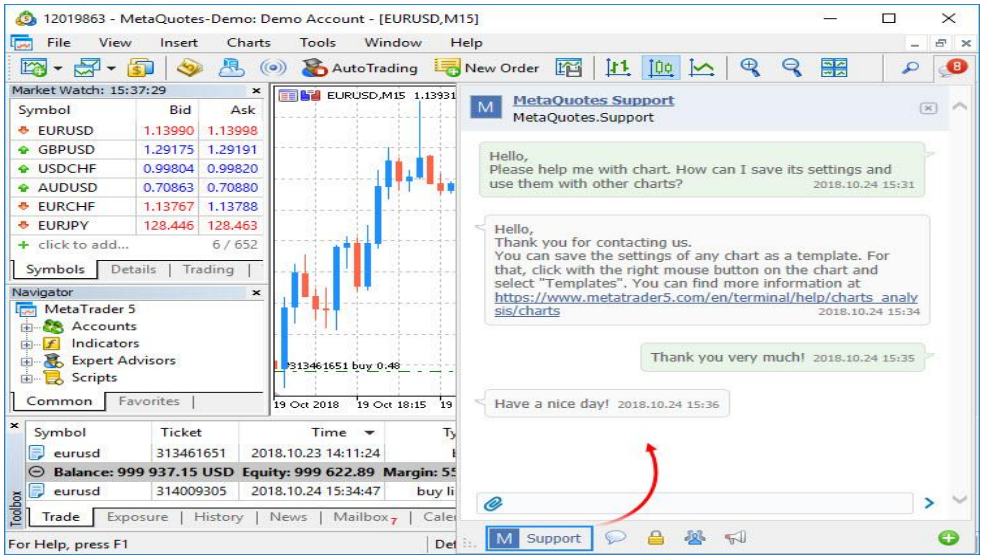
New MetaTrader 5 platform build 1930

Corporate broker accounts in the built-in chat

Now brokers can support their clients using the client terminal's built-in chats.

Traders can directly communicate with broker's technical support team right in the client terminal without deploying additional infrastructure. The technical support account assigned to the trade server remains in high priority at all times and is always available to traders.

Create a separate MQL5 account for your support team and report it via Service Desk so that we can assign it to your trade servers. Your teammates will be able to respond to traders via MetaTrader 5 terminals or on the www.MQL5.com website.

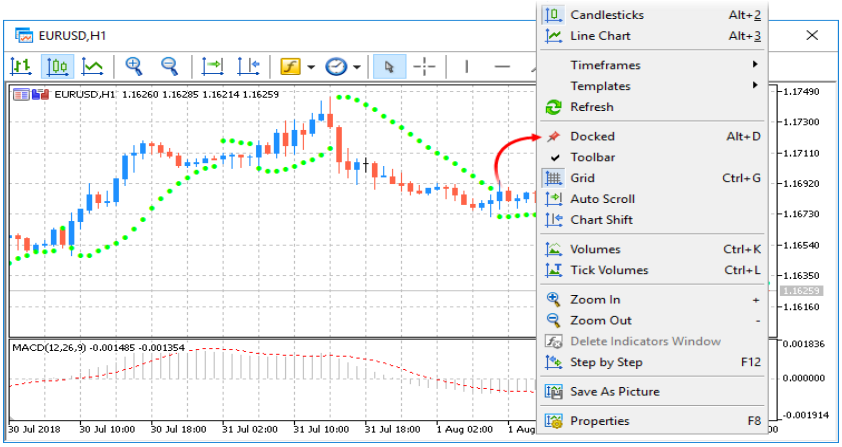


Send your Service Desk requests for opening corporate chat accounts right now!

MetaTrader 5 Client Terminal build 1930

1. Terminal: Now you can detach financial symbol charts from the trading terminal window.

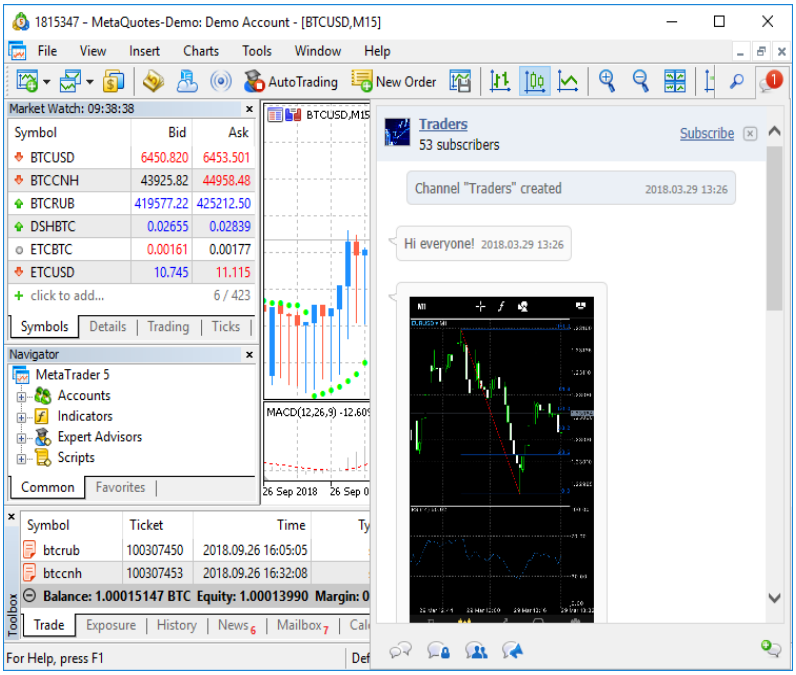
   This feature is convenient when using multiple monitors. Thus, you may set the main platform window on one monitor to control your account state, and move your charts to the second screen to observe the market situation. To detach a chart from the terminal, disable the Docked option in its context menu. After that, move the chart to the desired monitor.



A separate toolbar on detached charts allows applying analytical objects and indicators without having to switch between monitors. Use the toolbar context menu to manage the set of available commands or to hide it.

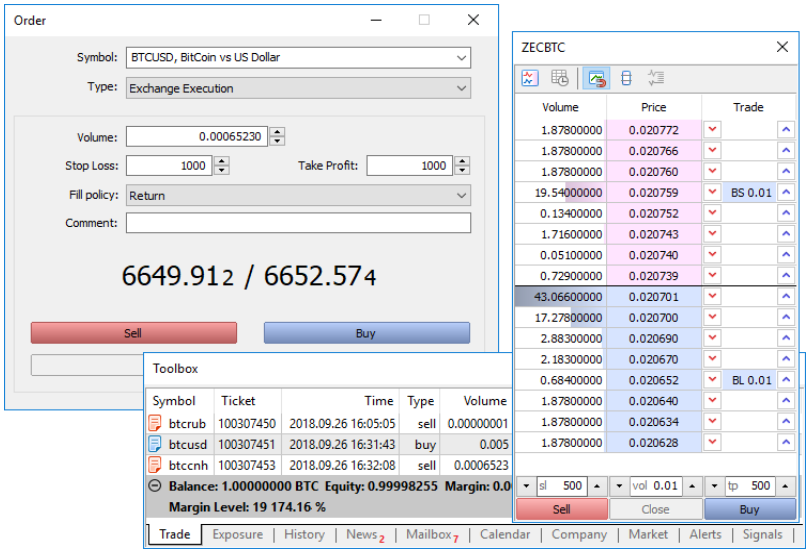
2. Terminal: Fully updated the built-in chats. Now they support group dialogs and channels. Conduct private discussions with a group of people in a unified environment without switching between different dialogs and create channels according to your interests and languages. Communicate with colleagues and friends at MQL5.community without visiting the website.

   Group chats and channels can be public or private. Their creators decide whether it is possible to join them freely or only by invitation. You can also assign moderators to channels and chats for additional communication control.

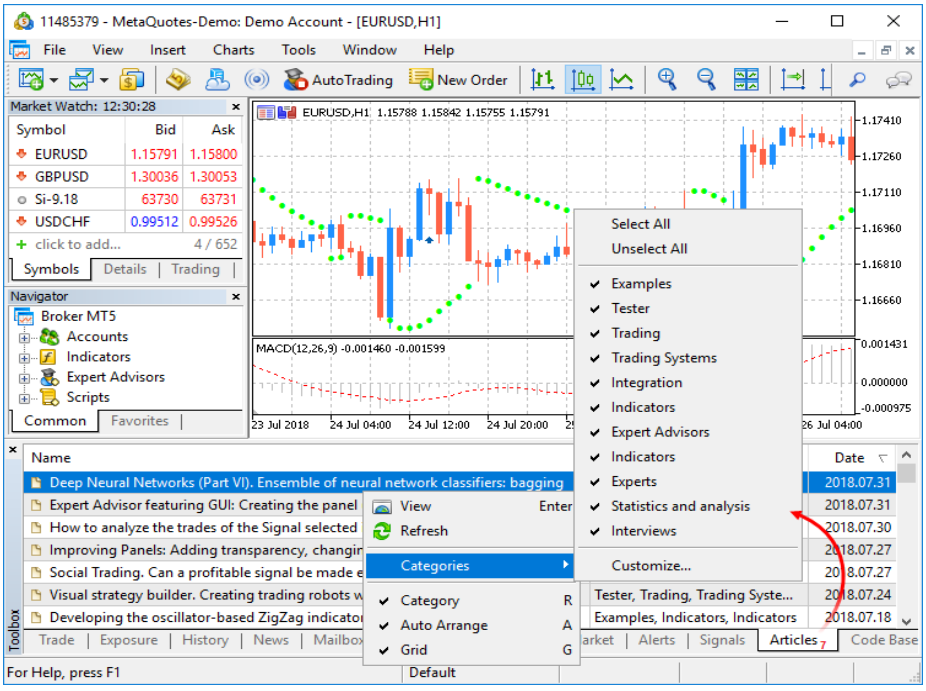


3. Terminal: Added support for extended volume accuracy for cryptocurrency trading. Now the minimum possible volume of trading operations is 0.00000001 lots. The market depth, the time and sales, as well as other interface elements now feature the ability to display volumes accurate to 8 decimal places.

