Digiteka

PLAYER DIGITEKA
API IOS & ANDROID SANS SDK

Sommaire

Table des matières

1. Intégration du Player Simple dans IOS

a. Rajouter la webview à votre vue

Ajouter une WebKit View sur la vue parent et adopter les protocoles "WKNavigationDelegate" et "WKUIDelegate".

b. Paramétrer la webview

Pour un fonctionnement optimal du Player Digiteka, il est nécessaire de respecter les étapes suivantes dans l'objet WKWebViewConfiguration.

Ce paramétrage peut être fait dans le code ou dans interface builder (iOS 11 minimum)

Dans le code :

1- créer un objet WKWebViewConfiguration

2- paramétrer l'objet WKWebViewConfiguration

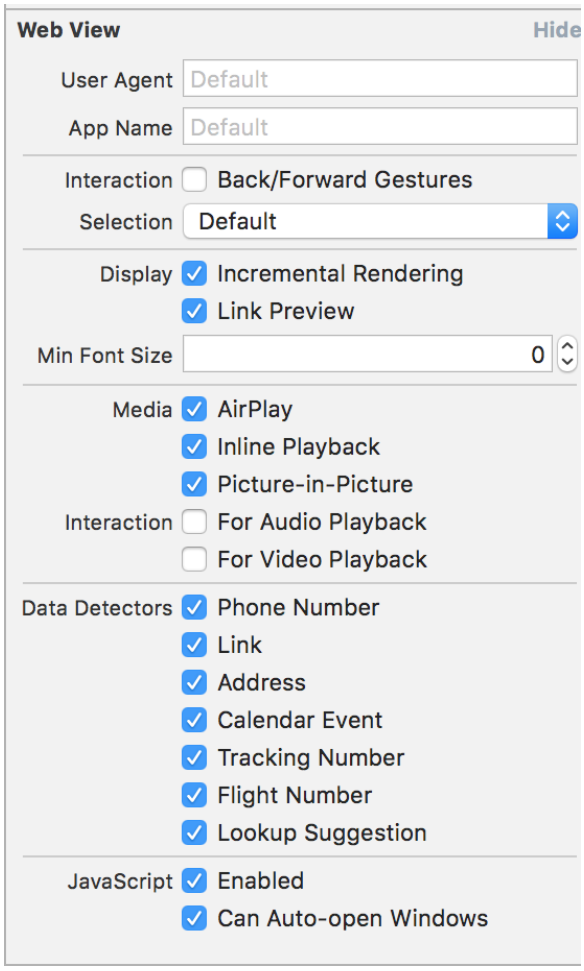
3- Initialiser la WKWebView avec l'objet WKWebViewConfiguration

```
//1
WKWebViewConfiguration *webViewConfiguration =
[[WKWebViewConfiguration alloc] init];

// 2
webViewConfiguration.requiresUserActionForMediaPlayback = NO;
    [webViewConfiguration setAllowsInlineMediaPlayback: YES];
    [webViewConfiguration setWebsiteDataStore: [WKWebsiteDataStore
nonPersistentDataStore]];
    [[webViewConfiguration preferences]
setJavaScriptCanOpenWindowsAutomatically: YES];
    [[webViewConfiguration preferences] setJavaScriptEnabled:YES];

// 3
self.webView = [[WKWebView alloc] initWithFrame:webViewFrame
configuration:webViewConfiguration];
```

```
// 1
let webViewConfiguration = WKWebViewConfiguration()

// 2
if #available(iOS 10.0, *) {
  webViewConfiguration.mediaTypesRequiringUserActionForPlayback = []
```

```
} else {
  webViewConfiguration.requiresUserActionForMediaPlayback = false
}

webViewConfiguration.allowsInlineMediaPlayback = true
webViewConfiguration.websiteDataStore = WKWebsiteDataStore.nonPersistent()
webViewConfiguration.preferences.javaScriptCanOpenWindowsAutomatically =
true
webViewConfiguration.preferences.javaScriptEnabled = true

// 3
webView = WKWebView(frame: webViewFrame, configuration:
webViewConfiguration)
```

