// This source code is subject to the terms of the Mozilla Public License 2.0 at https://mozilla.org/MPL/2.0/

// © ClayeWeight

// Sonarlabs - Order Block Finder

//

// This is a free script and a small snippet of what our premium Smart Money Concepts (SMC) Indicator offers.

// If you like this indicator I'd recommed trialing our SMC indicator as it offers even BETTER Order Blocks

// as well as; Market Structure, Volume Imbalances, LOTS of customisation - and MORE.

//

// Join our free discord if you want to stay up to date with updates or hang out with like minded people!

// https://discord.gg/jtcqBmXUe2

//

// You can trial our premium suite of indicators (includes the SMC indicator) by visiting our website

// https://www.sonarlab.io/

//

// ENJOY THE FREE SCRIPT

//@version=5

indicator(title='Sonarlabs OB Clone - by Rajesh ', shorttitle='Sonarlabs OB Clone by Rajesh', overlay=true, max\_boxes\_count=20)

\_v = input.string("1.0.2", title="Version", options=["1.0.2"], group="Sonarlab.io", tooltip="Modified version by Rajesh, Original : This is a free script based on our premium Smart Money Concepts (SMC) indicator. Get a free trial of our SMC indicator by visiting our website.\nhttps://www.sonarlab.io")

sens = input.int(28, minval=1, title='Sensitivity', group='Order Block', tooltip='Lower the sensitivity to show more order blocks. A higher sensitivity will show less order blocks.')

sens /= 100

//Rajesh : Input

blocksToShow = input.int(3, minval=1, maxval=25, title='No. of Recent Blocks', group='Order Block', tooltip='No. of RECENT order blocks to show.')

//Rajesh : End

//OB Colors

col\_bullish = input.color(#5db49e, title="Bullish OB Border", inline="a", group="Order Block")

col\_bullish\_ob = input.color(color.new(#64C4AC, 85), title="Background", inline="a", group="Order Block")

col\_bearish = input.color(#4760bb, title="Bearish OB Border", inline="b", group="Order Block")

col\_bearish\_ob = input.color(color.new(#506CD3, 85), title="Background", inline="b", group="Order Block")

// Alerts

buy\_alert = input.bool(title='Buy Signal', defval=true, group='Alerts', tooltip='An alert will be sent when price goes below the top of a bullish order block.')

sell\_alert = input.bool(title='Sell Signal', defval=true, group='Alerts', tooltip='An alert will be sent when price goes above the bottom of a bearish order block.')

// Delacring Variables

bool ob\_created = false

bool ob\_created\_bull = false

var int cross\_index = na

// Declaring Box Arrays

var box drawlongBox = na

var longBoxes = array.new\_box()

var box drawShortBox = na

var shortBoxes = array.new\_box()

// Custom Rate of Change (ROC) calculation. This is to calculate high momentum moves in the market.

pc = (open - open[4]) / open[4] \* 100

// If the ROC crossover our Sensitivty input - Then create an Order Block

// Sensitivty is negative as this is a Bearish OB

if ta.crossunder(pc, -sens)

ob\_created := true

cross\_index := bar\_index

cross\_index

// If the ROC crossover our Sensitivty input - Then create an Order Block

if ta.crossover(pc, sens)

ob\_created\_bull := true

cross\_index := bar\_index

cross\_index

// -------------------------------

// Bearish OB Creation

// -------------------------------

// Check if we should create a OB, Also check if we haven't created an OB in the last 5 candles.

if ob\_created and cross\_index - cross\_index[1] > 5

float last\_green = 0

float highest = 0

// Loop through the most recent candles and find the first GREEN (Bullish) candle. We will place our OB here.

for i = 4 to 15 by 1

if close[i] > open[i]

last\_green := i

break

// Draw our OB on that candle - then push the box into our box arrays.

drawShortBox := box.new(left=bar\_index[last\_green], top=high[last\_green], bottom=low[last\_green], right=bar\_index[last\_green], bgcolor=col\_bearish\_ob, border\_color=col\_bearish, extend=extend.right)

array.push(shortBoxes, drawShortBox)

//Rajesh : for Debugging

//label.new(bar\_index, na, "-OB: " + str.tostring(array.size(shortBoxes)) , yloc = yloc.abovebar, style = label.style\_none, textcolor = color.orange, size = size.normal)

// -------------------------------

// Bullish OB Creation

// -------------------------------

// Check if we should create a OB, Also check if we haven't created an OB in the last 5 candles.

if ob\_created\_bull and cross\_index - cross\_index[1] > 5

float last\_red = 0

float highest = 0

// Loop through the most recent candles and find the first RED (Bearish) candle. We will place our OB here.

for i = 4 to 15 by 1

if close[i] < open[i]

last\_red := i

break

// Draw our OB on that candle - then push the box into our box arrays.

