Institut Galilée
INFO2

Projet web sémantique

Objectifs :

Le but de ce projet est de créer une application Mushup dont le contenu provient de plusieurs sources d'informations :

Google Maps (API Google Maps), Youtube (API Youtube)

Deux bases de connaissances : Dbpedia, Freebase. Vous pouvez utiliser les SPARQL endpoint. Pour freebase, il ya une API qui permet un accées direct sans faire des requêtes SPARQL.

Mise en place :

Tout d'abord, suivez ces étapes:

- créez une application dans la console des développeurs Google: http://cloud.google.com/console/project
- Ensuite, dans l'onglet APIs & auth -> APIs, activez les API nécessaires au fur et à mesure de l'avancée de votre projet.
- Finalement dans l'onglet APIs & auth -> Credentials, on voit les différentes clés à utiliser pour le projet. Celle qui nous intéresse est la clé pour un accès publique.
- Dans "Edit allowed referers", on peut indiquer le site d'où les appels à l'API seront autorisés (évite les abus si quelqu'un vous vole votre clé). Renseignez ce champs si votre site est en ligne.

Votre Mission :

vous travaillez en binôme, Vous rendrez votre code ainsi qu'un rapport dans lequel vous détaillez le fonctionnement de votre application.

L'objectif est de créer une application de visualisation des tremblements de terre qui ont eu lieu dans un certain laps de temps, et avec une certaine magnitude. On demande les fonctionnalités suivantes: (qu'on peut changer à l'aide d'un formulaire html). Le projet comporte des fonctionnalités suivantes :

- L'emplacement des tremblements de terre est visualisé sur une carte Google Maps.
- Les marqueurs sont des cercles dont la taille dépend de la magnitude du séisme
- L'utilisateur peut choisir une fourchette de magnitude, et une fourchette de date

(difficile, bonus) comme si il contrôlait la lecture d'un film, l'utilisateur peut cliquer sur un bouton play, et voir les marqueurs apparaître au fur et à mesure. L'utilisateur peut contrôler la vitesse de lecture, et doit pouvoir visualiser la date et l'heure au dessus de la carte.

Lorsque l'utilisateur clique sur un marqueur, une info-bulle s'ouvre et donne les caractéristiques principales du tremblement de terre.

- Sous la carte Google Maps, l'utilisateur peut voir une liste de video youtube qui sont en lien avec l'emplacement du tremblement de terre sélectionné
    - vous utiliserez en particulier la méthode *search.list* de l'API youtube, et (au mieux) lecteurs video youtube entre la carte Google Maps et la liste de vidéo, on affiche:
    - quelques caractéristiques et une description succinte du lieu,
    - si le lieu est un pays, et pour les 10 premières personnalités connues qui sont nées à cet endroit, vous affichez une description succinte de cette personne.
    - vous utiliserez Freebase, en particulier les méthodes *search* et *topic* de l'API Google Freebase
    - voyez par exemple ce que Freebase sait sur: la nouvelle calédonie: http://www.freebase.com/m/05c17

Les bases de connaissances :

Freebase :

Freebase est une base de connaissances du web sémantique,c.f.,http://www.freebase.com/.

L'idée de base est qu'on peut associer une URI à tout et n'importe quoi. Par exemple, on a décidé que <http://www.freebase.com/m/0bkf4> représente Bob Marley, et qu'il s'agit d'un <http://www.freebase.com/music/artist>.

Freebase, a deux avantages:

- Il est accessible via l'API Google que vous connaissez bien;
- Pas besoin d'apprendre SPARQL, ni les recommandations du W3C. On n'utilisera que le format JSON, que vous connaissez bien.

DBpedia

DBpedia est la version RDF de wikipedia. L'équipe Wimmics à l'Inria a développé la version française (basée sur Wikipedia France) : http://fr.dbpedia.org/.

Source de documentation :

- a source de données: http://earthquake.usgs.gov/earthquak...month.geojsonp
- Description des sources de données: http://earthquake.usgs.gov/earthquak....0/geojson.php
- pour commencer avec Google maps: https://developers.google.com/maps/d...cript/tutorial
- La base de votre application: https://developers.google.com/maps/t...ng/earthquakes

- des exemples de code avec Google maps (les info windows vous intéressent): https://developers.google.com/maps/d...ript/examples/
- l'API youtube: https://developers.google.com/youtube/v3/
- des exemples de code avec Youtube API: https://developers.google.com/youtube/v3/code_samples
- l'explorateur d'API Google: http://developers.google.com/apis-explorer/#p/
- l'API freebase: https://developers.google.com/freebase/

Exemple d'utilisation des API google avec le client javascript pour API google.

