

# Tutorial Adaptation IHM

## Unity3D & Polymer

Comment avez vous réalisé l'exemple ? Avec quel outil de développement, de tests ? Comment déploie-t-on et exécute-t-on l'exemple ? Comment teste-t-on les capacités d'adaptations ?

Ce tutoriel a pour but d'expliquer comment nous avons réalisé notre application d'exemple qui permet de mettre en valeur des problématiques d'adaptation des IHMs avec une solution cross-plateforme (Unity3D) et une solution web components (Polymer).

Pour la partie Unity3D, afin de nous concentrer sur les problématiques d'adaptation et non sur le jeu en lui même nous avons choisi de réutiliser un jeu simple libre et gratuit fournis par les développeurs de Unity3D pour apprendre la technologie : Survival Shooter.

Pour la partie Polymer, nous développerons un site autour du thème de ce jeu.

Pour plus d'informations sur le jeu en lui même, vous pouvez réaliser le tutorial : <https://unity3d.com/learn/tutorials/projects/survival-shooter-project>

# Partie 1 : Unity 3D

Unity 3D est un des moteurs de jeux les plus répandus dans l'industrie du jeu vidéo. Il est apprécié notamment pour sa couche d'abstraction qu'il fournit pour gérer les problématiques 3D et permet ainsi de se concentrer sur le jeu. Il est réputé pour faire du prototypage rapide et cross-plateforms, pour de grands studios comme pour des studios indépendants.

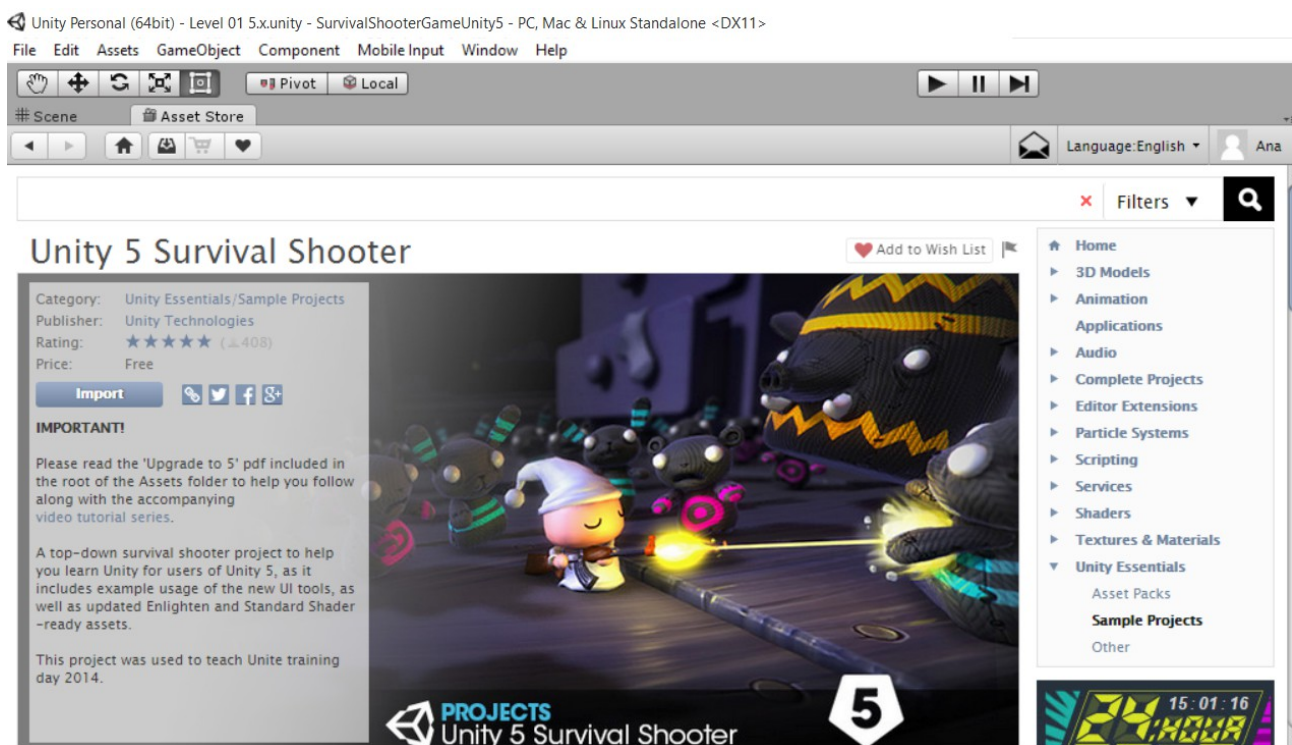
## Configuration :

- environnement de développement : Unity 5.2.1 personal edition (la version gratuite) sur Window 10
- environnement de tests : smartphone LG G3 (Android), firefox v41.0.1 et chrome v46.0.2490.71 sur Windows 10 et firefox, chromium Version 37.0.2062 et chrome Version 46.0.2490.71 et firefox v41.0.1 sur Ubuntu 14.0.2

## 1. Mise en place

Télécharger et installer Unity 5.2.1 personal edition (la version gratuite) depuis le site officiel : <https://unity3d.com/>

Importer le projet Survival Shooter (gratuit) depuis l'Asset Store de Unity



The screenshot shows the Unity Personal (64bit) interface with the Asset Store open. The main content area displays the 'Unity 5 Survival Shooter' project page. The page includes a category (Unity Essentials/Sample Projects), publisher (Unity Technologies), a 5-star rating (408 reviews), and a price of Free. There is an 'Import' button and social media sharing options. A prominent 'IMPORTANT!' notice is displayed, advising users to read the 'Upgrade to 5' pdf. Below this, a description states: 'A top-down survival shooter project to help you learn Unity for users of Unity 5, as it includes example usage of the new UI tools, as well as updated Enlighten and Standard Shader-ready assets. This project was used to teach Unite training day 2014.' The main image shows a 3D scene with a character and enemies. A sidebar on the right contains a navigation menu with categories like Home, 3D Models, Animation, Applications, Audio, Complete Projects, Editor Extensions, Particle Systems, Scripting, Services, Shaders, Textures & Materials, and Unity Essentials. The bottom of the page features a 'PROJECTS' logo and a '5' badge.

