cxx17;

 ostream obs;
 obs.write_v("hello world!");

 const auto& const_buffer = obs.buffer();

 // after this line, the obs will be empty
 auto move_buffer = std::move(obs.buffer());

 return 0;
}
```

## ostream::clear

6F7460FF4F4F587539  
6566416C4F4F587539

```
void clear();
```

6C61  
E80F

6BFD78651A2165buffer5158FF594E8A653759755F1753582758677583

## ostream::shrink\_to\_fit

2A524065E815623A1A55655882

```
void shrink_to_fit();
```

## ostream::save

5C6045764F8F52598278624F5852654F82

```
void save(const char* filename) const;
```

794F  
3A8B

```
// ostream_save.cpp
// compile with: /EHsc
#include "yasio/ostream.hpp"
#include "yasio/istream.hpp"

int main()
{
 using namespace yasio;
 using namespace cxx17;

 ostream obs;
 obs.write_v("hello world!");
 obs.save("ostream_save.bin");

 istream ibs;
 if(ibs.load("ostream_save.bin")) {
 // output should be: hello world!
 try {
 std::count << ibs.read_v() << "\n";
 }
 catch(const std::exception& ex) {
 std::count << "read_v fail: " <<
 << ex.message() << "\n";
 }
 }

 return 0;
}
```



[ibstream\\_view Class](#)

[ibstream Class](#)



# ibstream\_view Class

634f4f8b52725f57525f8032

6567

27c38f5716c77a4ef5524f63624b89f34792fa std::out\_of\_range 6568

886c  
ed05

```
namespace yasio {
 // 206f5752877a6c4f81528f6259825f6c8925a8516c4685
 using ibstream_view = basic_ibstream_view<endian::network_convert_tag>;

 // 206f5752877a6c4f81528f6259825f6c8925a8516c4685
 using fast_ibstream_view = basic_ibstream_view<endian::host_convert_tag>;
}
```

6254  
1058

62716728

| Name                         | Description                |
|------------------------------|----------------------------|
| ibstream_view::ibstream_view | 672812a ibstream_view 6961 |

5151656c  
6c718905

| Name                   | Description                                        |
|------------------------|----------------------------------------------------|
| ibstream_view::reset   | 917f5f235ff2139362<br>c6685c68f17169362            |
| ibstream_view::read    | 7d7821776c88637838                                 |
| ibstream_view::read_ix | f16321776c8863 (7bit Encoded Int/Int64) 737838     |
| ibstream_view::read_v  | 88625e772657 (7bit Encoded Int/Int64) 294e86327832 |

| Name                                      | Description                                                    |
|-------------------------------------------|----------------------------------------------------------------|
| <a href="#">ibstream_view::read_byte</a>  | <code>1145982</code>                                           |
| <a href="#">ibstream_view::read_bytes</a> | <code>8853635892584881526563<br/>F8D6879A7FA68C8B367862</code> |
| <a href="#">ibstream_view::empty</a>      | <code>2824415824517A</code>                                    |
| <a href="#">ibstream_view::data</a>       | <code>83634178626788</code>                                    |
| <a href="#">ibstream_view::length</a>     | <code>836341278F</code>                                        |
| <a href="#">ibstream_view::advance</a>    | <code>545278534078F8638869</code>                              |
| <a href="#">ibstream_view::seek</a>       | <code>78A84184F8626569</code>                                  |

`6C61  
E80F`

`ibstream_view 1924C++17892479 std::string_view, 8153275210585384235F17538F7A268D4FA7134E4E GC89`

`896C  
8142`

`34874E`: ibstream.hpp

## ibstream\_view::ibstream\_view

`6428662A` `ibstream_view` `588C38`

```
ibstream_view();

ibstream_view(const void* data, size_t size);

ibstream_view(const obstream* obs);
```

`5365  
C278`

*data*

`85285F575348885678628830575788`

*size*

5F C0 5F 17 53 40 8F 52 63 62 59 6C 80

*obs*

F0 8F 17 53 70 61 80

istream\_view::reset

21 7F istream\_view 15 62 80 50 80

```
void istream_view::reset(const void* data, size_t size);
```

53 65  
C2 70

*data*

5F C0 5F 17 53 40 8F 52 63 62 80 57 57 80

*size*

5F 53 5F 52 53 40 8F 52 63 59 5C 80  
85 C0 8F 17 16 8C 80 36 70 6E 27 6F 82

istream\_view::read

4E 60 40 80 53 63 50 80  
CE 41 20 80 00 70 5C 80

```
template<typename _Nty>
_Nty istream_view::read();
```

8F 50 50  
04 0E 3C

8F 50 80 52 70 50  
04 0E 80 30 64 30

6C 61  
E8 0F

\_Nty 50 80 70 80 52 4E 60 40 61 1~8 50 82 63 63 70 52 62 60 70 70 57 80

istream\_view::read\_ix

80 53 7Bit Encoded Int 53 20 16 01 70 53 60 80

```
template<typename _Intty>
_Intty istream_view::read_ix();
```



8F5650  
04DE3C

32/644F63785089

6C61  
E80F

\_Intty 795F985F4E4E7C57

- int32\_t
- int64\_t

7C7D787C8B4E Microsoft dotnet 894E7D78

- [BinaryReader.Read7BitEncodedInt\(\)](#)
- [BinaryReader.Read7BitEncodedInt64\(\)](#)

ibstream\_view::read\_v

8B5353974E8F52636282

```
cxx17::string_view read_v();
```

8F5650  
04DE3C

8F568B5352794E8F526363786E786E6E6E6E GC82

6C61  
E80F

7C7D787A4B8B53 7bit Encoded Int827576797897A658FF8B8978 read\_bytes 8B534E8F525982786282

ibstream\_view::read\_byte

8B5312A578282

```
uint8_t read_byte();
```

8F5650  
04DE3C

uint8\_t5082

6C61  
E80F

675163784F4E ibstream\_view::read

ibstream\_view::read\_bytes

FB62676A9F5A598263626C63 GC82

```
cxx17::string_view read_bytes();
```

8F5650  
D4DE3C

4F6B52636274 cxx17::string\_view 7B67636F62

ibstream\_view::empty

52636069544E7A82  
24AD412F263A7A82

```
bool empty() const;
```

8F5650  
D4DE3C

true 7A; false 604E8159535414F598262

6C61  
E80F

696363784F4E length == 082

ibstream\_view::data

8F56606363639482  
646E41786E678882

```
const char* data() const;
```

8F5650  
D4DE3C

8F6E6354694E28462A598276678882

## istream\_view::length

835260925E39

```
size_t length() const;
```

8F5659  
D4DE3C

5F5260925E39  
5340417FA682

## istream\_view::advance

545279526D8B536F6839  
1140FBA841F8D6386782

```
void advance(ptrdiff_t offset);
```

5365  
C278

*offset*

89545279527659799139  
811140FBA8644FEBCF82

6C61  
E86F

82\*offset\*4F8D65FF525364E379528B536F6839  
261F78419CB1E3FBA8FBD6386782

## istream\_view::seek

79528B536F6839  
FBA8FBA866386782

```
ptrdiff_t seek(ptrdiff_t offset, int whence);
```

5365  
C278

*offset*

84\*whence\*76517959799139  
8173844FEBCF82

*whence*

544E78644EC68F63674E59FF SEEK\_SET, SEEK\_CUR, SEEK\_END 82  
2849496C8E

8F5650  
04DE3C

8F5E78A85A78F84E6D885982597989

# ibstream Class

634F4E8B525962528F54536E5753528882

8B6C  
EDD5

