Canvas
WEBGL et Three.js

Christophe Vestri

Le mardi 24 janvier 2023

Plan du cours

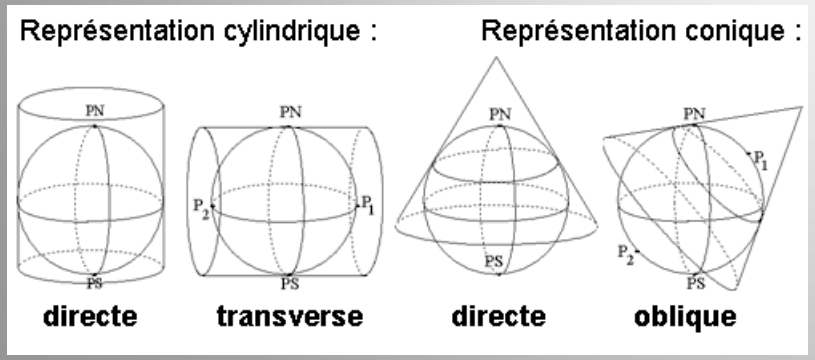
- 3 janvier : Intro, github, Capteur/Geoloc en HTML5
- 10 janvier: carto/geo, leaflet/mapBox, rest Api
- 24 janvier: 2D/3D: Canvas, WebGL et Three.js
- 31 janvier: Aframe/AR.js, exercice + projet
- 7 février : Projets

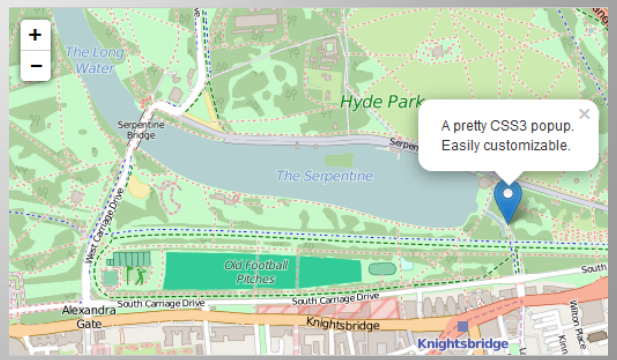
Plan Cours 3

- Rappel dernier cours
- Canvas et SVG
- CSS3D
- WebGL et Three.js
  - Exercice: ThreeJs et Device Events

Géo + Html5 + LeafLet.js

- Repères Géo et carto
- Acces capteur caméra: Géolocalisation, DeviceOrientation, DeviceMotion
- Leafletjs, Mapbox, mapQuest
- Données géolocalisées (REST API)





Solution exercice 3

- **Utiliser Leaflet**

L.geoJSON(geojsonfeature).addTo(map);

https://leafletjs.com/examples/geojson/

- **Ensuite requête html avec format Geojson**

```
function reqListener () {  L.geoJson(JSON.parse(this.response)).addTo(map); }
var xmlhttp = new XMLHttpRequest();
xmlhttp.addEventListener("load", reqListener);
xmlhttp.open("GET", "http://monapi")
xmlhttp.send()
```

https://www.w3schools.com/js/js_json_http.asp

Graphique en HTML

- Canvas 2D
- SVG: Scalable Vector Graphics
- CSS3D: pour des effets de rendu 3D
- WebGL: pour de la 3D basique
- Three.js: pour de la 3D plus poussée
  - Ex1 Canvas+SVG=1h
  - Exo2 Threejs: 2h

CANVAS HTML

- **Element Html pour dessiner**

```
<canvas id=“mycanvas" width="500" height=“300"></canvas>
<img id="scream" width="220" height="277“ src="pic_the_scream.jpg">
```

- **Context 2D (dessin) ou 3D (WebGL)**
- **Acces en javascript (dans le DOM)**

```
var canvas = document.getElementById(‘mycanvas');
var myimg = document.getElementById(‘scream');
var ctx = canvas.getContext('2d');
ctx.drawImage(myimg, 10, 120);
ctx.fillStyle = 'green';
ctx.fillRect(30, 30, 100, 100);
```

https://developer.mozilla.org/fr/docs/Tutoriel_canvas
https://www.w3schools.com/graphics/tryit.asp?filename=trycanvas_image
https://www.alsacreations.com/tuto/lire/1484-introduction.html

