

魅族手机应用软件审核标准

(V1.2)

2010-05-14 发布

2010-05-14 实施

珠海市魅族科技有限公司 发布

更新列表

版本	更新日期	更新内容
1.0	2010.04.07	首次发布
1.1	2010.05.05	1. 增加 CAB 包应用程序名称审核; 2. 增加程序在任务列表中显示审核; 3. 增加睡眠模式审核; 4. 增加常亮锁屏处理; 5. 增加审核测试流程;
1.2	2010.05.14	1. 修改程序声音控制审核 (二 2.17); 2. 修改收费软件审核标准 (四 4.1);

目 录

一. 应用软件打包规则

1	基本信息	4
2	CAB 文件解析	5
2.1	安装文件信息	5
2.2	注册表信息	6

二. 应用软件评定标准

1	内容审核	7
2	测试审核	8
3	插件审核标准	10
4	收费软件审核标准	10
5	审核测试流程	11

三. 自测项目

1	静态检查	11
2	交互测试	13

四. 应用软件系统事件处理范例 15

五. 图标规范 37

六. MZDN 非常规软件备案申请表 40

魅族手机应用软件审核标准

一. 应用软件打包规则

1. 基本信息


软件包的名称	应用名称_版本号.cab（应用名称与桌面显示名称一致）
软件包的大小	XX KB(不得超过 100MB)
软件包的打包日期	XX 年 XX 月 XX 日
软件包的打包工具	XX, 如: WinCE CAB Manager
软件所属类别	<ul style="list-style-type: none"> ✧ 工具类- {新闻资讯、聊天通信、理财工具、办公软件、阅读相关、多媒体、生活交通、系统工具} ✧ 游戏类- {动作游戏、对战游戏、益智游戏、棋牌游戏、休闲娱乐} ✧ 其他类- {其它、主题风格、系统固件}
适应的固件版本	<ul style="list-style-type: none"> ✧ 无版本限制, 均可用 ✧ XX (请一一列举)
手机上的安装目录	<ul style="list-style-type: none"> ✧ 工具类: 统一放在“Disk\Programs\Tools\软件名称”, 全部文件统一放在该目录下; ✧ 游戏类: 统一放在“Disk\Programs\Games\软件名称”下, 全部文件统一放在该目录下; ✧ 其他类: 统一放在“Disk\Programs\Other\软件名称”下, 全部文件统一放在该目录下; ✧ 插件类: 统一放在“Disk\Programs\Widgets\软件名称”下, 全部文件统一放在该目录下; ✧ 若一个开发者开发多款软件时, 可置于(需开发者递交《MZDN 非常规软件备案申请表》待审核通过后方可实施): ✓ Disk\Programs\开发者名称\软件名称, 全部文件统一放在该目录下; ✓ Disk\Programs\公司名称\软件名称, 全部文件统一放在该


	<p>目录下；</p> <p>✓ Disk\Programs\ 软件名称，全部文件统一放在该目录下</p> <p>注： 1. Mstore 列表软件名称最大正常显示为 8 个中文字，建议不超过此限制；</p> <p>2. 特殊软件无法安装在 Disk 目录下，需填写《MZDN 非常规软件备案申请表》，申请安装在 Program Files 相应目录下；</p> <p>如： Program Files\Tools Program Files\Games Program Files\Other Program Files\Widgets</p>
桌面图标要求	<p>1. 尺寸：90*90 (实际有效像素为 86*86，具体标准请参考图标规范)</p> <p>2. 格式：仅限 PNG，且显示为圆角图标</p> <p>3. 位置：不允许指定位置</p>
网络连接方式	<p>✧ 无需网络连接</p> <p>✧ 采用 SDK 连接方式 {WLAN、GPRS (中国移动、中国联通、中国电信)、Activesync}</p>


2. CAB 文件解析

以开卷有益为例

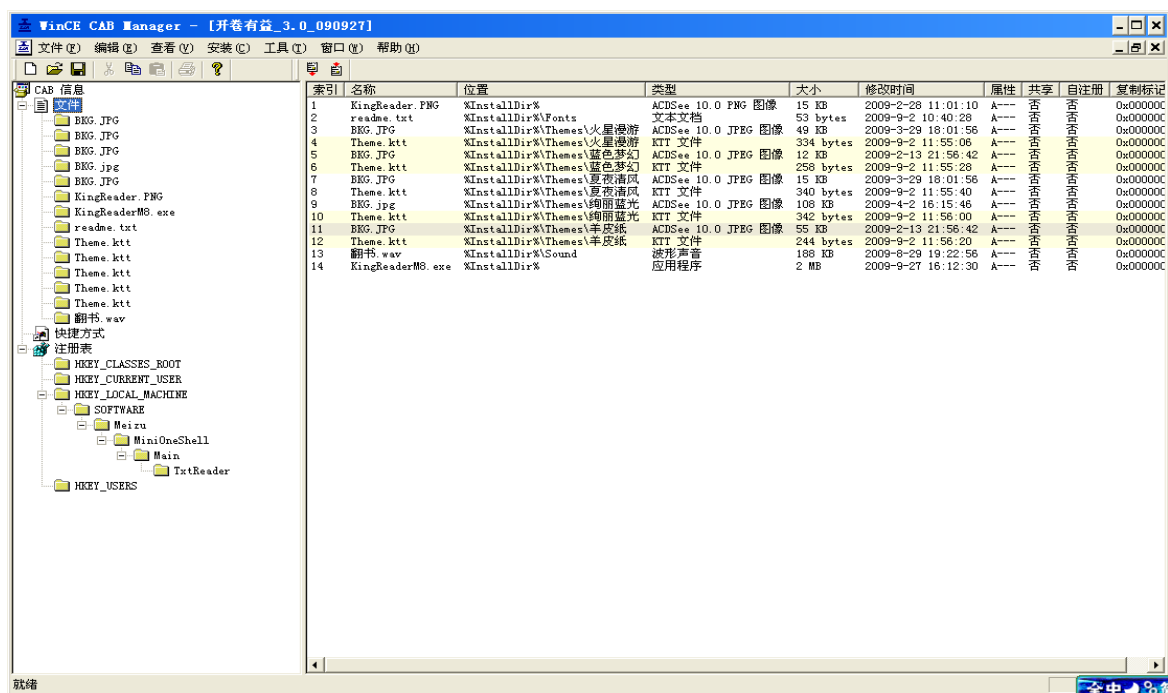
2.1 安装文件信息：

 BKG. JPG&Theme.ktt：相关的主题图片及文件

 KingReader.png：程序在桌面显示的图片（尺寸及格式要求：
90*90 仅限 PNG 格式, 圆角图形）

 KingReaderM8.exe：应用程序的执行文件 (命名规则参考基本信息中的第一项)

