

# FORMATION CONTINUE IGSO

## GLOBES VIRTUELS – MÉCANISMES ET APPLICATIONS

### MONTAGE D'UNE INTERFACE SUR LA BASE DES API'S GOOGLE EARTH ET GOOGLE MAPS

### CODE COMPLET ET COMMENTÉ DE L'INTERFACE

#### TABLE DES MATIÈRES

1.	Avant de commencer .....	2
1.1.	Squelette HTML .....	2
1.2.	Style CSS .....	2
2.	Utilisation basique de l'API Google Earth.....	3
2.1.	Obtention d'une clé et insertion .....	3
2.2.	Création d'un fichier JavaScript .....	3
2.3.	Squelette pour l'utilisation de l'API .....	3
2.4.	Vue par défaut (dans initCB).....	3
2.5.	Affichage des contrôles (dans initCB) .....	4
2.6.	Affichage des couches (dans initCB) .....	5
3.	Interactions avec Google Earth .....	6
3.1.	Contrôles .....	6
3.2.	Couches .....	6
3.3.	Affichage KML.....	7
3.4.	Géocodage.....	8
4.	Fichiers finaux .....	10
4.1.	index.htm.....	10
4.2.	ge.js.....	11
4.3.	style.css.....	13
5.	Liens utiles.....	14
5.1.	Google .....	14

## 1. AVANT DE COMMENCER

### 1.1. Squelette HTML

Séparation de la page en deux parties à l'aide de conteneurs

Le conteneur « map » accueillera le globe virtuel

→ fichier à enregistrer sous **index.htm**

```
<html>
  <head>
    <title>API Google Earth</title>
    <link rel="stylesheet" type="text/css" href="style.css" media="screen" />
  </head>
  <body>
    <div id="panel">
      <div id="content">
        <h1>API Google Earth</h1>
        <p>Démonstration de quelques fonctionnalités de l'API Google Earth...</p>
      </div>
    </div>
    <div id="map"></div>
  </body>
</html>
```

### 1.2. Style CSS

Mise en page sommaire avec une feuille de style

→ fichier à enregistrer sous **style.css**

```
body {
  font-family: arial;
  font-size: 12px;
  margin: 0;
}

h1 {
  font-size: 20px;
}

#map {
  height: 100%;
  float: left;
  width: 80%;
}

#panel {
  float: left;
  height: 100%;
  width: 20%;
}

#content {
  padding: 10px;
}

fieldset {
  margin-bottom: 12px;
}

input {
  font-size: 10px;
}

img {
  border: 0;
}
```

## 2. UTILISATION BASIQUE DE L'API GOOGLE EARTH

### 2.1. Obtention d'une clé et insertion

Adresse : <http://code.google.com/intl/fr/apis/maps/signup.html>

La clé est valable uniquement pour un nom de domaine

Accès à l'API de Google avec JavaScript à l'adresse <http://www.google.com/jsapi?key=> (avec la clé obtenue précédemment)

```
<script type="text/javascript" src="http://www.google.com/jsapi?key="></script>
```

### 2.2. Création d'un fichier JavaScript

Création d'un fichier JavaScript « ge.js » qui contiendra le code pour interagir avec Google Earth

```
<script type="text/javascript" src="ge.js"></script>
```

### 2.3. Squelette pour l'utilisation de l'API

Chargement de l'API Google Earth

Création d'une instance Google Earth dans le conteneur « map »

Affichage du globe virtuel

Lancement des fonctions « init » (et après « initCB ») à la fin du chargement de la page HTML

→ fichier à enregistrer sous **ge.js**

```
var ge;

google.load("earth", "1");

function init() {
    google.earth.createInstance("map", initCB);
}

function initCB(instance) {
    ge = instance;
    ge.getWindow().setVisibility(true);
}

google.setOnLoadCallback(init);
```

#### Méthodes

- **google.load** (*string moduleName, string moduleVersion, array optionalSettings*) :  
Chargement de l'API Google, avec sa version
- **google.earth.createInstance** (*object domNode, function initCallback*) :  
Création d'une instance du plugin Google Earth dans conteneur de la page HTML et appel d'une fonction en cas de succès
- **ge.getWindow().setVisibility** (*bool visibility*) :  
Rend visible le globe dans le navigateur
- **google.setOnLoadCallback** (*function init*) :  
Appel d'une fonction lorsque la page HTML et l'API sont chargées

