Chart information

Introduction

The way scripts can obtain information about the chart and symbol they are currently running on is through a subset of Pine Script™'s built-in variables. The ones we cover here allow scripts to access information relating to:

- The chart's prices and volume
- The chart's symbol
- The chart's timeframe
- The session (or time period) the symbol trades on

Prices and volume

The Pine Script™ built-ins for OHLCV values are:

- open: the bar's opening price.
- high: the bar's highest price, or the highest price reached during the realtime bar's elapsed time.
- low: the bar's lowest price, or the lowest price reached during the realtime bar's elapsed time.
- close: the bar's closing price, or the **current price** in the realtime bar.
- volume: the volume traded during the bar, or the volume traded during the realtime bar's elapsed time. The unit of volume information varies with the instrument. It is in shares for stocks, in lots for forex, in contracts for futures, in the base currency for crypto, etc.

Other values are available through:

- hl2: the average of the bar's high and low values.
- hlc3: the average of the bar's high, low and close values.
- ohlc4: the average of the bar's open, high, low and close values.

On historical bars, the values of the above variables do not vary during the bar because only OHLCV information is available on them. When running on historical bars, scripts execute on the bar's close, when all the bar's information is known and cannot change during the script's execution on the bar.

Realtime bars are another story altogether. When indicators (or strategies using `calc_on_every_tick = true`) run in realtime, the values of the above variables (except open) will vary between successive iterations of the script

on the realtime bar, because they represent their **current** value at one point in time during the progress of the realtime bar. This may lead to one form of repainting. See the page on Pine Script™'s execution model for more details.

The [] history-referencing operator can be used to refer to past values of the built-in variables, e.g., `close[1]` refers to the value of close on the previous bar, relative to the particular bar the script is executing on.

Symbol information

Built-in variables in the `syminfo` namespace provide scripts with information on the symbol of the chart the script is running on. This information changes every time a script user changes the chart's symbol. The script then re-executes on all the chart's bars using the new values of the built-in variables:

- syminfo.basecurrency: the base currency, e.g., "BTC" in "BTCUSD", or "EUR" in "EURUSD".
- syminfo.currency: the quote currency, e.g., "USD" in "BTCUSD", or "CAD" in "USDCAD".
- syminfo.description: The long description of the symbol.
- syminfo.mintick: The symbol's tick value, or the minimum increment price can move in. Not to be confused with *pips* or *points*. On "ES1!" ("S&P 500 E-Mini") the tick size is 0.25 because that is the minimal increment the price moves in.
- syminfo.pointvalue: The point value is the multiple of the underlying asset determining a contract's value. On "ES1!" ("S&P 500 E-Mini") the point value is 50, so a contract is worth 50 times the price of the instrument.
- syminfo.prefix: The prefix is the exchange or broker's identifier: "NASDAQ" or "BATS" for "AAPL", "CME_MINI_DL" for "ES1!".
- syminfo.root: It is the ticker's prefix for structured tickers like those of futures. It is "ES" for "ES1!", "ZW" for "ZW1!".
- syminfo.session: It reflects the session setting on the chart for that symbol. If the "Chart settings/Symbol/Session" field is set to "Extended", it will only return "extended" if the symbol and the user's feed allow for extended sessions. It is rarely displayed and used mostly as an argument to the `session` parameter in ticker.new().
- syminfo.ticker: It is the symbol's name, without the exchange part (syminfo.prefix): "BTCUSD", "AAPL", "ES1!", "USDCAD".
- syminfo.tickerid: This string is rarely displayed. It is mostly used as an argument for request.security()'s `symbol` parameter. It includes session, prefix and ticker information.
- syminfo.timezone: The timezone the symbol is traded in. The string is an IANA time zone database name (e.g., "America/New_York").
- syminfo.type: The type of market the symbol belongs to. The values are "stock", "futures", "index", "forex", "crypto", "fund", "dr", "cfd", "bond", "warrant", "structured" and "right".

This script will display the values of those built-in variables on the chart:

```
//@version=5
indicator("`syminfo.*` built-ins", "", true)
printTable(txtLeft, txtRight) =>
    var table t = table.new(position.middle_right, 2, 1)
    table.cell(t, 0, 0, txtLeft, bgcolor = color.yellow, text_halign = text.align_righ
    table.cell(t, 1, 0, txtRight, bgcolor = color.yellow, text_halign = text.align_lef

nl = "\n"
left =
  "syminfo.basecurrency: "  + nl +
  "syminfo.currency: "      + nl +
  "syminfo.description: "   + nl +
  "syminfo.mintick: "       + nl +
  "syminfo.pointvalue: "    + nl +
  "syminfo.prefix: "        + nl +
  "syminfo.root: "          + nl +
  "syminfo.session: "       + nl +
  "syminfo.ticker: "        + nl +
  "syminfo.tickerid: "      + nl +
  "syminfo.timezone: "      + nl +
  "syminfo.type: "

right =
  syminfo.basecurrency              + nl +
  syminfo.currency                  + nl +
  syminfo.description               + nl +
  str.tostring(syminfo.mintick)     + nl +
  str.tostring(syminfo.pointvalue)  + nl +
  syminfo.prefix                    + nl +
  syminfo.root                      + nl +
  syminfo.session                   + nl +
  syminfo.ticker                    + nl +
  syminfo.tickerid                  + nl +
  syminfo.timezone                  + nl +
  syminfo.type

printTable(left, right)
```

Chart timeframe

A script can obtain information on the type of timeframe used on the chart using these built-ins, which all return a “simple bool” result:

- timeframe.isseconds
- timeframe.isminutes
- timeframe.isintraday
- timeframe.isdaily
- timeframe.isweekly
- timeframe.ismonthly
- timeframe.isdwm

Two additional built-ins return more specific timeframe information:

- timeframe.multiplier returns a “simple int” containing the multiplier of the timeframe unit. A chart timeframe of one hour will return `60` because intraday timeframes are expressed in minutes. A 30sec timeframe will return `30` (seconds), a daily chart will return `1` (day), a quarterly chart will return `3` (months), and a yearly chart will return `12` (months). The value of this variable cannot be used as an argument to `timeframe` parameters in built-in functions, as they expect a string in timeframe specifications format.
- timeframe.period returns a string in Pine Script™’s timeframe specification format.

See the page on Timeframes for more information.

Session information

Session information is available in different forms:

- The syminfo.session built-in variable returns a value that is either session.regular or session.extended. It reflects the session setting on the chart for that symbol. If the “Chart settings/Symbol/Session” field is set to “Extended”, it will only return “extended” if the symbol and the user’s feed allow for extended sessions. It is used when a session type is expected, for example as the argument for the `session` parameter in ticker.new().
- Session state built-ins provide information on the trading session a bar belongs to.

Bar states

Colors