

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

```
//@version=6
```

```
indicator('Universal MFB - Gold session', shorttitle = 'MFB Gold', overlay = true,  
max_labels_count = 500, max_lines_count = 500)
```

```
// --- Constants ---
```

```
string TZ = 'America/New_York'
```

```
color BULL_COLOR = #089981
```

```
color BEAR_COLOR = #f23645
```

```
color INFO_COLOR = #5b9cf6
```

```
color OWL_COLOR = color.yellow
```

```
// --- FVG Settings ---
```

```
fvgMaxAgeInput = input.int(100, 'Max FVG Age (Bars)', minval = 1, group = 'FVG  
Settings', tooltip = 'Only consider FVGs created within this many bars.')
```

```
fvgMinWidthInput = input.float(0, 'Min FVG Width (Ticks)', minval = 0, group =  
'FVG Settings', tooltip = 'Minimum FVG width in ticks required for a valid  
setup.')
```

```
// --- Logic Settings ---
```

```
primingLookbackInput = input.int(30, 'Level Touch Lookback', minval = 1,  
group = 'Logic Settings', tooltip = 'How many bars a general key-level touch  
remains active.')
```

```
strictKeyLevelInput = input.bool(false, 'Only FVG-123 after key level touch',  
group = 'Logic Settings', tooltip = 'If enabled, display only Flow, Sweep, and  
Acceptance FVG-123 setups.')
```

```
comexSessionInput = input.session('0820-0930', 'COMEX Awareness  
Window', group = 'Logic Settings', tooltip = 'New Gold-session awareness  
signals are evaluated during this New York time window.')
```

```
confluenceDistanceInput = input.float(100, 'Asia/PDH Confluence Distance  
(Ticks)', minval = 0, group = 'Logic Settings', tooltip = 'Maximum distance for an  
Asia high/low and PDH/PDL to be treated as a confluence zone.')
```

```
// --- Display Settings ---
```

```
showDots1hInput = input.bool(false, 'Show Intention Dots on 1h Chart', group  
= 'Display Settings', tooltip = 'Displays purple dots on the 1h chart when price  
breaches key levels.')
```

```
showDots30mInput = input.bool(false, 'Show Intention Dots on 30m Chart',  
group = 'Display Settings', tooltip = 'Displays purple dots on the 30m chart  
when price breaches key levels.')
```

```
showDots15mInput = input.bool(false, 'Show Intention Dots on 15m Chart',  
group = 'Display Settings', tooltip = 'Displays purple dots on the 15m chart  
when price breaches key levels.')
```

```
showDots5mInput = input.bool(false, 'Show Intention Dots on 5m Chart',  
group = 'Display Settings', tooltip = 'Displays purple dots on the 5m chart when  
price breaches key levels.')
```

```
showDots1mInput = input.bool(false, 'Show Intention Dots on 1m Chart',  
group = 'Display Settings', tooltip = 'Displays purple dots on the 1m chart when  
a 5m interval ends after a breach.')
```

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

```
showLabelsInput = input.bool(true, 'Show Entry Labels', group = 'Display  
Settings', tooltip = 'Toggles Chd, CISD, FVG-123, and setup labels.')
```

```
showOwlInput = input.bool(true, 'Show O.W.L. Labels (Opportunity With  
Liquidity)', group = 'Display Settings', tooltip = 'O.W.L. means Opportunity With  
Liquidity. Shows O.W.L. above each Asia level that has not yet received a  
confirmed Sweep FVG-123.')
```

```
owlSizeInput = input.string('Large', 'O.W.L. Text Size', options = ['Tiny', 'Small',  
'Normal', 'Large', 'Huge'], group = 'Display Settings', tooltip = 'O.W.L. means  
Opportunity With Liquidity. Adjusts the label text size.')
```

```
showAwarenessSignalsInput = input.bool(false, 'Show Awareness Signals',  
group = 'Display Settings', tooltip = 'Shows COMEX touch signals and  
unmitigated Sweep FVG-123 signals.')
```

