Introduction

Contexte

La solution iDol permet de configurer des Scénarios correspondant à l'exécution sur borne tactile d'un parcours client incitant un utilisateur à remplir un questionnaire et laisser ses coordonnées. Les Scénarios se composent :

- d'une vidéo d'introduction optionnelle
- d'un module optionnel d'enrichissement de l'expérience
- d'un écran d'accueil Landing Page optionnel
- d'un questionnaire comprenant a minima le recueil nom/prénom et un moyen de contact (email ou mobile)
- d'un écran de fin Landing Page optionnel comprenant un bouton d'envoi des résultats

Ce guide concerne le développement de modules optionnels intervenant à l'étape 2.

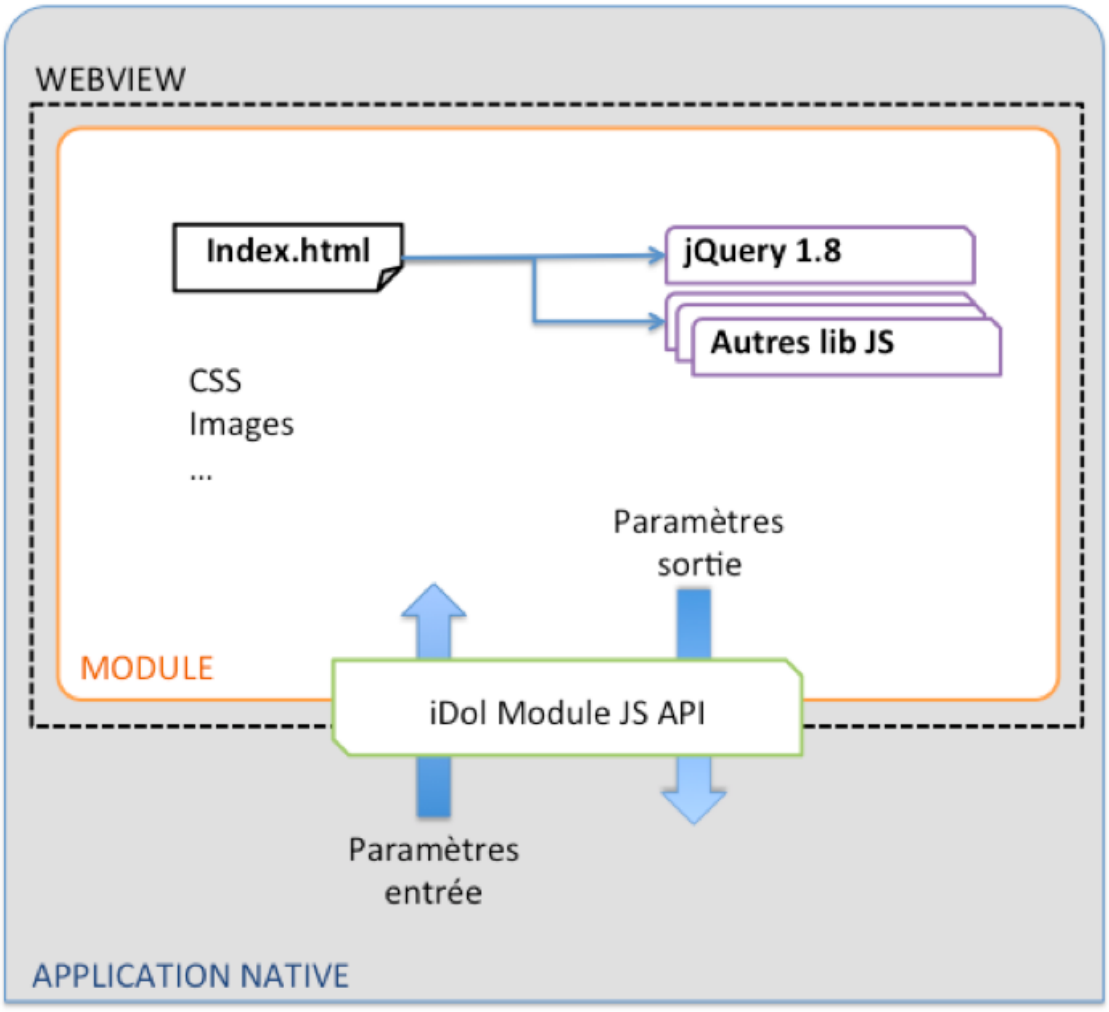
Principe

Les modules constituent une extension venant enrichir l'expérience utilisateur pour s'adapter à différents contextes d'usage.

Le module est un contenu d'une configuration pour iDol App au même titre qu'une vidéo ou qu'un questionnaire.

Les modules sont en mode portrait uniquement, à optimiser pour une résolution de 1536 x 2048

Architecture



Le module est réalisé en HTML 5/Javascript/CSS 3 (compatibilité Webkit iOS).

Il doit obligatoirement inclure depuis la page index.html la librairie jQuery 1.8.x (dernière version) qui est utilisée par l'application pour implémenter l'API de communication avec le module. La communication se fait au travers d'une API Javascript implémentée par l'application, appelée API iDol Module JS API. L'ensemble des ressources utilisées par le module (JS, CSS, images, …) doivent être intégrées dans l'archive et référencées par des URL relatives.

Premiers pas

Starter kit

Une structure de fichier prédéfinie permet d'initialiser un nouveau module : starter_kit.zip

Rôle des différents fichiers :

- index.html : couche présentation du module. Inclue de base les librairies Javascripts minimum
- css/style.css : feuille de style du module
- js/function.js : contient le Javascript du module. Doit au moins implémenter startModule et reStartModule (voir ci-après)
- lib : pour stocker les librairies js
- param : pour stocker les éléments paramètrables. Il est recommandé de placer les éléments de configuration dans data.js et les images dans le répertoire img