SVG-Scalable Vector Graphic

- **Format graphique d'image XML**
- **Image sans perte**
- **Manipulé en javascript (dans le DOM)**
- **Manipulé par CSS**
- https://www.w3schools.com/graphics/svg_intro.asp
- http://edutechwiki.unige.ch/fr/Tutoriel_SVG_avec_HTML5
- SVG or canvas: https://css-tricks.com/when-to-use-svg-vs-when-to-use-canvas/

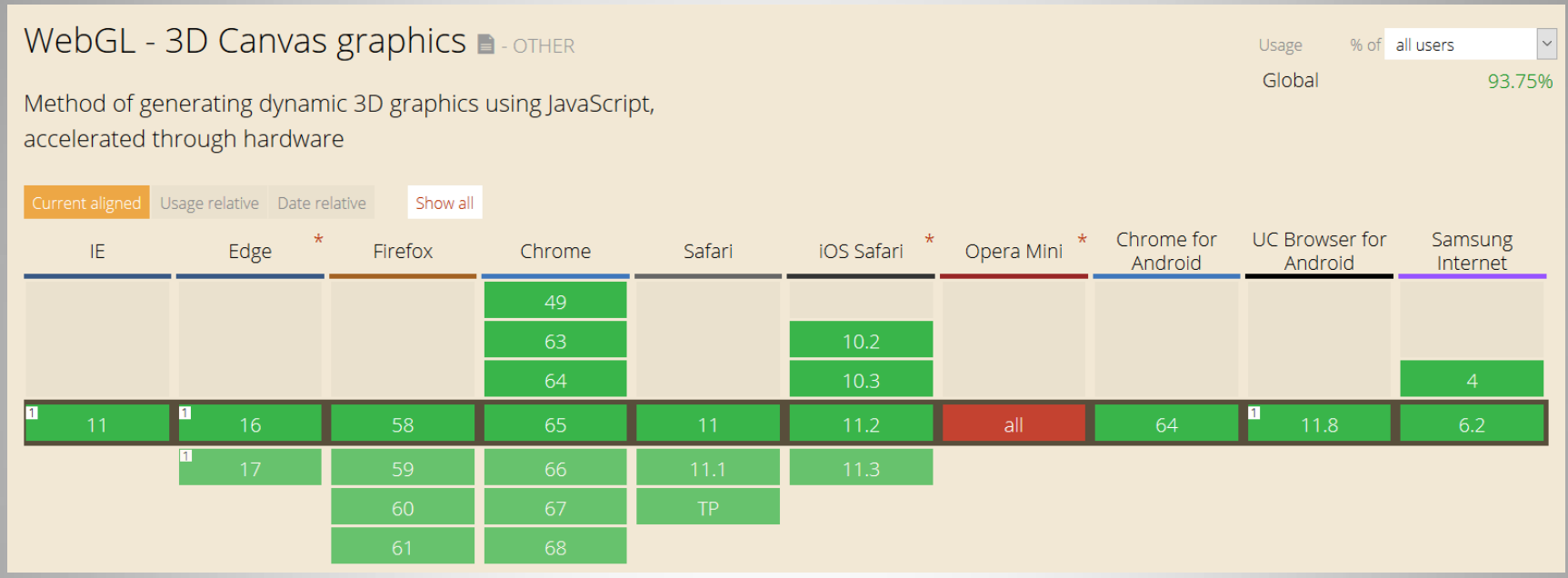
CSS 3D Transform

- Tous les elements (graphiques) peuvent être transformés:
  - Shift, rotation, perspective….
- **https://www.w3schools.com/css/css3_3dtransforms.asp**
- **https://drafts.csswg.org/css-transforms/**
- **https://keithclark.co.uk/labs/css-fps/**

