○「New MQL4（Build 600 以降）；基礎（その４）OnChartEvent()［１/２］」 2014.09.19

・アメンボです、

本稿での報告は「OnChartEvent()」関数です。
名称から推測される様に、この関数はチャート上オブジェクト（主にグラフィック）の
イベント発生時（例えばボタンが押された時）に呼出（割込み）されます。
これを呼び出すイベントは、大きく下記の「２種類」に分類されています。

① MQL5（システム）備え付けのイベント
② ユーザーが任意に設定するカスタム・イベント

本稿では、上記の「①」の検証結果を解説します。

・ところで、

既にお気づきの読者がおられるとおもいますが、本稿での検証結果内容は以前に筆者が
MQL5 の翻訳調査内容（未実行）として報告済みの内容と９９％同一です。
つまり、MQL5 の資産は特殊なもの（OnTrade()や OnBookEvent()等、NewMQL4 に無いもの）を
除き、多くは略そのままのコードでも、New MQL4 上で動作する！、と言う事です。

＜ MQL5 の過去の遺産は発掘する価値がある！、です ＞

！本当は、

New MQL4 でも「OnTrade()や OnBookEvent()」がサポートされると嬉しいのですが、
MQL5 の OnTrade()では「deal」の概念が入っているので難しいのかも！
しかし、New MQL4（Build670）では「板画面表示」はされるので、この先に何か進展が？？
（ でも、この「板画面表示」の使い方と、メリットが良く分からない！！ ）

＜本稿で使用した MQL4 コード＞

※使用コード；「new_mql4_2014_09_02.zip；New MQL4 ChartEvent() [1/2]」（ZIP 形式圧縮）
※本稿は「MT4；version 4.00 Build670」「MetaEditor；version 5.00 Buid966」にて確認済み。

目次：

1．「On****()」ハンドリング関数（MQL5 との比較）

New_MQL4 で使用可能な「ハンドリング関数」を MQL5 と比較しながら、使用方法を解説します

		機能ｻﾎﾟｰﾄ		New MQL4 のｻﾎﾟｰﾄ範囲			確認
				EA	Indicator	Script	
ﾊﾝﾄﾞﾘﾝｸﾞ関数	ｲﾍﾞﾝﾄ・ﾄﾘｶﾞ と ﾓｰﾄﾞ別	MQL5	New MQL4	関数 使用	ｲﾝﾃﾞｨｹｰﾀ 表示	ｽｸﾘﾌﾟﾄ 実行	
OnStart()	—	○	○	○	—	○	
OnInit()	開始	○	○	○	○	—	
OnDeinit()	終了	○	○	○	○	—	
OnTick()	ﾃｨｯｸ	○	○	○	—	—	
	ﾏﾙﾁｶﾚﾝｼｰ･ﾓｰﾄﾞ	○	?	?	—	—	
OnTimer()	ﾀｲﾏｰ	○	○	○	○	—	**済**
OnTrade()	order・deal・position	○	—	—	—	—	
OnTradeTransaction()		○	—	—	—	—	
OnTester()	ｽﾄﾗﾃｼﾞｰ・ﾃｽﾀｰ	○	○	○	—	—	**済**
OnBookEvent()	板(DOM)情報	○	—	—	—	—	
OnChartEvent()	**10 種類**	○	○	○	○	—	**本稿**
	ｶｽﾀﾑ･ｲﾍﾞﾝﾄ	○	○	○	○	—	
OnCalculate()	ｲﾝﾃﾞｨｹｰﾀ表示計算	○	○	—	○	—	**済**
	簡略ﾀｲﾌﾟ	○	?	—	?	—	

2．OnChartEvent()の関数書式と引数

```
void OnChartEvent (
                const int id,           //イベント ID（識別子）
                const long &lparam,     //イベント・パラメータ（long タイプ）
                const double &dparama, //イベント・パラメータ（double タイプ）
                const string &sparam    //イベント・パラメータ（string タイプ）
                  )
```

※「id」により、どの様なイベントが発生したかを判別することが可能であり、また
「パラメータ；lparam、dparam、sparam」により更に詳細な情報を得ることが出来る。