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

```
owlLabelSize = switch owlSizeInput
```

```
  'Tiny' => size.tiny
```

```
  'Small' => size.small
```

```
  'Large' => size.large
```

```
  'Huge' => size.huge
```

```
  => size.normal
```

```
// --- Levels Settings ---
```

```
pdhColorInput = input.color(color.new(#ab47bc, 30), 'PDH/PDL Color', group
= 'Levels Settings', inline = 'd', tooltip = 'Color for previous-day high and low.')

pwhColorInput = input.color(color.new(#5b9cf6, 30), 'PWH/PWL Color', group
= 'Levels Settings', inline = 'w', tooltip = 'Color for previous-week high and low.')

sessionColorInput = input.color(color.new(color.gray, 50), 'Session Levels
Color', group = 'Levels Settings', inline = 's', tooltip = 'Color for intraday session
levels.')
```

```
// --- Session Level Functions ---
```

```
isInSession(sessionInput) =>
```

```
  not na(time(timeframe.period, sessionInput, TZ))
```

```
getBounds(sessionInput) =>
```

```
  var float sessionHigh = na
```

```
  var float sessionLow = na
```

```
  bool inSession = isInSession(sessionInput)
```

```
  bool newSession = inSession and not inSession[1]
```

```
  if newSession
```

```
    sessionHigh := high
```

```
    sessionLow := low
```

```
  else if inSession
```

```
    sessionHigh := math.max(nz(sessionHigh, high), high)
```

```
    sessionLow := math.min(nz(sessionLow, low), low)
```

```
  [sessionHigh, sessionLow]
```

```
[asiaHigh, asiaLow] = getBounds('1800-0000')  
[londonHigh, londonLow] = getBounds('0000-0600')  
[open15High, open15Low] = getBounds('0930-0945')  
[open30High, open30Low] = getBounds('0930-1000')  
[open1hHigh, open1hLow] = getBounds('0930-1030')
```

```
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)
```

```
isSilverBulletTime = not na(time(timeframe.period, '0300-0400', TZ)) or not  
na(time(timeframe.period, '1000-1100', TZ)) or not na(time(timeframe.period,  
'1400-1500', TZ))
```

```
// --- Plot Levels ---
```

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

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

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

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

```
plot(showLevelsInput ? pdhPlot : na, 'PDH', color = pdhColorInput, style =  
plot.style_linebr, linewidth = 2)
```

```
plot(showLevelsInput ? pdlPlot : na, 'PDL', color = pdhColorInput, style =  
plot.style_linebr, linewidth = 2)
```

```
plot(showLevelsInput ? pwhPlot : na, 'PWH', color = pwhColorInput, style =  
plot.style_linebr, linewidth = 2)
```

```
plot(showLevelsInput ? pwlPlot : na, 'PWL', color = pwhColorInput, style =  
plot.style_linebr, linewidth = 2)
```

```
plot(showLevelsInput ? asiaHigh : na, 'Asian High', color = sessionColorInput,  
style = plot.style_linebr)
```

```
plot(showLevelsInput ? asiaLow : na, 'Asian Low', color = sessionColorInput,  
style = plot.style_linebr)
```

```
plot(showLevelsInput ? londonHigh : na, 'London High', color =  
sessionColorInput, style = plot.style_linebr)
```

```
plot(showLevelsInput ? londonLow : na, 'London Low', color =  
sessionColorInput, style = plot.style_linebr)
```

```
plot(showLevelsInput ? open15High : na, '9:30 Open High', color =  
sessionColorInput, style = plot.style_linebr)
```

```
plot(showLevelsInput ? open15Low : na, '9:30 Open Low', color =  
sessionColorInput, style = plot.style_linebr)
```

```
plot(showLevelsInput ? open30High : na, '10:00 Open High', color =  
sessionColorInput, style = plot.style_linebr)
```

```
plot(showLevelsInput ? open30Low : na, '10:00 Open Low', color =  
sessionColorInput, style = plot.style_linebr)
```

```
plot(showLevelsInput ? open1hHigh : na, '10:30 Open High', color =  
sessionColorInput, style = plot.style_linebr)
```

```
plot(showLevelsInput ? open1hLow : na, '10:30 Open Low', color =  
sessionColorInput, style = plot.style_linebr)
```

```
var label pdhLabel = na
```

```
var label pdlLabel = na
```

```
var label pwhLabel = na
```

```
var label pwlLabel = na
```

```
if barstate.islast and showLevelsInput
```

```
    label.delete(pdhLabel)
```

```
    label.delete(pdlLabel)
```

```
    label.delete(pwhLabel)
```

```
    label.delete(pwlLabel)
```

```
    if not na(pdh)
```

```
        pdhLabel := label.new(bar_index + 2, pdh, 'PDH', color = #00000000,  
textcolor = pdhColorInput, style = label.style_label_left, size = size.small)
```

```
    if not na(pdl)
```

```
        pdlLabel := label.new(bar_index + 2, pdl, 'PDL', color = #00000000,  
textcolor = pdhColorInput, style = label.style_label_left, size = size.small)
```

```
    if not na(pwh)
```

```
        pwhLabel := label.new(bar_index + 2, pwh, 'PWH', color = #00000000,  
textcolor = pwhColorInput, style = label.style_label_left, size = size.small)
```

```
    if not na(pwl)
```

```
        pwlLabel := label.new(bar_index + 2, pwl, 'PWL', color = #00000000,  
textcolor = pwhColorInput, style = label.style_label_left, size = size.small)
```

