

**M2 pro Sigma : Module 653
Cartographie et SIG interactifs en ligne**

**Séance 5 : Le WebMapping côté client
Introduction à la réalisation de cartes interactives
en utilisant les API Google Maps et Géoportail**

Webographie :

Comparatif de solutions, en français :

http://www.geowebsevice.org/index.php?option=com_content&task=view&id=43

Comparatif des API de cartographie en ligne (en anglais) :

<http://www.niallkennedy.com/blog/2007/08/map-api-comparison.html>

Site de l'API Google Maps :

<http://code.google.com/apis/maps/>

Blog officiel :

<http://googlemapsapi.blogspot.com>

Utilitaires divers :

<http://freegeographytools.com>

<http://www.nearby.org.uk/google.html>

Présentation

Depuis 2006, plusieurs grandes sociétés du web (Microsoft, Yahoo et Google) proposent un service de cartographie. Ce service consiste en des cartes routières et des photographies aériennes interactives, et interrogeables (adresses, trajets, équipements et services), cartes que l'on peut intégrer à des pages web en utilisant un accès par la programmation.

Ces services web permettent aussi de représenter ses propres données par-dessus les fonds servis, et d'utiliser certaines fonctions d'interactivité (réaction au clic et requêtes). Leur fonctionnement ne nécessite aucune installation logicielle sur le serveur Internet, l'intégration des cartes et leur interactivité étant réalisée par des fonctions javascript organisées en API (Application Programming Interface, interface d'accès à une application par la programmation). C'est donc le navigateur qui va exécuter les fonctions de cartographie, après avoir téléchargé un fichier javascript sur le serveur du fournisseur de cartes ou sur le serveur http de la page. (cf. schéma n°5).

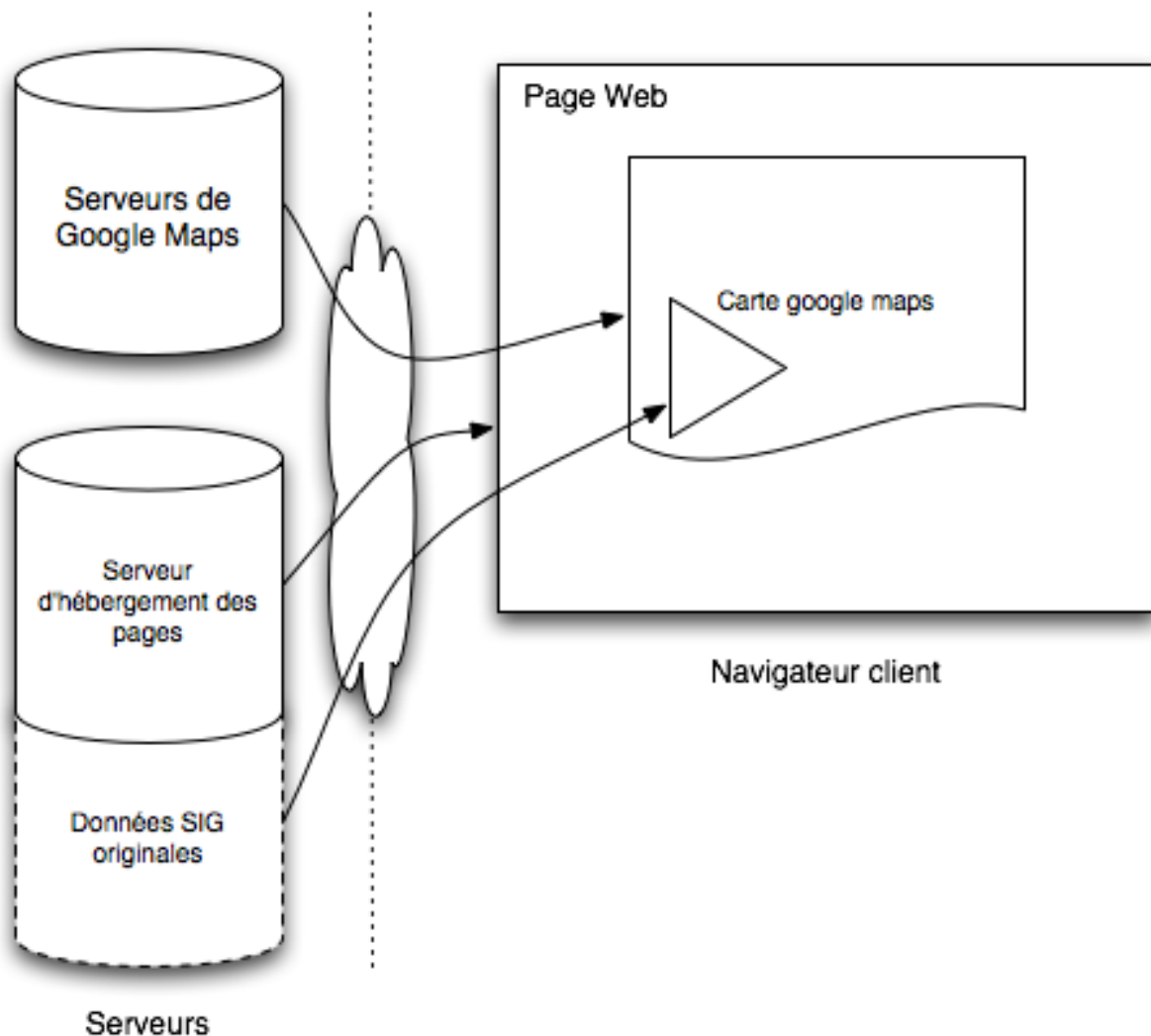
Il s'agit donc d'une offre couplant à la fois des données (des fonds de carte et des éléments localisés) et des outils de programmation pour les afficher et gérer les interactions avec l'utilisateur.