Dans interface builder :



c. Interactions avec la webview

i. Chargement

Il faut charger le contenu de la WebView en chargeant directement l'url d'une vidéo ,

Ex pour cette vidéo :
https://www.ultimedia.com/deliver/generic/iframe/mdtk/01946093/src/ss5rl5/zone/2/showtitle/1/

```
NSURL *baseURL = [[NSURL alloc] initWithString:
@"http://www.ultimedia.com/"];
NSString *videoURL = [[NSURL alloc] initWithString: @"http://um-
web53.dginfra.net:8082/player/iframe?mdtk=01946093&width=533&height=
300&zone=2&src=ukll3r"];
NSURLRequest *videoUrlRequest = [NSURLRequest alloc] initWithURL:
videoURL];
[self.webView loadRequest: videoUrlRequest];
```

```
if let video = URL(string: "http://um-
web53.dginfra.net:8082/player/iframe?mdtk=01946093&width=533&height=
300&zone=2&src=ukll3r") {
     let urlRequest = URLRequest(url: video)
     webView.load(urlRequest)
}
```

ii. Gestion de tap sur la vidéo

Pour pouvoir ouvrir la publicité en tapant sur la vidéo, il faut implémenter deux méthodes:

1- WKNavigation delegate

```
- (void)webView:(WKWebView *)webView
decidePolicyForNavigationAction:(WKNavigationAction
*)navigationAction decisionHandler:(void
(^)(WKNavigationActionPolicy))decisionHandler
{
   return decisionHandler(WKNavigationActionPolicyAllow);
}
```

```
public func webView(_ webView: WKWebView, decidePolicyFor
navigationAction: WKNavigationAction, decisionHandler: @escaping
(WKNavigationActionPolicy) -> Void) {

     decisionHandler(WKNavigationActionPolicy.allow)
}
```

2- WKUI delegate

```
- (BOOL)webView:(UIWebView *)webView
shouldStartLoadWithRequest:(NSURLRequest *)request
navigationType:(UIWebViewNavigationType)navigationType {
NSURL *url = [request URL];
        if ([[url scheme] isEqualToString:@"http"] || [[url scheme]
isEqualToString:@"https"]) {
            [[UIApplication sharedApplication] openURL:url];
            return NO;
        }
    return TRUE;
}
```

```
public func webView(_ webView: WKWebView, createWebViewWith
configuration: WKWebViewConfiguration, for navigationAction:
WKNavigationAction, windowFeatures: WKWindowFeatures) -> WKWebView?
{

     guard let url: URL = navigationAction.request.url, let
urlScheme = url.scheme else { return nil }

     if (urlScheme == "http" || urlScheme == "https") {
          UIApplication.shared.openURL(url)
          return nil
```

```
        }

        return nil
}
```

iii. Supprimer le player vidéo

Détruire la webview.

```
[webView removeFromSuperview];
```

```
webview.removeFromSuperview()
```

d. Interaction avec le Player Digiteka

Pour intéragir avec le player vidéo il faut envoyer des messages en Javascript a la webview. Pour cela on utilise la fonction

```
- (void)evaluateJavaScript:(NSString *)javaScriptString
         completionHandler:(void (^)(id, NSError
*error))completionHandler;
```

```
func evaluateJavaScript(_ javaScriptString: String,
completionHandler: ((Any?, Error?) -> Void)? = nil)
```

Play command

```
[self.webView evaluateJavaScript:@"dtkPlayer.play();"
completionHandler:nil];
```

```
webView.evaluateJavaScript("dtkPlayer.play();", completionHandler:
nil)
```

Pause command

```
[self.webView evaluateJavaScript:@"dtkPlayer.pause();"
completionHandler:nil];
```

```
webView.evaluateJavaScript("dtkPlayer.pause();", completionHandler:
nil)
```

Stop command

```
[self.webView evaluateJavaScript:@"dtkPlayer.stop();"
completionHandler:nil];
```

```
webView.evaluateJavaScript("dtkPlayer.Stop();", completionHandler:
nil)
```