// --- Timeframe and Intention Logic ---

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

crossPdh = ta.crossover(close, pdh)

crossAsiaHigh = ta.crossover(close, asiaHigh)

crossLondonHigh = ta.crossover(close, londonHigh)

crossOpen15High = ta.crossover(close, open15High)

crossPwh = ta.crossover(close, pwh)

crossOpen30High = ta.crossover(close, open30High)

crossOpen1hHigh = ta.crossover(close, open1hHigh)

breachBullLogic = crossPdh or crossAsiaHigh or crossLondonHigh or
crossOpen15High or crossPwh or crossOpen30High or crossOpen1hHigh

underPdl = ta.crossunder(close, pdl)

```
underAsiaLow = ta.crossunder(close, asiaLow)
underLondonLow = ta.crossunder(close, londonLow)
underOpen15Low = ta.crossunder(close, open15Low)
underPwl = ta.crossunder(close, pwl)
underOpen30Low = ta.crossunder(close, open30Low)
underOpen1hLow = ta.crossunder(close, open1hLow)

breachBearLogic = underPdl or underAsiaLow or underLondonLow or
underOpen15Low or underPwl or underOpen30Low or underOpen1hLow
```

```
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])

is5mEnd = ta.change(time('5')) != 0
```

```
plotBullDot = (is1hChart and showDots1hInput and breachBullLogic) or
(is30mChart and showDots30mInput and breachBullLogic) or (is15mChart
and showDots15mInput and breachBullLogic) or (is5mChart and
showDots5mInput and breachBullLogic) or ((is1mChart or is3mChart) and
showDots1mInput and is5mEnd and breachBull5m)
```

```
plotBearDot = (is1hChart and showDots1hInput and breachBearLogic) or  
(is30mChart and showDots30mInput and breachBearLogic) or (is15mChart  
and showDots15mInput and breachBearLogic) or (is5mChart and  
showDots5mInput and breachBearLogic) or ((is1mChart or is3mChart) and  
showDots1mInput and is5mEnd 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)
```

```
// --- General Key-Level Touch State ---
```

```
var bool touchedHigh = false
```

```
var bool touchedLow = false
```

```
highLevelTouched = (not na(pdh) and high >= pdh) or (not na(asiaHigh) and  
high >= asiaHigh) or (not na(londonHigh) and high >= londonHigh) or (not  
na(open15High) and high >= open15High) or (not na(open30High) and high >=  
open30High) or (not na(open1hHigh) and high >= open1hHigh) or (not na(pwh)  
and high >= pwh)
```

```
lowLevelTouched = (not na(pdl) and low <= pdl) or (not na(asiaLow) and low <=  
asiaLow) or (not na(londonLow) and low <= londonLow) or (not  
na(open15Low) and low <= open15Low) or (not na(open30Low) and low <=  
open30Low) or (not na(open1hLow) and low <= open1hLow) or (not na(pwl)  
and low <= pwl)
```

```
if highLevelTouched
```

```
    touchedHigh := true
```

```
if lowLevelTouched
```

```
    touchedLow := true
```

```
lastHighTouch = ta.barssince(highLevelTouched)
```

```
lastLowTouch = ta.barssince(lowLevelTouched)
```

```
if not na(lastHighTouch) and lastHighTouch > primingLookbackInput
```

```
    touchedHigh := false
```

```
if not na(lastLowTouch) and lastLowTouch > primingLookbackInput
```

```
    touchedLow := false
```

```
// --- CHoCH and CISD Logic ---
```

```
var float lastSwingHigh = na
```

```
var float lastSwingLow = na
```

