

```
1  /**
2   * Copyright (c) 2011-2025 libbitcoin developers (see AUTHORS)
3   *
4   * This file is part of libbitcoin.
5   *
6   * This program is free software: you can redistribute it and/or modify
7   * it under the terms of the GNU Affero General Public License as published by
8   * the Free Software Foundation, either version 3 of the License, or
9   * (at your option) any later version.
10  *
11  * This program is distributed in the hope that it will be useful,
12  * but WITHOUT ANY WARRANTY; without even the implied warranty of
13  * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
14  * GNU Affero General Public License for more details.
15  *
16  * You should have received a copy of the GNU Affero General Public License
17  * along with this program. If not, see <http://www.gnu.org/licenses/>.
18  */
19 #ifndef LIBBITCOIN_NETWORK_MESSAGES_RPC_INTERFACES_STRATUM_V2_HPP
20 #define LIBBITCOIN_NETWORK_MESSAGES_RPC_INTERFACES_STRATUM_V2_HPP
21
22 #include <bitcoin/network/define.hpp>
23 #include <bitcoin/network/messages/rpc/interface.hpp>
24
25 namespace libbitcoin {
26 namespace network {
27 namespace rpc {
28
29 struct stratum_v2_methods
30 {
31     static constexpr std::tuple methods
32     {
33         /// Common setup messages.
34         method<"setup_connection", number_t, number_t, number_t, number_t, string_t, string_t, string_t> ➤
35             { "min_version", "max_version", "flags", "endpoint_port", "endpoint_host", "vendor", ➤
```

```
    "user_agent" },
35     method<"setup_connection_success", number_t, number_t>{ "used_version", "flags" },
36     method<"setup_connection_error", string_t>{ "code" },
37
38     /// Standard mining channel (header-only, primary for devices/firmware).
39     method<"open_standard_mining_channel", string_t, number_t, string_t, number_t>{ "user_identity", ↗
        "nominal_hash_rate", "max_target", "min_extranonce_size" },
40     method<"open_standard_mining_channel_success", number_t, string_t, string_t, number_t> ↗
        { "channel_id", "target", "extranonce_prefix", "extranonce_size" },
41     method<"open_standard_mining_channel_error", string_t>{ "code" },
42
43     /// Extended mining channel.
44     method<"open_extended_mining_channel", string_t, number_t, string_t, number_t>{ "user_identity", ↗
        "nominal_hash_rate", "max_target", "min_extranonce_size" },
45     method<"open_extended_mining_channel_success", number_t, string_t, string_t, number_t, number_t, ↗
        number_t>{ "channel_id", "target", "extranonce_prefix", "extranonce_size", "max_coinbase_size", ↗
        "coinbase_output_max_additional_size" },
46     method<"open_extended_mining_channel_error", string_t>{ "code" },
47
48     /// Channel management.
49     method<"update_channel", number_t, number_t, optional<""_t>>{ "channel_id", "nominal_hash_rate", ↗
        "max_target" },
50     method<"update_channel_error", number_t, string_t>{ "channel_id", "code" },
51     method<"close_channel", number_t, optional<""_t>>{ "channel_id", "reason" },
52     method<"set_target", number_t, string_t>{ "channel_id", "maximum_target" },
53
54     /// Header-only job declaration.
55     method<"new_mining_job", number_t, boolean_t, number_t, string_t, string_t, number_t>{ "job_id", ↗
        "future_job", "version", "prev_hash", "merkle_root", "ntime_offset" },
56     method<"set_new_prev_hash", number_t, number_t, string_t, number_t, number_t>{ "channel_id", ↗
        "job_id", "prev_hash", "min_ntime", "nbits" },
57
58     /// Extended / custom jobs.
59     method<"new_extended_mining_job", number_t, boolean_t, number_t, string_t, string_t, array_t> ↗
        { "job_id", "future_job", "version", "coinbase_prefix", "coinbase_suffix", "merkle_path" },
```

```

...coin-network\include\bitcoin\network\messages\rpc\interfaces\stratum_v2.hpp 3
60     method<"set_custom_mining_job", number_t, boolean_t, number_t, string_t, number_t, number_t,  ↗
        array_t, array_t>{ "channel_id", "future_job", "job_id", "prev_hash", "version", "ntime",  ↗
            "merkle_branch", "transactions" },
61     method<"set_custom_mining_job_success", number_t>{ "job_id" },
62     method<"set_custom_mining_job_error", number_t, string_t>{ "job_id", "code" },
63
64     /// Share submission.
65     method<"submit_shares", number_t, number_t, number_t, number_t, number_t, number_t>{ "channel_  ↗
        id", "sequence_number", "job_id", "nonce", "ntime", "version" },
66     method<"submit_shares_success", number_t, optional<"_t">>{ "sequence_number", "new_target" },
67     method<"submit_shares_error", number_t, string_t>{ "sequence_number", "code" },
68
69     /// Miscellaneous.
70     method<"reconnect", string_t, number_t>{ "new_host", "new_port" }
71 };
72
73 // Derive this from above in c++26 using reflection.
74 using setup_connection = at<0, decltype(methods)>;
75 using setup_connection_success = at<1, decltype(methods)>;
76 using setup_connection_error = at<2, decltype(methods)>;
77 using open_standard_mining_channel = at<3, decltype(methods)>;
78 using open_standard_mining_channel_success = at<4, decltype(methods)>;
79 using open_standard_mining_channel_error = at<5, decltype(methods)>;
80 using open_extended_mining_channel = at<6, decltype(methods)>;
81 using open_extended_mining_channel_success = at<7, decltype(methods)>;
82 using open_extended_mining_channel_error = at<8, decltype(methods)>;
83 using update_channel = at<9, decltype(methods)>;
84 using update_channel_error = at<10, decltype(methods)>;
85 using close_channel = at<11, decltype(methods)>;
86 using set_target = at<12, decltype(methods)>;
87 using new_mining_job = at<13, decltype(methods)>;
88 using set_new_prev_hash = at<14, decltype(methods)>;
89 using new_extended_mining_job = at<15, decltype(methods)>;
90 using set_custom_mining_job = at<16, decltype(methods)>;
91 using set_custom_mining_job_success = at<17, decltype(methods)>;

```

```
92     using set_custom_mining_job_error = at<18, decltype(methods)>;
93     using submit_shares = at<19, decltype(methods)>;
94     using submit_shares_success = at<20, decltype(methods)>;
95     using submit_shares_error = at<21, decltype(methods)>;
96     using reconnect = at<22, decltype(methods)>;
97 };
98
99 using stratum_v2 = interface<stratum_v2_methods>;
100
101 } // namespace rpc
102 } // namespace network
103 } // namespace libbitcoin
104
105 #endif
106
```