



yasio `4E6565682D878763`

docs **passing**

yasio `2F4E4F8F917F8855F8765F6B2F6B5F6C13F86FA237E8CFA8E547A4E54796B625F64766E625B627A7F7E6752B9` socket `5EFC13F86FA237E8CFA8E547A4E54796B625F64766E625B627A7F7E6752B9`

- `656B8B8A`:
  - [English](#)
  - `78534E65`
- `8875F329`:
  - `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`

图 1-1-1:

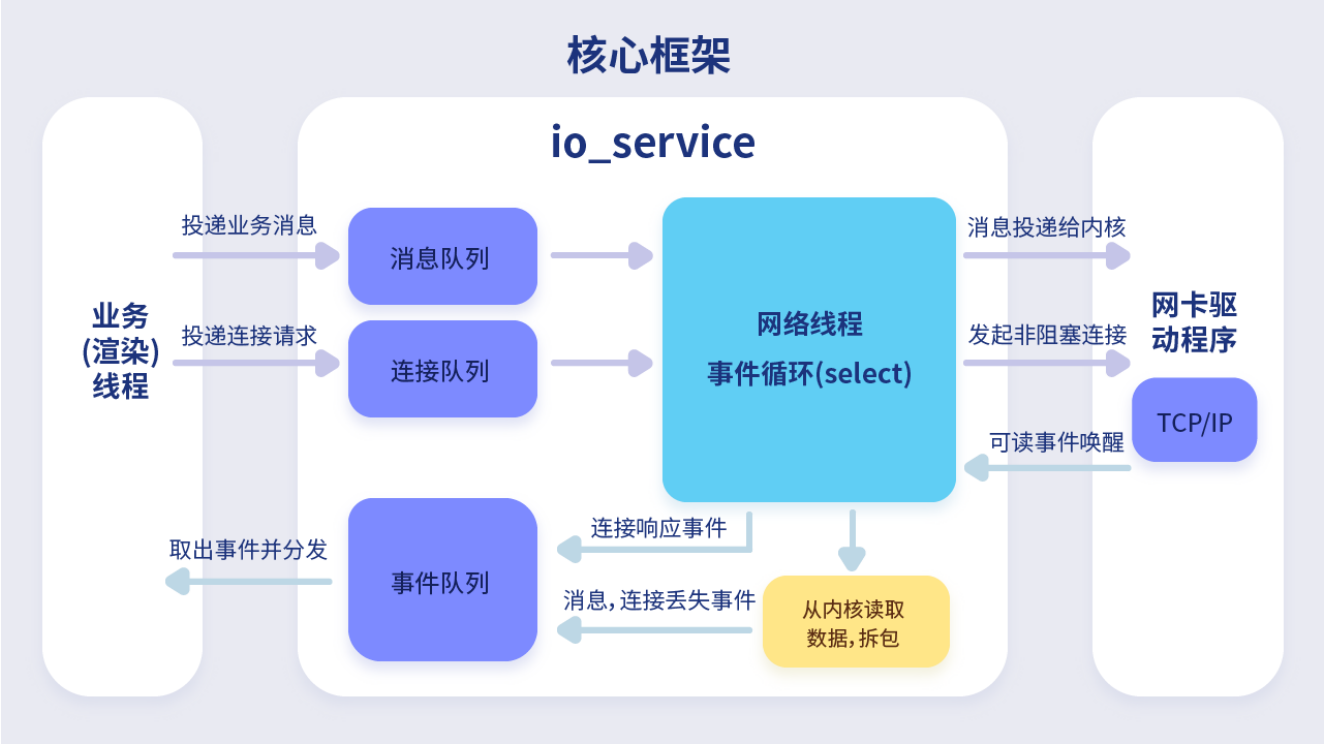


图 1-1-1

tool.chinaz.com http://

## 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_PACKET: {
            auto packet = std::move(ev->packet());
            fwrite(packet.data(), packet.size(), 1, stdout);
            fflush(stdout);
            break;
        }
        case YEK_CONNECT_RESPONSE:
            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_CONNECTION_LOST:
            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_PACKET) then
        respdata = respdata .. ev:packet():to_string()
    elseif k == yasio.YEK_CONNECT_RESPONSE 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_CONNECTION_LOST 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