例えば、「id」によりマウスがチャート上で「クリック」されたことを判別し、
「パラメータ」により、クリックされたチャート上の「位置」を知る事が出来る。

３．Chart Event の種類（OnChartEvent()が呼出されるイベント）

・OnChartEvent()を呼び出すイベントの内、システム備え付けのものは「10 種類」あります。

＜イベント；ID とパラメータ＞

	ｲﾍﾞﾝﾄ（割込）発生	ID	概要	返し値		
				lparam	dparama	sparam
1	キーが押された	CHARTEVENT_KEYDOWN	どのｷｰが押されたか	ｷｰ・ｺｰﾄﾞ	―	―
2	マウスが動いた	CHARTEVENT_MOUSE_MOVE	ﾏｳｽの動きをﾌｫﾛｰ	X 座標	Y 座標	ﾋﾞｯﾄ・ﾏｽｸ値 ﾎﾞﾀﾝ検出用
3	ｸﾞﾗﾌｨｶﾙ・ｵﾌﾞｼﾞｪｸﾄの作成	CHARTEVENT_OBJECT_CREATE	―	―	―	作成された ｵﾌﾞｼﾞｪｸﾄ名
4	ｸﾞﾗﾌｨｶﾙ・ｵﾌﾞｼﾞｪｸﾄの変更	CHARTEVENT_OBJECT_CHANGE	―	―	―	変更された ｵﾌﾞｼﾞｪｸﾄ名
5	ｸﾞﾗﾌｨｶﾙ・ｵﾌﾞｼﾞｪｸﾄの削除	CHARTEVENT_OBJECT_DELETE	―	―	―	削除された ｵﾌﾞｼﾞｪｸﾄ名
6	ﾁｬｰﾄ上で ﾏｳｽがｸﾘｯｸされた	CHARTEVENT_CLICK	ｸﾘｯｸした座標検出	X 座標	Y 座標	―
7	ｸﾞﾗﾌｨｶﾙ・ｵﾌﾞｼﾞｪｸﾄ上でﾏｳｽ・ｸﾘｯｸされた	CHARTEVENT_OBJECT_CLICK	ｵﾌﾞｼﾞｬｸﾄがある ﾁｬｰﾄ上の座標検出	X 座標	Y 座標	ｸﾘｯｸされた ｵﾌﾞｼﾞｪｸﾄ名
8	ｸﾞﾗﾌｨｶﾙ・ｵﾌﾞｼﾞｪｸﾄがﾏｳｽでﾄﾞﾗｯｸﾞされた	CHARTEVENT_OBJECT_DRAG	―	―	―	ﾄﾞﾗｯｸﾞされた ｵﾌﾞｼﾞｪｸﾄ名
9	ｵﾌﾞｼﾞｪｸﾄのﾗﾍﾞﾙが編集された	CHARTEVENT_OBJECT_ENDEDIT	―	―	―	ﾗﾍﾞﾙ編集済み ｵﾌﾞｼﾞｪｸﾄ名
10	チャート変更	CHARTEVENT_CHART_CHANGE	表示ﾁｬｰﾄの変更	―	―	―
11	ﾕｰｻﾞｰ定義ｲﾍﾞﾝﾄが発生した	CHARTEVENT_CUSOM+N	―	**※１**	**※１**	**※１**

※「パラメータ」を解析することで、詳細情報が判明する。

※１；EventChartCustom() によって設定した値が返る・・・本件は「**別稿**」にて解説予定

N=0 の場合は「ID= CHARTEVENT_CUSOM」、N=LastNo の場合は「ID= CHARTEVENT_CUSOM_LAST」

４．OnChartEvent()の使い方例（呼び出し例）

（１）例１；キーボード・イベント

－１．イベント（割込）概要；

・キーボード・イベントが発生する（キーが押される）と、OnChartEvent()が呼び出され、その「イベント・パラメータ；lparam」に押されたキーのコードが乗ってくる。

・キーボード・イベント；ＩＤとパラメータ（再確認）　　＜解説用の抜粋＞

	ｲﾍﾞﾝﾄ（割込）発生	ID（識別子）	概要	返し値		
				lparam	dparama	sparam
1	キーが押された	CHARTEVENT_KEYDOWN	どのｷｰが押されたか	ｷｰ･ｺｰﾄﾞ	―	―

