

// This Pine Script® code is subject to the terms of the Mozilla Public License 2.0 at
<https://mozilla.org/MPL/2.0/>

//@version=6

indicator('Momentum Flow Build w/ FVG v6 (Clean)', overlay = true, max_boxes_count =
500, max_labels_count = 500)

// =====

// 🚩 Sessions & Levels

// =====

session1_enabled = input.bool(true, title = 'Enable Asia Session', group = '🚩 Asia
Session')

session1_time = input.session('1800-0000', title = 'Asia Session Time', group = '🚩 Asia
Session')

session1_border_color = input.color(color.blue, title = 'Asia Border Color', group = '🚩 Asia
Session')

session1_bg_color = input.color(color.new(color.blue, 90), title = 'Asia Background Color',
group = '🚩 Asia Session')

session2_enabled = input.bool(true, title = 'Enable London Session', group = '🚩 London
Session')

session2_time = input.session('0000-0600', title = 'London Session Time', group = '🚩
London Session')

session2_border_color = input.color(color.new(color.yellow, 0), title = 'London Border
Color', group = '🚩 London Session')

session2_bg_color = input.color(color.new(color.yellow, 90), title = 'London Background
Color', group = '🚩 London Session')

```
session3_enabled = input.bool(true, title = 'Enable 5m Opening Range', group = '🕒 5m OR')
```

```
session3_time = input.session('0930-0935', title = '5m OR Time', group = '🕒 5m OR')
```

```
session3_border_color = input.color(color.new(color.yellow, 0), title = '5m OR Border Color', group = '🕒 5m OR')
```

```
session3_bg_color = input.color(color.new(color.yellow, 90), title = '5m OR Background Color', group = '🕒 5m OR')
```

```
session4_enabled = input.bool(true, title = 'Enable 15m Opening Range', group = '🕒 15m OR')
```

```
session4_time = input.session('0930-0945', title = '15m OR Time', group = '🕒 15m OR')
```

```
session4_border_color = input.color(color.new(color.yellow, 0), title = '15m OR Border Color', group = '🕒 15m OR')
```

```
session4_bg_color = input.color(color.new(color.yellow, 90), title = '15m OR Background Color', group = '🕒 15m OR')
```

```
grpPrev = '🟪 HTF Levels (Daily/Weekly)'
```

```
prev_enabled = input.bool(true, 'Enable HTF Levels', group = grpPrev)
```

```
daily_color = input.color(color.new(color.purple, 0), 'Daily Color', group = grpPrev)
```

```
weekly_color = input.color(color.new(#5b9cf6, 0), 'Weekly Color', group = grpPrev)
```

```
prev_width = input.int(1, 'Line Width', minval = 1, maxval = 4, group = grpPrev)
```

```
// Helper Functions
```

```
is_new_bar(sess) => t = time('D', sess, 'America/New_York'), na(t[1]) and not na(t) or t[1] < t
```

```
is_in_session(sess) => not na(time('D', sess, 'America/New_York'))
```

```
get_session_end(sess, durationHours, durationMinutes) =>
```

```
    t = time('D', sess, 'America/New_York')
```

na(t) ? na : t + durationHours * 3600000 + durationMinutes * 60000

process_session(session_enabled, session_time, border_color, bg_color, durationHours, durationMinutes) =>

var array<box> session_boxes = array.new_box()

var float s_high = na, var float s_low = na

bool in_sess = is_in_session(session_time)

if session_enabled and in_sess

new_bar = is_new_bar(session_time)

s_low := new_bar ? low : na(s_low) ? low : math.min(s_low, low)

s_high := new_bar ? high : na(s_high) ? high : math.max(s_high, high)

var int session_start = na

session_start := new_bar ? time : session_start

session_end = get_session_end(session_time, durationHours, durationMinutes)

if new_bar

array.push(session_boxes, box.new(left = session_start, top = s_high, right = session_end, bottom = s_low, xloc = xloc.bar_time, border_color = border_color, bgcolor = bg_color))

else if array.size(session_boxes) > 0

_bx = array.get(session_boxes, array.size(session_boxes) - 1)

box.set_right(_bx, session_end), box.set_top(_bx, s_high), box.set_bottom(_bx, s_low)

[s_high, s_low, in_sess]

[asiaH, asiaL, asiaAct] = process_session(session1_enabled, session1_time, session1_border_color, session1_bg_color, 22, 0)

