

# **MaLa Event Plugin API v1.2**

August 28, 2007

## **1. Event Functions**

---

**Delphi:**        **Procedure MaLaGameSelected(g: TPEventGame); cdecl;**  
**C++:**         **extern "C" DLL void MaLaGameSelected(EventGame\* g)**

Provides an EventGame record/structure everytime a game is selected in MaLa.

---

**Delphi:**        **procedure MaLaEmulatorSelected(e: TPEventEmulator); cdecl;**  
**C++:**         **extern "C" DLL void MaLaEmulatorSelected(EventEmulator\* e)**

Provides an EventEmulator record/structur evrytime an emulator is selected in MaLa.

---

**Delphi:**        **procedure MaLaListSelected(gl: TPEventGamelist); cdecl;**  
**C++:**         **extern "C" DLL void MaLaListSelected(EventGameList\* gl)**

Provides an EventGameList record/structur everytime a gamelist is selected in MaLa.

---

**Delphi:**        **procedure MaLaOrientationSwitch(orientation: Integer); cdecl;**  
**C++:**         **extern "C" DLL void MaLaOrientationSwitch(int orientation)**

Provides the orientation in degrees (0, 90, 180, 270 counter-clockwise) everytime MaLa switches the orientation.

---

**Delphi:**        **procedure MaLaGameStart(); cdecl;**  
**C++:**         **extern "C" DLL void MaLaGameStart()**

Triggered when MaLa start a game.

---

**Delphi:**        **procedure MaLaGameQuit(); cdecl;**  
**C++:**         **extern "C" DLL void MaLaGameQuit()**

Triggered when MaLa quit a game.

---

**Delphi:**        **procedure MaLaScreensaverStart(); cdecl;**  
**C++:**         **extern "C" DLL void MaLaScreensaverStart()**

Triggered when the MaLa Screensaver starts.

---

**Delphi:**        **procedure MaLaScreensaverSwitch(filename: string); cdecl;**  
**C++:**         **extern "C" DLL void MaLaScreensaverSwitch(const char\* filename)**

Provides a filename with full path everytime the MaLa Screensaver displays a new file (image or video).

---

**Delphi:**        **procedure MaLaScreensaverStop(); cdecl;**  
**C++:**         **extern "C" DLL void MaLaScreensaverStop()**

Triggered when the MaLa Screensaver stops.

---

**Delphi:**        **procedure MaLaAttractModeStart(); cdecl;**  
**C++:**         **extern "C" DLL void MaLaAttractModeStart()**

Triggered when the MaLa AttractMode starts.

---

**Delphi:**        **procedure MaLaAttractModeStop(); cdecl;**  
**C++:**         **extern "C" DLL void MaLaAttractModeStop()**

Triggered when the MaLa AttractMode stops.

---

**Delphi:**        **procedure MaLaTreeUp(t: TPEventTree); cdecl;**  
**C++:**         **extern "C" DLL void MaLaTreeUp(EventTree\* t)**

Triggered when the user navigates one tree up in a tree navigation.

---

**Delphi:**        **procedure MaLaTreeDown(t: TPEventTree); cdecl;**  
**C++:**         **extern "C" DLL void MaLaTreeDown(EventTree\* t)**

Triggered when the user navigates one tree down in a tree navigation.

---

**Delphi:**        **procedure MaLaStart(); cdecl;**  
**C++:**         **extern "C" DLL void MaLaStart()**

Triggered when MaLa starts.

---

**Delphi:**        **procedure MaLaQuit(); cdecl;**  
**C++:**         **extern "C" DLL void MaLaQuit()**

Triggered when MaLa quits.

---

**Delphi:**      **function PluginName(): PChar; cdecl;**  
**C++:**        **extern "C" DLL const char\* PluginName()**

Provides the plugin name.

---

**Delphi:**      **function PluginCopyright(): PChar; cdecl;**  
**C++:**        **extern "C" DLL const char\* PluginCopyright()**

Provides the plugin copyright message.