6C61  
E86F

8F88 Lua 794F7A5E9F4F62535F5E4F69 yasio - ibstream\_view::consume out of range! FF8F69794F7A5E9169615176FF8B4E5F576139  
6642 3A8B6B8F6C1A537662C38E16F 6C692F3A8B6B8F6C456F69846CF76D65286F62

- 6D8B:
  - echo\_server: TCP/UDP/KCP 6B646D2168
  - echo\_client: TCP/UDP/KCP 6B64A9627E
  - ssltest: SSL 8B8B2B627A, 8F69github.com 4B987B637B846E6262
  - tcptest: TCP 4B8B7A6F

- [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
- 88485946704E:

```
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  
C278

## cb

7F7E464E6983FF68A9C9534E57 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_CONNECT_RESPONSE)
        {
            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()
```

6061  
e80f

59697f7e67527f7a69528f88, 6951654f539990514f535e795f51685e995139  
64fd791a616160fae1f77649857663880fa62

## io\_service::open

525f4f9939

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

5365  
c270

*cindex*

4f53295f39

*kind*

4f53768939

6061  
e80f

5948 TCP, 664af9425168529997985832f14b6267fa7aef978e6382

*cindex* 7950c598594eio\_service52c953e3794f5376cf89

*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

1A5E68656C5E5F57515357F17E4E4E764E52908F7F7A8C75, 4F59Cocos2d-x766E677F7A8C4E53514E6E675F64(Unity,UE4)764E908F7F7A39

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
);
```

5365  
C270

*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

618188166E8189

8F5650  
04DE3C

765F538976598276, 55 < 0 F86F6F637F198F8F1A582F26834A88F273E5(TCP8E63A86678)89

6C61  
E88F

63F07845F7Z84E DGRAM 28834A886F73 UDP, KCP 39

538958166E81 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
EA	9A	49	51	DC	E5	D7	53	70	FD	70	02

6C  
E8 61  
OF

6A 51 6B 6E 6F 63 69 81 75 6C 4F 6F 53 78 62 97 89 51 66 51 7F 6C 7E 65 91 5B 5A 51 52 81 5B 4A 65 67 7B 7F 6F 6C  
6D 6D 78 2F 6F 63 69 81 75 6C 4F 6F 53 78 62 97 89 51 66 51 7F 6C 7E 65 91 5B 5A 51 52 81 5B 4A 65 67 7B 7F 6F 6C

52 97 5F 5B 67 75 6F 6C 8B 52 91 5B 5A 51 52 UE4 54 U3D 84 65 5F 8F 51 82  
52 97 5F 5B 67 75 6F 6C 8B 52 91 5B 5A 51 52 UE4 54 U3D 84 65 5F 8F 51 82

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)52E598837869F8656E6E89954787A5FE2A8869

io\_service::channel\_at

9A87EF83725F87624E99E36489

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

5365  
C270

*cindex*

4F937965

8F5650  
04DE3C

4F93E3646788, 53795F5889FA83F4F86E845E nullptr 89

io\_service::set\_option

887E889889

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

5365  
C270

*opt*

8893674E, 89E777 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,      // lengthFieldOffset, 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

83524f998f7a7ae3

```
int index() const
```

## io\_channel::remote\_port

83524f998f7a7ae3.

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

8f5650  
04de3c

8f564f998f7a7ae3f7

- f98e2377a4f9988799a1798f7a7ae3
- f98e6757a4f99887976547ae3

## io\_channel::bytes\_transferred

835258677af365685982f8

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

8f5650  
04de3c

4e8b63fa7a5f58565520ff6336585982f86c784ef88a2377af3656997

## io\_channel::connect\_id

835258677af4f9999587658f28f63id

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

8f5650  
04de3c

58677af4f995520ff63id



[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::packet	
io_event::timestamp	
io_event::transport	
io_event::transport_id	ID
io_event::transport_udata	

## io\_event::kind

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



8F5650  
D4DE3C

4E4E7C57FEF34E6F4E4E50:

- YEK\_PACKET : 606F636E4E
- YEK\_CONNECT\_RESPONSE : 8E63C65A4E4E
- YEK\_CONNECTION\_LOST : 8E632E594E4E

## io\_event::status

83534E4E726989

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

8F5650  
D4DE3C

- 0: 695E
- 70: 7A19, 7867E360897853637873E769

## io\_event::packet

83534E4E6A5E78696963

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

8F5650  
D4DE3C

686863784F578, 7867E3608978 std::move 68GC695F4E4E4E6868686382

## io\_event::timestamp

83534E4E4E7F786E797E63E36389

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

8F5650  
D4DE3C

54787E63E3E373785E797E63E36389

## io\_event::transport

87524E4E794F854F8B23C7B9

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

8F5650  
D4DE3C

5F5953C7FF5F6252555F4E4E656F26834F8B53C7F2595963FF4E53734F5757596B8F39  
D4DE53C7FF5F6252555F4E4E656F26834F8B53C7F2595963FF4E53734F5757596B8F39

## io\_event::transport\_id

87524E4E794F854F8BIDB9

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

8F5650  
D4DE3C

324F6378F7727883F48579534EIDB9

## io\_event::transport\_udata

887F1687524F854F8B78576562B9

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

6C61  
E80F

78579789E1F17B74 userdata 785358, 8882:

- 65528E63FA7A62524E4E655858userdata
- 65528E6372594E4E656574userdata

8B5396  
F7C205

[io\\_service 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>;  
}
```