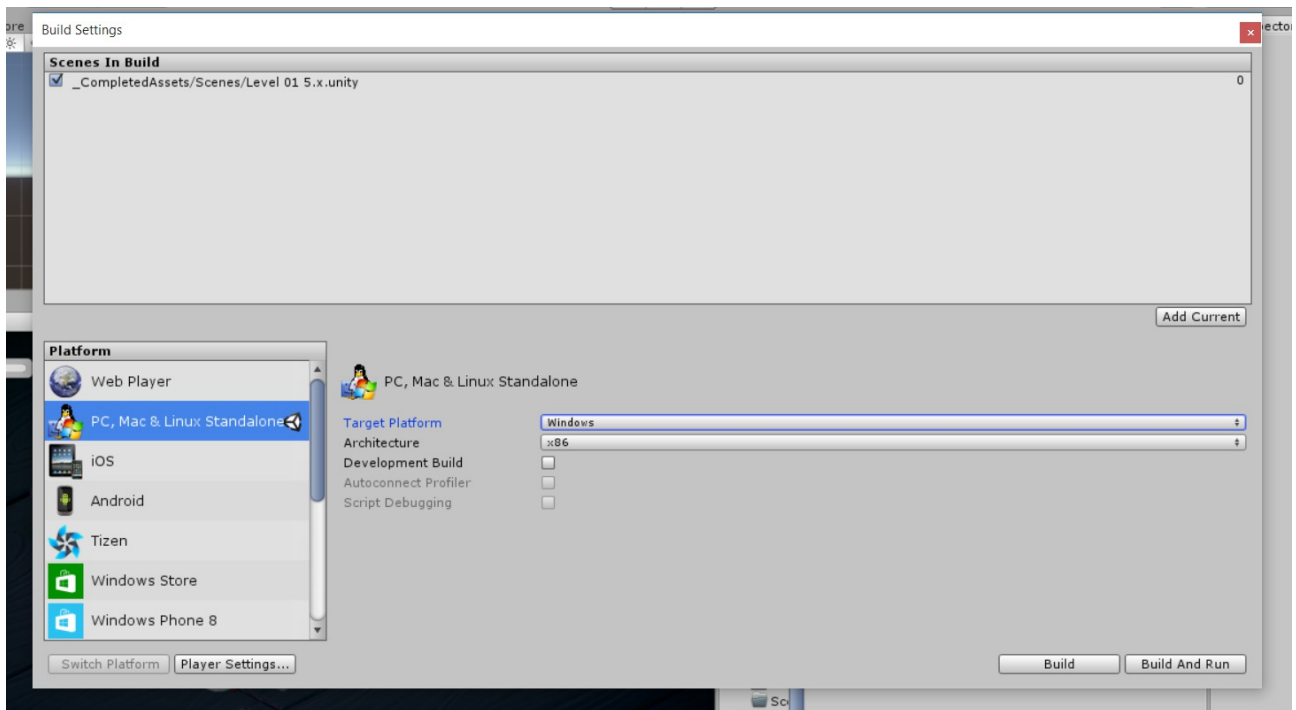
Ouvrir la scène intitulé « level 01 » et lancer le jeu afin de vérifier que tout compile et tout se lance dans l'aperçu de Unity.

## 2. Build

Nous venons de récupérer le code d'un jeu pensé pour desktop. Avant de changer quoi que ce soit dans le code, nous allons compiler et tester ce jeu sur les plateformes suivantes : PC, WebGL (navigateur web) et smartphone Android.

### 2.1 Build sur PC

Aller dans le menu edit/build, sélectionner la plateforme PC et lancer le build.



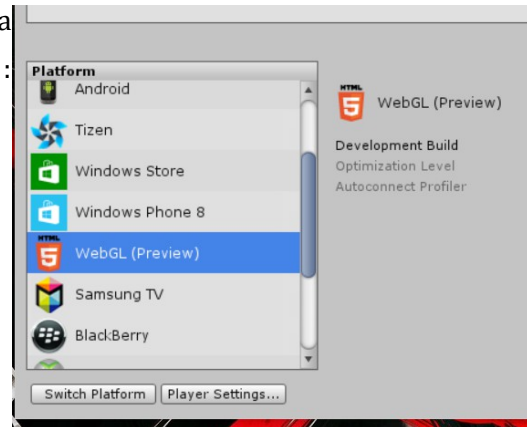
Lancer le .exe obtenu et vérifier que vous pouvez jouer au jeu sans problème particulier :

- les touches WASD (car clavier querty) ou flèche directionnelles pour se déplacer,
- la souris pour orienter le personnage,
- le clic gauche pour tirer,
- echap fait apparaître le menu de pause,
- vous devez entendre la musique du jeu
- etc.

