```
//+------------------------------------------------------------------+
//|                  Script_ImportCalendarDataAndDisplayOnChart.mq4 |
//|                                  Copyright 2017, M Wilson.       |
//|                                https://www.algotrader.blog       |
//+------------------------------------------------------------------+
#include <C_Calendar_ForexFactoryImport.mqh>
#include <C_Chart_Drawing.mqh>

#property copyright "Copyright 2017, M Wilson"
#property link      "https://www.algotrader.blog"
#property version   "1.00"
#property strict
#property show_inputs

//+------------------------------------------------------------------+
//| Constants                                                        |
//+------------------------------------------------------------------+
extern int I_CalendarOffsetInMinutes=120;          //Calendar Offset in mins
extern bool I_DeleteVerticalLinesBeforeRun=True;   //Delete Vertical Line
extern bool I_ImportLowImpact=False;               //Low Impact Events
extern bool I_ImportMediumImpact=False;            //Medium Impact Events
extern bool I_ImportHighImpact=True;               //High Impact Events
extern int I_BankHolidayLineWidth=1;
//Width of Bank Holiday Lines.   1 will use LineStyle Dot.
extern color I_BankHolidayLineColor=clrWhite;      //Color of Bank Holiday Lines.
extern int I_CalEventLineWidth=1;
//Width of News Event Lines.   1 will use LineStyle Dot.
extern bool I_PlotAllDayEvents=False;              //Do we process All Day events.
extern int I_ExpandAllDayEventsByXHours=0;         //Expands all day events by X hours

//+------------------------------------------------------------------+
//| Global Variables                                                 |
//+------------------------------------------------------------------+
C_Calendar_ForexFactoryImport *g_Calendar;
//+------------------------------------------------------------------+
//| Script program start function                                    |
//+------------------------------------------------------------------+
void OnStart()
  {

//--- Build the Calendar List - MAY NEED TO ADJUST THE 0 TO MATCH YOUR CALENDAR TIME.
   string strCountryList[];
   populateCountryArray(strCountryList);
   g_Calendar = new C_Calendar_ForexFactoryImport(strCountryList,I_ImportLowImpact,
I_ImportMediumImpact,I_ImportHighImpact,I_CalendarOffsetInMinutes*60);

//--- Import the Calendar
   //g_Calendar.SyncCalendar("ffcal_week_this.xml");
   g_Calendar.SyncCalendar("ffcal_week_history.xml");
   Print(__FUNCTION__," Calendar, Number of Events: ",g_Calendar.ArraySizeCalendar(),
", No BankHols: ",g_Calendar.ArraySizeBankHols(),", No All Day Events: ",
g_Calendar.ArraySizeAllDayEvents());

//--- Build the Chart Drawing Class
   C_Chart_Drawing *objChart = new C_Chart_Drawing();

//--- Delete any vertical lines from the chart
   if(I_DeleteVerticalLinesBeforeRun)  objChart.DeleteAllVerticleLines();


//--- Scan through the bars in the chart and see if there are any events between the candles
   for(int i=1;i<Bars;i++)
   {
      datetime dtStart=iTime(Symbol(),0,i);
      datetime dtEnd=iTime(Symbol(),0,i-1);

      //Add white vertical lines if today is a bank holiday
      if(IsBankHoliday(dtStart))
      {
         string strName="BH_"+IntegerToString((int)dtStart);
         ENUM_LINE_STYLE eStyle=STYLE_DOT;
         if(I_BankHolidayLineWidth>1)  eStyle=STYLE_SOLID;
```

```
         objChart.CreateVerticalLine(0,strName,0,dtStart,I_BankHolidayLineColor,eStyle,
I_BankHolidayLineWidth,True);
      }

      //Process Standard Calendar Events
      NewsImpact eImpact;
      if(g_Calendar.HighestImpactEventBetweenDates(dtStart,dtEnd,eImpact))
      {
         //Found an event, plot it on the graph.

