Floor Plan Language (FPL)

Xinwei Zhang (xz2663)
Chih-Hung Lu (cl3519)
Dongdong She (ds3619)
Yipeng Zhou (yz3169)
Introduction

- Replace AutoCAD
- C-like language using OpenGL
Scanner and Parser

- Built on the top of microC
- Built in data types: bed, door, desk, window, wall, circle and rectangle
- User-defined struct and function: Bedroom, MakeBedroom
- Built in functions: put, rotate, render
Semantic Checker

- Continue the design of MicroC Semantic Checker
- Built-in Types: wall, bed, door, desk, window, rectangle, circle
- User-defined Struct

Diagram:

- AST
- program
- structs
  - members
  - struct_name
- functions
  - formals
- globals
  - locals
  - ...

AST
int main()
{
    window d;
    door e;
    d = window(0.7, 0.8);
    d = window(0.8);
    e = desk(0.9, 1.0);
    put(d, 5.0, 5.0);
}

struct Bedroom {
    window e;
    rectangle f;
    circle g;
};

int main()
{
    Bedroom b;
    door e;
    b.e = window(0.7, 0.8);
    b.f = desk(0.9, 1.0);
    b.t = 1;
}

Structs: Bedroom

members

type: window

id:e

Struct_name: b

locals

type: window

id:d
Code Generator

- FPL object (6-tuple): “wall”, “bed”, “door”, “desk”, “window”, “rectangle”, “circle”
- External function: put, rotate, render
struct Bedroom {
    window e;
    rectangle f;
    circle g;
};

void foo(){
    Bedroom r;
    ...
}

int main(){
    Bedroom r;
    desk k;
    foo();
    ...
}
FPL Object Constructor

- FPL object: object with built-in type like “wall”, “bed”, “door”, “desk”, “window”, “rectangle”, “circle”
- 6-tuple: (type, degree, x of region, y of region, x of position, y of position)

```c
int main(){
  desk k;
  wall w;
  k = desk(1.0, 1.0);
  w = wall(1.0, 4.0);
  ....
}
```

<table>
<thead>
<tr>
<th>key</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPL Object</td>
<td>Map @ main</td>
</tr>
<tr>
<td>key</td>
<td>value</td>
</tr>
<tr>
<td>FPL object</td>
<td>type</td>
</tr>
<tr>
<td>k</td>
<td>3</td>
</tr>
<tr>
<td>w</td>
<td>0</td>
</tr>
</tbody>
</table>
FPL External Function Caller

- **put**: set the position of FPL object
  a. Find the members from Struct Map, and iterate the 6-tuples in FPL Object Map
  b. Change “x (position)” and “y (position)” fields
  c. Invoke the external “put”, with the 6-tuple as parameters
- **rotate**: set the degree of rotation
  a. Find the members from Struct Map, and iterate the 6-tuples in FPL Object Map
  b. Change “degree” field
- **render**: Invoke the external “render”

```c
int main(){
    ....
    k = desk(1.0, 1.0);
    w = wall(1.0, 4.0);
    put(k, 1.0, 2.0);
    rotate(w, 90);
```
Graph generator

- A native graph library based on OpenGL in C
- Handle most of heavy work of drawing task
- Take in parameters from FPL and perform corresponding task
Graph library design

- Use ancient OpenGL in C
- Parameters handling
- Drawing implementation
Drawing implementation

1. Basic support for primitive graph such as line, rectangle

   ```
   glBegin(GL_LINE);
   glVertex2f(1.0f,1.0f);   //starts drawing of line
   glVertex2f(-1.0f,-1.0f); //use coordinate to define right point
   glEnd();                 //use coordinate to define left corner
   //end coordinate to define left corner
   glEnd();                 //end drawing of line
   ```

2. Implement a special graph circle using approximation of polygon

   ```
   glBegin(GL_LINE_GROUP);  //draw a polygon of 600 angle to approximate a circle
   for(ii = 0; ii < 600; ii++)
   {
     float theta = 2 * 3.1415926f * ii / 600;   //get the current angle
     float x = r * cosf(theta);                     //calculate the x component
     float y = r * sinf(theta);                     //calculate the y component
     glVertex2f(x, y);                             //output vertex
   }
   glEnd();                                       //End quadrilateral coordinates
   ```
Demo