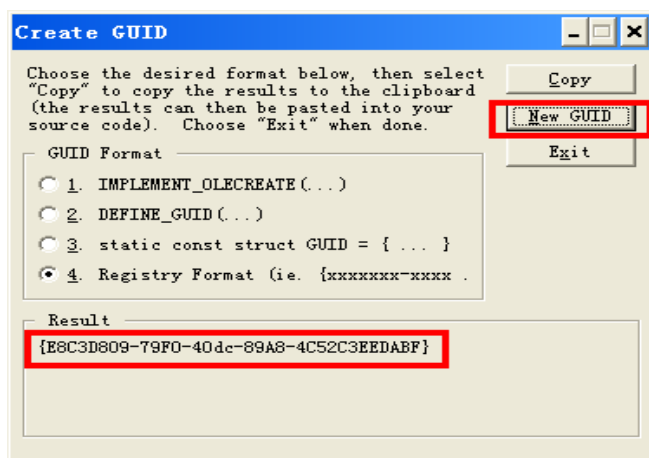
 翻书.Wav：声音文件



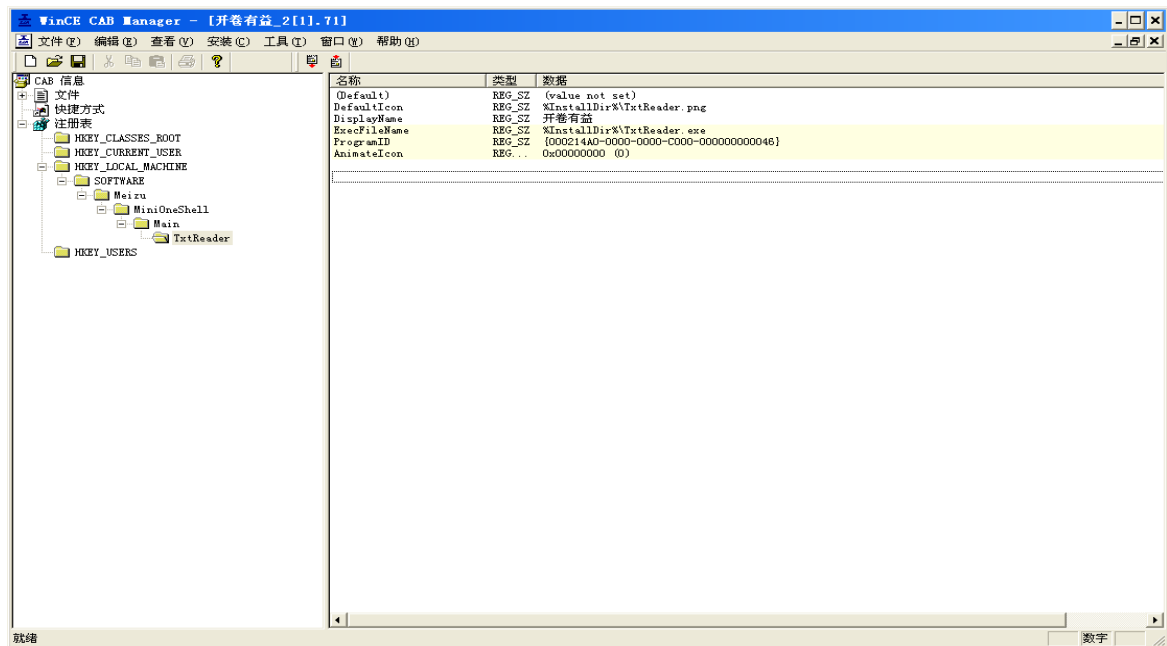
2.2 注册表信息:

- ✚ DefaultIcon: 程序在桌面显示的图标
- ✚ DisplayName: 桌面显示的图标名称
- ✚ ExecFileName: 程序的执行文件所在的路径
- ✚ ProgramID: 用来标识程序的唯一 ID 串

备注: 为了确保 ID 的唯一性, 关于 ProgramID 的生成, 建议开发者使用 GUID.exe 生成工具, 操作如图示:



- ✚ AnimateIcon: 动态图标的开关, 0 为关, 1 为开



二．应用软件评定标准

软件开发作者必须严格遵守《魅族开发者社区合作意向书》中
开发者所承诺的条款

1. 内容审核

1.1 文字敏感字符检查：检查内容包括：软件名称、发布者、关键字、应用程序说明、版本说明；

1.2 软件截图检查：检查上传截图是否与应用界面相符，以及软件截图是否包含一些广告、水印等非软件截图内容（网络应用可能包含一些网络内容、文本工具可能包含一些非法的文本等）；

