```
// 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

// ═══════════════════════════════════════════════════

grpDisp     = '🎨 Display Settings'

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

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


// ═══════════════════════════════════════════════════

// 🕐 Silver Bullet Windows Identification (NY Time)

// ═══════════════════════════════════════════════════

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'))


// ═══════════════════════════════════════════════════

// 🕐 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)
```

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

// 🟣 Intention Logic (Event-Based Breakouts)

// ==================================================

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

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


// Trigger ONLY on the crossover (The moment of breakthrough)

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

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


// HTF validation for 1m chart to ensure dot only appears on confirmed 5m crossovers

[breachBull5m, breachBear5m] = request.security(syminfo.tickerid, "5", [breachBull, breachBear])

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


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

plotBearDot = (is5mChart and show_dots_5m and breachBear) 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)


// ====================================================

// ⚡ Flow & Sweep Tracking

// ====================================================

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


// FVG-123 Arrays

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 label[] labelArray = array.new_label()

var line[] entryLines  = array.new_line(), var line[] slLines = array.new_line(), var
line[] tpLines = array.new_line()

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)

bool buyConfirmed = false, bool sellConfirmed = 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)
                if (bull ? touchedLow or touchedHigh : touchedHigh or touchedLow) and barstate.isconfirmed
                    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 ? touchedHigh : touchedLow
                    string arrowText = bull ? " ▲" : " ▼"
                    string promptBase = isFlow ? "Universal - Flow-FVG-123" : (is_sb_time ? "Sweep/FVG-123 setup" : "Universal - Sweep/FVG-123 setup")
                    string promptText = promptBase + arrowText
                    float textY  = bull ? low - atr * 2.8 : high + atr * 2.8
                    float labelY = bull ? textY - atr * 0.7 : textY + atr * 0.7
```

label nl = label.new(bar_index, labelY, promptText, color = #00000000, textcolor = yellow, style = bull ? label.style_label_up : label.style_label_down, size = size.normal)

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

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

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

string sessName = isFlow ? "Universal Flow" : (not na(time('D', lon_sb, 'America/New_York')) ? "London SB" : not na(time('D', ny_am_sb, 'America/New_York')) ? "NY Morning SB" : not na(time('D', ny_pm_sb, 'America/New_York')) ? "NY Afternoon SB" : "Universal Sweep")

table.cell(setupTable, 0, 0, "Most Recent Setup: " + sessName + " FVG-123", text_color = yellow, text_size = size.normal)

array.push(labelArray, nl), array.push(entryLines, el), array.push(slLines, slL), array.push(tpLines, tpL)

if array.size(labelArray) > 5

  label.delete(array.shift(labelArray)), line.delete(array.shift(entryLines)), line.delete(array.shift(slLines)), line.delete(array.shift(tpLines))

if bull

  buyConfirmed := true

else

  sellConfirmed := true

alert(sessName + " Confirmed: " + (bull ? "Buy" : "Sell"), alert.freq_once_per_bar_close)

```
            touchedHigh := false, touchedLow := false
        array.set(fvg_done, i, true)

alertcondition(buyConfirmed, title = "Universal BUY", message = "BUY
Confirmed")
alertcondition(sellConfirmed, title = "Universal SELL", message = "SELL
Confirmed")

if not (timeframe.isintraday and (timeframe.multiplier == 1 or
timeframe.multiplier == 5)) and barstate.islast
    runtime.error("Designed for 1m and 5m only.")
```