PROGRAMMEZ.COM

PROGRAMMEZ!

Mensuel n°187 - Juillet/Août 2015

le magazine du développeur

Drupal 8 : les nouveautés, comment migrer ?

Spécial été 2015 : coding à la plage

Faites le plein d'idées pour Arduino et Raspberry Pi

Poppy : un robot à monter

Automatiser vos volets

Créer un écran de contrôle aérien

Scratch
La programmation pour les enfants

Devoxx 2015
Le plein de conférences !

Android
Choisir son outil de tests

Les API OneDrive

Je débute (en douceur) avec C++

M 04319 - 187 - F: 5,95 € - RD

Printed in EU - Imprimé en UE - BELGIQUE 6,45 € SUISSE 12 FS - LUXEMBOURG 6,45 € - DOM Surf 6,90 € Canada 8,95 $ CAN - TOM 940 XPF - MAROC 50 DH

PROLONGATION JUSQU'AU 17 JUILLET

COMMANDEZ WINDEV MOBILE 20

OU WEBDEV 20 OU WINDEV 20

ET RECEVEZ LE NOUVEAU GALAXY S6 EDGE

INCLUS: UN CHARGEUR À INDUCTION !

SAMSUNG

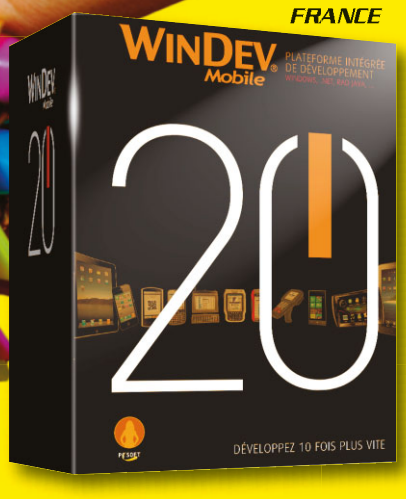
Galaxy S6 edge

Concept Mono-bloc • Ecran incurvé capacitif Super Amoled QHD 5.1" • Résolution 2560 x 1440 • Quadribande • 3G/4G • Android Lollipop 5.0 • APN 16 MP Autofocus f/1.9 • APN frontal 5.0 MP • Résolution vidéo en enregistrement UHD 4K (3840 x 2160) • Mémoire 32 Go • Processeur Octo Core 2.1 GHz • Bluetooth • Wifi • Batterie Li-ion 2600 mAh • USB 2.0 • Recharge rapide • GPS • NFC • ... *Livré avec un chargeur à induction*

Ou choisissez **1 Galaxy S6 64 Go** ou **2 Tablettes Galaxy Tab S 10,5"** ou **3 Tablettes Galaxy Tab 4 10,1"** ou encore **1 Télé Samsung 140cm Full HD Internet**

ou encore 1 ou 2 **PC Portables** ASUS.

WINDEV AGL N°1 en FRANCE



Aucun abonnement à souscrire pour bénéficier de cette offre.

PROFITEZ DE L'OPÉRATION « POUR 1 EURO DE + »

Pour bénéficier de cette offre exceptionnelle, il suffit de commander WINDEV Mobile 20 (ou WINDEV 20, ou WEBDEV 20) chez PC SOFT au tarif catalogue avant le 17 juillet 2015.
Offre réservée aux sociétés, administrations, mairies, GIE et professions libérales..., en France métropolitaine. Aucun abonnement n'est à souscrire pour bénéficier de cette offre.
Le développement pour Android et iOS s'effectue avec WINDEV Mobile ou WEBDEV. Le développement pour Windows et Linux s'effectue avec WINDEV ou WEBDEV. Voir tous les détails et des vidéos sur : **www.pcsoft.fr**
Le Logiciel et le matériel peuvent être acquis séparément; merci de vous connecter au site www.pcsoft.fr pour consulter la liste des prix et les dates de disponibilité. Tarifs modifiables sans préavis.

WINDEV Mobile 20 permet de créer facilement et rapidement des applications natives pour iOS, Android, Windows Phone et Windows Mobile. Liaison facile à votre SI et à toute base de données.
Les applications WINDEV sont directement importées.

Fournisseur Officiel de la Préparation Olympique

www.pcsoft.fr

Document non contractuel. Ceci est la 23ème opération «pour 1 euro de +» organisée par PC SOFT.

Codons à la plage, saison 2

Voilà, l'été est là.

Il est de tradition à Programmez ! que le numéro d'été soit plus fun, avec un ton plus léger. Mais un numéro d'été se prépare des mois à l'avance, avec des dizaines de réunions, de Webconférences. Enfin presque... Comme souvent, nous terminons les numéros à l'arrache quelques heures avant de partir à l'impression, négociant (plutôt suppliant un délai supplémentaire) : « pitié, pitié, soyez sympa accordez-nous 2-3h de plus, bon allez, disons 12h ». Et jusqu'au dernier moment, nous traquons la moindre page manquante. Parfois on se retrouve avec deux pages 67. Là on se dit : « bizarre, on a deux pages 67, ceci est une révolution ». On joue à Monument Valley (l'option shopping marche aussi) pour se détendre avant d'hurler : « pourquoi ? ». Et puis, tout se met en place. Je respire mieux et la pression retombe, avant de se dire, « ben, maintenant, on s'occupe du numéro suivant »... Nous sommes une boucle sans fin, sans else, ni end if.

Tu vas aimer Arduino et le Do It Yourself

Notre dossier d'été envoie du très très lourd cette année : 40 pages de gros délires, de codes ludiques, de montages, de projets maker ! Impossible de ne pas présenter quelques montages sympas à réaliser soi-même que ce soit en Arduino ou Raspberry Pi. Avec quelques idées, des capteurs, un peu de codes, on peut faire des trucs spectaculaires. De nombreux sujets y passent : ampoule avec capteur de mouvement, robotique avec Poppy, de l'impression 3D, automatiser ses volets ou encore créer un moniteur de contrôle aérien pour connaître la position des avions... Et bien d'autres choses comme l'étonnante Frog, une mini-carte de conception française !

Programmez ! Junior : nous n'oublions pas les enfants et les ados

Finis les cahiers de vacances classiques et les cris des enfants qui ne veulent pas les faire... Programmez ! se transforme en Programmez ! Junior. Nous vous proposons de découvrir un super environnement de programmation simple et très ludique : Scratch ! Pour les petits et les grands ! Vive le lolcat !

Programmez++ : Drupal 8, C++

Pour les adultes, nous vous proposons de vous plonger dans Drupal 8, la prochaine version majeure de Drupal qui arrivera sur nos serveurs dans quelques semaines. Cet été, nous démarrons une longue série de 5 articles sur la programmation C++ pour réapprendre ce langage et le faire découvrir à celles et ceux qui ne le connaissent pas. Have fun !

One more thing...

Et ce n'est pas fini. De nombreux autres sujets vous attendent pour l'été : la suite de notre retour de la conférence Devoxx 2015, ASP.Net 5, les techniques de tests pour les apps Android et les API OneDrive...

Bon code ! Bon été !

Rendez-vous le 29 août pour Programmez ! 188.

à lire dans le prochain numéro n°188 en kiosque le 29 Août 2015

LA RÉVOLUTION PHP 7
Le nouveau PHP arrive ! Découvrez toutes les nouveautés du langage, les bonnes pratiques de migrations, les outils. Quel futur pour PHP ?

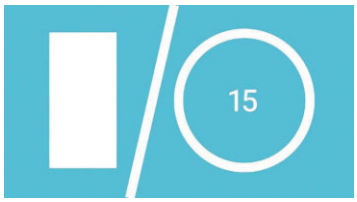
BABYLON.JS
Plus que jamais le Web affiche et utilise de la 3D. Exploitez un moteur 3D 100 % Web ! Une merveille !

ANDROID
Les dernières nouveautés du monde Android pour le développeur

Chaud devant avec Google I/O et la WWDC

Google et Apple ont donné à quelques jours d'intervalle leur conférence développeur. Le mot d'ordre était : améliorer, consolider l'existant, et évoluer en douceur, sans rupture profonde pour les utilisateurs et les développeurs. Chacun a dévoilé des projets et des outils qui vont beaucoup intéresser les développeurs. Certains diront : fade, d'autres : tant mieux, un peu de stabilité ! Nous reviendrons plus en détail sur ces sujets à partir de septembre.

Google I/O 2015



Android M

Il s'agit de la prochaine version du système mobile Android. De nombreuses améliorations et nouveautés seront au programme : gestion des permissions, productivité au quotidien, Android Pay (système de paiement), une gestion de la batterie améliorée, support de l'USB type-C. L'empreinte digitale devient une couche standard à Android (comme sur iOS).

Android Studio 1.3

L'IDE de développement Google revient avec des corrections de bugs et de belles améliorations : support du C et C++, début du support d'Android M. Actuellement en préversion.

Polymer 1.0

Présenté il y a un an, Polymer est une librairie pour créer des composants Web réutilisables. Il doit permettre une grande personnalisation des sites et applications Web. Cela permettra ainsi d'ajouter très rapidement des fonctions à vos pages Web. Un starter kit est disponible.

IoT/Domotique avec Brillo et Weave

Très attendu, Google dévoile Brillo, un système dédié aux objets connectés. Il s'agit d'un dérivé d'Android, mais ultra léger. Brillo pourra être déployé physiquement sur les objets. À cela se rajoute : Weave. Il s'agit d'un véritable protocole pour les objets connectés pour fournir un langage commun et standard à tout le monde ! Il assurera la communication les objets, les services Cloud et le smartphone. Il sera disponible sur Android et iOS (et peut-être plus).

WWDC 2015

Swift 2.0 et en open source

C'est l'annonce de la WWDC. Le nouveau langage d'Apple arrivera en v2 cet automne et en open source ! Les codes sources du langage et du compilateur seront ouverts. Apple livrera un portage sur Linux ! XCode 7 est disponible en bêta.

Un seul programme développeur !

Désormais, il n'y aura qu'un seul programme développeur couvrant OS X, iOS et watchOS avec toutes les ressources (bêtas, outils, documentations...). Tarif : 99 $

watchOS 2.0

L'Apple Watch aura droit à une sérieuse mise à jour système : WatchConnectivity (connexion WiFi), ClockKit (nouveau framework supportant les nouveautés ergonomiques et d'interfaces), accès aux couches matérielles par les SDK et les API, applications natives, nouvelles fonctions Apple Pay...

iOS 9

Le prochain iOS intégrera un vrai système multitâche avec partage d'écran pour plusieurs applications. Apparition de fonctions spécifiques à l'iPad. Apple introduira plusieurs nouveaux frameworks : GameplayKit (pour jeux ayant un gameplay complexe), replayKit (les jeux pourront enregistrer une vidéo des actions puis les rejouer), Model I/O (pour les modèles 3D).

HomeKit

1 an après sa présentation, Apple lance enfin HomeKit, les premiers objets compatibles sortent. HomeKit sera intégré à iOS 9 et OS X 10.11 et sans doute aussi à l'Apple TV.

Pour rejouer les sessions Google I/O : https://events.google.com/io2015/

Pour la WWDC : https://developer.apple.com/wwdc/

agenda

INNOROBO : DU 1ER AU 3 JUILLET/LYON

C'est l'événement robotique de l'année ! Lyon accueille la 4e édition d'Innorobo avec cette année encore de nombreux pays européens et surtout toutes les dernières avancées robotiques et les robots qui vont commencer à « envahir » notre quotidien. Site : http://innorobo.com

PARIS OPEN SOURCE SUMMIT : 18 & 19 NOVEMBRE (PARIS)

Solutions Linux, le salon Linux et Open Source, et Open World forum fusionnent pour donner un unique grand événement Open Source français : Paris Open Source Summit. Il s'agit de concentrer les efforts et de faire de Paris une place incontournable du monde Open Source et Linux en Europe.

FORUM 2015 PHP : 23 & 24 NOVEMBRE (PRÈS DE PARIS)

Le monde PHP fête les 20 ans du langage et la sortie de PHP 7. Le Forum PHP est la référence française des développeurs et entreprises utilisant ce langage open source ! site : http://www.afup.org/pages/forumphp2015/

Quelle interopérabilité entre mes différents fournisseurs Cloud ?

Avec Aruba Cloud,

vous avez l'assurance de ne pas être prisonnier d'un fournisseur. Nos services sont intégrés au **driver DeltaCloud** et compatibles **S3**. De plus, vous pouvez utiliser des formats standards d'images de machines virtuelles, **avec VHD et VMDK**, ainsi que des modèles personnalisés provenant éventuellement d'autres sources.

3 hyperviseurs

6 datacenters en Europe

APIs et connecteurs

70+ templates

Contrôle des coûts

Nous avons choisi Aruba Cloud car nous bénéficions d'un haut niveau de performance, à des coûts contrôlés et surtout car ils sont à dimension humaine, comme nous. Xavier Dufour - Directeur R&D - ITMP

Contactez-nous! 0810 710 300 www.arubacloud.fr

aruba CLOUD

MY COUNTRY. MY CLOUD.*

Cloud Public | Cloud Privé | Cloud Hybride | Cloud Storage | Infogérance

*MON PAYS, MON CLOUD.

Retour sur Devoxx Paris 2015

2e partie

Nous poursuivons notre résumé de Devoxx 2015. L'événement a été très riche, avec de très nombreuses sessions intéressantes. Ce mois-ci, nous parlons de Docker, de livraison continue, de Java 8.
Suite et fin de notre retour Devoxx à la rentrée, dans le n° 188 !

La rédaction.

DOCKER, effet boule de neige

L'intervention de Patrick Chanezon au Devoxx 2015 prédisait bien la suite à venir : « Microsoft vient de publier sur GitHub le premier fork Docker pour les containers Windows ».

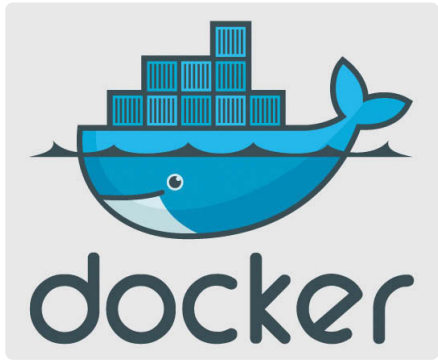
Robert DEGRET
Développeur & Architecte Senior chez **PALO IT**

L'efficacité du workflow Docker n'avait donc pas échappé aux équipes de Satya Nadella. Il faut dire que les briques se mettent tout doucement en place :

- Boot2docker développé avec les équipes Microsoft,
- Arrivée de Windows Nano Server qui sera capable à la fois de faire fonctionner des containers Windows ainsi que les machines virtuelles Hyper-V,
- Annonce d'Azure Stack, outillage nécessaire à la gestion de cloud hybride,
- Création du client Docker Windows.