```
namespace yasio {
using ibstream = basic_ibstream<endian::network_convert_tag>;
// F8F87F787A6CFF5F82FF883C6C7F8F8F8F
using fast_ibstream = basic_ibstream<endian::host_convert_tag>;
}
```

6254  
1058

515167885165  
6C718428FD78

| Name               | Description           |
|--------------------|-----------------------|
| ibstream::ibstream | 849818E ibstream F88C |

5151656C  
6C7189D5

| Name           | Description  |
|----------------|--------------|
| ibstream::load | FE68FE628F6D |

7E625C6B7E67  
E77F4221D384

ibstream\_view

ibstream

## ibstream::ibstream

6788464E ibstream F88C89

```
istream(std::vector<char> blob);

istream(const ostream* obs);
```

`blob`

`obs`

`ostream`

`load`

`load(const char* filename) const;`

`load`

`load(const char* filename) const;`

```
bool load(const char* filename) const;
```

`true`

`false`

`save`

`ostream::save`

`save`

`ostream Class`

`io_service Class`



# xxsocket Class

bsd socket API yasio

xxsocket accept\_n xxx\_n socket

```
namespace yasio { namespace inet { class xxsocket; } }
```

| Name               | Description |
|--------------------|-------------|
| xxsocket::xxsocket | xxsocket    |

| Name                  | Description |
|-----------------------|-------------|
| xxsocket::xpconnect   | TCP         |
| xxsocket::xpconnect_n | TCP         |
| xxsocket::pconnect    | TCP         |
| xxsocket::pconnect_n  | TCP         |
| xxsocket::pserve      | tcp         |
| xxsocket::swap        | socket      |
| xxsocket::open        | socket      |

| Name                                       | Description                                                                                |
|--------------------------------------------|--------------------------------------------------------------------------------------------|
| <a href="#">xxsocket::reopen</a>           |  socket   |
| <a href="#">xxsocket::is_open</a>          |  socket   |
| <a href="#">xxsocket::native_handle</a>    |  socket   |
| <a href="#">xxsocket::release_handle</a>   |  socket   |
| <a href="#">xxsocket::set_nonblocking</a>  |  socket   |
| <a href="#">xxsocket::test_nonblocking</a> |  socket   |
| <a href="#">xxsocket::bind</a>             |  socket   |
| <a href="#">xxsocket::bind_any</a>         |  socket   |
| <a href="#">xxsocket::listen</a>           |  TCP    |
| <a href="#">xxsocket::accept</a>           |  TCP    |
| <a href="#">xxsocket::accept_n</a>         |  TCP    |
| <a href="#">xxsocket::connect</a>          |  socket |
| <a href="#">xxsocket::connect_n</a>        |  socket |
| <a href="#">xxsocket::send</a>             |  socket |
| <a href="#">xxsocket::send_n</a>           |  socket |
| <a href="#">xxsocket::recv</a>             |  socket |
| <a href="#">xxsocket::recv_n</a>           |  socket |
| <a href="#">xxsocket::sendto</a>           |  DGRAM  |
| <a href="#">xxsocket::recvfrom</a>         |  DGRAM  |



| Name                                         | Description                                                                                       |
|----------------------------------------------|---------------------------------------------------------------------------------------------------|
| <a href="#">xxsocket::handle_write_ready</a> |  xxsocket        |
| <a href="#">xxsocket::handle_read_ready</a>  |  xxsocket        |
| <a href="#">xxsocket::local_endpoint</a>     |  xxsocket        |
| <a href="#">xxsocket::peer_endpoint</a>      |  xxsocket        |
| <a href="#">xxsocket::set_keepalive</a>      |  tcp keepalive   |
| <a href="#">xxsocket::reuse_address</a>      |  xxsocket        |
| <a href="#">xxsocket::exclusive_address</a>  |  xxsocket        |
| <a href="#">xxsocket::select</a>             |  xxsocket        |
| <a href="#">xxsocket::shutdown</a>           |  xxsocket      |
| <a href="#">xxsocket::close</a>              |  xxsocket      |
| <a href="#">xxsocket::tcp_rtt</a>            |  tcp rtt.      |
| <a href="#">xxsocket::get_last_errno</a>     |  xxsocket      |
| <a href="#">xxsocket::set_last_errno</a>     |  xxsocket      |
| <a href="#">xxsocket::strerror</a>           |  xxsocket      |
| <a href="#">xxsocket::gai_strerror</a>       |  getaddrinfo   |
| <a href="#">xxsocket::resolve</a>            |  xxsocket      |
| <a href="#">xxsocket::resolve_v4</a>         |  ipv4          |
| <a href="#">xxsocket::resolve_v6</a>         |  ipv6          |
| <a href="#">xxsocket::resolve_v4to6</a>      |  ipv4V4MAPPED |

| Name                             | Description      |
|----------------------------------|------------------|
| xxsocket::resolve_tov6           | ipv4V4MAPPEDipv6 |
| xxsocket::getipsv                | ip               |
| xxsocket::traverse_local_address |                  |

xxsocket::xxsocket

xxsocket

```
xxsocket::xxsocket();
xxsocket::xxsocket(socket_native_type handle);
xxsocket::xxsocket(xxsocket&& right);
xxsocket::xxsocket(int af, int type, int protocol);
```

handle

socketxxsocket

right

move

af

ip

protocol

TCP/UDP 0

xxsocket::xpconnect

TCP

```
int xxsocket::xpconnect(const char* hostname, u_short port, u_short local_port = 0);
```

hostname

IP 16 57 60 89

port

818f636752587a7af238

local\_port

7c5a9a4f7af23f7fc6b8a58 0 887a8f67529139

6c61  
e80f

4f8152686b2c67636178ip538f68727c8

8f5658  
b4be3c

0: 7252ff < 0 5129ff9a8f xxsocket::get\_last\_errno 875393ef7939

xxsocket::xpconnect\_n

8f8c7a675258fa7aTCP8f6389

```
int xxsocket::xpconnect_n(const char* hostname, u_short port, const std::chrono::microseconds&
wtimeout, u_short local_port = 0);
```

5365  
c270

hostname

818f636752587a7af23f7fc6b8a58 IP87a 16 5760 8

port

818f636752587a7af238

local\_port

7c5a9a4f7af23f7fc6b8a58 0 887a8f67529139

wtimeout

fa7a8f6389f8629539

8f5658  
b4be3c

0: 7252ff < 0 5129ff9a8f xxsocket::get\_last\_errno 875393ef7939

6c61  
e80f

4f8152686b2c67636178ip538f68727c8

## xxsocket::pconnect

54 8f 7a 67 57 58 ff 7a TCP 8f a3 39

```
int xxsocket::pconnect(const char* hostname, u_short port, u_short local_port = 0);
int xxsocket::pconnect(const endpoint& ep, u_short local_port = 0);
```

53 65  
c2 70

*hostname*

81 8f a3 60 57 58 4e 67 5a ff ef 4e 69 IP 57 57 57 57 00

*ep*

81 8f a3 60 57 58 7a 67 57 57 39

*port*

81 8f a3 60 57 58 7a 67 57 57 39

*local\_port*

67 57 99 4f 7a ff 53 ff 6c 08 a4 50 0 88 7a 8f 67 52 91 39

8f 56 50  
04 0e 3c

0: 63 52 ff < 0 51 29 6f 99 8f xxsocket::get\_last\_errno 87 53 93 ef 79 39

6c 61  
e8 0f

4f 4f c8 60 67 67 6f 61 79 ip 57 57 68 68

## xxsocket::pconnect\_n

54 8f 7a 67 57 58 ff 7a TCP 8f a3 39