Fonction de démarrage

Le module doit obligatoirement implémenter les fonctions `startModule` qui est appelée par l'application native.

```
function startModule(param)
{
        if (param) context = param;

}
```

Cette fonction prend un unique paramètre qui est un objet context qui est persisté par l'application et par défaut remis à zero quotidiennement. La structure de l'objet context est libre pour chaque module.

Cette fonction doit obligatoirement implémenter le lancement du Javascript du module, par exemple la création de l'app avec Backbone ou Angular. Le lancement du javascript ne doit pas être réalisé ailleurs.

Fonction de redémarrage

Le module doit obligatoirement implémenter les fonctions `reStartModule` qui est appelée par l'application native.

```
function reStartModule(param)
{
    if (param) context = param;

}
```

Cette fonction prend un unique paramètre qui est un objet context qui est persisté par l'application. Elle est appelé après chaque inscription afin que le module puisse se réinitialiser pour l'utilisateur suivant.

Gestion multi-langue

Le module peut se décliner en plusieurs langue.

Le code de la langue est défini par l'App dans le paramètre iDol.lang (fr, de, en)

Ce paramètre est mis à jour avant chaque appel de startModule ou restartModule. C'est donc au sein de ces fonctions que le module peut adapter sa langue.

Fonction de changement de jour

Le module peut implémenter la fonction `changeDay` qui est appelée à un changement de jour pour savoir si le contexte persisté doit être effacé.

La fonction doit retourner false pour conserver le contexte. Si la fonction n'est pas implémentée, par défaut le contexte est effacé.

```
function changeDay()
{
  return true;
}?
```

Fonction de sortie

Pour terminer l'action du module et envoyer les paramètres à l'application, il faut appeler la fonction suivante :

```
iDol.output('module_output_event', outputParam)
```

outputParam est un objet contenant les paramètres à transmettre à l'application et/ou remonter au serveur (voir la rubrique paramètres de sortie)

starter_kit.zip (46,5 ko) Manuel Nouvel, 12-11-2014 21:36

Paramètres de sortie

Les paramètres de sorties sont définis dans un objet `output` envoyé avec `iDol.output`