1- Aperçu des possibilités d'une API : Google Maps

Google Maps est au départ un site de visualisation de cartes et de recherches de lieux ou itinéraires. Mais la très grande quantité de données spatiales visualisables sur ce site est réutilisable de manière personnalisée sur un site Internet.

La licence d'utilisation de ces cartes est originale : on peut utiliser librement le service sur tout site Internet accessible directement, sans qu'il faille que les visiteurs s'enregistrent et/ou payent un abonnement. Ainsi, le contenu de Google Maps n'est pas utilisable en Intranet ou s'il fait partie d'un site réservé à des visiteurs abonnés. Dans ce cas il faut acquérir une licence d'utilisation de type « Entreprise ».

Enfin, pour pouvoir utiliser ce service, il faut posséder un compte google, et demander l'attribution d'une clé d'identification qui sera nécessaire de préciser dans les pages Web créées.



L'affichage de la carte est déclenché par un appel à une fonction JavaScript, du côté du navigateur, donc. Cet appel comporte des paramètres précisant la taille du rectangle de la carte, les coordonnées de son point central, et le type d'informations (les couches, choisies parmi celles proposées) à afficher.

Des données personnalisées peuvent ensuite être ajoutées à cette carte, sous la forme d'une couche qui viendrait se superposer, toujours en faisant appel à des fonctions JavaScript.

Exemple de réalisation simple

L'exemple est en ligne à cette adresse :

http://www.geotests.net/apis/google/carto_kml.html

Description du code HTML / Javascript

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<!-- Definition de la version du langage de la page en XHTML 1.0 strict -->
<head>
  <meta http-equiv="content-type" content="text/html; charset=utf-8"/>
  <title>Google Maps JavaScript API Example</title>
  <script src="http://maps.google.com/maps?file=api&v=2&key=CLE"
    type="text/javascript"></script>
  <!-- Chargement d'un fichier javascript externe, qui est l'API GMaps -->
</head>
<body onload="GUnload()">
<!-- Pour éviter les fuites mémoire, déchargement de l'API si on ferme la page -->
  <div id="map" style="width: 770px; height: 512px"></div>
  <!-- L'element de mise en page qui contiendra la carte -->
  <br><input type="button" value="hide" onclick="map.removeOverlay(kml)">
    <input type="button" value="show" onclick="map.addOverlay(kml)">
  <!-- Deux boutons pour afficher ou masquer le rectangle en kml ->
  <script type="text/javascript">
  //

  if (GBrowserIsCompatible()) {
    //L'objet map, et sera placé dans le div « map » de la page html.
    var map = new GMap2(document.getElementById("map"));
    //Boutons de zoom et de déplacement
    map.addControl(new GLargeMapControl());
    //Boutons de choix de type de carte
    map.addControl(new GMapTypeControl());
    //Centrage de la carte et choix de la vue satellite
    map.setCenter(new GLatLng(43.57782988, 1.402793039), 20, G_SATELLITE_MAP);
    //==== Couche de données en KML, stockée sur le serveur ====
    var kml = new GGeoXml("http://www.geotests.net/apis/google/carto.kml");
    map.addOverlay(kml);
  }

  //Affiche une information si le navigateur est incompatible.
  else {
    alert("Désolé, Google Maps n'est pas compatible avec votre navigateur.");
  }

  //]]&gt;
&lt;/script&gt;
&lt;/body&gt;