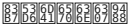


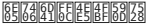
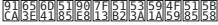
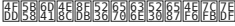
6254  
1058

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>	



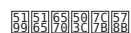
: obstream.hpp

## 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  
8159636430

6C61  
E88F

\_Nty 5849705757456948671~8598263637057626979705739

## ostream::write\_ix

6632/644F6968594E7Bit Encoded Int656F887F6451516989

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

5365  
C270

value

895151795989  
815963643082

6C61  
E88F

\_Intty 705753896959487057

- int32\_t
- int64\_t

64F0708879707069696F7068AE8Fdotnet504E6165

- [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  
89512A578282

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

5365  
C270

value

895151765938  
819963643C82

6C61  
E88F

69F1785289784F4E ostream::write

## ostream::write\_bytes

51515982637C38  
89635982787C82

```
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

8951515982637C4749865757.  
8199635782787C4749865757.



*length*

895151598265757479955.  
816963578278746477A6.

*offset*

895151598265747979685979.  
816963578278746477A6.

6C61  
E88F

offset + length 70505F985C4E  
645C51786F6E obstream::length

obstream::empty

5260406F524E7A82  
24A0496F263A7A82

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

8F5650  
04DE3C

true 7A; false 927A82

6C61  
E88F

695168784F4E length == 082  
697878494F4E

obstream::data

83536363639439  
87D8786E878882

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

8F5650  
04DE3C

598260636388575739  
598260636388575739

obstream::length

8353609F5F82  
87D8409FA682

```
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

634f4f88f52725f57525f8832

6567

27238f52716276a4ff5524f63624689f34762fa std::out\_of\_range 625832

886C  
EDD5

```
namespace yasio {  
  // 206ff7f287686cf4f81528f6259825ff6c8925a8516c4685  
  using ibstream_view = basic_ibstream_view<endian::network_convert_tag>;  
  
  // 206ff7f287686cf4f81528f6259825ff6c8925a8516c4685  
  using fast_ibstream_view = basic_ibstream_view<endian::host_convert_tag>;  
}
```



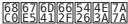


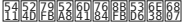
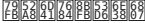
6254  
1058

62716728

Name	Description
ibstream_view::ibstream_view	672812a ibstream_view 6961

5151656C  
6C7189D5

Name	Description
ibstream_view::reset	917f5f238ff52716276a2c685c685f716276a
ibstream_view::read	2d682a766f88637a38
ibstream_view::read_ix	f3682a766f8863(7bit Encoded Int/Int64)737858
ibstream_view::read_v	88625e7257(7bit Encoded Int/Int64)294e86327332

Name	Description
<code>istream_view::read_byte</code>	 88 53 14 f5 82 fb 06 12 a5 98 2
<code>istream_view::read_bytes</code>	 88 53 63 58 92 5f 4e 8f 52 65 63 fb 06 07 6a 7f a6 8c bb 56 76 62
<code>istream_view::empty</code>	 68 62 6d 66 54 45 7a fb 06 41 2f 26 5a 7a
<code>istream_view::data</code>	 83 53 6d 66 62 63 84 fb 06 41 76 62 63 84
<code>istream_view::length</code>	 83 53 6d 59 5c 87 06 41 27 0f
<code>istream_view::advance</code>	 54 52 79 52 60 74 88 63 55 69 11 06 78 62 41 64 88 63 55 69
<code>istream_view::seek</code>	 78 52 6d 76 66 62 63 69 fb a8 41 64 88 63 63 69

