

1 Correction du TD 9

Exercice 1

Utilisation d'OpenStreetMap pour visualiser des données :

```
#coding:latin-1
import sys
sys.path.append(".././.././program/python/pyensae/src") # ligne inutile

from pyensae import download_data
import pandas

download_data("td9_data.zip", website = 'xd')
file1 = "td9_full.txt"
tbl = pandas.read_csv (file1, sep = "\t")

from pandas.tools.plotting import scatter_plot

gr = tbl.groupby(['lng', 'lat'], as_index = False).agg(lambda x: len(x))

# voir http://dev.openlayers.org/docs/files/OpenLayers/Marker-js.html pour changer le marker
html = """
<html><body>
  <div id="mapdiv"></div>
  <script src="http://www.openlayers.org/api/OpenLayers.js"></script>
  <script>
    map = new OpenLayers.Map("mapdiv");
    map.addLayer(new OpenLayers.Layer.OSM());
    var proj = new OpenLayers.Projection("EPSG:4326");

    var zoom=13;

    var markers = new OpenLayers.Layer.Markers( "Markers" );
    map.addLayer(markers);

    __VELIB__

    map.setCenter (lonLat0, zoom);
  </script>
</body></html>
"""

position ="""
  var lonLat{0} = new OpenLayers.LonLat( {1} ,{2} ).transform(proj, map.getProjectionObject() );
  markers.addMarker(new OpenLayers.Marker(lonLat{0}));
"""

lines = [ ]
for i,row in enumerate(gr.values) :
  y = lat = row[1]
  x = lng = row[0]
  line = position.format(i,x,y)
  lines.append(line)

text = "\n".join( lines )
html = html.replace("__VELIB__", text)
```

```
with open("velib.html", "w") as f : f.write(html)
```

Les stations VÃlib dans les zones de travail :

```
#coding:latin-1
import sys
sys.path.append(".././.././././program/python/pyensae/src") # ligne inutile

from pyensae import download_data
import pandas

download_data("td9_data.zip", website = 'xd')
file1 = "td9_full.txt"
tbl = pandas.read_csv (file1, sep = "\t")

from pandas.tools.plotting import scatter_plot

gr = tbl.groupby(['lng', 'lat'], as_index = False).agg(lambda x: len(x))

# voir http://dev.openlayers.org/docs/files/OpenLayers/Marker-js.html pour changer le marker
html = ""
<html><body>
  <div id="mapdiv"></div>
  <script src="http://www.openlayers.org/api/OpenLayers.js"></script>
  <script>
    map = new OpenLayers.Map("mapdiv");
    map.addLayer(new OpenLayers.Layer.OSM());
    var proj = new OpenLayers.Projection("EPSG:4326");

    var zoom=13;

    var markers = new OpenLayers.Layer.Markers( "Markers" );
    map.addLayer(markers);

    __VELIB__

    map.setCenter (lonLat0, zoom);
  </script>
</body></html>
"""

position =""
  var lonLat{0} = new OpenLayers.LonLat( {1} ,{2} ).transform(proj, map.getProjectionObject() );
  markers.addMarker(new OpenLayers.Marker(lonLat{0}));
"""

lines = [ ]
for i,row in enumerate(gr.values) :
  y = lat = row[1]
  x = lng = row[0]
  line = position.format(i,x,y)
  lines.append(line)

text = "\n".join( lines )
html = html.replace("__VELIB__", text)
```

```
with open("velib.html", "w") as f : f.write(html)
```

Seconde demi-heure : d3.js¹

Exercice 2

```
ï¿½<!DOCTYPE html>
<meta charset="utf-8">
<script src="http://d3js.org/d3.v2.js?2.9.1"></script>
<script src="td9_graph_lworld.js"></script>
<style>

.link {
  fill: none;
  stroke: #666;
  stroke-width: 1.5px;
}

.node circle {
  fill: #ccc;
  stroke: #fff;
  stroke-width: 1.5px;
}

text {
  font: 10px sans-serif;
  pointer-events: none;
}

</style>
<body>
<script>

var nodes = {};

// Compute the distinct nodes from the links.
links.forEach(function(link) {
  link.source = nodes[link.source] || (nodes[link.source] = {name: link.source});
  link.target = nodes[link.target] || (nodes[link.target] = {name: link.target});
});

var width = 960,
    height = 500;

var force = d3.layout.force()
  .nodes(d3.values(nodes))
  .links(links)
  .size([width, height])
  .linkDistance(60)
  .charge(-300)
```