### 2.2 Build sur WebGL (navigateur web)

## Firefox (sur Windows)

Faites comme pour le build PC mais en sélectionnant la plateforme WebGL. Pour plus d'informations sur WebGL : <https://fr.wikipedia.org/wiki/WebGL>



Lancer la page index.html obtenue avec firefox (sur windows) et si vous ne l'avez pas déjà, firefox va vous demander d'installer l'extension Unity Web Player.



Une fois l'extension installé, recharger la page et vérifier que vous pouvez jouer au jeu sans problème particulier.

## Chrome (sur Windows)

Sur Chrome, il y a peu de chances que le jeu se lance. En effet, depuis la version 45, Chrome n'accepte plus les plug'ins utilisant la NP API (pour des questions de sécurité) et il est donc désormais incompatible avec Unity Web Player. Une autre extension uniquement sur Chrome est sortie dans le but de faire fonctionner les jeux Unity via un e redirection du jeux depuis un environnement où il peut être exécuté sans contraintes. Cependant il nous a été impossible de faire fonctionner ce plug'in. <https://support.google.com/chrome/answer/6213033?hl=fr>

## Firefox, Chrome, Chromium (sur Ubuntu)

Inutile d'essayer, Unity Web Player n'est compatible que sur Windows.

## 2.3 Build sur Android

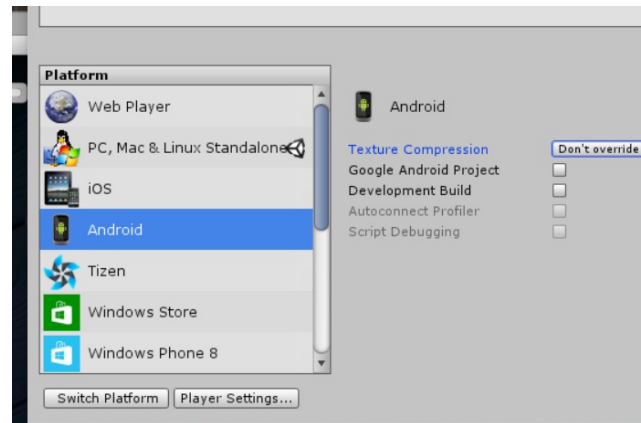
### 2.3.1 Installation du sdk Android

Là l'installation de Unity ne suffit pas, il faut aussi installer le sdk Android et il faut également un périphérique Android à disposition.

Pour installer le sdk : <https://developer.android.com/sdk/index.html>

Windows	<u>installer_r24.4-windows.exe(Recommended)</u>
---------	---

Maintenant essayer de build sur Unity en sélectionnant la plateforme Android.



- si erreur "Android outdated, minimum required is ..."

=> vérifier que vous avez toutes les bonnes API dans le android SDK manager (le lancer depuis tools dans le dossier Android)

(<http://answers.unity3d.com/questions/915567/android-sdk-is-missing-required-platform-api.html>)

=> que le chemin vers l'sdk est bien configuré dans Unity :

Edtit/Preference/ «mettre le bon chemin»

- Si erreur "no USB found"

=> brancher un appareil Android et relancer le build

On obtient un apk.

### 2.3.2 Installation d'un fichier .apk sur un périphérique Android

Déposer le fichier dans un dossier (exemple le dossier download) via un câble USB. Avec une application gestionnaire de fichiers sur le smartphone il faut aller à l'emplacement de l'apk et cliquer dessus pour l'installer comme application.

(Une notification va apparaître pour activer les sources inconnues, il faut le faire).

### 2.3.3 Lancement du jeu

Maintenant lancez le jeu et constater que :

- le jeu se lance en mode paysage
- la résolution n'est pas adapté
- une interaction de type simple touch est interprété comme le clic gauche et permet de tirer
- l'interaction avec le bouton "retour arrière" natif est interprété comme la touche echap et permet d'accéder au menu pause
- il est impossible de se déplacer (effectivement, on ne peut pas utiliser les touches Z,Q,S,D etc.)

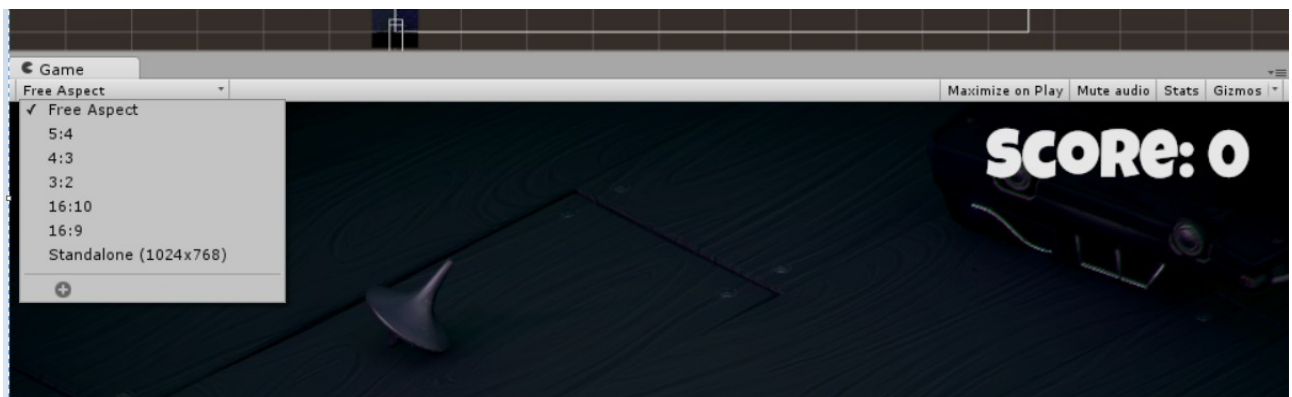
### 3. Tester l'adaptation

#### 3.1 Aperçu

Pour tester notre jeu sur les différentes plateformes, nous commençons par regarder l'aperçu en jeu dans Unity. Attention, cet aperçu n'est pas toujours fiable, vous rencontrerez probablement des surprises lors du déploiement.