WebGL

- Qu'est-ce que WebGL
  - Cross plateforme et libre de droits
  - OpenGL ES (OpenGL simplifié pour l'embarqué) dans le Web (HTML5)
  - Bonne intégration Html et mécanisme d'évènements
  - DOM API pour affichage 2D et 3D
  - Langage de type script (pas de compilation)
  - Accélérations matérielles et GPU (GLSL)

WebGL



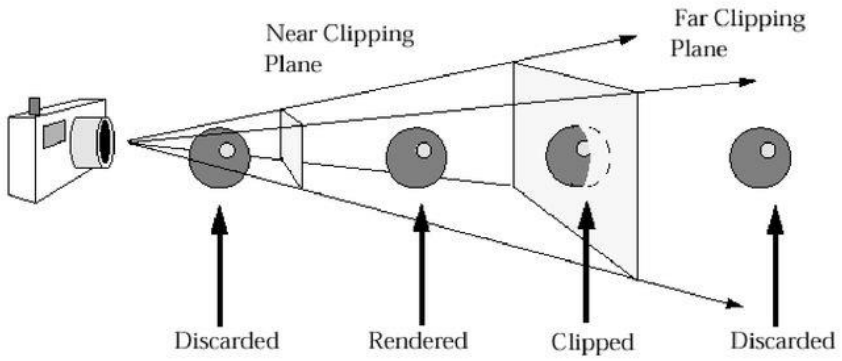
– **Blacklist (quelques vieux smartphones): https://www.khronos.org/webgl/wiki/BlacklistsAndWhitelists**

WebGL

- **Computer graphics**

3D Clipping

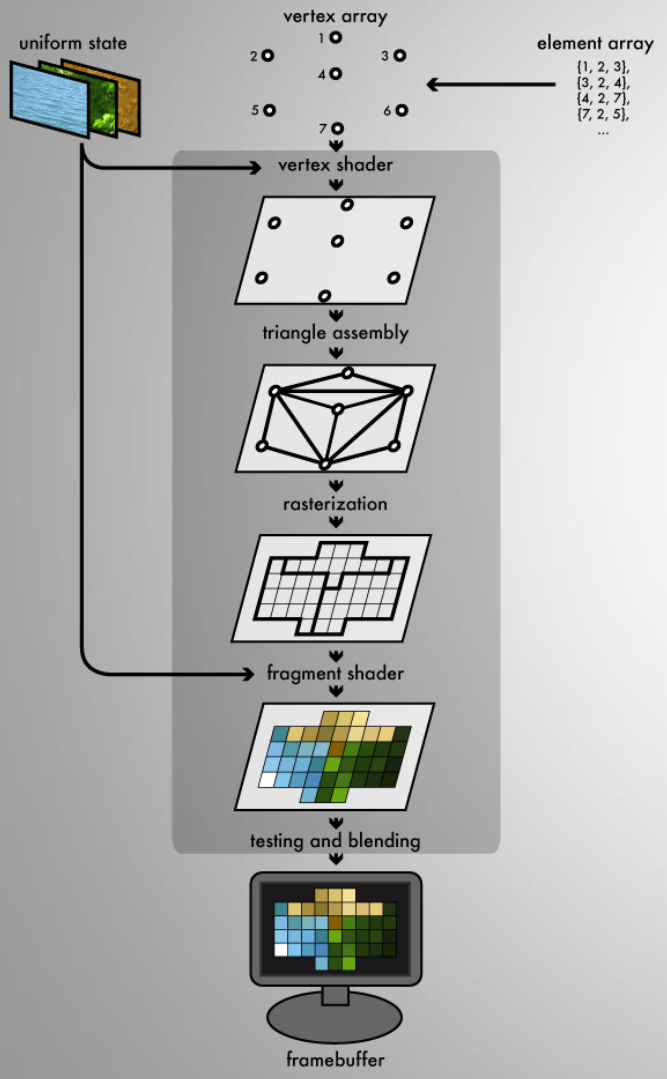
- Objects that are partially within the viewing volume need to be clipped – just like the 2D case