Les containers que Google utilise depuis une dizaine d'années pour ses applications critiques font tâche d'huile. Amazon, Azure et Google vont supporter la technologie à très court terme. Il reste cependant quelques problèmes de jeunesse à résoudre avant que Docker ne soit accepté dans tous les environnements de production. Par conception, les containers Docker ignorent le réseau et reçoivent une nouvelle adresse à chaque interruption temporaire, ce qui pose problème au cas où d'autres systèmes disposaient déjà de cette adresse. Différentes solutions sont en cours de développement (ex. Weave, Flannel, Pipework) pour combler ce manque. La communauté n'a pas encore décidé qui l'emportera car les discussions font encore rage autour des points forts mais aussi des problèmes de conception.

Pour beaucoup d'autres utilisateurs, le plus gros frein à l'adoption reste la sécurité des containers Docker. Il suffit de consulter la liste croissante des CVE publiées pour comprendre pourquoi il reste préférable de faire fonctionner des containers au sein de machines virtuelles, même si cela annule les gains en performance de la containerisation. Là encore, la bataille est rude pour trouver la solution :



- Twistlock propose une suite d'audit et d'outils de contrôle pour Docker,
- Joyent utilise une approche système avec un OS capable de faire fonctionner n'importe quel container Docker Linux avec une sécurité éprouvée grâce à l'héritage de Solaris.

Docker Inc continue à recevoir les dollars et à accaparer l'attention des développeurs. La société est consciente des challenges qui se dressent devant elle et essaie de combler les failles par des acquisitions ciblées (récemment KiteMatic et SocketPlane).

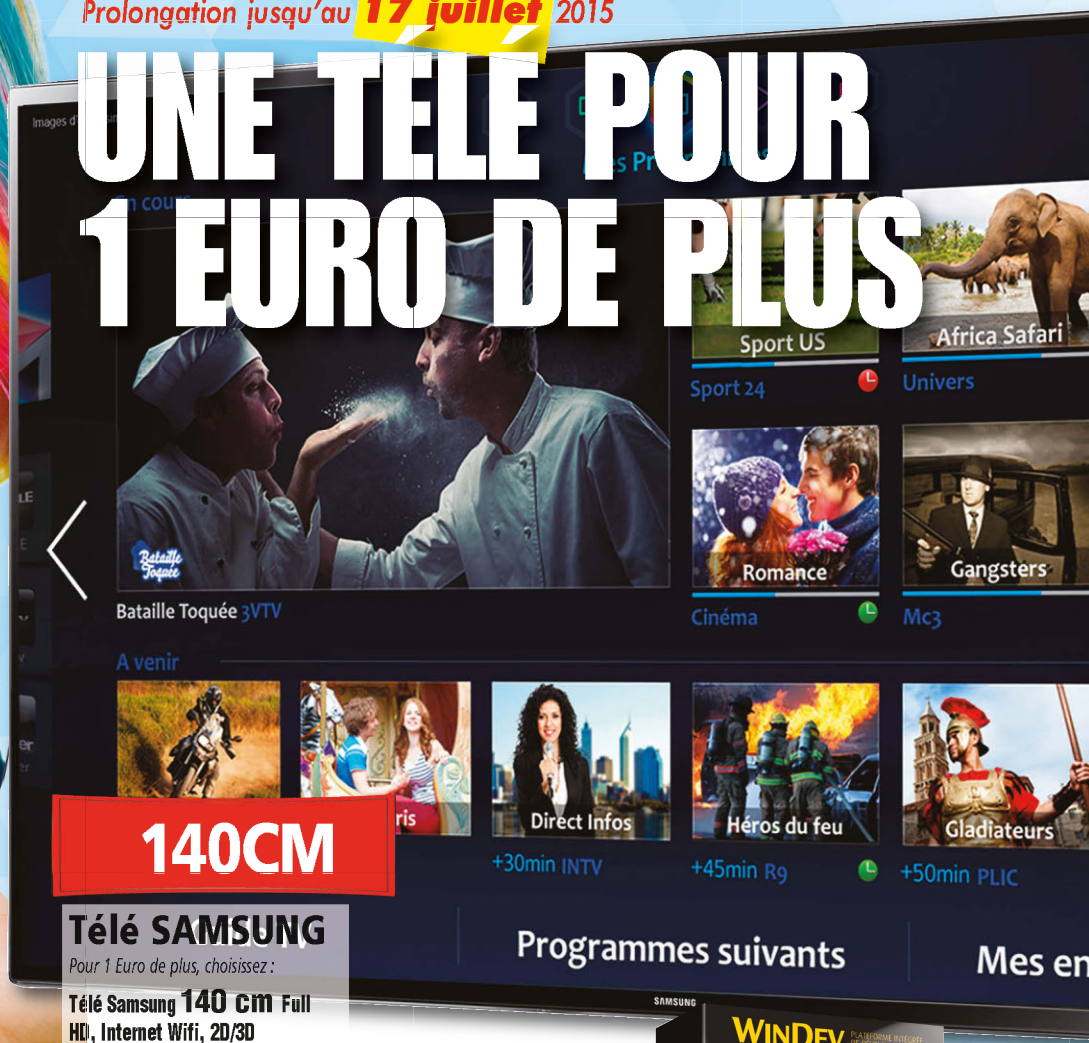
Bref, on n'a pas fini d'entendre parler de Docker!

Sources

Patrick Chanezon: http://cfp.devoxx.fr/2015/speaker/patrick_chanezon
Windows Nano: http://blogs.technet.com/b/windowsserver/archive/2015/04/08/microsoft-announces-nano-server-for-modern-apps-and-cloud.aspx
Windows Docker:
https://github.com/Microsoft/docker/commit/bad29cf9adb8fcc78347eb0a8c154c04a7b36e2e-Network
Docker réseau: http://weave.works/ & https://github.com/coreos/flannel & https://github.com/jpetazzo/pipework
Problème Weave: http://www.generictestdomain.net/docker/weave/networking/stupidity/2015/04/05/weave-is-kinda-slow/
Sécurité Docker: http://www.openwall.com/lists/oss-security/2015/05/07/10

Prolongation jusqu'au **17 juillet** 2015

UNE TÉLÉ POUR 1 EURO DE PLUS



140CM

Télé SAMSUNG

Pour 1 Euro de plus, choisissez :

Télé Samsung 140 cm Full HD, Internet Wifi, 2D/3D

Réf. UE55H6400

Elu «Langage le plus productif du marché»

COMMANDEZ WINDEV MOBILE 20 (OU WEBDEV OU WINDEV) CHEZ PC SOFT ET RECEVEZ UN SUPERBE MATÉRIEL AU CHOIX POUR «1 EURO DE PLUS»

WINDEV

Ou vous pouvez également choisir :

Nouveau smartphone Samsung Galaxy S6 Edge

Configurations détaillées sur www.pcsoft.fr

INCLUS: UN CHARGEUR À INDUCTION

(x2) Nouvelle tablette Samsung Galaxy Tab S 10,5p

Configurations sur www.pcsoft.fr

x2

Nouveau smartphone Samsung Galaxy S6

Configurations détaillées sur www.pcsoft.fr

INCLUS: UN CHARGEUR À INDUCTION

(x3) Nouvelle tablette Samsung Galaxy Tab 4 10,1p

Configurations sur www.pcsoft.fr

x3

Ou encore 1ou 2 PC portables ASUS...

Pour bénéficier de cette offre exceptionnelle, il suffit de commander **WINDEV** Mobile 20 (ou WINDEV 20 ou WEBDEV 20) chez PC SOFT au tarif catalogue avant le 17 Juillet 2015. Pour 1 Euro de plus, vous recevrez alors le ou les magnifiques matériels que vous aurez choisis. Offre réservée aux sociétés, administrations, mairies, GIE et professions libérales, en France métropolitaine. **L'offre s'applique sur le tarif catalogue uniquement.** Voir tous les détails et des vidéos sur : **www.pcsoft.fr** ou **appelez-nous.**
Le Logiciel et le matériel peuvent être acquis séparément. Tarif du logiciel au prix catalogue de 1.650 Euros HT (1.973,40 TTC). Merci de vous **connecter au** site www.**pcsoft.fr pour consulter la liste** des prix des matériels et les dates de disponibilité. Tarifs modifiables **sans** préavis.

Descriptif technique complet des matériels sur www.pcsoft.fr

Tél province: **04.67.032.032**
Tél Paris: **01.48.01.48.88**

Fournisseur Officiel de la Préparation Olympique

www.pcsoft.fr

Optaplanner ou comment optimiser les itinéraires, les plannings et bien plus encore

Bruno Doolaeghe
Soat

Vous souvenez-vous, durant vos folles années d'étudiant, d'avoir été confrontés, lors d'un cours d'algorithme, au problème du voyageur de commerce ? Rappelez-vous, il s'agit pour un voyageur de commerce, de trouver le plus court chemin passant par un ensemble de villes à visiter. C'est un problème NP-complet, pour lequel on n'a, jusqu'à aujourd'hui, pas de solution exacte tournant dans un temps "acceptable"... Il existe néanmoins, des solutions "approchantes" raisonnables : OptaPlanner en est une ! Voici un petit compte rendu de la présentation qui en a été faite par Geoffrey De Smet, leader technique et fondateur d'OptaPlanner et Frederic Hornain, *Solution Architect* chez Redhat, lors de la session 2015 de Devoxx France.

OPTAPLANNER, C'EST QUOI ?

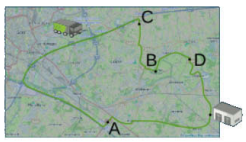
OptaPlanner est un moteur Open Source de recherche de solution optimale aux problèmes de satisfaction de contraintes ; en d'autres termes : il fait comme *Winston Wolfe*, "*il résout les problèmes" !* OptaPlanner est développé chez Redhat JBoss. Ecrit en java, intégrable et léger, il trouve des solutions de manière "efficace" aux problèmes de planification (du type problème du voyageur de commerce, ou problème du sac à dos, dits problèmes NP-complet), en combinant heuristiques d'optimisation et méta heuristiques avec calcul de score : en clair, il trouve une solution "quasi-optimale" en un temps "raisonnable" à des problèmes complexes !

QUELQUES EXEMPLES D'APPLICATION

Durant leur présentation à Devoxx, Geoffrey De Smet et Frederic Hornain nous ont exposé plusieurs exemples d'utilisation du moteur d'OptaPlanner en entreprise, à travers plusieurs démos...

LE PARCOURS AÉRIEN D'UN PDG

Le problème est identique à celui du problème du voyageur de commerce : un PDG souhaite se rendre en avion dans un certain nombre de villes et cherche à optimiser son temps de trajet. Durant cette démo utilisant la GUI et le moteur d'OptaPlanner, nous avons vu comment l'outil pouvait calculer en temps réel le trajet optimal (il est même possible d'ajouter / supprimer une ville pour qu'OptaPlanner recalcule à la volée le nouveau meilleur trajet à effectuer !) Fig.1.



LE CHAUFFEUR LIVREUR

Un autre problème du même type est celui du planning du chauffeur livreur, qui doit effectuer sa tournée avec son camion. OptaPlanner s'intègre alors avec Google Maps ou GraphHopper, pour calculer le meilleur trajet sur la carte routière :

LE CLOUD PROVIDER

Enfin, un autre exemple d'utilisation présenté durant le talk, tiré d'un cas réel est celui du *Cloud provider*, qui cherche à optimiser l'utilisation de son parc de machines : il doit en effet faire tourner l'ensemble des processus de ses clients sur un minimum de machines, en les regroupant

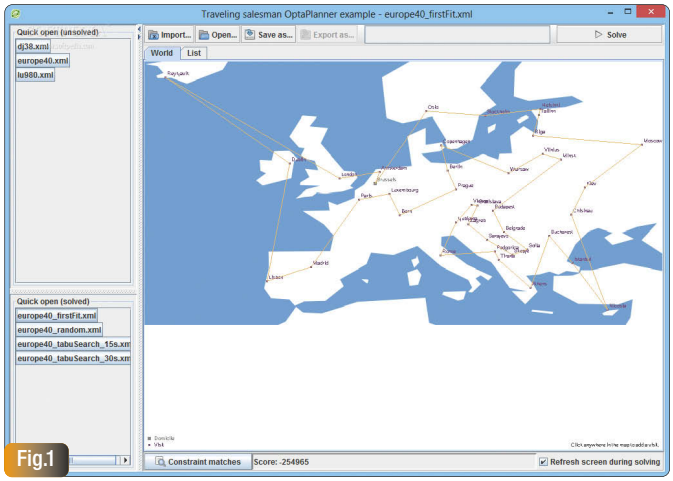


Fig.1

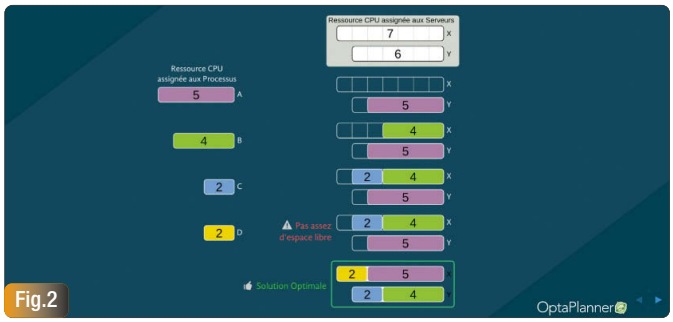


Fig.2

de manière optimale tout en respectant certaines contraintes (utilisation CPU, utilisation mémoire, bande passante réseau...). Dans ce cas concret, les solutions calculées par OptaPlanner ont apporté une économie de 17% au *Cloud provider* sur ses frais par rapport à sa solution "maison" ! Fig.2.

DANS CHAQUE BOÎTE D'OPTAPLANNER, ON TROUVE...

OptaPlanner est un projet Open Source (licence Apache) écrit en java (code source sur github), disponible sur le central maven repository :

```
1 <dependency>
2     <groupId>org.optaplanner</groupId>
3     <artifactId>optaplanner-core</artifactId>
4     <version>6.2.0.Final</version>
5 </dependency>
```

Il existe également sous forme d'un zip téléchargeable depuis le site d'OptaPlanner, qui contient notamment des exemples d'utilisation : Fig.3. La documentation complète y est incluse (elle est également disponible en ligne, avec un quick start guide), ainsi que la Javadoc et le code source.