1.3 关键字检查：如果有关键字，检查关键字是否与其应用程序说明及功能保持一致（因关键字会影响用户的搜索结果）；

1.4 侵犯版权检查：应用程序说明和版本说明是否存在明显的侵犯其他公司版本的行为；

1.5 产品定价检查：免费不做检查，如果收费，检查价格是否超出了其产品的定价范围；

1.6 CAB 包应用程序名称不得出现中文；

2. 测试审核

2.1 安装目录检查：检查安装目录是否符合打包规则；

2.2 软件包中一定要包含 ProgramID ，且确保其唯一性；

2.3 软件名称检查：程序安装后桌面显示的名称需与软件中心应用软件名称一致；

2.4 ICON 检查：如果有桌面图标，检查桌面图标是否与上传图标一致，桌面图标是否在正确位置；

2.5 软件运行时手机功能检查：

a. 在软件运行时，检查手机电话接通、短信接收等是否正常；

b. 所有软件按 M 键后要退出程序，不得进入后台工作（建议在下次运行程序时可记忆上一次的运行状态）；

c. 程序可以进入睡眠状态；

d. 在 Disk 下运行的程序，运行时连接 USB 后（U 盘模式）程序自动退出，回到桌面，不要弹出提示框；

e. 连接 USB（U 盘模式）时，再运行程序应弹出“该程序不能在 U 盘模式下运行，请先断开 USB 连接”。

2.6 软件语言检查：检查应用提供的版本是否支持相应的语言；

2.7 软件能力属性检查：检查软件所使用的手机能力是否与其选择的能力属性一致；

2.8 软件功能检查：检查软件是否正确实现了其应用程序说明和版本说明中的功能，如果有软件操作说明书，检查软件操作是否与说明书内容相符；

2.9 待测、待审核的软件，确保其在新 UI 固件上可以正常运行（先前已上架的软件需后期版本更新时修改）；

2.10 针对应用软件的开发建议开发者使用魅族官方公布的最新固件 SDK 软件开发工具；

2.11 不得包含政治、色情、邪教等违法内容；

2.12 禁止附带广告、寻求赞助、网络连接等其他商业信息；

2.13 程序只能单实例运行；

2.14 开启程序后，标题栏显示当前程序的相关信息，开发者需增加“应用程序当前窗口的 Text 文本”（5 秒自动切换系统时间）；

2.15 运行程序的界面不允许 180 度旋转, 只能左转或右转 90 度；

2.16 软件运行产生的所有文件规范到自己的安装目录下，不得在其他地方建立目录；

2.17 若程序有声音效果，需有声音开关键，默认音量不得对手机硬件造成伤害，程序运行时可使用手机音量键调节程序声音大小；

2.18 程序在任务列表显示的图标、名称须与桌面图标、名称一致；

2.19 程序不得阻止系统进入睡眠模式；

3. 插件审核标准

3.1 插件列表界面图标尺寸规范：250x108（**格式要求：仅限 PNG 格式**），不能超过这个标准范围；KingReader.png：程序在桌面显示的图片；

3.2 新添加以下两个注册表项：

"Width"=dword:00000004

"Height"=dword:00000002

以上项的值只对 Widget 有效，分别代表 Widget 的宽和高，单位为所占的格子数；

3.3 软件包中以 DLL 为插件主执行体，而不是 EXE。但其中也可以包含 EXE 文件；

3.4 运行插件程序后不能影响桌面的运行效率，不能弹出独占窗口影响用户的操作；

3.5 其他功能测试审核标准和应用软件标准一致；

4. 收费软件审核标准

4.1 所有上架 Mstore 的收费软件均须提供部分功能免费试用，免费与收费为同一个 CAB 包，具体制作标准请参考《软件中心收费软件 License 保护编程指南》；

4.2 收费软件验证失败统一提示语为：“授权文件校验失败，请重新下载安装”；

4.3 其他功能测试审核标准和应用软件标准一致；

5. 审核测试流程

5.1 每款软件均有 3 次正常审核测试机会，正常审核测试均会在 7 个工作日内完成；

5.2 若一款软件超过 3 次未通过审核测试，软件中心将优先审核测试其他软件，该软件的第四次审核测试期将会延长为 7 到 14 个工作日，以后每次未通过，下次审核测试期将继续延长 7 个工作日，以此类推；

5.3 若软件未修改上次审核测试所检测出 BUG 再次递交，软件将直接退回，并延长下次审核测试期为 7 到 14 个工作日，以后每次未通过，下次审核测试期将继续延长 7 个工作日，以此类推；

三、自测项目

1. 静态检查

软件包的名称	应用名称_版本号.cab（应用名与桌面显示的名称一致）
软件包的大小	小于 100MB
软件抓图	1. 提供一张软件运行后主界面的图片； 2. 与说明书内的图片相符；
软件说明书要求	1. 参照范例，限 DOC、TXT、PDF 格式； 2. 功能简介必须与实际功能相符；
适应的固件版本	✧ 无版本限制，均可用 ✧ XXX（请一一列举）

手机上的安装目录	<ul style="list-style-type: none"> ✧ 工具类：统一放在“Disk\Programs\Tools\软件名称”，全部文件统一放在该目录下； ✧ 游戏类：统一放在“Disk\Programs\Games\软件名称”，全部文件统一放在该目录下； ✧ 其他类：统一放在“Disk\Programs\Other\软件名称”，全部文件统一放在该目录下； ✧ 插件类：统一放在“Disk\Programs\Widgets\软件名称”下，全部文件统一放在该目录下； ✧ 若一个开发者开发多款软件时，可置于（需开发者递交《MZDN 非常规软件备案申请表》待审核通过后方可实施）： <ul style="list-style-type: none"> ✓ Disk\Programs\开发者名称\软件名称，全部文件统一放在该目录下； ✓ Disk\Programs\公司名称\软件名称，全部文件统一放在该目录下； ✓ Disk\Programs\软件名称，全部文件统一放在该目录下； <p>注：软件名称与桌面显示名称必须一致，避免出现重名情况</p>
ICON 尺寸	90*90 (实际有效像素为 86*86，具体标准请参考图标规范)
ICON 格式	PNG 格式
桌面图标位置	不允许指定位置
Program ID	确保其唯一性
网络连接方式	<ul style="list-style-type: none"> ✧ 无需网络连接 ✧ 采用 SDK 连接方式 {WLAN、GPRS、Activesync}

2. 交互测试

(若系统未弹出提示框则不处理)

2.1 来电

要求：来电界面显示正常，可接听、忽略来电；在使用耳机时，按一次线控按钮：接听电话；通话时按一次线控按钮：结束通话；长按线控按钮：拒听来电；挂断电话后是否出现游戏声音外放情况；