[lonH, lonL, lonAct] = process_session(session2_enabled, session2_time, session2_border_color, session2_bg_color, 16, 0)

```
[or5H, or5L, or5Act] = process_session(session3_enabled, session3_time,  
session3_border_color, session3_bg_color, 1, 30)
```

```
[or15H, or15L, or15Act] = process_session(session4_enabled, session4_time,  
session4_border_color, session4_bg_color, 1, 30)
```

```
_raw_pdh = request.security(syminfo.tickerid, 'D', high[1], lookahead =  
barmerge.lookahead_on)
```

```
_raw_pdl = request.security(syminfo.tickerid, 'D', low[1], lookahead =  
barmerge.lookahead_on)
```

```
_raw_pwh = request.security(syminfo.tickerid, 'W', high[1], lookahead =  
barmerge.lookahead_on)
```

```
_raw_pwl = request.security(syminfo.tickerid, 'W', low[1], lookahead =  
barmerge.lookahead_on)
```

```
is_4pm = hour(time, 'America/New_York') == 16 and minute(time, 'America/New_York') == 0
```

```
var float pdh = na, var float pdl = na, var float pwh = na, var float pwl = na
```

```
if is_4pm or na(pdh)
```

```
    pdh := _raw_pdh, pdl := _raw_pdl, pwh := _raw_pwh, pwl := _raw_pwl
```

```
y = year(time, 'America/New_York'), m = month(time, 'America/New_York'), d =  
dayofmonth(time, 'America/New_York')
```

```
today_4pm = timestamp('America/New_York', y, m, d, 16, 0)
```

```
prev_4pm = time < today_4pm ? timestamp('America/New_York', y, m, d - 1, 16, 0) :  
today_4pm
```

```
next_4pm = time < today_4pm ? today_4pm : timestamp('America/New_York', y, m, d + 1,  
16, 0)
```

```
f_draw_hft_line(val, col, txt) =>
```

```
    var line l = na
```

```

var label lbl = na

if prev_enabled and not na(val)

    if na(l)

        l := line.new(prev_4pm, val, next_4pm, val, xloc = xloc.bar_time, color = col, width =
prev_width)

        line.set_x1(l, prev_4pm), line.set_x2(l, next_4pm), line.set_y1(l, val), line.set_y2(l, val)

    if na(lbl)

        lbl := label.new(next_4pm, val, txt, xloc = xloc.bar_time, style = label.style_none,
textcolor = col, textalign = text.align_left)

        label.set_x(lbl, next_4pm), label.set_y(lbl, val)

if prev_enabled

    f_draw_htf_line(pdh, daily_color, "PDH"), f_draw_htf_line(pdl, daily_color, "PDL")

    f_draw_htf_line(pwh, weekly_color, "PWH"), f_draw_htf_line(pwl, weekly_color, "PWL")

// =====

// ✨ Fair Value Gaps (FVG) & Signals

// =====

grpFVG = ' ✨ Fair Value Gaps', fvg_enabled = input.bool(true, 'Enable FVG Detection', group
= grpFVG), fvg_extend_fill = input.bool(true, 'Extend Boxes Until Filled', group = grpFVG)

fvg_min_pct = input.float(0.00, 'Min Gap Size (% of Price)', step = 0.01, group = grpFVG),
fvg_bull_color = input.color(color.new(#089981, 70), 'Bullish FVG Color', group = grpFVG),
fvg_bear_color = input.color(color.new(#f23645, 70), 'Bearish FVG Color', group = grpFVG)

grp123 = ' 🗨 FVG-123 Signals', sig123_enabled = input.bool(true, 'Enable FVG-123 Signals',
group = grp123), sig123_showLong = input.bool(true, 'Show Long Labels', group = grp123),
sig123_showShort = input.bool(true, 'Show Short Labels', group = grp123)