6C	61
E8	0F

```
ibstream_view 50 92 C++17 68 51 76 std::string_view, 61 54 77 57 52 59 53 54 53 56 57 53 67 7A 4E 4E 4F 4E 75 4E 45 GC 30
```

89	6C
81	42

```
59654E: ibstream.hpp
```

```
istream_view::istream_view
```

```
67904E4E ibstream_view 5B8C30
8420002A F96102
```

```
ibstream_view();  
  
ibstream_view(const void* data, size_t size);  
  
ibstream_view(const ostream* obs);
```

53	65
C2	70

*data*

5F 53 5E 52 53 4E 8F 52 65 63 99 57 57 30  
85 CD 8F 17 16 8C DB 36 70 6E 96 30 40 02

*size*

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

*obs*

F0 5F 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 67 57 80

*size*

5F 53 5F 52 53 40 8F 52 63 59 6C 80  
85 C0 8F 17 16 8C 80 66 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 50 64 50

6C 61  
E8 0F

\_Nty 50 80 70 80 52 4E 60 40 67 1~8 50 82 63 63 70 52 62 60 70 70 57 80

istream\_view::read\_ix

80 53 7Bit Encoded Int 53 70 16 70 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

8B5353974E8F52636389

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

8F5650  
04DE3C

8F56885352794E8F5263638956FF65GC89

6C61  
E80F

7C7D787A4B8B537bit Encoded Int807576797897A658FF808978 read\_bytes 8B534E8F525982786289

ibstream\_view::read\_byte

8B5312A578289

```
uint8_t read_byte();
```

8F5650  
04DE3C

uint8\_t5089

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  
412BFB11632B12A598262

6C61  
E80F

696363784F4E length == 082

ibstream\_view::data

8F56606363639482  
D46E41786E678882

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

8F5650  
D4DE3C

8F6E6754694E78464A598276678A82  
646E6754694E78464A598276678A82

## istream\_view::length

8752609275E39

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

8F5659  
D4DE3C

5F52609275E39  
53404177A682

## istream\_view::advance

545279526D8B536F6839  
1140FBAB41FB66386782

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

5365  
C278

*offset*

89545279527659799139  
811140FBAB844FFBCE82

6C61  
E86F

82\*offset\*4F8D65FF525364E379528B536F6839  
261F70419CB1E3FBAB8B66386782

## istream\_view::seek

79528B536F6839  
FBABFB66386782

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

5365  
C278

*offset*

84\*whence\*76517959799139  
8173844FFBCE82

*whence*

544E78644EC68F65674E59FF SEEK\_SET, SEEK\_CUR, SEEK\_END 82  
2849496C8E

8F	56	50
D4	DE	3C

8F	56	79	52	54	76	5B	4E	6D	99	5B	82	50	79	30
D4	DE	FB	A8	0E	F8	F9	8E	41	96	57	82	4F	FB	02

## ibstream Class



63	4F	4E	8F	52	65	63	52	8F	54	53	5E	52	53	52	80	30
D0	9B	8C	DB	36	70	6E	A0	7D	8C	CD	8F	17	16	9F	FD	02

8B	6C
ED	D5

```
namespace yasio {
using ibstream = basic_ibstream<endian::network_convert_tag>;
// 0000000000000000000000000000000000000000000000000000000000000000
using fast_ibstream = basic_ibstream<endian::host_convert_tag>;
}
```

62	54
10	58

51	51	67	90	51	65
6C	71	84	20	FD	70

Name	Description
ibstream::ibstream	 ibstream 

51	51	65	6C
6C	71	B9	D5

Name	Description
ibstream::load	4E 65 4E 52 8F 6D CE 87 F6 A6 7D 41

7E	62	5C	6B	7E	67
E7	7F	42	21	D3	84

## istream\_view

ibstream

ibstream::ibstream

```
67904E4E ibstream 5B8C30
8420002A F96102
```

```
istream(std::vector<char> blob);  
  
istream(const ostream* obs);
```

`blob`

`obs`

`ostream`

`load`

`load(const char* filename) const;`

`load`

`load`

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

`load`

`load`

`load`

`load`

`load`

`load`

`load`



# 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
<code>xxsocket::resolve_tov6</code>	<code>ipv4</code> <code>ipv6</code> <code>V4MAPPED</code>
<code>xxsocket::getipsv</code>	<code>ip</code>
<code>xxsocket::traverse_local_address</code>	

