Geometrické praktikum

Jan Laštovička

24. února 2015

1 Kreslení objektů v rovině

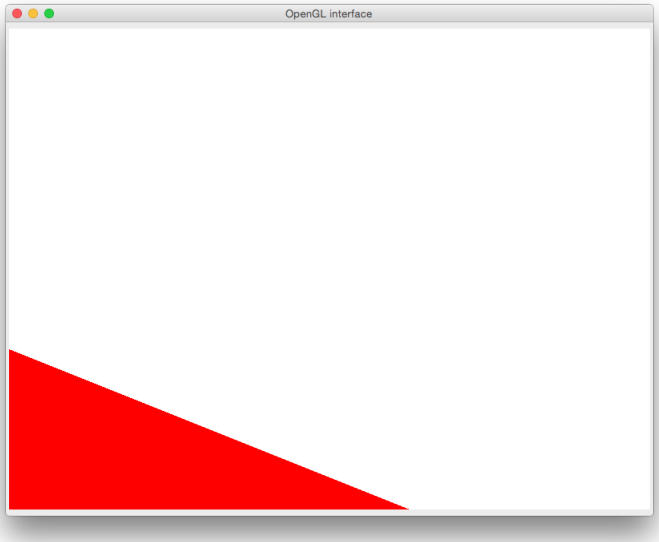
Začneme malým příkladem. Nahrajte knihovnu `lisp-gl` načtením (například z nabídky `File > Load...`) souboru `load.lisp` a vyhodnoťte následující kód:

```
(opengl
  (color (color-point :red))
  (polygon
    (vertex (point 0 0))
    (vertex (point 500 0))
    (vertex (point 0 200))))
```

Otevře se nové okno s červeným trojúhelníkem. Výsledek je zachycen na obrázku 1.

Postupně si rozebereme právě použitý kód. Začneme makrem `opengl`, které vykresluje obsah do nově vytvořeného OpenGL okna. Makro bere výrazy jejichž smyslem je kreslit obsah okna. Při potřebě překreslit okno jsou tyto výrazy postupně vyhodnocovány. Funkce `color` mění aktuální barvu. Do další změny jsou všechny objekty kresleny nastavenou barvou. Funkce jako svůj argument bere bod z jednotkové krychle určující barvu v RGB modelu. Funkce `color-point` vrací bod v krychli reprezentující barvu zadanou jejím názvem. Tedy `(color (color-point :red))` nastaví aktuální barvu na červenou. Makro `polygon` kreslí konvexní polygon. Jeho argumenty jsou výrazy, které jsou postupně vyhodnocovány. Jejich významem je zadávání vrcholů polygonu. Funkce `vertex` bere dvourozměrný bod, který je poslán do grafické karty jako další vrchol polygonu. Funkce `point` vytvoří bod o zadaných souřadnicích.

Před zadáním úkolů si představíme funkce pro práci s body a vektory. Ty budou užitečné pro řešení úkolů. Už jsme si představili funkci `point` na vytváření bodů. Podobně pracuje fuknce `vect` vytvářející vektory. Následující funkce pracují jak s vektory tak body. Funkce `coordinates` vrací souřadnice bodu nebo vektoru v seznamu. Funkce `x`, `y` a `z` zjišťují první,



Obrázek 1: Trojúhelník

druhou a třetí souřadnici. Dimenzi prostoru, ve kterém se bod nebo vektor nachází, zjistíme funkcí `dimension`.

Násobení skaláru vektorem realizuje funkce `mult`, která bere koeficient násobení a vektor. Dva vektory sčítá funkce `plus`. K bodu vektor přičte funkce `plus`. Vektor vedoucí od jednoho bodu k druhému zjistí funkce `minus`. U vektoru, který je z prostoru dimenze dva, zjišťuje funkce `phi` jeho odchylku od kladné poloosy x. Funkce `rotate` otočí vektor dimenze dva o zadaný úhel.

Úkol 1. Napište funkci `regular-polygon`, která bere střed (bod), poloměr a počet vrcholů. Funkce nakreslí pravidelný mnohoúhelník vepsaný do kružnice. Vrchol mnohoúhelníku směřuje nahoru.

Úkol 2. Naprogramujte funkci `circle`, která bere střed (bod) a poloměr, kreslící kružnici.

Úkol 3. Napište funkci `star` kreslící hvězdu. Funkce bere střed (bod), dva poloměry (r_1 a r_2) a počet cípů. Hvězdě je opsaná kružnice o poloměru r_1 a do hvězdy je vepsaná kružnice o poloměru r_2. Cíp hvězdy směřuje nahoru.

2 Konvexní polygony

Makro `opengl` neumožňuje ukládání vlastností kreslení (např. barva objektu) a ani definici obsluh uživatelských akcí (např. kliknutí myši). Tuto funkcionalitu poskytuje obecnější makro `opengl-interface` jehož zjednodušená syntax je následující:

`(opengl-interface` *slots* `(`&*key display mouse-press-primary mouse-press-secondary*`))`

Kde *slots* je definice slotů kreslícího plátna (canvas), které se definují stejně jako u makra `defclass`. Klíč *display* má za hodnotu funkci jednoho argumentu (plátno). Funkce se stará o kreslení obsahu na plátno. Klíč *mouse-press-primary* očekává jako hodnotu funkci, která bere plátno a bod. Funkce je volána v případě, že došlo ke zmáčknutí hlavního (primárního) tlačítka myšli. Do argumentů jsou dány kreslící plátno a pozice myš. Posledním klíčem je *mouse-press-secondary*, který funguje stejně jako *mouse-press-primary* s tím rozdílem, že je funkce volána v případě stisku sekundárního tlačítka.

Následující příklad zobrazí okno s trojúhelníkem jehož barva se mění po stisku tlačítka myši.

```
(opengl-interface
 ((color :initform :red))
 (:display
  (lambda (canvas)
    (with-slots (color) canvas
      (color (color-point color))
      (polygon
        (vertex (point 0 0))
        (vertex (point 500 0))
        (vertex (point 0 100)))))
  :mouse-press-primary
   (lambda (canvas point)
     (with-slots (color) canvas
       (if (eql color :red)
           (setf color :blue)
         (setf color :red))
       (invalidate canvas)))))
```

V příkladu je použita nová funkce `invalidate`, která vynutí překreslení daného plátna.

Před zadáním úkolů si představíme funkci `cross-product`, která vrátí vektorový součin dvou vektorů dimenze tři. Připomeňme, že výsledkem je vektor kolmý k oběma zadaným vektorům směřující do poloprostoru určeného pravidlem pravé ruky. S použitím představené funkce splňte následující úkoly.

Úkol 4. Napište funkci, která rozhodne, zda je polygon zadaný jako seznam vrcholů (bodů) konvexní.

Úkol 5. Napište funkci rozhodující, zda je bod v konvexním polygonu.

Úkol 6. Vytvořte okno s následujícím chováním. Po stisku hlavního tlačítka se přidá vrchol do konvexního polygonu. Po prvních dvou stiscích, kdy se zadávají první dva vrcholy polygonu, zůstane okno prázdné. Po třetím stisku se zobrazí trojúhelník. Další body budou přidávané mezi první a poslední vrchol polygonu, ale pouze v případě, že polygon zůstane konvexní. V opačném případě se zobrazí (pomocí funkce `capi:display-message`) chybová hláška. Pokud se do takto vznikajícího polygonu klikne sekundárním tlačítkem, dojde k změně jeho barvy.