```
int pconnect_n(const char* hostname, u_short port, const std::chrono::microseconds& wtimeout,
u_short local_port = 0);
int pconnect_n(const char* hostname, u_short port, u_short local_port = 0);
int pconnect_n(const endpoint& ep, const std::chrono::microseconds& wtimeout, u_short
local_port = 0);
int pconnect_n(const endpoint& ep, u_short local_port = 0);
```

53 65  
c2 70

*hostname*

81 8f a3 60 57 58 4e 67 5a ff ef 4e 69 IP 57 57 57 57 00

*ep*

818f63675258795757b9

*port*

818f63675258797af3b9

*local\_port*

6750904f7af3f76c6b8b50 0 88798f675291b9

*wtimeout*

fa7a8f6389f8f505b9

8f5650  
b4de3c

0: 1052ff < 0 51296c1a87 xxsocket::get\_last\_errno 8753938f79b9

xxsocket::pserve

5f546757TCP6752795430  
002f2c300da1d12c02

```
int pserve(const char* addr, u_short port);
int pserve(const endpoint& ep);
```

5365  
c270

*addr*

676a636a7f61 IP5a77 b9

*ep*

6767575739  
2c5a5040b2

*port*

TCP79547af3b9

8f5650  
b4de3c

0: 1052ff < 0 51296c1a87 xxsocket::get\_last\_errno 8753938f79b9

794f  
3a8b

```
// xxsocket-serve.cpp
#include <signal.h>
#include <vector>
#include "yasio/xxsocket.hpp"
```

```
using namespace yasio;
using namespace yasio::inet;

xxsocket g_server;
static bool g_stopped = false;
void process_exit(int sig)
{
 if (sig == SIGINT)
 {
 g_stopped = true;
 g_server.close();
 }
 printf("exit");
}
int main()
{
 signal(SIGINT, process_exit);

 if (g_server.pserve("0.0.0.0", 1219) != 0)
 return -1;
 const char reply_msg[] = "hi, I'm server\n";
 do
 {
 xxsocket cs = g_server.accept();
 if (cs.is_open())
 {
 cs.send(reply_msg, sizeof(reply_msg) - 1);
 std::this_thread::sleep_for(std::chrono::milliseconds(100));
 }
 } while (!g_stopped);

 return 0;
}
```

## xxsocket::swap

 socket

```
xxsocket& swap(xxsocket& who);
```



*who*





xxsocket 

## xxsocket::open

xxsocket

```
bool open(int af = AF_INET, int type = SOCK_STREAM, int protocol = 0);
```

*af*

AF\_INET (ipv4) AF\_INET6 (ipv6)

*type*

SOCK\_STREAM (TCP), SOCK\_DGRAM (UDP)

*protocol*

TCP/UDP 0

true: false xxsocket::get\_last\_errno

## xxsocket::reopen

xxsocket

```
bool reopen(int af = AF_INET, int type = SOCK_STREAM, int protocol = 0);
```

*af*

AF\_INET (ipv4) AF\_INET6 (ipv6)

*type*

SOCK\_STREAM (TCP), SOCK\_DGRAM (UDP)

*protocol*

TCP/UDP 0

true: false xxsocket::get\_last\_errno

6C61  
E80F

8267socket50635F6F6851654F5151F56F6B2168636F82

## xxsocket::is\_open

52A8socket2F24F2536682

```
bool is_open() const;
```

8F5650  
D4DE3C

true: 2953666F false: 5A636682

## xxsocket::native\_handle

8368socket654F6F6F7882

```
socket_native_type native_handle() const;
```

8F5650  
D4DE3C

socket6748636F786F yasio::inet::invalid\_socket 887A2868socket82

## xxsocket::release\_handle

21636555socket6F6F78A7564362

```
socket_native_type release_handle() const;
```

8F5650  
D4DE3C

21635276socket6748636F78

## xxsocket::set\_nonblocking

8876socket74827885825A6F82

```
int set_nonblocking(bool nonblocking) const;
```



5365  
C270

*nonblocking*

true: 9798586A5F6F false: 9858215F62

8F5650  
04DE3C

0: 6262FF < 0 5129FFA987 xxsocket::get\_last\_errno 875393287932

xxsocket::test\_nonblocking

6840socket2F544E9798586A5F62

```
int test_nonblocking() const;
```

8F5650  
04DE3C

1: 9798586A5F6F 0: 9858215F62

6C61  
E80F

6840winsock2FC6A8FA374 SOCK\_STREAM 7057socket1A845E -1 32

xxsocket::bind

7F5Asocket676767575739

```
int bind(const char* addr, unsigned short port) const;
int bind(const endpoint& ep) const;
```

5365  
C270

*addr*

676763587F53ip575739

*port*

817158737AFA39

*ep*

897158737A575739

8F5650  
04DE3C

0: 1052FF < 0 5129FF9A07 xxsocket::get\_last\_errno 875393EF7039

## xxsocket::bind\_any

7F5Asocket276A4B61506089

```
int bind_any(bool ipv6) const;
```

5365  
C270

*ipv6*

6F547F5B67674E61IPv65767  
2F26616A2C3A4B0F

8F5650  
04DE3C

0: 1052FF < 0 5129FF9A07 xxsocket::get\_last\_errno 875393EF7039

## xxsocket::listen

6F2B705267EA TCP69577A63E746F96589

```
int listen(int backlog = SOMAXCONN) const;
```

5365  
C270

*backlog*

6F5970546580  
8027612C7080

8F5650  
04DE3C

0: 1052FF < 0 5129FF9A07 xxsocket::get\_last\_errno 875393EF7039

## xxsocket::accept

6353464F5B627A8F6380  
A307602A5237E88F6380

```
xxsocket accept() const;
```

8F5650  
04DE3C

5458627A984F78 xxsocket FB8C39

## xxsocket::accept\_n

878858636F6A57664F58577A8E6389

```
int accept_n(socket_native_type& new_sock) const;
```

5365  
C278

*new\_sock*

83FAE2686C6458627A984F786554socketE3C455Z8

8F5650  
04DE3C

0: 6252FF < 0 5928FF98CF xxsocket::get\_last\_errno 8763938F7889

6C61  
E88F

826768F7658F58 0 6Fnew\_sock 4FA8888F6F4A2798586A6F82

897868F76548586CF8F6A588378 xxsocket::set\_nonblocking 66socket887E4A2798586A6F82

## xxsocket::connect

FA7A8E6389

```
int connect(const char* addr, u_short port);
int connect(const endpoint& ep);
```

5365  
C278

*addr*

8F7A4867ip575789

*port*

8F7A48677A5389

*ep*

8F7A4867575789

8F5650  
04DE3C

0: f252ff < 0 5929ff1a2f xxsocket::get\_last\_errno 8753932ff7839

6061  
e88f

TCP: 5189TCP 4f68e362

UDP: f57a43375758

## xxsocket::connect\_n

f57a8fa339  
fa7a8fa339

```
int connect_n(const char* addr, u_short port, const std::chrono::microseconds& wtimeout);
int connect_n(const endpoint& ep, const std::chrono::microseconds& wtimeout);
int connect_n(const endpoint& ep);
```

5365  
c298

*addr*

8f7a4667575739  
8f7a4667575739

*port*

8f7a46677a5339  
8f7a46677a5339

*ep*

8f7a4667575739  
8f7a4667575739

*wtimeout*

f57a8fa37a89f6f5f539  
f57a8fa37a89f6f5f539

8F5650  
04DE3C

0: f252ff < 0 5929ff1a2f xxsocket::get\_last\_errno 8753932ff7839

6061  
e88f

TCP: 5189TCP 4f68e362

UDP: f57a43375758

## xxsocket::connect\_n

xxsocket::connect\_n

```
int disconnect() const;
```

xxsocket::connect\_n

```
if (errno < 0) xxsocket::get_last_errno
```

xxsocket::connect\_n

SOCK\_DGRAM (UDP) socket

## xxsocket::send

xxsocket::send

