```
1
2   // Re-design this. See book!
3   // think of this: 1,000 pieces with 4-sides is 4,000 to this
    function?!?!?!?
4   var PuzzlePiece = function(game, x, y, j, i, width, height, pic, sides) {
5
6       this.game = game;
7
8       //assign variable for each side (clockwise)
9       this.top_side = sides.ts;
10      this.right_side = sides.rs;
11      this.bottom_side = sides.bs;
12      this.left_side = sides.ls;
13
14      //assign height and width per piec
15      this.i = i; //i height position
16      this.j = j; //j width position
17
18      //assign bitmap data width and height
19      var bmdwidth = Math.floor(width*1.30);
20      var bmdheight = Math.floor(height*1.30);
21
22      //Set the 4 Corners of the puzzle image
23      //   tl = top left
24      var tl = {x: Math.floor(width*0.15), y: Math.floor(height*0.15)};
25      //   tr = top right
26      var tr = {x: (width+Math.floor(width*0.15)), y: (Math.floor(height*
        0.15))};
27      //   bl = bottom left
28      var bl = {x: Math.floor(width*0.15), y: (height+Math.floor(height*0.15
        ))};
29      //   br = bottom right
30      var br = {x: (width+Math.floor(width*0.15)), y: (height+Math.floor(
        height*0.15))};
31
32      //Draw each individual Jigsaw Puzzle Piece as bitmap data (ppbmd)
33      this.ppbmd = this.game.add.bitmapData(bmdwidth,bmdheight);
34      this.ppbmd.ctx.clearRect(0, 0, bmdwidth, bmdheight);
35      this.ppbmd.ctx.strokeStyle = '#FFF';
36      this.ppbmd.ctx.fillStyle = '#dcdcdc';
37      this.ppbmd.ctx.lineWidth = 2;
38      this.ppbmd.ctx.fill();
39      this.ppbmd.ctx.beginPath();
40
41      if (this.left_side === 0) {
42          //left side flat (0)
43          this.ppbmd.ctx.moveTo(tl.x,tl.y);
44          this.ppbmd.ctx.lineTo(bl.x,bl.y);
45      }else if (this.left_side === -1){
46          //left side concaved
47          this.ppbmd.ctx.moveTo(tl.x,tl.y);
48          this.ppbmd.ctx.lineTo(tl.x,Math.floor(tl.y+height/3));
49          this.ppbmd.ctx.lineTo(Math.floor(tl.x+width*0.10),Math.floor(tl.y+
            height/3));
50          this.ppbmd.ctx.lineTo(Math.floor(tl.x+width*0.15),Math.floor(tl.y+
            height/3+height/6));
51          this.ppbmd.ctx.lineTo(Math.floor(tl.x+width*0.10),Math.floor(tl.y+
            height/3+2*height/6));
52          this.ppbmd.ctx.lineTo(tl.x,Math.floor(tl.y+height/3+2*height/6));
```

```
53              this.ppbmd.ctx.lineTo(bl.x,bl.y);
54          }else if (this.left_side === 1) {
55              //left side protuding
56              this.ppbmd.ctx.moveTo(tl.x,tl.y);
57              this.ppbmd.ctx.lineTo(tl.x,Math.floor(tl.y+height/3));
58              this.ppbmd.ctx.lineTo(Math.floor(tl.x-width*0.10),Math.floor(tl.y+
                height/3));
59              this.ppbmd.ctx.lineTo(Math.floor(tl.x-width*0.15),Math.floor(tl.y+
                height/3+height/6));
60              this.ppbmd.ctx.lineTo(Math.floor(tl.x-width*0.10),Math.floor(tl.y+
                height/3+2*height/6));
61              this.ppbmd.ctx.lineTo(tl.x,Math.floor(tl.y+height/3+2*height/6));
62              this.ppbmd.ctx.lineTo(bl.x,bl.y);
63          }
64
65          if (this.bottom_side === 0) {
66              //bottom side flat
67              this.ppbmd.ctx.lineTo(br.x, br.y);
68          }else if (this.bottom_side === -1) {
69              //bottom side concaved
70              this.ppbmd.ctx.lineTo(Math.floor(bl.x+width/3), bl.y);
71              this.ppbmd.ctx.lineTo(Math.floor(bl.x+width/3), Math.floor(bl.y-
                width*0.10));
72              this.ppbmd.ctx.lineTo(Math.floor(bl.x+width/3+width/6), Math.floor
                (bl.y-width*0.15));
73              this.ppbmd.ctx.lineTo(Math.floor(bl.x+width/3+2*width/6), Math.
                floor(bl.y-width*0.10));
74              this.ppbmd.ctx.lineTo(Math.floor(bl.x+width/3+2*width/6), bl.y);
75              this.ppbmd.ctx.lineTo(br.x, bl.y);
76          }else if (this.bottom_side === 1) {
77              //bottom side protuding
78              this.ppbmd.ctx.lineTo(Math.floor(bl.x+width/3), bl.y);
79              this.ppbmd.ctx.lineTo(Math.floor(bl.x+width/3), Math.floor(bl.y+
                width*0.10));
80              this.ppbmd.ctx.lineTo(Math.floor(bl.x+width/3+width/6), Math.floor
                (bl.y+width*0.15));
81              this.ppbmd.ctx.lineTo(Math.floor(bl.x+width/3+2*width/6), Math.
