

// 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)

// --- Group Name: FVG Settings ---

fvg_max_age = input.int(100, 'Max FVG Age (Bars)', minval = 1, group = 'FVG Settings', tooltip = 'Only consider FVGs created within this many bars. Helps avoid "historical" FVG signals.')

fvg_min_width = input.float(0, 'Min FVG Width (Ticks)', minval = 0, group = 'FVG Settings', tooltip = 'Minimum size of the FVG in ticks to be considered valid.')

priming_lookback = input.int(30, 'Level Touch Lookback', minval = 1, group = 'Logic Settings', tooltip = 'How many bars the "Level Touch" remains active before resetting. Prevents signals from very old level sweeps.')

strict_key_level = input.bool(false, 'Only FVG-123 after key level touch', group = 'Logic Settings', tooltip = 'If enabled, only "Flow", "Sweep", and "Acceptance" FVG setups will be displayed. Standard "FVG-123" signals that do not interact with key levels will be ignored, reducing chart clutter.')

// --- Group Name: Display Settings ---

show_dots_1h = input.bool(true, 'Show Intention Dots on 1h Chart', group = 'Display Settings', tooltip = 'Displays purple dots on the 1h chart when price breaches key levels.')

show_dots_30m = input.bool(true, 'Show Intention Dots on 30m Chart', group = 'Display Settings', tooltip = 'Displays purple dots on the 30m chart when price breaches key levels.')

```
show_dots_15m = input.bool(true, 'Show Intention Dots on 15m Chart', group = 'Display Settings', tooltip = 'Displays purple dots on the 15m chart when price breaches key levels.')
```

```
show_dots_5m = input.bool(true, 'Show Intention Dots on 5m Chart', group = 'Display Settings', tooltip = 'Displays purple dots on the 5m chart when price breaches key levels.')
```

```
show_dots_1m = input.bool(false, 'Show Intention Dots on 1m Chart', group = 'Display Settings', tooltip = 'Displays purple dots on the 1m chart when the 5m interval ends and a breach occurred.')
```

```
show_levels = input.bool(false, 'Show Key Levels (PDH/PDL/etc.)', group = 'Display Settings', tooltip = 'Toggles the visibility of PDH, PDL, and other session levels.')
```

```
show_labels = input.bool(true, 'Show Entry Labels', group = 'Display Settings', tooltip = 'Toggles the visibility of the "Chd" and setup labels.')
```

```
entry_line_width = input.int(2, 'Trade Entry Line Width', minval = 1, maxval = 10, group = 'Display Settings', tooltip = 'Adjusts the thickness of the trade entry, stop loss, and take profit lines.')
```

```
// --- Group Name: Levels Settings ---
```

```
pdh_color = input.color(color.new(#ab47bc, 30), 'PDH/PDL Color', group = 'Levels Settings', inline = 'd')
```

```
pwh_color = input.color(color.new(#5b9cf6, 30), 'PWH/PWL Color', group = 'Levels Settings', inline = 'w')
```

```
sess_color = input.color(color.new(color.gray, 50), 'Session Levels Color', group = 'Levels Settings', inline = 's')
```