### 2.4. Vue par défaut (dans initCB)

Définition de l'emplacement de la vue au chargement

Paramètres : longitude et latitude, hauteur, inclinaison

```
/* View
***** */
var lookAt = ge.createLookAt(" ");
```

```
// Coordonates
lookAt.setLatitude(46.777);
lookAt.setLongitude(6.65);

lookAt.setRange(2500);

// Camera inclinaison
lookAt.setTilt(60);

// Go to
ge.getView().setAbstractView(lookAt);
```

### Méthodes

- **ge.createLookAt()** :  
Création d'une nouvelle vue (position de la caméra sur un objet à visualiser)
- **lookAt.setLatitude** (*double latitude*) :  
Latitude du point que la caméra regarde, valeurs comprises entre -90° (Pôle sud) et 90° (Pôle nord)
- **lookAt.setLongitude** (*double longitude*) :  
Longitude du point que la caméra regarde, valeurs comprises entre -180° et 180°
- **lookAt.setRange** (*double range*) :  
Distance en mètres par rapport au point
- **lookAt.setTilt** (*double tilt*) :  
Angle entre la direction du point et la normale à la surface de la Terre, valeurs comprises entre 0° (vue de dessus) et 90° (vue le long de l'horizon)
- **ge.getView().setAbstractView** (*KmlAbstractView view*) :  
Défini la caméra dans Google Earth

## 2.5. Affichage des contrôles (dans initCB)

Affichage du contrôle de navigation

Affichage des différentes options (barre de statut, barre d'échelle)

```
ge.getNavigationControl().setVisibility(ge.VISIBILITY_SHOW);

ge.getOptions().setUnitsFeetMiles(false); // Metric units
ge.getOptions().setStatusbarVisibility(true);
ge.getOptions().setScaleLegendVisibility(true);
ge.getOptions().setGridVisibility(true);
ge.getOptions().setOverviewMapVisibility(true);
ge.getSun().setVisibility(true);
```

### Méthodes

- **ge.getNavigationControl().setVisibility** (*GEVisibilityEnum visibility*) :  
Visibilité des contrôles de navigation :
  - ge.VISIBILITY\_SHOW : Toujours
  - ge.VISIBILITY\_HIDE : Jamais
  - ge.VISIBILITY\_AUTO : Caché si non utilisé (affichage avec survol de la souris)
- **ge.getOptions().setUnitsFeetMiles** (*bool unitsFeetMiles*) :  
Défini le système d'unité d'affichage (vrai : impérial ou faux : métrique)
- **ge.getOptions().setStatusbarVisibility** (*bool statusBarVisibility*) :  
Visibilité de la barre de statut
- **ge.getOptions().setScaleLegendVisibility** (*bool scaleLegendVisibility*) :  
Visibilité de la barre d'échelle
- **ge.getOptions().setGridVisibility** (*bool gridVisibility*) :  
Visibilité de la grille

- **ge.getOptions().setOverviewMapVisibility** (*bool overviewMapVisibility*) :  
Visibilité de la mini carte
- **ge.getSun().setVisibility** (*bool visibility*) :  
Visibilité du soleil (composante temporelle)

## 2.6. Affichage des couches (dans initCB)

Activation des couches (Frontières et noms, routes)

```
ge.getLayerRoot().enableLayerById(ge.LAYER_BORDERS, true);
ge.getLayerRoot().enableLayerById(ge.LAYER_ROADS, true);
ge.getLayerRoot().enableLayerById(ge.LAYER_BUILDINGS, true);
ge.getLayerRoot().enableLayerById(ge.LAYER_BUILDINGS_LOW_RESOLUTION, true);
```

### Méthode

- **ge.getLayerRoot().enableLayerById** (*string id, bool visibility*) :  
Active une couche en fonction de son identifiant :
  - LAYER\_BORDERS : Frontières des pays et régions, noms des villes, lieux, états, lacs, etc.
  - LAYER\_BUILDINGS : Bâtiments 3D
  - LAYER\_BUILDINGS\_LOW\_RESOLUTION : Bâtiments gris (non texturé)
  - LAYER\_ROADS : Routes et noms de routes
  - LAYER\_TERRAIN : Terrain 3D

## 3. INTERACTIONS AVEC GOOGLE EARTH

### 3.1. Contrôles

#### HTML

Création de checkboxes pour chaque option (échelle, barre de statut)

Ajout d'un événement JavaScript de type « onClick » sur les checkboxes, qui appelle la fonction toggleOption avec en paramètre son identifiant (id = nom de l'option)

```
<fieldset>
  <legend>Options</legend>
  <p><label><input type="checkbox" id="StatusBar" onclick="toggleOption(this.id)" /> Barre de
status</label></p>
  <p><label><input type="checkbox" id="ScaleLegend" onclick="toggleOption(this.id)" />
Echelle</label></p>
</fieldset>
```