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

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

```
if not na(pivotHigh)
```

```
    lastSwingHigh := pivotHigh
```

```
if not na(pivotLow)
```

```
    lastSwingLow := pivotLow
```

```
var float lastBearFvgTop = na
```

```
var float lastBullFvgBottom = na
```

```
if high < low[2]
```

```
lastBearFvgTop := low[2]
```

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

```
lastBullFvgBottom := high[2]
```

```
chochBullEvent = ta.crossover(close, lastSwingHigh)
```

```
chochBearEvent = ta.crossunder(close, lastSwingLow)
```

```
cisdBullEvent = ta.crossover(close, lastBearFvgTop)
```

```
cisdBearEvent = ta.crossunder(close, lastBullFvgBottom)
```

```
bullChd = (is1mChart or is3mChart) and touchedLow and (chochBullEvent or  
cisdBullEvent) and close > lastSwingHigh and close > lastBearFvgTop
```

```
bearChd = (is1mChart or is3mChart) and touchedHigh and (chochBearEvent  
or cisdBearEvent) and close < lastSwingLow and close < lastBullFvgBottom
```

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

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

```
if bullChd
```

```
    alert('Universal MFB - Bullish Chd trade entry confirmation on ' +  
syminfo.ticker, alert.freq_once_per_bar_close)
```

```
if bearChd
```

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

```
var bool chdBullActive = false
```

```
var bool chdBearActive = false
```

```
if bullChd
```

```
    chdBullActive := true
```

```
    touchedLow := false
```

```
if bearChd
```

```
    chdBearActive := true
```

```
    touchedHigh := false
```

```
// --- 5m CISD / Sweep Logic ---
```

```
var float displacedHighLevel = na
```

```
var float displacedHighLow = na
```

```
var bool isHighDisplaced = false
```

```
var float displacedLowLevel = na
```

```
var float displacedLowHigh = na
```

```
var bool isLowDisplaced = false
```

```
var int displacedHighBar = na
```

```
var int displacedLowBar = na
```

```
if is5mChart
```

```

if breachBullLogic

    float breachedHighLevel = na

    if crossPdh

        breachedHighLevel := pdh

    if crossPwh

        breachedHighLevel := na(breachedHighLevel) ? pwh :
math.max(breachedHighLevel, pwh)

    if crossAsiaHigh

        breachedHighLevel := na(breachedHighLevel) ? asiaHigh :
math.max(breachedHighLevel, asiaHigh)

    if crossLondonHigh

        breachedHighLevel := na(breachedHighLevel) ? londonHigh :
math.max(breachedHighLevel, londonHigh)

    if crossOpen15High

        breachedHighLevel := na(breachedHighLevel) ? open15High :
math.max(breachedHighLevel, open15High)

    if crossOpen30High

        breachedHighLevel := na(breachedHighLevel) ? open30High :
math.max(breachedHighLevel, open30High)

    if crossOpen1hHigh

        breachedHighLevel := na(breachedHighLevel) ? open1hHigh :
math.max(breachedHighLevel, open1hHigh)

    isHighDisplaced := true

    displacedHighLevel := breachedHighLevel

    displacedHighLow := low