Reload Player command

```
[self.webView evaluateJavaScript:@"dtkPlayer.reload();"
completionHandler:nil];
```

```
webView.evaluateJavaScript("dtkPlayer.reload();", completionHandler:
nil)
```

Mute command

```
[self.webView evaluateJavaScript:@"dtkPlayer.toggleMute(true,
true);" completionHandler:nil];
```

```
webView.evaluateJavaScript("dtkPlayer.toggleMute(true, true);",
completionHandler: nil)
```

Unmute command

```
[self.webView evaluateJavaScript:@"dtkPlayer.toggleMute(false,
true);" completionHandler:nil];
```

```
webView.evaluateJavaScript("dtkPlayer.toggleMute(false, true);",
completionHandler: nil)
```

2. Intégration du Smart Player dans IOS

a. Rajouter la webview à votre vue

Ajouter une WebKit View sur la vue parent et adopter les protocoles "WKNavigationDelegate" et "WKUIDelegate".

b. Paramétrer la webview

Pour un fonctionnement optimal du Player Digiteka, il est nécessaire de respecter les étapes suivantes dans l'objet WKWebViewConfiguration.

Ce paramétrage peut être fait dans le code ou dans interface builder (iOS 11 minimum)

Dans le code :

1- créer un objet WKWebViewConfiguration

2- paramétrer l'objet WKWebViewConfiguration

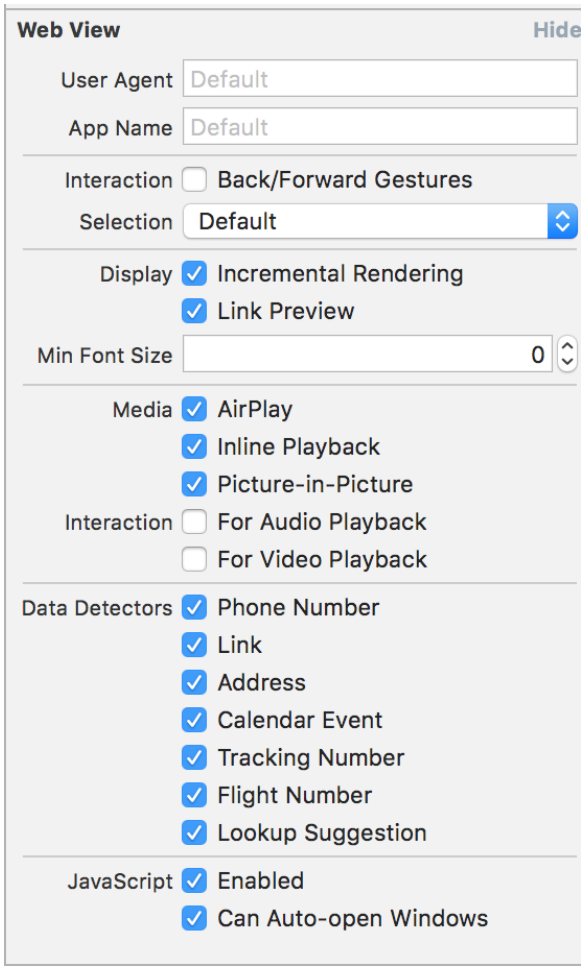
3- Initialiser la WKWebView avec l'objet WKWebViewConfiguration

```
//1
WKWebViewConfiguration *webViewConfiguration =
[[WKWebViewConfiguration alloc] init];

// 2
webViewConfiguration.requiresUserActionForMediaPlayback = NO;
    [webViewConfiguration setAllowsInlineMediaPlayback: YES];
    [webViewConfiguration setWebsiteDataStore: [WKWebsiteDataStore
nonPersistentDataStore]];
    [[webViewConfiguration preferences]
setJavaScriptCanOpenWindowsAutomatically: YES];
    [[webViewConfiguration preferences] setJavaScriptEnabled:YES];

// 3
self.webView = [[WKWebView alloc] initWithFrame:webViewFrame
configuration:webViewConfiguration];
```

```
// 1
let webViewConfiguration = WKWebViewConfiguration()

// 2
if #available(iOS 10.0, *) {
  webViewConfiguration.mediaTypesRequiringUserActionForPlayback = []
} else {
  webViewConfiguration.requiresUserActionForMediaPlayback = false
}

webViewConfiguration.allowsInlineMediaPlayback = true
```

```
webViewConfiguration.websiteDataStore = WKWebsiteDataStore.nonPersistent()
webViewConfiguration.preferences.javaScriptCanOpenWindowsAutomatically =
true
webViewConfiguration.preferences.javaScriptEnabled = true

// 3
webView = WKWebView(frame: webViewFrame, configuration:
webViewConfiguration)
```