－２．上記のコード構成（例）；

```
//+------------------------------------------------------------------+
//|                                                ChartEvent_02.mq4 |
//|                                                           amenbo |
//|                                                 泉の森の弁財天池 |
//+------------------------------------------------------------------+
#property copyright "amenbo"
#property link      "泉の森の弁財天池"
#property version   "1.00"
#property strict
#property indicator_chart_window
//
#define KEY_NUMPAD_5       12 //  Ｔ５  テンキー NumLock_OFF
#define KEY_LEFT           37 //  ←  キー
#define KEY_UP             38 //  ↑  キー
#define KEY_RIGHT          39 //  →  キー
#define KEY_DOWN           40 //  ↓  キー
#define KEY_NUMLOCK_DOWN   98 //  Ｔ２  テンキー NumLock_ON
#define KEY_NUMLOCK_LEFT  100 //  Ｔ４  テンキー NumLock_ON
#define KEY_NUMLOCK_5     101 //  Ｔ５  テンキー NumLock_ON
#define KEY_NUMLOCK_RIGHT 102 //  Ｔ６  テンキー NumLock_ON
#define KEY_NUMLOCK_UP    104 //  Ｔ８  テンキー NumLock_ON
//+------------------------------------------------------------------+
//| Custom indicator initialization function                         |
//+------------------------------------------------------------------+
int OnInit()
  {
   //--- indicator buffers mapping
   //---
   return(INIT_SUCCEEDED);
  }
//
void OnDeinit(const int reason)
  {
   //---
  }
//+------------------------------------------------------------------+
//| Custom indicator iteration function                              |
//+------------------------------------------------------------------+
int OnCalculate(const int rates_total,
                const int prev_calculated,
                const datetime &time[],
                const double &open[],
                const double &high[],
                const double &low[],
                const double &close[],
                const long &tick_volume[],
                const long &volume[],
                const int &spread[])
  {
   //---
```

```
   //  ここにインディケータ本体を記述
   //--- return value of prev_calculated for next call
   return(rates_total);
  }
  //
//+------------------------------------------------------------------+
//| ChartEvent function                                              |
//+------------------------------------------------------------------+
//チャート・イベント検出（この場合はキーボード・イベント）
void OnChartEvent(const int id,         //  「CHARTEVENT_KEYDOWN」が返される
                  const long &lparam,   //  押された「キーのコード」が返される
                  const double &dparam, // (使用せず)
                  const string &sparam  // (使用せず)
                  )
  {
  //キーボード上のキーが押されたら実行する
   if(id==CHARTEVENT_KEYDOWN)
     {
      switch((int)lparam)
        {
         case KEY_NUMLOCK_LEFT:  Print("[T4] テンキーが押された");    break;
         case KEY_LEFT:          Print("[←] が押された");            break;
         case KEY_NUMLOCK_UP:    Print("[T8] テンキーが押された");    break;
         case KEY_UP:            Print("[↑] が押された");            break;
         case KEY_NUMLOCK_RIGHT: Print("[T6] テンキーが押された");    break;
         case KEY_RIGHT:         Print("[→] が押された");            break;
         case KEY_NUMLOCK_DOWN:  Print("[T2] テンキーが押された");    break;
         case KEY_DOWN:          Print("[↓] が押された");            break;
         case KEY_NUMPAD_5:      Print("[T5] が押された NumLock_OFF"); break;
         case KEY_NUMLOCK_5:     Print("[T5] が押された NumLock_ON "); break;
         default:                Print("定義した以外のキーが押された");
        }
   //
      ChartRedraw();
     }
   //
      PlaySound("tick");
   //
  }
```

－３．実行結果

※キーボードの「押されたキー」に従って、プリント文が実行される。

（2）例２；グラフィック・オブジェクト（ボタン）のクリック検出

－１．イベント（割込）概要；

・チャート上のオブジェクトをマウスでクリックすると、割込みが発生し
OnChartEvent()で、これを検出することができる。

＜解説用の抜粋＞

	ｲﾍﾞﾝﾄ種類	ID（識別子）	概要	返し値		
				lparam	dparama	sparam
7	ｸﾞﾗﾌｨｶﾙ・ｵﾌﾞｼﾞｪｸﾄ上でﾏｳｽ･ｸﾘｯｸされた	CHARTEVENT_OBJECT_CLICK	ﾁｬｰﾄ上のｵﾌﾞｼﾞｪｸﾄ（ﾎﾞﾀﾝ等）名や、座標検出など	X 座標	Y 座標	ｸﾘｯｸされたｵﾌﾞｼﾞｪｸﾄ名

