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
  - 1.1 指定资源目录
  - 1.2 通过 XML 创建第一个 UI 窗口
  - 1.3 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。
  - 2.1 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。
  - 2.2 读取 XML 节点的属性，调用实例对象的 SetXMLProperty 接口设置属性。
  - 2.3 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。
  - 2.4 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