&lt;/html&gt;</pre></div><div data-bbox="113 854 889 889" data-label="Text"><p>La clé d'utilisation de l'API n'est pas nécessaire lorsque l'on travaille en local. Pour obtenir une clé pour pouvoir travailler sur un hébergement, rendez-vous sur la page suivante :</p></div><div data-bbox="113 887 577 905" data-label="Text"><p><a href="http://code.google.com/intl/fr-FR/apis/maps/signup.html">http://code.google.com/intl/fr-FR/apis/maps/signup.html</a></p></div>
```

Comment ajouter ses propres données a une carte Google Maps ?

On l'a vu dans l'exemple précédent, un des moyens possibles pour ajouter du contenu par-dessus une carte Google Maps est d'utiliser un fichier KML. Mais qu'est-ce que ce format ?

Comme son nom l'indique, il s'agit d'une variante du format XML, qui est, on le rappelle un format texte où les informations sont organisées par des balises descriptives et hiérarchisées.

Le format KML est en fait une évolution du format GML, pour Geographic Markup Language. Il permet de décrire des objets géographiques vectoriels divers, repérés spatialement par des coordonnées latitudes / longitudes décimales (Datum WGS84 par défaut, mais on peut définir une autre projection).

Ce format est exportable par OGR par exemple, ce qui permet de convertir (après avoir éventuellement reprojété) des fichiers SIG vectoriels dans les formats classiques. L'application Google Earth (globe virtuel) permet de dessiner des objets (points, linéaires, polygones, commentés), et de les exporter en KML. Pour plus de compacité, les informations décrivant les objets géographiques peuvent être décrites en binaire, ou le fichier KML peut être compressé avec l'algorithme usuel gzip pour devenir un fichier KMZ.

On peut trouver, sur les blogs de géomatique notamment, des utilitaires permettant de faciliter l'utilisation du format KML : la conversion de formats, la génération automatique de points à partir de relevés GPS, de marqueurs...

Utilisation d'une base de données pour stocker et servir les données KML à la demande.

Dans beaucoup de cas, la mise en place d'une solution de WebMapping est limitée par l'impossibilité d'installer de nouvelles applications sur la machine serveur. La plupart des hébergements bon marché se réduisent au triptyque Apache / PHP / MySQL. Dans ce contexte, l'utilisation d'une API JavaScript de cartographie pour fournir le fond de carte et l'interactivité, couplée à un stockage des données dans une base de données MySQL représente une solution intéressante.

Ainsi, on peut stocker dans une table les éléments d'une couche SIG au format KML, la mettre en forme et l'exporter sous la forme d'un *fichier* KML correctement structuré, et l'intégrer finalement à une page utilisant l'API Google Maps.

Exemple avec MySQL

Cette technique a été utilisée pour un projet Sigma en 2007 : <http://www.dynamosolidaire.fr/>

Le trajet du parcours à afficher, composé de segments, à été dessiné sous Google Earth (pour correspondre à la carte google maps sur laquelle il sera affiché), exporté en KML, et déstructuré en éléments pour être stocké dans une table MySQL. La table contient en fait les coordonnées de chaque segment. Ces données sont requêtées et structurées en données KML par du code PHP, qui ajoute la syntaxe XML nécessaire, pour générer l'objet vectoriel utile.

Ce KML est ensuite fourni à l'API Google Maps qui l'intègre à la carte interactive affichée sur le site. L'avantage de cette solution c'est que l'on peut n'utiliser que les données nécessaires au type de carte demandé, ce qui améliore la réactivité de l'application. On peut ainsi simplifier à la volée les tracés, les compresser.

Exemple avec PostGIS

L'avantage de PostGIS c'est qu'il est capable de générer directement des exports de données au format KML, à partir de son format interne de données spatiales vectorielles.

- Fichier php qui génère le KML : http://www.geotests.net/apis/google/pg_kml.php

```
<?php

$dbconn = pg_connect("host=localhost port=5432 dbname=nom
user=utilisateur password=motdepasse ");

$requete = "select nom_dept, ST_asKML(geomunion(transform(the_geom,
4326))) as thegeom from departements where nom_dept = 'Gers' or nom_dept
= 'Landes' group by nom_dept";

$result = pg_exec($dbconn, $requete) or die (pg_errormessage());

$kml = '<?xml version="1.0" encoding="UTF-8"?>' . "\n";
$kml .= '<kml xmlns="http://earth.google.com/kml/2.2">' . "\n";

$kml .= '<Document><name>Departements</name>';

for ($i = 0; $i < pg_num_rows($result); $i++) {
    $nom = pg_result($result, $i, 0);
    $kml_i = pg_result($result, $i, 1);
    $kml .= "<Placemark><name>Departement :
".$nom."</name><description>".$nom."</description>".$kml_i."</Placemark>\n
";
}

$kml .= "</Document></kml>";

pg_close($dbconn);

header("Content-Type: text/xml; charset=utf8");
echo $kml;