1. <http://d3js.org/>

```

        .on("tick", tick)
        .start();

var svg = d3.select("body").append("svg")
    .attr("width", width)
    .attr("height", height);

var link = svg.selectAll(".link")
    .data(force.links())
    .enter().append("line")
    .attr("class", "link");

var node = svg.selectAll(".node")
    .data(force.nodes())
    .enter().append("g")
    .attr("class", "node")
    .call(force.drag);

node.append("circle")
    .attr("r", 8);

node.append("text")
    .attr("x", 12)
    .attr("dy", ".35em")
    .text(function(d) { return d.name; });

function tick() {
    link
        .attr("x1", function(d) { return d.source.x; })
        .attr("y1", function(d) { return d.source.y; })
        .attr("x2", function(d) { return d.target.x; })
        .attr("y2", function(d) { return d.target.y; });

    node
        .attr("transform", function(d) { return "translate(" + d.x + "," + d.y + ")"; });
}
</script>

```

Troisième demi-heure : Un graphique avec des Animations

Exercice 2

```

<!DOCTYPE html>
<meta charset="utf-8">
<script src="http://d3js.org/d3.v2.js?2.9.1"></script>
<script src="td9_graph_lworld.js"></script>
<style>

.link {
    fill: none;
    stroke: #666;
    stroke-width: 1.5px;
}

```

```

.node circle {
  fill: #ccc;
  stroke: #fff;
  stroke-width: 1.5px;
}

text {
  font: 10px sans-serif;
  pointer-events: none;
}

</style>
<body>
<script>

var nodes = {};

// Compute the distinct nodes from the links.
links.forEach(function(link) {
  link.source = nodes[link.source] || (nodes[link.source] = {name: link.source});
  link.target = nodes[link.target] || (nodes[link.target] = {name: link.target});
});

var width = 960,
    height = 800;

var force = d3.layout.force()
  .nodes(d3.values(nodes))
  .links(links)
  .size([width, height])
  .linkDistance(30)
  .charge(-300)
  .on("tick", tick)
  .start();

var svg = d3.select("body").append("svg")
  .attr("width", width)
  .attr("height", height);

var link = svg.selectAll(".link")
  .data(force.links())
  .enter().append("line")
  .attr("class", "link");

var node = svg.selectAll(".node")
  .data(force.nodes())
  .enter().append("g")
  .attr("class", "node")
  .on("mouseover", mouseover) // on ajoute deux événements: la souris passe au-dessus d'un noeud
  .on("mouseout", mouseout) // la sortie sort de la zone du noeud
  .call(force.drag); // pour qu'on puisse tirer les noeuds

node.append("circle")
  .attr("r", 8);

node.append("text")
  .attr("x", 12)
  .attr("dy", ".35em")
  .text(function(d) { return d.name; })

```

```

        .style("font-size", "10px")
    ;

function tick() {
    link
        .attr("x1", function(d) { return d.source.x; })
        .attr("y1", function(d) { return d.source.y; })
        .attr("x2", function(d) { return d.target.x; })
        .attr("y2", function(d) { return d.target.y; });

    node
        .attr("transform", function(d) { return "translate(" + d.x + "," + d.y + ")"; });
}

// la fonction suivante définit ce qui doit se passer au cas où le curseur passe au-dessus d'un noeud
function mouseover() {
    d3.select(this)
        .select("text")
            .style("font-size", "30px")
            .text(function(d,i){return "** " + d.name + " **" ;});

    d3.select(this)
        .style("fill", "red")
        .select("circle").transition()
            .duration(750)
            .attr("r", 16)
            .style("fill", "yellow");
}

// la fonction suivante définit ce qui doit se passer au cas où le curseur sort de la zone du noeud
function mouseout() {
    d3.select(this)
        .select("text")
            .style("font-size", "10px")
            .text(function(d,i){return d.name ;});

    d3.select(this)
        .style("fill", "grey")
        .select("circle").transition()
            .duration(350)
            .attr("r", 8)
            .style("fill", "grey");
}
</script>

```

Quatrième demi-heure : Zoomer

La correction est en deux parties :

```

<!DOCTYPE html>
<!-- http://jsfiddle.net/KSAbK/1/ -->
<!-- https://leanpub.com/D3-Tips-and-Tricks/read -->
<meta charset="utf-8">
<style>

```

```

body {
  font: 10px sans-serif;
}
.plot {
  fill: rgba(250, 250, 255, 0.6);
}
.grid .tick {
  stroke: lightgrey;
  opacity: 0.7;
}
.grid path {
  stroke-width: 0;
}
.axis path, .axis line {
  fill: none;
  stroke: #000;
  shape-rendering: crispEdges;
}
.x.axis path {
  display: none;
}
.line {
  fill: none;
  stroke: steelblue;
  stroke-width: 1.5px;
}

</style>
<body>
<script src="http://d3js.org/d3.v3.js"></script>
<script src="td9_by_hours_data.js"></script> <!-- the data, contains the variable data used by the following script -->
<!--
data =
[
{'velo': 12817.0, 'last_update': '10/9/13 11:35', 'minute': 35.0, 'heure': 11.0, 'place': 25202.0, 'name': '10/9/13 11:35'},
{'velo': 12744.0, 'last_update': '10/9/13 11:40', 'minute': 40.0, 'heure': 11.0, 'place': 25273.0, 'name': '10/9/13 11:40'},
...
] ;
-->
<script>

// defines the graph area (usually the same for for every graph)
margin = {
  top: 20,
  right: 20,
  bottom: 20,
  left: 45
};

width = 800 - margin.left - margin.right;
height = 500 - margin.top - margin.bottom;

// defines the range of each axis
var x = d3.time.scale()
  .domain(d3.extent(data, function (d) {
    return d.last_update;
  })))
  .range([0, width]);