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

```
// 🚫 Levels Logic
```

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

```
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
```

```
    // Reset levels at start of new daily session to clear previous day's data
```

```
    if ta.change(time('D', 'America/New_York')) != 0
```

```
        h := na
```

```
        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')
```

```
[o30h, o30l] = get_bounds('0930-1000')
```

```
[o1hh, o1hl] = get_bounds('0930-1030')
```

```
// Use lookahead_on for HTF levels to get the value available at the start of the  
current bar
```

```
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)
```

```
// Silver Bullet Times
```

```
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'))
```

```
// --- Plot Levels (Matching Image Style) ---
```

```
pdh_plot = ta.change(time('D')) != 0 ? na : pdh
```

```
pdl_plot = ta.change(time('D')) != 0 ? na : pdl
```

```
pwh_plot = ta.change(time('W')) != 0 ? na : pwh
```

```
pwl_plot = ta.change(time('W')) != 0 ? na : pwl
```

```
plot(show_levels ? pdh_plot : na, 'PDH', color = pdh_color, style =  
plot.style_linebr, linewidth = 2)
```

```
plot(show_levels ? pdl_plot : na, 'PDL', color = pdh_color, style =  
plot.style_linebr, linewidth = 2)
```

```
plot(show_levels ? pwh_plot : na, 'PWH', color = pwh_color, style =  
plot.style_linebr, linewidth = 2)
```

```
plot(show_levels ? pwl_plot : na, 'PWL', color = pwh_color, style =  
plot.style_linebr, linewidth = 2)
```

```
plot(show_levels ? ash : na, 'Asian High', color = sess_color, style =  
plot.style_linebr)
```

```
plot(show_levels ? asl : na, 'Asian Low', color = sess_color, style =  
plot.style_linebr)
```

```
plot(show_levels ? loh : na, 'London High', color = sess_color, style =  
plot.style_linebr)
```

```
plot(show_levels ? lol : na, 'London Low', color = sess_color, style =  
plot.style_linebr)
```

```
plot(show_levels ? o15h : na, '9:30 Open High', color = sess_color, style =  
plot.style_linebr)
```

```
plot(show_levels ? o15l : na, '9:30 Open Low', color = sess_color, style =  
plot.style_linebr)
```

```
plot(show_levels ? o30h : na, '10:00 Open High', color = sess_color, style =  
plot.style_linebr)
```

```
plot(show_levels ? o30l : na, '10:00 Open Low', color = sess_color, style =  
plot.style_linebr)
```

```
plot(show_levels ? o1hh : na, '10:30 Open High', color = sess_color, style =  
plot.style_linebr)
```

```
plot(show_levels ? o1hl : na, '10:30 Open Low', color = sess_color, style =  
plot.style_linebr)
```

```
// Labeling levels on the right
```

```
var label lbl_pdh = na, var label lbl_pdl = na, var label lbl_pwh = na, var label  
lbl_pwl = na
```

```
if barstate.islast and show_levels
```

```
    label.delete(lbl_pdh), label.delete(lbl_pdl), label.delete(lbl_pwh),  
    label.delete(lbl_pwl)
```

```
    lbl_pdh := label.new(bar_index + 2, pdh, "PDH", color = #00000000,  
textcolor = pdh_color, style = label.style_label_left, size = size.small)
```

```
    lbl_pdl := label.new(bar_index + 2, pdl, "PDL", color = #00000000, textcolor =  
pdh_color, style = label.style_label_left, size = size.small)
```

```
    lbl_pwh := label.new(bar_index + 2, pwh, "PWH", color = #00000000,  
textcolor = pwh_color, style = label.style_label_left, size = size.small)
```

```
    lbl_pwl := label.new(bar_index + 2, pwl, "PWL", color = #00000000, textcolor  
= pwh_color, style = label.style_label_left, size = size.small)
```

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

```
// ⚡ Intention & Level Tracking
```

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

is1hChart = timeframe.isintraday and timeframe.multiplier == 60 and
timeframe.isminutes

is30mChart = timeframe.isintraday and timeframe.multiplier == 30 and
timeframe.isminutes

is15mChart = timeframe.isintraday and timeframe.multiplier == 15 and
timeframe.isminutes

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

is3mChart = timeframe.isintraday and timeframe.multiplier == 3 and
timeframe.isminutes

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

cross1 = ta.crossover(close, pdh)

cross2 = ta.crossover(close, ash)

cross3 = ta.crossover(close, loh)

cross4 = ta.crossover(close, o15h)

cross5 = ta.crossover(close, pwh)

cross6 = ta.crossover(close, o30h)

cross7 = ta.crossover(close, o1hh)

breachBullLogic = cross1 or cross2 or cross3 or cross4 or cross5 or cross6 or
cross7

under1 = ta.crossunder(close, pdl)

under2 = ta.crossunder(close, asl)

```
under3 = ta.crossunder(close, lol)
under4 = ta.crossunder(close, o15l)
under5 = ta.crossunder(close, pwl)
under6 = ta.crossunder(close, o30l)
under7 = ta.crossunder(close, o1hl)

breachBearLogic = under1 or under2 or under3 or under4 or under5 or under6
or under7
```

```
var int lastBullBreachBar = na
var int lastBearBreachBar = na

if breachBullLogic
    lastBullBreachBar := bar_index

if breachBearLogic
    lastBearBreachBar := bar_index
```

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

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

```
plotBullDot = (is1hChart and show_dots_1h and breachBullLogic) or
(is30mChart and show_dots_30m and breachBullLogic) or
(is15mChart and show_dots_15m and breachBullLogic) or
(is5mChart and show_dots_5m and breachBullLogic) or
```