```
int send(const void* buf, int len, int flags = 0);
```

xxsocket::send

buf

xxsocket::send

len

xxsocket::send

flags

xxsocket::send

xxsocket::send

```
if (len < 0) xxsocket::get_last_errno
```

## xxsocket::send\_n

xxsocket::send\_n

```
int send_n(const void* buf, int len, const std::chrono::microseconds& wtimeout, int flags = 0);
```

5365  
C270

*buf*

89518863627680595982575739

*len*

895188636276955E39

*wtimeout*

618188F6F6F482

*flags*

618178625542678839

8F5650  
04DE3C

```
==len: 6252FF < len 598DFF9A8F xxsocket::get_last_errno 835395887839
```

xxsocket::recv

AE516863298F7A4E6751888F67769656239

```
int recv(void* buf, int len, int flags = 0) const;
```

5365  
C270

*buf*

A352786275B25A39

*len*

A352786275B25A725E39

*flags*

A35278625542678839

8F5650  
04DE3C

```
==len: 6252FF < len 598DFF9A8F xxsocket::get_last_errno 835395887839
```

6C61  
E88F

68F17869547A538F59FF53514Esocket678E695469 8288589A5E 39

## xxsocket::recv\_n

5480F2F2F2A1566A4E846862FA636A7F56766289

```
int recv_n(void* buf, int len, const std::chrono::microseconds& wtimeout, int flags = 0)
const;
```

5365  
C270

*buf*

6369696275715289

*len*

636969627571527F5689

*wtimeout*

636980F2F2F2F289

*flags*

636969635F5C688889

8F5050  
04DE3C

```
==len: 6262FF < len 59896C7A8F xxsocket::get_last_errno 836399887889
```

## xxsocket::sendto

54807A4E675189 DGRAM 68UDP69696289

```
int sendto(const void* buf, int len, const endpoint& to, int flags = 0) const;
```

5369  
C270

*buf*

65618969627571528A89

*len*

65618969627571527F5689

*to*

61817869574689

*flags*

618178625542698889

8F5650  
04DE3C

```
==len: 6252FF < len 5989FF99FF xxsocket::get_last_errno 835395887839
```

## xxsocket::recvfrom

4E815888248C7A4E675189276374986282

```
int recvfrom(void* buf, int len, endpoint& peer, int flags = 0) const;
```

5365  
C278

*buf*

A358788215B25A7F2682

*len*

A358788215B25A7F2682

*peer*

A35878826468FF85FA237889

*flags*

A35878825E5C688882

8F5650  
04DE3C

```
==len: 6252FF < len 5989FF99FF xxsocket::get_last_errno 835395887839
```

6C61  
E88F

68215859547A538458FF52514Esocket6788585468 828858255F 82

## xxsocket::handle\_write\_ready

785Fsocket535182

```
int handle_write_ready(const std::chrono::microseconds& wtimeout) const;
```

5365  
C278

*wtimeout*

785F88F8F8F82



8F5650  
04DE3C

0: 80F3FF 1: F652FF < 0: 5189FF9A8F xxsocket::get\_last\_errno 876393887989

6C61  
E80F

9A585F516861897F6252A16F79C95146FF69F3784A7A53846889

## xxsocket::handle\_read\_ready

7865socket278889

```
int handle_read_ready(const std::chrono::microseconds& wtimeout) const;
```

5365  
C270

*wtimeout*

7865806565F389

8F5650  
04DE3C

0: 80F3FF 1: F652FF < 0: 5189FF9A8F xxsocket::get\_last\_errno 876393887989

## xxsocket::local\_endpoint

8763453741904F796757505089

```
endpoint local_endpoint() const;
```

8F5650  
04DE3C

8458675757575089

6C61  
E80F

59676A6789788F xxsocket::connect 1688TCP8E633281E1626768626E9945845879575028 0.0.0.0

## xxsocket::peer\_endpoint

8763453741904F79677A505089

```
endpoint peer_endpoint() const;
```

8F5650  
04DE3C

8F566257575739  
04DE2C30304082

6C61  
E88F

5962666969728F xxsocket::connect 1689 TCP8E63369E31626268626E99458F567957572F 0.0.0.0

## xxsocket::set\_keepalive

8B7FTCP5E52518B79C5F9C27682

```
int set_keepalive(int flag = 1, int idle = 7200, int interval = 75, int probes = 10);
```

5365  
C278

*flag*

1 : 55546555524F8B79C5F9FF 0 : 71E582

*idle*

5351725C6967464F6D60454E54FF54525E5C538B79C5F9A24B7967598D6868F46C534DFF79FF39

*interval*

5351626552C5F96654F86F6B596189C5F9A24B62F8F4F4846C534DFF68B26982

*probes*

5351626552C5F96654F86F6B596189C5F9A24B62F8F4F4846C534DFF68B26982

8F5650  
04DE3C

0 : 188F6F < 0 5129FF9A8F xxsocket::get\_last\_errno 8768938B7982

794F  
3A8B

```
// xxsocket-keepalive.cpp
#include "yasio/xxsocket.hpp"
using namespace yasio;
using namespace inet;

int main(){
 xxsocket client;
 if(0 == client.pconnect("192.168.1.19", 80)) {
```

```
 client.set_keepalive(1, 5, 10, 2);
 }
 return 0;
}
```

## xxsocket::reuse\_address

xxsocket::reuse\_address

```
void reuse_address(bool reuse);
```

xxsocket::reuse\_address

reuse

xxsocket::reuse\_address

xxsocket::reuse\_address

xxsocket::reuse\_address

## xxsocket::exclusive\_address

xxsocket::exclusive\_address

```
void exclusive_address(bool exclusive);
```

xxsocket::exclusive\_address

exclusive

xxsocket::exclusive\_address

xxsocket::exclusive\_address

xxsocket::exclusive\_address

## xxsocket::set\_optval

xxsocket::set\_optval

```
template <typename _Ty> int set_optval(int level, int optname, const _Ty& optval);
```

5365  
C270

*level*

socket899979A72B39

*optname*

8999797C5739

*optval*

89995039

8F5650  
04DE3C

0: 1252FF < 0 5129FF1A2F xxsocket::get\_last\_errno 8753958F7839

6C61  
E80F

69F07854bsd socket setsockopt 528878546FE32F4F282A7F51886CF4694F4F7539

xxsocket::get\_optval

887Esocket899939

```
template <typename _Ty> _Ty get_optval(int level, int optname) const
```

5365  
C270

*level*

socket899979A72B39

*optname*

8999797C5739

8F5650  
04DE3C

8A5E89795039

6C61  
E80F

69F07854bsd socket getsockopt 528878546FE32F4F282A7F51886CF4694F4F7539

## xxsocket::select

7952socket846848484839

```
int select(fd_set* readfds, fd_set* writefds, fd_set* exceptfds, const
std::chrono::microseconds& wtimeout)
```

5365  
C270

*readfds*

F2F84848F2F878657439

*writefds*

F2514848F2F878657439

*exceptfds*

55584848F2F878657439

*wtimeout*

7851484889F8484839

8F5650  
D4DE3C

0:80F2FF > 0:6252FF < 0:5989FF998F xxsocket::get\_last\_errno 875893887839

## xxsocket::shutdown

73E8TCP488598939

```
int shutdown(int how = SD_BOTH) const;
```

5365  
C270

*how*

73E898997888FFEF284848674850

- SD\_SEND : 51899899
- SD\_RECEIVE : 63639899
- SD\_BOTH : 88E873E8

8F5650  
04DE3C

0: 1852FF < 0: 5129FF992F xxsocket::get\_last\_errno 8762932F7932

## xxsocket::close

732Fsocket6F2A62767F446662