Voici un Minimal Working Example pour l'appel à l'API youtube en utilisant la librairie client javascript des API google:

view plain print

```
01. <html>
02.   <head>
03.     <script>
04.       // la méthode makeRequest est appelée après le chargement de la librairie youtube
05.       function makeRequest() {
06.       // on paramètre l'appel à l'API avec la méthode search.list, c.f., https://developers.google.com/youtube/v3/docs/search/list
07.     var request = gapi.client.youtube.search.list({
08.           q: "api youtube",
09.           part: 'snippet'
10.         });
11.
12.       // on effectue l'appel et on affiche les résultats dans la console
13.         request.execute(function(response) {
14.           console.log(response.result);
15.         });
16.       }
17.
18.       // la méthode load est appelée après le chargement de l'api javascript google
19.       function load() {
20.         gapi.client.setApiKey("LA CLE DE L'APPLICATION");
21.         gapi.client.load('youtube', 'v3', makeRequest);
22.       }
23.     </script>
24.     <script src="https://apis.google.com/js/client.js?onload=load"></script>
25.   </head>
26. </html>
```

Exemple d'utilisation de L'API freebase avec javascript

```
view plain  print  ?
01. <!DOCTYPE html>
02. <html>
03.   <head>
04.     <script src="http://code.jquery.com/jquery-1.10.2.min.js"></script>
05.   </head>
06.   <body>
07.     <script>
08.       var topic_id = '/m/0f8l9c';
09.       var service_url = 'https://www.googleapis.com/freebase/v1/topic';
10.       var params = {filter: "/location/location/people_born_here",
11.       limit:100};
12.       $.getJSON(service_url + topic_id + '?callback=?', params, function(topic) {
13.       console.log(topic);
14.       });
15.     </script>
16.   </body>
17. </html>
```

La notion de clôture de javascript :

Vous semblez avoir besoin de la notion de cloture en Javascript. Je me suis inspiré de la page http://tinyhippos.com/2010/07/05/clo...with-examples/

Une cloture est une fonction spéciale:

- elle se comporte comme un objet (on peut la balader comme un objet et l'appeler plus tard)
- elle **capture la valeur** de toutes les variables qui étaient **dans la portée** au moment où elle a été créée, et qui ont été **passées en paramètre**. La cloture peut alors accéder à **ces valeurs** de variables lorsqu'on l'appel, même si les variables ont eu de nouvelles valeurs depuis, ou ne sont plus à portée.

Exécutez les deux exemples suivant, et essayez de comprendre la différence.

Mauvais exemple pour le code avec les marqueurs de Google Maps

```
view plain  print  ?

<html>
  <head>
    <script>
    function addButtonsWithHandlersBad() {
      var i, button;
      for(i = 0; i < 5; i++) {
        button = document.createElement("button");
        button.innerText = "Button " + i;
        button.onclick = function() {
          console.log("You just clicked Button " + i);
        };
        document.body.appendChild(button);
      }
    }
    window.onload = addButtonsWithHandlersBad
    </script>
  </head>
  <body>
  Les boutons:
  </body>
</html>
```

Ici, la variable i est modifiée à chaque fois qu'on passe dans la fonction, et elle vaut donc 5 à la fin.

```
view plain  print  ?
01. <html>
02.   <head>
03.     <script>
04.
05.     function createClosure(x) {
06.       return function(){
07.         console.log("You just clicked Button " + x );
08.       };
09.     }
10.
11.     function addButtonsWithHandlersGood() {
12.       var i, button;
13.       for(i = 0; i < 5; i++) {
14.         button = document.createElement("button");
15.         button.innerText = "Button " + i;
16.         button.onclick = createClosure(i);
17.         document.body.appendChild(button);
18.       }
19.     }
20.     window.onload = addButtonsWithHandlersGood
21.     </script>
22.   </head>
23.   <body>
24.   Les boutons:
25.   </body>
26. </html>
```

ici, createClosure est une fonction qui renvoie une fonction (la fonction renvoyée est la cloture). Elle **capture** la valeur de i.

Dernier exemple (plus concis):

Pour faire plus concis, on peut utiliser une fonction anonyme qui renvoie une fonction comme ceci:

F

view plain print ?

```
<html>
  <head>
    <script>
    function addButtonsWithHandlersGood() {
      var i, button;
      for(i = 0; i < 5; i++) {
        button = document.createElement("button");
        button.innerText = "Button " + i;
        button.onclick = (function(x) {
          return function(){
            console.log("You just clicked Button " + x + " and i is equal to " + i);
          };
        }(i));
        document.body.appendChild(button);
      }
    }
    window.onload = addButtonsWithHandlersGood
    </script>
  </head>
  <body>
  Les boutons:
  </body>
</html>
```