                floor(bl.y+width*0.10));
82              this.ppbmd.ctx.lineTo(Math.floor(bl.x+width/3+2*width/6), bl.y);
83              this.ppbmd.ctx.lineTo(br.x, br.y);
84          }
85
86          if (this.right_side === 0) {
87              //right side flat
88              this.ppbmd.ctx.lineTo(tr.x, tr.y);
89          }else if (this.right_side === -1) {
90              //right side concaved
91              this.ppbmd.ctx.lineTo(br.x,Math.floor(br.y-height/3));
92              this.ppbmd.ctx.lineTo(Math.floor(br.x-width*0.10),Math.floor(br.y-
                height/3));
93              this.ppbmd.ctx.lineTo(Math.floor(br.x-width*0.15),Math.floor(br.y-
                height/3-height/6));
94              this.ppbmd.ctx.lineTo(Math.floor(br.x-width*0.10),Math.floor(br.y-
                height/3-2*height/6));
95              this.ppbmd.ctx.lineTo(br.x,Math.floor(br.y-height/3-2*height/6));
96              this.ppbmd.ctx.lineTo(tr.x,tr.y);
97          }else if (this.right_side === 1) {
98              //right side protuding
99              this.ppbmd.ctx.lineTo(br.x,Math.floor(br.y-height/3));
```

```
100             this.ppbmd.ctx.lineTo(Math.floor(br.x+width*0.10),Math.floor(br.y-
                height/3));
101             this.ppbmd.ctx.lineTo(Math.floor(br.x+width*0.15),Math.floor(br.y-
                height/3-height/6));
102             this.ppbmd.ctx.lineTo(Math.floor(br.x+width*0.10),Math.floor(br.y-
                height/3-2*height/6));
103             this.ppbmd.ctx.lineTo(br.x,Math.floor(br.y-height/3-2*height/6));
104             this.ppbmd.ctx.lineTo(tr.x,tr.y);
105         }
106
107         if (this.top_side === 0) {
108             //top side flat
109             this.ppbmd.ctx.lineTo(tl.x, tl.y);
110         }else if (this.top_side === -1) {
111             //top side concaved
112             this.ppbmd.ctx.lineTo(Math.floor(tr.x-width/3), tr.y);
113             this.ppbmd.ctx.lineTo(Math.floor(tr.x-width/3), Math.floor(tr.y+
                width*0.10));
114             this.ppbmd.ctx.lineTo(Math.floor(tr.x-width/3-width/6), Math.floor
                (tr.y+width*0.15));
115             this.ppbmd.ctx.lineTo(Math.floor(tr.x-width/3-2*width/6), Math.
                floor(tr.y+width*0.10));
116             this.ppbmd.ctx.lineTo(Math.floor(tr.x-width/3-2*width/6), tr.y);
117             this.ppbmd.ctx.lineTo(tl.x, tl.y);
118         }else if (this.top_side === 1) {
119             //top side protuding
120             this.ppbmd.ctx.lineTo(Math.floor(tr.x-width/3), tr.y);
121             this.ppbmd.ctx.lineTo(Math.floor(tr.x-width/3), Math.floor(tr.y-
                width*0.10));
122             this.ppbmd.ctx.lineTo(Math.floor(tr.x-width/3-width/6), Math.floor
                (tr.y-width*0.15));
123             this.ppbmd.ctx.lineTo(Math.floor(tr.x-width/3-2*width/6), Math.
                floor(tr.y-width*0.10));
124             this.ppbmd.ctx.lineTo(Math.floor(tr.x-width/3-2*width/6), tr.y);
125             this.ppbmd.ctx.lineTo(tl.x, tl.y);
126         }
127
128         this.ppbmd.ctx.fill();
129
130         var src_image = this.game.add.image(GAMEAPP.viewportWidth/2, GAMEAPP.
            viewportHeight/2, pic);
131         src_image.anchor.setTo(0.5);
132         src_image.visible = false;
133
134         var w = src_image.width;
135         var h = src_image.height;
136
137         var offsetX = Math.floor(width*0.15);
138         var offsetY = Math.floor(height*0.15);
139
140         var padX = Math.floor(width*0.15);
141         var padY = Math.floor(height*0.15);
142
143         if (this.top_side === 1) { padY = 0; }
144         if (this.left_side === 1) { padX = 0; }
145
146         var img = this.game.make.bitmapData(w, h);
147         area = new Phaser.Rectangle(j*width-(Math.abs(padX- width*0.15)), i*
            height-(Math.abs(padY - height*0.15)), w, h);
```

```
148         img.copyRect(pic, area, padX,padY);
149         img.update();
150 
151         var mask = this.game.make.bitmapData(bmdwidth, bmdheight);
152         mask.copyRect(this.ppbmd, area, bmdwidth, bmdheight);
153 
154         var bmd = this.game.make.bitmapData(bmdwidth, bmdheight);
155         bmd.alphaMask(img, this.ppbmd);
156 
157         Phaser.Sprite.call(this, this.game, x-offsetX, y-offsetY, bmd);
158 
159         this.initialX = this.x;
160         this.initialY = this.y;
161 
162         this.inputEnabled = true;
163         this.input.enableDrag(true);
164 
165         this.game.add.existing(this);
166 
167     };
168     // Phaser v2.x.x replaced with Phaser III
169     // forcing OO onto JS; use JS-OLOO delegation instead.
170     PuzzlePiece.prototype = Object.create(Phaser.Sprite.prototype);
171     PuzzlePiece.prototype.constructor = PuzzlePiece;
172 
173 
```