---

Note for the C++ functions: DLL = \_\_declspec(dllexport)

## 2. Delphi: Record EventGame

---

type

TEventGame = record	
Description: Pchar;	: Name of the game
Rom: Pchar;	: Rom file of the game
CloneOf: Pchar;	: Master rom file if any
Manufacturer: Pchar;	: Manufacturer of the game
Genre: Pchar;	: Genre of the game (catver.ini)
Year: Integer;	: Year
InputButtons: Integer;	: Count of input buttons (mame.xml)
InputCoins: Integer;	: Count of coins (mame.xml)
InputPlayers: PChar;	: Count of players + alternating
InputControl: Pchar;	: Control type (mame.xml)
VideoScreen: Pchar;	: Screen type
VideoOrientation: Pchar;	: Screen orientation
VideoWidth: Integer;	: Width of screen
VideoHeight: Integer;	: Height of screen
DriverStatus: Pchar;	: Driver status
Controls: Pchar;	: Control type (controls.ini)
JoyUp: Pchar;	: Game function Joy Up (controls.ini)
JoyDown: Pchar;	: Game function Joy Down (controls.ini)
JoyLeft: Pchar;	: Game function Joy Left (controls.ini)
JoyRight: Pchar;	: Game function Joy Right (controls.ini)
Button1: Pchar;	: Game function Button 1 (controls.ini)
Button2: Pchar;	: Game function Button 2 (controls.ini)
Button3: Pchar;	: Game function Button 3 (controls.ini)
Button4: Pchar;	: Game function Button 4 (controls.ini)
Button5: Pchar;	: Game function Button 5 (controls.ini)
Button6: Pchar;	: Game function Button 6 (controls.ini)
Button7: Pchar;	: Game function Button 7 (controls.ini)
Button8: Pchar;	: Game function Button 8 (controls.ini)
Details: Pchar;	: Control details (controls.ini)
RomPath: Pchar;	: Path of rom file
Extension: Pchar;	: Extension of rom file
Played: Integer;	: Play counter
end;	

TPEventGame = ^TEventGame;

### **3. C++: Struct EventGame**

---

```
struct EventGame
{
    const char* Description;
    const char* Rom;
    const char* CloneOf;
    const char* Manufacturer;
    const char* Genre;
    int Year;
    int InputButtons;
    int InputCoins;
    const char* InputPlayers;
    const char* InputControl;
    const char* VideoScreen;
    const char* VideoOrientation;
    int VideoWidth;
    int VideoHeight;
    const char* DriverStatus;
    const char* Controls;
    const char* JoyUp;
    const char* JoyDown;
    const char* JoyLeft;
    const char* JoyRight;
    const char* Button1;
    const char* Button2;
    const char* Button3;
    const char* Button4;
    const char* Button5;
    const char* Button6;
    const char* Button7;
    const char* Button8;
    const char* Details;
    const char* RomPath;
    const char* Extension;
    int Played;
};
```

## **4. Delphi: Record EventEmulator**

---

type

```
TEventEmulator = record
  Name: Pchar;           : Name of the emulator
  Filename: Pchar;       : Executable of emulator, full path
  RomPath: Pchar;        : Path to the emulator rom files
  Extensions: Pchar;     : Extensions of the rom files (comma separated)
  Commandline: Pchar;    : Emulator command line
  SnapPath: Pchar;       : Path to the snap images
  MarqueePath: Pchar;    : Path to the marquee images
  CPanelPath: Pchar;     : Path to the control panel images
  VideoPath: Pchar;      : Path to the videos
  EncoderConfigFile: Pchar; : Default encoder file
  EncoderGameBased: Integer; : Game based encoder programming(0,1)
  EncoderConfigFilesPath: Pchar; : Path to the encoder config files
  EncoderConfigFileExtension: Pchar; : Extension of the encoder config files
end;
```

```
TPEventEmulator = ^TeventEmulator;
```

## **5. C++: Struct EventEmulator**

---

struct EventEmulator

```
{
  const char* Name;
  const char* Filename;
  const char* RomPath;
  const char* Extensions;
  const char* Commandline;
  const char* SnapPath;
  const char* MarqueePath;
  const char* CPanelPath;
  const char* VideoPath;
  const char* EncoderConfigFile;
  int EncoderGameBased;
  const char* EncoderConfigFilesPath;
  const char* EncoderConfigFileExtension;
};
```

