> **Attention !** L'api utilisé pour la récupération de vidéo n'est maintenant plus fonctionnelle, remplacé par la V3 de Google.

Une extension pour **Google Chrome** où **Chromium** c'est quoi ? Comment fait-on ? Dans cet article je vais essayer de répondre à cette question. Nous allons donc voir les outils nécessaires, le langage qui n'est rien de plus que du HTML/CSS et Javascript et enfin nous finirons par en réalisé une. Article sur l'API V3 de Youtube ici !

Déjà, une extension c'est quoi ?

Une **extension chrome** ces tout simplement une page web qui s'exécute et s'affiche en général dans un **Pop-up** *(Un ou une pop-up, parfois dit fenêtre surgissante est une fenêtre secondaire qui s'affiche dans le navigateur)*. Attention, il ne faut pas **confondre** extension et application. L'application elle utilise le moteur de rendu de chrome et d'affiche dans une fenêtre à part. Comme un logiciel "normal".

Google Chrome étant basé sur le projet Open Source **Chromium** notre extension **fonctionnera sans problème sur les deux navigateurs**.

Les outils:

Un projet Open Source du nom de **Chrome Dev Editor** est disponible depuis peu et permet de simplifier la mise en oeuvre, nous créant les branches de notre extension.

C'est une application Google Chrome, qui va nous permettre de faire d'autres applications ou encore, comme dans notre cas une extension.

L'installation n'a rien de compliqué. La seule condition est d'avoir, comme dit plusieurs fois plus haut Google Chrome ou Chromium: Télécharger. Puis appuyer sur le bouton **+GRATUIT**.

Voilà, nous avons maintenant l'outil nécessaire, n'ayant pas eu beaucoup de retour sur ma façon de présenter le code je vais donc continuer sur cette lancer, n'hésiter pas à faire de part de vos avis et réactions. Pour tester votre application il n'y a rien de plus simple. Dans les paramètres du navigateur activé le **mode développeur** dans le menu *extensions* puis dans notre IDE (Environnement de développement) donc **chrome Dev Editor** cliquer sur la petite flèche en haut, cela installera automatiquement l'extension sur la naviguateur.

Architecture d'une extension:

Une extension comment c'est fait ? C'est tout bonnement une page web **HTML** avec du **CSS**, des "**assets**": images et enfin un fichier **Javascript**. Avec un petit fichier de configuration du nom de **manifest.json**.

Au final une extension n'est rien de plus qu'une page web assez statique. Permettant

l'affichage de données.
La pratique, réalisation de notre extension:
Que va faire notre extension . Et bien rien de bien compliquer, notre extension permettra l'affichage **2 à 3 vidéos d'une chaîne Youtube bien précise**. Donc, nous utiliserons l'**API Youtube**, malheureusement pas la dernière version car je la trouve plus complexe pour un résultat identique. De cette façon nos interlocuteurs pourront en un clique voir si une nouvelle vidéo est disponible.
Je vais prendre comme exemple cette chaîne que je vous conseille au passage: Gamers Addict.

> Extension Gamers Addict pour Google Chrome et Chromium.

La création du projet est assez simple, en haut à gauche cliquer sur les 3 barres puis **new projet**, remplissez le nom de votre projet puis pour **Project type** sélectionner *Javascript chrome App*. Nous voilà maintenant avec l'architecture de notre extension. Dans assets se trouvent nos images, manifest.Json la configuration, **index.html** la page principale qui s'affichera que je vais remplacer par **popup.html**, le fichier CSS et le Javascript **main.js** que je vais aussi renommer pour des questions d'habitude en **popup.js**.

> Architecture de notre extension. (En haut par défaut, en bas la mienne)

Voila le code, commençon par le fichier de configuration **(manifest.json)**:

```
{
  "manifest_version": 2,
  "name": "Gamers Addict",
    "version": "1.0.0",
    "description": "Application non-officiel de la chaîne Youtube
Gamers Addict",
    "browser_action": {
            "default_icon": "assets/icon19.png",
            "default_title": "Gamers Addict",
            "default_popup": "popup.html"
    },
    "icons": {
```

```
            "16": "assets/icon16.png",
            "19": "assets/icon19.png",
          "48": "assets/icon48.png",
          "128": "assets/icon128.png" },
    "permissions": [
    "tabs"
  ],
  "content_security_policy": "script-src 'self'
https://ajax.googleapis.com https://gdata.youtube.com; object-src
'self'"
}
```

