builder（构建模式）

简述：

separate the construction of a complex object from its representation so that the same construction process can create different representation

（分离复杂的构建方法和表示方法，以至于同样的构建方法创建不同的表示方法）

原理：

利用builder模式，可以使用一套构建方法，构建出不同的表示方式。

接口：

```
struct _XmlBuilder;
typedef struct _XmlBuilder XmlBuilder;

typedef void (*XmlBuilderOnStartElementFunc)(XmlBuilder* thiz, const char* tag, const char** attrs);
typedef void (*XmlBuilderOnEndElementFunc)(XmlBuilder* thiz, const char* tag);
typedef void (*XmlBuilderOnTextFunc)(XmlBuilder* thiz, const char* text, size_t length);
typedef void (*XmlBuilderOnCommentFunc)(XmlBuilder* thiz, const char* text, size_t length);
typedef void (*XmlBuilderOnPiElementFunc)(XmlBuilder* thiz, const char* tag, const char** attrs);
typedef void (*XmlBuilderOnErrorFunc)(XmlBuilder* thiz, int line, int row, const char* message);
typedef void (*XmlBuilderDestroyFunc)(XmlBuilder* thiz);

struct _XmlBuilder
{
        XmlBuilderOnStartElementFunc on_start_element;
        XmlBuilderOnEndElementFunc   on_end_element;
        XmlBuilderOnTextFunc         on_text;
        XmlBuilderOnCommentFunc      on_comment;
        XmlBuilderOnPiElementFunc    on_pi_element;
        XmlBuilderOnErrorFunc        on_error;
        XmlBuilderDestroyFunc        destroy;

        char priv[1];
};
```

应用：

```
XmlBuilder* xml_builder_dump_create(FILE* fp)
{
        XmlBuilder* thiz = (XmlBuilder*)calloc(1, sizeof(XmlBuilder));

        if(thiz != NULL)
        {
```

```
        PrivInfo* priv = (PrivInfo*)thiz->priv;

        /* 绑定已经实现了的构建方法（XML文件格式） */
        thiz->on_start_element  = xml_builder_dump_on_start_element;
        thiz->on_end_element    = xml_builder_dump_on_end_element;
        thiz->on_text           = xml_builder_dump_on_text;
        thiz->on_comment        = xml_builder_dump_on_comment;
        thiz->on_pi_element     = xml_builder_dump_on_pi_element;
        thiz->on_error          = xml_builder_dump_on_error;
        thiz->destroy           = xml_builder_dump_destroy;

        priv->fp = fp != NULL ? fp : stdout;
    }

    return thiz;
}

int main(int argc, char* argv[])
{
    const char* pi_attrs[] = {"version", "1.0", "encoding", "utf-8", NULL};
    const char* root_attrs[] = {"name", "lixianjing", "blog", "http://www.limodev.cn/blog",NULL};

    XmlBuilder* thiz = xml_builder_dump_create(stdout);

    /* 利用构建方法，构建各种各样的表示方法（以XML格式文件为例）。 */
    xml_builder_on_pi_element(thiz, "xml", pi_attrs);
    xml_builder_on_comment(thiz,"comment", 6);
    xml_builder_on_start_element(thiz, "programmer", root_attrs);
    xml_builder_on_text(thiz,"text", 4);
    xml_builder_on_end_element(thiz, "programmer");

    xml_builder_destroy(thiz);

    return 0;
}
```