```

displacedHighBar := bar_index

if breachBearLogic

float breachedLowLevel = na

if underPdl

breachedLowLevel := pdl

if underPwl

breachedLowLevel := na(breachedLowLevel) ? pwl :
math.min(breachedLowLevel, pwl)

if underAsiaLow

breachedLowLevel := na(breachedLowLevel) ? asiaLow :
math.min(breachedLowLevel, asiaLow)

if underLondonLow

breachedLowLevel := na(breachedLowLevel) ? londonLow :
math.min(breachedLowLevel, londonLow)

if underOpen15Low

breachedLowLevel := na(breachedLowLevel) ? open15Low :
math.min(breachedLowLevel, open15Low)

if underOpen30Low

breachedLowLevel := na(breachedLowLevel) ? open30Low :
math.min(breachedLowLevel, open30Low)

if underOpen1hLow

breachedLowLevel := na(breachedLowLevel) ? open1hLow :
math.min(breachedLowLevel, open1hLow)

isLowDisplaced := true

displacedLowLevel := breachedLowLevel

displacedLowHigh := high

displacedLowBar := bar_index

if is5mChart

if isHighDisplaced and close > displacedHighLevel and low >
displacedHighLow and bar_index - displacedHighBar > 10

isHighDisplaced := false

if isLowDisplaced and close < displacedLowLevel and high <
displacedLowHigh and bar_index - displacedLowBar > 10

isLowDisplaced := false

bear5mCisd = is5mChart and isHighDisplaced and close <
displacedHighLevel and close < displacedHighLow

bull5mCisd = is5mChart and isLowDisplaced and close > displacedLowLevel
and close > displacedLowHigh

if bear5mCisd

isHighDisplaced := false

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

if showLabelsInput

entryPrice = close

stopPrice = high

risk = math.abs(stopPrice - entryPrice)

```

target1 = entryPrice - risk

target2 = entryPrice - risk * 2

line.new(bar_index, entryPrice, bar_index + 10, entryPrice, color =
INFO_COLOR, style = line.style_dotted, width = entryLineWidthInput)

line.new(bar_index, stopPrice, bar_index + 10, stopPrice, color =
BEAR_COLOR, style = line.style_dotted, width = entryLineWidthInput)

line.new(bar_index, target1, bar_index + 10, target1, color = BULL_COLOR,
style = line.style_dotted, width = entryLineWidthInput)

line.new(bar_index, target2, bar_index + 10, target2, color = BULL_COLOR,
style = line.style_dotted, width = entryLineWidthInput)

if bull5mCisd

isLowDisplaced := false

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

if showLabelsInput

entryPrice = close

stopPrice = low

risk = math.abs(entryPrice - stopPrice)

target1 = entryPrice + risk

target2 = entryPrice + risk * 2

line.new(bar_index, entryPrice, bar_index + 10, entryPrice, color =
INFO_COLOR, style = line.style_dotted, width = entryLineWidthInput)

line.new(bar_index, stopPrice, bar_index + 10, stopPrice, color =
BEAR_COLOR, style = line.style_dotted, width = entryLineWidthInput)