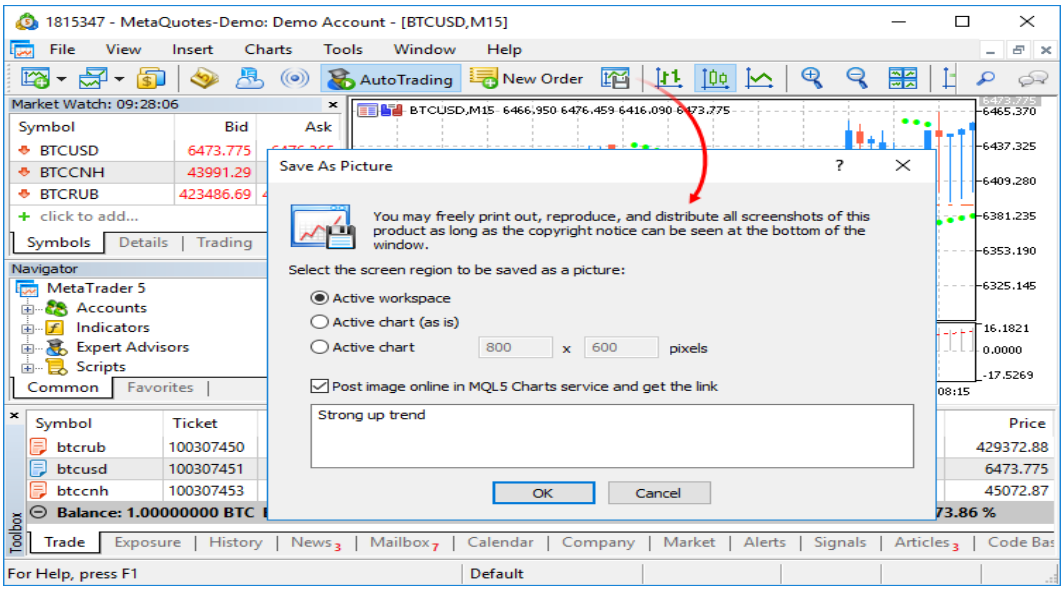
   The minimal volume and its change step depend on financial instrument settings on the broker's side.



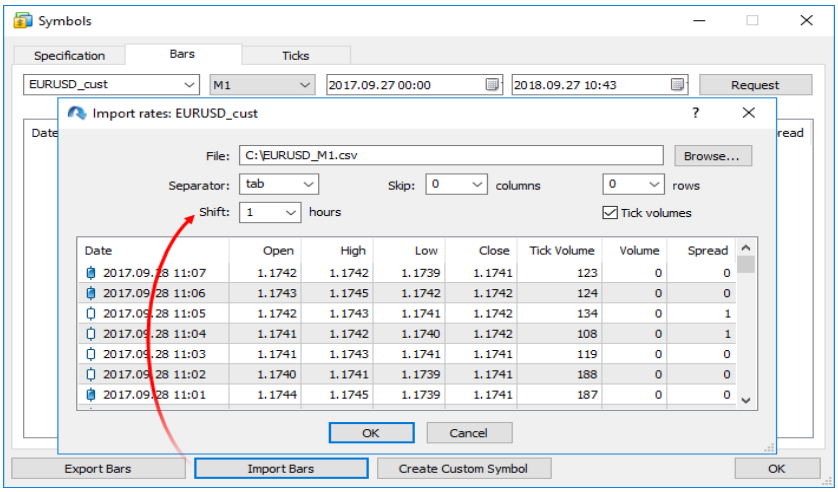
4. Terminal: Added the tab of articles published on MQL5.community to the Toolbox window. Over 600 detailed materials on the development of trading strategies in MQL5 are now available directly in the terminal. New articles are published every week.



5. Terminal: Added support for extended authentication using certificates when working under Wine.
6. Terminal: Fixed display of the market depth when it is limited to one level.
7. Terminal: Added the "Save As Picture" command to the Standard toolbar. Now, it is much easier to take pictures of charts and share them in the community.



8. Terminal: Fixed applying the time shift when importing bars and ticks. Previously, the shift was not applied in some cases.



9. Terminal: Fixed terminal freezing in case of a large amount of economic calendar news.

10. MQL5: Added native support for .NET libraries with "smart" functions import. Now .NET libraries can be used without writing special wrappers — MetaEditor does it on its own.

    To work with .NET library functions, simply import DLL itself without defining specific functions. MetaEditor automatically imports all functions it is possible to work with:
    - Simple structures (POD, plain old data) — structures that contain only simple data types.
    - Public static functions having parameters, in which only simple types and POD structures or their arrays are used

    To call functions from the library, simply import it:

```
#import "TestLib.dll"

//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {
   int x=41;
   TestClass::Inc(x);
   Print(x);
  }
```

    The C# code of the Inc function of the TestClass looks as follows:

```
public class TestClass
{
   public static void Inc(ref int x)
   {
    x++;
   }
}
```

    As a result of execution, the script returns the value of 42.

    The work on support for .NET libraries continues. Their features are to be expanded in the future.

11. MQL5: Added support for working with WinAPI functions to Standard Library. Now, there is no need to import libraries manually and describe function signatures to use operating system functions in an MQL5 program. Simply include the header file from the MQL5\Include\WinAPI directory.

    WinAPI functions are grouped in separate files by their purpose:

    - libloaderapi.mqh — working with resources
    - memoryapi.mqh — working with memory
    - processenv.mqh — working with environment
    - processthreadsapi.mqh — working with processes
    - securitybaseapi.mqh — working with OS security system
    - sysinfoapi.mqh — obtaining system information
    - winbase.mqh — common functions
    - windef.mqh — constants, structures and enumerations
    - wingdi.mqh — working with graphical objects
    - winnt.mqh — working with exceptions
    - winreg.mqh — working with the registry
    - winuser.mqh — working with windows and interface
    - errhandlingapi.mqh — handling errors

- fileapi.mqh — working with files
- handleapi.mqh — working with handles
- winapi.mqh — including all functions (WinAPI header files)

The binding works only with the 64-bit architecture.

12. MQL5: Added support for the inline, __inline and __forceinline specifiers when parsing code. The presence of the specifiers in the code causes no errors and does not affect the compilation. At the moment, this feature simplifies transferring C++ code to MQL5.
    Find more information about specifiers in MSDN.

13. MQL5: Significantly optimized execution of MQL5 programs. In some cases, the performance improvement can reach 10%. Recompile your programs in the new MetaEditor version to make them run faster.

    Unfortunately, new programs will not be compatible with previous terminal versions due to this additional optimization. Programs compiled in MetaEditor version 1910 and later cannot be launched in terminal version 1880 and below. Programs compiled in earlier MetaEditor versions can run in new terminals.

14. MQL5: Significantly optimized multiple MQL5 functions.
15. MQL5: Added new properties for attaching/detaching charts from the terminal main window and managing their position.

    Added the following properties to the ENUM_CHART_PROPERTY_INTEGER enumeration:

    - CHART_IS_DOCKED — the chart window is docked. If set to 'false', the chart can be dragged outside the terminal area.
    - CHART_FLOAT_LEFT — the left coordinate of the undocked chart window relative to the virtual screen.
    - CHART_FLOAT_TOP — the upper coordinate of the undocked chart window relative to the virtual screen.
    - CHART_FLOAT_RIGHT — the right coordinate of the undocked chart window relative to the virtual screen.
    - CHART_FLOAT_BOTTOM — the bottom coordinate of the undocked chart window relative to the virtual screen.

    Added the following functions to the ENUM_TERMINAL_INFO_INTEGER enumeration:

    - TERMINAL_SCREEN_LEFT — the left coordinate of the virtual screen. A virtual screen is a rectangle that covers all monitors. If the system has two monitors ordered from right to left, then the left coordinate of the virtual screen can be on the border of two monitors.
    - TERMINAL_SCREEN_TOP — the top coordinate of the virtual screen.
    - TERMINAL_SCREEN_WIDTH — terminal width.
    - TERMINAL_SCREEN_HEIGHT — terminal height.
    - TERMINAL_LEFT — the left coordinate of the terminal relative to the virtual screen.
    - TERMINAL_TOP — the top coordinate of the terminal relative to the virtual screen.
    - TERMINAL_RIGHT — the right coordinate of the terminal relative to the virtual screen.
    - TERMINAL_BOTTOM — the bottom coordinate of the terminal relative to the virtual screen.

16. MQL5: Added the volume_real field to the MqlTick and MqlBookInfo structures. It is designed to work with extended accuracy volumes. The volume_real value has a higher priority than 'volume'. The server will use this value, if specified.

```
struct MqlTick
 {
  datetime        time;         // Last price update time
  double          bid;          // Current Bid price
  double          ask;          // Current Ask price
  double          last;         // Current price of the Last trade
  ulong           volume;       // Volume for the current Last price
  long            time_msc;     // Last price update time in milliseconds
  uint            flags;        // Tick flags
  double          volume_real;  // Volume for the current Last price with greater accuracy
 };
```

```
struct MqlBookInfo
  {
   ENUM_BOOK_TYPE    type;             // order type from the
ENUM_BOOK_TYPE enumeration
   double            price;            // price
   long              volume;           // volume
   double            volume_real;      // volume with greater accuracy
  };
```

17. MQL5: Added new properties to the ENUM_SYMBOL_INFO_DOUBLE enumeration:

    - SYMBOL_VOLUME_REAL — the volume of the last executed deal;
    - SYMBOL_VOLUMEHIGH_REAL — the highest deal volume for the current day;
    - SYMBOL_VOLUMELOW_REAL — the lowest deal volume for the current day.

    Use the SymbolInfoDouble function to get these properties.