MISE EN ROUTE

Utiliser OptaPlanner pour résoudre un problème se fait de manière programmatique et par configuration ; prenons l'exemple du *Cloud provider.* Il s'agit de suivre les étapes suivantes :

DÉFINIR LE PROBLÈME

Le problème est celui défini précédemment : il s'agit d'assigner des processus à des machines, en tenant compte de leurs capacités (CPU, mémoire, bande passante), et en optimisant l'utilisation du parc de

Abonnez-vous à PROGRAMMEZ!

le magazine du développeur

Nos classiques

PDF 30 €(*)
1 an - 11 numéros

Etudiant 39 €
1 an - 11 numéros

1 an 49 €
11 numéros

2 ans 79 €
22 numéros

(*) Uniquement sur le site internet

Tarifs France métropolitaine

Nos offres d'été

1 an → **64,90 €** *
2 ans → **94,90 €** *

Inclus un kit complet CPL dLAN 550 de DEVOLO et une clé USB contenant tous les numéros de Programmez depuis le n°100

(*) Valeur du CPL 79,90 €
Valeur clé USB 29,90 €
Frais logistiques : 15,90 €

Toutes nos offres sur www.programmez.com

Oui, je m'abonne

ABONNEMENT à retourner avec votre règlement à :
Service Abonnements PROGRAMMEZ, 4 Rue de Mouchy, 60438 Noailles Cedex.

☐ ***Abonnement 1 an au magazine :*** 49 €*
☐ ***Abonnement 2 ans au magazine :*** 79 €*
☐ ***Abonnement étudiant 1 an au magazine :*** 39 €*
Photocopie de la carte d'étudiant à joindre

Offre d'été : Programmez + kit CPL dLAN 550 + Clé USB

☐ ***Abonnement 1 an :*** 64,90 € (au lieu de 158,80 €)
☐ ***Abonnement 2 ans :*** 94,90 € (au lieu de 188,80 €)

Tarifs France métropolitaine

☐ M. ☐ Mme ☐ Mlle Entreprise : Fonction :

Prénom : Nom :

Adresse :

Code postal : Ville :

email indispensable pour l'envoi d'informations relatives à votre abonnement

E-mail : @

☐ Je joins mon règlement par chèque à l'ordre de Programmez !
☐ Je souhaite régler à réception de facture

** Tarifs France métropolitaine*

PROG 187
***Offre limitée**, valable jusqu'au 29 août 2015*

machines pour en minimiser le coût... Plus formellement, on pourra dire que ce problème doit :

- Respecter les *contraintes fortes* suivantes :
  - La puissance CPU d'une machine doit être supérieure ou égale à la somme des puissances CPU requises par chaque processus hébergé,
  - La taille mémoire d'une machine doit être supérieure ou égale à la somme des tailles mémoire requises par chaque processus hébergé,
  - La bande passante consommée par l'ensemble des processus est inférieure ou égale à la bande passante maximale de la machine,
- Optimiser la *contrainte faible* suivante :
  - le coût d'utilisation du parc de machines doit être minimal.

MODÉLISER LE MÉTIER

Nous allons définir nos objets métier sous forme de POJO, qui permettront de modéliser le problème :

- *Computer* représentera une machine, avec ses capacités (CPU, mémoire, bande passante) et un coût d'utilisation,
- *Process* un processus devant tourner sur une machine,
- *CloudBalance* qui modélise le problème proprement dit : il référence un dataset de machines et de processus clients que l'on souhaite associer.

Nous utiliserons ensuite des annotations OptaPlanner pour interagir avec le moteur : Fig.4.

INJECTER LE DATASET

OptaPlanner peut s'intégrer avec toutes sortes de datasource, pour récupérer le dataset sur lequel on souhaite résoudre un problème :

- Flux XML (via XStream, JAXB...),
- Base de données (JDBC, JPA...),
- REST/SOAP,
- Infinispan, Hadoop...

Dans cet exemple, nous utiliserons un générateur aléatoire de *Computer / Process :*

```
1  // Load a problem with 400 computers and 1200 processes
2  CloudBalance unsolvedCloudBalance = new CloudBalancingGenerator().createCloudBalance(400, 1200);
```

IMPLÉMENTER LES CONTRAINTES

Cette étape (la plus difficile !) consiste à traduire les contraintes définies ci-dessus en contraintes compréhensibles par OptaPlanner, lui permettant de calculer un score pour chaque solution testée, et ainsi trouver la solution "optimale". Elles peuvent être écrites sous 3 formes possibles :

- Une simple classe java,
- Une classe java "incrémentale" (améliore les performances),
- Un fichier DRL (script Drools).

CONFIGURER LE MOTEUR DE RÉSOLUTION

Le moteur de résolution est configuré via un fichier XML, dans lequel on déclare notamment :

- Les POJO utilisés pour modéliser le métier,
- Comment calculer le score (quelle classe java ou fichier DRL utiliser),
- Éventuellement, comment optimiser l'algorithme de résolution utilisé (fonctionnalités avancées).

ASSEMBLER ET EXÉCUTER !

Enfin, l'assemblage et l'exécution du planner se fait simplement par API :

```
1  public class CloudBalancingHelloWorld {
2
3      public static void main(String[] args) {
4          // Build the Solver
```

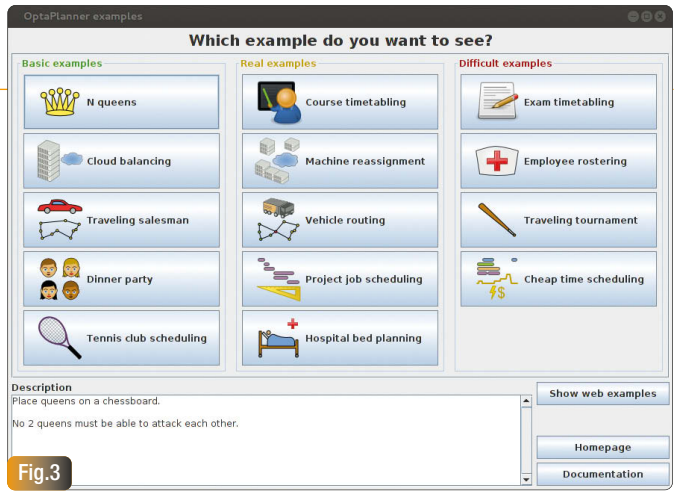


Fig.3

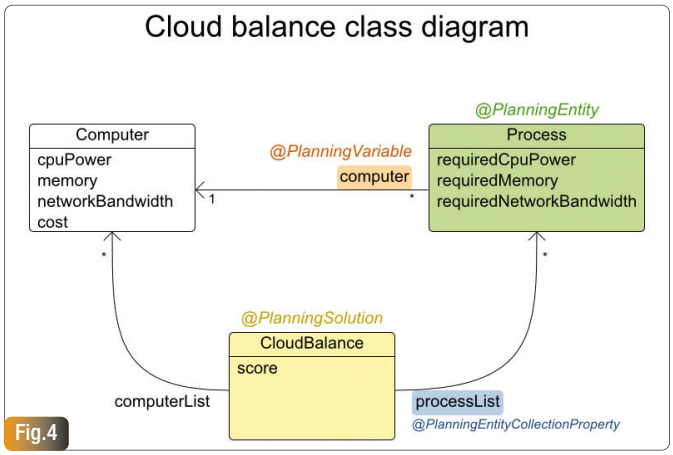


Fig.4

```
5          SolverFactory solverFactory = SolverFactory.createFromXmlResource("org/optaplanner/
examples/cloudbalancing/solver/cloudBalancingSolverConfig.xml");
6          Solver solver = solverFactory.buildSolver();
7
8          // Load a problem with 400 computers and 1200 processes
9          CloudBalance unsolvedCloudBalance = new CloudBalancingGenerator().createCloudBalance(400, 1200);
10
11         // Solve the problem
12         solver.solve(unsolvedCloudBalance);
13         CloudBalance solvedCloudBalance = (CloudBalance) solver.getBestSolution();
14         // Display the result
15         System.out.println("\nSolved cloudBalance with 400 computers " +
16             "and 1200 processes:\n" +
17             toDisplayString(solvedCloudBalance));
18     }
19     ...
20
21     }
22
23
```

NB : Pour plus de détails sur cet exemple, consulter le Quick Start guide

CONCLUSION

OptaPlanner est un outil fort prometteur de recherche de solution optimale à des problèmes sous contraintes, relativement simple à prendre en main, et intégrable dans une application Java. Même si durant leur démonstration, Geoffrey De Smet et Frederic Hornain nous ont montré un outil performant et réactif, on peut s'interroger sur sa scalabilité, face à un problème comportant un grand nombre de contraintes, ou un dataset big data. Même si aujourd'hui la librairie ne peut pas fonctionner sur un cluster, ce qui ouvrirait la porte à une scalabilité horizontale, Frederic et Geoffrey nous l'ont promis : ils y travaillent déjà !

« Livrer chaque jour ce qui est pret »

Du 8 au 10 avril 2015 se déroulait le Devoxx, LA conférence pour Développeurs à Paris. En tant que nouvel arrivant dans la capitale, je me devais d'y faire un tour ! Une conférence a particulièrement retenu mon attention : « Livrer chaque jour ce qui est prêt », présentée par Dimitri BAELI (@dbaeli) et Benjamin DEGERBAIX (@benbaix) de chez LesFurets.com. Je commencerai par vous résumer cette présentation. Nous approfondirons ensuite le sujet en nous penchant sur une pratique de plus en plus adoptée : le Continuous Delivery.

Kevin ALDEBERT
Développeur Java chez
PALO IT

Slides de la présentation :
http://fr.slideshare.net/beastiefurets/devoxx-france-2015-livrer-chaque-jour-ce-qui-est-pret

CONTEXTE

La problématique est posée dès le début : comment passer d'une organisation de type « éditeur logiciel » (cycle Plan/Dev/Recette/Prod) avec une release par mois, à une organisation non moins solide de type « Continuous Delivery » avec une mise en production par jour de « ce qui est prêt » ?

Quels avantages ?

Une réactivité accrue vis à vis du business,

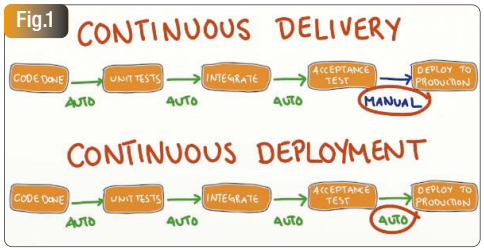
- L'accélération du Go-To-Market,
- Une productivité et efficacité améliorées,
- Un produit de meilleure qualité,
- Une satisfaction client accrue.

Quelles difficultés ?

- Le coût initial ;
- L'investissement de chaque partie (de la conception à l'exploitation).

Déployer en production plusieurs fois par jour sans que l'utilisateur n'y voie quelque chose, ça ne vous rappelle personne ? Facebook, Google, Etsy, Netflix et j'en passe ont déjà franchi le cap depuis plusieurs années ! Avant toute chose, veillez à ne pas confondre Continuous Delivery et Continuous Deployment. Dans le cadre du premier, la décision de mettre en production n'est pas systématique et reste une étape manuelle. A chacun de déterminer quelle solution lui convient le mieux selon ses besoins Fig.1. Vous l'avez compris, c'est un cheminement. Il est donc impossible de faire du Continuous Deployment sans Continuous Delivery, comme du Continuous Delivery sans Continuous Integration. Pour des équipes qui démarrent de zéro, l'investissement risque de s'avérer conséquent. Chaque partie prenante peut émettre des objections quant à la valeur ajoutée du changement. Il faut donc y aller étape par étape ! Fig.2.

Fig.1



MAIS ALORS, PAR OU COMMENCER ?

L'équipe des Furets nous explique l'approche qu'ils ont choisie afin d'implémenter le Continuous Delivery : améliorer par la fin ! Ils se sont tout d'abord concentrés sur l'exploitation avant de remonter progressivement jusqu'à la conception.

Exploitation

Monitoring / Measure

Pour livrer tous les jours - plus précisément à J+2, c'est à dire que les livraisons contiennent ce qui a été développé l'avant-veille - il ne faut pas négliger la qualité. On pense alors tests unitaires et tests d'intégration. Mais avant cela, il y a toute une partie outillage. En effet, Les Furets ont mis en place des outils de monitoring qui leur permettent de s'appuyer sur de multiples métriques et pour le coup, ce sont des métriques fonctionnelles (taux de conversion par exemple) à l'aide de sondes Zabbix.

Conseil n°1 : ajouter des métriques métiers sur les applications et habituer l'équipe à s'en servir.

Logs

Réagir vite est un des facteurs clés. Encore faut-il être au courant et à l'écoute de la moindre perturbation au sein du système. Les logs sont une source d'information importante et doivent être suivis, notamment après la mise en production. Pour cela, il faut maintenir un niveau de remontée d'exception visible afin de ne pas noyer l'information. D'autre part, dans le cas des Furets, la page d'accueil est une application single-page, ce qui induit beaucoup de code Javascript. Les erreurs levées par ce code doivent être également lisibles dans les logs. Dernier point : si les logs sont bien visibles, ils doivent bien sûr être lus. Ils ont fait le choix d'envoyer la trace de chaque erreur par email. Et pas que les logs de production Fig.3. A chacun d'adapter par la suite la fréquence en envoyant un email toutes les heures ou tous les jours.

Conseil n°2 : ne pas laisser les erreurs Javascript pourrir le navigateur des utilisateurs.

Mise en production

Infrastructure As Code

Le code est versionné, testé, intégré, validé et soumis à des tests de charge, mais qu'en est-il de l'infrastructure ? Avec l'utilisation progressive des services de Cloud, on a grand besoin de pouvoir faire de même avec les environnements. C'est là qu'intervient un principe issu de la culture DevOps : l'Infrastructure As Code. De la configuration de l'applicatif à celle des serveurs de chaque environnement, tout est inclus dans un livrable dédié. C'est uniquement lors de l'installation de l'application à l'aide de scripts de déploiement par exemple, que la bonne configuration sera chargée et utilisée en détectant sur quel environnement on se trouve. C'est tout aussi faisable pour les modifications sur le schéma de données et ainsi cela permet d'automatiser la migration des bases de données. Si vous avez besoin de changer des paramètres de configuration directement sur un environnement sans passer par le process de Continuous Delivery, vous pouvez utiliser une solution plus dynamique en stockant cette partie en base de données. On peut encore aller plus loin en ajoutant des tests unitaires sur l'infrastructure et l'analyse de qualité.

