NexTransmitter多语言概要设计文档

1. **设计目的**

多国语言和不同语言操作系统的存在，软件所使用的系统环境的存在差异，为了便于软件出口到多个国家，软件需要提供多国家语言的支持。

1. **设计要点**

多语言设计考虑的关键点主要有如下四点：

1）识别当前的操作系统的语言类型（暂时只考虑中/英两个版本的操作系统）；

2）多语言资源的管理，根据当前操作系统的语言类型，选择对应的语言类型；

3）多语言支持的扩展性

1. **设计步骤**
2. 识别当前的操作系统的语言类型：

+ // 获取系统语言类型，并设置界面语言类型

+ LANGID langID = GetSystemDefaultLangID();

+ if (langID == 0x0804)

+ {

+ g\_emLanType = enumLangIdCHN;

+ }

+ else

+ {

+ g\_emLanType = enumLangIdENG;

+ }

首先获取当前操作系统的语言类型，langID为0x0804即为中文系统。

1. 多语言资源的管理，根据当前操作系统的语言类型，选择对应的语言类型

+enum EMLangID

+{

+ enumLangIdCHN = (u8)0, //简体中文

+ enumLangIdENG = (u8)1 //English

+};

定义两种语言类型，

+extern EMLangID g\_emLanType; //语言类型

+

+#define TRANSLATE(text) \

+ CString( g\_emLanType == enumLangIdCHN ? text##CHN : text##ENG )

+

+ #define NET\_STATUS\_CONNECTING\_CHN "正在连接显示端"

+ #define NET\_STATUS\_CONNECTING\_ENG "Connecting"

+#define STRING\_NET\_STATUS\_CONNECTING TRANSLATE(NET\_STATUS\_CONNECTING\_)

+

+ #define NET\_STATUS\_CONNECTED\_CHN "已连接显示端"

+ #define NET\_STATUS\_CONNECTED\_ENG "Connected"

+#define STRING\_NET\_STATUS\_CONNECTED TRANSLATE(NET\_STATUS\_CONNECTED\_)

……

利用宏参数，根据系统语言g\_emLanType，选择对应的语言类型。

1. 多语言支持的扩展性

宏字符串的拼接，具有可扩展性。

1. **设计约束**

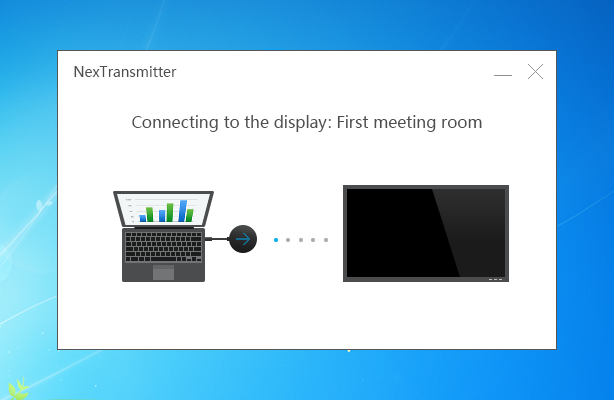
Windows操作系统。

1. **设计结果**

中文界面：

****

英文界面：

****