?>
```

- Carte obtenue : http://www.geotests.net/apis/google/pg_kml.html

(Même principe que pour le premier listing de ce support, on utilise une fenêtre Google Maps à laquelle on ajout une couche au format KML, mais cette fois elle n'est pas en dur sur le serveur mais générée par le fichier php précédent.)

2- Utilisation de l'API GéoPortail

Le GéoPortail de l'IGN propose depuis avril 2008, en version bêta, l'accès à ses données par le biais d'une API JavaScript basée sur la bibliothèque de fonctions OpenLayers. C'est toujours la version 2.7 d'OL qui est utilisée actuellement, avec l'ajout de fonctions supplémentaires par surcharge.

La gestion des projections est réalisée par la version JavaScript de PROJ4, et les effets graphiques le sont par la bibliothèque script.aculo.us (concurrente d'EXT, autre bibliothèque graphique JS connue, utilisée dans MapFish et GeoEXT).

Dans cette version bêta, le fichier de l'API pèse 370ko compacté, ce qui est assez important (ce fichier doit être téléchargé par le navigateur client pour être exécuté, au premier chargement de la page).

Pour l'instant l'accès au service est gratuit, dans les limites précisées par une licence d'utilisation (qui fixe notamment un nombre maximal de requêtes par jour). Lorsque la version définitive sera disponible, l'utilisation gratuite de l'API ne permettra de visualiser que le contenu classique du GéoPortail : cartes scannées et orthophotos.

Il est aussi possible d'accéder aux données du GéoPortail par les services WMS/WMS-C et WFS, avec un système d'identification et de gestion des accès (payant).

Affichage simple d'une fenêtre GéoPortail avec les paramètres par défaut :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>API Geoportail - votre carte personnelle</title>
  <script
src="http://api.ign.fr/api?v=1.0beta3&key=XXXXXXXX&instance=map"></script>

  <script type="text/javascript">

    function initGeoportalMap() {

      geoportalLoadmap("GeoportalMapDiv", "normal", "FXX");
      if (map.allowedGeoportalLayers) {
        map.addGeoportalLayers (map.allowedGeoportalLayers);
      }
    }
  </script>
</head>

<body>
  <div id="GeoportalMapDiv" style="width:400px;height:400px;"></div>
</body>
</html>
```

Personnalisation

- Ne charger que la couche cartes :

```
function initGeoportalMap() {  
  
    geoportalLoadmap("GeoportalMapDiv", "normal", "FXX");  
  
    if(map.allowedGeoportalLayers){  
        for (var i= 0; i<map.allowedGeoportalLayers.length; i++) {  
            var overloaded_options= null;  
            switch (map.allowedGeoportalLayers[i]) {  
                case 'GEOGRAPHICALGRIDSYSTEMS.MAPS': // cartes  
                    overloaded_options= {  
                                                    opacity: 1.0  
                    };  
                    // On ajoute la couche  
                    map.addGeoportalLayer(map.allowedGeoportalLayers[i]  
                    ,overloaded_options);  
                    break;  
                case 'ORTHOIMAGERY.ORTHOPHOTOS'      : // ortho-photos  
                    // On ne fait rien  
                    break;  
                default                                :  
                    break;  
            }  
        }  
    }  
}
```

- Modification de l'apparence des couches

Pour modifier l'apparence des couches affichées par l'API GéoPortail, il faut rechercher dans le DOM de la page les objets qui représentent les couches, pour ensuite pouvoir modifier leurs paramètres d'affichage.

Exemple : Rendre invisible couche orthophotos

<http://www.geotests.net/apis/geoportail/ex1.html>

```
for(var i = 1; i < map.olMap.layers.length; i++){  
    var layer= map.olMap.layers[i];  
    if(layer.name == 'ORTHOIMAGERY.ORTHOPHOTOS')  
    {  
        layer.setVisibility(false);  
    }  
}
```

On passe en revue les couches (tableau layers de l'objet olMap) avec une boucle for. Si le layer en cours possède un paramètre nommé « ORTHOIMAGERY.ORTHOPHOTOS », on le rend invisible.

- Ajout de données KML

On utilise simplement la fonction OpenLayers correspondant :

```
function initGeoportalMap() {  
    geoportalLoadmap("GeoportalMapDiv", "normal", "FXX");  
  
    if(map.allowedGeoportalLayers){  
        map.addGeoportalLayers(map.allowedGeoportalLayers);  
    }  
  
    map.addLayer("KML",  
        {  
            'terrasse.kml.name':  
            {  
                'fr':"Terrasse Carto",  
            }  
        },  
        "../google/carto.kml",  
        {  
            minZoomLevel:10,  
            maxZoomLevel:17  
        }  
    );  
    map.setCenterAtLonLat(1.40220, 43.577531, 18);  
    map.setZoom();  
}
```

Exemple : <http://www.geotests.net/apis/geoportal/ex2.html>