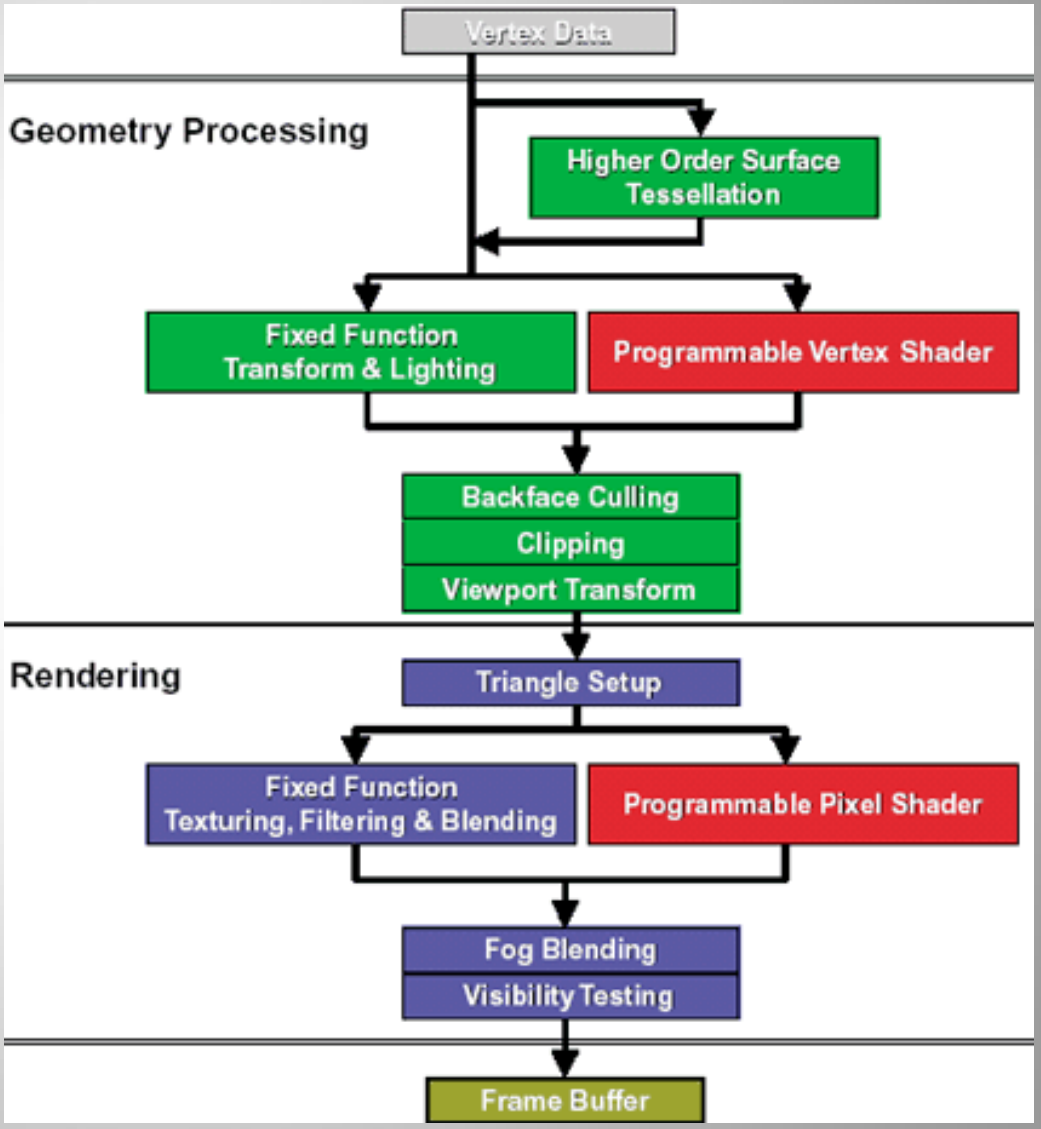


WebGL

- **Low-level API**
  - **GLSL OpenGL Shading Langage**
  - **Machine d'état: OpenGL Context**
  - **Calcul de matrices et transformations**
  - **Buffers de vertex: positions, normals, color, texture**
  - **Depth buffer, Blending, transparency**
  - **Lighting, Cameras…**
  - **https://developer.mozilla.org/fr/docs/Web/API/WebGL_API**
  - **https://webglfundamentals.org/webgl/lessons/fr/**