#### 3.2 Tester l'affichage

Il est possible de faire de changer la résolution de l'aperçu pour tester le rendu de l'interface.



#### 3.3 Tester le comportement selon une plateforme spécifique

Si le code contient des « platform dependent compilation » (ce sont des sortes d'instructions « if » qui s'appliquent en fonction d'un support spécifique), il est impossible d'observer le comportement du jeu sur une plateforme autre que celle sur laquelle vous codez. Par exemple pour tester le comportement du jeu sur Android (avec les touches etc.) l'aperçu ne suffit pas. Il faut alors soit déployer le jeu, soit utiliser un émulateur.

## 4. Adaptations apportées sur mobile

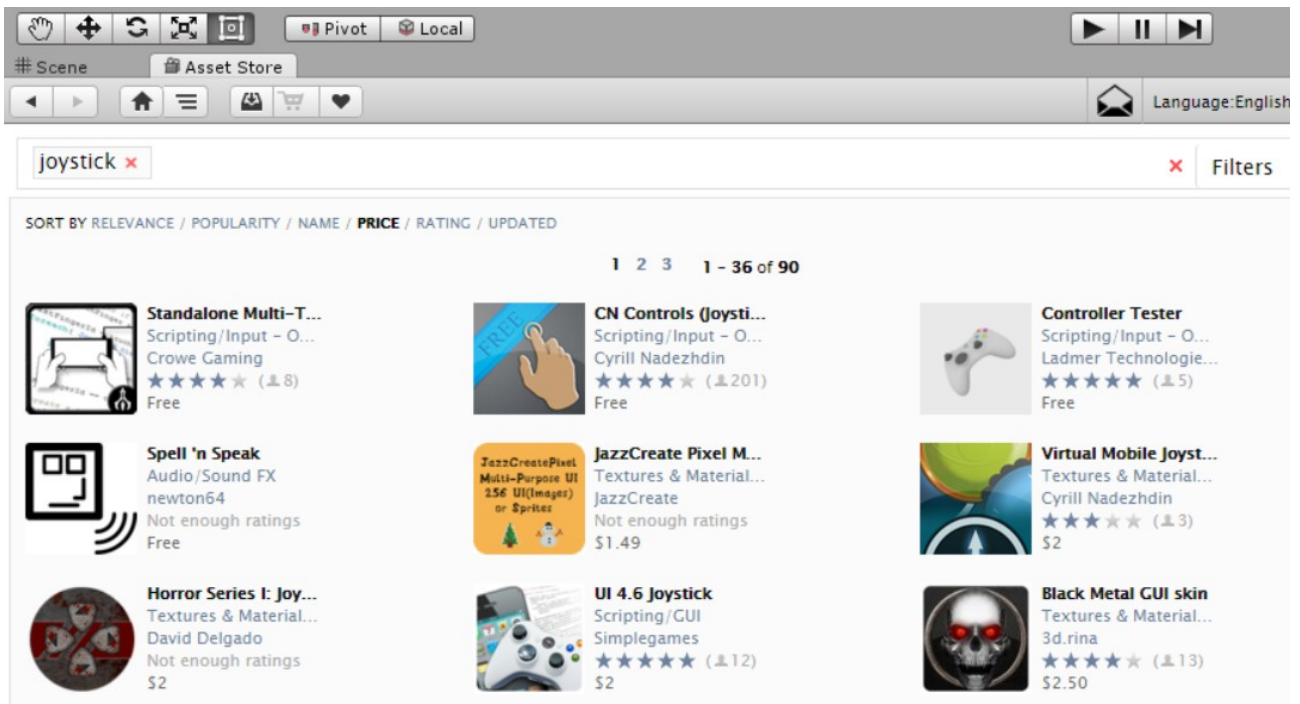
Pour que le jeu soit jouable, on voudrait ajouter pouvoir se déplacer. Il y a plusieurs solutions.

#### 4.1 Mise en place d'un joystick

Comme cela se fait souvent dans l'industrie du jeu vidéo sur mobile, il est possible de mettre en place deux joysticks, un pour le déplacement et un pour l'orientation.

Si vous avez 5\$ à investir, c'est très simple : suivez cette vidéo <https://www.youtube.com/watch?v=UiamR0nr9wg> à partir de 1min.

Sinon, nous avons essayé d'utiliser un joystick gratuit développé par des âmes charitables : (CnControls disponible dans l'assert store ("joystick" / price / free)



Cependant ce joystick ne gère pas le multitouchs et donc il n'est pas possible d'en utiliser deux joysticks à la fois.



Une fois CnControls importé, dans le dossier prefabs, il suffit de drag & dropper le composant joystick dans l'interface du jeu (CanvasHUD) et le placer à l'emplacement désiré (veiller à le placer par rapport une ancre cohérente pour ne pas avoir des surprises après le build). Puis il faut mapper les inputs de contrôle par les inputs fournis.

```
Using CnControls;  
float h = CnInputManager.GetAxis("Horizontal");  
float v = CnInputManager.GetAxis("Vertical");
```

Par ailleurs prenez note que du coup l'action touch sur le mobile effectue un tir.