Zero Downtime Deployment

Livrer tous les jours c'est bien, mais sans éteindre et redémarrer la production, c'est mieux. C'est ce qu'on appelle le Zero Downtime Deployment. Plusieurs solutions existent pour cela :

- **Blue/Green Deployment :** c'est la solution mise en œuvre chez les Furets. L'équipe a créé un

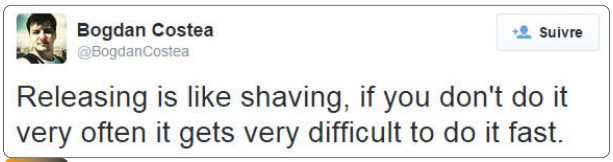


Fig.2

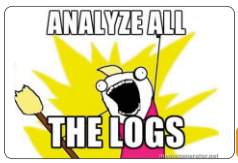


Fig.3

deuxième environnement de production, afin d'y déployer la nouvelle version de l'application. Par exemple, supposons que le blue contienne la version courante et le green la nouvelle : il suffit de rediriger les utilisateurs du blue vers le green et cela grâce au routage. Ce pattern est aussi utilisé pour faire de l'A/B Testing en ne redirigeant qu'une partie des utilisateurs sur la nouvelle version et il suffit de surveiller l'adoption de la nouvelle fonctionnalité à l'aide de métriques. Concernant les sessions HTTP, il est possible d'utiliser des sessions partagées en utilisant un cache de session commun entre les serveurs.

- **Canary Release** : lorsque la nouvelle version de l'application est prête, il suffit de la déployer sur votre infrastructure et de la rendre accessible seulement à un sous-ensemble de vos utilisateurs. Similaire au Blue/Green Deployment, vous pouvez à tout moment revenir sur la version précédente en cas de problèmes rencontrés avec la nouvelle Fig.4.
- **Dark Launch** : les tests de charge sont très utiles lorsqu'on souhaite simuler l'impact d'un nombre important d'utilisateurs. Mais quand vous êtes une société telle que Facebook, simuler plusieurs millions de personnes tentant d'accéder à une nouvelle fonctionnalité n'est pas chose simple. Dans le cadre du lancement du chat de Facebook, les développeurs ont choisi de lancer une période de "Dark Launch", pendant laquelle les pages Facebook feraient des connexions aux serveurs de chat. Des requêtes simulaient alors l'envoi de messages mais sans que la partie UI ne soit visible. Le but était de surveiller si le site pouvait supporter la charge engendrée par la nouvelle fonctionnalité. Une fois les bugs corrigés, il ne restait plus qu'à activer l'interface utilisateur.

Validation

La validation de la release s'effectue à l'aide d'outils manuels et automatiques :

- Tests unitaires et d'intégration,
- Code review,
- Validation fonctionnelle,
- Grid Selenium,
- Zeno Pixel.

Grid Selenium

Les tests Selenium sont très souvent intégrés au build de l'application, notamment à l'aide de Maven. Dans le cas des Furets, les tests sont exécutés après build sur un serveur. Leurs 200 tests mettant six heures en exécution classique. Ils ont choisi de mettre en place une grille afin de les paralléliser et ainsi accélérer leur exécution. Résultat : une heure ! C'est mieux mais pas encore assez lorsque l'on souhaite livrer tous les jours. En utilisant un système de virtualisation tel que LXC (LinuX Container) en s'appuyant sur la mémoire au lieu du disque, dix minutes suffisent pour exécuter les 200 tests Selenium.

Zeno Pixel

Zeno Pixel est un outil Open Source développé en interne chez LesFurets qui permet de faire de la non-régression sur la partie frontend du site. Pour cela, l'outil prend des screenshots des pages de l'application et les compare à la version précédente, ou à un autre environnement. Il est développé en NodeJS et PhantomJS et permet également de tester plusieurs résolutions (mobile, tablette, desktop). Vous pouvez retrouver Zeno Pixel sur le Github des Furets : https://github.com/lesfurets/zeno-pixel

Développement

Feature Branch

Le principe de la Feature Branch est que tout développement de fonctionnalités doit avoir sa propre branche, tirée depuis la branche principale. Elle est donc parfaitement isolée des autres développements. Lorsque le développement de celle-ci est terminé, le développeur émet une Pull Request afin que ses modifications soient prises en compte et intégrées Fig.5.

Continuous Merge

Le Feature Branching soulève des difficultés quant à l'intégration continue. Afin de pallier ce problème, Les Furets se sont appuyés sur le merge octopus de git et ont même développé une commande pour simplifier son utilisation : git-octopus. Après chaque commit, le serveur d'intégration continue va essayer de merger les n branches de features avec le master, et le résultat est commité dans une branche jetable appelée octopus. La sélection des branches à prendre en compte lors du merge se fait sur nommage de celles-ci (exemple : features/*). Enfin, c'est sur branche jetable que sont effectués tous les tests. Concernant la gestion des conflits, il n'y a aucune résolution donc s'il existe un conflit, le merge échoue automatiquement. Et si une feature branche empêche le build de l'application, il suffit de renommer la branche afin qu'elle ne soit pas prise en compte lors du prochain merge. Ce n'est pas du feature toggle mais ça s'en approche.

Conception

Fonctionnalités indépendantes

Pour livrer ce qui est prêt le plus tôt possible, les fonctionnalités développées doivent être indépendantes les unes des autres. Si l'une est terminée, mais qu'elle dépend d'une autre en cours de réalisation, elle ne pourra pas être livrée en production comme il se doit. Le découplage des fonctionnalités est primordial. Si vous êtes dans l'incapacité de faire cela, fusionnez les évolutions entre elles.

Fast feedback

Le fast feedback est une partie importante du Continuous Delivery. En déployant tous les jours, il est possible de tester une nouvelle fonctionnalité, et de prendre rapidement en compte le retour des utilisateurs. Mais l'avis de l'utilisateur n'est pas le seul, le feedback intervient à chaque étape du delivery, que ce soit après le commit, les tests d'intégration ou la phase de validation. Et cela grâce notamment aux logs et monitoring mis en place tout au long du processus. La décision de stopper ou de continuer intervient plus tôt, la réactivité de l'équipe est plus accrue et l'amélioration du produit est continue.

CONCLUSION

Le Continuous Delivery est l'une des composantes du manifeste Agile. Pourtant, peu d'entreprises le mettent en place pour plusieurs raisons, notamment l'investissement qui peut être conséquent sur des projets d'envergure. Mais son adoption devrait devenir de plus en plus courante grâce à l'aide d'outils qui simplifient sa mise en œuvre. On peut évoquer notamment Docker qui permet de déployer son application sur n'importe quel environnement au sein d'un conteneur logiciel. Ou encore Chef ou Puppet, des outils de gestion automatisée de configuration. Et bien sûr l'intégration continue avec un des serveurs les plus connus : Jenkins. Cette liste est loin d'être exhaustive, il existe pléthore d'outils pour cela, qu'ils soient Open Source ou bien sous licence propriétaire.

Fig.4

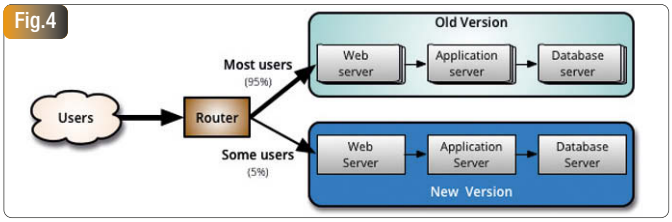


Fig.5

Traitement de données en Java 8 : les Streams sont parmi nous !

Comme tout développeur Java, j'ai attendu pendant longtemps les lambdas, Streams et Java 8. J'ai travaillé avec des outils Big Data tels que Flume Java et Map Reduce, et plus récemment avec JavaScript : ces technologies fournissent des fonctions map-filter-reduce. Maintenant qu'elles sont nativement intégrées à Java, j'ai eu envie d'assister à la conférence proposée par José Paumard sur « Java, JVM, Java SE/EE » lors de mon passage au Devoxx 2015.

Ariel Güelfi,
Technical Leader JAVA chez **PALO IT**

Slides de la présentation :
http://fr.slideshare.net/jpaumard/les-streams-sont-parmi-nous

Une des fonctionnalités les plus puissantes de Java 8 est sans conteste Streams module. Comme José Paumard l'a décrit, Streams fournit une solution pour traiter les gros volumes de données avec efficacité et facilité, en fournissant les fonctions Map/Reduce/Filter form et la parallélisation des traitements.
Sa présentation s'est focalisée sur trois aspects du problème en question :

- Streams API,
- GS Collestions,
- RxJava.

Streams et GS Collections ont pour but de traiter efficacement les structures de données alors que RxJava, par définition, est une bibliothèque pour utiliser conjointement des programmes asynchrones et event-based en utilisant des observable sequences.

L'API Streams

L'API Streams fournit des interfaces et des méthodes simples mais puissantes qui permettent une manipulation des données beaucoup plus aisée qu'avec l'API Collections. Une des fonctionnalités les plus marquantes est l'interface Collectors. On peut par exemple citer Group By et/ou la somme des autres attributs. Les Collectors peuvent être multiples, cumulatifs, et sont « lazy », c'est à dire qu'ils ne seront invoqués et exécutés que lorsque les résultats sont demandés. L'un des autres points importants est la facilité d'utilisation de la parallélisation des traitements. En appelant **iterable.stream().parallel(),** la JVM va exécuter toutes les transformations en parallèle, en créant des Threads utilisant tous les CPUS et les cores disponibles. Enfin, un nouveau concept apparaît : le Spliterator, avec lequel nous pouvons définir un Splitterable Iterator. Cet itérateur a la possibilité d'être partitionné et donc parcouru de manière parallèle.

GS Collections

https://github.com/goldmansachs/gs-collections

GS Collections est par définition « un supplément ou un remplacement du framework Java Collections ». Créé par Goldman Sachs en 2004, GS Collections fournit des implémentations basées sur le framework SmallTalk Collections, de la même manière que Google le fait avec Guava. Bags, Multimaps, MutableSets (etc.) sont des implémentations qui utilisent les Lambdas Java 8 pour fournir une puissante bibliothèque de gestion des collections. GS Collections peut aussi traiter les données en lazy et permet la parallélisation des traitements sur certaines interfaces.

RxJava

https://github.com/ReactiveX/RxJava

Comme évoqué lors de la présentation, RxJava est une implémentation du pattern Reactor. Il est possible de définir un Observer qui souscrit à un Observable notifié de tout changement et qui agit en conséquence. La bibliothèque est très complète et complexe. José Paumard a très bien expliqué ces points clés. Une des remarques les plus importantes est que RxJava crée des Deamon Threads pour traiter les données, ce qui signifie que ces Threads n'existeront que si le Thread principal existe. Enfin, un Observable peut avoir plusieurs Observers, contrairement à l'API Streams qui ne peut en n'avoir qu'un, ce qui constitue un avantage.
Les trois options sont différentes et il faut évaluer précisément le besoin afin d'en choisir une :

- L'API Stream est très simple d'utilisation et le code facile à comprendre,
- GS Collections reste assez simple avec l'utilisation des Lambas Java 8, comme l'API Streams,
- Si le besoin est de transformer des données comme le permettent Streams ou GS Collection, RxJava demandera un code plus complexe et moins efficace.

La présentation s'est illustrée par quelques tests simples de performance. L'API Streams a montré les meilleurs résultats, que ce soit en temps de réponse ou en consommation mémoire. Juste derrière, GS Collection montre également de bonnes performances. RxJava répond à d'autres problèmes : il est donc difficile de le comparer aux deux premiers.

2020, 2035 et au-delà : les robots de demain

Le Devoxx 2015 était, comme à son habitude, le lieu de rencontre des "geeks" Java. Mais cette année, il était placé sous le signe des possibilités infinies, avec en particulier un thème repris sur les keynotes du jeudi et du vendredi : « FUTURE <DEVOXX> ».

Raphaël Jeannin, SOAT

Après avoir envahi nos chaînes de production industrielles, on peut voir des robots de plus en plus sophistiqués prendre place dans nos maisons. Aujourd'hui, des sociétés voudraient franchir une nouvelle étape, en proposant des robots capables d'apprentissage. Ils deviendront nos compagnons et réaliseront de plus en plus de tâches quotidiennes fastidieuses.

Alors, quand les robots cesseront-il d'être des sujets de curiosité et deviendront-ils des assistants personnels évolués ?

Pourront-ils embarquer toute la puissance et l'intelligence nécessaires, ou devront-ils toujours être connectés à des centres de calculs, comme l'est actuellement l'assistant vocal d'Apple, SIRI ?

Reconnaissance et identification

Les robots commencent à maîtriser la marche. Ils sont capables de voir en 3 dimensions. Ils peuvent maintenant analyser les visages pour reconnaître les personnes qui leurs sont proches et savent identifier des bruits familiers. Ils sont donc capables de réaliser des tâches de reconnaissance et d'identification, ce qui va leur permettre de suivre un objet ou une personne qui les accompagne.

Les difficultés de reconnaître et d'identifier sont partout comme, par exemple, dans le fait que le robot prélève l'information dont il a besoin autour de lui. Contrairement à une application sur ordinateur, l'utilisateur ne saisit pas les données (la question ainsi que le contexte). C'est au robot de dissocier lui-même les informations utiles du bruit ambiant. S'il se trompe sur les données du problème, sa réponse sera à fortiori erronée.

Autre exemple, ils peuvent entendre grâce à des micros, mais sont gênés par le son de leur propre voix. Ainsi, ils sont incapables de parler et d'entendre simultanément.

Robotique développementale

Avec la robotique développementale, on revient aux sources. Plutôt que d'essayer de copier les comportements élaborés, tels que peut les réaliser un être humain, ne pourrait-on pas apprendre au robot à les apprendre par lui-même ?

La théorie du Flow, élaborée par le psychologue Mihaly Csikszentmihalyi, indique qu'un individu est dans un état de concentration maximale s'il se trouve au-delà d'un seuil d'ennui et en-deçà d'un seuil d'anxiété (*Cf. Flow Channel Wave illustration extraite de "The Art of Game Design" de Jesse Schell*) Fig.1.

Pierre-Yves Oudeyer s'appuie sur cette théorie pour doter les robots de la curiosité. Parmi les expérimentations que le robot va réaliser, certaines vont lui apporter plus d'informations, plus de reconnaissance et donc plus de plaisir. Bien sûr, aucun des comportements n'aura été pré-programmé, et c'est donc le robot qui va sélectionner ceux qui lui apportent le plus, et éliminer les autres. En multipliant ces expérimentations, on pousse le robot à s'intéresser à des activités nouvelles et donc à évoluer par l'auto-apprentissage. Le même schéma, basé sur des robots curieux et doués de la capacité d'émettre et d'entendre des sons, permet de développer de véritables langages. Grâce à l'interaction de plusieurs robots, ils vont petit à petit se mettre d'accord sur des sons pour designer des objets.

