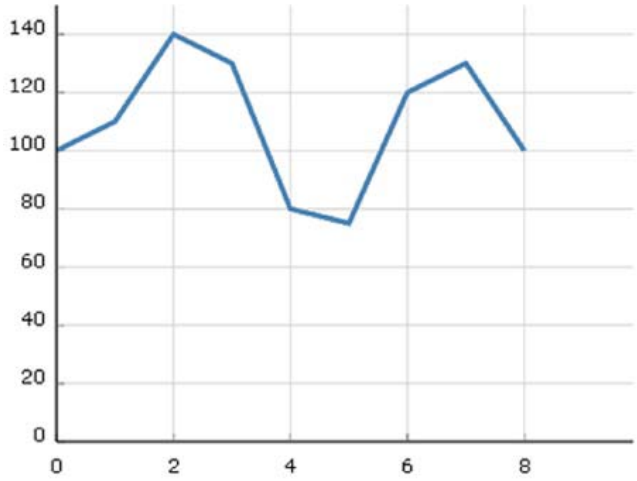

Figure 3-5. *A line chart with a grid covering the blue lines*

Look carefully at the Figure 3-5. The gray lines of the grid are drawn above the blue line representing the data. In other words, to be more explicit, you must be careful about the order in which you draw the SVG elements. In fact it is convenient to first draw the axes and the grid and then eventually to move on to the representation of the input data. Thus, you need to put all items that you want to draw in the right order, as shown in Listing 3-14.

Listing 3-14. ch3_01.html

```
<script>
var data = [100,110,140,130,80,75,120,130,100];
w = 400;
h = 300;
margin_x = 32;
margin_y = 20;
y = d3.scale.linear().domain([0, d3.max(data)]).range([0 + margin_y, h - margin_y]);
x = d3.scale.linear().domain([0, data.length]).range([0 + margin_x, w - margin_x]);
var svg = d3.select("body")
    .append("svg:svg")
    .attr("width", w)
    .attr("height", h);

var g = svg.append("svg:g")
    .attr("transform", "translate(0," + h + ")");

var line = d3.svg.line()
    .x(function(d,i) { return x(i); })
    .y(function(d) { return -y(d); });

// draw the y axis
g.append("svg:line")
    .attr("x1", x(0))
    .attr("y1", -y(0))
    .attr("x2", x(w))
    .attr("y2", -y(0));
```

```
// draw the x axis
g.append("svg:line")
    .attr("x1", x(0))
    .attr("y1", -y(0))
    .attr("x2", x(0))
    .attr("y2", -y(d3.max(data))-10);

//draw the xLabels
g.selectAll(".xLabel")
    .data(x.ticks(5))
    .enter().append("svg:text")
    .attr("class", "xLabel")
    .text(String)
    .attr("x", function(d) { return x(d) })
    .attr("y", 0)
    .attr("text-anchor", "middle");

// draw the yLabels
g.selectAll(".yLabel")
    .data(y.ticks(5))
    .enter().append("svg:text")
    .attr("class", "yLabel")
    .text(String)
    .attr("x", 25)
    .attr("y", function(d) { return -y(d) })
    .attr("text-anchor", "end");

//draw the x ticks
g.selectAll(".xTicks")
    .data(x.ticks(5))
    .enter().append("svg:line")
    .attr("class", "xTicks")
    .attr("x1", function(d) { return x(d); })
    .attr("y1", -y(0))
    .attr("x2", function(d) { return x(d); })
    .attr("y2", -y(0)-5);

// draw the y ticks
g.selectAll(".yTicks")
    .data(y.ticks(5))
    .enter().append("svg:line")
    .attr("class", "yTicks")
    .attr("y1", function(d) { return -1 * y(d); })
    .attr("x1", x(0)+5)
    .attr("y2", function(d) { return -1 * y(d); })
    .attr("x2", x(0));

//draw the x grid
g.selectAll(".xGrids")
    .data(x.ticks(5))
    .enter().append("svg:line")
    .attr("class", "xGrids")
    .attr("x1", function(d) { return x(d); })
```

```
    .attr("y1", -y(0))
    .attr("x2", function(d) { return x(d); })
    .attr("y2", -y(d3.max(data))-10);

// draw the y grid
g.selectAll(".yGrids")
    .data(y.ticks(5))
    .enter().append("svg:line")
    .attr("class", "yGrids")
    .attr("y1", function(d) { return -1 * y(d); })
    .attr("x1", x(w))
    .attr("y2", function(d) { return -y(d); })
    .attr("x2", x(0));

// draw the x axis
g.append("svg:line")
    .attr("x1", x(0))
    .attr("y1", -y(0))
    .attr("x2", x(w))
    .attr("y2", -y(0));

// draw the y axis
g.append("svg:line")
    .attr("x1", x(0))
    .attr("y1", -y(0))
    .attr("x2", x(0))
    .attr("y2", -y(d3.max(data))+10);

// draw the line of data points
g.append("svg:path").attr("d", line(data));
</script>
```

Figure 3-6 shows the line chart with the elements drawn in the correct sequence. In fact, the blue line representing the input data is now on the foreground covering the grid and not vice versa.

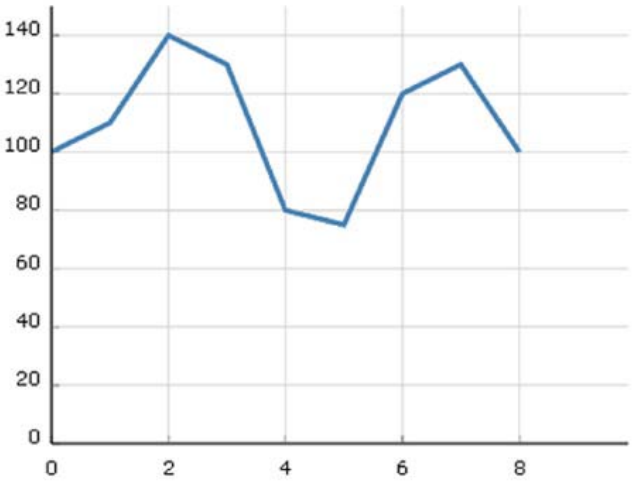

Figure 3-6. *A line chart with a grid drawn correctly*