         string strName="CAL_"+IntegerToString((int)dtStart);
         color colLine=clrYellow;
         if(eImpact==MediumImpact)
         {
            colLine=clrOrange;
         }
         else if(eImpact==HighImpact)
         {
            colLine=clrRed;
         }
         ENUM_LINE_STYLE eStyle=STYLE_DOT;
         if(I_CalEventLineWidth>1)  eStyle=STYLE_SOLID;

         objChart.CreateVerticalLine(0,strName,0,dtStart,colLine,eStyle,
I_CalEventLineWidth,True);
      }

      //Process All Day Events
      if(I_PlotAllDayEvents)
      {

         datetime dtNewStart=dtStart+(I_ExpandAllDayEventsByXHours*60*60);
         datetime dtNewEnd=dtStart-(I_ExpandAllDayEventsByXHours*60*60);
         NewsImpact eImpactStart,eImpactEnd;

         if(g_Calendar.HighestImpactAllDayEvent(dtNewStart,eImpactStart) ||
g_Calendar.HighestImpactAllDayEvent(dtNewEnd,eImpactEnd))
         {
            //Found an event, plot it on the graph.

            string strName="CAL_ALLDAY_"+IntegerToString((int)dtStart);
            color colLine=clrYellow;
            if(eImpactStart==MediumImpact || eImpactEnd==MediumImpact)
            {
               colLine=clrOrange;
            }
            else if(eImpactStart==HighImpact || eImpactEnd==HighImpact)
            {
               colLine=clrRed;
            }
            ENUM_LINE_STYLE eStyle=STYLE_DOT;
            if(I_CalEventLineWidth>1)  eStyle=STYLE_SOLID;

            objChart.CreateVerticalLine(0,strName,0,dtStart,colLine,eStyle,
I_CalEventLineWidth,True);
         }
      }
   }

   objChart.RefreshChart();

   delete g_Calendar;
   delete objChart;

  }
//+------------------------------------------------------------------+
bool IsBankHoliday(const datetime dtIn=0)
{
   bool boolRet=False;


//We do not create a new trade on bank holidays.   Do not auto-include the US in the countri
   string strCountries[];
```

```
   populateCountryArray(strCountries,False);
   int intSize=ArraySize(strCountries);
   for(int i=0;i<intSize;i++)
   {
      string strC=strCountries[i];
      if(StringLen(strC)>0)
      {
         if(strC=="EUR")
         {  //Only count EUR bank holidays that include France and Germany.
            bool boolFrench=g_Calendar.TodayIsBankHoliday(dtIn,strC,"France") ||
g_Calendar.TodayIsBankHoliday(dtIn,strC,"French");
            bool boolGerman=g_Calendar.TodayIsBankHoliday(dtIn,strC,"German");
            if(boolFrench && boolGerman)
            {
               boolRet=True;
               break;
            }
         }
         else
         {
            if(g_Calendar.TodayIsBankHoliday(dtIn,strC,""))
            {
               boolRet=True;
               break;
            }
         }
      }
   }
   ArrayFree(strCountries);

   return boolRet;
}
void populateCountryArray(string &strCountries[], const bool boolAddUSIfMissing=True,
const bool boolAddALLCountry=True)
{

//We are going to ensure that we have the account currency, the base currency, the profit cu
   //in the array of countries.

   bool boolAddUS=True;
   int intSize=ArraySize(strCountries);

   string strAccount=StringTrimLeft(StringTrimRight(AccountCurrency()));
   StringToUpper(strAccount);
   if(strAccount=="USD")   boolAddUS=False;
   ArrayResize(strCountries,intSize+1);
   strCountries[intSize]=strAccount;
   intSize++;

   string strBase=StringTrimLeft(StringTrimRight(SymbolInfoString(Symbol(),
SYMBOL_CURRENCY_BASE)));
   StringToUpper(strBase);
   if(strBase=="USD")   boolAddUS=False;
   if(strBase!=strAccount)
   {
      ArrayResize(strCountries,intSize+1);
      strCountries[intSize]=strBase;
      intSize++;
   }

   string strProfit=StringTrimLeft(StringTrimRight(SymbolInfoString(Symbol(),
SYMBOL_CURRENCY_PROFIT)));
   StringToUpper(strProfit);
   if(strProfit=="USD")   boolAddUS=False;
   if(strProfit!=strAccount && strProfit!=strBase)
   {
      ArrayResize(strCountries,intSize+1);
      strCountries[intSize]=strProfit;
      intSize++;
   }

   if(boolAddUS && boolAddUSIfMissing)
   {
```

```
      ArrayResize(strCountries,intSize+1);
      strCountries[intSize]="USD";
      intSize++;
   }

   if(boolAddALLCountry)
   {
      ArrayResize(strCountries,intSize+1);
      strCountries[intSize]="ALL";
      intSize++;
   }

}
```