```

```
line.new(bar_index, target1, bar_index + 10, target1, color = BULL_COLOR,
style = line.style_dotted, width = entryLineWidthInput)
```

```
line.new(bar_index, target2, bar_index + 10, target2, color = BULL_COLOR,
style = line.style_dotted, width = entryLineWidthInput)
```

```
plotshape(showLabelsInput and bear5mCisd, title = '5m Bearish Cisd', text =
'Cisd', style = shape.labeldown, location = location.abovebar, color =
BEAR_COLOR, textcolor = color.white, size = size.small)
```

```
plotshape(showLabelsInput and bull5mCisd, title = '5m Bullish Cisd', text =
'Cisd', style = shape.labelup, location = location.belowbar, color =
BULL_COLOR, textcolor = color.white, size = size.small)
```

```
// --- Gold Session Awareness State ---
```

```
newNewYorkDay = ta.change(time('D', TZ)) != 0
```

```
inComexAwareness = not na(time(timeframe.period, comexSessionInput, TZ))
```

```
asiaSessionActive = isInSession('1800-0000')
```

```
var bool asiaHighSwept = false
```

```
var bool asiaLowSwept = false
```

```
var bool asiaHighTouchContext = false
```

```
var bool asiaLowTouchContext = false
```

```
var bool asiaHighTouchAlerted = false
```

```
var bool asiaLowTouchAlerted = false
```

```
var bool owlIssued = false
```

```
var label owlHighLabel = na
```

```
var label owlLowLabel = na
```

```
if newNewYorkDay
```

```
    asiaHighSwept := false
```

```
    asiaLowSwept := false
```

```
    asiaHighTouchContext := false
```

```
    asiaLowTouchContext := false
```

```
    asiaHighTouchAlerted := false
```

```
    asiaLowTouchAlerted := false
```

```
    owlIssued := false
```

```
    label.delete(owlHighLabel)
```

```
    label.delete(owlLowLabel)
```

```
    owlHighLabel := na
```

```
    owlLowLabel := na
```

```
if not asiaSessionActive and not na(asiaHigh) and high >= asiaHigh
```

```
    asiaHighTouchContext := true
```

```
if not asiaSessionActive and not na(asiaLow) and low <= asiaLow
```

```
    asiaLowTouchContext := true
```

```
confluenceDistance = confluenceDistanceInput * syminfo.mintick
```

```
asiaHighConfluence = not na(asiaHigh) and not na(pdh) and
```

```
math.abs(asiaHigh - pdh) <= confluenceDistance
```

```
asiaLowConfluence = not na(asiaLow) and not na(pdl) and math.abs(asiaLow  
- pdl) <= confluenceDistance
```

```
// --- FVG-123 Logic ---
```

```
type FVG
```

```
    float top
```

```
    float bottom
```

```
    bool isBull
```

```
    bool active
```

```
    int createdBar
```

```
    bool touched
```

```
    bool done
```

```
    int lastTouchBar
```

```
var array<FVG> 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))
```

```
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
```

```
bool buyFvg123Only = false
```

```
bool sellFvg123Only = false
```

```
bool unmitigatedSweepFvg123Detected = false
```

```
bool unmitigatedHighSweep = false
```

```
bool unmitigatedLowSweep = false
```

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

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

```
        int fvgIndex = array.size(fvgArray) - 1 - offset
```

```
        FVG fvgObj = array.get(fvgArray, fvgIndex)
```

```

if fvgObj.active

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

        fvgObj.active := false

        fvgObj.done := true

    if not fvgObj.done

        afterCreation = bar_index > fvgObj.createdBar

        if afterCreation 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 afterCreation and (fvgObj.isBull ? close < fvgObj.bottom : close >
fvgObj.top)

            fvgObj.done := true

        hasValidTouch = fvgObj.touched and not na(fvgObj.lastTouchBar) and
bar_index > fvgObj.lastTouchBar and afterCreation

        confirmation = (fvgObj.isBull and hasValidTouch and close >
fvgObj.top) or (not fvgObj.isBull and hasValidTouch and close <
fvgObj.bottom)

        if confirmation

            isFlowSetup = fvgObj.isBull ? lastIntention == 'Bullish' : lastIntention
== 'Bearish'

```

```
isSweepSetup = fvgObj.isBull ? lastIntention == 'Bearish' :  
lastIntention == 'Bullish'
```

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

```
flowEngaged = fvgObj.isBull ? isFlowSetup and touchedHigh :  
isFlowSetup and touchedLow
```

```
sweepEngaged = fvgObj.isBull ? (isSweepSetup and touchedLow) or  
low <= pdl or low <= asiaLow or low <= londonLow : (isSweepSetup and  
touchedHigh) or high >= pdh or high >= asiaHigh or high >= londonHigh
```

```
fvgValid = bar_index - fvgObj.createdBar <= fvgMaxAgeInput and  
math.abs(fvgObj.top - fvgObj.bottom) / syminfo.mintick >= fvgMinWidthInput
```

```
isKeyLevelResponse = isAcceptanceSetup or sweepEngaged or  
flowEngaged
```

```
if fvgValid and barstate.isconfirmed and (isKeyLevelResponse or not  
strictKeyLevelInput)
```

```
    fvg123SetupDetected := true
```

```
    if fvgObj.isBull
```

```
        buyFvg123Only := true
```

```
    else
```

```
        sellFvg123Only := true
```

```
    if isKeyLevelResponse
```

```
        keyLevelResponseDetected := true
```