## xxsocket::xxsocket

`xxsocket`

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

`handle`

`handle`

`socket` `xxsocket`

`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`

`hostname`

`IP` `16` `5760`

port

818f63675258797af2389

local\_port

7c50984f7af23f76c6b8850 0 88798967529189

6c61e80f

4fa152686b2c67626378ip538f68727c89

8f5650b4be3c

0: 6252ff < 0 5929ff908f xxsocket::get\_last\_errno 8753938f7989

xxsocket::xpconnect\_n

8f6c7a675258ffa7aTCP8f6389

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

5365c270

hostname

818f636752584b6754ffef4e6f IP6787 16 5760 89

port

818f63675258797af2389

local\_port

7c50984f7af23f76c6b8850 0 88798967529189

wtimeout

fa7a8f6389f8629582

8f5650b4be3c

0: 6252ff < 0 5929ff908f xxsocket::get\_last\_errno 8753938f7989

6c61e80f

4fa152686b2c67626378ip538f6882

## xxsocket::pconnect

548f7a675258ffa7aTCP8fa339

```
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);
```

5365  
c270

*hostname*

818fa3675258ffa7a60ffef4e69 IP5670 16 5f64 82

*ep*

818fa3675258ffa7a64a39

*port*

818fa3675258ffa7a2339

*local\_port*

6752994f7aef3f6c088a50 0 88798967529139

8f5650  
646e3c

0: 6252ff < 0 51296f998f xxsocket::get\_last\_errno 875393ef7939

6c61  
e80f

4f4f6860676762626179ip538b6882

## xxsocket::pconnect\_n

548f7a675258ffa7aTCP8fa339

```
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);
```

5365  
c270

*hostname*

818fa3675258ffa7a60ffef4e69 IP5670 16 5f64 82

*ep*

818f63675258795757b9

*port*

818f63675258797af3b9

*local\_port*

6750904f7af3f76c6b8b50 0 88798f675291b9

*wtimeout*

fa7a8f6389f8f505b9

8f5650  
b4de3c

0: 1052ff < 0 51296c1a87 xxsocket::get\_last\_errno 875393ef79b9

xxsocket::pserve

5f546757TCP6752795430  
602f2c30 60a1d12c62

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

5365  
c270

*addr*

676a636a7f61 IP6a77 b9

*ep*

6767575739  
2c5a5040b2

*port*

TCP79547af3b9

8f5650  
b4de3c

0: 1052ff < 0 51296c1a87 xxsocket::get\_last\_errno 875393ef79b9

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

xxsocket

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

who

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 887A2848socket82

## xxsocket::release\_handle

21636555socket6F6F78A7564382

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

8F5650  
D4DE3C

21635274socket6748636F78

## xxsocket::set\_nonblocking

8876socket7482788585A6F82

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



5365  
C270

*nonblocking*

true: 9798586A5F6F false: 9858215F69

8F5650  
04DE3C

0: 6262FF < 0 5929FFA987 xxsocket::get\_last\_errno 875393297939

## xxsocket::test\_nonblocking

6840socket6F644E9798586A5F69

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

8F5650  
04DE3C

1: 9798586A5F6F 0: 9858215F69

6C61  
E80F

694Ewinsock26C6A8FA374 SOCK\_STREAM 7057socket1A645E -1 69

## xxsocket::bind

7F59socket676767575739

```
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

6F2B705263EA 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  
A307602A5237E80F6380

```
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

897868F76548586CF8F782A38373 xxsocket::set\_nonblocking 66socket887E4A2798586A6F82

## xxsocket::connect

FA7A8E6389

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

5365  
C278

*addr*

8F7A4667ip575789

*port*

8F7A46677A5389

*ep*

8F7A4667575789

8F5650  
04DE3C

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

6061  
e88f

TCP: 5189TCP 6921e162

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*

8f7a4a67575739  
8f7a4a67575739 ip 575739

*port*

8f7a4a67575739  
8f7a4a67575739 575739

*ep*

8f7a4a67575739  
8f7a4a67575739 575739

*wtimeout*

f57a8fa339  
f57a8fa339 8753932ff7839

8F5650  
04DE3C

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

6061  
e88f

TCP: 5189TCP 6921e162

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) xxsocket::connect\_n

## 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 8288586A5E 39

## xxsocket::recv\_n

5480F2F2F2A156644E846862FA63687F56766289

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

5365  
C278

*buf*

6368786E75625A82

*len*

6368786E75625A7F5682

*wtimeout*

636880F2F2F282

*flags*

636363635E5C688839  
A538786E9542678882

8F5659  
D4DE3C

```
==len: 6252FF < len 598D6C7A8F xxsocket::get_last_errno 835395887882
```

