OpenCPN 探索——S57Obj

本文介绍 OpenCPN 的重要数据结构 S570bj。S570bj 是连接 SENC 文件和显示的桥梁，S570bj 是 OpenCPN 核心数据之一。

下面是针对 S570bj 的分析。

1. 首先看一下 S57Obj 的定义，

```
class S570bj
{
public:
      //  Public Methods
      S570bj();
      ~S570bj();
      S570bj(char *first_line, wxInputStream *fpx, double ref_lat, double ref_lon);
      wxString GetAttrValueAsString ( char *attr );
      int GetAttributeIndex( const char *AttrSeek );
      // Private Methods
private:
      bool IsUsefulAttribute(char *buf);
      int my_fgets( char *buf, int buf_len_max, wxInputStream& ifs );
      int my_bufgetl( char *ib_read, char *ib_end, char *buf, int buf_len_max );
public:
      // Instance Data
      char                    FeatureName[8];
      GeoPrim_t               Primitive_type;
  …
  …

}
```

其中 S570bj(char *first_line, wxInputStream *fpx, double ref_lat, double ref_lon)是从 SENC 文件读取内容，构建 S570bj 对象的重要函数，构建时会把 S570bj 的属性赋值给各个属性字段。下面是依次调用的顺序。

```
InitReturn s57chart::Init( const wxString& name, ChartInitFlag flags )
InitReturn s57chart::PostInit( ChartInitFlag flags, ColorScheme cs )
int s57chart::BuildRAZFromSENCFile( const wxString& FullPath )
S570bj *obj = new S570bj( buf, &fpx, 0, 0 );
```

在形成 S570bj 的过程中，会形成另一个重要的数据结构：ObjRazRules

```
typedef struct _ObjRazRules{
   LUPrec          *LUP;
   S570bj          *obj;
   s57chart        *chart;                 //dsr ... chart object owning this rule set
```

```
   struct _ObjRazRules *child;           // child list, used only for MultiPoint Soundings
   struct _ObjRazRules *next;
}ObjRazRules;
```

里面的 LUP 查找表是供 s52plib 使用的。

```
class s52plib {
public:
     s52plib( const wxString& PLib, bool b_forceLegacy = false );
    ~s52plib();
    void SetPPMM( float ppmm ) { canvas_pix_per_mm = ppmm;  }
    float GetPPMM() { return canvas_pix_per_mm; }
    LUPrec *S52_LUPLookup( LUPname LUP_name, const char * objectName,
        S57Obj *pObj, bool bStrict = 0 );
    int _LUP2rules( LUPrec *LUP, S57Obj *pObj );
    S52color* getColor( const char *colorName );
    wxColour getwxColour( const wxString &colorName );
    void UpdateMarinerParams( void );
    void ClearCNSYLUPArray( void );
    void GenerateStateHash();
    long GetStateHash() { return m_state_hash;  }
    void SetPLIBColorScheme( wxString scheme );
    wxString GetPLIBColorScheme( void ) { return m_ColorScheme; }
    void SetGLRendererString(const wxString &renderer);
    bool ObjectRenderCheck( ObjRazRules *rzRules, ViewPort *vp );
    bool ObjectRenderCheckPos( ObjRazRules *rzRules, ViewPort *vp );
    bool ObjectRenderCheckCat( ObjRazRules *rzRules, ViewPort *vp );
    bool ObjectRenderCheckCS( ObjRazRules *rzRules, ViewPort *vp );
    static void DestroyLUP( LUPrec *pLUP );
    static void ClearRulesCache( Rule *pR );
//    Temporarily save/restore the current colortable index
//    Useful for Thumbnail rendering
    void SaveColorScheme( void ) { m_colortable_index_save = m_colortable_index;}
    void RestoreColorScheme( void ) {}
….
…
}
```

s52plib 是 s52 Presentation Lib 的实现。

OpenCPN 没有完全实现 S52 标准的相关规定，其中条件符号化过程仅仅实现一小部分内容。如果要做到完全符合 S52 标准，还有很多工作要做。

s52plib 画图时，分为对象和区域不同显示。

S52plib 中还引用 RenderFromHPGL 实现绘图功能。

```
class RenderFromHPGL {
public:
    RenderFromHPGL( s52plib* plibarg );
    void SetTargetDC( wxDC* pdc );
```

```
    void SetTargetOpenGl();
    void SetTargetGCDC( wxGCDC* gdc );
    bool Render(char *str, char *col, wxPoint &r, wxPoint &pivot, double rot_angle = 0);

private:
    const char* findColorNameInRef( char colorCode, char* col );
    void RotatePoint( wxPoint& point, double angle );
    wxPoint ParsePoint( wxString& argument );
    void SetPen();
    void Line( wxPoint from, wxPoint to );
    void Circle( wxPoint center, int radius, bool filled = false );
    void Polygon();
    s52plib* plib;
    int scaleFactor;
    wxDC* targetDC;
    wxGCDC* targetGCDC;
    wxColor penColor;
    wxPen* pen;
    wxColor brushColor;
    wxBrush* brush;
    long penWidth;
    int noPoints;
    wxPoint polygon[100];
    bool renderToDC;
    bool renderToOpenGl;
    bool renderToGCDC;
    bool havePushedOpenGlAttrib;
}
```

OpenCPN 中绘图机制是最复杂的过程之一，也是最为核心的内容。其中涉及的内容多，算法复杂，理解了绘图机制，也就理解了 OpenCPN 最核心的东西，可以从 S57Obj 入手了解整个过程。SENC 的内容如何赋值给 S57Obj，如何根据 S57Obj 形成查找表，如果根据查找表形成 s52 绘制对象等。