bool confirmsAsiaHighSweep = not fvgObj.isBull and
asiaHighTouchContext and sweepEngaged and not asiaHighSwept

bool confirmsAsiaLowSweep = fvgObj.isBull and
asiaLowTouchContext and sweepEngaged and not asiaLowSwept

if confirmsAsiaHighSweep

asiaHighSwept := true

unmitigatedHighSweep := true

unmitigatedSweepFvg123Detected := true

if confirmsAsiaLowSweep

asiaLowSwept := true

unmitigatedLowSweep := true

unmitigatedSweepFvg123Detected := true

entryPrice = close

stopPrice = fvgObj.isBull ? fvgObj.bottom - syminfo.mintick :
fvgObj.top + syminfo.mintick

risk = math.abs(entryPrice - stopPrice)

targetPrice = fvgObj.isBull ? entryPrice + risk * 2 : entryPrice - risk *

2

setupColor = INFO_COLOR

setupName = 'FVG-123'

if isAcceptanceSetup

setupName := 'Acceptance'

else if sweepEngaged

```

        setupName := 'Sweep'

    else if flowEngaged

        setupName := 'Flow'

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

    if showLabelsInput and setupName != 'Acceptance'

        label.new(bar_index, textY, setupName + (fvgObj.isBull ? ' ▲' : ' ▼'), color = #00000000, textcolor = setupColor, style = fvgObj.isBull ?
label.style_label_up : label.style_label_down, size = size.normal)

        line.new(bar_index, entryPrice, bar_index + 10, entryPrice, color
= INFO_COLOR, style = line.style_dotted, width = entryLineWidthInput)

        line.new(bar_index, stopPrice, bar_index + 10, stopPrice, color =
BEAR_COLOR, style = line.style_dotted, width = entryLineWidthInput)

        line.new(bar_index, targetPrice, bar_index + 10, targetPrice,
color = BULL_COLOR, style = line.style_dotted, width = entryLineWidthInput)

    if setupName != 'Acceptance'

        table.cell(setupTable, 0, 0, 'Most Recent: ' + setupName,
text_color = setupColor, text_size = size.normal)

        alert(setupName + ' trade entry confirmation on ' + syminfo.ticker,
alert.freq_once_per_bar_close)

        fvgObj.done := true

        array.set(fvgArray, fvgIndex, fvgObj)

// Remove the O.W.L. marker as soon as its side receives a confirmed Sweep
FVG-123.