C'est notre fichier de configuration, le fichier qu'utilise Chrome lorsqu'il charge l'extension. `manifest_version` est très important, la valeur 2 est obligatoire pour que tout fonctionne sur les nouvelles versions du naviguateur. Ensuite ce n'est que de la configuration basique, **nom** de votre application, **description**. les choses se corsent et encore, c'est un bien grand mot avec `browser_action` qui permet des configurations plus poussées. *default_icon* permet de choisir l'icon en 19×19 pixels qui s'affichera, comme sur la première image plus haut à droite de la barre de recherche. *default_title* indique le nom qui s'affiche lors du survol de l'icon dans le naviguateur et enfin `default_popup` définit la page d'accueil qui sera affichée lors du clique. Pour moi c'est **popup.html**, si vous gardez l'architecture par défaut alors il vous faudra mettre **index.html**. `"icons": {` permet dénumérer les différentes résolutions d'icons. Vous pourrez télécharger ce que j'utilise à la fin de l'article.
Notion importante lorsque l'on développe une extension, toutes les librairies externes comme l'API Youtube, Jquery sont par défaut bloqué, pour des questions de sécurité. Du coup la seule solution est de les autoriser grâce à la ligne suivante `"content_security_policy": "script-src 'self' https://ajax.googleapis.com https://gdata.youtube.com; object-src 'self'"`. Ici nous avons donc JQuery `https://ajax.googleapis.com` et l'API Youtube `https://gdata.youtube.com`. Certaines librairies, même en les ajoutant ici ne sont pas toujours autorisées, pour plus d'informations référez-vous à la documentation de Google sur ce sujet.
Maintenant attaquons le code HTML, je ne vais pas l'expliquer en détail. Le point important

est de ne pas oublier d'inclure les scripts Jquery et Youtube API ainsi que votre fichier JS (pour moi **popup.js**), le manifest ne faisant qu'autoriser leurs inclusions (**popup.html**):

```
<html>
  <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
        <link rel="stylesheet" media="screen" type="text/css"
href="style.css" />
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.1/jquery.min.js
"></script>
        <script src="popup.js"></script>
    </head>
    <body>
      <img style="width: 100%;" alt="header" src="assets/banner.jpg">
      <ul id="link">
        <li><a id="Youtube" href="" title="Youtube"><img alt="Youtube"
src="assets/social/youtube.png"></a></li>
        <li><a id="Google+" href="" title="Google+"><img alt="Google+"
src="assets/social/googleplus.png"></a></li>
        <li><a id="Twitter" href="" title="Twitter"><img alt="Twitter"
src="assets/social/twitter.png"></a></li>
        <li><a id="Facebook" href="" title="Facebook"><img
alt="Facebook" src="assets/social/facebook.png"></a></li>
      </ul>

      <h4>Dernières vidéos:</h4>
      <div id="videos" />
      <script
src="https://gdata.youtube.com/feeds/users/BUROS1ST/uploads?alt=json-i
```

```
n-script&format=5&callback=showVideos"></script>
      <p style="font-size: 10px;">Développer par <a id="footerLink"
href="http://brouilles.github.io/">DEZEIRAUD Gaëtan</a>.</p>
    </body>
</html>
```

Je précise que, dans l'état actuel il est impossible d'utiliser les liens, c'est pour cela que certains `href=""` sont vides, il va nous falloir utilisé Jquery. Pour les personnes n'ayant jamais utilisé l'API Youtube je vous conseille cette vidéo. Sachez juste que nos vidéos s'afficheront dans `<div id="videos" />`.
Je vous mais quand même le CSS, même si dans cette partie il n'y a rien du tout d'extraordinaire. Juste une indication, dans le body le `width: 360px;` indique au naviguateur la largeur de votre popup (**style.css**).

```
body {
  width: 360px;
  margin: 0 0 0 0;
  font-family: "Segoe UI Web Light", "Segoe UI Light", Arial;
}

#link {
  margin-top: 0;
  padding-left: 0;

  width: 100%;
  height: 20px;
  background-color: #d6d6d6;
}

#link li{
  margin-top: 2px;
  list-style-type:none;
```

```
  float: left;
  margin-left: 6px;
}

/* VIDEO CSS */
#videos li {
  list-style-type: none;
  margin-top: 16px;
}

#videos li a {
  text-decoration: none;
  color: black;
}
```