Pour éviter cela, on peut remplacer le tir sur le clic droit au lieu du clic gauche ou encore revoir complètement le gameplay.

Pour mettre le tir sur le clic droit pour la version desktop, il faut changer dans le script PlayerShooting, «Fire1» par «Fire2» :

```
// If the Fire1 button is being press and it's time to fire...  
if(Input.GetButton ("Fire1") && timer >= timeBetweenBullets && Time.timeScale != 0)
```

## 4.2 Implémenter le déplacement

Pour gérer à la fois le déplacement et l'orientation, nous opterons pour une solution plus simple bien que moins maniable en terme de jouabilité : un touch pour le déplacement et un touch pour l'orientation.



Pour cela il faut modifier le script PlayerMovement et ajouter les interactions avec les touches :

```
void FixedUpdate () {
    // Store the input axes.
    float h = Input.GetAxis("Horizontal");
    float v = Input.GetAxis("Vertical");

    float xPos = Input.GetAxis("Mouse X");
    float yPos = Input.GetAxis("Mouse Y");

    if (Input.touchCount > 0) {
        // mouvement
        xPos = Input.touches[0].deltaPosition.x; // position du premier touch
        yPos = Input.touches[0].deltaPosition.y;

        // orientation
        h = Input.touches[1].deltaPosition.x; // // position du second touch
        v = Input.touches[1].deltaPosition.y;
    }
    Move (xPos, yPos);

    // Turn the player to face the mouse cursor.
    Turning (h, v);

    // Animate the player.
    Animating (h, v);
}
```

Dans tous les cas (solution joystick ou multitouchs), il faut cocher l'option «interactable » du CanvasHUD de la scène.

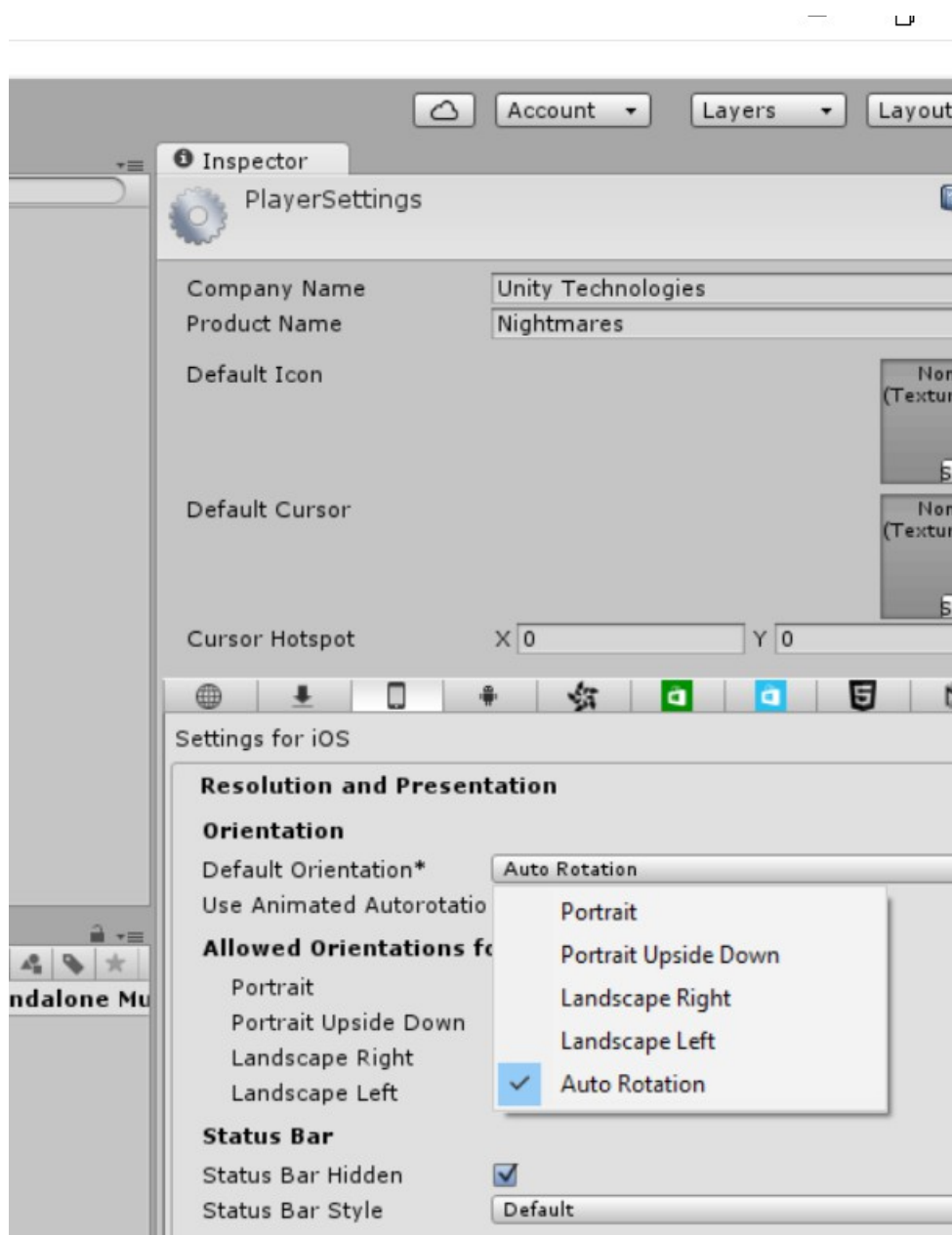
### 4.3 Gérer la résolution

Il est possible de définir une résolution fixe ou de la définir en fonction de la taille de l'écran Screen.width, Screen.height. Puis il faut la mettre à jour avec l'instruction SetResolution :

```
#if UNITY_ANDROID
    Screen.SetResolution(640, 480, true);
#endif
```

### 4.4 Gérer l'orientation

Dans file/build setting, il est possible de modifier l'orientation (landscape, portrait, auto).