18. MQL5: Added the MQL_FORWARD property to the ENUM_MQL_INFO_INTEGER enumeration — forward test mode flag.
19. MQL5: Added the pack(integer_value) property for structures. It allows you to set the alignment of the fields arrangement within a structure, which can be necessary when working with DLL. The values of 1, 2, 4, 8 and 16 are possible for integer_value.
    If the property is not defined, the default alignment of 1 byte is used — pack(1).

    Example of use:

```
20. //+------------------------------------------------------------------+
21. //| Default packing                                                   |
22. //+------------------------------------------------------------------+
23. struct A
24.   {
25.    char              a;
26.    int               b;
27.   };
28. //+------------------------------------------------------------------+
29. //| Specified packing                                                |
30. //+------------------------------------------------------------------+
31. struct B pack(4)
```

```
32.    {
33.     char              a;
34.     int               b;
35.    };
36.  //+------------------------------------------------------------------+
37.  //| Script program start function                                    |
38.  //+------------------------------------------------------------------+
39.  void OnStart()
40.    {
41.     Print("sizeof(A)=",sizeof(A));
42.     Print("sizeof(B)=",sizeof(B));
43.    }
   //+------------------------------------------------------------------+
```

Conclusion:

```
sizeof(A)=5
sizeof(B)=8
```

Find more information about alignment within structures in MSDN.

44. MQL5: Relaxed requirements for casting enumerations. In case of an implicit casting, the compiler automatically substitutes the value of a correct enumeration and displays a warning.

For the following code:

```
45.  enum Main
46.    {
47.     PRICE_CLOSE_,
48.     PRICE_OPEN_
49.    };
50.
51.  input Main Inp=PRICE_CLOSE;
52.  //+------------------------------------------------------------------+
53.  //| Start function                                                   |
54.  //+------------------------------------------------------------------+
55.  void OnStart()
56.    {
     }
```

The compiler displays the warning:

implicit conversion from 'enum ENUM_APPLIED_PRICE' to 'enum Main'
'Main::PRICE_OPEN_' instead of 'ENUM_APPLIED_PRICE::PRICE_CLOSE' will be used

Previously, the following error was generated in that case:

'PRICE_CLOSE' - cannot convert enum

The compiler will still display the error if enumerations are used incorrectly in the function parameters.

57. MQL5: Fixed compilation of template functions. Now, when using overloaded template functions, only the necessary overload, rather than all existing ones, is instantiated.

```
58.  class X {  };
59.
60.  void f(int)  {  }
```

```
61.
62.  template<typename T>
63.  void a(T*) { new T(2); }  // previously, the compiler generated the
   error here
64.
65.  template<typename T>
66.  void a()  { f(0); }
67.
68.
   void OnInit()  { a<X>(); }
```

69. MQL5: Optimized some cases of accessing tick history via the CopyTicks* functions.
70. MQL5: Added the new TesterStop function allowing for early completion of a test/optimization pass. When calling it, the entire trading statistics and OnTester result are passed to the client terminal just like during the normal test/optimization completion.
71. MQL5: Added the new property for custom indicators #property tester_everytick_calculate. It is used in the strategy tester and allows for forced indicator calculation at each tick.
72. Tester: Now, in case of a non-visual test/optimization, all used indicators (standard and custom ones) are calculated only during a data request. The exceptions are indicators containing the EventChartCustom function calls and applying the OnTimer handler. Previously, all indicators were unconditionally calculated in the strategy tester at each incoming tick (even from some other instrument). The new feature significantly accelerates testing and optimization.

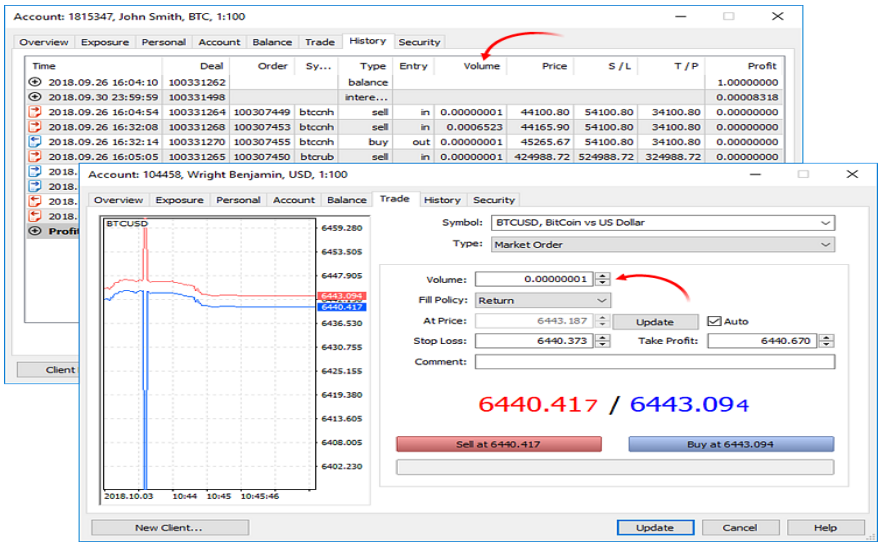
    To enable the forced indicator calculation at each tick, add the #property tester_everytick_calculate property for the program.
    Indicators compiled using the previous compiler versions are calculated as before — at each tick.

73. Tester: Fixed calculating the deposit currency accuracy when testing/optimizing and generating relevant reports.
74. Tester: Optimized and accelerated the strategy tester operation.
75. Tester: Fixed a few testing and optimization errors.
76. MetaEditor: Fixed search for entire words. Now when searching, the underscore is counted as a regular character, rather than a word delimiter.
77. Updated documentation.

MetaTrader 5 Manager build 1930

1. Added support for extended volume accuracy for cryptocurrency trading. Now the minimum possible volume of trading operations is 0.00000001 lots. The market depth, account history, trading dialogs as well as other interface elements now feature the ability to display volumes accurate to 8 decimal places.

    The minimal volume and its change step depend on financial instrument settings on the server.

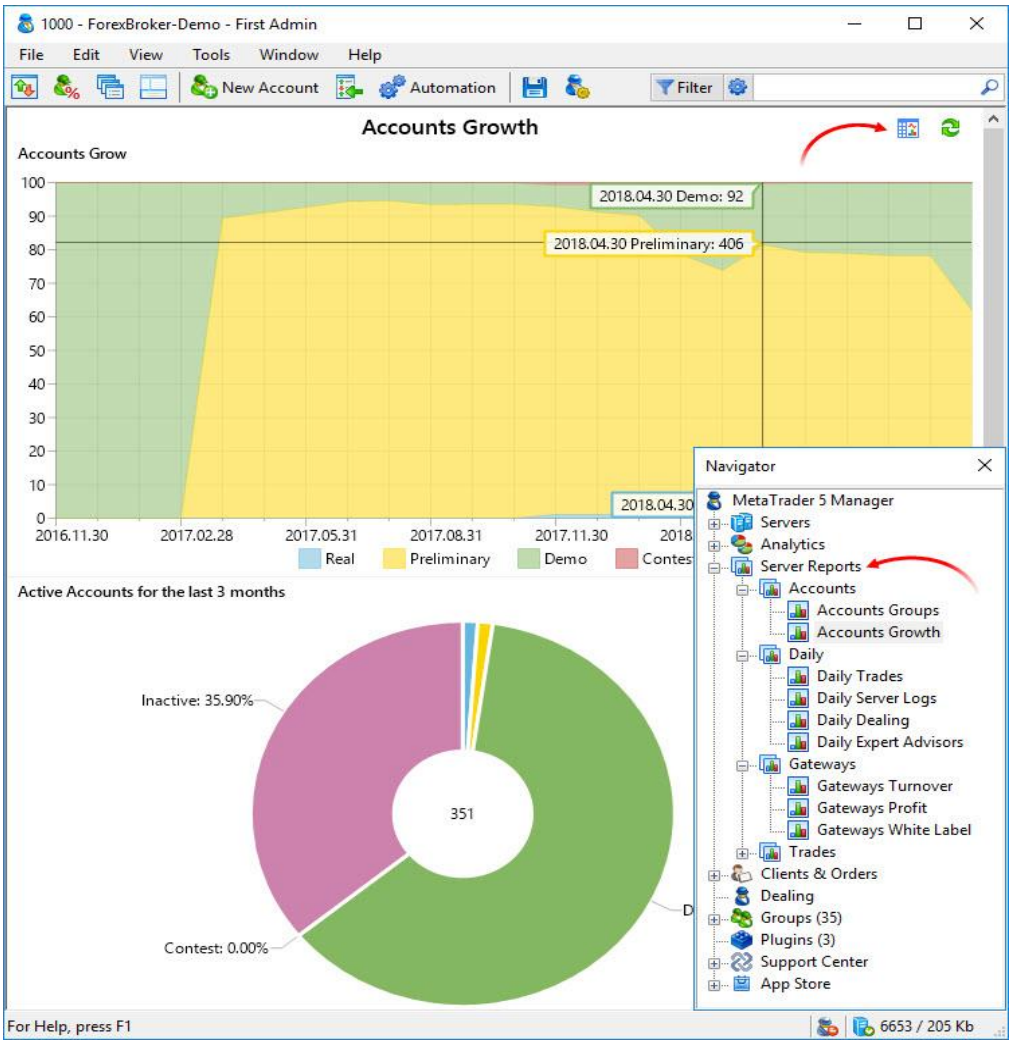


2. Significantly revised the work with server reports.

   Updated design — reports are now more convenient to use. All reports are now grouped into categories, and a tree list is now displayed instead of a usual one in the Navigator window.

   A new type of reporting has been added — Dashboards. To switch between the previous view and the new one, use the icon in the upper right corner of the section.