((is1mChart or is3mChart) and show_dots_1m and is_5m_end and
breachBull5m)

plotBearDot = (is1hChart and show_dots_1h and breachBearLogic) or

(is30mChart and show_dots_30m and breachBearLogic) or

(is15mChart and show_dots_15m and breachBearLogic) or

(is5mChart and show_dots_5m and breachBearLogic) or

((is1mChart or is3mChart) 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 >= o30h or
high >= o1hh or high >= pwh

touchedHigh := true

if low <= pdl or low <= asl or low <= lol or low <= o15l or low <= o30l or low <= o1hl or low <= pwl

touchedLow := true

// --- Priming Reset Logic ---

```
last_high_touch = ta.barssince(high >= pdh or high >= ash or high >= loh or  
high >= o15h or high >= o30h or high >= o1hh or high >= pwh)
```

```
last_low_touch = ta.barssince(low <= pdl or low <= asl or low <= lol or low <=  
o15l or low <= o30l or low <= o1hl or low <= pwl)
```

```
if not na(last_high_touch) and last_high_touch > priming_lookback
```

```
    touchedHigh := false
```

```
if not na(last_low_touch) and last_low_touch > priming_lookback
```

```
    touchedLow := false
```

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

```
// 💎 CHoCH & CISD (Ch'd) Logic
```

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

```
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
```

```
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 or is3mChart) and touchedLow and  
(CHoCH_Bull_Event or CISD_Bull_Event) and close > lastSwingHigh and close  
> lastBearFVGTop
```

```
bool bearChd = (is1mChart or is3mChart) and touchedHigh and  
(CHoCH_Bear_Event or CISD_Bear_Event) and close < lastSwingLow and  
close < lastBullFVGBot
```

```
plotshape(show_labels and bullChd ? true : false, title="Bullish Ch'd",  
text="Ch'd", style=shape.diamond, location=location.belowbar,  
color=#089981, textcolor=#089981, size=size.small)
```

```
plotshape(show_labels and bearChd ? true : false, title="Bearish Ch'd",  
text="Ch'd", style=shape.diamond, location=location.abovebar,  
color=#f23645, textcolor=#f23645, size=size.small)
```

```
// Trigger dynamic alerts
```

```
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
```

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

```
// 🌟 5m CISD (Sweep) Logic for Gold
```

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

```
var float displaced_high_level = na
```

```
var float displaced_high_low  = na
```

```
var bool is_high_displaced   = false
```

```
var float displaced_low_level = na
var float displaced_low_high = na
var bool is_low_displaced = false
```

```
var int displaced_high_bar = na
var int displaced_low_bar = na
```

```
if is5mChart
```

```
  if breachBullLogic
```

```
    float breached_lvl = na
```

```
    if ta.crossover(close, pdh)
```

```
      breached_lvl := math.max(nz(breached_lvl), pdh)
```

```
    if ta.crossover(close, pwh)
```

```
      breached_lvl := math.max(nz(breached_lvl), pwh)
```

```
    if ta.crossover(close, ash)
```

```
      breached_lvl := math.max(nz(breached_lvl), ash)
```

```
    if ta.crossover(close, loh)
```

```
      breached_lvl := math.max(nz(breached_lvl), loh)
```

```
    if ta.crossover(close, o15h)
```

```
      breached_lvl := math.max(nz(breached_lvl), o15h)
```

```
    if ta.crossover(close, o30h)
```

```
      breached_lvl := math.max(nz(breached_lvl), o30h)
```

```
    if ta.crossover(close, o1hh)
```

```
breached_lvl := math.max(nz(breached_lvl), o1hh)
```

```
is_high_displaced := true
```

```
displaced_high_level := breached_lvl
```

```
displaced_high_low := low
```

```
displaced_high_bar := bar_index
```

```
if breachBearLogic
```

```
float breached_lvl = na
```

```
if ta.crossunder(close, pdl)
```

```
breached_lvl := na(breached_lvl) ? pdl : math.min(breached_lvl, pdl)
```

```
if ta.crossunder(close, pwl)
```

```
breached_lvl := na(breached_lvl) ? pwl : math.min(breached_lvl, pwl)
```

```
if ta.crossunder(close, asl)
```

```
breached_lvl := na(breached_lvl) ? asl : math.min(breached_lvl, asl)
```

```
if ta.crossunder(close, lol)
```

```
breached_lvl := na(breached_lvl) ? lol : math.min(breached_lvl, lol)
```

```
if ta.crossunder(close, o15l)
```

```
breached_lvl := na(breached_lvl) ? o15l : math.min(breached_lvl, o15l)
```

```
if ta.crossunder(close, o30l)
```

```
breached_lvl := na(breached_lvl) ? o30l : math.min(breached_lvl, o30l)
```

```
if ta.crossunder(close, o1hl)
```

```
breached_lvl := na(breached_lvl) ? o1hl : math.min(breached_lvl, o1hl)
```

```
is_low_displaced := true  
displaced_low_level := breached_lvl  
displaced_low_high := high  
displaced_low_bar := bar_index
```

```
if is5mChart
```

```
    // Reset if we go too far without a CISD
```

```
    if is_high_displaced and close > displaced_high_level and low >  
displaced_high_low and bar_index - displaced_high_bar > 10
```

```
        is_high_displaced := false
```

```
    if is_low_displaced and close < displaced_low_level and high <  
displaced_low_high and bar_index - displaced_low_bar > 10
```

```
        is_low_displaced := false
```

```
bool bear_5m_cisd = is5mChart and is_high_displaced and close <  
displaced_high_level and close < displaced_high_low
```

```
bool bull_5m_cisd = is5mChart and is_low_displaced and close >  
displaced_low_level and close > displaced_low_high
```

```
if bear_5m_cisd
```

```
    is_high_displaced := false
```

```
    alert("Universal MFB - 5m Bearish CISD detected on " + syminfo.ticker,  
alert.freq_once_per_bar_close)
```

```
    if show_labels
```

```

float ep = close

float sl = high

float r = math.abs(sl - ep)

float tp1 = ep - r

float tp2 = ep - (r * 2)

line.new(bar_index, ep, bar_index + 10, ep, color = #5b9cf6, style =
line.style_dotted, width = entry_line_width)

line.new(bar_index, sl, bar_index + 10, sl, color = #f23645, style =
line.style_dotted, width = entry_line_width)

line.new(bar_index, tp1, bar_index + 10, tp1, color = #089981, style =
line.style_dotted, width = entry_line_width)

line.new(bar_index, tp2, bar_index + 10, tp2, color = #089981, style =
line.style_dotted, width = entry_line_width)

if bull_5m_cisd

    is_low_displaced := false

    alert("Universal MFB - 5m Bullish CISD detected on " + syminfo.ticker,
alert.freq_once_per_bar_close)

    if show_labels

        float ep = close

        float sl = low

        float r = math.abs(ep - sl)

        float tp1 = ep + r

        float tp2 = ep + (r * 2)

