

---

Subject: Re: Help with a possible design problem?  
Posted by [snap2000](#) on Sat, 03 Feb 2007 00:04:52 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

I tried implementing the changes, but I'm getting a few compiler errors with this particular code:

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

#define SCREENWIDTH 600
#define SCREENHEIGHT 480

#define PADDLEWIDTH 12
#define PADDLEHEIGHT 40

#define BALLSIZE 8
#define BALLSPEED 5

enum Which { PLAYER, COMPUTER };
class Ball;
class Player;
class Pong;

class Ball {
private:
    int bx;
    float by;        // Ball pos
    int sx;
    float sy;        // Ball speed
    int size;        // Ball size
    Point *samples;  // Vector samples for collision detection
    int numSamples;  // Number of samples to be taken
    int serve;       // Who the ball is being served to
    bool paused;     // If game is paused

    Player *player, *computer;    // Paddles
public:
    Ball() { serve = -1; Reset(); }
    Rect GetRect() { Rect r(int(bx), int(by), int(bx + size), int(by + size)); return r; }
    void Update();           // Update ball position
    void CalculateVectorSamples(); // Calculate collision sample points
    void Reset();           // Reset ball pos/attrs
    void SetPaddles(Player *p, Player *c); // Set pointers to paddles
    void Pause(bool p = true) { paused = p; } // Pause/unpause the game
    bool IsPaused() { return paused; } // Return if paused or not;
};

void Ball::SetPaddles(Player *p, Player *c) {
    player = p;
```

```

computer = c;
}

void Ball::Reset() {
    size = BALLSIZE;        // Init ball size
    bx = (SCREENWIDTH - size) / 2;    // center ball horizontally
    by = (SCREENHEIGHT - size) / 2;    // center ball vertically
    sx = BALLSPEED * serve;    // Init ball x-speed
    sy = 0; //TODO: Pick random angle    // Init ball y-speed (angle)
}

void Ball::CalculateVectorSamples() {
    numSamples = (sx > size) ? sx / size : 1;    // Number of samples to take
    float dy = sy / numSamples;    // Slope of ball movement
    samples = new Point[numSamples];
    for(int s = 1; s <= numSamples; s++) {
        samples[s - 1].x = bx + (s * size);    // Calculate sample x-pos
        samples[s - 1].y = int(by + (s * dy));    // Calculate sample y-pos
    }
}

void Ball::Update() {
    if(IsPaused()) {
        return;
    } else if(bx + sx > SCREENWIDTH) {    // Ball exits to right of screen
        // TODO: add score and re-serve ball
        serve = -1;
        Reset();
    } else if(bx + size + sx < 0) {    // Ball exits to left of screen
        // TODO: add score and re-serve ball
        serve = 1;
        Reset();
    } else {    // Ball is still in play
        CalculateVectorSamples();
        PromptOK(AsString(player->GetScore()));
        Pause();
        // [[ HOW DO I GET ACCESS TO PADDLE? player/computer are in Pong class ]]
        // TODO: Test each sample against nearest paddle for collision
        // TODO: Correct angle and direction of ball if collided
        // TODO: Else continue in direction:
        bx += sx;
        by += sy;
    }
}

class Player {
private:
    int score, px, py, height;
}

```

```

Which who;
Ball *ball;
public:
Player() {}
Player(Which w, Ball *b);
int GetScore() { return score; }
Rect GetRect() { Rect r(px, py, px + PADDLEWIDTH, py + height); return r; }
void Update();
};

```

```

Player::Player(Which w, Ball *b) {
ball = b;
score = 0;          // Init score
py = (SCREENHEIGHT - PADDLEHEIGHT) / 2; // Init paddle y-pos
height = PADDLEHEIGHT; // Init paddle height
px = (w == PLAYER)
? 10 // Set player paddle x-pos
: SCREENWIDTH - PADDLEWIDTH - 10; // Set computer paddle x-pos
}

```

```

void Player::Update() {
if(ball->IsPaused())
return;
switch(who) {
case PLAYER:
// TODO: Get mouse y pos
break;
case COMPUTER:
// TODO: Move computer paddle toward ball
break;
}
}

```

```

class Pong : public TopWindow {
public:
void Paint(Draw& w);
private:
Player player, computer;
Ball ball;
public:
typedef Pong CLASSNAME;
Pong();
void Update();
};

```

```

void Pong::Update() {
// player.Update(); // Update player paddle position
// computer.Update(); // Update computer paddle position
}

```

```

ball.Update();      // Update ball position
Refresh();
}

void Pong::Paint(Draw& w) {
w.DrawRect(GetSize(), SBlack);
w.DrawRect(player.GetRect(), SWhite);
w.DrawRect(computer.GetRect(), SWhite);
w.DrawRect(ball.GetRect(), SWhite);
/* For showing projected samples ( samples and numSamples need to be made public to work )
for(int i = 0; i < ball.numSamples; i++) {
//w.DrawText(SCREENWIDTH / 2, 10 + 10 * i, ball.samples[i].ToString(), StdFont(), SWhite);
w.DrawRect(ball.samples[i].x, ball.samples[i].y, BALLSIZE, BALLSIZE, SWhite);
}
*/
}

Pong::Pong() {
Title("Snap.Pong");
SetRect(0, 0, SCREENWIDTH, SCREENHEIGHT);
MinimizeBox();

player = Player(PLAYER, &ball);    // Init player
computer = Player(COMPUTER, &ball); // Init computer
ball.SetPaddles(&player, &computer); // Pass paddles to Ball class

BackPaint();
Update();
SetTimeCallback(-25, THISBACK(Update));
}

GUI_APP_MAIN
{
Pong().Run();
}

```

The errors say:

```

Quote:C:\Apps\Pong\main.cpp: In member function `void Ball::Update()':
C:\Apps\Pong\main.cpp:77: error: invalid use of undefined type `struct Player'
C:\Apps\Pong\main.cpp:14: error: forward declaration of `struct Player'

```

Apparently it doesn't like me referring to player, and if I reverse the ball and player classes, me referring to the ball class. So, I'm stuck again.

For some reason it doesn't like me