```
void close(int shut_how = SD_BOTH);
```

5365  
C270

### *shut\_how*

732F1A937B576F6F2F454B4545674E52

- SD\_SEND : 53899899
- SD\_RECEIVE : 63539999
- SD\_BOTH : 71282325
- SD\_NONE : 58287325

8F5650  
04DE3C

0: 1852FF < 0: 5129FF992F xxsocket::get\_last\_errno 8762932F7932

6C61  
E80F

5267 shut\_how != SD\_NONE 6C62F07B1A488328 shutdown 6C808C786522 close 62

## xxsocket::tcp\_rtt

8752TCP8F6379RTT89

```
uint32_t tcp_rtt() const;
```

8F5650  
04DE3C

8458TCP79RTT65F36C634F: 5E79 89

## xxsocket::get\_last\_errno

xxsocket::get\_last\_errno

```
static int get_last_errno();
```

xxsocket::get\_last\_errno

```
0: errno > 0 ? xxsocket::strerror(errno) : 0;
```

xxsocket::get\_last\_errno

xxsocket::get\_last\_errno

## xxsocket::set\_last\_errno

xxsocket::set\_last\_errno

```
static void set_last_errno(int error);
```

xxsocket::set\_last\_errno

error

xxsocket::set\_last\_errno

xxsocket::set\_last\_errno

xxsocket::set\_last\_errno

## xxsocket::not\_send\_error

xxsocket::not\_send\_error

```
static bool not_send_error(int error);
```

xxsocket::not\_send\_error

error

xxsocket::not\_send\_error

8F5650  
04DE3C

true : socket69586C false : socket7260F27E6179938BFC655373E6socket7E6299AF62

6C61  
E80F

4E536389644F8F6E50 < 0 F6FC837869F07832

xxsocket::not\_recv\_error

7260F27E6179938BFC655373E6socket7E6299AF62

```
static bool not_recv_error(int error);
```

5365  
C270

error

7260F27E6179938BFC655373E6

8F5650  
04DE3C

true : socket69586C false : socket7260F27E6179938BFC655373E6socket7E6299AF62

6C61  
E80F

4E536389644F8F6E50 < 0 F6FC837869F07832

xxsocket::strerror

5C958B788F634E597B4E30  
8619EE616C623A57263262

```
static const char* strerror(int error);
```

5365  
C270

error

7260F27E6179938BFC655373E6

8F5650  
04DE3C

858BFF697659784E30  
19EEFF616457263262



## xxsocket::gai\_strerror

`getaddrinfo`

```
static const char* gai_strerror(int error);
```

`error`

*error*

`error`

`error`

`error`

## xxsocket::resolve

`resolve`

```
int resolve(std::vector<endpoint>& endpoints, const char* hostname, unsigned short port = 0,
 int socktype = SOCK_STREAM);
```

`endpoints`

*endpoints*

`endpoints`

*hostname*

`hostname`

*port*

`port`

*socktype*

`socktype`

`socktype`

`0: 0 > 0 0 xxsocket::strerror 0`

## xxsocket::resolve\_v4

IPv4

```
int resolve_v4(std::vector<endpoint>& endpoints, const char* hostname, unsigned short port = 0, int socktype = SOCK_STREAM);
```

*endpoints*

*hostname*

*port*

*socktype*

*socket*

0: 0 > 0 1A 8F xxsocket::strerror 8C 63 4A E8 7E 19 E8 4F 69 39

## xxsocket::resolve\_v6

IPv6

```
int resolve_v6(std::vector<endpoint>& endpoints, const char* hostname, unsigned short port = 0, int socktype = SOCK_STREAM);
```

*endpoints*

*hostname*

*port*

*socktype*

socket7957397b8b62

8f5650d4de3c

0: 636366ff > 0 9a2f xxsocket::strerror 8f634a8876f9884f6939

## xxsocket::resolve\_v4to6

4f89675754535476c5e3606f67652b24IPv457575f8f634aIPv6 V4MAPPED685f393c6f62

```
int resolve_v4to6(std::vector<endpoint>& endpoints, const char* hostname, unsigned short port
= 0, int socktype = SOCK_STREAM);
```

5365c270

*endpoints*

8f5153653983faC27062

*hostname*

5754396f6062

*port*

7af339ef

*socktype*

socket7957397b8b62

8f5650d4de3c

0: 636366ff > 0 9a2f xxsocket::strerror 8f634a8876f9884f6939

## xxsocket::resolve\_tov6

89676760632b64606757576fc5e3606f67652b24IPv457574f8f634aIPv6 V4MAPPED685f393c6f62

```
int resolve_tov6(std::vector<endpoint>& endpoints, const char* hostname, unsigned short port =
0, int socktype = SOCK_STREAM);
```

|    |    |
|----|----|
| 53 | 65 |
| C2 | 70 |

*endpoints*

|    |    |    |    |    |
|----|----|----|----|----|
| 8F | 51 | 53 | 65 | 30 |
| 93 | FA | C2 | 70 | 02 |

*hostname*

|    |    |    |
|----|----|----|
| 57 | 54 | 30 |
| DF | 0D | 02 |

*port*

|    |    |    |
|----|----|----|
| 7A | 53 | 30 |
| EF | E3 | 02 |

socktype

socket<sup>7C</sup><sub>7B</sub><sup>57</sup><sub>8B</sub><sup>30</sup><sub>02</sub>

|    |    |    |
|----|----|----|
| 8F | 56 | 50 |
| D4 | DE | 3C |

```
0: 65 05 8B FF > 0 90 8F xxsocket::strerror 8F 63 4E 8B 7E 95 8B 4F 60 30
E0 19 EF 0C 1A C7 6C 62 3A E6 C6 19 EF E1 6F 02
```

## xxsocket::getipsv

83 53 67 67 65 63 76 IP 53 88 68 68 5F 4F 30  
B7 D6 2C 3A 2F 01 84 4F AE 08 07 D7 4D 02

```
static int getipsv();
```

|    |    |    |
|----|----|----|
| 8F | 56 | 50 |
| D4 | DE | 3C |

- `ipsv_ipv4`: 67 67 63 63 | IPv4 47 88 89
- `ipsv_ipv6`: 67 67 63 63 | IPv6 47 88 89
- `ipsv_dual_stack`: 67 67 63 63 | IPv4 47 | IPv6 77 68 43 88 89

|    |    |
|----|----|
| 6C | 61 |
| E8 | 0F |

5F 8F 56 50 65 63 53 68 53 8B 66 FF 75 62 5E 5F 59 7E 4F 51 4F 75 IPv4 90 4F FF  
53 D4 DE 3C 2F 01 CC 08 4F AE 2F 0C 28 37 94 53 CB C8 18 48 7F 28 1A E1 0C

4f 59 66 80 62 67 88 59 57 54 65 5f 54 wifi 54 87 7a 7f 7e 65 ff 5c 4f 4f 51 90 62  
8b 82 7a fd 48 3a be 07 28 0c f6 00 2f 8c 02 9d 51 dc f6 0c 06 1a 18 48 09 e9 wifi ff

 <https://github.com/halx99/yasio/issues/130>

|    |    |
|----|----|
| 79 | 4F |
| 3A | 8B |

```
// xxsocket-ipsv.cpp
#include <vector>
#include "yasio/xxsocket.hpp"
using namespace yasio;
using namespace yasio::inet;
```

```
int main(){
 const char* host = "github.com";
 std::vector<ip::endpoint> eps;
 int flags = xxsocket::get_ipsv();
 if(flags & ipsv_ipv4) {
 xxsocket::resolve_v4(eps, host, 80);
 }
 else if(flags & ipsv_ipv6) {
 xxsocket::resolve_tov6(eps, host, 80);
 }
 else {
 std::cerr << "Local network not available!\n";
 }
 return 0;
}
```

## xxsocket::traverse\_local\_address

674E6767575769  
9A3E676A564069