Dans interface builder :



c. Interactions avec la webview

i. Chargement

1- Il faut construire une page web simple avec une div nommée “ultimedia_wrapper" et charger cette url dans la webView

2- Définir la commande Javascript d’insertion du Smart Player

3- Il faut injecter le smart player dans cette div a l’aide de la commande Javascript, une fois la page chargée

```
// 1
NSString* htmlPage = @"<!DOCTYPE HTML><html><head><meta
charset='UTF-8'><meta name='viewport' content='width=device-width,
initial-scale=1' id='mvp'/></head><body><div
id='ultimedia_wrapper'></div></body></html>";

NSURL *baseURL = [[NSURL alloc] initWithString:
@"www.ultimedia.com"];

[self.webView loadHTMLString:htmlPage baseURL:baseURL];

// 2
NSString *insertCommand = @"javascript:(function() { var element =
document.getElementById('ultimedia_wrapper'); ULTIMEDIA_target =
'ultimedia_wrapper'; ULTIMEDIA_mdtk = '01124706'; ULTIMEDIA_zone =
'18'; ULTIMEDIA_async = 'true'; var scriptURL =
'https://www.ultimedia.com/js/common/smart.js'; var head =
document.getElementsByTagName('head')[0]; var script =
document.createElement('script'); script.type = 'text/javascript';
script.src = scriptURL; script.onload = function() {};
head.appendChild(script);})()"

// 3
- (void)widgetDidFinishLoad:(UltimediaWidget *)widget {
    [self.webView evaluateJavaScript:insertCommand
completionHandler:nil];
}
```

```
// 1
let htmlPage = "<!DOCTYPE HTML>" +
            "<html>" +
            "<head>" +
```

```
            "<meta charset='UTF-8'>" +
            "<meta name='viewport' content='width=device-width,
initial-scale=1' id='mvp'/>"+
            "</head>" +
            "<body>" +
            "<div id='ultimedia_wrapper'></div>" +
            "</body>" +
            "</html>"

if let baseURL: URL = URL.init(string: "www.ultimedia.com") {
     webView.loadHTMLString(htmlPage, baseURL: baseURL)
}

// 2
let insertCommand = "javascript:(function() { " +
        "var element =
document.getElementById('ultimedia_wrapper');" +
        "ULTIMEDIA_target = 'ultimedia_wrapper';" +
        "ULTIMEDIA_mdtk = '01124706';" +
        "ULTIMEDIA_zone = '18';" +
        "ULTIMEDIA_async = 'true';" +
        "var scriptURL =
'https://www.ultimedia.com/js/common/smart.js';" +
        "var head = document.getElementsByTagName('head')[0];" +
        "var script = document.createElement('script');" +
        "script.type = 'text/javascript';" +
        "script.src = scriptURL;" +
        "script.onload = function() { " +
        "};" +
        "head.appendChild(script);" +
    "})()"

//3
func webView(_ webView: WKWebView, didFinish navigation:
WKNavigation!) {
     webView.evaluateJavaScript(insertCommand, completionHandler:
nil)
}
```

ii. Gestion de tap sur la vidéo

Pour pouvoir ouvrir la publicité en tapant sur la vidéo, il faut implémenter deux méthodes:

