PB 的扩展 DLL 开发（超级篇）（七）

(PB 史上第一次开放的开发技术)

对象实例化与销毁

假设有 PB 代码是这样：

```
uo_json js
js = create uo_json
```

DLL 里实现同样的功能代码是：

```
OB_INST_ID obInstID = NULL;
if (rt_create_obinst(obThis, (LPTSTR)_T("uo_json"), &obInstID) == 1)
{
    //实例化成功
}
```

如果要把实例返回给 PB

```
OB_DATA obReturn = {0};
OB_CLASS_HNDL hndl = ob_get_obinst_class_hndl(obThis, obInstID);
OB_CLASS_ID classID =   hndl.class_id ; //得到其类型
ob_set_data_obinst(&obReturn, obInstID, classID, OB_INSTVAR_FIELD);
if(obInstID == NULL)
    ob_set_info_nullval(obReturn .info, TRUE); //如果创建失败，是空值，要把 NULL 标志打上，这样在 PB 里用 isnull() 或 isvalid() 判断时会是 false
ot_set_return_val(obThis, &obReturn); //返回对象实例
```

其他可以对象实例化的函数还有：

```
ob_create_obinst
ot_create_obinst_with_name
ot_create_obinst_at_lval
ob_create_object
ob_create_object_using
```

有 ot_开头的和 ob_开头的函数，ot_开头的应该是 runtime 类函数，比 ob_开头要高级一些，建议尽量使用 ot_开头的函数。

一个对象实例后，返回给 PB 环境，可以显式的 destroy js，如果不写 destroy 代码，PB 也会自动 销毁它。

DLL 里可以自己销毁

```
ob_destroy_obinst(obThis,obInstID);
```

要注意， PB 代码里并不知道你 DLL 里已销毁，会继续销毁，造成程序崩溃。所以，通常不需要调用 ob_destroy_obinst 这个销毁函数。

我在思考：这个销毁导致崩溃，应该跟引用计数有关系。只有当它的引用计数为 0 时，才会真正销毁。有个函数 rtDataCopy 它的最后一个参数表示是否增加引用计数。如果是传入参数，我们可以用 rtDataCopy 复制一份自己用，用完可以 ob_destroy_obinst，不会崩溃。

可惜，我没找到在哪可以直接添加这个引用计数。有知道的请告诉我一下，我好把这个坑填一下。(*(DWORD*)((DWORD)obInstID + 24)) ++; 倒是可以实现计数增 1，但总不敢用，还是有 SDK 函数用起来更放心。

以上说的是不可视对象和实例化和销毁。至于可视化对象，还是在 PB 设计环境里用鼠标拖放吧，DLL 里实现起来非常复杂，我也没掌握好，这里就不说了。

如果你对 system library 相关开发方式感兴趣，可到 QQ 群 624409252 共享里大自在的专用目录下下载 DEMO。