## xxsocket::sendto

54807A4E675388 DGRAM 68UDP68766282

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

5365  
C278

*buf*

656188786E75625A82

*len*

656188786E75625A7F5682

*to*

61817868574882

*flags*

6181786E5542688882

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*

6358786275515A89

*len*

6358786275515A955E82

*peer*

635878626468FF85FA237889

*flags*

635878625E5C688889

8F5650  
04DE3C

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

6C61  
E88F

68715859547A538458FF52514Esocket6788585468 828858255F 89

## xxsocket::handle\_write\_ready

785Fsocket535182

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

5365  
C278

*wtimeout*

785F88F8F8F882



8F5650  
04DE3C

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

6C61  
E80F

9A585F516861897F6251696F76C95146FF69F67684A7A53846889

## xxsocket::handle\_read\_ready

7865socket278889

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

5365  
C270

*wtimeout*

7865806565F989

8F5650  
04DE3C

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

## xxsocket::local\_endpoint

8763453C41904F766757575789

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

8F5650  
04DE3C

8F58675757575789

6C61  
E80F

59676A6789788F xxsocket::connect 6289TCP8E633289E1626768626E9945846879575768 0.0.0.0

## xxsocket::peer\_endpoint

8763453C41904F76677A575789

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

8F5658  
04DE3C

8F566257575739  
04DE2C30304082

6C61  
E88F

5962666969728F xxsocket::connect 1689 TCP8E63369E31626268626E99458F587957572F 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*