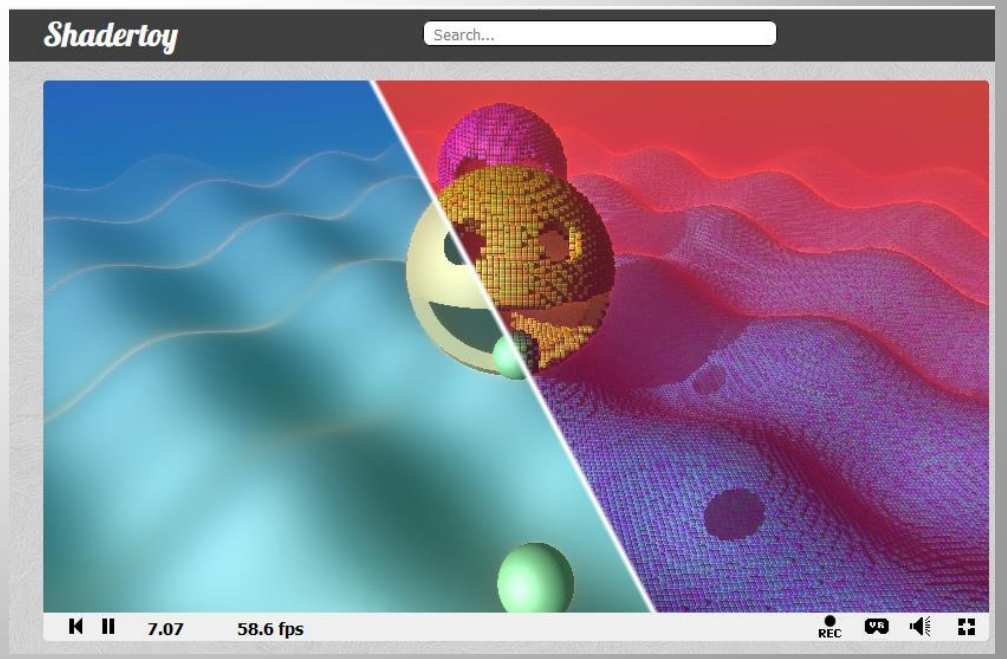
WebGL Pipeline





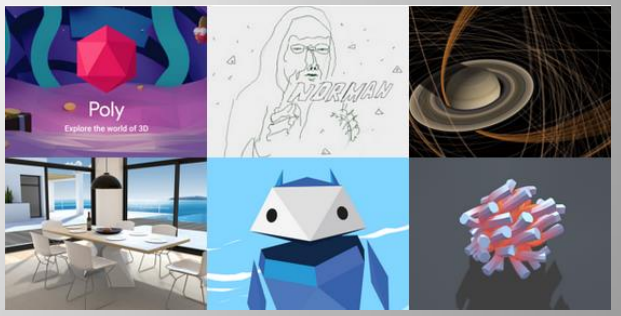
WebGL Exemples

- https://www.khronos.org/webgl/wiki/Tutorial
- https://webglfundamentals.org/
- https://www.shadertoy.com/



Three.js

- **Qu’est-ce que Three.js**
  - **Couche abstraite et haut niveau de WebGL**
  - **Librairie javascript pour créer des scènes 3D**
  - **Cross-plateforme et gratuit**
  - **Rendus en webGL, CSS3D et SVG**
  - **https://threejs.org/**