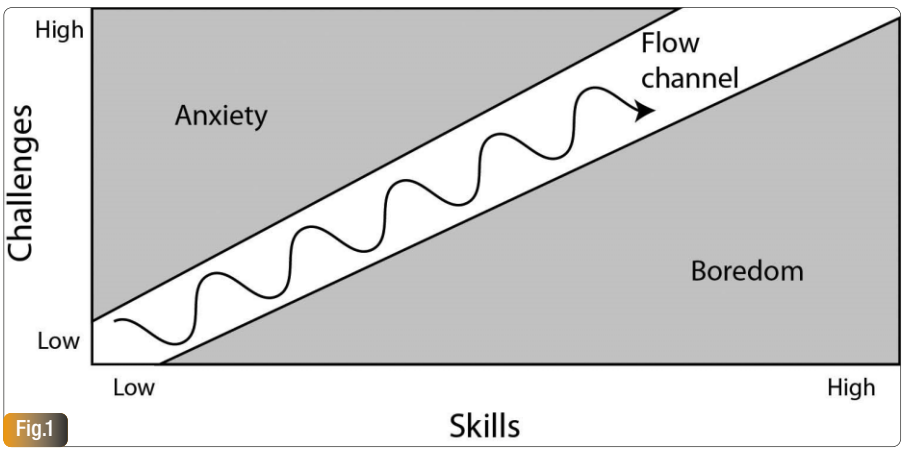


Fig.1

Le champ lexical est entièrement contextuel et, si l'expérience est renouvelée avec d'autres objets et d'autres couleurs, alors la langue obtenue sera différente.

2035

A l'horizon 2035, l'objectif pour certains robots pourrait-être de devenir des « elfes de maison ». C'est à dire des aides non plus seulement à mêmes d'exécuter des tâches simples, mais capables de noter les habitudes, de comprendre les situations ou les comportements et finalement d'anticiper nos besoins.

Quand on côtoie quelqu'un régulièrement, il est très difficile de prendre conscience des petits changements quotidiens. C'est vrai pour les enfants, que leurs parents ne voient pas grandir, mais c'est surtout vrai pour les personnes âgées. Celles-ci sont très routinières et un robot domestique sera capable de déterminer les habitudes prises quotidiennement, hebdomadairement ou sur des périodes annuelles. Une fois le profil-type établi, il sera simple de noter les changements d'habitude. Il détectera aussi les petites évolutions quotidiennes, qu'il pourra consolider et, au-delà d'un seuil fixé, déclenchera une alerte. Les informations collectées seront autant de symptômes qui faciliteront le diagnostic du médecin.

Déshumanisation

On peut penser que l'utilisation de robots domestiques va renfermer les utilisateurs. Les rendre dépendants d'outils plus que de leurs proches. Mais c'est avant tout un moyen qui permettra à une personne âgée de rester plus longtemps à domicile. En créant l'interaction, le robot oblige la personne âgée à sortir de son silence, à interagir avec son environnement, à s'exercer quotidiennement. Enfin, dans un cercle restreint, le robot attise la curiosité et déclenche la conversation et l'échange.

Vie privée

Le problème vient de la capacité forcement

limitée de traitement d'une unité mobile. Les robots peuvent entendre puis comprendre une requête énoncée par la parole, mais pour ce faire, ils doivent échanger avec des centres de calcul d'une puissance mille fois supérieure à la leur.

A l'image des assistants vocaux « SIRI » ou « OK google », la demande est enregistrée, envoyée pour décodage à un serveur qui va proposer une réponse au robot, qui n'aura plus qu'à l'énoncer.

On le voit, pour être de plus en plus pertinent, le robot va collecter d'énormes quantités d'informations qu'elles soient auditives ou visuelles. Dans le cas d'un robot de maison accompagnant une personne âgée, il aura certainement une connaissance plus approfondie des habitudes et des capacités de celle-ci que ses proches, voire que la personne elle-même.

Ces données sont à caractère hautement personnel, voire intime dans certain cas.

Or, toutes vont transiter, être analysées sur des réseaux et dans le «Cloud ».

La solution passera probablement par des algorithmes de cryptage, mais seront-il à l'abri de tout décryptage ? Et seront-ils exempts de bugs ?

La sécurité

En 2014, trois bugs majeurs (Heartbleed, Shellshock et dans SSL v3) ont été découverts, et un grand nombre de sites Web ont été touchés.

Il y a là une première responsabilité des développeurs : il faut apprendre à développer avec pour objectif « zéro bug », et se doter des outils nécessaires pour garantir la sécurité des utilisateurs de nos programmes.

Les vulnérabilités peuvent toucher aussi bien des bibliothèques logicielles récentes que des bibliothèques largement utilisées depuis longtemps.

L'âge, ou la durée de vie, d'un composant n'est pas une garantie de sa qualité. Quel que soit le logiciel, il est nécessaire de maintenir une veille technologique sur les failles qu'il pourrait contenir et de mettre à jour en permanence ses composants.

Open source

L'utilisation de logiciel open source pourrait être une partie de la réponse. Un logiciel open source est un morceau de code source que chacun peut télécharger, analyser, comprendre ou encore faire évoluer, avec un « fork ».

Ces éléments apportent des garanties en termes de contenu et donc une sécurité pour l'utilisateur. Si le logiciel contient une « back door », il est probable qu'elle soit vite découverte, de même pour les bugs.

L'open source permet aussi de créer des communautés d'utilisateurs, qui vont alimenter des forums de discussion, ce qui pourrait faire émerger les défauts et justifier les choix de conception. Mais un robot est un objet physique, donc qui ne peut pas être open-source.

Cependant, il y a une importante part de logiciel pour faire fonctionner le robot, et ce dernier est fabriqué suivant des plans qui peuvent être open source. La fabrication d'un robot pouvant s'avérer fastidieuse, même avec des plans précis, l'impression 3D va permettre de se libérer de cette contrainte : en rendant possible le téléchargement des plans et l'impression des pièces, il sera donc possible de réaliser le montage d'un robot fonctionnel en quelques heures.

Avec l'impression 3D, on entre dans l'ère du « rapid prototyping » sur des objets : expérimenter des formes de robots différentes, corriger ou améliorer les fonctions, spécialiser des robots pour réaliser des tâches pour lesquelles ils n'étaient pas conçus initialement, tout cela deviendra possible.

Enfin, pour en revenir à l'Homme, il est possible de concevoir et de réaliser des prothèses toujours plus performantes et mieux adaptées.

Ces éléments mécaniques, issus de la robotique et optimisés au plus fin, vont permettre de pallier les défaillances de l'Homme.

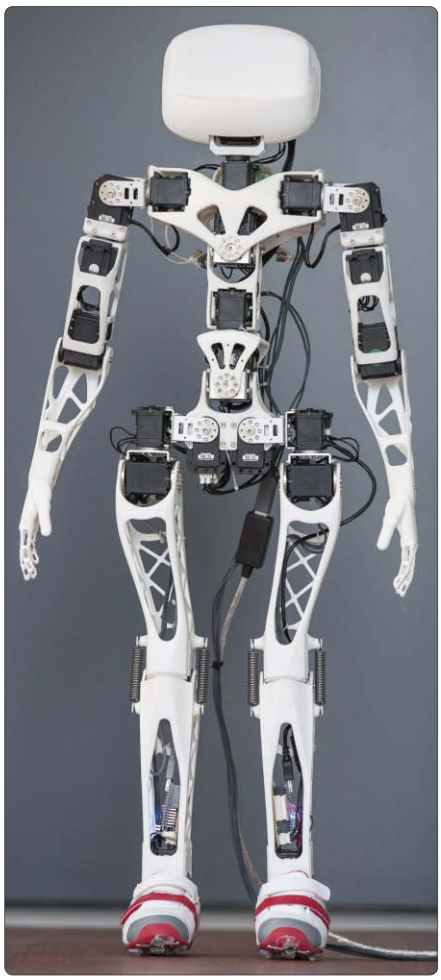
On l'a vu, les robots n'ont pas fini de nous étonner. Même si l'on peut anticiper les capacités croissantes de ces nouveaux compagnons, nous ne sommes aujourd'hui qu'au balbutiement et le chemin est semé d'embûches.

Ils vont évoluer du simple statut de "mécanique qui étend notre main" à celui d'élément autonome. Les conséquences de la robotique développementale seront-elles acceptées par nos sociétés ?

Les technologies open source basées sur le partage de la connaissance seront-elles la clé pour se prémunir des dérives ?

Une chose est sûre, les robots seront toujours plus présents autour de nous, alors il vaut mieux apprendre à vivre avec eux !

D.R.



Poppy

Références :

[DEVOXX]
Le futur de la robotique personnelle,
Rodolphe Gelin

[DEVOXX]
La problématique du contrôle des technologies de l'information,
Eric Filiol

[DEVOXX]
Un robot peut-il apprendre comme un enfant?
Pierre-Yves Oudeyer

Robotique développementale
2014 :
Rapport n° 1 de la CERNA,
Commission de réflexion sur l'Éthique de la Recherche en sciences et technologies du Numérique (résumé)

Drupal 8 : tour d'horizon des nouveautés

Tarik LARBI
Chef de projet technique
chez **Trained People**

Etes-vous prêt pour Drupal 8 ?

Avec plus de 200 fonctionnalités et plusieurs améliorations, Drupal 8, annoncé pour une sortie prochaine, sera un leader des produits open source qui va en séduire plus d'un. Drupal 8 est actuellement en version bêta. Il passera en version RC (release candidate) une fois que tous les bugs critiques passent à zéro. Vous pouvez voir la tendance actuelle et la fréquence de résolution des bugs par la communauté à cette URL : https://www.drupal.org/drupal-8.0/get-involved.

Deux des points clés qui sont en toile de fond pour cette version majeure sont, la courbe d'apprentissage, et le coût d'une migration personnalisée. Ce sont deux préoccupations pas négligeables pour les développeurs et les propriétaires de sites. Ils représentent, à tous deux, un grand investissement de temps et d'argent. En effet, si vous n'êtes pas habitué à la programmation orientée objet, la nouvelle structure des fichiers Drupal 8 va vous sembler un peu déconcertante au premier abord.

Puis, la mise à niveau vers Drupal 8 sera aussi fastidieuse que la précédente mise à jour (Drupal 7) qui exige une courbe d'apprentissage technique, même si vous êtes un expert Drupal sur des versions antérieures. En revanche, gardez à l'esprit que cette courbe d'apprentissage n'est pas en vain, mais au contraire, elle fera de vous un meilleur développeur, plus moderne; la bonne nouvelle est que les futures mises à jour seront, probablement, beaucoup plus digestes.

La prochaine version de Drupal est une victoire sans équivoque. Certains développeurs Symfony auront peut-être envie de rejoindre la communauté Drupal. De même, que les développeurs Drupal actuels vont acquérir des nouvelles compétences et des nouvelles normes utilisées largement dans la communauté PHP.

Maintenant que les caractéristiques sont gelées, nous allons jeter un coup d'oeil sur les nouveautés et les concepts généraux. Avant de commencer souvenez-vous combien il était difficile de passer de la version 6 à la version 7. Enjoy !

Mon premier module Drupal 8

D'emblée commençons par créer notre premier module Drupal 8. Maintenant, si vous avez travaillé avec Drupal avant, vous êtes probablement familier avec le fichier de point info. Il inclut des métadonnées sur le module, comme le nom du module, la description et les dépendances sur d'autres modules. Avec Drupal 8 le fichier point info fait place à un fichier YAML avec une extension yml. Maintenant, l'avantage de cela est qu'à l'aide de cette norme spécifique soutenue par d'autres bibliothèques, nous pouvons faire des choses comme analyser les données et les implémenter dans votre projet. Le contenu du fichier first.info.yml :

```
name: Mon premier module Drupal 8
description: Mon premier module de test pour Drupal 8
core: 8.x
package: Mes modules
type: module
```

Le contenu du fichier nommé first.module :

```
<?php

/**
* Implements hook_menu().
*/
function first_menu() {
 return array(
  'hello-word' => array(
   'title' => 'Hello word !',
   'route_name' => hello.hello_word,
  )
 );
}
```

Ensuite un fichier nommé first.routing.yml. Le fichier de routage avec les lignes suivantes :

```
hello.hello_word:
 pattern: '/hello-word'
 defaults:
  _content: '\Drupal\first\Controller\FirstController::helloWordPage'
 requirements:
  _permission: 'access content'
```

Notre controller FirstController.php

```
<?php

namespace Drupal\first\Controller;

use Drupal\Core\Controller\ControllerBase

/**
* Notre Controller du module first Drupal 8.
*/
class FirstController extends ControllerBase {
 public function helloWordPage() {
  return array(
  '#markup' => t('<p><Hello World/p>'),
  );
 }
}
```

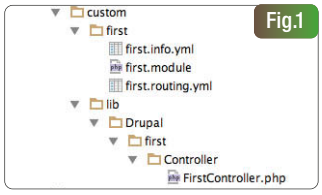


Fig.1

Structure de fichier de notre module Drupal 8 compatible PSR-4. Drupal 8 est passé de PSR-0 à PSR-4 en Juin (Fig1). Qu'est-ce que cela signifie ? Le terme PSR (PHP Standard Recommandation) décrit une spécification pour le chargement automatique des classes (Autoloading) à travers le chemin des fichiers. L'auto-chargement permet d'inclure des fichiers sans avoir recours à des déclarations de type "include ou include_once", en se basant sur le chemin des namespaces. L'autoload permet aussi le "Lazyloading", soit ne charger que les classes qui vont être utilisées.

Donc, muni de nos deux fichiers et de notre point info yml fichier rempli, nous devrions être en mesure de voir notre module apparaître sur la page liste des modules.

On remarque que dans Drupal 7, cette page est attachée à l'item du menu d'administration Modules. Plus maintenant, sur Drupal 8 il faut cliquer sur Extend (étendre). Et voilà !

actency

A CREATIVE AGENCY POWERED BY DRUPAL

Audit
- applicatif
- performance
- sécurité

Maintenance
- préventive
- corrective
- évolutive

Architecture technique

Développement

Théming

Support

Formation

Hébergement

PROPULSEZ VOTRE PROJET
AVEC L'AGENCE **100% DRUPAL**

ACTENCY PARIS
82 rue d'Hauteville / 75010 PARIS

ACTENCY STRASBOURG
45 avenue de Colmar / 67000 STRASBOURG

01 47 70 47 70

www.actency.fr

contact@actency.fr

C'en est fini de variable_get et variable_set

Si vous essayez d'utiliser les fonctions variable_get et variable_set lors de votre prochain projet avec Drupal 8 vous risquez d'être bien surpris en ayant une grosse Fatal error : Call to undefined function. Donc, à ce stade, si vous êtes en transition de Drupal 7 à Drupal 8, vous pourriez être un peu dérouté en disant "Oh non !".

