# easyX+

## 意图

为什么写这个图形库呢？

因为

1. easyX官网自称easyX是一套针对VC的图形库，但却没有使用C++的功能，对不起“EasyX Library for C++”这个名字（也许叫它EasyX Library for C更好）。
2. 它的函数命名并不合理，既不是Camel命名法也不是Pascal命名法（我的意思并不是说不采用这两种方法就不合理），而是一股脑的小写字母，严重模糊了代码的可读性，没有引导给初学者良好的代码风格。
3. easyX宣称面向初学者，但很多内容对初学者并不友好（例如颜色模型中的HSL、HSV）。对于这些内容，初学者基本很少接触。而当他们晋升为高级程序员之后，他们又不满足于简单的easyX转而使用openGL等。

## 特点

* 简单，易上手
* 组织合理
* 计时器功能（timer）。
* 键盘支持

## 使用平台

Windows，系统版本待定。

## 使用语言

C++。

## 模块划分

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 剪裁区 | Window | 颜色模型 | Geometry | Text | Mouse | Keyboard | Timer |

## 度量单位

统一以**像素**为单位。

## Window

所有的可绘绘操作（包括几何图形和文字输出）都与特定的window相关联。除此之外，剪裁区也与特定的window关联。Window相当于一个画布，所以的绘制都在某个特定的窗口上进行。

为了方便图形的绘制，window类应该支持设置默认窗口的操作。

#### 属性

该模块只包含于窗口有关的一些属性：

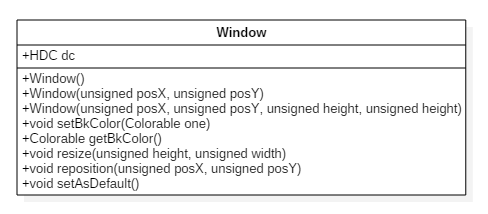
* 大小，即长、宽。
* 背景色。

#### 特点

* 多窗口支持
* 窗口之间各自独立

因为该图形库致力于简单，所以那些复杂的且与图形绘制无关的属性都不被支持（例如边框类型，窗口按钮等等）。

### UML图



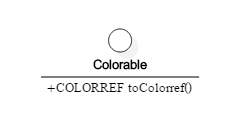
### 函数细节

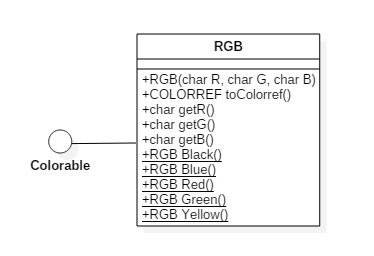
1. **构造函数**： 创建窗口并显示。 构造函数采用函数重载方式来提供灵活的创建。 这里不采用 耽单个构造函数配合默认参数的方法 是为了防止window类的错误构造（例如三个参数的构造是不被支持的）。
2. **setBkColor**： 设置窗口的背景颜色。传入一个实现Colorable接口的参数。
3. **getBkColor**： 获取窗口当前的颜色。 回Colorable接口。
4. **resize**： 重新设置窗口的大小。传入参数为新的窗口大小，unsigned类型表示的大小足以支持市面上的屏幕分辨率。
5. **reposition**： 设置窗口的位置。传入新的窗口位置，坐标0点为屏幕左上角，向右向下位置逐渐变大。
6. **setAsDefault**：将该窗口设置为默认窗口。 若绘制图形时未显示指明输出窗口，则使用此默认窗口为输出窗口。

## 颜色模型

颜色模型模塑图形的颜色，并遵循统一的Colorable接口。默认支持的颜色模型仅为RGB颜色模型。

### UML图





### Colorable接口

1. toColorref：返回代表颜色值的COLORREF值。

### 函数细节(RGB)

1. **构造函数**：构造函数从传入的三原色值构造代表RGB模型的RGB类。
2. **toColorref**：RGB类实现的Colorable接口，返回代表颜色的COLORREF值。
3. **getR**：返回红色值。
4. **getG**：返回绿色值。
5. **getB**：返回蓝色值。
6. **Black**（静态）：返回代表黑色的RGB类。
7. **Blue**（静态）：返回代表蓝色的RGB类。
8. **Red**（静态）：返回代表红色的RGB类。
9. **Green**（静态）：返回代表绿色的RGB类。
10. **Yellow**（静态）：返回代表黄色的RGB类。
11. **…**（可以有更多的静态函数）