`Ex :`

```
  {
      context:context,
      message : "Vous avez gagné",
      winner : true,
      gift : "5EUROS",
      restart: false,
      validate: false,
      answers:[
                { label:"titre", value:"texte libre" },
                {
                        label:"Satisfaction",
                        choices:{"value":"Valeur 1"}
                },
                {
                        label:"Satisfaction",
                        choices:[{label:"Valeur 1",sel:true},{label:"Valeur 2",sel:false}]
        }
       ]
  }
```

Les champs de cet objet sont positionnés pour agir sur l'application ou remonter des données. Ci après sont décrits les champs supportés :

context (obligatoire)

Ce champ obligatoire permet de mettre à jour le contexte persisté par l'application.

Il s'agit d'un objet javascript qui peut être librement renseigné. La conservation du contexte est utile essentiellement pour le décomptage quotidien des lots. C'est pourquoi l'objet contexte est remis à null par l'application tous les jours à minuit.

gift

Ce champ permet de remonter l'identifiant du cadeau gagné.

Il vaut soit :

- LOST : pas de cadeau gagné, le contact n'apparaitra pas dans la liste des gagnants
- un identifiant : le contact apparaitra dans la liste des gagnants avec ce cadeau et l'envoi

N.B. : Sur Pilot, l'envoi d'un SMS et/ou email peut être associé à un identifiant de gain

message

Ce champ s'il est renseigné entraîne l'apparition d'un message bloquant à la fin de l'inscription nécessitant l'intervention d'un vendeur. A utiliser typiquement lorsque qu'un lot doit être retiré immédiatement.

restart

Disponible uniquement depuis la version 4.7.5.

Ce champ s'il est renseigné et valorisé à true, permet de demander un redémarrage immédiat de l'inscription. Il est possible de gérer un ainsi une fin d'inscription dès le module de jeu pour certains scénarios.

validate

Disponible uniquement depuis la version 4.7.5.

Ce champ s'il est renseigné et valorisé à true, permet de valider l'inscription. Cela signifie que quoi qu'il arrive par la suite (abandon de l'utilisateur par exemple), la fiche d'inscription sera remontée au serveur et enregistrée. Dans le cas contraire, la fiche n'est remontée qu'une fois la page d'opin atteinte et l'optin renseigné.

answers

Permet de remonter des questions/réponses de manière équivalente à celles posées dans les formulaires de l'application. Format :

```
[
// Champ texte libre
{ label:"titre", value:"texte libre" },

// Choix simple
 {
  label:"Satisfaction",
  choices:{value:"Valeur 1"}
 },

?// Choix multiples
```

```
 {
  label:"Satisfaction",
  choices:[{label:"Valeur 1",sel:true},{label:"Valeur 2",sel:false}]
}]
```

Les questions sont une liste d'objets. Tous les objets comprennent un champ `label` qui correspond au titre de la question.

Il existe 3 types de questions :

- Réponse texte libre : champ value transmettant la valeur saisie
- Réponse à choix unique : champ choices:{value:"Valeur 1"}, value transmettant la valeur saisie
- Réponse à choix multiple : choices:[{label:"Valeur 1",sel:true},...], liste de tous les choix, sel indiquant si le choix est coché ou non

pictures

Permet de remonter une liste d'images ou photos (JPEG ou PNG)

```
[
  { name : "nom.jpg", "Photo encodée en base 64" }
  ...
]
```

dynamics

Permet d'indiquer des contenus dynamiques qui seront inclus dans l'email associé au contact.

```
[
  { "[CLE1]" : "....", "[CLE2]" : "...." }
  ...
]
```

L'email dans pilote devra inclure les champs dynamiques correspondant sous cette forme :

[CLE1]
[CLE2]

Développer, tester, déployer

La réalisation de module pour iDol App comprend 3 étapes :

- le **développement** en technologie HTML5/Javascript avec des outils standard
- la **validation** des modules développés dans l'environnement iDol et sur iPad
- le **déploiement** sur les iPads

Développement

Le développement est réalisé via un éditeur HTML de votre choix.