## **6. Delphi: Record EventGamelist**

---

type

```
TEventGamelist = record
  Name: Pchar;           : Name of the Gamelist
  GameCount: Integer;    : Overall count of games
  FilterCount: Integer;  : Count of games when filter is applied
end;
```

```
TPEventGamelist = ^TEventGamelist;
```

## **7. C++: Struct EventGamelist**

---

```
struct EventGameList
{
  const char* Name;
  int GameCount;
  int FilterCount;
};
```



## **8. Delphi: Record EventTree**

---

```
type
  TEventTree = record
    Name: Pchar;           : Name/Title of the menu
    NodeCount: Integer;    : Count of menu nodes
    Menu: Integer;         : Is menu(1) or gamelist(0)
  end;

TPEventTree = ^TEventTree;
```

## **9. C++: Struct EventTree**

---

```
struct EventTree
{
  const char* Name;
  int NodeCount;
  int Menu;
};
```

## 10. Messages

---

It is also possible to send messages from a plugin or application to MaLa to trigger various action within MaLa. The following messages are available:

---

```
MALAMSG_STOPBGMUSIC = WM_USER + 10;
MALAMSG_STARTBGMUSIC = WM_USER + 11;
MALAMSG_NEXTSONG = WM_USER + 12;
MALAMSG_PREVIOUSSONG = WM_USER + 13;

MALAMSG_SWITCHORIENTATION = WM_USER + 20;
MALAMSG_FLIPORIENTATION = WM_USER + 21;

MALAMSG_STARTSCREENSAVER = WM_USER + 30;
MALAMSG_STOPSCREENSAVER = WM_USER + 31;

MALAMSG_ONEUP = WM_USER + 40;
MALAMSG_ONEDOWN = WM_USER + 41;
MALAMSG_XUP = WM_USER + 42;
MALAMSG_XDOWN = WM_USER + 43;
MALAMSG_LETTERUP = WM_USER + 44;
MALAMSG_LETTERDOWN = WM_USER + 45;

MALAMSG_NEXTGAMELIST = WM_USER + 50;
MALAMSG_PREVIOUSGAMELIST = WM_USER + 51;
MALAMSG_NEXTEMULATOR = WM_USER + 52;
MALAMSG_PREVIOUSSEMULATOR = WM_USER + 53;

MALAMSG_STARTGAME = WM_USER + 60;
MALAMSG_STARTRNDGAME = WM_USER + 61;
MALAMSG_QUITGAME = WM_USER + 62;

MALAMSG_EXIT = WM_USER + 70;

MALAMSG_SETLED = WM_USER + 80;
MALAMSG_ALLLEDON = WM_USER + 81;
MALAMSG_ALLLEDOFF = WM_USER + 82;
MALAMSG_RESETLEDSTATE = WM_USER + 83;
```

---

An example on how to send an orientation switch message to MaLa with Delphi and WINAPI functions:

```
SendMessage(FindWindow(nil,'MaLa'), MALAMSG_SWITCHORIENTATION, 0, 0);
or
PostMessage(FindWindow(nil,'MaLa'), MALAMSG_SWITCHORIENTATION, 0, 0);
```

Should be similar in C++.

To use the message **MALMSG\_SETLED** you need to send or post two parameter like in the following example:

```
PostMessage(FindWindow(nil,'MaLa'), MALMSG_SETLED, int ledNR, bool OnOff);
```

An example application written in Delphi is included.

## **11. General Information**

---

Please rename the compiled \*.dll file to \*.mplugin to get it recognized by MaLa.

Check also the include example plugin (EventLogger) to get an idea of how the plugin stuff is working.

## **12. License**

---

- ✓ The MaLa Event Plugin SDK and all associated texts, graphics, copyrights belong wholly to the copyright holder.
- ✓ The MaLa Event Plugin SDK is provided on an 'as is' basis.
- ✓ No warranties at all.
- ✓ Private non-commercial use only!