2.2 新短信、彩信

要求：可正常弹出提示框，分为“预览”或“忽略”新信息内容两种情况；点击“预览”可直接进入信息界面并查看信息内容，点击“忽略”可关闭提示信息框继续运行程序；

2.3 Activesync、U 盘模式

要求：同步模式下可对手机进行读写；在 Disk 下运行的程序，连接 USB 后(u 盘模式)程序自动退出. 回到桌面, 不要弹出提示框；

2.4 相关按键操作

要求：单击 M 键结束程序；按音量键可调节音量大小；按音乐键可调用音乐程序等；

2.5 满内存时操作

要求：在对文件进行储存操作时，可弹出相应提示“磁盘空间不足，请清理多余文件”；

2.6 锁屏

要求：可按设置的时间进入锁屏，解锁后直接返回到上一个界面；

2.7 关机或者拔电池

要求：选择关机或拔电池时能立即终止程序，重启手机后程序能正常运行；

2.8 低电提示

要求：可弹出提示框，不影响软件正常操作；电池电量显示正常；

2.9 插拔耳机

要求：声道可正常切换；

2.10 插拔充电器

要求：充电程序正常且插拔充电器时充电图标显示正确；

2.11 闹钟、日历提醒

要求：提示框弹出，用户可推迟、查看或忽略提醒，界面显示正常；

2.12 蓝牙配对及接收文件请求

要求：弹出请求框后可对其做相应处理，接受或忽略请求；界面显示均正常；

2.13 程序开启、退出、页面切换要求

要求：必须有动画过渡效果，可设置为与系统一致的动画效果；

四、应用软件系统事件处理范例

1. U 盘中断事件处理;

- ✧ 由于目前应用软件都安装在 Disk 目录下,当软件运行后再连接 PC USB,手机端的 Disk 内容不可见,在 Disk 下运行的程序,连接 USB 后 (u 盘模式) 程序自动退出. 回到桌面, 不要弹出提示框;

- ✧ 程序参考: U 盘的相关 API 在 UsbNotifyApi.h 头文件中

U 盘状态判断: INT GetUsbConnectType (void)

获取 USB 当前的连接类型

返回:

USB_MASSSTORAGE_ATTACH - U 盘 (Mass Storage) 模式;
 USB_ACTIVESYNC_ATTACH -同步 (ActiveSync) 模式;
 USB_FUNCTION_DETACH - USB 已断开;
 USB_FUNCTION_ERROR -错误;

U 盘状态改变通知: INT RegisterUsbNotifyMsg (void)

向系统中注册一个窗口通知消息, 并返回该消息的值 Note:当 USB 的状态发生变化时,系统会向所用应用程序的主窗口广播一个通知消息. 通常, 在应用程序初始化的过程中调用 该函数来获得系统广播的消息值, 应用程序把该函数的返回值保存在一个变量中, 然后在窗口过程处理函数中处理该消息。

广播通知消息约定如下 :

Message ID :本函数的返回值. wParam :事件类型
 -USB_MASSSTORAGE_ATTACH/USB_ACTIVESYNC_ATTACH/USB_FUNCTION_DETACH

返回:

: 0 -失败, MessageID = 0xc000-0xffff -成功

Code example:

```
INT g_iUsbNotifyMsg = 0;
BOOL CTestWin::OnInitDialog()
{
    g_iUsbNotifyMsg = RegisterUsbNotifyMsg();
    .....
}
.....

LRESULT CTestWin::MzDefWndProc(UINT message, WPARAM
wParam, LPARAM lParam)
{
    if(message == g_iUsbNotifyMsg)
    {
        INT iEventType = (INT)wParam;
        //处理该事件
        .....
    }
    //其他消息处理
    .....
}
```

- ✧ 双击 M 键可直接调用手机中的任务管理器，点触选择的程序可进入，点触 X 关闭运行的程序，还会涉及到弹出效果的问题，不得出现返回其他屏幕的情况；

✧ 程序参考：

暂无

2. 单击 M 键返回桌面处理；

- ✧ 单击 M 键可返回桌面并结束应用软件进程；

✧ 程序参考：

M 键通知消息的机制：

当单击 M 键时，系统会向“已经注册此通知的应用程序窗

口”和“Z-Order 处于桌面之上的应用程序的主窗口”发送 M 键通知消息。

MZFC 应用程序：

CMzWnd::MzDefWndProc() 会注册并处理此消息。并且提供了以下虚成员函数以让您重载自定义处理 M 键通知：

```
void CMzWnd::SetShellHomekeyReturnValue ( int nValue )
```

设置当窗口收到 Shell Home key 消息时的返回值

普通 WIN32 应用程序（即非 MZFC 窗口程序）：

应用程序的主窗口可以使用如下 API 获取 M 键通知消息：

```
UINT g_nWmShellHomeKey =  
RegisterWindowMessage(WM_MINIONE_SHELL);
```

然后在窗口消息处理函数中添加处理代码。

还可以通过以下 API 来给任意窗口注册接收 M 键的通知消息：

```
BOOL RegisterShellMessage(HWND hWnd, UINT uMsg);
```

3. 来电响铃处理；

✧ 运行应用的过程中来电，界面需正常弹出且不得影响通话质量；

✧ 程序参考：

通过调用 RegisterWindowMessage(pstrMsgName) 即可获得其消息值。

pstrMsgName 的取值如下：（在 MzCommon.h 头文件中）

来电：WM_CALL_MESSAGE

这些消息的参数定义请参考 MzCommon.h 头文件。

4. 来短信处理;

- ✧ 来短信时可正常弹出提示框，分为“预览”或“忽略”新短信内容两种情况;点击“预览”可直接进入信息界面并查看信息内容，点击“忽略”可关闭提示信息框继续运行程序;

- ✧ 程序参考:

通过调用 RegisterWindowMessage(pstrMsgName)即可获得其消息值。

pstrMsgName 的取值如下: (在 MzCommon.h 头文件中)

短信: WM_SMS_MESSAGE

这些消息的参数定义请参考 MzCommon.h 头文件。

5. 来彩信处理;

- ✧ 来彩信跟短信情况类似，也分为“预览”或“忽略”新彩信两种情况;点击“预览”可直接进入信息界面并查看信息内容，点击“忽略”可关闭提示信息框继续运行程序;

- ✧ 程序参考:

通过调用 RegisterWindowMessage(pstrMsgName)即可获得其消息值。

pstrMsgName 的取值如下: (在 MzCommon.h 头文件中)

彩信: WM_MMS_MESSAGE

这些消息的参数定义请参考 MzCommon.h 头文件。

6. 单击电源键锁屏处理;

✧ 短按电源键锁屏，解锁后需直接返回到上一个界面;

处理方法:

解锁后均会显示上锁前的当前应用程序界面。

至于应用程序如果想获取解锁通知，可以通过以下 API:

```
BOOL RegisterShellMessage(HWND hWnd, UINT uMsg);
```

uMsg 取以下值:

```
#define WM_MZSH_ENTRY_LOCKPHONE    0x00000004    //进入锁机界面
#define WM_MZSH_LEAVE_LOCKPHONE    0x00000008    //离开锁机界面
```

7. 插耳机玩游戏，来电处理;