## 5. Ajout de touches personnelles (menu de connexion, cinématique)

### 5.1 Ajouter un menu de connexion

- Reprendre le menu pause (dupliquer)
- Supprimer les elements sliders etc.
- Ajouter des inputfields
- Remplacer le script PauseManager par un nouveau script
- Dans le script penser à mettre Time.scale = 0, 1 pour demarrer le jeu au bon moment (clic sur le bouton "connect")

Voici le script :

```
public class ConnexionManager : MonoBehaviour {
    Canvas canvas;
    void Start() {
        canvas = GetComponent<Canvas>();
        canvas.enabled = true;
        Time.timeScale = 0;
    }

    public void Connect() {
        canvas.enabled = false;
        Time.timeScale = 1;
    }

    public void Quit() {
#if UNITY_EDITOR
        EditorApplication.isPlaying = false;
#else
        Application.Quit();
#endif
    }
}
```

### 5.2 Ajouter une cinématique

- Installer quicktime (windows uniquement)

- Relancer Unity3D en mode administrateur
- Créer un game object : plane et lui associer une movie texture  
<http://docs.unity3d.com/Manual/class-MovieTexture.html>

Attention les movies textures n'étant pas supportées sur Android, il faut utiliser du « full-screen streaming » à la place.

<http://docs.unity3d.com/ScriptReference/Handheld.PlayFullScreenMovie.html>

```
void Start() {
    // démarrer la vidéo
    ((MovieTexture)GetComponent<Renderer>().material.mainTexture).Play();
    Handheld.PlayFullScreenMovie ("SurvivalShooterCinematic.mp4" , Color.black,
    FullScreenMovieControlMode.Full);
}

// Update is called once per frame
void Update () {
    // passer la cinématique
    if (Input.anyKeyDown) {
        Renderer r = GetComponent<Renderer>();
        MovieTexture movie = (MovieTexture) r.material.mainTexture;
        if (movie.isPlaying) {
            movie.Stop();
        }
    }
}
```

## **Liens qui nous ont été utile :**

Unity : tutoriel du jeu survival shooter

<https://unity3d.com/learn/tutorials/projects/survival-shooter-project>

Unity : article sur les limites de portages selon les plateformes

<http://docs.unity3d.com/Manual/HOWTO-PortingBetweenPlatforms.html>

Unity : faire de la "platform dependent compilation"

<http://docs.unity3d.com/Manual/PlatformDependentCompilation.html>

Tuto vidéo pour l'ajout d'un joystick pour jouer sur smartphone

<https://www.youtube.com/watch?v=UiamR0nr9wg>

Unity config requirements

<https://unity3d.com/unity/system-requirements>

<http://angezanni.com/unity3d-google-opensim/>

<https://fr.wikipedia.org/wiki/WebGL>

<http://blogs.unity3d.com/2015/08/26/unity-comes-to-linux-experimental-build-now-available/>

<http://www.ubuntuvibes.com/2012/03/how-to-enable-webgl-in-firefox-for.html>

<http://blogs.unity3d.com/2014/10/07/benchmarking-unity-performance-in-webgl/>

<https://blog.mozilla.org/blog/2015/03/03/unity-5-ships-and-brings-one-click-webgl-export-to-legions-of-game-developers/>

# Partie 2 : Polymer

Polymer est un framework récent développé par Google qui permet de faire du web components. Il s'agit d'une recommandation en cours de standardisation par le W3C qui préconise la création et réutilisation de composants HTML indépendants. Polymer est aussi apprécié pour sa facilité à développer en responsive design.

## Configuration :

Ubuntu 14.04 32bits, firefox, google chrome, Nexus 5 (Android 6.0)

## 1. Installation

- installer node.js
- mettre à jour npm : `$ sudo npm install npm -g`
- Installer bower pour les dependences frontend: `$ npm install -g bower` ([Bower](#) requires [node](#), [npm](#) )
- Faire un `$ bower init` qui va créer un fichier bower.json qui contiendra les dépendances frontend
- installer Polymer avec: `$ bower install --save Polymer/polymer#^1.1.0`
  - cela crée un dossier bower\_components qui contient deux dossiers: webcomponentjs et polymer
- lancer un serveur (pour visualisation après déploiement)
  - avec polylserve (pour tester les composants) :
    - `$ npm install -g polylserve` (requires npm-debug.log)
    - se placer dans le dossier de l'élément polymer puis: `$ polylserve`
    - Navigate to localhost:8080/components/my-element/demo.html
  - avec python : `$ sudo apt-get install python`
    - se placer dans le dossier de l'application contenant bower\_components
    - `$ python -m SimpleHTTPServer 8080`
    - Navigate to localhost:8080/dossierApplication

avec le réseau de l'école si erreur "com/Polymer/polymer.git", exit code of #128 fatal: unable to connect to github.com: github.com[0: 192.30.252.129]: errno=Connexion terminée par expiration du délai d'attente"  
=>git config --global url.https://github.com/.insteadOf git://github.com/

## 2. Pour les composants

Pour chaque composant utilisé il faut télécharger le web component associé:

- `bower install --save PolymerElements/paper-elements` (exemple pour les paper-elements)

Tous ces éléments sont placés dans le dossier `bower_components` situé à la racine du projet et dans le fichier `bower.json` ainsi lors du partage du projet on doit seulement faire un `bower install` pour récupérer les bonnes dépendances.