Donc, que faisons-nous sans variable_get et variable_set ? Eh bien, il semble donc essentiel que la nouvelle version de Drupal soit en mesure de stocker des ensembles de variables. La réponse est State, une API permettant de stocker toute sorte de variable système, plus besoin d'utiliser variable_get et variable_set. Jetons un coup d'oeil sur le fonctionnement de cette API.

```
// Get a value.
$valeur = \Drupal::state()->get('key');
// Multiple key/value pairs:
$pairs = \Drupal::state()->getMultiple($keys);

// Set a value:
\Drupal::state()->set('key','value');
// Set multiple values:
\Drupal::state()->setMultiple($keyvalues);
// Delete a value:
\Drupal::state()->delete('key');
```

Finies les fonctions drupal_set_title et drupal_get_title

Les deux fonctions drupal_set_title et drupal_get_title risquent aussi de vous surprendre si vous voulez les utiliser !

Exemple un titre static dans un fichier de configuration :

```
block.admin_add:
  path: '/admin/structure/block/add/{plugin_id}/{theme}'
  defaults:
    _controller: ...
    _title: 'Configure block'
  requirements:
    _permission: 'administer blocks'
```

Exemple d'un titre dynamique dans un fichier de configuration

```
mymodule.test:
  path: '/mymodule/test'
  defaults:
    _controller: '\Drupal\mymodule\Controller\Test::getContent'
    _title_callback: '\Drupal\mymodule\Controller\Test::getTitle'
```

Notre Controller :

```
<?php
class Test {

  /**
   * Returns a page title.
   */
  public function getTitle() {
    return  'Foo: ' . \Drupal::config()->get('system.site')->get('name');
  }

  /**
   * Returns a page render array.
   */
  public function getContent() {
    $build = array();
    $build['#markup'] = 'Hello Drupal';
    return $build;
  }
}
```

Drupal 7

```
$title = drupal_get_title();
```

Drupal 8

```
$request = \Drupal::request();
$route_match = \Drupal::routeMatch();
$title = \Drupal::service('title_resolver')->getTitle($request, $route_match->getRouteObject());
```

DU CÔTÉ DE PHP ET SYMFONY 2

Programmation orientée objet (POO)

Drupal 7 essuie souvent des critiques pour son code source procédural au lieu d'une programmation orientée objet (POO). Par exemple, le mot «class» ne figure pas dans le code de Drupal aussi souvent qu'on pourrait s'y attendre. Certains ont vu cela, comme une carence.

Pourquoi programmer en OOP ? Comme son nom l'indique, il s'agit d'une méthode de programmation qui permet de créer des objets. Les trois principales caractéristiques de la OOP sont le polymorphisme, l'héritage et l'encapsulation.

Les avantages et les inconvénients de la OOP.

Les principaux avantages :

- Atteindre un niveau d'abstraction qui permet de modéliser des architectures applicatives complexes,
- Améliorer la productivité, avec un refactoring du code qui en facilitera la maintenance,
- Accroître la productivité et la stabilité applicative, et faciliter le travail collaboratif par la même occasion.

Les inconvénients :

- La courbe d'apprentissage, notamment en raison de son niveau d'abstraction,
- Des compétences plus élevées dans l'approche objet,
- La lecture du code qui peut être déroutante.

Les bases de la POO peuvent sembler quelque peu théoriques et abstraites, mais une pratique régulière vous confortera dans votre courbe d'apprentissage.

Correctement utilisée, dans le cadre de votre prochain projet Drupal 8, et sur lesquels plusieurs développeurs sont amenés à travailler, la POO est idéale. Elle permet de produire du code plus efficace, évolutif et portable. La POO ouvre les portes du niveau supérieur de la programmation.

Namespace et autoloading PHP

Depuis la version 5.3 de PHP une nouvelle fonctionnalité appelée namespace fait son apparition. Cette notion de "package" n'est pas une nouveauté pour des langages comme Java ou C#. PHP n'a pas inventé la poudre. Néanmoins, chaque nouveauté a un but. Sans namespace, vous ne pouvez pas avoir deux noms de classes ou deux fonctions ayant le même nom. Ils doivent être uniques. Effrayant non ! Voilà pourquoi le but de namespace, c'est de nous permettre de contourner cette contrainte. De plus, ça nous permet de structurer nos packages contenant des classe PHP et de naviguer dans cette nomenclature.

Composer

Parmi les nouveautés, Drupal 8 embarque Composer dans le cadre de sa distribution principale. Composer est un outil de gestion de dépendances pour PHP. Il

permet de gérer des bibliothèques et des utilitaires séparément pour chaque projet. Il est particulièrement utile quand on veut grouper des tests automatiques. Tout comme l'App Store pour Mac ou le Play Store pour Android, Composer est pour PHP. Il permet de télécharger des bibliothèques: on trouve un grand référentiel des bibliothèques hébergé sur ce site https://packagist.org/ .Grâce à Composer, PHP a fait un réel bond en avant dans le domaine de la réutilisabilité et du partage. Il permet aux développeurs de produire du code non dépendant du Framework. Il n'est maintenant pas rare de voir des Frameworks PHP utiliser des morceaux de code d'autres Frameworks (coucou Symfony !).

YAML

L'acronyme YAML veut dire "YAML Ain't Markup Language", soit "YAML n'est pas un langage de balises". D'après la documentation officielle http://yaml.org/spec/1.2/spec.html, YAML est un langage de sérialisation de données conçu pour être lisible par des humains.

Pourquoi YAML ? Premièrement, nous sommes à la recherche d'un format de fichier qui est à la fois analysable et modifiable par l'ordinateur et l'humain. Nous ne voulions pas d'un format Drupal spécifique (fichiers info). Nous ne voulions pas de XML, car alors le schéma est contraignant pour comprendre la structure. JSON ne permet pas des commentaires et les caractères non-ASCII doivent être échappés, ce n'est pas agréable pour les humains. Donc, YAML sera le meilleur compromis par rapport aux autres formats.

Une des choses que vous êtes susceptible de faire lorsque vous écrivez un module custom avec Drupal 7, ou un thème, est que vous voulez inclure une librairie JavaScript. Habituellement, votre premier reflexe est d'utiliser le hook_library_info(). Or, dans Drupal 8 la fonction PHP est remplacée par un fichier au format YAML. Voyons voir comment faire si on souhaite ajouter la librairie colorbox pour notre module contrib colorbox.

Le nouveau fichier au format YAML aura la convention de nommage nomdumodule.libraries.yml . Donc, le nôtre sera colorbox.libraries.yml et résidera dans le niveau supérieur du répertoire de module, comme tous les autres fichiers au format YAML.

```
colorbox:
  version: VERSION
  js:
    js/colorbox.js: {}
  dependencies:
    - core/jquery
    - core/drupal
```

Twig (Fig3)

Twig, tout comme Symfony, est le fruit du travail de la société SensioLabs. Drupal 8 utilise tout naturellement Twig comme theme engine (moteur de template). Drupal 8 remplace son theme engine PHPTemplate par Twig qui s'inscrit dans une logique de séparation de couches.

Pourquoi Twig ?

- Plus de sécurité au niveau du thème en raison des possibilités limitées fournies par la syntaxe basée sur des balises,
- Plus de PHP mixé avec du HTML sur des templates,
- Simplification du thème engine en raison de la réduction des fonctions de prétraitement,
- Facilité à apprendre la syntaxe basée sur des balises,
- Réutilisabilité des templates à travers des inclusions de fichiers de template,
- Interprétation des balises par des IDE (PHPStorm, NetBeans, Eclipse, Sublime, etc.
- Bonne documentation http://twig.sensiolabs.org/documentation.

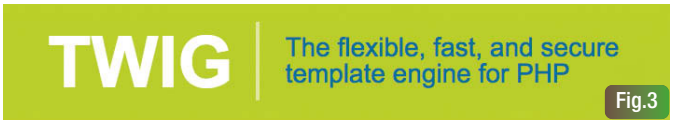


Fig.3

Sur Drupal 7, le texte soumis sur des templates doit passer par un traitement avec check_plain() afin de prévenir d'une vulnabilarité Web de type Cross Site Scripting (XSS). Comparons deux snippets du theme Bartik de Drupal 7 et Drupal 8. Ces deux snippets ont une fonction identique.

Le contenu du template node.tpl.php pour Drupal 7 :

```
<?php print render($title_prefix); ?>
<?php if (!$page): ?>
<h2<?php print $title_attributes; ?>>
<a href="/<?php print $node_url; ?>"><?php print $title; ?></a>
</h2>
<?php endif; ?>
```

Le contenu du template node.html.twig avec Drupal 8 :

```
{{ title_prefix }}
{% if not page %}
<h2{{ title_attributes }}>
<a href="/{{ node_url }}">{{ label }}</a>
</h2>
{% endif %}
{{ title_suffix }}
```

La partie la plus intéressante dans Twig est l'héritage des templates.

Le couple Drupal 8 et Twig élimine le besoin de copier et coller les fichiers de template de votre thème custom. Tel que SASS et LESS (Compilateur et générateur dynamique de CSS), Twig va réduire considérablement la quantité des templates et de code, par la même occasion, afin d'organiser votre structure de template de votre thème.

```
{% Extends "themes/bartik/templates/node.html.twig"%}
```

Tout comme l'adoption de SASS et LESS, Twig se révélera à la fois comme une victoire et un défi pour les développeurs qui adopteront cette gestion de workflow du theming Drupal.

L'injection de dépendances (Dependency injection)

Drupal 8 utilise Symfony 2 sous le capot. L'injection de dépendance est l'épine dorsale de Drupal 8. Presque chaque portion de code que vous allez écrire dans Drupal 8 va toucher, en quelque sorte, ce design pattern. Ainsi, comprendre le fonctionnement structurel de ce composant vous permettra de démystifier cette magie noir et vous aidera à l'utiliser correctement en toute confiance. L'injection de dépendances est un mécanisme qui permet d'implémenter le principe de l'inversion de contrôle. Il consiste à créer dynamiquement (injecter) les dépendances entre les différentes classes en s'appuyant sur un fichier de description Yaml ou de manière programmatique. Ainsi les dépendances entre composants ne sont plus exprimées dans le code de manière statique mais déterminées dynamiquement à l'exécution. (Source : http://fr.wikipedia.org/wiki/Injection_de_dependances) Qu'est-ce que cela signifie pour nous, les développeurs Drupal ? Cela signifie que nous pouvons écrire du code plus efficace en réduisant la charge. Nous chargeons seulement ce dont nous avons besoin et, par la même occasion, on renforce la modularité.

Gestion de la configuration (CMI)

Drupal 8 est livré avec un système de gestion de configuration basé sur un système de fichier, ce qui rend plus facile à déployer des changements de

Fig.4

configuration tels qu'un nouveau type de contenu, des Views ou des champs (Fields), d'un environnement de développement vers la production (fig 4.) De plus, il vous permet même d'utiliser le contrôle de versioning de votre configuration. Gardez vos données de configuration dans des fichiers, et séparez ces derniers des environnements; s'en passer dans la base de données rendra le déploiement moins contraignant. En un mot, finies les Features !

Drupal 8 et l'accessibilité

Drupal 8 a ajouté un support étendu pour les technologies d'accessibilité standard y compris WAI-ARIA (Web Accessibility Initiative - Accessible Rich Internet Applications) avec du HTML5 plus sémantique. Avec la nouvelles API ARIA Live TabManager, Drupal 8 offre un contrôle centralisé pour des applications Internet plus riches.

Beaucoup de petites améliorations ont été apportées dans le contraste des couleurs, et les tailles de police, qui mettront à profit des boîtes de dialogue de jQuery UI. De plus, Views est maintenant dans le core et donc beaucoup plus accessible pour offrir aux visiteurs et aux administrateurs une expérience optimale.

Fig.5

Drupal 8 embarque le projet Spark (Fig 5.)

L'un des principaux centres d'intérêt de Drupal 8 est l'expérience de contribution (Authoring Experience) out-of-the-box pour les contributeurs de contenu.

Le projet Spark est une initiative lancée par Dries Buytaert dans le but d'améliorer l'expérience de création de contenu dans le core de Drupal par defaut.

Editeur WYSIWYG (Editeur de texte enrichi)

Drupal 8 est désormais livré avec un CKEditor un éditeur out of the box. Sans aucun recours à l'installation de librairies JavaScript ni de modules contrib. De plus, la nouvelle version de CKEditor offre plusieurs widget très intuitifs, développés spécifiquement pour Drupal dans le respect de la structure HTML.

Edit in place

Drupal 8 offre une fonctionnalitéde l'édition rapide : consolider. En passant la souris sur le contenu on verra apparaître un stylo en picto permettant de cliquer dessus et d'accéder à l'édition rapide (edit in place). Une fonctionnalité qui permet aux contributeurs d'éditer du contenu de page en front-end, sans passer par le back-office (Fig 6).

De plus avec Spark, l'édition de contenu ne sera plus une contrainte sur des mobiles ou tablettes. Drupal 8 offre un back-office complètement responsive et plus user-friendly grâce toujours au projet Spark. Plus la peine de passer du temps à télécharger des librairies JavaScript dans tous les sens pour construire votre Texte enrichi (RTE Rich text editor). Drupal 8, l'embarque nativement dans le core.

L'interface de création de contenu de Drupal 8 à subi une nouvelle mise en forme. Deux colonnes sont mises à disposition pour améliorer l'expérience de contribution.

Une pour tous les champs concernant le contenu. Et l'autre, pour les extras, des paramètres optionnels, afin de permettre au contributeur de se concentrer davantage sur la tâche à accomplir. (Fig 7)

Gestion des langues et de la traduction (Fig 8)

Dans la nouvelle version de D8 plus besoin de télécharger des modules pour gérer le multilingue. La traduction occupe une place importante dans le programme d'installation.

Contrairement à Drupal 7 où vous êtes confronté à un mur d'instructions textuelles sur la façon de localiser et de télécharger un fichier de traduction (.po) et, de placer ce fichier dans un répertoire spécifique. Drupal 8, quant à lui, automatise tout ce processus de traduction à partir du choix de langue. Environ 100 langues au choix pour installer Drupal 8.

La nouvelle version de Drupal est également livrée avec une meilleure capacité de détection de langue basée sur le navigateur Web. Il tentera d'identifier automatiquement votre langue préférée pour cette installation sur la base de la configuration de votre navigateur. Ainsi, dans la plupart des cas, vous aurez probablement appuyé simplement sur le bouton pour continuer l'installation.

Sur Drupal 8, après son installation avec une langue étrangère, vous pouvez le garder uniquement avec cette langue, sans une cohabitation "forcée" de l'anglais qui figure un peu partout comme une option alternative. Comparé à Drupal 7 où l'anglais n'était pas possible à supprimer.

Conclusion

Bien que beaucoup de ces concepts pourraient sembler étranges à des développeurs Drupal expérimentés, il n'en demeure pas moins que c'est le chemin incontournable vers la maîtrise de Drupal 8. Si vous faites ce saut, vous allez devenir un développeur plus compétent, et de nombreuses opportunités s'ouvriront à vous.