## Geometry

Geometry模块囊括大部分的几何图形，包括（注意：该模块不支持‘体’）：

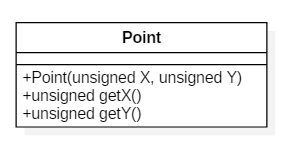
* 点 ： 点
* 线 ： 直线、弧、贝塞尔曲线、折线
* 面 ： Chord、椭圆、Pie、多边形、矩形、圆角矩形

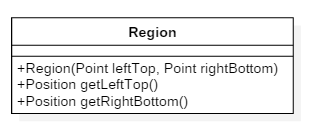
注意**线**与**面**的区别：线不是封闭区域，而面是。

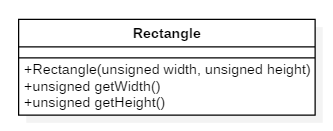
### 几何属性

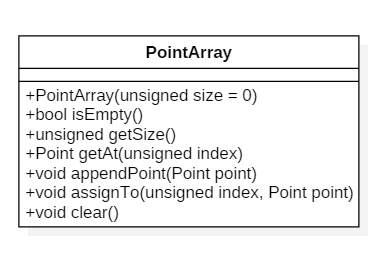
几何属性是与几何图形本身相关联的属性。就图形学来说，图形的属性主要分为**颜色**和**形状**（这个库中表现为点的数量语位置）。

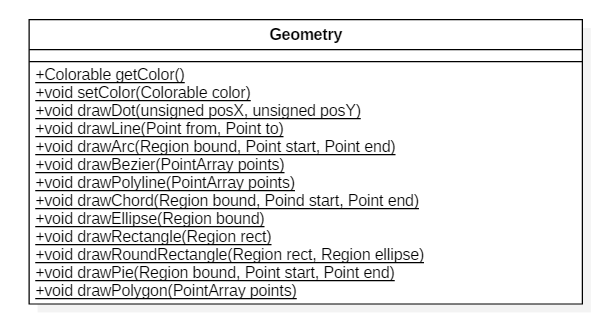
### UML图











### 函数细节（Point）

1. **构造函数**：传入x、y轴的坐标，构造Point类。
2. **getX**：返回x轴的坐标。
3. **getY**：返回y轴的坐标。

### 函数细节（Region）

1. **构造函数**：传入左上角与右下角的位置坐标，构造区域范围类。Region由两个坐标点表示，分别为左上角与右下角。
2. **getLeftTop**：返回左上角位置。
3. **getRightBottom**：返回右下角位置。

### 函数细节（Rectangle）

1. **构造函数**：传入长、宽，构造矩形。
2. **getWidth**：获取宽。
3. **getHeight**：获取长。

### 函数细节（PointArray）

1. **构造函数**：传入point的个数（后续支持聚合初始化）。
2. **isEmpty**：检测数组是否为空。
3. **getSize**：获取数组的大小。
4. **getAt**：获取以传入参数为下标的point。
5. **appendPoint**：在尾部追加point元素。
6. **assignTo**：给某个元素赋值。
7. **clear**：清除数组。

### 函数细节（Geometry）

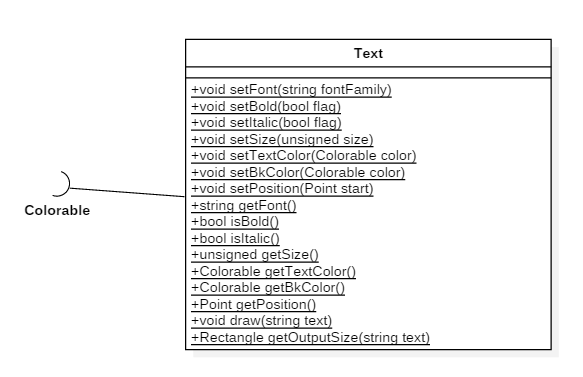
1. **getColor**（静态）：获取当前的颜色设置。
2. **setColor**（静态）：设置颜色。
3. **drawXXX**（静态）：集合图形绘制系列函数。

## Text

该模块为文本支持模块，假定文本与以下属性相关联：

* 字体，包括字体类别、大小、样式（斜体，加粗等）
* 颜色，包括字体颜色、背景填充颜色
* 位置
* 输出区域（以矩形表示）

### UML图



### 函数细节

1. **setXXX**（静态）：设置系列函数，支持字体、大小、样式的设置、输出位置等。
2. **getXXX**（静态）：获取系列函数，同上。
3. **draw**（静态）：绘制函数，即输出。 输出的位置有setPosition决定。
4. **getOutputSize**（静态）：获得文本的输出大小信息，按照调用前的设定来估算。