## 3. Utilisation des web-components

- dans le fichier html, importer le lien du fichier html du composant (`bower_components/NOM_COMPOSANT/NOM_COMPOSANT.html`)
- utilisation du composant comme une balise HTML classique
- se documenter sur les styles css disponibles pour le composant (il y a déjà certaines propriétés existantes que l'on peut utiliser, on les retrouve sur la page du composant polymer dans le catalogue, cf image ci-dessous) <https://elements.polymer-project.org/elements/paper-item?active=paper-item-body>

```
<paper-item>
  <paper-item-body two-line>
    <div>Show your status</div>
    <div secondary>Your status is visible to everyone</div>
  </paper-item-body>
</paper-item>
```

The child elements with the `secondary` attribute is given secondary text styling.

### Styling

The following custom properties and mixins are available for styling:

Custom property	Description	Default
<code>--paper-item-body-two-line-min-height</code>	Minimum height of a two-line item	72px
<code>--paper-item-body-three-line-min-height</code>	Minimum height of a three-line item	88px
<code>--paper-item-body-secondary-color</code>	Foreground color for the secondary area	<code>--secondary-text-color</code>
<code>--paper-item-body-secondary</code>	Mixin applied to the secondary area	<code>{}</code>

## 4. Pour le menu

On s'est inspiré des applications natives (facebook, hangouts, skype), menu sur le côté

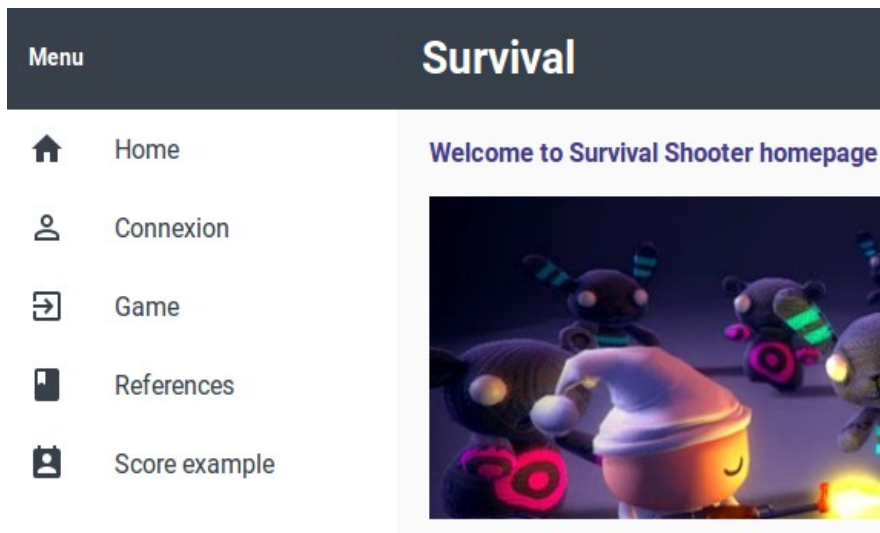
Utilisation des composants:

- “<paper-scroll-header-panel drawer fixed>” pour que lors du scroll le menu ne bouge pas
- “<paper-icon-item>” pour les items du menu
- “<iron-icon icon=“CHOIX\_ICON”>” pour mettre un icône pour l’item
- “<paper-icon-button id=“paperToggle” icon=“menu” paper-drawer-toggle></paper-icon-button>” dans le header de la page pour adaptabilité mobile

Pour faire que le menu soit sous forme de liste sur le côté nous avons utilisé une liste:

- “<div class=“list short” role=“list”>”

Voici l'aperçu web desktop :

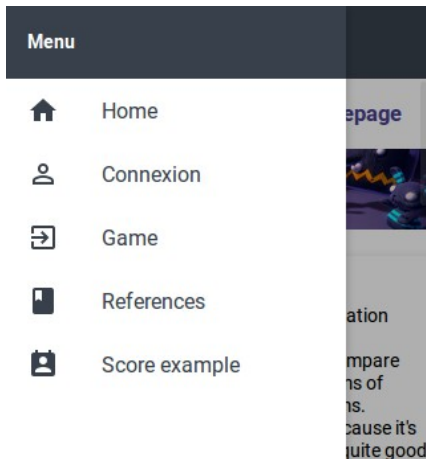


Voici l'aperçu web mobile :





Et voici l'aperçu web mobile après clic sur le bouton toggle :



## 5. Pour les vidéos

Il y a 2 composants pour comparer :

- google-castable-video  
<https://elements.polymer-project.org/elements/google-castable-video>  
ou jquery
- google-youtube  
<https://elements.polymer-project.org/elements/google-youtube>

Pour la vidéo google-youtube: création d'un élément polymer afin de passer les types des paramètres nécessaires au composant (id,playsupported,currenttime,...) ainsi que le paramètre video-id pour permettre au composant de retrouver la vidéo souhaitée.

Pour la google-castable-video pas besoin de créer d'élément polymer, il suffit de mettre les liens de la source de la vidéo dans la balise

```
<video is="google-castable-video">
```

Le déploiement de ces deux composants affecte l'affichage: en effet non déployé les vidéos respectent les styles css qu'on donne, mais en déployant (sur Ubuntu Firefox) la google-castable-video prend tout l'écran et la vidéo youtube ne s'affiche pas. Les iron-image ne s'affichent également pas lors du déploiement sous firefox Ubuntu mais parfaitement bien sous chrome ubuntu. Le résultat sous Android est encore différent: les iron-image s'affichent très bien mais les vidéos sont de taille démesurée et donc inexploitable sur mobile.