－２．上記のコード構成（例）；

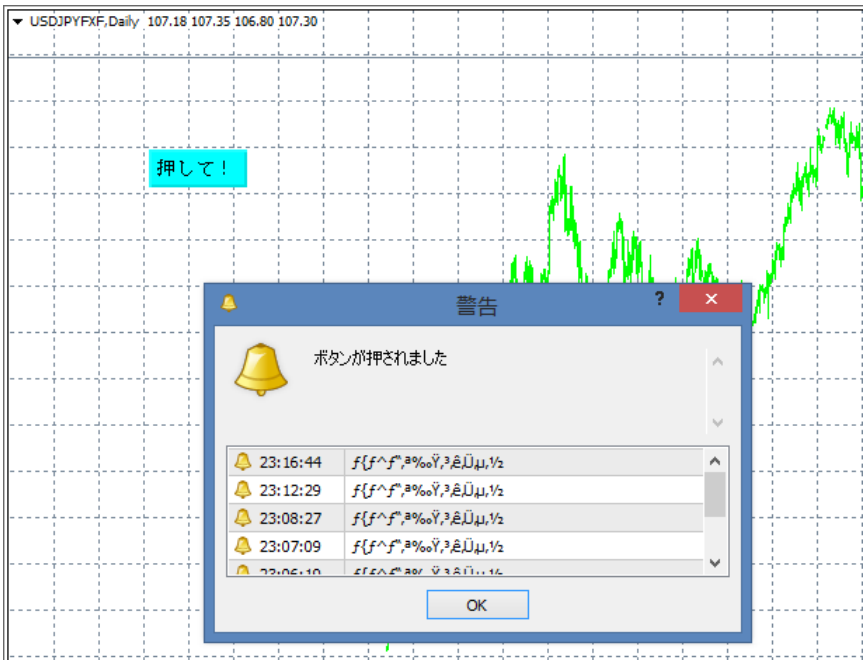
```
//+------------------------------------------------------------------+
//|                                                 ChartEvent_01.mq4 |
//|                                                           amenbo |
//|                                                 泉の森の弁財天池 |
//+------------------------------------------------------------------+
#property copyright "amenbo"
#property link      "泉の森の弁財天池"
#property version   "1.00"
#property strict
//
//    標準関数のみ使用、標準クラスは使わないで記述
//    このレベルでは、標準クラスを使うメリットは無し
//+------------------------------------------------------------------+
//| Expert initialization function                                   |
//+------------------------------------------------------------------+
int OnInit()
  {
   //---
        bool Result;
        ResetLastError();
        //作成するボタンの属性（ﾌﾟﾛﾊﾟﾃｨｰ）
        long Chart_ID=0;
        string Name="button1";
        string Title ="押して！";
        color BorderColor = Black;
        color ButtonColor = Aqua;
        int DistanceX = 100;
        int DistanceY = 100;
        int Width = 70;
        int Height = 25;
   //
        //新規にボタンを作成
        Result = ObjectCreate(Chart_ID,Name,OBJ_BUTTON,0,0,0);
        //ボタンにタイトルを設定
        ObjectSetString(Chart_ID,Name,OBJPROP_TEXT,Title);
        ObjectSetString(Chart_ID,Name,OBJPROP_FONT,"ＭＳ 明朝");

        ObjectSetInteger(Chart_ID,Name,OBJPROP_SELECTABLE,false);
        //ボタンをチャート上に置く位置を設定
        ObjectSetInteger(Chart_ID,Name,OBJPROP_XDISTANCE,DistanceX);
        ObjectSetInteger(Chart_ID,Name,OBJPROP_YDISTANCE,DistanceY);
```

```
        //ボタンのサイズ（幅と高さ）を設定
        ObjectSetInteger(Chart_ID,Name,OBJPROP_XSIZE,Width);
        ObjectSetInteger(Chart_ID,Name,OBJPROP_YSIZE,Height);
        //ボタンの色を設定（境界色と塗りつぶす色）
        ObjectSetInteger(Chart_ID,Name,OBJPROP_COLOR,BorderColor);
        ObjectSetInteger(Chart_ID,Name,OBJPROP_BGCOLOR,ButtonColor);
        //
        if(_LastError!=0)
        {
               Result = false;
        }
    //---
    return(INIT_SUCCEEDED);
  }
//+------------------------------------------------------------------+
//| Expert deinitialization function                                 |
//+------------------------------------------------------------------+
void OnDeinit(const int reason)
{
        //ボタン削除
        ObjectsDeleteAll(0);
}
//+------------------------------------------------------------------+
//| Expert tick function                                             |
//+------------------------------------------------------------------+
void OnTick()
  {
//---
        //通常は、ここがＥＡの本体コード
  }
//+------------------------------------------------------------------+
//| ChartEvent function                                              |
//+------------------------------------------------------------------+
//チャート・イベント検出（この場合はオブジェクトのクリック）
void OnChartEvent(const int id,         //  返されるイベント id
                  const long &lparam,   //  通常「Ｘ座標」が返される
                  const double &dparam, //  通常「Ｙ座標」が返される
                  const string &sparam  //「オブジェクト名称」が返される
                  )
{
   //id として「CHARTEVENT_OBJECT_CLICK」が返されたら
  if(id==CHARTEVENT_OBJECT_CLICK)
  {
    if(sparam=="button1")
    {
      if(ObjectGetInteger(0,sparam,OBJPROP_STATE)==true)
      {
      Alert("ボタンが押されました");
      ObjectSetInteger(0,sparam,OBJPROP_STATE,0);//ボタンを押されていない状態に戻す
      //
      ChartRedraw();
      }
    }
    //  確認用に print する
    Print("(X,Y)= (",lparam,",",DoubleToString(dparam,0),")");
    Print("Object_Name= ",sparam);
  }
   //
}
//--------------------------------------------------------------------------
```