Pour beaucoup de développeurs c'est le bon moment pour commencer à se familiariser et à apprendre de nouveaux concepts et des bonnes pratiques. Comment organiser son code, comment faire du code réutilisable, comment construire des projets évolutifs, etc. Maintenant, vous avez la possibilité d'apprendre ces choses tout en approfondissant vos connaissances et votre expertise Drupal.

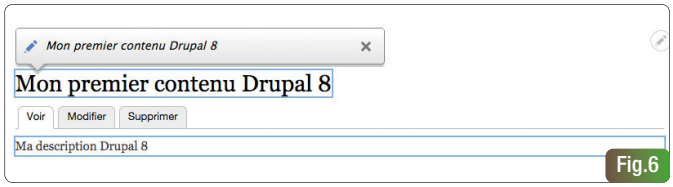


Fig.6

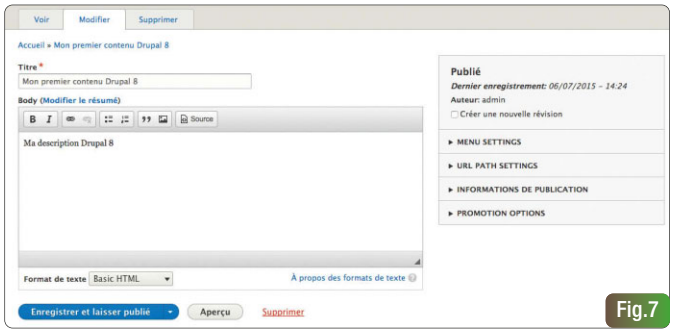


Fig.7

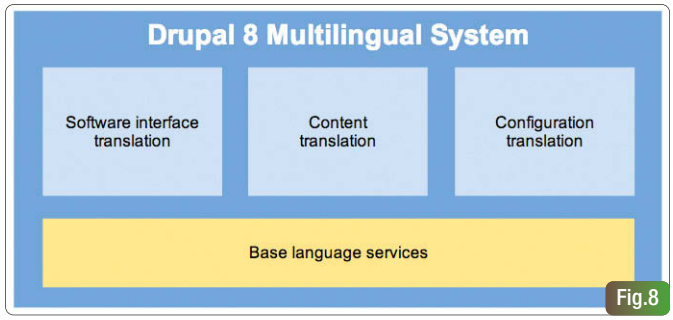


Fig.8

TRAINEDPEOPLE

FORMATION & CONSULTING DRUPAL

NOUVEAU FORMATIONS Drupal 8

Formation Drupal

DÉVELOPPEZ VOS COMPÉTENCES DRUPAL

8 ans d'experience

220 jours de formation par an

800 sociétés nous font confiance

+DE 1200 stagiaires ont suivi nos formations

10 formations personnalisables

PROGRAMMES DE FORMATION DRUPAL 7 ET 8

- CONTRIBUTEUR / PUBLISHER
- RESPONSIVE DESIGN
- WEBMASTER
- DÉVELOPPEUR
- COMMERCE
- DÉPLOIEMENT
- THEMEUR / DESIGNER
- PERFORMANCE & SÉCURITÉ
- SYMFONY2 AVEC DRUPAL
- BACKBONE

ACCOMPAGNEMENT DE PROJET DRUPAL

UNE ÉQUIPE D'EXPERTS À VOTRE SERVICE

CONSEIL · ASSISTANCE TECHNIQUE · AUDIT · SUPPORT · MAINTENANCE

Plus d'infos sur le site
www.drupalfrance.com

T / +33 1 85 08 22 65
F / +33 1 48 24 95 80
8 rue du Faubourg Poissonnière / 75010 Paris / contact@trainedpeople.com / www.trainedpeople.com

Les migrations sur Drupal 8

Aperçu d'une roadmap ambitieuse

Ceux qui ont tenté l'expérience de migrer des sites Drupal d'une version majeure à une autre (Drupal 5 vers 6 ou encore aujourd'hui de 6 vers 7) savent à quel point une telle opération peut être longue, fastidieuse et parfois même vouée à l'échec en termes de rentabilité. Bien des agences ont cédé à la tentation de reprendre tout à zéro et de reconstruire intégralement un site puis de migrer les contenus séparément.

Nicolas LOYE,
Chef de projet technique chez Actency. Développeur drupal depuis 2006. Nicoloye sur drupal.org.

Les deux méthodes fonctionnent avec plus ou moins de réussite selon les projets, leur complexité et les budgets engagés. Dressons un rapide état des lieux des approches actuelles avant de nous pencher sur les promesses faites par Drupal 8 pour pousser plus loin encore les capacités de migration du core.

Migrer sur Drupal aujourd'hui

Migration entre deux versions majeures

Dans les versions antérieures ce type de migration passe par l'utilisation d'une partie du core proche de celle qui réalise les mises à jour de base de données lors des maintenances ordinaires. La principale fonction de ce processus est l'altération des éléments en base de données afin de rendre leur structure compatible avec la nouvelle structure attendue par la version cible.

Le core remplit parfaitement sa tâche dans la mise à jour des éléments de base de données mais se borne à ce strict usage. Il n'offre aucune possibilité d'extension ou d'altération qui permette d'introduire des actions spécifiques personnalisées à une migration donnée. Il suffit de jeter un oeil au fichier includes/update.inc de Drupal 7 pour s'en rendre compte, il s'agit d'un chemin de migration strict qui définit des méthodes figées se bornant à leur tâche de migration d'éléments du core.

Migration après reconstruction

Pour beaucoup de sites, il est légitime de penser leur migration différemment. Le passage d'une version du core à une autre est généralement un moment propice à une refonte fonctionnelle. Ce faisant il est possible que le socle de modules diffère entre la version source et sa nouvelle mouture mais bien souvent le contenu, les utilisateurs et même une partie des configurations doivent être conservés. Attachons-nous au module Migrate en particulier. C'est un module plébiscité par la communauté pour diverses raisons.

Premièrement il offre une architecture OOP, fait trop rare au sein des versions 7 et antérieures : celle-ci lui confère une très grande flexibilité en termes de développement. Il offre une gamme de méthodes suffisamment vastes pour accueillir des sources de données multiples (base de données, flux XML, JSON, etc) et les manipuler à sa convenance. Il ne fixe pas de structure, le développeur est libre de définir les actions à réaliser durant sa migration. Il permet la création de stubs lorsqu'un contenu en référence un autre qui n'a pas encore été créé. Il offre une possibilité de rollback des données en cas de problème. Enfin il met à disposition des commandes drush permettant de lancer ses migrations depuis une ligne de commande. Ce module permet donc de créer des migrations totalement personnalisées, ne se bornant pas simplement à de la copie de contenu mais permettant tout type d'actions sur l'ensemble du site de destination. Il est ainsi possible, par exemple, de prévoir une méthode de copie de médias au sein d'une classe de migration, ou bien une méthode de définition de permission d'accès à un dossier. Les possibilités sont vastes mais demandent plus d'implication qu'une simple série de clics dans une interface, puisqu'il revient au développeur de coder la totalité des actions à réaliser, ce qui peut être une opération particulièrement longue. De plus il ne gère nativement que des aspects de manipulation de données et de mapping, il n'est pas capable de créer les structures pour accueillir les données (content types ou blocks par exemple) à moins de créer soi-même de tels processus en début de migration.

Deux points importants sont à noter ici :

- Ce processus d'upgrade ne prend en charge que les éléments du core. Les modules contribs se doivent d'assurer leur propre mécanisme de migration s'ils définissent des éléments hors du core. C'est pourquoi le fichier UPGRADE.txt à la racine d'une installation Drupal indique qu'il est nécessaire de désactiver tous les modules contribs avant d'installer la version suivante, rendant le processus plutôt long et fastidieux.
- Le chemin de mise à jour d'un Drupal vers un autre n'est possible que d'une version majeure à la suivante (Drupal 5 vers 6) ou d'une version mineure une autre (Drupal 7.1 vers 7.5). Il n'est pas possible de passer d'une version 5 à une version 7 sans faire d'abord un passage obligé par une version 6.

L'approche Drupal 8

Quid des promesses faites par Drupal 8 ? Une question qui trouve réponse en un seul et unique module : Migrate. Ses nombreuses qualités, sa fiabilité et sa grande popularité l'ont désigné comme un candidat idéal dans la difficile tâche de prendre en charge la totalité des aspects de migration de Drupal 8. Il rejoint donc le core en digne successeur du système d'upgrade utilisé jusqu'ici.

Le core gagne ainsi ce qui lui manquait le plus dans les passages d'une version majeure à l'autre : la flexibilité. Pour le moment, il est décomposé en deux modules, le premier, Migrate, embarque l'ensemble des méthodes nécessaires au bon fonctionnement des migrations, la définition des sources, des destinations, le mapping, etc. Le deuxième, Migrate Drupal embarque une centaine de

Dans un article précédent nous avions déjà présenté plus en profondeur ce module tant apprécié de la communauté. Pour plus de détail, voir notre numéro 178 de Programmez.com. D'autres modules proposent des fonctionnalités similaires : Feeds, Node Export et bien d'autres.

Concernant les configurations, le module Features est la référence de la communauté et Drupal 8 lui a trouvé un digne successeur dans la fameuse Configuration Management Initiative dont nous ne parlerons pas ici mais qui dispose d'articles fournis dans la communauté.

classes de migration (chiffre annoncé mais non encore concrétisé, le module étant encore en cours de développement) qui ouvre le champ des possibles en autorisant cette fois bien plus qu'une simple transition depuis la version précédente. Trois types de migrations sont proposés :

- Drupal 6 vers Drupal 8 : l'arrivée de cette nouvelle version majeure implique un arrêt imminent du support de Drupal 6 et donc l'urgence d'une transition pour rester dans un cadre sécurisé. L'équipe de Migrate en a bien conscience et a donc commencé son travail par ce développement afin d'assurer la possibilité d'une migration Drupal 6 dès la release 8.0)
- Drupal 7 vers Drupal 8 : ces classes de migration constituent le principal développement en cours, annoncé pour la version 8.1.
- Drupal 8 vers Drupal 8 : Si un tel chemin de migration peut paraître curieux il est primordial car il propose une résolution à l'une des problématiques les plus anciennes de Drupal : le staging, ou sa capacité à décorréler un contenu de l'environnement sur lequel il se trouve pour facilement le migrer d'un environnement de préproduction à un environnement de production par exemple. Cet effort est largement simplifié par l'arrivée dans le core d'un autre module indispensable à ce type de manoeuvre : UUID (Universal Unique Identifier), permettant d'attribuer à chaque entité un identifiant unique fiable et portable entre plusieurs environnements.

En parallèle de ces classes préconstruites pour les besoins du core, les développeurs peuvent toujours créer leurs propres classes de migrations pour répondre à leurs besoins spécifiques, en s'aidant si besoin de celles embarquées par défaut.

Etant données les différences de structures, certains éléments importés ne respectent pas tout à fait leur état initial. Par exemple un bloc ne mémorise pas son placement au sein d'un thème car il n'est pas certain de retrouver des régions identiques et un thème identique à l'arrivée. De même les traductions ne fonctionnent plus de la même façon. Tous les éléments sont désormais traductibles depuis le core sans avoir à installer pléthores de modules. C'est pourquoi de multiples nodes représentant chacun une langue différente d'un même contenu seront migrés en un seul et unique node contenant toutes les traductions.

Migrate Drupal

Un dernier module vient compléter les modules Migrate du core. Celui-ci a pour l'instant un statut assez particulier : c'est un module contrib ayant vocation à intégrer le core d'ici la sortie de Drupal 8.1. Ce module se nomme Migrate Upgrade et met à disposition une interface graphique permettant de lancer les migrations. A terme ce module est amené à remplacer l'ancien système d'upgrade qui se faisait via update.php comme vu précédemment. Pour le moment il n'expose qu'une simple interface de commande à l'url /upgrade Fig.1.
Les versions précédentes de Migrate proposaient déjà une interface de même type permettant de sélectionner les classes à exécuter et les actions à réaliser (lancer la migration, faire un rollback, reset l'état des migrations en cours, etc.) S'il est certain qu'une telle interface fera certainement son retour, l'interface actuelle ne propose qu'une version extrêmement basique, très éloignée de son potentiel. Elle permet de renseigner les informations nécessaires à une migration depuis Drupal 6 : adresse du site source, accès à la base de donnée et répertoire des fichiers publics et privés (ces types de fichiers ne pouvaient être migrés que difficilement auparavant.) Un simple clic sur le bouton perform lance automatiquement l'intégralité des migrations existantes.
Les configurations sont importées en premier, ainsi Content types ou Blocs manquants sont créés. La méthode prepareRow() permet d'altérer les données comme c'était déjà le cas dans les versions précédentes de Migrate et est exécutée juste avant l'écriture des données. Le traitement dans ce cas est réalisé à l'aide d'un traitement batch. Ceux-ci bénéficient désormais d'une API beaucoup plus stable, il est plus rare de voir un tel processus planter, c'est un point important pour des migrations qui peuvent être très volumineuses Fig.2.