✧ 来电时按一次线控按钮：接听电话；通话时按一次线控按钮：

结束通话；长按线控按钮：拒听来电；以及关于挂断电话后

游戏声音外放的情况处理；

✧ 程序参考:

通过调用 RegisterWindowMessage(pstrMsgName) 即可获得其消息值。

pstrMsgName 的取值如下：（在 MzCommon.h 头文件中）

来电：WM_CALL_MESSAGE

这些消息的参数定义请参考 MzCommon.h 头文件。

8. 红外消息如何获取;

✧ 运行过程中可弹出提示框，并进行相应的操作；

✧ 程序参考：

红外感应器的示例：（摘自 Meizu M8 PlatformApi 帮助文档）

红外相关 API 声明在<IRSensorApi.h>头文件中

红外感应

此示例代码展示如何使用红外感应（IRSensor）。

将障碍物放置于手机听筒上方时，背景灯将会熄灭，三秒钟后自动亮起

启用红外感应设备：

```
//启用设备
```

```
SetIRSensorState(true);
```

```
//注册红外感应消息
```

```
RegisterIRSensorNotify(m_hWnd, MZ_IRSENSORID);
```

获得红外感应强度等级：

```
IRSensorLevel = GetIRSensorLevel();
```

完整示例

```

/*****
*****/
/*
* Copyright (C) Meizu Technology Corporation Zhuhai China
* All rights reserved.
*中国珠海,魅族科技有限公司,版权所有.
*
* Author: ZYK
* Create on: 2009-07-03
*/
/*****
*****/

```

//请按照以步骤运行此实例代码：

```
//首先,打开 VS2005/2008 创建一个 Win 32 智能设备项目

//在项目向导中选择 M8SDK,并勾选空项目

//然后,在项目中新建一个 cpp 文件,将此处代码拷贝到 cpp 文件中

//最后,按照 M8SDK 的帮助文档,配置项目属性

//现在,可以运行此程序了

//包含 MZFC 库的头文件
#include <mzfc_inc.h>
#include <IRSensorApi.h>
#include <BackLightApi.h>

//此代码演示了:

// 创建和初始化应用程序

// 创建和初始化窗体

// 红外探测 IRSensorAPI 的使用

// 背景光 BackLightAPI 的使用

// 将障碍物放置于听筒上方时会关闭背景光

#define MZ_IDC_IRSENSORBTN 101
#define MZ_IRSENSORID      102
#define MZ_IDC_EXITBTN    103

//从 CMzWndEx 派生的主窗口类
class CSample1MainWnd: public CMzWndEx
{
    MZ_DECLARE_DYNAMIC(CSample1MainWnd);
public:
    UiButton m_IRSensorBtn;
    UiButton m_ExitBtn;
protected:

    //窗口的初始化

    virtual BOOL OnInitDialog()
    {

        //必须先调用基类的初始化
```

```

        if (!CMzWndEx::OnInitDialog())
        {
            return FALSE;
        }
        SetIRSensorState(true);
        RegisterIRSensorNotify(m_hWnd, MZ_IRSENSORID);
        m_IRSensorBtn.SetPos(100, 150, 280, 100);
        m_IRSensorBtn.SetID(MZ_IDC_IRSENSORBTN);

        m_IRSensorBtn.SetText(L"关闭红外感应");

        AddUiWin(&m_IRSensorBtn);
        m_ExitBtn.SetID(MZ_IDC_EXITBTN);
        m_ExitBtn.SetPos(100, 350, 280, 100);

        m_ExitBtn.SetText(L"退出");

        AddUiWin(&m_ExitBtn);
        return TRUE;
    }

    //重载命令消息的处理函数
    virtual void OnMzCommand(WPARAM wParam, LPARAM lParam)
    {
        UINT_PTR id = LOWORD(wParam);
        switch(id)
        {
            case MZ_IDC_EXITBTN:
            {
                if(1 == MzMessageBoxEx(m_hWnd, L"You have pressed
Exit button, Really want exit?", L"Exit", MB_YESNO,
false))
                    PostQuitMessage(0);
            }
            break;
            case MZ_IDC_IRSENSORBTN:
            {
                if (wcscmp(L"关闭红外感应",
m_IRSensorBtn.GetText()) == 0)
                {
                    m_IRSensorBtn.SetText(L"打开红外感应");
                    m_IRSensorBtn.Invalidate();
                    m_IRSensorBtn.Update();
                }
            }
        }
    }

```

```

        if (!UnRegisterIRSensorNotify(m_hWnd))
        {
            MzMessageBoxEx(m_hWnd, L" 注销红外感应消息失败

", L"警告");
        }

        if (!SetIRSensorState(false))
        {
            //MzMessageBoxEx(m_hWnd, L" 关闭红外感应失败 ", L"

警告");
        }
    }
}

else if ( wcscmp(L" 打 开 红 外 感 应 ",
m_IRSensorBtn.GetText()) == 0)
{
    m_IRSensorBtn.SetText(L"关闭红外感应");
    m_IRSensorBtn.Invalidate();
    m_IRSensorBtn.Update();
    if (!SetIRSensorState(true))
    {
        MzMessageBoxEx(m_hWnd, L" 打开红外感应失败 ", L" 警告

");
    }
    if (!RegisterIRSensorNotify(m_hWnd,
MZ_IRSENSORID))
    {
        MzMessageBoxEx(m_hWnd,
L"RegisterIRSensorNotify false", L"alert");
    }
    if (!SetIRSensorLevel(LEVEL_HIGH))
    {
        MzMessageBoxEx(m_hWnd,      L"SetIRSensorLevel
false", L"alert");
    }
    int IRSensorLevel ;
    IRSensorLevel = GetIRSensorLevel();
    if (IRSensorLevel == LEVEL_HIGH)

```

```

        {
            MzMessageBoxEx(m_hWnd, L"红外探测等级：高", L"消
息");
        }
        else if (IRSensorLevel == LEVEL_MEDIUM)
        {
            MzMessageBoxEx(m_hWnd, L"红外探测等级：中", L"消
息");
        }
        else if (IRSensorLevel == LEVEL_LOW)
        {
            MzMessageBoxEx(m_hWnd, L"红外探测等级：低", L"消
息");
        }
    }
    break;
}
}

LRESULT MzDefWndProc(UINT message, WPARAM wParam,
LPARAM lParam)
{
    switch(message)
    {
        case MZ_IRSENSORID:
        {
            if (lParam)
            {
                //关闭背景光
                SetBackLightState(false);
            }
            else
            {
                //打开背景光
                SetBackLightState(true);
            }
        }
    }
}