#### JavaScript

Création de la fonction « toggleOption »

Récupération de la checkbox (identifiant) qui a appelée la fonction

Test du statut (coché ou décoché) et création d'une variable avec le statut

Concaténation de l'option et du statut pour modifier le contrôle via l'API

(p. ex. : ge.getOptions().setGridVisibility(true))

Exécution de la commande

```
function toggleOption(option) {
  var objCheckbox = document.getElementById(option);

  (objCheckbox.checked) ? newStatus = "true" : newStatus = "false";

  var command = "ge.getOptions().set" + option + "Visibility(" + newStatus + ")";
  eval(command);
}
```

### 3.2. Couches

#### HTML

Création de checkboxes pour chaque couche (frontières et noms, routes)

Ajout d'un événement JavaScript de type « onClick » sur les checkboxes, qui appelle la fonction toggleLayer avec en paramètre son identifiant (id = nom de la couche)

```
<p><label><input type="checkbox" id="BORDERS" onclick="toggleLayer(this.id)" /> Frontières et
noms</label></p>
<p><label><input type="checkbox" id="ROADS" onclick="toggleLayer(this.id)" />
Routes</label></p>
```

#### JavaScript

Création de la fonction « toggleLayer »

Récupération de la checkbox (identifiant) qui a appelée la fonction

Test du statut (coché ou décoché) et création d'une variable avec le statut

Concaténation de la couche et du statut pour modifier la couche via l'API

(p. ex. : ge.getLayerRoot().enableLayerById(ge.LAYER\_ROADS, true))

Exécution de la commande

```
function toggleLayer(layer) {
    var objCheckbox = document.getElementById(layer);

    (objCheckbox.checked) ? newStatus = "true" : newStatus = "false";

    var command = "ge.getLayerRoot().enableLayerById(ge.LAYER_" + layer + ", " + newStatus +
    ")";
    eval(command);
}
```

### 3.3. Affichage KML

#### HTML

Création de checkboxes pour chaque fichier

Ajout d'un événement JavaScript de type « onClick » sur les checkboxes, qui appelle la fonction toggleFile avec en paramètre son identifiant (id = nom du fichier)

```
<fieldset>
  <legend>KML</legend>
  <p><label><input type="checkbox" id="batiments.kmz" onclick="toggleFile(this.id)" />
  Bâtiments</label></p>
  <p><label><input type="checkbox" id="luminaires.kmz" onclick="toggleFile(this.id)" />
  Luminaires</label></p>
  <p><label><input type="checkbox" id="eclairage.kml" onclick="toggleFile(this.id)" />
  Eclairage</label></p>
</fieldset>
```

#### JavaScript

Création d'un tableau contenant les noms de fichier à afficher (à paramétrer)

```
var currentKml = {
  "fichier1.kmz": null, // To set
  "fichier2.kml": null, // To set
  "fichier3.kmz": null // To set
};
```

Création de la fonction « toggleFile »

Récupération de la checkbox (identifiant) qui a appelée la fonction

Test du statut (coché ou décoché)

Si coché, appel de la fonction « readKml »

Autrement, désactivation du KML actuellement affiché

```
function toggleFile(file) {
    var objCheckbox = document.getElementById(file);

    if (objCheckbox.checked) {
        readKml(file);
    }
    else {
        ge.getFeatures().removeChild(currentKml[file]);
        currentKml[file] = null;
    }
}
```

#### Méthode

- **ge.getFeatures().removeChild (KmlObject oldChild) :**  
Enlève l'objet KML

Création de la fonction « readKml »

Définition de l'emplacement (dossier) des fichiers (à paramétrer)



```

function goTo() {
  var geocodeLocation = document.getElementById("location").value;
  var geocoder = new google.maps.ClientGeocoder();

  geocoder.getLatLng(geocodeLocation, function(point) {

    if (!point) {
      alert(geocodeLocation + " pas trouvé !")
    }

    else {

      // Camera
      var lookAt = ge.createLookAt("");

      lookAt.setLatitude(point.y);
      lookAt.setLongitude(point.x);
      lookAt.setRange(1000);
      lookAt.setTilt(60);

      ge.getView().setAbstractView(lookAt);

      // Marker
      var placemark = ge.createPlacemark("");
      placemark.setName(geocodeLocation);

      var place = ge.createPoint("");
      place.setLatitude(point.y);
      place.setLongitude(point.x);
      placemark.setGeometry(place);

      ge.getFeatures().appendChild(placemark);
    }
  });
}

