Canvas + Box2D

codelab instructions

1: setup a canvas

HTML:

```html
   <!-- width/height styles: appearance
        width/height attributes: coordinate space -->
   <canvas id="canvas" width="800" height="600"
           style="width: 800px; height: 600px" />
```

JS glue code:

```javascript
document.addEventListener('DOMContentLoaded', function() {

   var canvasElement = document.getElementById('canvas');
   var ctx = canvasElement.getContext("2d");
   ctx.clearRect(0, 0, ctx.canvas.width, ctx.canvas.height);
   // start drawing...
});
```

2: draw lines and shapes

```
// Drawing styles
ctx.save();  // ctx.restore() to undo styles
ctx.strokeStyle = "red";
ctx.fillStyle = "blue";
ctx.lineWidth = 2;

// Drawing primitives
ctx.beginPath(); // start new path
ctx.moveTo(x, y);
ctx.lineTo(x, y);
ctx.rect(x1, y1, w, h);
ctx.arc(x, y, r, bAng, eAng, dir);
ctx.bezierCurveTo(vx, vy, wx, wy, x, y);
// optionnally, ctx.endPath() to close the path

ctx.stroke(); // stroke the above path
ctx.fill();   // fill the above path
```

3: draw an image

```
var img = new Image();
img.onload = function()
{
  // drawImage, but you have to
  // pay attention to loading time
  ctx.drawImage(img, x, y, w, h);
}
img.src = "images/tile_compuser.png";
```

3: geometric transforms

Use *drawGuy()* and *drawTarget()*. Add transforms to display the little guy in the circle, standing on its nose.

```
drawTarget(ctx);
// transforms in reverse order of "manual" placement
ctx.translate(x, y);
ctx.rotate(angRadians);
ctx.scale(sx, sy);
ctx.translate(-w/2, -h/2);
drawGuy(ctx);
```

4: setup Box2D

Use *createWorldWithGravity()* and *createBox()* helper functions (please have a look at what they do). For now, display with Box2D's debug draw.

Warning: we must use a scaling factor between screen coordinates and Box2D world coordinates to keep the physics computations stable (positions between -10 and 10). See *createBox and setupDebugDraw*.

```
setupDebugDraw(world, ctx);

var world = createWorldWithGravity();
var body = createBox(world, cx, cy, w, h); // uses scale
body.SetAngle(/*radians*/);   /* warning: (cx,cy) = box center */

world.DrawDebugData(); // also uses scale
```

5: add animation loops

```
function runWorld()
{
   // Box2D simulation step at 60 fps, 10 iterations for solvers
   world.Step(1/60, 10, 10); // seconds
   world.ClearForces(); // Box2D specific

   setTimeout(runWorld, 1000/60); // loop at 60 fps (milliseconds)
}


function runAnimation()
{
   ctx.clearRect(0,0,ctx.canvas.width, ctx.canvas.height); // erase
   world.DrawDebugData(); // draw
   requestAnimationFrame(runAnimation); // let the browser decide fps
}
```

6: add a ground, walls, ...

The *createBox* helper can also create fixed objects.

```
// world, centerX, centerY, width, height, fixed)
createBox(world, 450, 570, 900, 60, true);
createBox(world, 30, 300, 60, 600, true);
createBox(world, 770, 300, 60, 600, true);
```