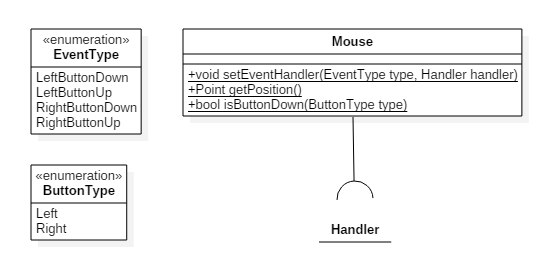
## Mouse

我希望鼠标与键盘的支持对初学者来说是非常友好的。因此，这个模块只提供这些功能：

* 检测单个按键状态：按下、释放
* 鼠标事件：按下、释放
* 获取鼠标位置

这里不可能出现鼠标左右按键同时按下的情况，因为我觉得这不太可能（-\_-）。所以，同时检测两个以上的鼠标按钮状态在这里是不支持的， 并且除传统的左右键以外的按钮都在这个库的功能范围外。

### UML图



### 函数细节

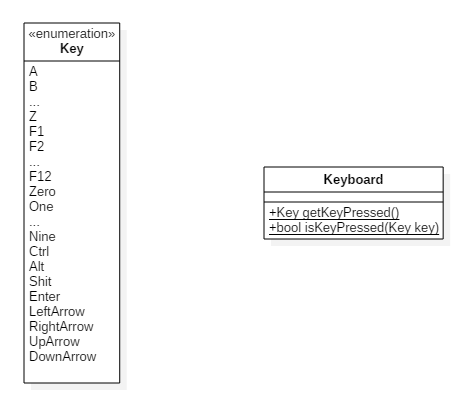
1. **getPosition**（静态）：获取当前鼠标位置。
2. **isButtonDown**（静态）：检测鼠标按键是否按下。
3. **setEventHandler**（静态）：设置时间处理器，支持的事件如图。

## Keyboard

键盘模块提供的功能为：

* 检测是否有按键按下，并返回对应按键
* 检测单个按键的状态（按下或释放）

### UMl图



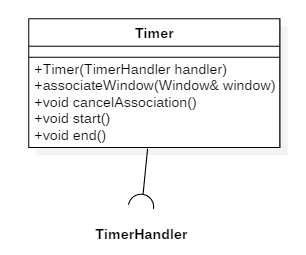
### 函数细节

1. **getKeyPressed**（静态）：检测是否有按键按下，若有，则返回最先按下的按键码；若无，返回代码表示无。
2. **isKeyPressed**（静态）：检测按键是否按下。

## Timer（可选）

每个Timer由一段执行代码（即函数）进行初始化，每个window最多只能由一个timer与之相关联。Window关联timer后，可以进行timer的取消与更改。

### UML图



### 函数细节

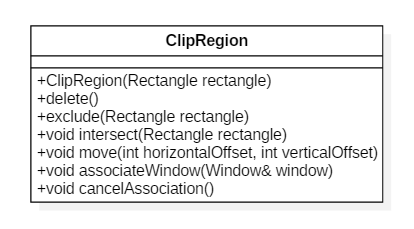
1. **构造函数**：传入一个计时器处理器，用以执行计时器触发时的动作。
2. **associateWindow**：关联窗口。 关联窗口前，计时器不能开始，否则会抛出异常。
3. **cancelAssociation**：取消窗口的关联。
4. **start**：打开计时器。
5. **end**：关闭计时器。

## 剪裁区（可选）

easyX+支持剪裁区功能，每个剪裁区需要与窗口关联，具体支持的剪裁区操作为：

* 创建
* 删除
* Exclude
* Intersect
* Move

### UML图



### 函数细节

1. **构造函数**：传入一个区域作为剪裁区。
2. **delete**：删除该剪裁区。
3. **exclude**：取当前剪裁区域传入区域的并集为新的剪裁区。
4. **intersect**：取当前剪裁区域传入区域的交集为新的剪裁区。
5. **move**：移动剪裁区。
6. **associateWindow**：关联窗口。
7. **cancelAssociation**：取消窗口的关联。