○ＭＱＬ５；翻訳まとめ「OnChartEvent()の使い方、他（その１）」　翻訳のみ実施 2012.11.22

・アメンボです、
本稿の翻訳対象は「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。
①MQL5（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、上記のうち簡単な「①」の使用例を翻訳・解説しています。
（　カスタム・イベントの方は判りにくくて、理解に現在悪戦苦闘中　）

注意；　・本資料は、まだＭＴ５での動作・検証を行っていません、
・本編は近々の検証用資料として、英文資料を意訳しながら纏めたもの（メモ）です。
訳した資料がある程度たまったところで、ＭＴ５をダウンロードして確認していくつもりです。・・・すいません、まだＭＴ５は使ったことが無いのです！
（実機で未検証の内容ですので、誤訳があるかもしれません）
・以上の状況を理解されたうえで、本稿内容を参照ください。
○本稿を「(その１)」としたのは、基本内容のみを記述したので、別の機会に応用や実施例等を報告しようと考えているからです。

目次：

１．イベント「ハンドリング関数とトリガ」一覧（現状の理解）

※表１；現時点での理解範囲で、全体と一応解説済みのものを整理してみます。

		EA;ExpertAdviser		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	関数使用	OrderSend 関数内発行	ｲﾝﾃﾞｨｹｰﾀ 表示	ｽｸﾘﾌﾟﾄ 実行	解説
OnStart()	―	―	○	―	○	？改めて解説必要
OnInit()	開始	○	―	○	―	済
OnDeinit()	終了	○	―	○	―	済
OnTick()	ﾃｨｯｸ	○	○	―	―	？改めて解説必要
	ﾏﾙﾁｶﾚﾝｼｰ･ﾓｰﾄﾞ	○	○	―	―	**未＜別途＞**
OnTimer()	ﾀｲﾏｰ	○	○	―	―	済
OnTrade()	order・deal・position	○	？	―	―	済
OnTester()	ｽﾄﾗﾃｼﾞｰ・ﾃｽﾀｰ	○	―	―	―	済
OnBookEvent()	板(DOM)情報	○	○	―	―	済
OnChartEvent()	**ｸﾞﾗﾌｨｶﾙ・ｵﾌﾞｼﾞｪｸﾄ**	○	○	○	―	**本稿**
	ｶｽﾀﾑ･ｲﾍﾞﾝﾄ	○	○	○	―	**未＜別途＞**
OnCalculate()	ｲﾝﾃﾞｨｹｰﾀ表示計算	―	―	○	―	済；**半分残？**
	簡略ﾀｲﾌﾟ	―	―	○	―	**済？**

※DOM：Depth of Market　要するに「板情報」のこと

※「**青書**」部は、追加・修正した部分

※「OrderSend 関数内発行」とは、例えば、「OnTimer()」内で「OrderSend」発行が可能と言う意味で使いました。

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()が呼出され実行されるイベント）

・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＞

				返し値		
	ｲﾍﾞﾝﾄ（割込）発生	ID	概要	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	ｵﾌﾞｼﾞｪｸﾄがある 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

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＞

				返し値		
	ｲﾍﾞﾝﾄ（割込）発生	ID（識別子）	概要	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実施例

※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。

③上記のコード構成（例）；ポイントのみ記述

```
#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
 ・・・・・・・・・・・・・・・・
void OnTick()
{
        //通常は、ここがＥＡの本体コード
}
 ・・・・・・・・・・・・・・・・
 //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(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();
      }
   }
```

（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＞

				返し値		
	ｲﾍﾞﾝﾄ種類	ID（識別子）	概要	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実施例

※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。

③上記のコード構成（例）；ポイントのみ記述

```
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 ="";
        color BorderColor = Black;
        color ButtonColor = Aqua;
        int DistanceX = 30;
        int DistanceY = 30;
        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);
        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(0);
}
//
void OnDeinit(const int reason)
{
        //ボタン削除
        ObjectsDeleteAll(0);
}
//------------------------
void OnTick()
{
        //通常は、ここがＥＡの本体コード
}
・・・・・・・・・・・・・・・
//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,         //「CHARTEVENT_OBJECT_CLICK」が返される
                  const long& lparam,   //「Ｘ座標」が返される
                  const double& dparam, //「Ｙ座標」が返される
                  const string& sparam  //「オブジェクト名称」が返される
                  )
{
  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();
      }
    }
  }
・・・・・・・・・・
}
・・・・・・・・・・
```

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
   ...
   datetime      time30=0,       // 第 30 番アンカー・ポイント「時間」座標
   double        price30=0       // 第 30 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」
time30=0	[in]	第 30 番アンカー・ポイント「時間」座標、デフォルトは「0」
price30=0	[in]	第 30 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

※「ObjectCreate()」で作り出せるオブジェクト郡

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_TRENDBYANGLE		トレンド・ライン（角度）
OBJ_CHANNEL		Equidistant Channel
OBJ_STDDEVCHANNEL		標準偏差チャンネル
OBJ_REGRESSION		Linear Regression Channel
OBJ_PITCHFORK		Andrews’ Pitchfork
OBJ_GANNLINE		ギャン・ライン
OBJ_GANNFAN		ギャン・ファン（アングル）
OBJ_GANNGRID		ギャン・グリッド
OBJ_FIBO		フィボナッチ・リトレースメント

OBJ_FIBOTIMES		Fibonacci Time Zones
OBJ_FIBOFAN		フィボナッチ・ファン（アングル）
OBJ_FIBOARC		Fibonacci Arcs
OBJ_FIBOCHANNEL		フィボナッチ・チャンネル
OBJ_EXPANSION		Fibonacci Expansion
OBJ_ELLIOTWAVE5		エリオット推進波動
OBJ_ELLIOTWAVE3		Elliott Correction Wave
OBJ_RECTANGLE		四角
OBJ_TRIANGLE		三角
OBJ_ELLIPSE		楕円
OBJ_CYCLES		Cycle Lines
OBJ_ARROW_THUMB_UP		Thumbs Up
OBJ_ARROW_THUMB_DOWN		Thumbs Down
OBJ_ARROW_UP		上向き矢印
OBJ_ARROW_DOWN		下向き矢印
OBJ_ARROW_STOP		ストップ・サイン
OBJ_ARROW_CHECK		チェック・サイン
OBJ_ARROW_LEFT_PRICE		Left Price Label
OBJ_ARROW_RIGHT_PRICE		Right Price Label
OBJ_ARROW_BUY		買いサイン
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_CHART		チャート
OBJ_BITMAP		ビットマップ画像
OBJ_BITMAP_LABEL		ビットマップ・ラベル
OBJ_EDIT		編集
OBJ_ARROWED_LINE		Arrowed Line
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