1- WKNavigation delegate

```
- (void)webView:(WKWebView *)webView
decidePolicyForNavigationAction:(WKNavigationAction
*)navigationAction decisionHandler:(void
(^)(WKNavigationActionPolicy))decisionHandler
{
   return decisionHandler(WKNavigationActionPolicyAllow);
}
```

```
public func webView(_ webView: WKWebView, decidePolicyFor
navigationAction: WKNavigationAction, decisionHandler: @escaping
(WKNavigationActionPolicy) -> Void) {

     decisionHandler(WKNavigationActionPolicy.allow)
}
```

2- WKUI delegate

```
- (BOOL)webView:(UIWebView *)webView
shouldStartLoadWithRequest:(NSURLRequest *)request
navigationType:(UIWebViewNavigationType)navigationType {
NSURL *url = [request URL];
        if ([[url scheme] isEqualToString:@"http"] || [[url scheme]
isEqualToString:@"https"]) {
            [[UIApplication sharedApplication] openURL:url];
            return NO;
        }
    return TRUE;
}
```

```
public func webView(_ webView: WKWebView, createWebViewWith
configuration: WKWebViewConfiguration, for navigationAction:
WKNavigationAction, windowFeatures: WKWindowFeatures) -> WKWebView?
{

     guard let url: URL = navigationAction.request.url, let
urlScheme = url.scheme else { return nil }

     if (urlScheme == "http" || urlScheme == "https") {
          UIApplication.shared.openURL(url)
          return nil
```

```
        }

        return nil
}
```

iii. Supprimer le player vidéo

Détruire la webview.

```
[webView removeFromSuperview];
```

```
webview.removeFromSuperview()
```

d. Interaction avec le Player Digiteka

Pour intéragir avec le player vidéo il faut envoyer des messages en Javascript a l'iframe de la webview. contenant le Smart Player. Pour cela on utilise la fonction

```
- (void)evaluateJavaScript:(NSString *)javaScriptString
         completionHandler:(void (^)(id, NSError
*error))completionHandler;
```

```
func evaluateJavaScript(_ javaScriptString: String,
completionHandler: ((Any?, Error?) -> Void)? = nil)
```

Play command

```
NSString *playCommand = @"var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('play','*');";

[self.webView evaluateJavaScript:playCommand completionHandler:nil];
```

```
let playCommand = "var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('play','*');"

webView.evaluateJavaScript(playCommand, completionHandler: nil)
```

Pause command

```
NSString *pauseCommand = @"var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('pause','*');";

[self.webView evaluateJavaScript:pauseCommand
completionHandler:nil];
```

```
let pauseCommand = "var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('pause','*');"

webView.evaluateJavaScript(pauseCommand, completionHandler: nil)
```

Stop command

```
NSString *stopCommand = @"var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('stop','*');";

[self.webView evaluateJavaScript:stopCommand completionHandler:nil];
```

```
let stopCommand = "var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('stop','*');"

webView.evaluateJavaScript(stopCommand, completionHandler: nil)
```

Reload Player command

```
NSString *reloadCommand = @"var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('reload','*');";

[self.webView evaluateJavaScript:reloadCommand
completionHandler:nil];
```

```
let reloadCommand = "var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('reload','*');"

webView.evaluateJavaScript(reloadCommand, completionHandler: nil)
```

Switch Mute command

```
NSString *switchMuteCommand = @"var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('mute','*');";

[self.webView evaluateJavaScript:switchMuteCommand
completionHandler:nil];
```

```
let switchMuteCommand = "var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('mute','*');"

webView.evaluateJavaScript(switchMuteCommand, completionHandler:
nil)
```

Mute command

```
NSString *muteCommand = @"var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('mute=1','*');";

[self.webView evaluateJavaScript:muteCommand completionHandler:nil];
```

```
let muteCommand = "var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('mute=1','*');"

webView.evaluateJavaScript(muteCommand, completionHandler: nil)
```

Unmute command

```
NSString *unmuteCommand = @"var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('mute=0','*');";

[self.webView evaluateJavaScript:unmuteCommand
completionHandler:nil];
```

```
let unmuteCommand = "var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
```

```
    "player.contentWindow.postMessage('mute=0','*');"

webView.evaluateJavaScript(unmuteCommand, completionHandler: nil)
```