Le rendu du module peut-être testé dans un navigateur, de préférence Safari pour se rapprocher des conditions de l'iPad, en ouvrant le fichier index.html en local.

Une fonction doit être prévue pour appeler automatiquement `startModule` lorsque le User-Agent n'est pas mobile, afin de tester. Ex:

```
var isMobile = new RegExp('/iphone|ipad|ipod|android|blackberry/i').test(navigator.userAgent.toLowerCase());

if(!isMobile){

    $(window).load(function () {

        alert('Demarrage depuis le navigateur');

         startModule();

    });
}
```

Une fois le développement réalisé, l'arborescence des fichiers doit être compressée dans une archive zip pour être testée dans iDol.

Test et validation

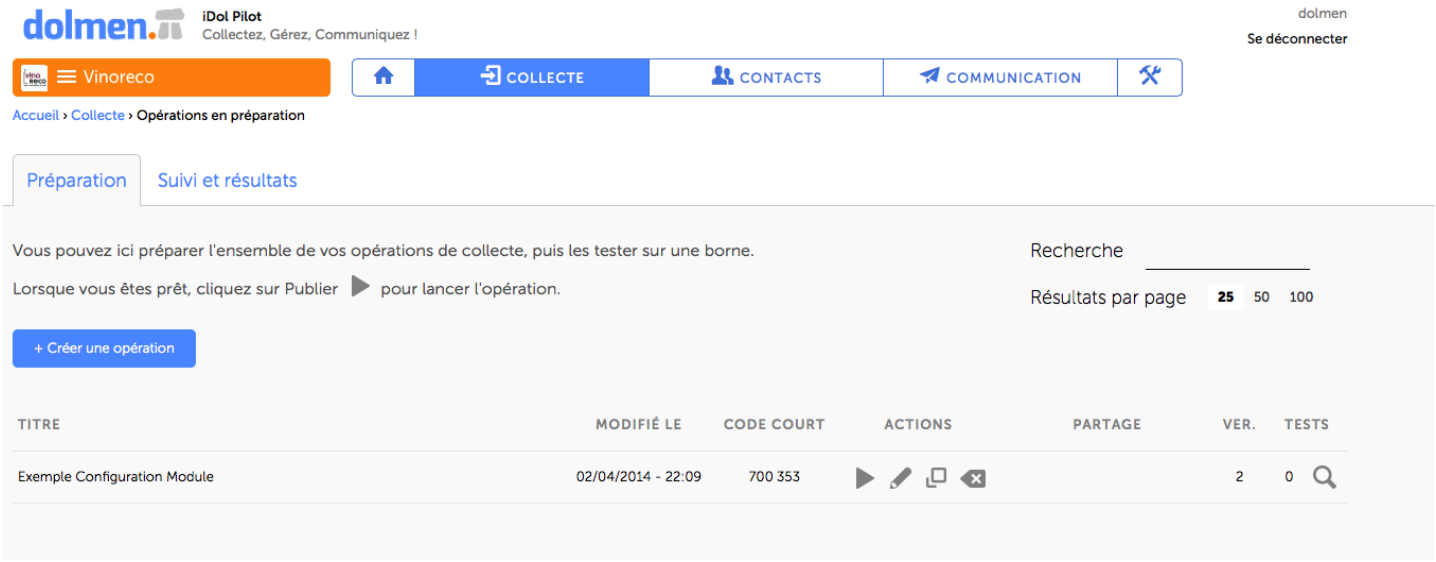
Méthode 1 : Upload direct sur une configuration

Afin de tester le module en condition réelle vous devez disposer :