－３．実行結果

※チャート上に作成した［ボタン］をクリックすると、「ボタンが押されました」と
アラート・ボックスに表示される。

押して！　をクリック　　↓



※アラート内の履歴表示が文字化けしているのですが、修正方法が未だ判らず！？

［エキスパート］タブの表示；

５．参考

（１）キー・コード一覧（例）・・・「青書部」を「例 1」で使用した

アルファベット				数字		テンキー数字		テンキー記号		記号	
A	65	O	79	0	48	T0	96	T*	106	:*	186
B	66	P	80	1	49	T1	97	T+	107	;+	187
C	67	Q	81	2	50	T2	98			,<	188
D	68	R	82	3	51	T3	99	T-	109	-=	189
E	69	S	83	4	52	T4	100	T.	110	.>	190
F	70	T	84	5	53	T5	101	T/	111	/?	191
G	71	U	85	6	54	T6	102			@`	192
H	72	V	86	7	55	T7	103			[{	219
I	73	W	87	8	56	T8	104			¥|	220
J	74	X	88	9	57	T9	105			]}	221
K	75	Y	89							^~	222
L	76	Z	90							¥_	226
M	77										
N	78										

ファンクションキー		制御キー					
F1（ヘルプ）	112	BackSpace	8	End	35	英数	240
F2	113	NumLockOFF の T5	12	Home	36	カタカナ/ひらがな	242
F3（検索）	114	Enter / T Enter	13	←	37	Esc	243
F4（アドレスバー）	115	Shift	16	↑	38	半角/全角	244
F5（更新）	116	Ctrl	17	→	39	Tab	9
F6（フォーカス）	117	Alt	18	↓	40		
F7	118	Pause	19	Insert	45		
F8	119	変換	28	Delete	46		
F9	120	無変換	29	Win	91		
F10（Alt）	121	スペース	32	Apps	93		
F11（全画面）	122	PageUp	33	NumLock	144		
F12	123	PageDown	34	ScrollLock	145		

（2）ObjectCreate 関数

・ObjectCreate 関数は「初期座標」に「指定名称・タイプ」のオブジェクトを作成する、
同一のオブジェクトを一度に 30 個まで置く事ができる。

```
bool  ObjectCreate(
   long          chart_id,       // チャート識別子
   string        name,           // オブジェクト名
   ENUM_OBJECT   type,           // オブジェクト・タイプ
   sub_window    nwin,           // ウインドウ・インデックス（オブジェクトを置く）
   datetime      time1,          // 第１番アンカー・ポイント「時間」座標
   double        price1,         // 第１番アンカー・ポイント「価格」座標
   ...
   datetime      timeN=0,        // 第Ｎ番アンカー・ポイント「時間」座標
   double        priceN=0,       // 第Ｎ番アンカー・ポイント「価格」座標
   );
```