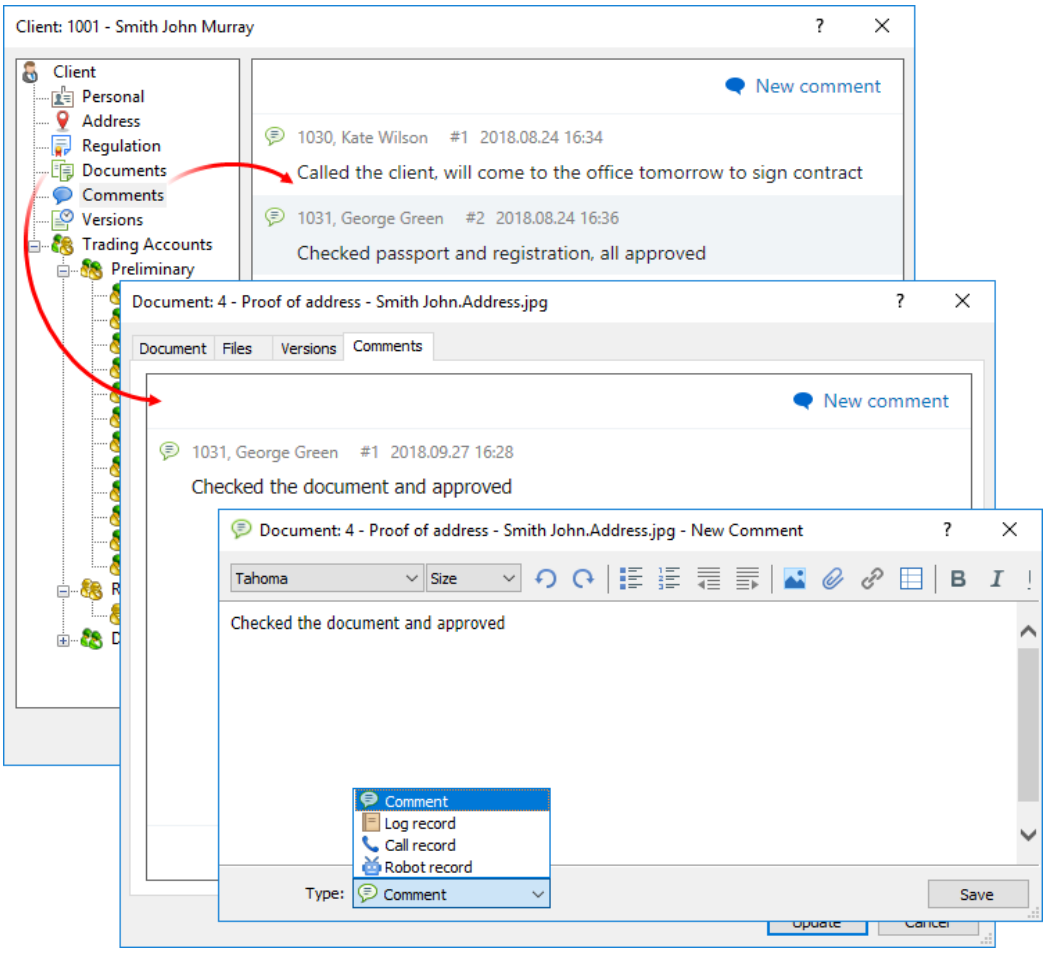
   Now you do not have to request reports manually. They are displayed immediately upon opening due to a new data caching system on the Manager terminal side. To request new data, change the report settings or click the update icon in the upper right corner.



The dashboard presentation is implemented only for standard reports. You can implement it in your own reports using the new functions of the MetaTrader 5 Report API.

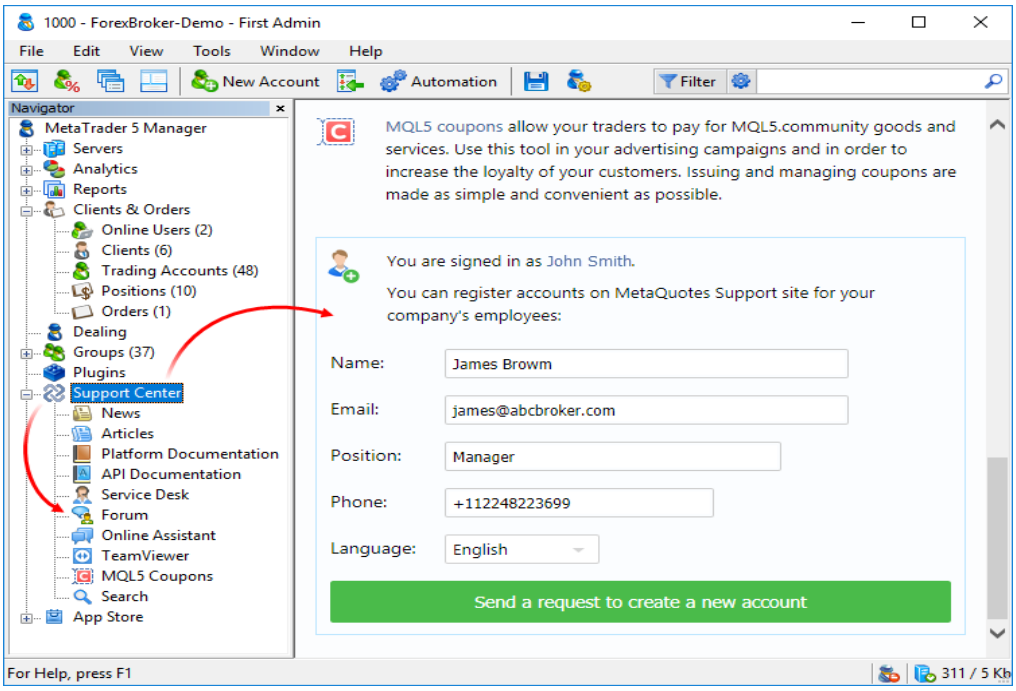
3. Added ability to comment on client records and uploaded documents in the client management system.

   Now your sales team can keep a history of communication with a client directly in the platform, while your supervisory staff can send instructions to managers to contact the client if his or her documents are not in order.

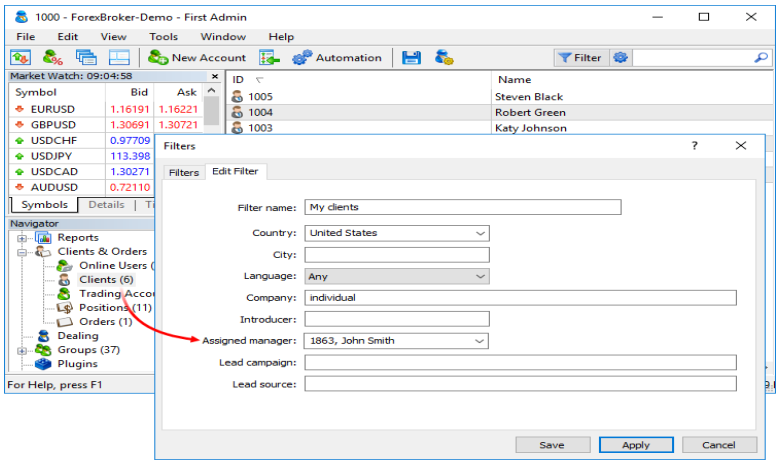


4. Fully updated the Support Center section design and improved its functionality.

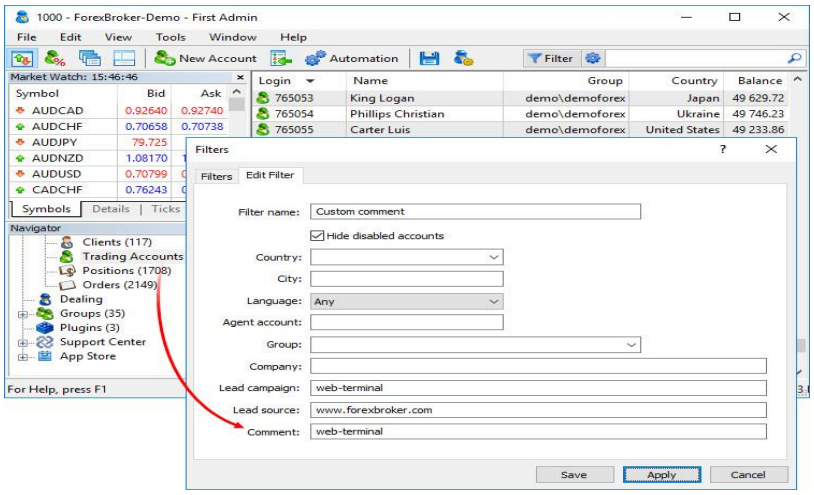
   Now it features the support website forum. Communicate with your teammates and find answers to questions without leaving the terminal. The articles and documentation sections now feature subcategories. Also, if you have sufficient rights, you can request accounts for your colleagues on the support website. To do this, simply fill out a simple form. Our technical support team will check the request and create an account. Its details will be sent to the email address specified in the request.



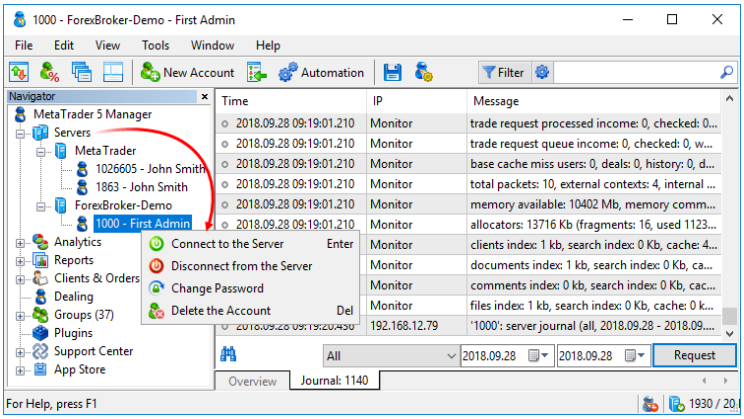
5. Added the client filter by an assigned manager. It is useful for controlling the performance of managers, as well as for arranging their work.