```



```

        }
        break;
    default:
    {
        return CMzWndEx::MzDefWndProc(message, wParam,
lParam);
    }
    break;
}
return 0;
}
};
MZ_IMPLEMENT_DYNAMIC(CSample1MainWnd)

//从 CMzApp 派生的应用程序类
class CSample1App: public CMzApp
{
public:

    //应用程序的主窗口
    CSample1MainWnd m_MainWnd;

    //应用程序的初始化
    virtual BOOL Init()
    {
        //初始化 COM 组件
        CoInitializeEx(0, COINIT_MULTITHREADED);

        //创建主窗口
        RECT rcWork = MzGetWorkArea();
m_MainWnd.Create(rcWork.left,rcWork.top,RECT_WIDTH(rc
Work),RECT_HEIGHT(rcWork), 0, 0, 0);
        m_MainWnd.Show();

        //成功则返回 TRUE
        return TRUE;
    }
};

//全局的应用程序对象
CSample1App theApp;

```

9. 重力感应消息如何获取；

✧ 对于支持重力感应的软件，旋转手机时屏幕可作相应变化；

✧ 程序参考：

重力感应的示例：（摘自 Meizu M8 PlatformApi 帮助文档）

重力感应相关 API 声明在<acc_api.h>头文件中

重力感应(01_AccApi)

重力感应

此示例代码展示如何使用重力感应（AccApi）。

通过改变手机的位置，屏幕中的两个按钮控件分别在 X 轴，Y 轴上运动

开启 Acc 设备，获得 XYZ 轴加速度值：

```
signed char m_XAxis ;
signed char m_YAxis;
signed char m_ZAxis;

//开启 acc 功能
MzAccOpen();

//获取 XYZ 轴加速度值 1 == 18mg, 56 == 1g
MzAccGetXYZ(&m_XAxis, &m_YAxis, &m_ZAxis);
处理得到的加速度值：

//设置定时器每隔一段时间获取一次加速度值
SetTimer(m_hWnd, 10, 10, NULL);

//分别说的 X,Y 轴的加速度值，再做出计算
virtual void OnTimer(UINT_PTR nIDEvent)
{
    switch(nIDEvent)
    {
        case 10:
```

```

{
    //获得 X 轴加速度
    MzAccGetX(&m_XAxis);
    //获得 Y 轴加速度
    MzAccGetY(&m_YAxis);

    //计算控件的位置
    m_XPos = m_XPos m_XSpeed m_XAxis;
    if (m_XPos<0)
    {
        m_XPos = 0;
    }

    if (m_XPos>GetWidth() - m_XButton.GetWidth())
    {
        m_XPos = GetWidth() - m_XButton.GetWidth();
        SetTimer(m_hWnd, 11, 10, NULL);
    }

    m_YPos = m_YPos m_YSpeed m_YAxis;
    if (m_YPos<0)
    {
        m_YPos = 0;
    }

    if (m_YPos>600)
    {
        m_YPos = 600;
    }

    //重新设定控件的位置
    m_XButton.SetPos(m_XPos, 300, 150, 100);
    m_YButton.SetPos(200, m_YPos, 150, 100);
    Invalidate();
}
break;
default:
{
}
break;
}
}

```

完整示例

```

/*****
*****/
/*
* Copyright (C) Meizu Technology Corporation Zhuhai China
* All rights reserved.
*中国珠海, 魅族科技有限公司, 版权所有.
*
* Author:    ZYK
* Create on: 2009-07-03
*/
/*****
*****/

```

//请按照以步骤运行此实例代码:
 //首先, 打开 VS2005/2008 创建一个 Win 32 智能设备项目
 //在项目向导中选择 M8SDK, 并勾选空项目
 //然后, 在项目中新建一个 cpp 文件, 将此处代码拷贝到 cpp 文件中
 //最后, 按照 M8SDK 的帮助文档, 配置项目属性
 //现在, 可以运行此程序了

```

//包含 MZFC 库的头文件
#include <mzfc_inc.h>
#include <acc_api.h>
#include <MotorVibrate.h>
//此代码演示了:
// 创建和初始化应用程序
// 创建和初始化窗体
// 按钮控件的使用及其命令消息的处理
// acc_api 的使用

```

```

//按钮控件的 ID
#define MZ_IDC_EXITBTN 101

//从 CMzWndEx 派生的主窗口类
class CSample1MainWnd: public CMzWndEx
{
    MZ_DECLARE_DYNAMIC(CSample1MainWnd);
public:
    //窗口中的按钮控件
    UIButton m_XButton;

```

```

UIButton m_YButton;
UIButton m_ExitBtn;

signed char m_XAxis ;
signed char m_YAxis;
signed char m_ZAxis;
int m_XPos;
int m_XSpeed;

int m_YPos;
int m_YSpeed;
protected:
//窗口的初始化
virtual BOOL OnInitDialog()
{
    //必须先调用基类的初始化
    if (!CMzWndEx::OnInitDialog())
    {
        return FALSE;
    }

    //开启 acc 功能
    MzAccOpen();

    //获取 XYZ 轴加速度值 1 == 18mg, 56 == 1g
    MzAccGetXYZ(&m_XAxis, &m_YAxis, &m_ZAxis);

    SetTimer(m_hWnd, 10, 10, NULL);

    //设置初始的速度和位置
    m_XPos = 200;
    m_XSpeed = 0;

    m_YPos = 280;
    m_YSpeed = 0;

    m_XButton.SetPos(m_XPos, 300, 150, 100);
    AddUiWin(&m_XButton);

    m_YButton.SetPos(200, m_YPos, 150, 100);
    AddUiWin(&m_YButton);

    m_ExitBtn.SetID(MZ_IDC_EXITBTN);
    m_ExitBtn.SetPos(100, 500, 280, 100);