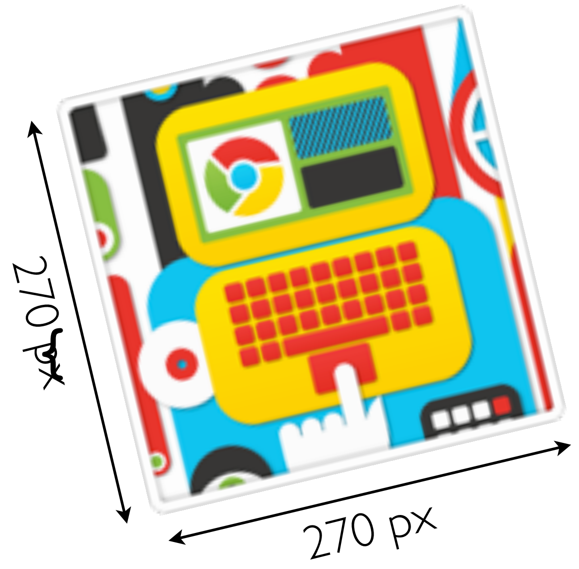
7: add real display

Instead of Box2D’s debug draw use /images/tile_*.png

dirty trick: add image to b2Body objects retuned by *createBox*

```
var body = createBox(world, 400, 10, 270, 270);
body.image = new Image();
body.image.src = "images/tile_bberry.png"; // custom image
body.image.onload = function() {draw(world, ctx);}

function draw(world, ctx){
   if (ctx !== undefined && world !== undefined)
      for (var b = world.GetBodyList(); b; b = b.GetNext()) {
         if (b.image !== undefined) {
            ctx.save();
            var pos = b.GetPosition(); // center position
            ctx.translate(pos.x*scale, pos.y*scale);
            ctx.rotate(b.GetAngle());
            ctx.drawImage(b.image, -b.image.width/2, - b.image.height/2);
            ctx.restore();
         }
      }
```

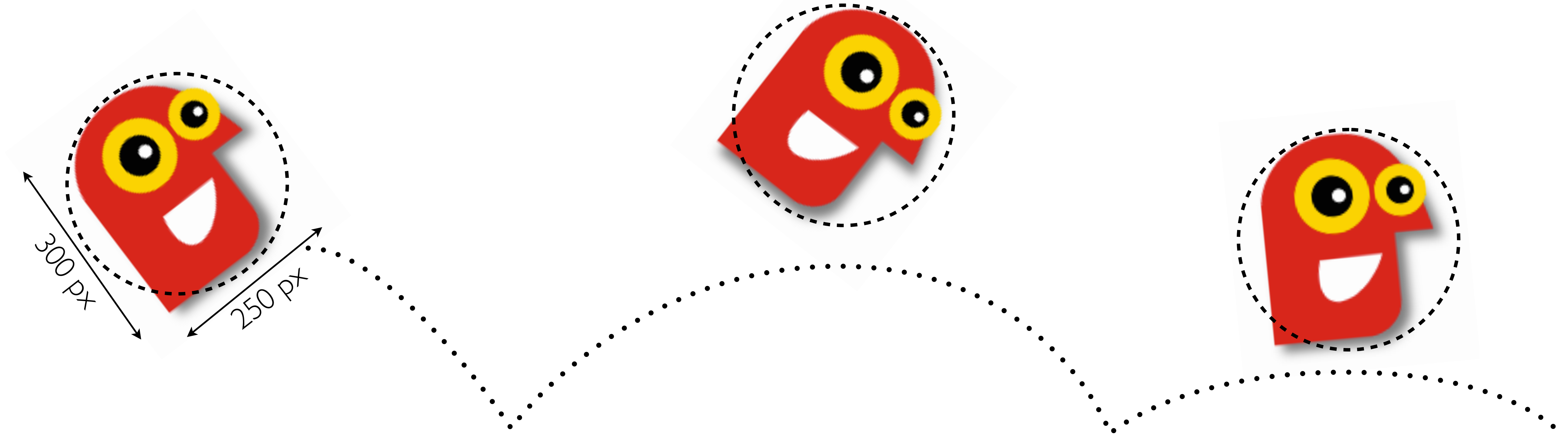


8: try with a ball

There is a *createBall* helper in the playground. See what it does.

```
createBall(world, r, x, y);
```

Bounce the *drawGuy(ctx)* around. To simplify, represent it with a ball shape in the Box2d world.



Go crazy!

Bonus: use the *sleepWorld*, *wakeWorld* and *isWorldAsleep* helpers to stop the animation and the simulation when nothing is moving.

html5rocks.com