```

### Méthodes

- **google.maps.ClientGeocoder ()** :  
Création d'une instance du « géocodeur » de Google (API Maps)
- **geocoder.getLatLng (string address)** :  
Envoi d'une requête qui géocode une adresse et, en cas de succès, retourne ses coordonnées
- **ge.createPlacemark ()** :  
Crée un marqueur
- **placemark.setName (string name)** :  
Définition du nom (étiquette) du marqueur
- **ge.createPoint ()** :  
Création d'un point
- **place.setLatitude (double latitude)** :  
Latitude du point, valeurs comprises entre -90° (Pôle sud) et 90° (Pôle nord)
- **place.setLongitude (double longitude)** :  
Longitude du point, valeurs comprises entre -180° et 180°
- **placemark.setGeometry (KmlGeometry geometry)** :  
Définition de la géométrie du marqueur
- **ge.getFeatures().appendChild (KmlObject object)** :  
Ajout du marqueur sur le globe

## 4. FICHIERS FINAUX

### 4.1. index.htm

```

<html>
  <head>
    <title>API Google Earth</title>
    <link rel="stylesheet" type="text/css" href="style.css" media="screen" />
    <script type="text/javascript"
src="http://www.google.com/jsapi?key=ABQIAAAAMnUP20scziB71LfpQBVDtBS36fy1C67uFvqw6k5574VKBBkJh
xRBUPP4gx_hzn2xUl_73xpVKBj0cw"></script>
    <script type="text/javascript" src="ge.js"></script>
  </head>
  <body>
    <div id="panel">
      <div id="content">
        <h1>API Google Earth</h1>
        <p>Démonstration de quelques fonctionnalités de l'API Google Earth dans le cadre du
cours &laquo; Globes virtuels &raquo; de l'IGSO du 27 mai 2010.</p>
        <fieldset>
          <legend>Options</legend>
          <p><label><input type="checkbox" id="StatusBar" onclick="toggleOption(this.id)" />
Barre de status</label></p>
          <p><label><input type="checkbox" id="ScaleLegend" onclick="toggleOption(this.id)" />
Echelle</label></p>
          <p><label><input type="checkbox" id="BORDERS" onclick="toggleLayer(this.id)" />
Frontières et noms</label></p>
          <p><label><input type="checkbox" id="ROADS" onclick="toggleLayer(this.id)" />
Routes</label></p>
        </fieldset>
        <fieldset>
          <legend>KML</legend>
          <p><label><input type="checkbox" id="batiments.kmz" onclick="toggleFile(this.id)" />
Bâtiments</label></p>
          <p><label><input type="checkbox" id="luminaires.kmz" onclick="toggleFile(this.id)" />
Luminaires</label></p>
          <p><label><input type="checkbox" id="eclairage.kml" onclick="toggleFile(this.id)" />
Eclairage</label></p>
          <p><label><input type="checkbox" id="heigvd_mnt.kmz" onclick="toggleFile(this.id)" />
HEIG-VD (Cheseaux)</label></p>
          <p><label><input type="checkbox" id="st-roch.kmz" onclick="toggleFile(this.id)" />
HEIG-VD (St-Roch)</label></p>
          <p><label><input type="checkbox" id="viaduc_des_vaux.kmz"
onclick="toggleFile(this.id)" /> Viaduc des Vaux</label></p>
          <p><label><input type="checkbox" id="sargans.kml" onclick="toggleFile(this.id)" />
Vol ADS40</label></p>
        </fieldset>
        <fieldset>
          <legend>Géocodage</legend>
          <form action="" onsubmit="goTo(); return false">
            <p><input id="location" type="text" size="30" /> &nbsp; <input type="submit"
value="Aller"></p>
          </form>
        </fieldset>
        <p>
          <a href="http://www.heig-vd.ch/" target="_blank"></a> &nbsp; &nbsp;
          <a href="http://www.igso.ch/" target="_blank"></a>
        </p>
      </div>
    </div>
    <div id="map"></div>
  </body>
</html>