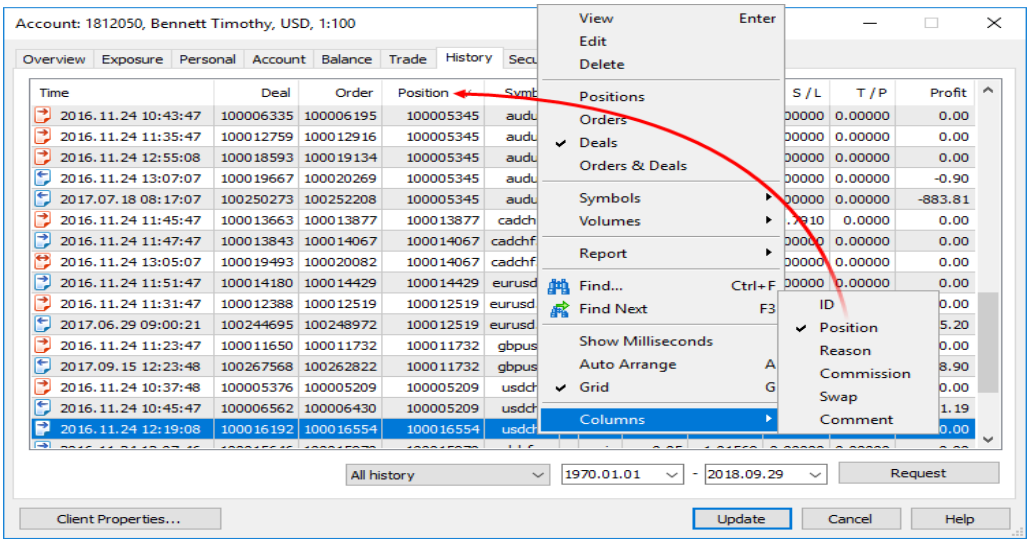
6. Added account sorting by the Comment field. It will allow you to view accounts according to your own internal criteria.



7. Moved server and account selection, as well as the connection commands from the toolbar to Navigator for more convenience.

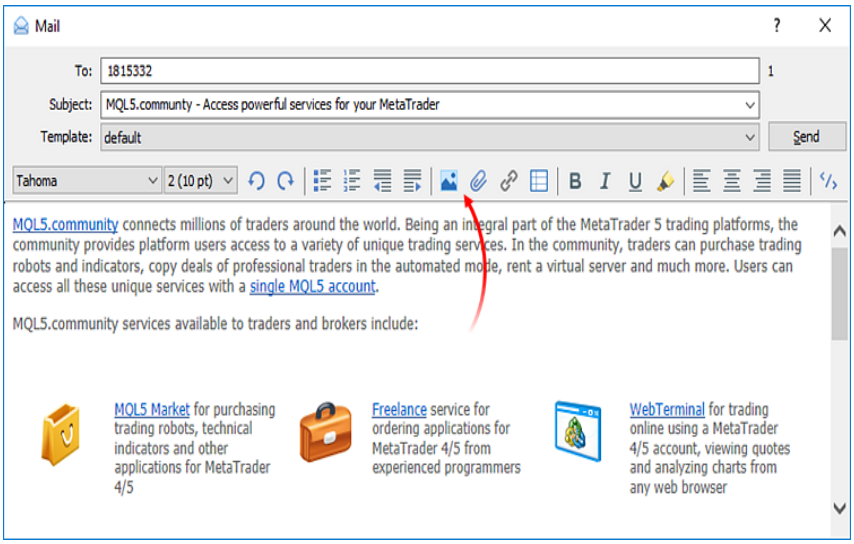


8. Added support for the #LEVERAGE# macro to internal emails. It inserts the email recipient's current leverage into the email.
9. Added display of ID of the positions related to corresponding trading operations to the history of orders and deals.



10. Fixed order price auto update in the dialog of trading on behalf of clients. The error occurred when a client group featured a spread markup.
11. Updated design of the message editor in the email and news sections:

    - Updated all icons
    - Removed the tabs for switching between views (edit, view, source code). Now the view is switched via the toolbar



12. Optimized and considerably accelerated the work of the Manager terminal featuring a large quantity (millions of records) of trading accounts, orders and open positions.
13. Optimized and accelerated sorting trading accounts, orders and positions in case of a large number (one million and more) of records.

14. Removed the margin size check for funds debit operations of Correction type. Previously, this operation did not allow debiting funds in an amount exceeding the current free margin.

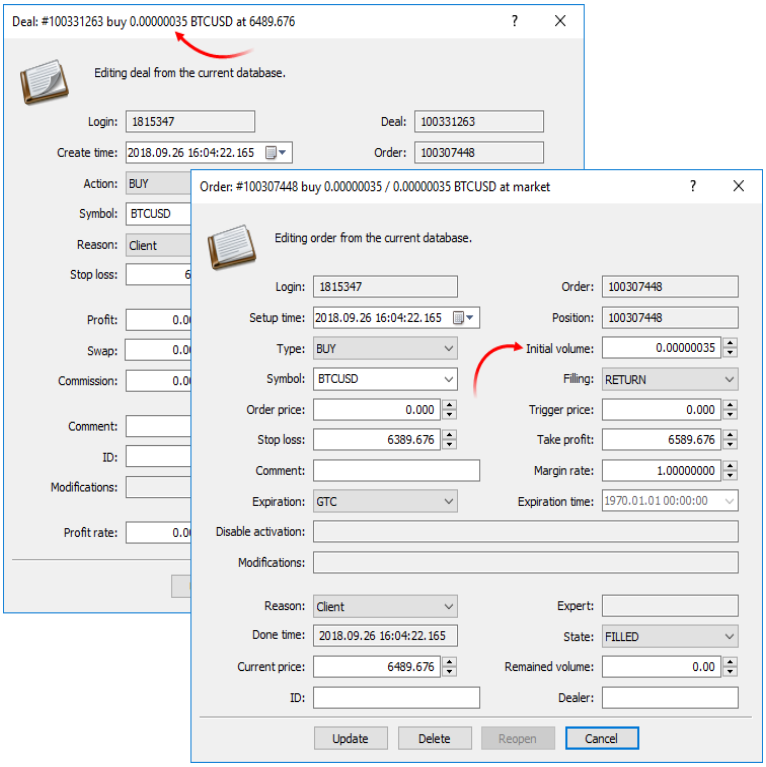
    Be careful during a debit operation. If a client has open positions, and you debit the funds exceeding the free margin amount, a stop out is activated on the account.

15. Fixed exporting trading accounts, orders and positions. An error could occur when exporting a large number (one million and more) of records.
16. Fixed errors reported in crash logs.

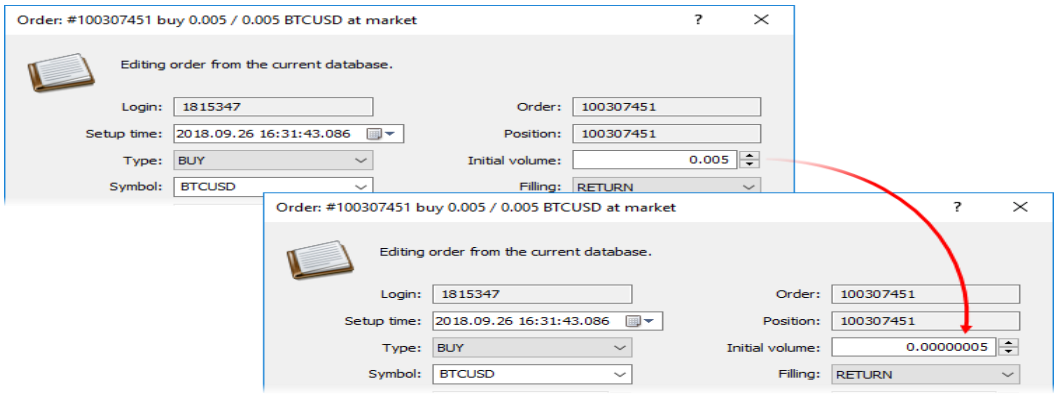
MetaTrader 5 Administrator build 1930

1. Added support for extended volume accuracy for cryptocurrency trading. Now the minimum possible volume of trading operations is 0.00000001 lots. The orders, deals and positions dialog, as well as other interface elements now feature the ability to display volumes accurate to 8 decimal places.

    The minimal volume and its change step depend on financial instrument settings on the server.

