Subject: Re: Using GoogleMaps from U++
Posted by mirek on Wed, 03 Feb 2010 17:30:20 GMT
View Forum Message <> Reply to Message

Didier wrote on Fri, 29 January 2010 14:38Fantastic,

Just what I needed !!!

Thanks

Now with latitude/longitude <-> pixel conversions:

```
#include <CtrlLib/CtrlLib.h>
#include <Web/Web.h>

using namespace Upp;

#define LLOG(x) // LOG(x)

String apikey = "";

void SetGoogleMapsKey(const char *key)
{
 apikey = key;
}

String GetGoogleMap(double center_x, double center_y, int zoom, int cx, int cy,
                    const char *format = "png", String *error = NULL)
{
 String request;
 request << "http://maps.google.com/maps/api/staticmap?center=" <<
        AsString(center_y) << ',' << AsString(center_x) <<
        "&zoom=" << zoom <<
        "&size=" << cx << 'x' << cy <<
        "&format=" << format <<
        "&sensor=false&key=" << apikey;
 LLOG(request);
 return HttpClientGet(request, NULL, error);
}

Image GetGoogleMapImage(double center_x, double center_y, int zoom, int cx, int cy,
                    const char *format = "png", String *error = NULL)
{
 return StreamRaster::LoadStringAny(GetGoogleMap(center_x, center_y, zoom, cx, cy, format,
error));
}
```

```
double CvDeg(int deg, int minutes, double seconds)
{
 return deg + (double)minutes / 60 + seconds / 3600;
}

static const double sOffset = 268435456;
static const double sRadius = sOffset / M_PI;

static int LToX_(double x)
{
 return int(sOffset + sRadius * x * M_PI / 180);
}

static int LToY_(double y)
{
 return int(sOffset - sRadius * log((1 + sin(y * M_PI / 180))/(1 - sin( y * M_PI / 180))) / 2);
}

static double XToL_(int x)
{
 return ((x - sOffset) / sRadius) * 180 / M_PI;
}

static double YToL_(int y)
{
 return (M_PI / 2 - 2 * atan(exp((y - sOffset) / sRadius))) * 180 / M_PI;
}

Pointf GoogleMapsPixelToGps(Pointf center, int zoom, Point diff)
{
 return Pointf(XToL_(LToX_(center.x) + (diff.x << (21 - zoom))),
               YToL_(LToY_(center.y) + (diff.y << (21 - zoom))));
}

Pointf GoogleMapsPixelToGps(Pointf center, int zoom, Size sz, Point p)
{
 return GoogleMapsPixelToGps(center, zoom, p - sz / 2);
}

Pointf GoogleMapsGpsToPixelDiff(Pointf center, int zoom, Pointf gps)
{
 return Pointf((LToX_(center.x) - LToX_(gps.x)) >> (21 - zoom),
               (LToY_(center.y) - LToY_(gps.y)) >> (21 - zoom));
}

Pointf GoogleMapsGpsToPixel(Pointf center, int zoom, Size sz, Pointf gps)
{
 return sz / 2 - GoogleMapsGpsToPixelDiff(center, zoom, gps);
```

```
}

struct App : public TopWindow {
 Pointf home;
 Pointf center;
 String error;
 Image  map;
 int    zoom;

 virtual void LeftDown(Point p, dword)
 {
  center = GoogleMapsPixelToGps(center, zoom, Size(640, 640), p);
  map = GetGoogleMapImage(center.x, center.y, zoom, 640, 640, "png", &error);
  Refresh();
 }

 virtual void Paint(Draw& w) {
  Size sz = GetSize();
  String error;
  w.DrawRect(sz, SColorPaper());
  if(IsNull(map))
   w.DrawText(0, 0, error);
  else {
   w.DrawImage(0, 0, map);
   w.DrawImage(320, 320, CtrlImg::arrow());
   Point hp = GoogleMapsGpsToPixel(center, zoom, Size(640, 640), home);
   w.DrawImage(hp.x, hp.y, CtrlImg::arrow());
  }
 }

 App() {
  zoom = 15;
  home = center = Pointf(CvDeg(14, 22, 21.75), CvDeg(50, 7, 57.96));
  map = GetGoogleMapImage(center.x, center.y, zoom, 640, 640, "png", &error);
 }
};

GUI_APP_MAIN
{
 SetGoogleMapsKey(....);

 App().Run();
}
```