```

```
sig123_onlySessions = input.bool(false, 'Only Show Signals During Sessions', tooltip = 'If unchecked, signals show at any time of day.', group = grp123)
```

```
sig123_longColor = input.color(color.new(#089981, 20), 'Long Color', group = grp123),  
sig123_shortColor = input.color(color.new(#f23645, 20), 'Short Color', group = grp123),  
sig123_labelSize = input.string('Small', 'Size', options = ['Tiny', 'Small', 'Normal', 'Large', 'Huge'], group = grp123)
```

```
sig123_longChar = input.string('L', 'Long Char', group = grp123), sig123_shortChar =  
input.string('S', 'Short Char', group = grp123)
```

```
var array<box> fvg_boxes = array.new_box(), var array<float> fvg_top = array.new_float(), var  
array<float> fvg_bottom = array.new_float(), var array<bool> fvg_isBull = array.new_bool(),  
var array<bool> fvg_active = array.new_bool(), var array<int> fvg_createdBar =  
array.new_int(), var array<bool> fvg_touched = array.new_bool(), var array<bool> fvg_done  
= array.new_bool(), var array<int> fvg_lastTouchBar = array.new_int()
```

```
f_labelSizeFromString(_s) => _s == 'Tiny' ? size.tiny : _s == 'Small' ? size.small : _s ==  
'Normal' ? size.normal : _s == 'Large' ? size.large : size.huge
```

```
f_add_sig_label(_isLong, _y) =>
```

```
inSess = is_in_session(session1_time) or is_in_session(session2_time) or  
is_in_session(session3_time) or is_in_session(session4_time)
```

```
if not sig123_onlySessions or inSess
```

```
label.new(bar_index, _y, _isLong ? sig123_longChar : sig123_shortChar, style = _isLong  
? label.style_label_up : label.style_label_down, color = _isLong ? sig123_longColor :  
sig123_shortColor, textcolor = color.white, size = f_labelSizeFromString(sig123_labelSize))
```

```
f_add_fvg(_isBull, _top, _bottom) =>
```

```
bcolor = _isBull ? fvg_bull_color : fvg_bear_color
```

```
array.push(fvg_boxes, box.new(left = time, top = _top, right = time, bottom = _bottom, xloc  
= xloc.bar_time, bgcolor = bcolor, border_color = bcolor))
```

```
array.push(fvg_top, _top), array.push(fvg_bottom, _bottom), array.push(fvg_isBull,  
_isBull), array.push(fvg_active, true), array.push(fvg_createdBar, bar_index),
```

```
array.push(fvg_touched, false), array.push(fvg_done, false), array.push(fvg_lastTouchBar,
na)
```

```
if fvg_enabled and bar_index >= 2
```

```
    bull_cond = low > high[2], bear_cond = high < low[2]
```

```
    bull_size = bull_cond ? low - high[2] : 0.0, bear_size = bear_cond ? low[2] - high : 0.0
```

```
    if bull_cond and (fvg_min_pct <= 0 or bull_size / close * 100 >= fvg_min_pct)
```

```
        f_add_fvg(true, low, high[2])
```

```
    if bear_cond and (fvg_min_pct <= 0 or bear_size / close * 100 >= fvg_min_pct)
```

```
        f_add_fvg(false, low[2], high)
```

```
var bool sig123_long_alert = false, bool sig123_short_alert = false
```

```
sig123_long_alert := false, sig123_short_alert := false
```

```
if fvg_enabled and array.size(fvg_boxes) > 0
```

```
    for i = 0 to array.size(fvg_boxes) - 1
```

```
        if array.get(fvg_active, i)
```

```
            bx = array.get(fvg_boxes, i), top = array.get(fvg_top, i), bot = array.get(fvg_bottom, i),
            bull = array.get(fvg_isBull, i), createdBar = array.get(fvg_createdBar, i)
```

```
            box.set_right(bx, time)
```

```
            if (bull ? low <= bot : high >= top) and fvg_extend_fill
```

```
                array.set(fvg_active, i, false), array.set(fvg_done, i, true)
```

```
            if sig123_enabled and not array.get(fvg_done, i)
```

```
                afterPrint = bar_index > createdBar, touches_box = (high >= bot) and (low <= top),
                respects_close = bull ? (close >= bot) : (close <= top)
```

```
                if afterPrint and touches_box and respects_close
```

```
                    array.set(fvg_touched, i, true), array.set(fvg_lastTouchBar, i, bar_index)
```

```
lastTouchBar = array.get(fvg_lastTouchBar, i), have_touch = array.get(fvg_touched, i)
and not na(lastTouchBar) and (bar_index > lastTouchBar) and afterPrint
```

```
if bull and have_touch and (close > top)
```

```
  if sig123_showLong
```

```
    f_add_sig_label(true, low)
```

```
    sig123_long_alert := true, array.set(fvg_done, i, true)
```

```
else if (not bull) and have_touch and (close < bot)
```

```
  if sig123_showShort
```

```
    f_add_sig_label(false, high)
```

```
    sig123_short_alert := true, array.set(fvg_done, i, true)
```

```
if afterPrint and ((not bull and close > top) or (bull and close < bot))
```

```
  array.set(fvg_done, i, true)
```

```
alertcondition(sig123_long_alert, title = 'FVG-123 LONG', message = 'FVG-123 LONG on
{{ticker}}')
```

```
alertcondition(sig123_short_alert, title = 'FVG-123 SHORT', message = 'FVG-123 SHORT on
{{ticker}}')
```