```

## 4.2. ge.js

```

var ge;

var currentKml = {
  "fichier1.kmz": null, // To set
  "fichier2.kml": null, // To set
  "fichier3.kmz": null // To set
};

google.load("earth", "1");
google.load("maps", "2");

function init() {
  google.earth.createInstance("map", initCB);
}

function initCB(instance) {
  ge = instance;
  ge.getWindow().setVisibility(true);

  ge.getOptions().setUnitsFeetMiles(false); // Metric units

  /* View
  ***** */
  var lookAt = ge.createLookAt("");

  // Coordinates
  lookAt.setLatitude(46.777);
  lookAt.setLongitude(6.65);

  lookAt.setRange(2500);

  // Camera inclinaison
  lookAt.setTilt(60);

  // Go to
  ge.getView().setAbstractView(lookAt);

  /* Controls
  ***** */
  ge.getNavigationControl().setVisibility(ge.VISIBILITY_SHOW);
  //ge.getOptions().setStatusbarVisibility(true);
  //ge.getOptions().setScaleLegendVisibility(true);
  //ge.getOptions().setGridVisibility(true);
  //ge.getOptions().setOverviewMapVisibility(true);
  //ge.getSun().setVisibility(true);

  /* Layers
  ***** */
  //ge.getLayerRoot().enableLayerById(ge.LAYER_BORDERS, true);
  //ge.getLayerRoot().enableLayerById(ge.LAYER_ROADS, true);
  //ge.getLayerRoot().enableLayerById(ge.LAYER_BUILDINGS, true);
  //ge.getLayerRoot().enableLayerById(ge.LAYER_BUILDINGS_LOW_RESOLUTION, true);
}

google.setOnLoadCallback(init);

/* Options
***** */
function toggleOption(option) {
  var objCheckbox = document.getElementById(option);

  (objCheckbox.checked) ? newStatus = "true" : newStatus = "false";

  var command = "ge.getOptions().set" + option + "Visibility(" + newStatus + ")";
  eval(command);
}

/* Layers
***** */
function toggleLayer(layer) {
  var objCheckbox = document.getElementById(layer);

  (objCheckbox.checked) ? newStatus = "true" : newStatus = "false";
}

```

```

var command = "ge.getLayerRoot().enableLayerById(ge.LAYER_" + layer + ", " + newStatus +
");";
eval(command);
}

/* Display file
***** */
function toggleFile(file) {
    var objCheckbox = document.getElementById(file);

    if (objCheckbox.checked) {
        readKml(file);
    }
    else {
        ge.getFeatures().removeChild(currentKml[file]);
        currentKml[file] = null;
    }
}

/* Read KML
***** */
function readKml(file) {
    var kmlFolder = "http://site.com/dossier/"; // To set
    var kmlUrl = kmlFolder + file;

    google.earth.fetchKml(ge, kmlUrl, function(kml) {
        if (!kml) {
            alert("Erreur : Lecture du fichier KML impossible !");
            return;
        }

        ge.getFeatures().appendChild(kml);
        currentKml[file] = kml;

        if (kml.getAbstractView()) {
            ge.getView().setAbstractView(kml.getAbstractView());
        }
    });
}

/* Geocoding
***** */
function goTo() {
    var geocodeLocation = document.getElementById("location").value;
    var geocoder = new google.maps.ClientGeocoder();

    geocoder.getLatLng(geocodeLocation, function(point) {

        if (!point) {
            alert(geocodeLocation + " pas trouvé !")
        }

        else {
            // Camera
            var lookAt = ge.createLookAt("");
            lookAt.setLatitude(point.y);
            lookAt.setLongitude(point.x);
            lookAt.setRange(1000);
            lookAt.setTilt(60);
            ge.getView().setAbstractView(lookAt);

            // Marker
            var placemark = ge.createPlacemark("");
            placemark.setName(geocodeLocation);
            var place = ge.createPoint("");
            place.setLatitude(point.y);
            place.setLongitude(point.x);
            placemark.setGeometry(place);
            ge.getFeatures().appendChild(placemark);
        }
    });
}

```

### 4.3. style.css

```
body {
  font-family: arial;
  font-size: 12px;
  margin: 0;
}

h1 {
  font-size: 20px;
}

#map {
  background-color: #e6e6e6;
  height: 100%;
  float: left;
  width: 80%;
}

#panel {
  float: left;
  height: 100%;
  width: 20%;
}

#content {
  padding: 10px;
}

fieldset {
  margin-bottom: 12px;
}

input {
  font-size: 10px;
}

img {
  border: 0;
}
```

## 5. LIENS UTILES

---

### 5.1. Google

- Guide : <http://code.google.com/intl/fr/apis/earth/documentation/>
- Exemples : <http://code.google.com/intl/fr/apis/earth/documentation/examples.html>
- Galerie de démonstration : <http://code.google.com/intl/fr/apis/earth/documentation/demogallery.html>
- Référence : <http://code.google.com/intl/fr/apis/earth/documentation/reference/>
- Clé : <http://code.google.com/intl/fr/apis/maps/signup.html>