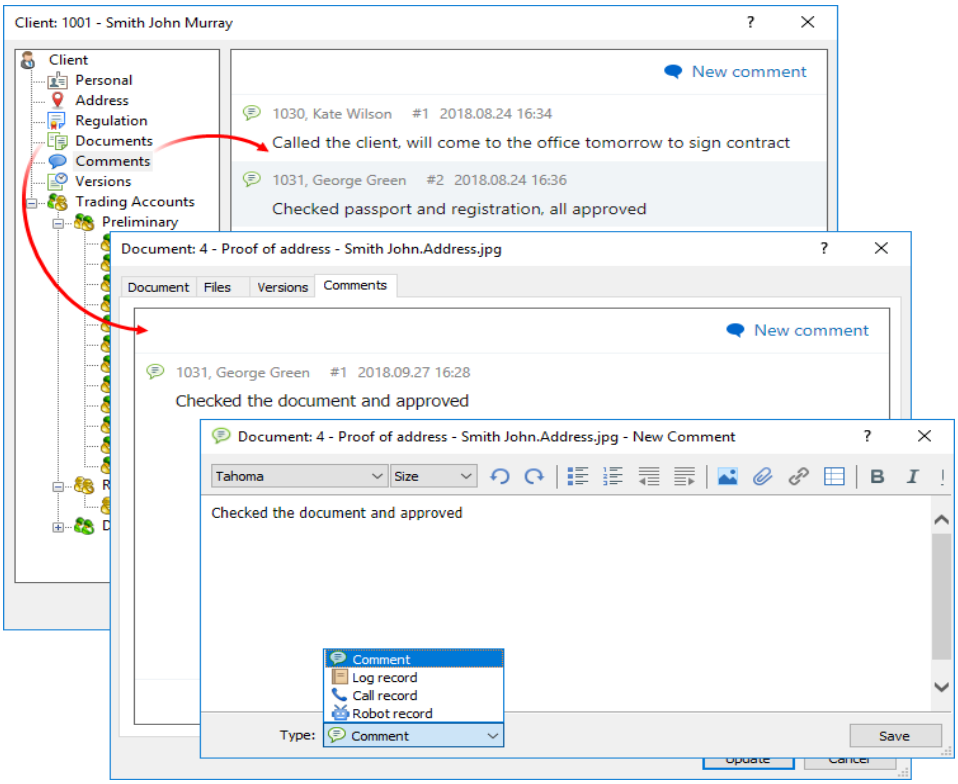


Even if the field initially displays a volume with less accuracy, you can correct it manually:



2. Added ability to comment on client records and uploaded documents in the client management system.

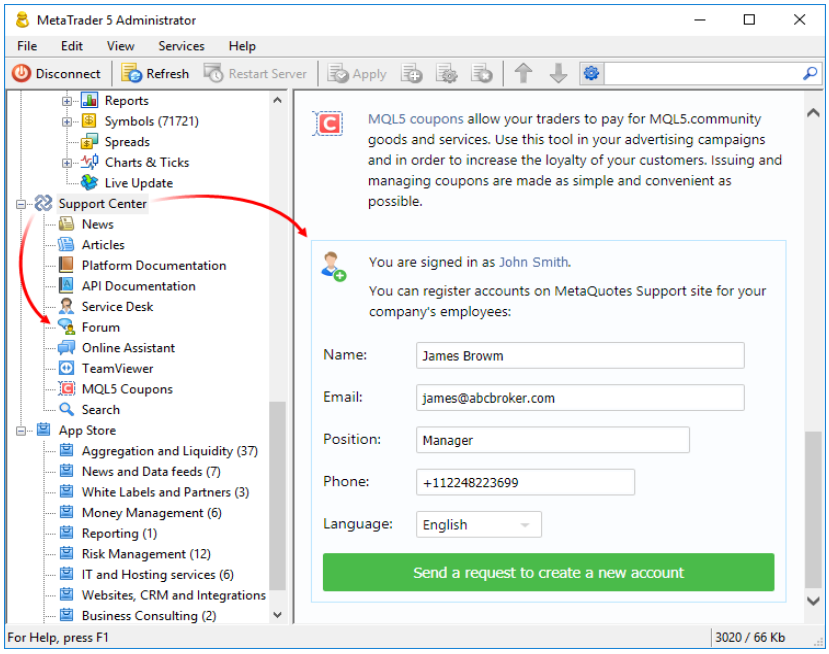
   Now your sales team can keep a history of communication with a client directly in the platform, while your supervisory staff can send instructions to managers to contact the client if his or her documents are not in order.



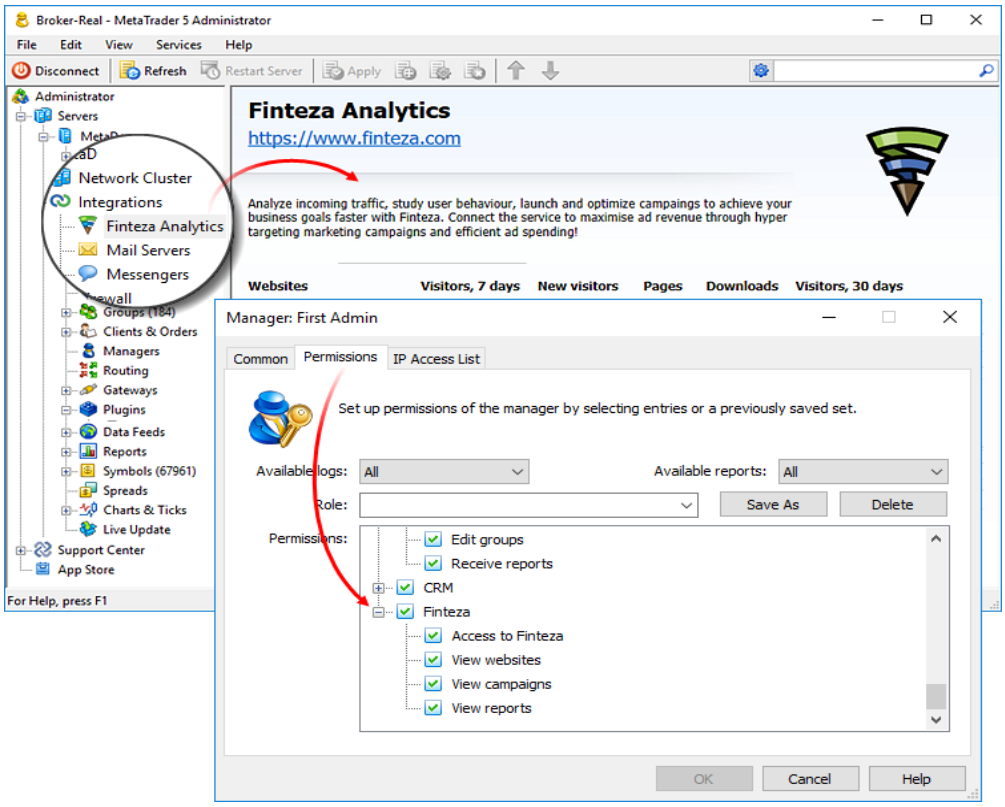
3. Fully updated the Support Center section design and improved its functionality.

   Now it features the support website forum. Communicate with your teammates and find answers to questions without leaving the terminal. The articles and documentation sections now feature subcategories.

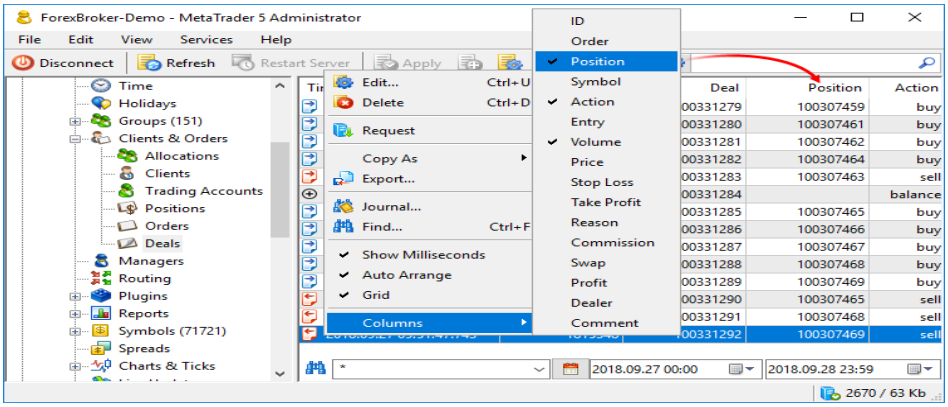
Also, if you have sufficient rights, you can request accounts for your colleagues on the support website. To do this, simply fill out a simple form. Our technical support team will check the request and create an account. Its details will be sent to the email address specified in the request.



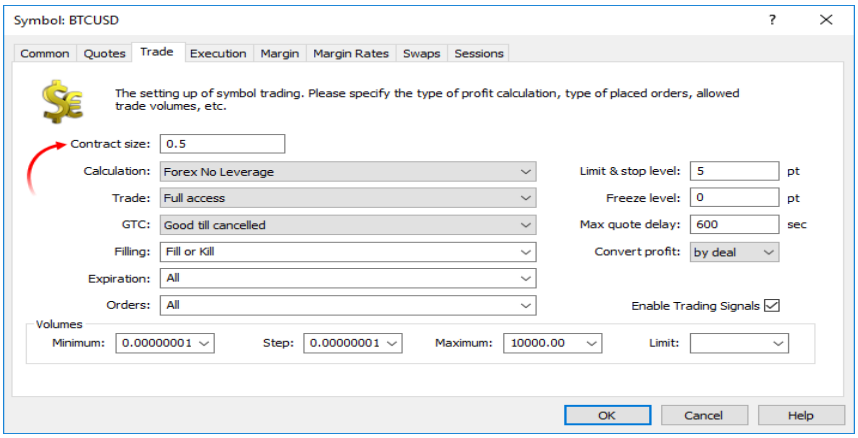
4. Simplified access to Finteza advertising analytics system data via the trading cluster. You no longer need to specify your Finteza account login and password in the Integration section of the Administrator terminal. If a company the server belongs to has subscription to Finteza, the trading cluster automatically gains access to data. You just have to provide access to manager accounts.



5. Added display of ID of the positions related to corresponding trading operations to the list of orders and deals.



6. Added ability to specify fractional values of a contract size. This ability is useful when arranging cryptocurrency trading.

