**Bài tập về nhà 3**

**Due date: 28/11/2020, 22:00**

**Mô tả**

Bài tập số 3 yêu cầu lập trình phần mềm trực quan hóa có tương tác, đặc biệt yêu cầu các anh/chị cài đặt kỹ thuật tương tác “truy vấn động” (dynamic queries), tương tự như phần mềm HomeFinder (trong bài giảng tương tác). Tuy nhiên, thay vì tìm nhà thì chúng ta tìm dữ liệu về các nhà hàng tại thành phố San Francisco.

**Yêu cầu**

Dữ liệu có thông tin vị trí (kinh độ/vĩ độ) của từng nhà hàng, cùng với một số thuộc tính mô tả khác. Mục tiêu của bài tập là trình bày những điểm dữ liệu nhà hàng này lên bản đồ thành phố San Francisco và cung cấp các chức năng truy vấn động như sau:

* Cho phép người dùng chỉ định 2 điểm: A (vị trí quan tâm), và B (vị trí quan tâm khác) cũng như bán kính cho từng điểm (xuất phát từ tâm là A và B). Và sau đó lọc các điểm dữ liệu, chỉ giữ lại các điểm nằm trong vùng giao của vòng tròn tâm A và vòng tròn tâm B.
* Cung cấp ít nhất 3 điều kiện lọc bổ sung cho phép người dùng lọc dữ liệu ở những khía cạnh cụ thể, ví dụ như giới hạn các điểm dữ liệu theo risk\_category, ngày hoặc tháng kiểm tra, điểm kiểm tra, v.v…

Các anh/chị có thể sử dụng bất kỳ ngôn ngữ lập trình nào, bất kỳ toolkits nào để lập trình cho bài tập này. Nhưng đề nghị nên sử dụng Javascript và D3. Cho dù các anh/chị sử dụng ngôn ngữ nào, bộ thư viện nào, cũng cần chú ý nộp chương trình thực thi sao cho giáo viên có thể run trên máy Mac OS X hoặc Windows.

**Tài nguyên**

* Dữ liệu nhà hàng tại thành phố San Francisco trong file restaurant\_scores.csv
* Bản đồ SVG của thành phố San Francisco: sf-map.svg
* Mã nguồn D3 cần để thiết lập cho bản đồ SVG

// Set up size

var width = 750,

height = width;

// Set up projection that map is using

var projection = d3.geoMercator()

.center([-122.433701, 37.767683])

// San Francisco, roughly

.scale(225000)

.translate([width / 2, height / 2]);

// This is the mapping between <longitude, latitude> position to <x, y> pixel position on the map

projection([lon, lat]) returns [x, y]

projection.invert([x, y]) returns [lon, lat]

// Add an svg element to the DOM

var svg = d3.select("body").append("svg")

.attr("width", width)

.attr("height", height);

// Add svg map at correct size, assumes map is saved in a subdirectory called "data"

svg.append("image")

.attr("width", width)

.attr("height", height)

.attr("xlink:href", "data/sf-map.svg");

* Mã nguồn vẽ 1 điểm dữ liệu cho trước kinh độ, vĩ độ

Var projectedLocation = projection([business\_longitude, business\_latitude]);

var circle = svg.append('circle')

.attr('cx', projectedLocation[0])

.attr('cy', projectedLocaiton[1])

.attr('r', 1);

* D3 cung cấp 1 giao tiếp đơn giản để lắng nghe sự kiện xãy ra: phương thức on cho bất kỳ chọn lựa nào. Ví dụ lắng nghe sự kiện nhấp chuột vào vòng tròn và trả về dữ liệu tương ứng, mã nguồn như sau:

d3.selectAll('circle')

.on('click', function(d) { console.log(d); });

* Mã nguồn định nghĩa dữ liệu: trong JavaScript, các yêu cầu HTTP được xử lý không đồng bộ. Khi gọi d3.csv, trình duyệt bắt đầu thực hiện yêu cầu HTTP truy xuất tới tài nguyên dữ liệu, do đó cần phải thực hiện đoạn mã nguồn sau:

// In D3 v5, the csv function uses Promises instead of asynchronus callbacks (v4) to load data

d3.csv("file.csv").then(function(data) {

console.log(data);

});

// This code is going to run before data is loaded, and you cannot use the data here

console.log('We don't have the data yet.');

nonDataRelatedStuff();

// This will print:

// => We don't have the data yet.

// => We have the data now!

**Công cụ khác**

* Protovis : http://mbostock.github.io/protovis/
* Prefuse : http://prefuse.org/
* InfoVis Toolkit : http://ivtk.sourceforge.net/
* Piccolo : http://www.cs.umd.edu/hcil/piccolo/

Hết ☺.