  - **https://davidlyons.dev/threejs-intro/**

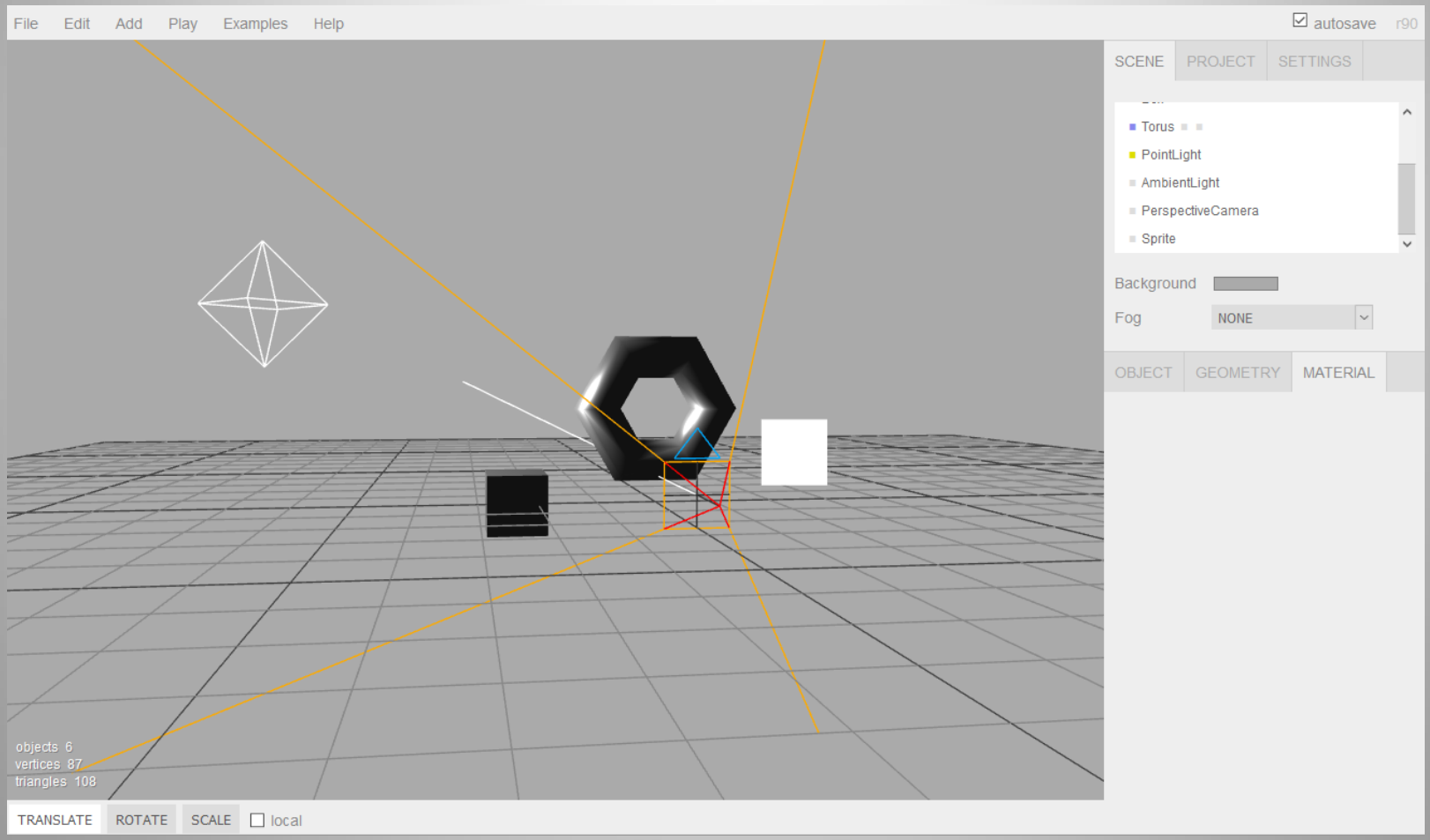
Fonctionalités

THREE.JS

- Scenes, Cameras, Renderer,
- Geometry, Materials, Textures
- Lights, Shadows
- Shaders, Particles, LOD
- Loaders: Json compatible Blender, 3D max, Wavefront OBJ, Autodesk FBX
- Animation, Trackballcontrols, Math Utilities