5351626552C5F9B6E54F86FCB1595189C5F9A24B62F8F4F4846C534DFF68B26982

*probes*

5351626552C5F9B6E54F86FCB1595189C5F9A24B62F8F4F4846C534DFF68B26982

8F5658  
04DE3C

0 : 188F6C < 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*

socket89997F755239

*optname*

89997C75739

*optval*

89995039

8F5650  
04DE3C

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

6C61  
E80F

64F07354bsd socket setsockopt 528878546FE32F7F736A6750886CF4696F7F7369

xxsocket::get\_optval

887Esocket899939

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

5365  
C270

*level*

socket89997F755239

*optname*

89997C75739

8F5650  
04DE3C

846E89995039

6C61  
E80F

64F07354bsd socket getsockopt 528878546FE32F7F736A6750886CF4696F7F7369

## xxsocket::select

7952socket846848484839

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

5365  
C270

*readfds*

F2F84848F2F878657430

*writefds*

F2514848F2F878657430

*exceptfds*

55584848F2F878657430

*wtimeout*

7851484889F8484830

8F5650  
D4DE3C

0:80F2FF > 0:6252FF < 0:5989FF908F xxsocket::get\_last\_errno 870819887930

## xxsocket::shutdown

73E8TCP48851A9930

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

5365  
C270

*how*

73E81A997988FFEF204E486A674E50

- SD\_SEND : 51891A99
- SD\_RECEIVE : 63631A99
- SD\_BOTH : 88E873E8

8F5650  
04DE3C

0: f252ff < 0: 5129ff992f xxsocket::get\_last\_errno 8762932f7932

## xxsocket::close

732fsocket6f2a62767f446662

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

5365  
c270

### *shut\_how*

732f1a937b576cf2e54b4545674e52

- SD\_SEND : 51811a99
- SD\_RECEIVE : 63539999
- SD\_BOTH : 21205125
- SD\_NONE : 58287325

8F5650  
04DE3C

0: f252ff < 0: 5129ff992f xxsocket::get\_last\_errno 8762932f7932

6c61  
e80f

5261 shut\_how != SD\_NONE 6c62f07b1a4a8328 shutdown 6c608c72b6522 close 62

## xxsocket::tcp\_rtt

8752TCP8f6379RTT39

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

8F5650  
04DE3C

8458TCP79RTTf2f2ff6c534b: 5e79 39

## 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 F6FC837869FD78B2

xxsocket::not\_recv\_error

72AD2F245F6352F2socketFA68E86577E664F3E662

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

5365  
C278

error

79EF78B2

8F5650  
04DE3C

true : socket69586C false : socket7260F27E6179938BFC655373E6socket7E6299AF62

6C61  
E80F

4E536363644F8F6E50 < 0 F6FC837869FD78B2

xxsocket::strerror

5C958B788F634E5B7B4E30  
8619EE616C623A5A57263262

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

5365  
C278

error

79EF78B2

8F5650  
04DE3C

79EF7F697659784E30  
19EF7F6457263262



## 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: 00000000 > 0 0000 xxsocket::strerror 00000000

## 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

89676760632b64606757576f89676760632b64606757576fIPv457574f8f634aIPv6 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 95 80 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 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`: `676A2F63` | `Pv4` `5F8B89`
- `ipsv_ipv6`: `676A2F63` | `Pv6` `5F8B89`
- `ipsv_dual_stack`: `676A2F63` | `Pv4` `5C` | `Pv6` `5C684F8E89`

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 87 7A 7F 7E 65 FF 5C 4F 4F 51 90 62 wifi FF  
88 82 7A FD 4B 3A BE 07 28 0C F6 00 2F 02 9D 51 DC F6 0C 06 1A 18 48 09 E9 wifi 0C

📶wifi📶IPv4: <https://github.com/halx99/yasio/issues/130>

794F  
3A8B

```
// 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

674E5757598C39  
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 params: callback:defer_event_cb_t remarks: a. User can do custom packet resolve at network thread, such as decompress and crc check. b. Return true, io_service will continue enqueue to event queue. c. Return false, io_service will drop the event.
<i>YOPT_S_DEFERRED_EVENT</i>	Set whether deferred dispatch event, default is: 1 params: deferred_event:int(1)
<i>YOPT_S_RESOLV_FN</i>	Set custom resolve function, native C++ ONLY params: func:resolv_fn_t*
<i>YOPT_S_PRINT_FN</i>	Set custom print function native C++ ONLY parmas: func:print_fn_t remarks: you must ensure thread safe of it
<i>YOPT_S_PRINT_FN2</i>	Set custom print function with log level parmas: func:print_fn2_t you must ensure thread safe of it
<i>YOPT_S_EVENT_CB</i>	Set event callback params: func:event_cb_t*
<i>YOPT_S_TCP_KEEPALIVE</i>	Set tcp keepalive in seconds, probes is tries. 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. params: value:int(0)
<i>YOPT_S_SSL_CACERT</i>	Sets ssl verification cert, if empty, don't verify. params: path:const char*



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

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

Name	Description
<code>YOPT_B_SOCKOPT</code>	Sets io_base sockopt. 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	876A6362A7F6F433
yasio::highp_clock	876A6362A7F6F433
yasio::clock	876A6362A7F6F433
yasio::set_thread_name	8B7E8C23887F7A5A
yasio::basic_strerror	686F72592B45

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  
646e3c

7f7e59825f7d5039

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

7f7e59825f7d5039

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

818f637665506765598265ff5380691~4598239

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

876865727f78936382

59654e  
3487f6

yasio/detail/utils.hpp

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

6a675365  
217f6278

\_Ty

- yasio::steady\_clock\_t: 8f685527767ff823237169f82383
- yasio::system\_clock\_t: 8f68787fUTCf8f463

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  
6662A7F6F43382

59654E  
3487F6

yasio/detail/thread\_name.hpp

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

5365  
C278

*name*

87898B7E7F7A5479  
06818E6E6F6B6DFA

6C61  
E80F

6B5165754E8B65597E7A7A5E975E677538  
64FD78217F6ECAAD1A6F6B6B8F5E38692862

## yasio::basic\_strfmt

51636A67FF685F535B7B4E6289585B7B4E38  
FD78217F6E6C6F16572B3216896D5B7B4E38

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*

686F587843FF54687C655 printf 7864

8F5650  
04DE3C

8F68685F536278 std::string 7C5759784E39  
046E6C6F106E64

6C61  
E80F

68994F73FCE778634F75597E584859785254  
0E957F786CE778634F75597E584859785254

- yasio::strfmt: 686F53597865
- yasio::wcsfmt: 686F5358597865

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 yasio/detail/config.hpp

Name	Description
YASIO_HAVE_KCP	kcpkcp
YASIO_HEADER_ONLY	yasio
YASIO_SSL_BACKEND	SSLSSL OpenSSL/MbedTLS 3.36.0 (YASIO_HAVE_SSL) 1 (OpenSSL) 2 (mbedtls)
YASIO_ENABLE_UDS	unix domain socket unix win10 RS5+
YASIO_HAVE_CARES	c-ares c-ares yasioDNS 10 c-ares
YASIO_VERBOSE_LOG	
YASIO_NT_COMPAT_GAI	Windows XP getaddrinfo API
YASIO_USE_SPSC_QUEUE	SPSC io_service::write
YASIO_USE_SHARED_PACKET	std::shared_ptr
YASIO_HAVE_HALF_FLOAT	half.hpp
YASIO_DISABLE_OBJECT_POOL	
YASIO_DISABLE_CONCURRENT_SINGLETON	



yasio 7C535974  
98050406

yasio 76706853597446468458TCP 6F694EUDP 6C826763897A697E63618067526F4E6F45786476656F597439734F6582674E79695F:

- 1A2Fio\_service0909 YOPT\_C\_LFBFD\_PARAMS 8B7F4F09536539
- 1A2Fio\_service0909 YOPT\_C\_LFBFD\_FN 8B7F815845639F5E8361F078 decode\_len\_fn\_t 09

6C61E80F

8158456876537F46705878F8F554E598269468863int38F86C455846897F78678845A806C6247363F8F8863ARM87775982F98F8288 SIGBUS 6258E88882E24E5289817FE961639F467078828 io\_channel::\_\_builtin\_decode\_len 09

YOPT\_C\_LFBFD\_PARAMS

8B7F4F09C262F36289

F365C278

max\_frame\_length

6759539F7F466F888756893A5F586389

length\_field\_offset

639F46596978F84E68686368628859824FF889

length\_field\_length

639F465969596F6F62631~489829298( uint8\_t,uint16\_t,uint24\_t,int32\_t )89

length\_adjustment

639F4663745C88995859784F737F7E58825E7F7838886777517F89785395585873 0 6C520124A 8869597F5E 09

6C61E80F

886F639F465768C578787E28510C57825E7F7838886777517F89785395585873:

```
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 b6 3c

84 68 88 68 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 68 53 68 99 9f 56 6f yasio 55 a2 4a 7f 7e a3 63 63 62 6f 48 63 63 52 63 63 62 6f 4f 6b 21 83 73 62 51 78 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 68 53 9f a6 fd 78 6f 57 98 2f 7e 7a 58 51 76 6f 4f 59 f9 4e Lua 19 46 2f 61 88 7e 2a 5a 49 93 78 88 68 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**?

yasio 是 libevent, libev, libuv, iocp, kqueue, epoll, select, TCP, UDP, KCP, Transport, SSL/TLS, MbedTLS, OpenSSL, Boost.Asio 的封装。yasio 是 C++ 的，boost.asio 是 C 的。

**yasio?**

- asio, libevent, libev, libuv, iocp, kqueue, epoll, select, TCP, UDP, KCP, Transport, SSL/TLS, MbedTLS, OpenSSL, Boost.Asio
- yasio TCP
- yasio TCP, UDP, KCP, Transport
- yasio select

**yasio**?

c-ares, DNS, select, SSL/TLS, MbedTLS, OpenSSL, Boost.Asio

**yasio** SSL/TLS?

OpenSSL, MbedTLS

**io\_event userdata**

[https://github.com/yasio/ftp\\_server/blob/master/ftp\\_server.cpp#L98](https://github.com/yasio/ftp_server/blob/master/ftp_server.cpp#L98)

**yasio** SIGPIPE

3.30  
iOS, SIGPIPE, APP, TCP, 32: Broken pipe  
<https://github.com/yasio/yasio/issues/170>

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

- **2c5f50:**
  - **APP** **ios14.1+** **Local Network Permission**
  - **10161** **SNMP via TLS**
- **4f7271667a53f74e4a** **APP** **/79f87a536e83f9fa** **49152~65535**
- **76b6c289:** <https://www.speedguide.net/port.php>

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

- **62f0:** **macOS** **UDP** **9126**
- **83616968:** **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**

**2f4e69**

**std::thread**

**4e697689f293fe89c9:** <https://github.com/yasio/yasio/issues/244>

**xxsocket** **\_nfd\_7661f0ff**

**\_nfdnonblock7661f0ff\_i29internal(667262f29669)39**

**8b7e4e** **YOPT\_TCP\_KEEPALIVE**

**468246808916f6437887f8487c7ef6f557a7678676f88c9:** <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` `__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`