

// 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('Universal MFB - Flow/Sweep/FVG-123 Setup', overlay = true, max_labels_count =
500, max_lines_count = 500)

// =====

//  Display Settings

// =====

show_dots_5m = input.bool(true, 'Show Intention Dots on 5m Chart')

show_dots_1m = input.bool(false, 'Show Intention Dots on 1m Chart')

// =====

//  Levels Logic (Mirrored from Clean v6)

// =====

is_in_sess(s) => not na(time('D', s, 'America/New_York'))

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

get_bounds(s) =>

var float h = na, var float l = na

if is_in_sess(s)

new_bar = is_new_b(s)

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

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

[h, l]

```

[ash, asl] = get_bounds('1800-0000')
[loh, lol] = get_bounds('0000-0600')
[o15h, o15l] = get_bounds('0930-0945')

pdh = request.security(syminfo.tickerid, 'D', high[1], lookahead = barmerge.lookahead_on)
pdl = request.security(syminfo.tickerid, 'D', low[1], lookahead = barmerge.lookahead_on)
pwh = request.security(syminfo.tickerid, 'W', high[1], lookahead = barmerge.lookahead_on)
pwl = request.security(syminfo.tickerid, 'W', low[1], lookahead = barmerge.lookahead_on)

lon_sb = '0300-0400', ny_am_sb = '1000-1100', ny_pm_sb = '1400-1500'

is_sb_time = not na(time('D', lon_sb, 'America/New_York')) or not na(time('D', ny_am_sb,
'America/New_York')) or not na(time('D', ny_pm_sb, 'America/New_York'))

// =====

// ⚡ Intention & Level Tracking

// =====

is5mChart = timeframe.isintraday and timeframe.multiplier == 5
is1mChart = timeframe.isintraday and timeframe.multiplier == 1

breachBullLogic = ta.crossover(close, pdh) or ta.crossover(close, ash) or
ta.crossover(close, loh) or ta.crossover(close, o15h) or ta.crossover(close, pwh)

breachBearLogic = ta.crossunder(close, pdl) or ta.crossunder(close, asl) or
ta.crossunder(close, lol) or ta.crossunder(close, o15l) or ta.crossunder(close, pwl)

[breachBull5m, breachBear5m] = request.security(syminfo.tickerid, "5", [breachBullLogic,
breachBearLogic])

is_5m_end = ta.change(time("5")) != 0

```

```
plotBullDot = (is5mChart and show_dots_5m and breachBullLogic) or (is1mChart and  
show_dots_1m and is_5m_end and breachBull5m)
```

```
plotBearDot = (is5mChart and show_dots_5m and breachBearLogic) or (is1mChart and  
show_dots_1m and is_5m_end and breachBear5m)
```

```
plotshape(plotBullDot, title="Bullish Breakout Dot", style=shape.circle,  
location=location.belowbar, color=color.purple, size=size.tiny)
```

```
plotshape(plotBearDot, title="Bearish Breakout Dot", style=shape.circle,  
location=location.abovebar, color=color.purple, size=size.tiny)
```

```
var bool touchedHigh = false, var bool touchedLow = false
```

```
if high >= pdh or high >= ash or high >= loh or high >= o15h or high >= pwh
```

```
    touchedHigh := true
```

```
if low <= pdl or low <= asl or low <= lol or low <= o15l or low <= pwl
```

```
    touchedLow := true
```

```
// =====
```

```
// 💎 CHoCH & CISD (Ch'd) Logic - REFINED ONE-SHOT
```

```
// =====
```

```
var float lastSwingHigh = na
```

```
var float lastSwingLow = na
```

```
ph = ta.pivohigh(high, 3, 3)
```

```
pl = ta.pivotlow(low, 3, 3)
```

```
if not na(ph)
```

```
    lastSwingHigh := ph
```

```
if not na(pl)
```

```
    lastSwingLow := pl
```

```

var float lastBearFVGTop = na
var float lastBullFVGBot = na
if high < low[2]
    lastBearFVGTop := low[2]
if low > high[2]
    lastBullFVGBot := high[2]

// Event detection for one-shot triggering
bool CHoCH_Bull_Event = ta.crossover(close, lastSwingHigh)
bool CHoCH_Bear_Event = ta.crossunder(close, lastSwingLow)
bool CISD_Bull_Event = ta.crossover(close, lastBearFVGTop)
bool CISD_Bear_Event = ta.crossunder(close, lastBullFVGBot)

bool bullChd = is1mChart and touchedLow and (CHoCH_Bull_Event or CISD_Bull_Event)
and close > lastSwingHigh and close > lastBearFVGTop
bool bearChd = is1mChart and touchedHigh and (CHoCH_Bear_Event or
CISD_Bear_Event) and close < lastSwingLow and close < lastBullFVGBot

plotshape(bullChd, title="Bullish Ch'd", text="Ch'd", style=shape.diamond,
location=location.belowbar, color=color.green, textcolor=color.green, size=size.small)

plotshape(bearChd, title="Bearish Ch'd", text="Ch'd", style=shape.diamond,
location=location.abovebar, color=color.red, textcolor=color.red, size=size.small)

// Trigger dynamic alert for Ch'd
if bullChd
    alert("Universal MFB - Bullish Ch'd trade entry confirmation on " + syminfo.ticker,
alert.freq_once_per_bar_close)