```

```
line.new(bar_index, ep, bar_index + 10, ep, color = #5b9cf6, style =  
line.style_dotted, width = entry_line_width)
```

```
line.new(bar_index, sl, bar_index + 10, sl, color = #f23645, style =  
line.style_dotted, width = entry_line_width)
```

```
line.new(bar_index, tp1, bar_index + 10, tp1, color = #089981, style =  
line.style_dotted, width = entry_line_width)
```

```
line.new(bar_index, tp2, bar_index + 10, tp2, color = #089981, style =  
line.style_dotted, width = entry_line_width)
```

```
plotshape(show_labels and bear_5m_cisd ? true : false, title="5m Bearish  
CISD", text="CISD", style=shape.labeldown, location=location.abovebar,  
color=#f23645, textcolor=color.white, size=size.small)
```

```
plotshape(show_labels and bull_5m_cisd ? true : false, title="5m Bullish  
CISD", text="CISD", style=shape.labelup, location=location.belowbar,  
color=#089981, textcolor=color.white, size=size.small)
```

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

```
// ✨ FVG-123 Logic (Optimized with UDT)
```

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

```
type FVG
```

```
float top
```

```
float bottom
```

```
bool isBull
```

```
bool active
```

```
int createdBar
```

bool touched

bool done

int lastTouchBar

var fvgArray = array.new<FVG>()

if bar_index >= 2

if low > high[2]

array.push(fvgArray, FVG.new(low, high[2], true, true, bar_index, false, false, na))

if high < low[2]

array.push(fvgArray, FVG.new(low[2], high, false, true, bar_index, false, false, na))

// Memory Management: Keep array size reasonable

if array.size(fvgArray) > 100

array.shift(fvgArray)

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

bool keyLevelResponseDetected = false

if array.size(fvgArray) > 0

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

fvgObj = array.get(fvgArray, i)

if fvgObj.active

if (fvgObj.isBull ? low <= fvgObj.bottom : high >= fvgObj.top)

fvgObj.active := false

fvgObj.done := true

if not fvgObj.done

afterP = bar_index > fvgObj.createdBar

if afterP and (high >= fvgObj.bottom) and (low <= fvgObj.top) and
(fvgObj.isBull ? close >= fvgObj.bottom : close <= fvgObj.top)

fvgObj.touched := true

fvgObj.lastTouchBar := bar_index

if afterP and ((not fvgObj.isBull and close > fvgObj.top) or
(fvgObj.isBull and close < fvgObj.bottom))

fvgObj.done := true

haveT = fvgObj.touched and not na(fvgObj.lastTouchBar) and
bar_index > fvgObj.lastTouchBar and afterP

if (fvgObj.isBull and haveT and close > fvgObj.top) or (not fvgObj.isBull
and haveT and close < fvgObj.bottom)

