开发者文档

# 意图

本文档只是一个引导入门，讲解了工程结构，可能并不会详细到函数或者类的级别，如果阅读者看不懂这篇文档，那么我想你也看不懂工程代码了。

# 开发原则

代码风格无法限制，但请尽量保证一致性。

尽量能不使用宏定义就不要用，类模板也是。但是如果确实不得不用或者有很重要的作用，则可用。好的例子是DefineProperty，不好的例子是OnXMsg

xui的目前目标是简单的第一代Direct UI库，初衷是为了配合VNOC项目，并不一定要支持非常绚丽的效果，但需要保证一定的可扩展性和良好的代码组织，保证重构及升级的便捷。

整个工程的原则是，只需包含xui.h即可使用整个ui库，所以除了三方库以外，不要使用cpp文件，原则上使用hpp，代码包含为树型结构，xui.h为树根。

xui生成可见ui是通过xml来完成的，但通过代码级别也可以创建。

# 工程组织

目前为两部分：

Main.cpp 测试xui的代码，主要负责加载xui

Xui目录 包含了所有xui代码

类的关系从上到下：

1. 主模块，保证UI库运行态。
2. UI生成器，通过给定的XML生成UI
3. UI对象。父类皆为CXElement，用来描述各种ui元素，并负责各自的绘制工作。

# 主要模块

## 主模块

主模块为CXUI类，单例。其主要功能是运行WTL的CAppModule

## UI生成器

CXGaia类。通过指定的xml文件来生成ui对象。

## UI对象基类

所有ui对象继承自CXElement，其包含了一些基础功能，也实现了一些简单的ui功能。

## UI对象树

CXTree类，为CXElement的基类。维护了UI对象树。

每个元素都有一个ID值，存储在这一层。

## 属性

CXProperty 类。CXElement中有该成员，其只有Set和Get两个接口，存储了(key,value)形式的值，用于存储元素的自我属性。可以认为这就是一个map

在CXElement中，有如下属性表，描述了支持的属性，如：

**XProperty\_Begin**

**XFakeProperty(Size)**

**XProperty(Rect)**

**XProperty(Color)**

**XProperty(BorderColor)**

**XProperty(BorderWidth)**

**XProperty(ToolTip)**

**XProperty\_End;**

那么，CXElement及其子类，就一定有对应的SetXXX及GetXXX成员，如SetSize、GetSize。

在XProperty下方有一段代码，定义了所有支持的属性。这些属性被各种UI对象类选择性的实现:

**DefineProperty**(ID, CString, \_T(""));

DefineProperty(Rect, CRect, CRect());

DefineProperty(Position, CPoint, CPoint());

DefineProperty(Size, CSize, CSize());

DefineProperty(Text, CString, \_T(""));

第一个参数为名称，第二个为类型，第三个为默认值。

每种参数类型都有对应的XML解析器，也在该头文件中，工作方式及时机详见CXElement::SetXMLProperty成员函数，每个UI对象所支持的属性，不一定都要在解析XML的时候也做支持，在该函数中的转换器列表不一定等于属性表。

# 消息

所有信息流动及工作流程基本都是基于消息，皆定义在XMsg.h中。消息可以指定发给某个元素，也可以广播给下方的子节点，也可以顺着树型结构向上发。通过CXElement的两个成员函数完成这个工作：

**VOID \_SendXMessageToChildren(CXMsg& pMsg);**

**VOID \_RaiseXMessageUp(CXMsg& msg);**

如果只想让某一个对象处理消息，请调用该对象的ProcessXMessage。

在ProcessXMessage中，有消息表:

**BEGIN\_XMSG\_MAP(msg)**

**OnXMsg(CXMsg\_PropertyChanged)**

**OnXMsg(CXMsg\_SizeChanged)**

**OnXMsg(CXMsg\_Layout)**

**OnXMsg(CXMsg\_Paint)**

**OnXMsg(CXMsg\_MouseEnter)**

**OnXMsg(CXMsg\_MouseLeave)**

**OnXMsg(CXMsg\_PaintElement)**

**END\_XMSG\_MAP;**

指明了该对象关心哪些消息。

# 绘制

在CXElement中存在一个内存DC对象m\_memDC，所有元素将绘制到该DC中，一旦收到WM\_PAINT，将利用该元素的内存DC直接合成。所以，每个元素应当在其属性变更后，按需更新这个内存DC。

# 工作流程

1. 初始化环境
   1. 指定资源目录
   2. 通过XML创建第一个UI窗口
   3. 调用CXUI的Run，使UI库工作起来。此时UI库将占用住这个线程，直到收到WM\_QUIT消息，才会往下执行。
2. 通过给定的XML创建UI对象，组成UI树，树根应当为可见窗口。
   1. 循环递归读取XML，按照节点名称找到对应的UI类，new出实例对象，挂接到UI树上。
   2. 读取XML节点的属性，调用实例对象的SetXMLProperty接口设置属性。
   3. 完成创建后，发送排版消息给UI树，完成排版；发送XMsg\_PaintElement，完成第一次绘制。
   4. 若根节点为窗口，此时会被Create出来并显示，收到WM\_PAINT，转换为内部消息，组织子节点按序绘制，最后贴到真窗口中，完成绘制。

\*现存问题较多，文档可能不定时更新

\*很多地方设计的是有问题的