```
static void traverse_local_address(std::function<bool(const ip::endpoint&)> handler);
```

5365  
C270

### handler

674E5757568C39  
9A3E30400E0362

794F  
3A8B

```
// xxsocket-traverse.cpp
#include <vector>
#include "yasio/xxsocket.hpp"
using namespace yasio;
using namespace yasio::inet;
int main(){
 int flags = 0;
 xxsocket::traverse_local_address([&](const ip::endpoint& ep) -> bool {
 switch (ep.af())
 {
 case AF_INET:
 flags |= ipsv_ipv4;
 break;
 case AF_INET6:
 flags |= ipsv_ipv6;
 break;
 }
 return (flags == ipsv_dual_stack);
 });
 YASIO_LOG("Supported ip stack flags=%d", flags);
}
```

```
 return flags;
}
```

|    |    |    |
|----|----|----|
| 8B | 53 | 96 |
| F7 | C2 | 05 |

io\_service Class



## io\_service options

io\_service

| Name                         | Description                                                                                                                                                                                                                                                                                    |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>YOPT_S_DEFER_EVENT_CB</i> | Set defer event callback<br>params: callback:defer_event_cb_t<br>remarks:<br>a. User can do custom packet resolve at network thread, such as decompress and crc check.<br>b. Return true, io_service will continue enqueue to event queue.<br>c. Return false, io_service will drop the event. |
| <i>YOPT_S_DEFERRED_EVENT</i> | Set whether deferred dispatch event, default is: 1<br>params: deferred_event:int(1)                                                                                                                                                                                                            |
| <i>YOPT_S_RESOLV_FN</i>      | Set custom resolve function, native C++ ONLY<br>params: func:resolv_fn_t*                                                                                                                                                                                                                      |
| <i>YOPT_S_PRINT_FN</i>       | Set custom print function native C++ ONLY<br>parmas: func:print_fn_t<br>remarks: you must ensure thread safe of it                                                                                                                                                                             |
| <i>YOPT_S_PRINT_FN2</i>      | Set custom print function with log level<br>parmas: func:print_fn2_t<br>you must ensure thread safe of it                                                                                                                                                                                      |
| <i>YOPT_S_EVENT_CB</i>       | Set event callback<br>params: func:event_cb_t*                                                                                                                                                                                                                                                 |
| <i>YOPT_S_TCP_KEEPALIVE</i>  | Set tcp keepalive in seconds, probes is tries.<br>params: idle:int(7200), interal:int(75), probes:int(10)                                                                                                                                                                                      |
| <i>YOPT_S_NO_NEW_THREAD</i>  | Don't start a new thread to run event loop.<br>params: value:int(0)                                                                                                                                                                                                                            |
| <i>YOPT_S_SSL_CACERT</i>     | Sets ssl verification cert, if empty, don't verify.<br>params: path:const char*                                                                                                                                                                                                                |



| Name                              | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>YOPT_S_CONNECT_TIMEOUT</i>     | Set connect timeout in seconds.<br>params: connect_timeout:int(10)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <i>YOPT_S_DNS_CACHE_TIMEOUT</i>   | Set dns cache timeout in seconds.<br>params: dns_cache_timeout : int(600),                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <i>YOPT_S_DNS_QUERIES_TIMEOUT</i> | Set dns queries timeout in seconds, default is: 5000.<br>params: dns_queries_timeout : int(5000)<br>remark:<br>a. this option must be set before 'io_service::start'<br>b. only works when have c-ares<br>c. since v3.33.0 it's milliseconds, previous is seconds.<br>d. the timeout algorithm of c-ares is complicated, usually, by default, dns queries will failed with timeout after more than 75 seconds.<br>e. for more detail, please see:<br><a href="https://c-ares.haxx.se/ares_init_options.html">https://c-ares.haxx.se/ares_init_options.html</a> |
| <i>YOPT_S_DNS_QUERIES_TRIES</i>   | Set dns queries tries when timeout reached, default is: 5.<br>params: dns_queries_tries : int(5)<br>remarks:<br>a. this option must be set before 'io_service::start'<br>b. relative option: <i>YOPT_S_DNS_QUERIES_TIMEOUT</i>                                                                                                                                                                                                                                                                                                                                 |
| <i>YOPT_S_DNS_DIRTY</i>           | Set dns server dirty.<br>params: reserved : int(1)<br>remarks:<br>a. this option only works with c-ares enabled<br>b. you should set this option after your mobile network changed                                                                                                                                                                                                                                                                                                                                                                             |
| <i>YOPT_C_LFBFD_FN</i>            | Sets channel length field based frame decode function.<br>params: index:int, func:decode_len_fn_t*<br>remark: native C++ ONLY                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <i>YOPT_C_LFBFD_PARAMS</i>        | Sets channel length field based frame decode params.<br>params:<br>index:int,<br>max_frame_length:int(10MBytes),<br>length_field_offset:int(-1),<br>length_field_length:int(4),<br>length_adjustment:int(0),                                                                                                                                                                                                                                                                                                                                                   |

| Name                          | Description                                                                                                                               |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <i>YOPT_C_LFBFD_IBTS</i>      | Sets channel length field based frame decode initial bytes to strip.<br>params: index:int, initial_bytes_to_strip:int(0)                  |
| <i>YOPT_C_REMOTE_HOST</i>     | Sets channel remote host.<br>params: index:int, ip:const char*                                                                            |
| <i>YOPT_C_REMOTE_PORT</i>     | Sets channel remote port.<br>params: index:int, port:int                                                                                  |
| <i>YOPT_C_REMOTE_ENDPOINT</i> | Sets channel remote endpoint.<br>params: index:int, ip:const char*, port:int                                                              |
| <i>YOPT_C_LOCAL_HOST</i>      | Sets local host for client channel only.<br>params: index:int, ip:const char*                                                             |
| <i>YOPT_C_LOCAL_PORT</i>      | Sets local port for client channel only.<br>params: index:int, port:int                                                                   |
| <i>YOPT_C_LOCAL_ENDPOINT</i>  | Sets local endpoint for client channel only.<br>params: index:int, ip:const char*, port:int                                               |
| <i>YOPT_C_MOD_FLAGS</i>       | Mods channel flags.<br>params: index:int, flagsToAdd:int, flagsToRemove:int                                                               |
| <i>YOPT_C_ENABLE_MCAST</i>    | Enable channel multicast mode.<br>params: index:int, multi_addr:const char*, loopback:int                                                 |
| <i>YOPT_C_DISABLE_MCAST</i>   | Disable channel multicast mode.<br>params: index:int                                                                                      |
| <i>YOPT_C_KCP_CONV</i>        | The kcp conv id, must equal in two endpoint from the same connection.<br>params: index:int, conv:int                                      |
| <i>YOPT_T_CONNECT</i>         | Change 4-tuple association for io_transport_udp.<br>params: transport:transport_handle_t<br>remark: only works for udp client transport   |
| <i>YOPT_T_DISCONNECT</i>      | Dissolve 4-tuple association for io_transport_udp.<br>params: transport:transport_handle_t<br>remark: only works for udp client transport |

| Name                        | Description                                                                           |
|-----------------------------|---------------------------------------------------------------------------------------|
| <code>YOPT_B_SOCKOPT</code> | Sets io_base sockopt.<br>params: io_base*,level:int,optname:int,optval:int,optlen:int |

8B5396  
F7C205

[io\\_service Class](#)



API

54547A95  
7D0D7AF4