## 6. Création d'un élément polymer

- créer un fichier(A) html portant le nom du composant que l'on souhaite créer
- dans ce fichier (A):
  - importer <link rel="import" href="bower\_components/polymer/polymer.html">
  - importer les web-components si nécessaires
  - déclarer le composant comme cela:

```

<dom-module id="demo-element">
  <template id="page-template">
    <style>
      div {
        margin-bottom: 1em;
      }
    </style>
    <google-youtube id="googleYouTube"
      playsupported="{{playSupported}}"
      video-id="UiamR0nr9wg"
      state="{{state}}"|
      currenttime="{{currentTime}}"
      currenttimeformatted="{{currentTimeFormatted}}"
      duration="{{duration}}"
      durationformatted="{{durationFormatted}}"
      fractionloaded="{{fractionLoaded}}"
      on-google-youtube-state-change="handleStateChange"
      on-google-youtube-error="handleYouTubeError">
    </google-youtube>
  </template>
</dom-module>

```

Le dom-module sert à identifier que cet élément est un élément polymer, l'id est le nom de cet élément, c'est le nom de la baliser pour utiliser cet élément. Le fichier html contenant cet élément doit porter le même nom que l'id du dom-module.

- si le composant possède des propriétés telles que un id (par exemple pour l'id de la vidéo):
  - ajouter un script de la forme:

```

<script>
  Polymer({
    is: 'demo-element',
    properties: {
      playSupported: Boolean,
      state: String,
      currentTime: Number,
      currentTimeFormatted: String,
      duration: Number,
      durationFormatted: String,
      fractionLoaded: Number,
      events: {
        type: Array,
        value: []
      }
    },
    handleStateChange: function(ev) {
      this.events.push({data: ev.detail.data});
    },
    handleYouTubeError: function(ev) {
      console.error('YouTube playback error', ev.detail);
    }
  });
</script>

```

La liaison entre le script polymer et les propriétés de l'élément se fait via les accolades.

- dans le fichier (B) dans lequel on souhaite utiliser le composant:
  - importer le fichier correspondant (A)

## 7. Utilisation d'un composant Polymer

Dans le fichier dans lequel vous souhaitez utiliser le composant, utilisez le composant comme une balise HTML classique:

```
<iron-icon icon="book" item-icon></iron-icon> References
```

Puis importer le composant correspondant :

```
<link rel="import" href="bower_components/iron-icons/iron-icons.html">
```

### Pour la connexion :

On a regardé comment était fait les sites mobiles de unity, skype, wikipédia, facebook et openclassroom.

Un bouton/icône de connexion apparaît soit sur la page principale soit dans le menu principal. Quand on clique on accède à une page entière pour la connexion. On a décidé de mettre en place un bouton dans le menu principal.

Composants utilisés pour le formulaire de connexion:

- iron-form qui s'utilise de la façon suivante:

```
<form is="iron-form" id="formGet" method="get" action="/">
  <paper-input name="name" id="name" label="Pseudo" required></paper-input>
  <paper-input name="pwd" id="pwd" label="Password" type="password" required></paper-input>
  <br><br>
  <paper-button raised
    onclick="clickHandler(event)" id="submitConnexion">Submit</paper-button>
</form>
```

- paper-input pour les champs à remplir (pseudo, et mot de passe)
- paper-button pour le submit avec fonction js associée à l'évènement onclick (utilisé comme un bouton normal)

## 8. Déploiement

Pour le déploiement nous avons utiliser Heroku. Nous avons créé un compte sur heroku puis nous avons exécuté les commande suivante afin d'héberger notre projet:

- `wget -O- https://toolbelt.heroku.com/install-ubuntu.sh | sh`
- heroku login (avec l'adresse mail et le mot de passe utilisé pour créer le compte)
- heroku create projet-adaptation (crée un projet heroku)
- git init
- ajout du npm-debug.log dans le gitignore

- git remote -v
- git remote add heroku https://git.heroku.com/projet-adaptation.git
- git add .
- git commit -m "Test"
- git push heroku master (lance le déploiement)
  - déroulement du déploiement:
    - écoute + récupération du push
    - lecture du procfile (détection que c'est du web)
    - exécution des npm install de ce qui est nécessaire (dans le package.json)
    - bower install pour récupérer les dépendances à partir du bower.json

### **Fichiers nécessaires au déploiement :**

- Pour déployer sur Heroku :
  - procfile qui décrit quelle commande heroku doit effectuer pour lancer l'application
  - web.js qui est un script javascript qui décrit le lancement d'un serveur web en node.js nécessaire pour le déploiement sur heroku (il n'est pas nécessaire pour le déploiement sur autre chose)
  - package.json qui décrit les dépendances pour la partie serveur web:
    - gzip (middleware utilisé pour la connexion)
    - express (framework d'applications nodejs)
    - bower

### **Test de l'adaptabilité :**

Une fois le site web déployé, on a la possibilité de le tester sur tous les navigateurs web ainsi que sur téléphone.

## **Liens qui nous ont été utile :**

Pour récupérer les composants des différentes librairies

<https://elements.polymer-project.org/>

Site de Heroku pour le déploiement:

<https://www.heroku.com/>

Simulation de déploiement / tests :

<http://www.alsacreation.com/article/lire/1634-comment-tester-un-site-responsive-partie-1.html>