Set current time command

```
NSString *setCurrentCommand = @"var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +
    "player.contentWindow.postMessage('setCurrentTime=20','*');";

[self.webView evaluateJavaScript:setCurrentCommand
completionHandler:nil];
```

```
// Positionner la vidéo a timeToSet
let timeToSet = 20 // Temps en secondes
let setCurrentCommand = "var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');" +

"player.contentWindow.postMessage('setCurrentTime=\(timeToSet)','*')
;"

webView.evaluateJavaScript(setCurrentCommand, completionHandler:
nil)
```

3. Intégration du Player Simple dans Androïd sans SDK

a. Ajouter le droit d'accéder à internet à votre application

Pour autoriser l'accès à internet modifier le fichier AndroidManifest.xml

```
<uses-permission android:name="android.permission.INTERNET"/>
```

b. Rajouter la webview à votre vue

```
<WebView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/idDtkWebView"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignParentRight="true"
```

```
        android:layout_alignParentEnd="true" />
```

c. Paramétrer la webview

Il faut activer certaines options pour faire fonctionner le Player Digiteka de façon optimale, comme le JavaScript, les cookies

Pour un fonctionnement optimal du Player Digiteka, il est nécessaire d'activer les options suivantes dans la webview

Ex pour cette vidéo : http://www.ultimedia.com/default/index/videogeneric/id/ss5rl5

(l'url ci-dessous est celle contenue dans le code de l'iframe de la video)

```
final WebView dtkWebView = (WebView)
findViewById(R.id.idDtkWebView);
dtkWebView.getSettings().setJavaScriptCanOpenWindowsAutomatically(tr
ue);
dtkWebView.getSettings().setJavaScriptEnabled(true);
dtkWebView.getSettings().setUseWideViewPort(true);
dtkWebView.getSettings().setLoadWithOverviewMode(true);
dtkWebView.getSettings().setDomStorageEnabled(true);
dtkWebView.setWebViewClient(new WebViewClient());
dtkWebView.loadUrl("http://www.ultimedia.com/deliver/generic/iframe/mdtk/01637594/src/ss5rl5/z
one/1/showtitle/1/");
```

d. Interaction avec la webview

Reload

Recharger la webview

```
webview.reload()
```

Destroy

Détruire la webview.

```
webview.destroy()
```

e. Interaction avec le Player Digiteka

Important : Ces fonctionnalités sont disponibles uniquement de les versions Android 4.2 et ultérieures.

Play

Permet de lancer le visionnage de la vidéo.

```
webview.loadUrl("javascript:dtkPlayer.play();");
```

Stop

Permet d'arrêter la vidéo

```
webview.loadUrl("javascript:dtkPlayer.stop();");
```

Reload

Cette fonction recharge le Player

```
webview.loadUrl("javascript:dtkPlayer.reload();");
```

Sound ON

Permet de remettre le son.

```
webview.loadUrl("javascript:dtkPlayer.toggleMute(false, true);");
```

Sound OFF

Permet de couper le son

```
webview.loadUrl("javascript:dtkPlayer.toggleMute(true, true);");
```

4. Intégration du Smart player dans Android sans SDK

Ajouter le code ci-dessous dans le webview avec l'id du target div **ultimedia_wrapper** qui sera utilisé pour faire embed de code smart.

```
String targetDiv="" +
"<!DOCTYPE html><html>" +
"<head>" +
"<meta name='viewport' content='width=device-width, initial-scale=1'
id='mvp' />"+
"</head>" +
"<body>" +
"<div id='ultimedia_wrapper'></div>" +
"</body>" +
"</html>";
dtkWebView.loadData(targetDiv, "text/html; charset=utf-8", "UTF-8");
```