```
namespace yasio {}
```

5151656C  
6C71B9D5

| Name                   | Description            |
|------------------------|------------------------|
| yasio::host_to_network | 4B6A59825F8C717E59825F |
| yasio::network_to_host | 717E59825F8C717E59825F |
| yasio::xhighp_clock    | 876A6B62A7F6F4B3       |
| yasio::highp_clock     | 876A6B62A7F6F4B3       |
| yasio::clock           | 876A6B62A7F6F4B3       |
| yasio::set_thread_name | 8B7E8C72887E7A5A       |
| yasio::basic_strerror  | 686F72597B45           |

yasio::host\_to\_network

4B6A59825F8C717E59825F82

59654E  
3487F6

yasio/detail/endian\_portable.hpp

```
template <typename _Ty>
inline _Ty host_to_network(_Ty value);
inline int host_to_network(int value, int size);
```

5365  
c270

value

818f637d7950ffvalue7657\_Ty 2345294f011~859827d507c5739

size

818f637d7950ff3c694857827d6cfafd2f1~4598239

8f5650  
64be3c

7f7e59825f695039

794f  
3a8b

```
#include <stdio.h>
#include <inttypes.h>
#include "yasio/detail/endian_portable.hpp"
int main(){
 uint16_t v1 = 0x1122;
 uint32_t v2 = 0x11223344;
 uint64_t v3 = 0x1122334455667788;
 // output will be: net.v1=2211, net.v2=44332211, net.v3=8877665544332211
 printf("net.v1=%04" PRIx16 " ", net.v2=%08" PRIx32 " ", net.v3=%016" PRIx64 "\n",
 yasio::host_to_network(v1),
 yasio::host_to_network(v2),
 yasio::host_to_network(v3));
 return 0;
}
```

yasio::network\_to\_host

7f7e59825f695039

59654f  
3487f6

yasio/detail/endian\_portable.hpp

```
template <typename _Ty>
inline _Ty network_to_host(_Ty value);
inline int network_to_host(int value, int size);
```

5365  
c270

value

818f637d7950ffvalue7657\_Ty 2345294f011~859827d507c5739

size

818f6376655067655b8265ff5380691~45b8239

8f5658  
646e3c

4f5758825f785839

794f  
3a8b

```
#include <stdio.h>
#include <inttypes.h>
#include "yasio/detail/endian_portable.hpp"
int main(){
 uint16_t v1 = 0x2211;
 uint32_t v2 = 0x44332211;
 uint64_t v3 = 0x8877665533442211;
 // output will be: net.v1=1122, net.v2=11223344, net.v3=1122334455667788
 printf("host.v1=%04" PRIx16 " ", host.v2=%08" PRIx32 " ", host.v3=%016" PRIx64 " \n",
 yasio::network_to_host(v1),
 yasio::network_to_host(v2),
 yasio::network_to_host(v3));
 return 0;
}
```

yasio::xhighp\_clock

876863627978f8f36382

59654e  
3487f6

yasio/detail/utils.hpp

```
template <typename _Ty = steady_clock_t>
inline highp_time_t xhighp_clock();
```

6a675365  
2177c278

\_Ty

- yasio::steady\_clock\_t: 8f685527767ff8f3637169f8f363
- yasio::system\_clock\_t: 8f68787fUTCf8f363

8F5650  
040E3C

7E797F63F36389

## yasio::highp\_clock

8762AE7379F2F36389

59654E  
3487F6

yasio/detail/utils.hpp

```
template <typename _Ty = steady_clock_t>
inline highp_time_t highp_clock();
```

6A675365  
217FC270

\_Ty

- yasio::steady\_clock\_t: 8F5650527C7E63F3638979F2F363
- yasio::system\_clock\_t: 846E7B7EUTC F2F363

8F5650  
040E3C

7E797F63F36389

## yasio::clock

8762AE7379F2F36389

59654E  
3487F6

yasio/detail/utils.hpp

```
template <typename _Ty = steady_clock_t>
inline highp_time_t clock();
```

6A675365

\_Ty

- yasio::steady\_clock\_t: 846E53527C7E63F3638979F2F363



- `yasio::system_clock_t`: 8A5B7C7EUTC F2F453

8F5659  
04DE3C

EB797F65256238  
62A7F6F43382

## yasio::set\_thread\_name

8B7B8573897E7A6439  
6E6E8373897E7A6439

59654E  
3487F6

yasio/detail/thread\_name.hpp

```
inline void set_thread_name(const char* name);
```

5365  
C278

*name*

87898B7E7F7A5479  
86818E6E6F6B6D78

6C61  
E80F

515165754E8B65597E7A7A5E975E677538  
6478217F6C6C6F1657284E62895859784E38

## yasio::basic\_strfmt

51656A67FF685F535B784E62895859784E38  
FD78217F6C6C6F1657284E62895859784E38

59654E  
3487F6

yasio/detail/strfmt.hpp

```
template <class _Elem, class _Traits = std::char_traits<_Elem>,
 class _Alloc = std::allocator<_Elem>>
inline std::basic_string<_Elem, _Traits, _Alloc> basic_strfmt(size_t n, const _Elem*
format, ...);
```

5365  
2270

*n*

5758buffer5959  
1058

*format*

686F597843FF54695155 printf 7864

8F5650  
04DE3C

8F68685F535278 std::string 7C5759784E39  
046E3C6F106E64

6C61  
E80F

6B994F73FCE778634F75597E584859785254  
0E997F736CE778634F75597E584859785254

- yasio::strfmt: 686F52597855
- yasio::wcsfmt: 686F5358597845

794F  
3A8B

```
#include "yasio/detail/strfmt.hpp"
int main() {
 std::string str1 = yasio::strfmt(64, "My age is %d", 19);
 std::wstring str2 = yasio::wcsfmt(64, L"My age is %d", 19);
 return 0;
}
```





`yasio` [yasio/detail/config.hpp](#)

| Name                                            | Description                                                                                                                                                                                  |
|-------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>YASIO_HAVE_KCP</code>                     | <code>kcp</code>                                                                                                                                                                             |
| <code>YASIO_HEADER_ONLY</code>                  | <code>yasio</code>                                                                                                                                                                           |
| <code>YASIO_SSL_BACKEND</code>                  | <code>SSL</code> <code>SSL</code> <code>OpenSSL/MbedTLS</code> <code>3.36.0</code> <code>(YASIO_HAVE_SSL)</code> <code>1</code> <code>(OpenSSL)</code> <code>2</code> <code>(mbedtls)</code> |
| <code>YASIO_ENABLE_UDS</code>                   | <code>unix domain socket</code> <code>unix</code> <code>win10 RS5+</code>                                                                                                                    |
| <code>YASIO_HAVE_CARES</code>                   | <code>c-ares</code> <code>c-ares</code> <code>yasio</code> <code>DNS</code> <code>10</code> <code>c-ares</code>                                                                              |
| <code>YASIO_VERBOSE_LOG</code>                  |                                                                                                                                                                                              |
| <code>YASIO_NT_COMPAT_GAI</code>                | <code>Windows XP</code> <code>getaddrinfo</code> <code>API</code>                                                                                                                            |
| <code>YASIO_USE_SPSC_QUEUE</code>               | <code>SPSC</code> <code>io_service::write</code>                                                                                                                                             |
| <code>YASIO_USE_SHARED_PACKET</code>            | <code>std::shared_ptr</code>                                                                                                                                                                 |
| <code>YASIO_HAVE_HALF_FLOAT</code>              | <code>half.hpp</code>                                                                                                                                                                        |
| <code>YASIO_DISABLE_OBJECT_POOL</code>          |                                                                                                                                                                                              |
| <code>YASIO_DISABLE_CONCURRENT_SINGLETON</code> |                                                                                                                                                                                              |
| <code>YASIO_ENABLE_PASSIVE_EVENT</code>         | <code>open/close</code>                                                                                                                                                                      |



yasio 7C535974  
98050406

yasio 76706853597446468458TCP 6F694EUDP 6C826763897A697E63618067526F4E6945786476656F597439734F6582674E79695F:

- 1A2Fio\_service0909 YOPT\_C\_LFBFD\_PARAMS 8B7F4F99536539
- 1A2Fio\_service0909 YOPT\_C\_LFBFD\_FN 8B7F815845639F5E8361F078 decode\_len\_fn\_t 09

6C61E80F

8158456976397F5E7D785B78F8F55CE5598269468B63int58F86C455846897F78678845A806C6247363F8F8863ARM87775982F980E288 SIGBUS 6258E28882E245C289817FE961639F5E7D785B78io\_channel::\_\_builtin\_decode\_len 09

YOPT\_C\_LFBFD\_PARAMS

8B7F4F997462536539

5365C278

max\_frame\_length

6759539F5E6F888756893A5E586389

length\_field\_offset

639F5E596978F84E68686368628859824FF889

length\_field\_length

639F5E5969596F6F62631~489829293(uint8\_t,uint16\_t,uint24\_t,int32\_t)89

length\_adjustment

639F5E8C645C8899585978426C8B5243A8F04FA88A1886869595468684FF5592F5E58685C8328886954F86F1264584E 0 6C52124A 8869597F5E 09

6C61E80F

886F639F5E58688F984F7872810C57825E7F7838886777517F89785395585873:

```
int io_channel::__builtin_decode_len(void* d, int n)
{
 int loffset = uparams_.length_field_offset;
 int lsize = uparams_.length_field_length;
 if (loffset >= 0)
 {
 int len = 0;
 if (n >= (loffset + lsize))
```

```
{
 ::memcpy(&len, (uint8_t*)d + loffset, lsize);
 len = yasio::network_to_host(len, lsize);
 len += uparams_.length_adjustment;
 if (len > uparams_.max_frame_length)
 len = -1;
}
return len;
}
return n;
}
```

## decode\_len\_fn\_t

41 93 93 73 68 68 53 9f 56 fd 78 6f 57 88 89

```
typedef std::function<int(void* d, int n)> decode_len_fn_t;
```

53 65  
c2 70

*d*

65 93 53 63 62 88 58 82 57 57 89

*n*

6f 93 53 63 62 9f a6 89

8f 56 50  
84 6e 3c

84 68 88 69 53 78 62 6e 99 9f a6 89

- > 0 : 93 78 63 9f 56 62 57 89
- == 0 : a3 62 78 62 46 89 4f 93 78 88 69 53 68 99 9f 56 6f yasio 55 42 4a 7f 7e a3 53 63 62 6f 48 63 53 52 63 63 62 6f 4f 6b 21 83 73 62 51 73 89
- < 0 : 93 78 63 9f 56 62 56 6c 4f 99 61 55 52 48 85 4f 88 63 66 89

6c 61  
e8 0f

93 78 88 69 53 9f a6 51 75 c5 98 29 7e 7a 58 51 76 f6 6b 82 f9 4e Lua 19 46 2f 61 88 7e 2a 5a 49 93 78 88 69 53 9f a6 fd 78 89

## yasio Interop

Unity C# yasio C: [https://github.com/yasio/yasio/blob/master/yasio/bindings/yasio\\_ni.cpp](https://github.com/yasio/yasio/blob/master/yasio/bindings/yasio_ni.cpp)

## OpenNSM2

Unity yasio-3.37.1 NetworkServiceManager unity: <https://github.com/yasio/OpenNSM2>





## FAQ

91709598897B  
CD89EE98E354

yasio 24469874687688?

yasio 74583f4579988464f80464f7945655f4e4f804e4f8e51904f884e5f595c7478777f651454e9f2189597576125717f55ffxxsocket1c99yasio 749965f23675f69  
886f61697648f25465686584eboost.asio82

4545454f75  
5Ac0487728yasio?

- 666672f3f27c6886764797c7c5567asio,libevent,libev,libuv6c4e4e634e74f9829258f748747c9858f788io697329576c746c6475f3855247f73  
iocp,kqueue,epoll,select7829876f74352967747c7c7c686863747646TCP7663616478f88768f6382AF9626289
- yasio6c8f637974ffTCPc263f951c883245554269
- yasio5cTCP,UDP,KCP7ef4668281f2Transportf952654ef7f2869
- yasiof9982f75f6c26275f3274f72select717289

yasio 5f54696397995857548967?

6f616c4687894688c-ares596f6546552f8865873DNS57886f76f745587845d7387select6158765647f736c67669967f8876567f7a888967465652fc-ares6fyasio5  
284f4a69696764896765676a88464f8887558106282f6f225f5f6528c2f56876a59897488

yasio 2f246663SSL/TLS4688?

6f616c4687894688OpenSSL7289MbedTLS89

594f4eio\_event 45887f548353  
6265ce userdata 26886e8c8706

8853831fhttps://github.com/yasio/ftp\_server/blob/master/ftp\_server.cpp#L98

yasio 66545974 SIGPIPE 4f53ff  
2f266466 f1f71f

4e 3.30 782765f8f9697489

836246697a616e6882fios527279f8966688839f24a2263SIGPIPE7963f8f1APP288f663762626f254e2263TCP8f632866f6f389326f838f4f69: Broken pipe  
8889: https://github.com/yasio/yasio/issues/170

**ios** **ec=65, detail:No route to host**

- **ios**:
  - **APP** **ios14.1+** **Local Network Permission**
  - **10161** **SNMP via TLS**
- **APP** **49152~65535**
- **https://www.speedguide.net/port.php**

**macOS** **UDP** **40** **Message too long**

- **macOS** **UDP** **9126**
- **socket** **UDP**:
  - **xxsocket**: **sock\_udp.set\_optval(SOL\_SOCKET, SO\_SNDBUF, (int)65535);**
  - **io\_service**: **https://github.com/yasio/yasio/blob/master/tests/speed/main.cpp#L223**

**io\_service** **schedule**

**io\_service**

**std::thread**

**https://github.com/yasio/yasio/issues/244**

**xxsocket** **\_nodelay**

**\_nonblock** **internal**

**YOPT\_TCP\_KEEPALIVE**

**https://github.com/yasio/yasio/issues/117**

## Lua

### • c++11:

- `kaguya` `c++` `lua_newuserdata`, `placement new`  
`xcode clang release`
- `bing.com` `LUA_USER_ALIGNMENT_T=max_align_t`
- `, max_align_t` `xcode clang` `long double` `16`
- `: http://lua-users.org/lists/lua-l/2019-07/msg00197.html`

### • c++14:

- `sol2` `sol2` `kaguya` `sol2` `lua_newuserdata`  
`clang release`

### • c++17:

- `sol2` `xcode` `ios11` `C++17` `new` `Aligned allocation/deallocation`  
`-faligned-allocation`
- `ios10` `std::shared_mutex` `(yasio` `Apple` `pthread`)
- 
- `Lua` `double` `max_align_t` `C11` `https://en.cppreference.com/w/c/types/max_align_t` `Lua` `ANSI C89` `LUA_USER_ALIGNMENT_T`
- `C++11` `std::max_align_t` `https://en.cppreference.com/w/cpp/types/max_align_t`

`max_align_t` `llvm` `__stddef_max_align_t.h`

## Can't load xxx.bundle on macOS?

The file `xxx.bundle` needs change attr by command `sudo xattr -r -d com.apple.quarantine xxx.bundle`

## xxsocket resolve socketype

- `winsock`
- `0` `IP`

`F41A38C1EE98`

`F7C265 987E5E52F3`