```

```
if unmitigatedHighSweep
```

```
    label.delete(owlHighLabel)
```

```
    owlHighLabel := na
```

```
if unmitigatedLowSweep
```

```
    label.delete(owlLowLabel)
```

```
    owlLowLabel := na
```

```
// --- O.W.L. and COMEX Touch Signals ---
```

```
// O.W.L. means Opportunity With Liquidity: the Asia level is still unmitigated  
when the COMEX window begins.
```

```
bool owlCreationEvent = inComexAwareness and not owlIssued and not  
na(asiaHigh) and not na(asiaLow)
```

```
if owlCreationEvent
```

```
    owlIssued := true
```

```
    if showOwlInput and not asiaHighSwept
```

```
        string highOwlText = asiaHighConfluence ? 'O.W.L.\nPDH Confluence' :  
'O.W.L.'
```

```
        owlHighLabel := label.new(bar_index, asiaHigh + atr * 0.25, highOwlText,  
color = #00000000, textcolor = OWL_COLOR, style = label.style_label_down,  
size = owlLabelSize)
```

```
    if showOwlInput and not asiaLowSwept
```

```
        string lowOwlText = asiaLowConfluence ? 'O.W.L.\nPDL Confluence' :  
'O.W.L.'
```

```
        owlLowLabel := label.new(bar_index, asiaLow + atr * 0.25, lowOwlText,  
color = #00000000, textcolor = OWL_COLOR, style = label.style_label_down,  
size = owlLabelSize)
```

```
bool asiaHighTouchSignal = inComexAwareness and not asiaHighSwept and  
not asiaHighTouchAlerted and not na(asiaHigh) and high >= asiaHigh
```

```
bool asiaLowTouchSignal = inComexAwareness and not asiaLowSwept and  
not asiaLowTouchAlerted and not na(asiaLow) and low <= asiaLow
```

```
if asiaHighTouchSignal
```

```
    asiaHighTouchAlerted := true
```

```
if asiaLowTouchSignal
```

```
    asiaLowTouchAlerted := true
```

```
plotshape(showAwarenessSignalsInput and asiaHighTouchSignal, title =  
'Unmitigated Asia High Touch', text = 'Touch', style = shape.triangledown,  
location = location.abovebar, color = OWL_COLOR, textcolor = color.yellow,  
size = size.small)
```

```
plotshape(showAwarenessSignalsInput and asiaLowTouchSignal, title =  
'Unmitigated Asia Low Touch', text = 'Touch', style = shape.triangleup, location  
= location.belowbar, color = OWL_COLOR, textcolor = color.yellow, size =  
size.small)
```

```
plotshape(showAwarenessSignalsInput and unmitigatedHighSweep, title =  
'Unmitigated Asia High Sweep FVG-123', text = 'U-Sweep', style =  
shape.labeldown, location = location.abovebar, color = OWL_COLOR,  
textcolor = color.black, size = size.small)
```

```
plotshape(showAwarenessSignalsInput and unmitigatedLowSweep, title =  
'Unmitigated Asia Low Sweep FVG-123', text = 'U-Sweep', style =  
shape.labelup, location = location.belowbar, color = OWL_COLOR, textcolor =  
color.black, size = size.small)
```

```
if asiaHighTouchSignal
```

```
    alert('Universal MFB - COMEX touched unmitigated Asia High on ' +  
syminfo.ticker, alert.freq_once_per_bar_close)
```

```
if asiaLowTouchSignal
```

```
    alert('Universal MFB - COMEX touched unmitigated Asia Low on ' +  
syminfo.ticker, alert.freq_once_per_bar_close)
```

```
if unmitigatedHighSweep
```

```
    alert('Universal MFB - Unmitigated Asia High Sweep FVG-123 on ' +  
syminfo.ticker, alert.freq_once_per_bar_close)
```

```
if unmitigatedLowSweep
```

```
    alert('Universal MFB - Unmitigated Asia Low Sweep FVG-123 on ' +  
syminfo.ticker, alert.freq_once_per_bar_close)
```

```
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(buyFvg123Only, 'Buy FVG-123 only', 'Buy FVG-123 trade entry confirmation on {{ticker}}')

alertcondition(sellFvg123Only, 'Sell FVG-123 only', 'Sell FVG-123 trade entry confirmation on {{ticker}}')

alertcondition(bull5mCisd, '5m Bullish CISD', '5m Bullish CISD detected on {{ticker}}')

alertcondition(bear5mCisd, '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(bull5mCisd or bear5mCisd or keyLevelResponseDetected, 'CISD / Key Level FVG', 'CISD or FVG-123 after key-level touch on {{ticker}}')

alertcondition(owlCreationEvent, 'O.W.L. available', 'O.W.L. unmitigated Asia liquidity is available for the COMEX window on {{ticker}}')

alertcondition(asiaHighTouchSignal, 'COMEX touched unmitigated Asia High', 'COMEX touched the unmitigated Asia High on {{ticker}}')

alertcondition(asiaLowTouchSignal, 'COMEX touched unmitigated Asia Low', 'COMEX touched the unmitigated Asia Low on {{ticker}}')

alertcondition(unmitigatedHighSweep, 'Unmitigated Asia High Sweep FVG-123', 'Unmitigated Asia High Sweep FVG-123 confirmed on {{ticker}}')

alertcondition(unmitigatedLowSweep, 'Unmitigated Asia Low Sweep FVG-123', 'Unmitigated Asia Low Sweep FVG-123 confirmed on {{ticker}}')

alertcondition(unmitigatedSweepFvg123Detected, 'Unmitigated Sweep FVG-123', 'Unmitigated Asia liquidity Sweep FVG-123 confirmed on {{ticker}}')