Il ne nous reste plus que le Javascript, c'est le plus gros du travail (**popup.js**):

```
$(function() {

  //Link
  $('#Youtube').click(function() {
     chrome.tabs.create({url:
'https://www.youtube.com/user/BUROS1ST'});
  });
  $('#Google+').click(function() {
     chrome.tabs.create({url:
'https://plus.google.com/u/0/+BUROS1ST'});
  });
  $('#Twitter').click(function() {
     chrome.tabs.create({url: 'https://twitter.com/Buros1st'});
  });
  $('#Facebook').click(function() {
```

```
    chrome.tabs.create({url: 'https://www.facebook.com/burosdmo'});
  });

  //Footer
  $('#footerLink').click(function() {
    chrome.tabs.create({url:
'http://dezeiraudgaetan.azurewebsites.net/'});
  });
});

//YOUTUBE API v2
function showVideos(data) {
  var urlArray = [];

  var feed = data.feed;
  var entries = feed.entry || [];

  var html = ['<ul>'];
  for (var i = 0; i < 3; i++) {
    var entry = entries[i];
    var title = entry.title.$t;
    var thumbnail = entry.media$group.media$thumbnail[0].url;
    var videoUrl = entry.media$group.media$player[0].url;

    urlArray.push(videoUrl);
    html.push('<li> <a id="videoLink'+i+'" href=""><img style="width:
260px;" alt="thumbnail" src="', thumbnail,'"> <br/>',
title,'</a></li>');
  }
  html.push('</ul>');
```

```
  document.getElementById('videos').innerHTML = html.join('');

  //Add click Event
  $(function() {
    $('#videoLink0').click(function() {
          chrome.tabs.create({url: urlArray[0]});
    });

    $('#videoLink1').click(function() {
          chrome.tabs.create({url: urlArray[1]});
    });

    $('#videoLink2').click(function() {
          chrome.tabs.create({url: urlArray[2]});
    });
  });
}

document.addEventListener('DOMContentLoaded');
```

Je ne vais pas m'attarder sur l'utilisation de l'API Youtube version 2 (la version 3 est sortie il y a déjà quelque temps) mais surtout sur l'ouverture des liens. Comme je les dis plus haut, par défaut un lien ne s'ouvre pas. Il nous faut donc géré manuellement avec une fonction Javascript propre à Chrome et Chromium `chrome.tabs.create`. Comme vous pouvez le voir au début du code, chaque lien de notre extension doit être défini ici. Prenons cet exemple:

```
$('#Youtube').click(function() {
    chrome.tabs.create({url:
'https://www.youtube.com/user/BUROS1ST'});
```

Si vous avez déjà fait du Jquery vous ne serez je pense pas dépaysé. Nous donnons l'ID de notre lien à Jquery `$('#Youtube')`, et lors du clique `click(function() ...` la fonction

magique `chrome.tabs.create` est utilisé et ouvre un nouveau tab/onglet. Comme vous pouvez le voir dans le code de l'API Youtube:

```
$('#videoLink1').click(function() {
      chrome.tabs.create({url: urlArray[1]});
});
```

Il y a possibilité de renseigner nos liens dans un array, un tableau dynamique. J'ai dû faire de cette façon car les liens vers les vidéos dépendent, tout simplement de la vidéo et sont donc à chaque fois quasiment différents. Les déclarer de façon static comme je les fais plus haut avec twitter est Facebook est donc impossible.
Voilà notre extension est maintenant terminée. Mais ce n'est pas fini, nous allons voir comment la publier sur le chrome Web store.
Publiez et partagez notre application:
Maintenant que notre extension est prête nous allons voir comment la publier. Avant toute chose il vous faut un compte Google et je préviens, la publication n'est malheureusement pas gratuite. Il vous faudra vous acquitter de 5.00$ et vous pouvez publier jusqu'à 20 articles(application/extension). Après rien de bien compliquer, connectez-vous sur le chrome Web store, puis en haut à droite l'icon en forme rouage puis **Tableau de bord du développeur**. Ensuite il vous suffit une fois sur cette page (voir image) d'appuyer sur **Ajouter un nouvel article**, la suite de la procédure vous est expliquée. Il vous suffira d'envoyer votre projet en .zip et le store s'occupera du reste.
Voilà, nous en avons fini avec le développement d'extension pour Google Chrome. Vous pouvez retrouver le code source plus bas ainsi qu'un lien vers la documentation en anglais de Google.

- Documentation Google.
- Télécharger le projet et le code source.