// --- Logic Refinement ---

bool is_flow_setup = fvgObj.isBull ? (lastIntention == "Bullish") :
(lastIntention == "Bearish")

bool is_sweep_setup = fvgObj.isBull ? (lastIntention == "Bearish") :
(lastIntention == "Bullish")

bool is_acceptance_setup = fvgObj.isBull ? (not
na(lastBullBreachBar) and bar_index - lastBullBreachBar <= 10) : (not
na(lastBearBreachBar) and bar_index - lastBearBreachBar <= 10)

// A bull setup is engaged if:

// 1. We are in "Flow" and recently touched a High level
(continuation)

// 2. We are in "Sweep" and recently touched a Low level (reversal)

// 3. A Ch'd event occurred

// 4. We pushed beyond a key level (Acceptance)

bool flowEngaged = fvgObj.isBull ? ((is_flow_setup and
touchedHigh)) : ((is_flow_setup and touchedLow))

```
bool sweepEngaged = fvgObj.isBull ? ((is_sweep_setup and
touchedLow) or low <= pdl or low <= asl or low <= lol) : ((is_sweep_setup and
touchedHigh) or high >= pdh or high >= ash or high >= loh)
```

```
// Filter: Age and Width
```

```
bool fvg_valid = (bar_index - fvgObj.createdBar <= fvg_max_age) and
(math.abs(fvgObj.top - fvgObj.bottom) / syminfo.mintick >=
fvg_min_width)
```

```
if fvg_valid and barstate.isconfirmed
```

```
bool is_key_level_response = is_acceptance_setup or
sweepEngaged or flowEngaged
```

```
if is_key_level_response or not strict_key_level
```

```
fvg123SetupDetected := true
```

```
if is_key_level_response
```

```
keyLevelResponseDetected := true
```

```
float ep = close
```

```
float sl = fvgObj.isBull ? fvgObj.bottom - syminfo.mintick :
fvgObj.top + syminfo.mintick
```

```
float r = math.abs(ep - sl)
```

```
float tp = fvgObj.isBull ? ep + (r * 2) : ep - (r * 2)
```

```
color text_col = #5b9cf6
```

```

string promptBase = "FVG-123"

if is_acceptance_setup
    promptBase := "Acceptance"

else if sweepEngaged
    promptBase := "Sweep"

else if flowEngaged
    promptBase := "Flow"

float textY = fvgObj.isBull ? low - atr * 2.8 : high + atr * 2.8

if show_labels
    label.new(bar_index, textY, promptBase + (fvgObj.isBull ? " ▲ "
: " ▼"), color = #00000000, textcolor = text_col, style = fvgObj.isBull ?
label.style_label_up : label.style_label_down, size = size.normal)

    line.new(bar_index, ep, bar_index + 10, ep, color = #5b9cf6,
style = line.style_dotted, width = entry_line_width)

    line.new(bar_index, sl, bar_index + 10, sl, color = #f23645,
style = line.style_dotted, width = entry_line_width)

    line.new(bar_index, tp, bar_index + 10, tp, color = #089981,
style = line.style_dotted, width = entry_line_width)

table.cell(setupTable, 0, 0, "Most Recent: " + promptBase,
text_color = text_col, text_size = size.normal)

```

```
        alert(promptBase + " trade entry confirmation on " +  
syminfo.ticker, alert.freq_once_per_bar_close)
```

```
touchedHigh := false
```

```
touchedLow := false
```

```
chd_bull_active := false
```

```
chd_bear_active := false
```

```
fvgObj.done := true
```

```
if not (timeframe.isintraday and (timeframe.multiplier == 1 or  
timeframe.multiplier == 3 or timeframe.multiplier == 5 or timeframe.multiplier  
== 15 or timeframe.multiplier == 30 or timeframe.multiplier == 60)) and  
barstate.islast
```

```
    runtime.error("Designed for 1m, 3m, 5m, 15m, 30m, and 1h 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}}")

alertcondition(bull_5m_cisd, "5m Bullish Cisd", "5m Bullish Cisd detected on {{ticker}}")

alertcondition(bear_5m_cisd, "5m Bearish Cisd", "5m Bearish Cisd detected on {{ticker}}")

alertcondition(keyLevelResponseDetected, "Any Key Level Response", "Key Level Response (Acceptance/Sweep/Flow) confirmed on {{ticker}}")

alertcondition((bull_5m_cisd or bear_5m_cisd) or
keyLevelResponseDetected, "Cisd / Key Level FVG", "Cisd or FVG-123 only after key level touch on {{ticker}}")