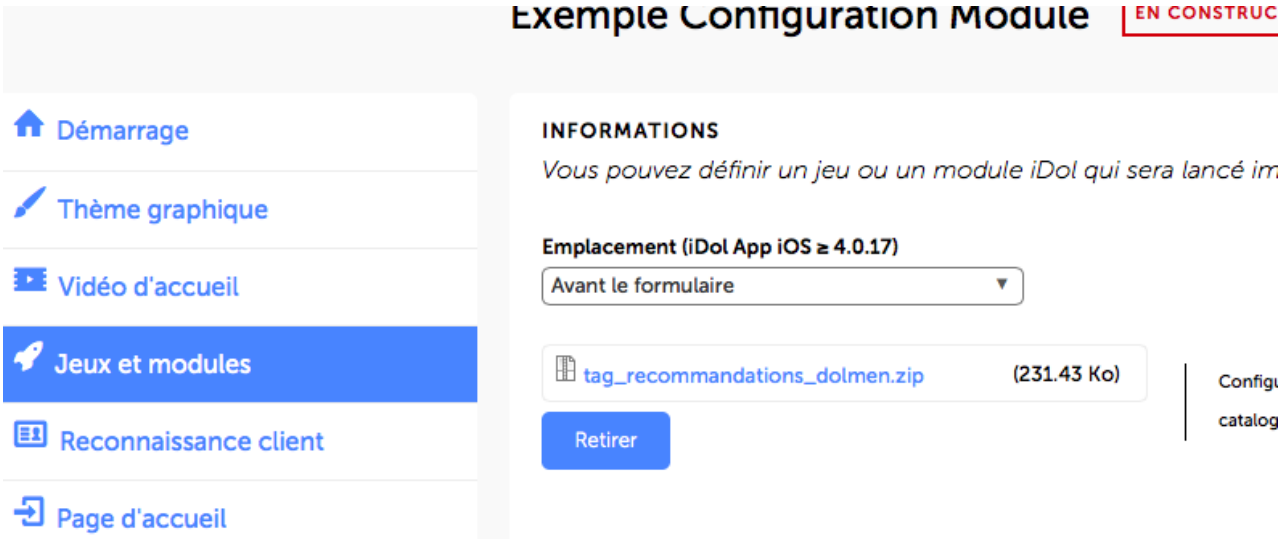
- de la dernière version d'iDol App pour iPad disponible sur http://lc.cx/idola
- d'un compte sur la plateforme Pilot sur https://pilot.dolmen-tech.com

Etape 1 : téléchargement du module dans une configuration

Connectez vous à Pilot et rendez-vous dans l'onglet "Collecte". Une configuration "Exemple Configuration Module" prête à l'emploi est disponible.



Cliquez sur l'icône crayon pour éditer la configuration et allez dans la séction Jeux & Modules.



Vous pouvez télécharger le module configuré de base dans la configuration et qui peut être un exemple intéressant pour vos propres développements.

Cliquez ensuite sur Retirer et téléchargez votre propre module.

Finissez l'édition en cliquant sur "Enregistrer et Quitter"

Etape 2 : récupération des identifiants

Notez le code court de la configuration (700 353 pour le module en exemple) dans l'onglet "Collecte"

Rendez-vous dans le dernier onglet (icône outils) puis dans le sous-onglet "Paramètre du Réseau". Notez le code réseau et le code secret qui vous sont attribués.

Etape 3 : chargement et lancement sur iPad

Sur un iPad, lancez iDol App.

Authentifiez vous avec le code réseau et le code secret que vous avez récupéré dans Pilot.

Cliquez sur "Nouvelle Configuration" et saisissez le code court que vous avez précédemment noté.

Lancez la configuration avec le bouton PLAY. Votre module est appelé dans le déroulé de l'expérience.

Déploiement

Le déploiement se fait au travers des configurations mises en oeuvre par les clients, de manière similaire à la validation.

Le seul élément à fournir est le zip du module à déployer. Ce zip est téléchargé sur les configurations utilisées dans les opérations sur les bornes en magasin.