```

```

var y = d3.scale.linear()
    .domain(d3.extent(data, function (d) {
        return d.velo;
    }))
    .range([height, 0]);

// graph type, also defines the columns to be used (last_update and velo in this case)
var line = d3.svg.line()
    .x(function (d) {
        return x(d.last_update);
    })
    .y(function (d) {
        return y(d.velo);
    });

// defines the function to call when zooming
var zoom = d3.behavior.zoom()
    .x(x)
    // .y(y)
    .on("zoom", zoomed);

// creates a svg section in the body section
svg = d3.select('body')
    .append("svg")
    .attr('width', width + margin.left + margin.right)
    .attr('height', height + margin.top + margin.bottom)
    .append("g")
    .attr("transform", "translate(" + margin.left + "," + margin.top + ")")
    .call(zoom);

svg.append("rect")
    .attr("width", width)
    .attr("height", height)
    .attr("class", "plot");

// two functions uses for the zoom
var make_x_axis = function () {
    return d3.svg.axis()
        .scale(x)
        .orient("bottom")
        .ticks(5);
};

var make_y_axis = function () {
    return d3.svg.axis()
        .scale(y)
        .orient("left")
        .ticks(10);
};

// defines the axis
var xAxis = d3.svg.axis()
    .scale(x)
    .orient("bottom")
    .ticks(10);

svg.append("g")
    .attr("class", "x axis")

```



```

    .attr("transform", "translate(0, " + height + ")")
    .call(xAxis);

var yAxis = d3.svg.axis()
    .scale(y)
    .orient("left")
    .ticks(10);

svg.append("g")
    .attr("class", "y axis")
    .call(yAxis);

svg.append("g")
    .attr("class", "x grid")
    .attr("transform", "translate(0," + height + ")")
    .call(make_x_axis())
    .tickSize(-height, 0, 0)
    .tickFormat("");

svg.append("g")
    .attr("class", "y grid")
    .call(make_y_axis())
    .tickSize(-width, 0, 0)
    .tickFormat("");

// objects for the zooming
var clip = svg.append("clipPath")
    .attr("id", "clip")
    .append("rect")
    .attr("x", 0)
    .attr("y", 0)
    .attr("width", width)
    .attr("height", height);

var chartBody = svg.append("g")
    .attr("clip-path", "url(#clip)");

chartBody.append("path")
    .datum(data)
    .attr("class", "line")
    .attr("d", line);

// zooming functions
function zoomed() {
    //console.log(d3.event.translate); // display information in the logging console of the browser (using dev tools)
    //console.log(d3.event.scale);
    svg.select(".x.axis").call(xAxis);
    svg.select(".y.axis").call(yAxis);
    svg.select(".x.grid")
        .call(make_x_axis())
        .tickSize(-height, 0, 0)
        .tickFormat("");
    svg.select(".y.grid")
        .call(make_y_axis())
        .tickSize(-width, 0, 0)
        .tickFormat("");
    svg.select(".line")
        .attr("class", "line")
        .attr("d", line);
}

```

```
}  
</script>  
</body>  
</html>
```

Un extrait des données contenant dans le fichier `td9_by_hours_data.js` :

```
data =  
[  
{'velo': 12817.0, 'last_update': '10/9/13 11:35', 'minute': 35.0, 'heure': 11.0, 'place': 25202.0, 'name': '10/9/13 11:35'},  
{'velo': 12744.0, 'last_update': '10/9/13 11:40', 'minute': 40.0, 'heure': 11.0, 'place': 25273.0, 'name': '10/9/13 11:40'},  
{'velo': 12674.0, 'last_update': '10/9/13 11:45', 'minute': 45.0, 'heure': 11.0, 'place': 25349.0, 'name': '10/9/13 11:45'},  
{'velo': 12646.0, 'last_update': '10/9/13 11:50', 'minute': 50.0, 'heure': 11.0, 'place': 25380.0, 'name': '10/9/13 11:50'},  
{'velo': 12568.0, 'last_update': '10/9/13 11:55', 'minute': 55.0, 'heure': 11.0, 'place': 25457.0, 'name': '10/9/13 11:55'},  
// ...  
]  
;  
  
var parseDate = d3.time.format("%d/%m/%Y %H:%M").parse;  
  
for (var i = 0; i < data.length; i++) {  
  element = data[i] ;  
  element['last_update'] = parseDate(element['last_update']) ;  
}
```

fin correction TD ?? □