```

```

        m_ExitBtn.SetText(L"退出");

        AddUiWin(&m_ExitBtn);

        return TRUE;
    }

    //重载命令消息的处理函数
    virtual void OnMzCommand(WPARAM wParam, LPARAM lParam)
    {
        UINT_PTR id = LOWORD(wParam);
        switch(id)
        {
        case MZ_IDC_EXITBTN:
        {
            if(1 == MzMessageBoxEx(m_hWnd, L"You have pressed Exit button, Really want exit?", L"Exit", MB_YESNO, false))
                PostQuitMessage(0);
        }
        break;
        }
    }

    virtual void OnTimer(UINT_PTR nIDEvent)
    {
        switch(nIDEvent)
        {
        case 10:
        {
            //获得 X 轴加速度
            MzAccGetX(&m_XAxis);
            //获得 Y 轴加速度
            MzAccGetY(&m_YAxis);

            //计算控件的位置
            m_XPos = m_XPos + m_XSpeed * m_XAxis;
            if (m_XPos < 0)
            {
                m_XPos = 0;
            }

            if (m_XPos > GetWidth() - m_XButton.GetWidth())
            {
                m_XPos = GetWidth() - m_XButton.GetWidth();
            }
        }
        }
    }

```

```

        SetTimer(m_hWnd, 11, 10, NULL);
    }

    m_YPos = m_YPos m_YSpeed m_YAxis;
    if (m_YPos<0)
    {
        m_YPos = 0;
    }

    if (m_YPos>600)
    {
        m_YPos = 600;
    }

    //重新设定控件的位置
    m_XButton.SetPos(m_XPos, 300, 150, 100);
    m_YButton.SetPos(200, m_YPos, 150, 100);
    Invalidate();
}
break;
default:
{
}
break;
}
}
};

```

```

MZ_IMPLEMENT_DYNAMIC(CSample1MainWnd)

```

```

//从 CMzApp 派生的应用程序类
class CSample1App: public CMzApp
{
public:
    //应用程序的主窗口
    CSample1MainWnd m_MainWnd;

    //应用程序的初始化
    virtual BOOL Init()
    {
        //初始化 COM 组件
        CoInitializeEx(0, COINIT_MULTITHREADED);

        //创建主窗口
    }
}

```

```
RECT rcWork = MzGetWorkArea();

m_MainWnd.Create(rcWork.left, rcWork.top, RECT_WIDTH(rcWork), RECT_
HEIGHT(rcWork), 0, 0, 0);
m_MainWnd.Show();

//成功则返回 TRUE
return TRUE;
}
};

//全局的应用程序对象
CSample1App theApp;
```

10. 应用软件横屏时如何处理系统弹出框；

- ✧ 确保屏幕在切换时显示正常，用户体验感问题；**例如：在开启麻将或斗地主一些横屏显示的游戏时，界面有可能出现断截，显示异常的情况**
- ✧ 程序参考：暂无

11. 如何调用系统输入法键盘；

- ✧ 程序参考：

调用以下 M8SDK 中的 API：

```
/**
 * @brief 打开输入面板
 * @param dwMode 输入法打开模式，参见 IM_SIP_MODE。
 * @param dwSipOffset 输入法面板底部偏移值，xffffffff 代表系统默认，即竖屏为，横屏为。
 */
RECT MZFC_API MzOpenSip(DWORD dwMode = IM_SIP_MODE_KEEP, DWORD
dwSipOffset = 0xffffffff);
```

dwMode 的取值：

//!切换输入法键盘输入方式

MZFC_API enum IM_SIP_MODE

```
{
    IM_SIP_MODE_KEEP = 0,                //!<维持输入法现有模式
    IM_SIP_MODE_GEL_PY ,                  //!<切换为拼音输入法
    IM_SIP_MODE_GEL_LETTER,               //!<切换标准的英文键盘输入
    IM_SIP_MODE_MAIL_LETTER,              //!<切换为邮件键盘英文方式
    IM_SIP_MODE_WEB_LETTER,               //!<切换为网页键盘英文方式
    IM_SIP_MODE_ADDRESSEE_123,            //!<切换为数字键盘方式
    IM_SIP_MODE_ADDRESSEE_LETTER,         //!<切换为联系人英文键盘方式
    IM_SIP_MODE_SEARCH_LETTER,            //!<切换为搜索英文键盘方式
    IM_SIP_MODE_GEL_ENGLISH,              //!<切换智能英文键盘输入
    IM_SIP_MODE_DIGIT,                   //!<数字键盘
    IM_SIP_MODE_DIGITEX,                  //!<扩展数字键盘
    IM_SIP_MODE_NONE = 0x7fffff          //!<无输入法
};
```

12. 闹钟、日历提醒;

✧ 提示框弹出后，用户可推迟、查看或忽略提醒，界面显示

正常；例如：在运行游戏时闹钟或日历提醒无法弹出

✧ 程序参考：暂无

13. 插拔充电器;

✧ 充电程序正常，且插拔充电器时充电图标显示正确；例如：

在开启某一个游戏时，插拔充电器，充电图标显示不正确

✧ 程序参考：暂无

14. 音量键、音乐键调用;

- ✧ 按音量键可调节音量大小；按音乐键可调用音乐程序，界面显示正常；例如：在开启某一个游戏后按音量键，音量图标不显示，且无法调节游戏音量
- ✧ 程序参考：暂无

15. 关机或者拔电池；

- ✧ 选择关机或拔电池时能立即终止程序，重启手机后程序能正常运行；例如：在线听歌时关机，音乐不能立即停止；把电池重启手机后，无法再次开启游戏等
- ✧ 程序参考：暂无

16. 蓝牙配对及接收文件请求；

- ✧ 弹出请求框后可对其做相应处理，接受或忽略请求；界面显示均正常；
- ✧ 程序参考：
通过调用 RegisterWindowMessage(pstrMsgName) 即可获得其消息值。
pstrMsgName 的取值如下：（在 MzCommon.h 头文件中）
蓝牙：WM_BT_MESSAGE、WM_BT_IF_RECEIVE
这些消息的参数定义请参考 MzCommon.h 头文件。

17. 满内存情况处理；

- ✧ 对于一些需要进行文件储存的应用软件，需增加满内存时的处理信息，在储存文件时应弹出“磁盘空间不足，请清理多余文件”；例如：手机 DISK 内存满时，Flvshow 还能继续下载，

没有内存满的提示

✧ 程序参考：暂无

18. 开启程序时的动画效果应该怎么设置；

程序启动的放大动画：

```
if ( MzGetParam(MZGP_APP_START_ANIMATION)==TRUE )
    m_MainWnd.AnimateWindow( MZ_ANIMTYPE_ZOOM_IN , true);
m_MainWnd.Show();
```