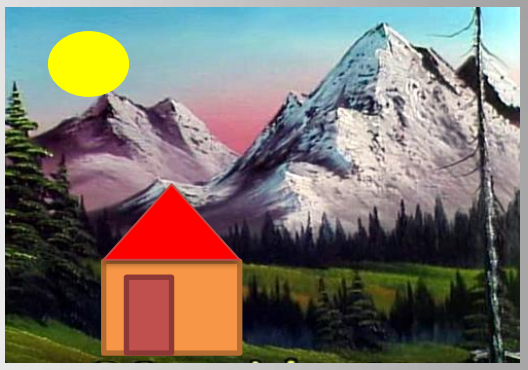
Threejs Editor

- https://threejs.org/editor/

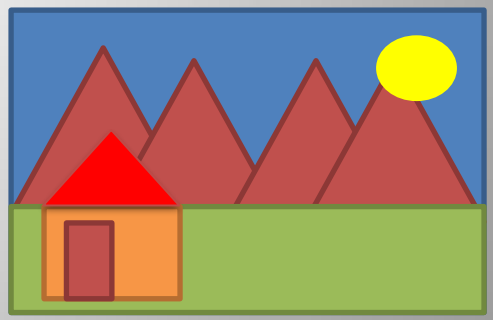


Exercice 1 (1h)

- **Dessiner dans un canvas et un svg (1 pages ou 2 pages séparées)**
  - **Canvas:**
    - **Choisir et afficher une image avec ciel, prairie, montagne**



    - **Dessiner Maison + soleil: Rectangle + triangle + ronds**
  - **SVG:**
    - **Dessiner ciel, montagne, soleil et maison**
    - **Quand on passe curseur sur la porte, elle change de couleur**
    - **Quand on clique sur Soleil ciel gris**



Outils de debug

- En local (besoin pour charger modèles 3D):
  - Avoir python (miniconda ou autre)
  - Se placer dans le répertoire html
  - python3 -m http.server
  - http://localhost:8000/ firefox ou chrome
- Smartphone android -> Chrome
- https://developers.google.com/web/tools/chrome-devtools/javascript
  - Simulation de smartphone (F12)
  - Connecté à un smartphone: chrome://inspect/

Exercice 2 – Three.js

- **Exercice 2 (2h) :**
  - **Créez une scène + caméra + light + renderer**
  - **Créez un objet générique (sphère ou cube)**
  - **Texturez cet objet**
  - **Téléchargez un objet 3D**
  - **Animez les objets avec les DeviceEvents: DeviceOrientation et/ou DeviceMotion**
  - **Ajoutez Fog/pluie ou particules**
- **Exo3 (bonus): mettre un contexte: compas/gyro, système solaire, animation….**

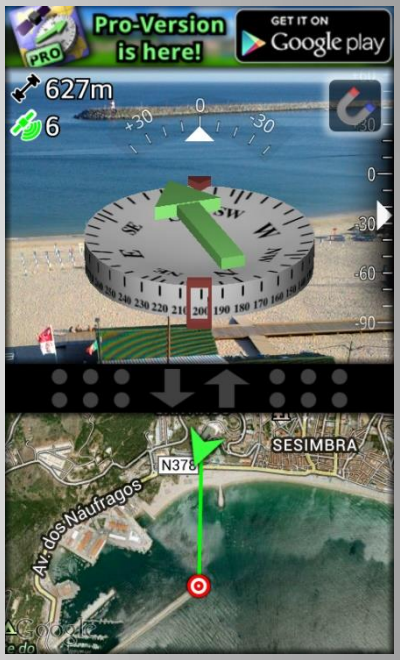
Projet final

Evaluation:

Exos des cours (50%)

Projet (50%)

- **Projet final (7 février)**
  - **Capteurs mouvement/orientation**
  - **GéoLocalisation et/ou objets geolocalisés**
  - **UI et scene 3D, interaction**
  - **Exemples:**
    - **Compas 2D/3D: carte 2D + geoloc et directions 3D**
    - **Objets 3D animés avec interaction smartphone**



- **Présentation**
  - **Qqs slides, 5/10min chacun (contexte/code) avec démo sur écran/smartphone**