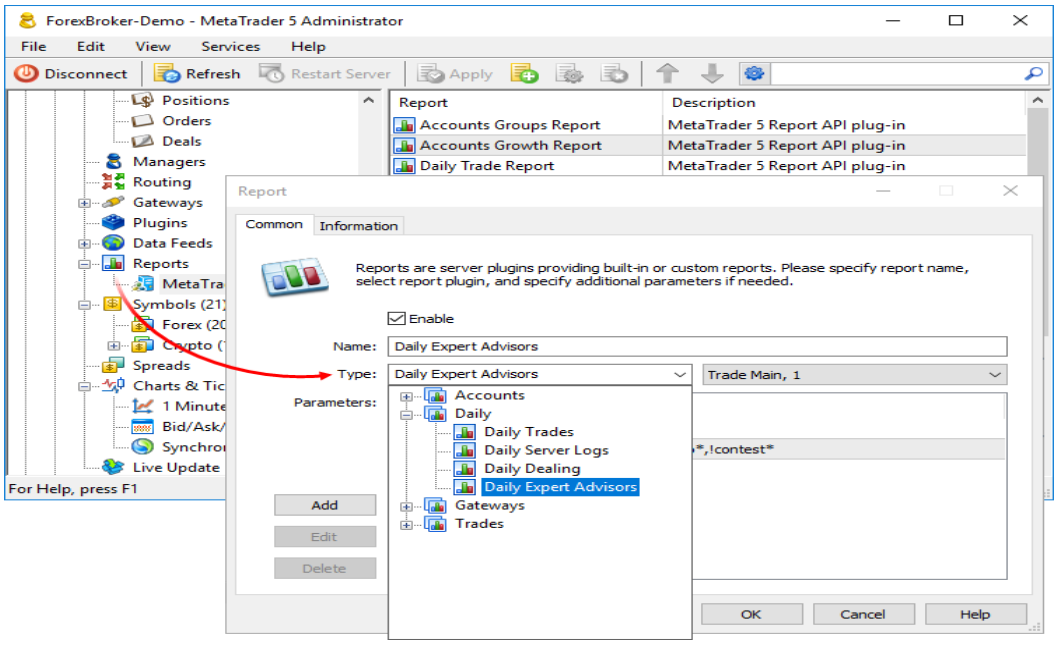


7. Fixed the calculation of the servers performance graphs. Sometimes, the values on higher timeframes (M5, M15, H1, D1) did not match the values on the lower ones. Now they are calculated correctly.

   Depending on the parameter type, its minimum, maximum or total value is displayed on the graphs when switching to higher timeframes:

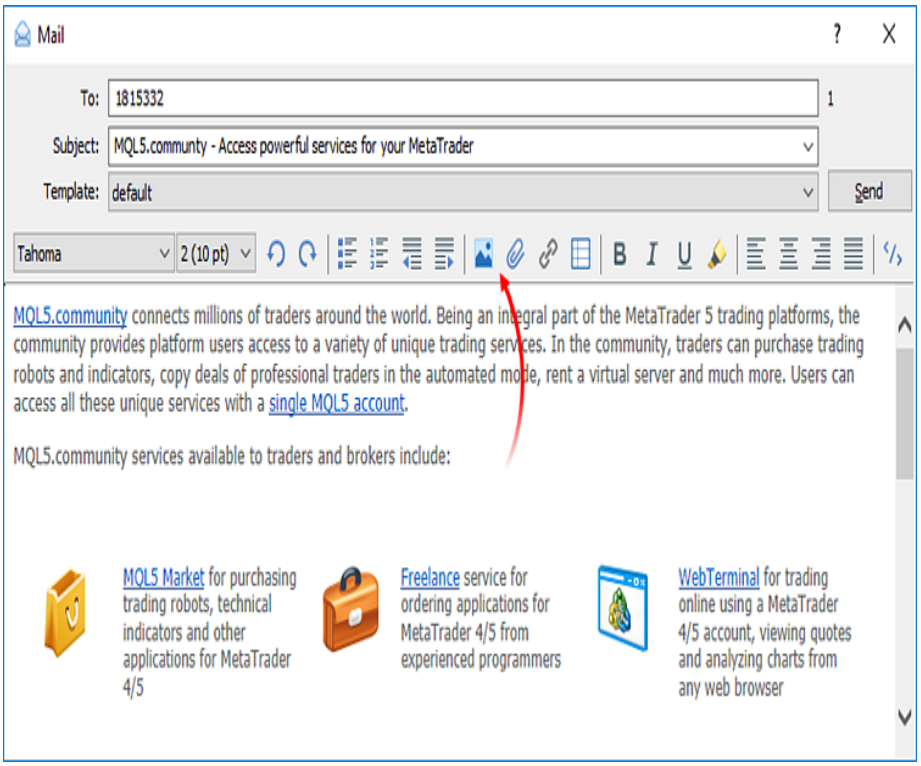
   - CPU — maximum parameter values.
   - Memory — the lowest (for free memory and the memory block) and the highest (for the memory allocated by the server) parameter values.
   - Disk — the lowest (for free space) and the highest (for the queue length and speed) parameter values.
   - Connections — the highest (for connections) and total (for sockets and blocked connections) parameter values.
   - Network Packets — the highest (for speed) and total (for packets) parameter values.
   - Trade — the highest (for accounts) and total (for errors, deals and requests) parameter values.
   - History — the highest (for data feeds and gateways) and total (for price data) parameter values.

8. Updated the report configuration section. Instead of a template name and a module, the list now contains the report description. When selecting a template, a tree list divided into modules appears instead of a conventional one.

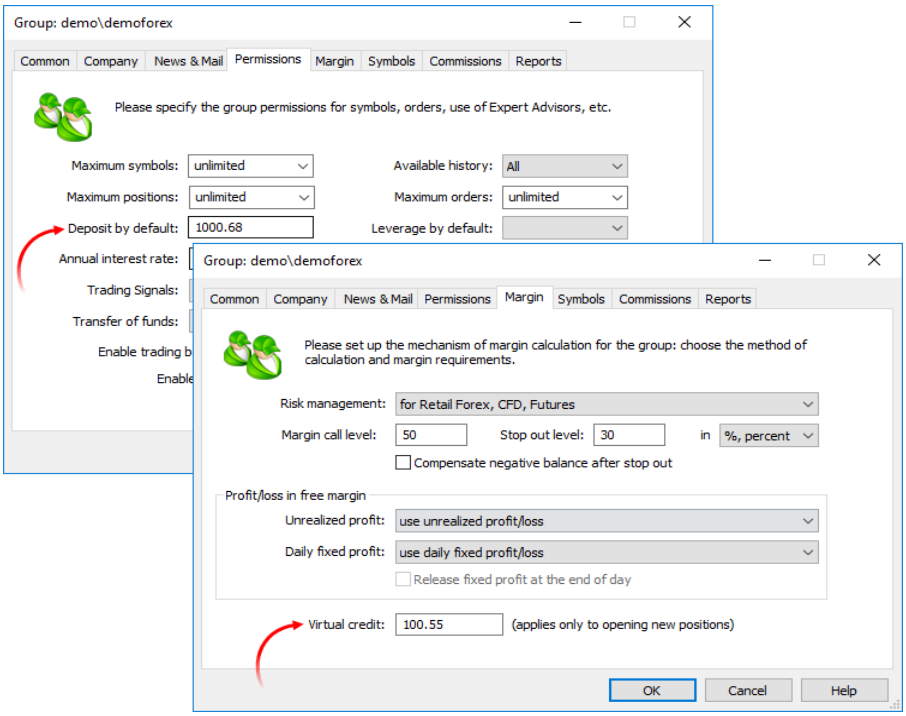


9. Updated design of the message editor in the email and news section:

    - Updated all icons
    - Removed the tabs for switching between views (edit, view, source code). Now the view is switched via the toolbar



10. Fixed the limitation of a gateway address maximum length. Now, a gateway address can be up to 127 characters long.
11. Fixed entering values to the virtual credit and default deposit fields in the client group parameters. Now the accuracy of their input is determined by the accuracy of the group's deposit currency.



12. Fixed export and import of time settings in JSON format.
13. Optimized connection to a trade server featuring a large number of trading instruments (tens of thousands of records).
14. Fixed errors reported in crash logs.

MetaTrader 5 Trade Server build 1930

1. Added support for extended volume accuracy for cryptocurrency trading. Now the minimum possible volume of trading operations is 0.00000001 lots.
2. Fixed checks of deviation on Instant Execution, in case of a client deviation request and the request price better than a market one — in some cases, the server did not requote such requests.
3. Added support for a new Dashboard view of reports.
4. Optimized and significantly accelerated managing trading history.
5. Fixed logging of dealer's trade requests confirmation.
6. Fixed trading requests handling error that caused "try to confirm unknown request" messages in the trade server journal.
7. Optimized and significantly accelerated the recalculation of the visibility scope of trading tools in the client group settings. An error could occur in case of a large number of trading instruments (tens of thousands of records).
8. Fixed trading account's assets calculation if positions on Collateral instruments are present. An error could occur on accounts in groups having the "for Retail Forex, CFD, Futures with hedging" margin calculation type.
9. Fixed deleting the settings of trading instruments and client groups via Web API.
10. Fixed errors reported in crash logs.

MetaTrader 5 History Server build 1930

1. Added support for extended volume accuracy for cryptocurrency trading. Now the minimum possible volume of trading operations is 0.00000001 lots.
2. Fixed an error that could occasionally make price history split impossible to execute.
3. Added minimum/maximum spread control for OTC instruments with the enabled market depth (in addition to the filtration parameters control).
4. Disabled applying price filtration to instruments with the enabled Market Depth and the Last price-based charting mode.
5. Fixed errors reported in crash logs.

MetaTrader 5 Access Server build 1930

1. Added support for extended volume accuracy for cryptocurrency trading. Now the minimum possible volume of trading operations is 0.00000001 lots.
2. Fixed errors reported in crash logs.

MetaTrader 5 Backup Server build 1930

1. Added support for extended volume accuracy for cryptocurrency trading. Now the minimum possible volume of trading operations is 0.00000001 lots.
2. Fixed updating the status of clients' trading accounts when exporting data to SQL (mt5_accounts table).
3. Fixed errors reported in crash logs.

MetaTrader 5 Server API build 1930

1. Added new paired methods for working with volumes of extended accuracy to all interfaces featuring the methods for working with volumes. For example:

    - IMTConfirm::Volume, IMTExecution::DealVolume — standard accuracy volume (4 decimal places)
    - IMTConfirm::VolumeExt, IMTExecution::DealVolumeExt — extended accuracy volume (8 decimal places)

    All increased volume accuracy methods have names with the "Ext" addition.