```

```
if bearChd
```

```
    alert("Universal MFB - Bearish Ch'd trade entry confirmation on " + syminfo.ticker,  
alert.freq_once_per_bar_close)
```

```
var bool chd_bull_active = false
```

```
var bool chd_bear_active = false
```

```
if bullChd
```

```
    chd_bull_active := true
```

```
    touchedLow    := false
```

```
if bearChd
```

```
    chd_bear_active := true
```

```
    touchedHigh   := false
```

```
// =====
```

```
// 💎 FVG-123 Logic
```

```
// =====
```

```
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()
```

```
if bar_index >= 2
```

```
    if low > high[2]
```

```
        array.push(fvg_top, low), array.push(fvg_bottom, high[2]), array.push(fvg_isBull, true),  
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 high < low[2]
```

```
    array.push(fvg_top, low[2]), array.push(fvg_bottom, high), array.push(fvg_isBull, false),
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)
```

```
var table setupTable = table.new(position.bottom_right, 1, 1, bgcolor =
color.new(color.black, 70), border_width = 1, border_color = color.gray)
```

```
atr = ta.atr(14)
```

```
var string lastIntention = ""
```

```
if breachBullLogic
```

```
    lastIntention := "Bullish"
```

```
else if breachBearLogic
```

```
    lastIntention := "Bearish"
```

```
bool fvg123SetupDetected = false
```

```
if array.size(fvg_active) > 0
```

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

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

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

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

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

```
                afterP = bar_index > createdBar
```

```

if afterP and (high >= bot) and (low <= top) and (bull ? close >= bot : close <= top)
    array.set(fvg_touched, i, true), array.set(fvg_lastTouchBar, i, bar_index)

if afterP and ((not bull and close > top) or (bull and close < bot))
    array.set(fvg_done, i, true)

lastT = array.get(fvg_lastTouchBar, i), haveT = array.get(fvg_touched, i) and not
na(lastT) and bar_index > lastT and afterP

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

    bool levelEngaged = bull ? (touchedLow or chd_bull_active or low <= pdl or low <=
asl or low <= lol) : (touchedHigh or chd_bear_active or high >= pdh or high >= ash or high >=
loh)

    if levelEngaged and barstate.isconfirmed

        fvg123SetupDetected := true

        float ep = close, float sl = bull ? bot - syminfo.mintick : top + syminfo.mintick,
float r = math.abs(ep - sl), float tp = bull ? ep + (r * 2) : ep - (r * 2)

        color yellow = color.rgb(255, 245, 157)

        bool isFlow = bull ? (lastIntention == "Bullish") : (lastIntention == "Bearish")

        string promptBase = isFlow ? "Universal - Flow-FVG-123" : (is_sb_time ?
"Sweep/FVG-123 setup" : "Universal - Sweep/FVG-123 setup")

        float textY = bull ? low - atr * 2.8 : high + atr * 2.8

        label.new(bar_index, textY, promptBase + (bull ? " ▲ " : " ▼ "), color =
#00000000, textcolor = yellow, style = bull ? label.style_label_up : label.style_label_down,
size = size.normal)

        line.new(bar_index, ep, bar_index + 10, ep, color = color.rgb(91, 156, 246), style
= line.style_dotted, width = 2)

        line.new(bar_index, sl, bar_index + 10, sl, color = color.rgb(242, 54, 69), style =
line.style_dotted, width = 2)

```

```
line.new(bar_index, tp, bar_index + 10, tp, color = color.rgb(8, 153, 129), style =
line.style_dotted, width = 2)
```

```
table.cell(setupTable, 0, 0, "Most Recent Setup: " + (isFlow ? "Flow" : "Sweep") +
" FVG-123", text_color = yellow, text_size = size.normal)
```

```
// Trigger dynamic alert for FVG-123
```

```
string alertMsg = promptBase + " trade entry confirmation on " + syminfo.ticker
```

```
alert(alertMsg, alert.freq_once_per_bar_close)
```

```
touchedHigh := false, touchedLow := false, chd_bull_active := false,
chd_bear_active := false
```

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

```
if not (timeframe.isintraday and (timeframe.multiplier == 1 or timeframe.multiplier == 5))
and barstate.islast
```

```
runtime.error("Designed for 1m and 5m only.")
```

```
// =====
```

```
// 🚨 Alerts
```

```
// =====
```

```
alertcondition(fvg123SetupDetected, "Universal MFB entry", "Universal MFB trade entry
confirmation on {{ticker}}")
```

```
alertcondition(bullChd, "Bullish Ch'd", "Bullish Change of Character / CISD detected on
{{ticker}}")
```

```
alertcondition(bearChd, "Bearish Ch'd", "Bearish Change of Character / CISD detected on
{{ticker}}")
```

```
alertcondition(fvg123SetupDetected, "FVG-123 Setup", "New FVG-123 Setup confirmed on
{{ticker}}")
```