19. 如何使应用软件在运行时不会自动锁屏；

✧ 对于一些游戏类的应用软件，可加入不会自动锁屏的设置，
便于用户操作；

✧ 程序参考：

可以使用以下 API：

```
#include <ShellNotifyMsg.h>
/**
 * @brief 使屏幕常亮(不自动黑屏进锁机状态)，使用完之后，一定要
调用 SetScreenAutoOff();
 * 该函数不能嵌套调用
 * @param hWnd [IN]，调用该函数的窗口句柄
 * @return TRUE, 成功; FALSE, 失败
 * @see SetScreenAutoOff()
 */
BOOL SetScreenAlwaysOn(HWND hWnd);

/**
 * @brief 使屏幕自动变暗，解除屏幕常亮状态
 * @return TRUE, 成功; FALSE, 失败
 * @see SetScreenAlwaysOn()
 */
BOOL SetScreenAutoOff();
```

20. 屏幕常亮锁屏

```
// 相关头文件
#include <ShellNotifyMsg.h>
#include <pm.h>

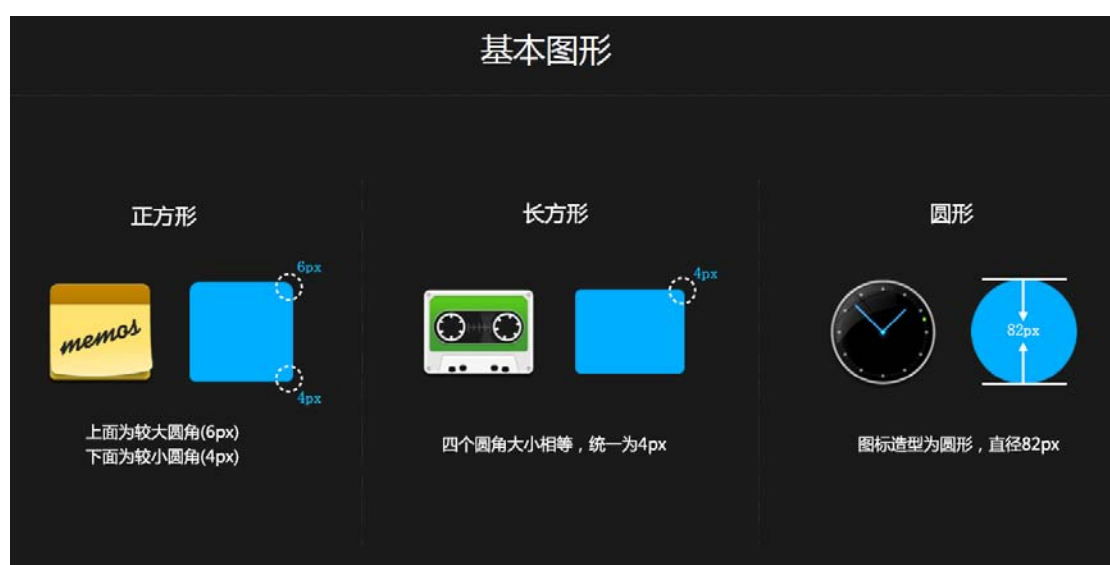
// 注册硬按键消息
RegisterShellMessage(m_hWnd, WM_MZSH_ALL_KEY_EVENT);

// 处理电源按键消息，实现锁屏
virtual LRESULT MzDefWndProc(UINT message, WPARAM wParam, LPARAM lParam)
{
    if (message == GetShellNotifyMsg_AllKeyEvent())
    {
        if (wParam == WPARAM_KEY_EVENT_CLICK_POWER)
        {
            // 先解除屏幕常亮状态
            SetScreenAutoOff();

            // 使屏幕黑屏
            ::SetSystemPowerState(NULL, POWER_STATE_IDLE,
POWER_FORCE);
        }
    }

    return CMzWndEx::MzDefWndProc(message, wParam, lParam);
}
```

五、图标规范



色彩标准



• 构成图标主体灰度部分的色彩标准

White	Black	Light gradient	Medium gradient	Dark gradient
r 0 g 0 b 0	r 255 g 255 b 255	• r 255 g 255 b 255 • r 248 g 248 b 248	• r 221 g 219 b 217 • r 137 g 137 b 137	• r 36 g 36 b 36 • r 6 g 6 b 6

• 图标的灰度部分由以上色板提供的标准来应用

white:用于高光和空白区域

Black:用于阴影部分的着色标准

Light gradient: 较浅的灰度渐变

Medium gradient:用于处于中间灰度过渡的区域

Dark gradient:用于灰度较深的区域渐变

色彩标准



日历



便签



音乐



浏览器

• 色彩标准



默认图标的色彩是以97%的饱和度(S)和78%的明度(B)为基准的, 围绕着色相 (H) 从0°到360°的范围进行调节。

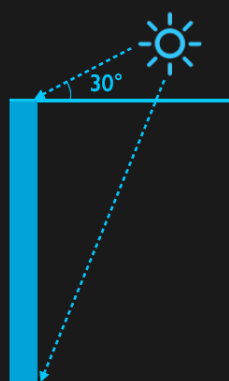
图标角度



• 图标角度

默认图标都是以正视角的角度来进行绘制, 眼睛与图标之间形成的夹角度数为0度。

光源和投影



- 光源位置
默认图标光源都是在其上方与图标成30度夹角的地方
- 投影位置
默认图标投影都是在其正下方,大约3px保持投影的统一位置,可以随着造型的变化来构造投影的形状。

字体

• 字体预览

汉体书写信息技术标准相
容档案下载使用界面简单
支援服务升级资讯专业制
作创意空间快速无线上网

Aa

Bb

Cc

- 图标中出现字体推荐使用微软雅黑
- 字体大小 : 13px

MZDN 非常规软件备案申请表

软件名称：		ICON：	
ProgramID：			
开 发 者：		电话号码：	
真实姓名：		邮政编码：	
通讯地址：			
身份证号码：			
软件简介：			
软件截图：			
非常规情况描述：	(请详细描述非常规情况)		

注：申请人必须真实填写以上信息，并在申请人一栏签名后与身份证复印件一同传真至魅族科技有限公司，待审核、测试通过后方为有效。传真号码：0756-6116250

申请人：

审核：

测试：

审批：