    In all methods which use standard accuracy, the volume is indicated as multiplied by 10,000. For example, 100 means 0.01 lots. In all extended accuracy methods, the volume is indicated as multiplied by 100,000,000. For example, 10,000,000 means 0.1 lot.

    On the platform side, volume storage is not divided into two entities (normal and increased accuracy). For example, IMTConfirm::Volume and IMTConfirm::VolumeExt methods read and set the same value on the platform side. If you set 1-lot volume using the IMTConfirm::Volume(10000) set-method, the same 1-lot volume will be obtained via the IMTConfirm::VolumeExt get-method, but it will be represented as a value of 100,000,000.

    All the older methods using standard accuracy volumes will continue to operate without changes.

    New methods in common interfaces

    - IMTConfirm::VolumeExt — get and set the extended accuracy volume, in which the trade request was confirmed.

- IMTRequest::VolumeExt — get and set the extended accuracy operation volume in a request.
- IMTRequest::ResultVolumeExt — get the extended accuracy deal volume confirmed by a dealer for this request.

- IMTExecution::OrderVolumeExt — get and set the extended accuracy order volume in lots.
- IMTExecution::DealVolumeExt — get and set the extended accuracy deal volume in lots.
- IMTExecution::DealVolumeRemaindExt — get and set the extended accuracy remaining order volume.

- IMTOrder::VolumeInitialExt — get and set the extended accuracy initial volume of an order.
- IMTOrder::VolumeCurrentExt — get and set the extended accuracy current unfilled order volume.

- IMTDeal::VolumeExt — get and set the extended accuracy deal volume.
- IMTDeal::VolumeExtClosed — get and set the extended accuracy position volume that was closed by the deal.

- IMTPosition::VolumeExt — get and set the extended accuracy trading position volume.

- IMTConGroupSymbol::VolumeMinExt — get and set the extended accuracy minimum volume of trade operations for a symbol for the group.
- IMTConGroupSymbol::VolumeMinExtDefault — get the default minimum value of the extended accuracy volume of trade operations for a symbol.
- IMTConGroupSymbol::VolumeMaxExt — get and set the extended accuracy maximum volume of trade operations for a symbol for the group.
- IMTConGroupSymbol::VolumeMaxExtDefault — get the default maximum value of the extended accuracy volume of trade operations for a symbol.
- IMTConGroupSymbol::VolumeStepExt — get and set the step of change of the extended accuracy trade operations volume for the group symbol.
- IMTConGroupSymbol::VolumeStepExtDefault — get the default value of the change step for the extended accuracy trade operations volume for a symbol.
- IMTConGroupSymbol::VolumeLimitExt — get and set the extended accuracy maximum aggregate volume of positions and orders for a symbol in one direction for this group.
- IMTConGroupSymbol::VolumeLimitExtDefault — get the default extended accuracy maximum aggregate volume of positions and orders for a symbol in one direction.
- IMTConGroupSymbol::IEVolumeMaxExt — get and set the extended accuracy maximum volume of a trade operation that can be executed in the instant execution mode.
- IMTConGroupSymbol::IEVolumeMaxExtDefault — get the extended accuracy default maximum volume of a trade operation that can be executed in the instant execution mode.

- IMTConRoute::ParamVolumeExt — get and set the value of an extended accuracy additional parameter that expresses the volume.

- IMTConCondition::ValueVolumeExt — get and set the value of an extended accuracy condition that expresses the volume.

- IMTConSymbol::VolumeMinExt — get and set the extended accuracy minimum volume allowed for trading operations on a symbol.
- IMTConSymbol::VolumeMaxExt — get and set the extended accuracy maximum volume allowed for trading operations on a symbol.
- IMTConSymbol::VolumeStepExt — get and set the change step of the extended accuracy volume allowed for trading operations on a symbol.
- IMTConSymbol::VolumeLimitExt — get and set the extended accuracy maximum aggregate volume of positions and orders for a symbol in one direction.
- IMTConSymbol::IEVolumeMaxExt — get and set the extended accuracy maximum volume of a trade operation that can be executed in the instant execution mode.

New methods in the server API interface

- IMTServerAPI::TradeProfitExt — calculate profit for specified trading conditions using the extended accuracy volume.
- IMTServerAPI::TradeMarginCheckExt — check the availability of the margin required for the execution of an order with the extended accuracy volume.

New methods of auxiliary classes

- SMTFormat::FormatVolumeExt — format the extended accuracy volume to a string.
- SMTFormat::FormatVolumeExtOrder — format the extended accuracy order volume to a string.

- SMTMath::VolumeExtToInt — convert the extended accuracy volume from double to int.
- SMTMath::VolumeExtToDouble — convert the extended accuracy volume from int to double.
- SMTMath::VolumeExtToSize — convert the extended accuracy volume from lots to amount.
- SMTMath::VolumeExtFromSize — obtain the extended accuracy volume in lots from the volume specified as amount.

Renamed the SMTFormat::FormatVolume method with the double type volume parameter to SMTFormat::FormatVolumeDouble.

New fields in structures

- MTBookItem.volume_ext — extended accuracy request volume.

- MTTick.volume_ext — extended accuracy last deal volume.

- MTTickShort.volume_ext — extended accuracy last deal volume.

- MTTickStat.trade_volume_ext — total extended accuracy deal volume per session.
- MTTickStat.trade_buy_volume_ext — total extended accuracy buy request volume.
- MTTickStat.trade_sell_volume_ext — total extended accuracy sell request volume.
- MTTickStat.vol_high_ext — extended accuracy maximum deal volume per day.
- MTTickStat.vol_low_ext — extended accuracy minimum deal volume per day.

All new fields are similar to the ones with standard accuracy volumes, but their value is passed with the fixed number of decimal places (8).

- An extended accuracy value has a higher priority. The server will use this value, if specified.
- If only an extended accuracy value is specified, a standard accuracy field is filled automatically based on it (up to 4 decimal places).
- If only a standard accuracy value is specified, an extended accuracy field is automatically filled based on it.

2. Added new hooks for dynamic routing of requests.

   HookTradeRequestRuleFilter — hook for checking whether a trade request meets a routing rule. The hook is called before a check of request's correspondence to each of existing rules, and allows the dynamic change of request routes. For example, a plugin may route a request by its own rule during such a check, even if the request does not initially match this rule.

```
virtual MTAPIRES  IMTTradeSink::HookTradeRequestRuleFilter(
   IMTRequest*          request,      // a pointer to the request
object
```

```
5.    IMTConRoute*         rule,         // a pointer to the routing rule
   object
6.    const IMTConGroup*   group         // a pointer to the group
   configuration object
      )
```

HookTradeRequestRuleApply — hook of the application of a routing rule to a trade request. The hook is called after a check of request's correspondence to a rule, and allows the dynamic redefinition of the route. For example, a request that should be processed by a plugin in accordance with the routing rule can instead be sent to a dealer for processing.

```
virtual MTAPIRES  IMTTradeSink::HookTradeRequestRuleApply(
   const IMTRequest*    request,      // a pointer to the request
object
   const IMTConRoute*   rule,         // a pointer to the routing rule
object
   const IMTConGroup*   group         // a pointer to the group
configuration object
   )
```

7. Added the IMTConGroupSymbol::IEFlags property — flags of operation in the instant execution mode. The IMTConSymbol::EnInstantFlags enumeration is used to set the flags.
8. Added the IMTServerAPI::TimeCurrentMsc method for obtaining the current trading time up to milliseconds.
9. Fixed the text of messages passed to the IMTServerSink::OnServerLog event when editing configuration entries.
10. Fixed calling some events and hooks before the plugin returns control from the IMTServerPlugin::Start method.

MetaTrader 5 Manager API build 1930

1. Added new paired methods for working with volumes of extended accuracy to all interfaces featuring the methods for working with volumes. For more details please read the what's new list of Server API.

   IMTSummary::VolumeBuyClientsExt — get the extended accuracy aggregate volume of client's buy positions.
   IMTSummary::VolumeBuyCoverageExt — get the extended accuracy aggregate volume of hedging buy positions.
   IMTSummary::VolumeSellClientsExt — get the extended accuracy aggregate volume of client's sell positions.
   IMTSummary::VolumeSellCoverageExt — get the extended accuracy aggregate volume of hedging sell positions.

   IMTManagerAPI::TradeProfitExt — calculate profit for specified trading conditions using the extended accuracy volume.
   IMTManagerAPI::TradeMarginCheckExt — check the availability of the margin required for the execution of an order with the extended accuracy volume.

2. Fixed calling the IMTOrderSink::OnOrderSync and IMTPositionSink::OnPositionSync events. Previously, they could be called before the corresponding data was fully synchronized.
3. Fixed an error of setting a data folder different from the default one.

MetaTrader 5 Gateway API build 1930

1. Optimized and significantly reduced memory consumption.
2. Fixed an error, which could cause loss of connection between the gateway and the trade server in case of mass update of trading symbol settings.

MetaTrader 5 Report API build 1930

1. Added new interfaces and methods for implementing reports with the Dashboard view type — IMTReportAPI::Dashboard*, IMTReportDashboardData, IMTReportDashboardHtml and IMTReportDashboardWidget.
2. Added working with the Dashboard view to Daily Dealing Report, Daily Expert Report, Daily Server Report, Daily Trade Report, Account Grow and Money Flow sample reports.
3. Added the IMTReportAPI::TradeProfitExt method — calculate profit for specified trading conditions using the extended accuracy volume. For more details regarding extended accuracy volumes, please read the what's new list of Server API.
4. Added the wchar_t category[64] field to the MTReportInfo structure for subsequent grouping of reports in the Manager terminal.

MetaTrader 5 Gateway to MetaTrader 5 build 1930

- Added support for trading in lots with accuracy of 0.00000001. For trading volumes less than 0.0001 lots, the gateway should be connected to an updated MetaTrader 5 server (build 1930 and higher). Otherwise, trading orders with a volume less than 0.0001 lots are rejected.