Apres ce code html est chargé, faire embed de code smart en utilisant le code suivant.

```
dtkWebView.setWebViewClient(new WebViewClient(){
// Pour permettre d'ouvrir la pub dans un nouvel onglet
public boolean shouldOverrideUrlLoading(WebView view, String url) {
view.getContext().startActivity(
new Intent(Intent.ACTION_VIEW, Uri.parse(url)));
return true;
}
public void onPageFinished(WebView view, String url) {
super.onPageFinished(view, url);
dtkWebView.loadUrl("javascript:" +
"var ULTIMEDIA_target = 'ultimedia_wrapper';"+
"var ULTIMEDIA_mdtk = '01124706';"+
"var ULTIMEDIA_zone = '18';"+
"var ULTIMEDIA_async = 'true';"+
"var ULTIMEDIA_search = 'titre de ma page à chercher';"+
"var scriptURL = 'https://www.ultimedia.com/js/common/smart.js';"+
"var script = document.createElement('script');"+
"script.type = 'text/javascript';"+
"script.src = scriptURL;"+
"var head = document.getElementsByTagName('head')[0];"+
"head.appendChild(script);"+
}
});
```

Apres le widget est chargé complètement, le player nous envoie des messages pour informer le statut de player. On peut l'utiliser pour envoyer un post message pour contrôler le player.

Par exemple, pour executer Play sur le player apres on reçois le message 'event=ready',utiliser le javascript ci-dessous.

```
dtkWebView.loadUrl("javascript:" +
      "window.addEventListener('message', function (e) { " +
      "if (typeof e.data == 'string' && e.data == 'event=ready') {"
+
      "var player =
document.getElementById('um_ultimedia_wrapper_iframeUltimedia');"+
     "player.contentWindow.postMessage('play','*');"+
     "}" +
     "}, false);"+
     "");
```

Attention: l'id de l'iframe um_ultimedia_wrapper_iframeUltimedia peut être modifié si le nom de target div est changé. Dans ce cas,il faut remplacer ultimedia_wrapper par l'id de target div utilisé.

a. Message envoyé par Iframe.

Voici la liste d'evenments envoyés par notre player.

En écoutant les message reçus, il est possible de détecter certains événements.

Instruction pour ajouter un listener sur les postMessage :

```
window.addEventListener("message", function (message) { /*your
implementation*/ }, false);
```

Possible valeur pour **message.data** :

playerevent_onSetupError_{"code":\d,"message":"\s"}

Emis lorsqu'une erreur survient au niveau du player. \d sera remplacé par le code de l'erreur et \s par le message associé.

event=ready

Emis lorsque le player est prêt

event=ended

Emis lorsque la video arrive à son terme.

event=AdPlay

Envoyé à la reprise de la lecture de la publicité (mais pas à l'impression)

event=AdPause

Emis lors de la mise en pause de la publicité.

event=AdImpression

Emis lorsque la pub commence

event=AdError

Emis lorsque la publicité n'a pu être chargée

event=AdComplete

Emis à la fin de la pub (non émis si skip)

event=paused

Emis lorsque la video est mise en pause

event=played

Emis lorsque la video commence (donc après la pub) ou quand la video reprend (après une pause)

event=resize

Emis lorsque le player est redimensionné.

b. Communication avec l'iframe avec la post message.

En ecoutant les messages d'iframe, on peut passer la commande à Player via postMessage suivants

```
document.getElementById("idIframe").contentWindow.postMessage(message, '*');
```

Les messages possibles sont les suivants :

play

Demande la lecture de la vidéo (lance le preroll si programmé)

pause

Met en pause la vidéo

mute

Basculement de l'état du mute (vidéo ou pub)

mute=1

Force le player en muet (vidéo ou pub)

mute=0

Réactive le son du player (vidéo ou pub)

getCurrentTime

Demande la position actuelle de lecture. Le postMessage event=getCurrentTime&time=\f sera émis en réponse (\f sera remplacé par la position dans la vidéo)

getDuration

Demande la durée de la video. Le postMessage event=getDuration&duration=\f sera émis en réponse. (\f sera remplacé par la durée de la vidéo)

setCurrentTime=\f

Déplace la position de lecture à \f

exemple : setCurrentTime=15