パラメータ；

chart_id	[in]	チャート識別子、「0」は現在のチャートを示す
name	[in]	オブジェクト名称、サブウインドウを含むチャート上で、ユニークな名称であること。
type	[in]	オブジェクト・タイプ、ENUM_OBJECT 列挙型の一つであること。
sub_window	[in]	チャート・サブウインドウ番号、「0」はメイン・チャート。指定した番号が存在しないと「false」を返す。
time1	[in]	第１番アンカー・ポイント「時間」座標
price1	[in]	第１番アンカー・ポイント「価格」座標
timeN=0	[in]	第Ｎ番アンカー・ポイント「時間」座標、デフォルトは「0」
priceN=0	[in]	第Ｎ番アンカー・ポイント「価格」座標、デフォルトは「0」

返し値；

新規に、オブジェクト作成；

・成功　→　TRUE、　・失敗　→　FALSE

既に、同一「名称、タイプ」のオブジェクトが存在している場合；

・「時間・価格」座標を変更する

※オブジェクト・タイプによって、指定が必要な「アンカー・ポイント」数が異なる。

ENUM_OBJECT　列挙型　＜解説用の抜粋＞

チャート識別子	種類	アンカー・ポイント
OBJ_BUTTON	**ボタン**	アンカー・ポイントに対して OBJPROP_XDISTANCE と OBJPROP_YDISTANCE を指定する

ENUM_OBJECT_PROPERTY_STRING　列挙型　＜解説用の抜粋＞

・「ObjectSetString()」で使用

識別子	解　説	ﾌﾟﾛﾊﾟﾃｨｰ・ﾀｲﾌﾟ
OBJPROP_TEXT	オブジェクトの記述（テキスト解説）に用いる	string

ENUM_OBJECT_PROPERTY_INTEGER　**列挙型**　　＜解説用の抜粋＞

・「ObjectSetInteger()」と「ObjectGetInteger()」で使用する

識別子	解　説	ﾌﾟﾛﾊﾟﾃｨｰ・ﾀｲﾌﾟ
OBJPROP_COLOR	配色	color
OBIPOP_BGCOLOR	オブジェクトのバックグラウンド（塗り潰し）色 対象；OBJ_EDIT, OBJ_BUTTON, OBJ_RECTANGLE_LABEL	color
OBJPROP_SELECTABLE	オブジェクト選択可否	bool
OBJPROP_XDISTANCE	アンカー・ポイントから、オブジェクトを X軸方向に動かすピクセル数	int
OBJPROP_YDISTANCE	アンカー・ポイントから、オブジェクトを Y軸方向に動かすピクセル数	int
OBJPROP_XSIZE	X軸方向のオブジェクト寸法（サイズ） （ピクセル数で表す幅）	int
OBJPROP_YSIZE	Y軸方向のオブジェクト寸法（サイズ） （ピクセル数で表す高さ）	int
OBJPROP_STATE	ボタンの状態（押された／離された）	bool

（3）グラフィカル・オブジェクトの種類（抜粋）

※「ObjectCreate()」で作り出せるオブジェクト郡（**40 個**）からの抜粋

ENUM_OBJECT　；翻訳しずらいものは、原文のまま（アメンボの不勉強が原因です）

ID（識別子）		解　説
OBJ_VLINE		垂直線
OBJ_HLINE		水平線
OBJ_TREND		トレンド・ライン
・・・・・・・・・・		
OBJ_RECTANGLE		四角
OBJ_TRIANGLE		三角
OBJ_ELLIPSE		楕円
・・・・・・・・・・		
OBJ_ARROW_SELL		売りサイン
OBJ_ARROW		矢印
OBJ_TEXT		テキスト
OBJ_LABEL		ラベル
OBJ_BUTTON		ボタン
・・・・・・・・・・		
OBJ_EVENT		The "Event" object corresponding to an event in the economic calendar
OBJ_RECTANGLE_LABEL		四角いラベル・オブジェクト、 ユーザーが任意のグラフィカル・インターフェースを 作成するために使う

以　上