drawlongBox := box.new(left=bar\_index[last\_red], top=high[last\_red], bottom=low[last\_red], right=bar\_index[last\_red], bgcolor=col\_bullish\_ob, border\_color=col\_bullish, extend=extend.right)

array.push(longBoxes, drawlongBox)

//Rajesh : for Debugging

//label.new(bar\_index, na, "+OB: " + str.tostring(array.size(longBoxes)) , yloc = yloc.abovebar, style = label.style\_none, textcolor = color.lime, size = size.normal)

// ----------------- Bearish Order Block Clean Up -------------------

// Clean up OB boxes and place alerts

if array.size(shortBoxes) > 0

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

sbox = array.get(shortBoxes, i)

top = box.get\_top(sbox)

bot = box.get\_bottom(sbox)

// If the two last closes are above the high of the bearish OB - Remove the OB

if close[1] > top and close[2] > top

array.remove(shortBoxes, i)

box.delete(sbox)

// Alerts

if high > bot and sell\_alert

alert('Price inside Bearish OB', alert.freq\_once\_per\_bar)

// ----------------- Bullish Order Block Clean Up -------------------

// Clean up OB boxes and place alerts

if array.size(longBoxes) > 0

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

sbox = array.get(longBoxes, i)

bot = box.get\_bottom(sbox)

top = box.get\_top(sbox)

// If the two last closes are below the low of the bullish OB - Remove the OB

if close[1] < bot and close[2] < bot

array.remove(longBoxes, i)

box.delete(sbox)

// Alerts

if low < top and buy\_alert

alert('Price inside Bullish OB', alert.freq\_once\_per\_bar)

//Rajesh : RECENT blocksToShow OBs, multiple Long/Short TPs

//Rajesh : Declaration

TBL\_POS\_TOP = position.top\_right

TBL\_POS\_BOTTOM = position.bottom\_right

TBL\_POS\_MIDDLE = position.middle\_right

TXT\_SIZE\_NORMAL = size.normal

TXT\_SIZE\_LARGE = size.large

//Arrays for multiple TPs

var tpsLong = array.new\_float()

var tpsShort = array.new\_float()

//Nearest (single) TPs

var float tpL = 0

var float tpS = 0

//Rajesh : End

//Helper Function to debug

printTable(txt, txtSize, pos, cellBG) => var table t = table.new(pos, 1, 1), table.cell(t, 0, 0, txt, text\_size = txtSize, bgcolor = color.new(cellBG, 75), text\_color = color.white)

// retain only RECENT blocksToShow OBs, remove rest ALL

if barstate.islast

//flush out the old TP values

array.clear(tpsLong)

array.clear(tpsShort)

//Bearish OBs :

for [i, sBox] in shortBoxes

if i < (array.size(shortBoxes) - blocksToShow)

//simply remove OB

box.delete(sBox)

else

//just to distinguish from original SonarLabs OBs

box.set\_border\_style(sBox, line.style\_dashed)

//TPs for Longs : take Bottom value

array.push(tpsLong, box.get\_bottom(sBox))

//sort TP values

array.sort(tpsLong)

//multiple TPs for Shorts : all TPs available in array tpsLong.

//Use array.get() method to get the 1st, 2nd, 3rd .. etc. TP (refer to code getting Single TP)

printTable("Multiple TPs for Long :\n\n" + (array.size(tpsLong) > 0 ? str.tostring(tpsLong) : "N/A"), TXT\_SIZE\_LARGE, TBL\_POS\_BOTTOM, color.green)

//Bullish OBs :

for [i, lBox] in longBoxes

if i < (array.size(longBoxes) - blocksToShow)

//simply remove OB

box.delete(lBox)

else

//just to distinguish from original SonarLabs OBs

box.set\_border\_style(lBox, line.style\_dashed)

//TPs for Shorts : take Top value

array.push(tpsShort, box.get\_top(lBox))

//sort TP values

array.sort(tpsShort, order.descending)

//multiple TPs for Longs : all TPs available in array tpsShort.

//Use array.get() method to get the 1st, 2nd, 3rd .. etc. TP (refer to code getting Single TP)

printTable("Multiple TPs for Short :\n\n" + (array.size(tpsShort) > 0 ? str.tostring(tpsShort) : "N/A"), TXT\_SIZE\_LARGE, TBL\_POS\_TOP, color.red)

//getting Single Long/Short TP

tpS := array.size(tpsShort) > 0 ? array.get(tpsShort, 0) : 0

tpL := array.size(tpsLong) > 0 ? array.get(tpsLong, 0) : 0

//Displaying Single TPs

strSingleTPs = "Single TP : \n\nLong = " + str.tostring(tpL) + "\nShort = " + str.tostring(tpS)

printTable(strSingleTPs, TXT\_SIZE\_LARGE, TBL\_POS\_MIDDLE, color.gray)

//Rajesh : End