Drush et le Manifest file

Drush est toujours disponible sur migrate pour lancer des migrations. Pour ce faire il faut

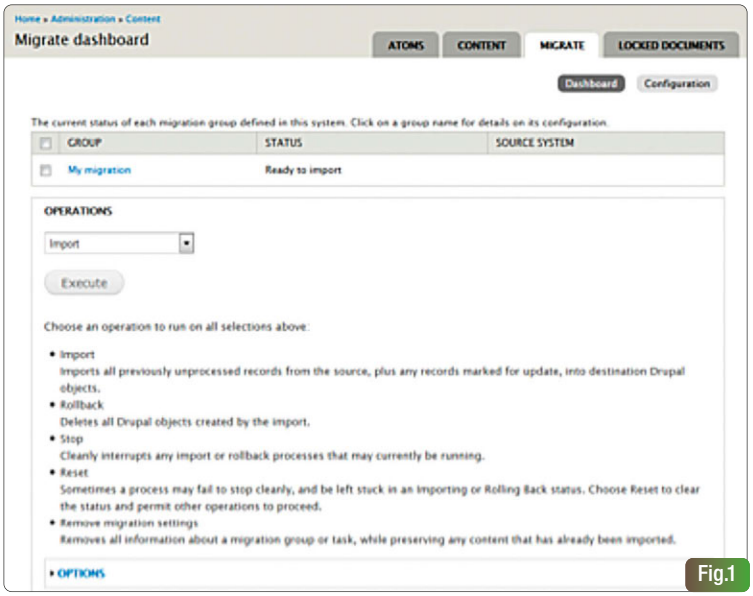


Fig.1

Interface de Migrate sur Drupal 7

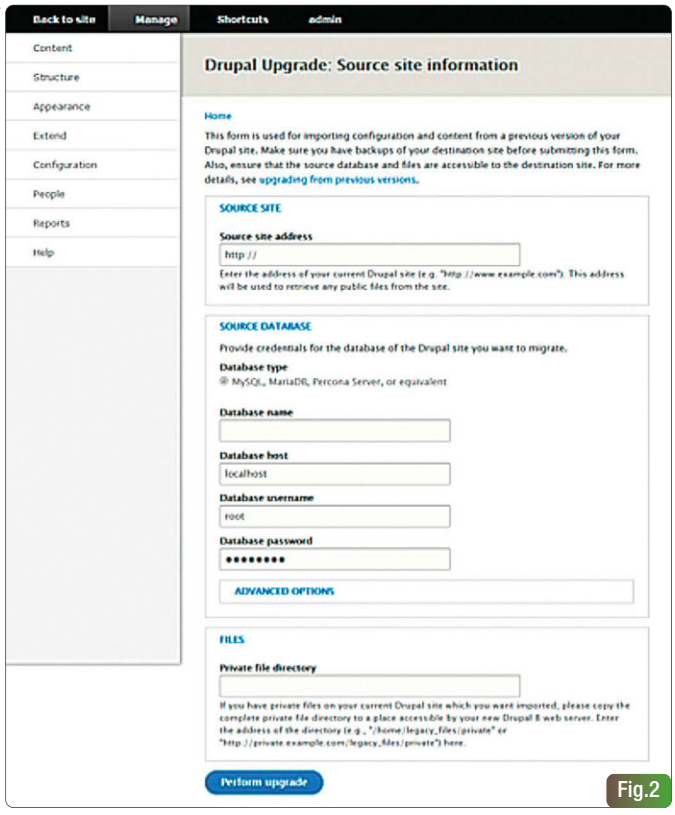


Fig.2

Nouvelle interface, ne permettant pour le moment de migrer que depuis Drupal 6.

passer par la création d'un fichier manifest. Ce fichier est un fichier yml qui permet de lister l'ensemble des migrations que l'on souhaite lancer avec quelques paramètres particuliers si nécessaire : Fig.3.

On peut, si on le souhaite, créer un module qui permettra d'exclure des contenus de façon sélective, de filtrer des données grâce à de nouveaux hooks. Il est également possible de réaliser des overrides des plugins de Migrate pour modifier complètement la façon dans les entités sont migrées grâce à la flexibilité de son puissant système de plugin.

Une fois tous ces aspects mis en place il suffit d'enregistrer ce fichier manifest.yml à la racine de Drupal et de lancer la commande drush suivante :

```
drush migrate-manifest manifest.yml --legacy-db-url
=mysql://<username>:<password>@<host>:<portno>
/<db_name>
```

Exemple de classe de migration Drupal 8

Le modèle de classe de migration est annoncé comme très proche de ce qui existe pour Drupal 7. Un module d'exemple est téléchargeable avec la sandbox Migrate Plus pour tester son comportement, il est d'ailleurs utilisé dans le cadre des développements en cours pour tester le bon fonctionnement des classes ajoutées et modifiées chaque jour. Si vous voulez comprendre comment fonctionne une migration Drupal 8, amusez-vous avec cette sandbox. Voici quelques extraits définissant un simple plugin source de migration de node : Fig.4.

Conclusion

Drupal 8 a choisi de bâtir le coeur de ses migrations sur les fondations solides d'un module qui a fait ses preuves. Migrate, s'il n'est pas encore totalement terminé, annonce des ambitions à la hauteur de la qualité à laquelle il nous a habitués sur les versions précédentes. L'API est d'ores est déjà disponible et des aides en ligne vous aideront à faire vos premiers pas avec ce module plein de promesses.

How-to Migrate in Drupal 8 :
https://www.drupal.org/node/2257723
API Migrate Drupal 8 :
https://www.drupal.org/node/2127611
Migrate Upgrade :
https://www.drupal.org/project/migrate_upgrade
Migrate Plus :
https://www.drupal.org/sandbox/mikeryan/migrate_plus

```php
// Définition du namespace et des librairies à utiliser.
namespace Drupal\migrate_example\Plugin\migrate\source;
use Drupal\migrate\Row;

// Nous étendons la classe permettant de se connecter à une base de donnée.
class BeerNode extends MigrateExampleSqlBase {

  // Définition d'une query initiale qui constituera la source de la migration
  public function query() {
    $query = $this->select('migrate_example_beer_node', 'b')
                  ->fields('b', array('bid', 'name', 'body', 'excerpt', 'aid',
                    'countries', 'image', 'image_alt', 'image_title',
                    'image_description'));
    $query->leftJoin('migrate_example_beer_topic_node', 'tb', 'b.bid = tb.bid');
    // Gives a single comma-separated list of related terms
    $query->groupBy('tb.bid');
    $query->addExpression('GROUP_CONCAT(tb.style)', 'terms');
    return $query;
  }

  // Définition du mapping des champs
  public function fields() {
    $fields = array(
      'bid' => $this->t('Beer ID'),
      'name' => $this->t('Name of beer'),
      'body' => $this->t('Full description of the beer'),
      'excerpt' => $this->t('Abstract for this beer'),
      'aid' => $this->t('Account ID of the author'),
      'countries' => $this->t('Countries of origin. Multiple values, delimited by pipe'),
      'image' => $this->t('Image path'),
      'image_alt' => $this->t('Image ALT'),
      'image_title' => $this->t('Image title'),
      'image_description' => $this->t('Image description'),
      'terms' => $this->t('Applicable styles'),
    );

    return $fields;
  }

  // Méthode définissant les clés primaires de l'import permettant, entre autres, le rollback
(non encore implémenté)
  public function getIds() {
    return array(
      'bid' => array(
        'type' => 'integer',
        'alias' => 'b',
      ),
    );
  }

  // Méthode permettant l'altération des items avant traitement.
  public function prepareRow(Row $row) {
    if ($value = $row->getSourceProperty('countries')) {
      $row->setSourceProperty('countries', explode('|', $value));
    }
    if ($value = $row->getSourceProperty('terms')) {
      $row->setSourceProperty('terms', explode(',', $value));
    }
  }
}
```

Fig.4

La migration sera lancée après un rigoureux calcul des dépendances. Si une dépendance est manquante ou un module non activé, Drush renverra une alerte indiquant le problème rencontré. Attention sur Drupal 8 il est nécessaire d'utiliser la version 7 de Drush

manifest.yml

```
- d6_filter_format
- d6_user
- d6_file:
  source:
    conf_path:''
  destination:
    source_base_path:''
    source_path_property:filepath
    destination_path_property:uri
    move:FALSE
    urlencode:FALSE
    rename:FALSE
```

Fig.3

ÉTÉ 2015

Soleil, plage et coding
Saison 2 : la revanche des boards !

© Programmez ! / par CommitStrip

Vous avez cru pouvoir passer un été loin du code et de la technologie ? Nous revenons à la charge, avec des renforts pour tout l'été. Ce n'est pas parce qu'il y a la plage, du soleil (on espère) ou la montagne, que Programmez ! allait vous laisser partir sans rien dire... Ce n'est pas dans nos habitudes.

Pour ce numéro d'été, nous vous avons trouvé de nouveaux projets à monter avec des Arduino et Raspberry Pi 2 ! On découvrira en avant-première une mini carte compatible Arduino ! Elle promet beaucoup. Sans oublier, un peu de science fiction, avec le robot Poppy, un robot à monter soi-même et à programmer !

Mais une angoissante question surgit : que faire des enfants ? Et là c'est le drame, l'écran de la mort, l'écran bleu que l'on aime à détester (troll à Windows). Vous sentez le kernel panic venir ?

Don't panic ! Nous avons la solution : Scratch ! Au lieu de jouer dans l'eau ou dans le sable, jouons avec la programmation ou à faire découvrir le code très simplement tout en s'amusant (ah, heureusement qu'il y a le lolcat)... On vous explique tout ! Il n'y a pas d'âge pour devenir un geek et un codeur.*

Maintenant vous pouvez boucler les sacs et les valises. Dernier conseil : n'oubliez pas le chargeur et les rallonges... Ça peut servir.

François Tonic
G.O. du code au camp d'été C++

* nous avons fait option poésie de la programmation

Les connecteurs des cartes

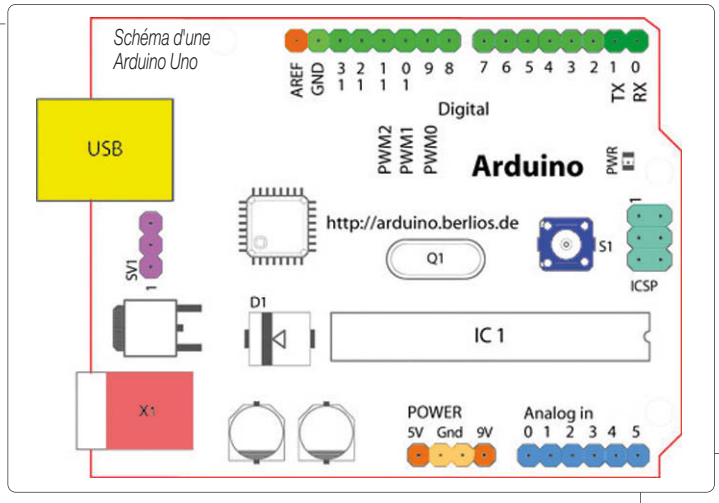
Une carte Arduino, Raspberry, etc. comporte différents éléments : interfaces, entrées / sorties, alimentation, ... Vous devez les connaître pour pouvoir réaliser vos prototypes.

La rédaction

Arduino UNO

Les cartes Arduino Uno possèdent différents blocs :

- Broches analogiques en entrée : A0, A1, A2, A3, A4, A5. Elles permettent de connecter des capteurs en entrée pour récupérer des données (température, fréquence cardiaque, capteur ultrason, etc.)
- Broches digitales : 2 à 13. Les broches 0 et 1 servent à recevoir et à transmettre des données.
- Bouton reset
- Connecteur d'alimentation
- Connecteur USB
- Broches ICSP
- Broches de tensions : GND, Vin, 3,3V, 5V.

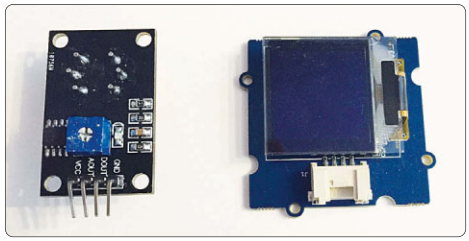


Pour les broches numériques et digitales, on parlera de GPIO (entrée/sortie pour un usage général en Français). Pour connecter un capteur ou une LED, vous devrez le brancher sur différentes broches pour l'alimenter et pouvoir l'utiliser.
L'Arduino (et d'autres cartes) supporte le I2C (Inter Integrated Circuit).
Il s'agit d'un protocole très pratique pour connecter des modules / capteurs, mais aussi pour communiquer entre plusieurs cartes (en mode Maître – Esclave).
I2C est utilisé par différents modules comme les écrans OLED / LCD. Sur les cartes Nano ou Mini, vous retrouvez les mêmes GPIO.

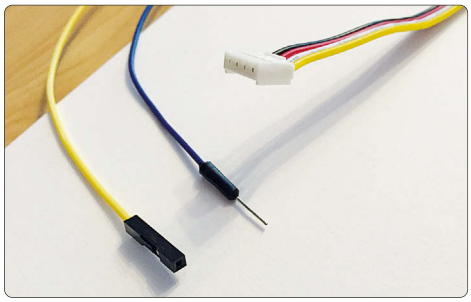
Pour en savoir plus :
http://www.arduino.cc/en/Reference/Board

Le GPIO du Raspberry Pi 2

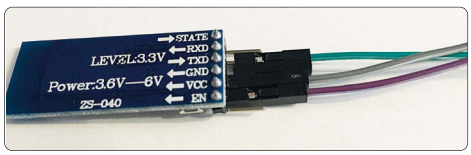
La Pi 2 est compatible avec le GPIO des premières Pi.
La Pi 2 possède deux rangées de broches (26 en tout) : 17 d'entre elles sont des GPIO, 5 GND, 2 pour une tension 5V et 2 pour du 3,3V. Contrairement aux Arduino, les broches ne sont pas identifiées directement sur le circuit imprimé. Ayez toujours sous la main le schéma d'organisation des broches !



Différents types de connecteurs sur les composants



Quelques exemples de câbles que vous allez utiliser



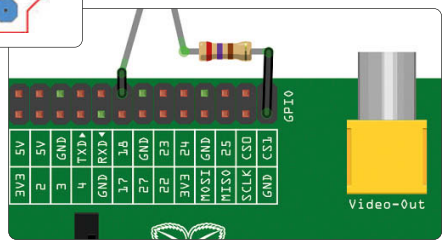
Les modules électroniques indiquent la fonction des bronches, il suffit de les connecter une à une à votre carte

Gadgeteer

Les cartes Gadgeteer proposent aussi des broches de connecteurs, mais sous forme de connecteurs complets comportant différentes broches. Par exemple, la carte FEZ Raptor possède 88 GPIO, le support du I2C, etc.

Avertissements

- Tout montage électronique comporte des risques. Respectez les mesures de sécurité.
- Utilisez des outils et des matériaux adaptés.
- Travaillez dans une pièce bien aérée.
- Programmez ! dégage toute responsabilité et ne saurait être tenu responsable de toute dégradation matérielle ou corporelle.



Raspberry Pi

Comment connecter les capteurs et modules à votre carte ?

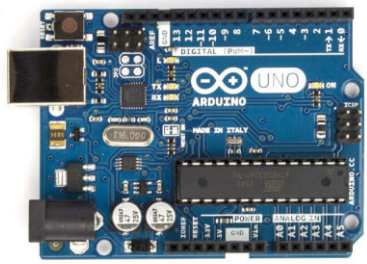
Plusieurs solutions sont possibles pour connecter les composants à votre carte et créer ainsi votre prototype :

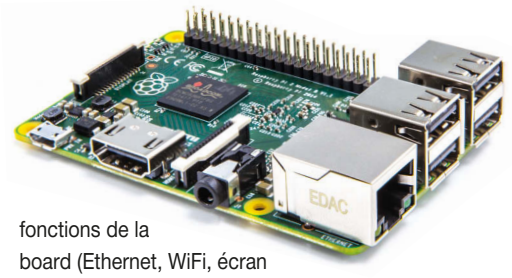
- Directement à la carte en branchant les composants (par des câbles adaptés) sur les différentes broches.
- Utiliser une planche à pain pour y poser les différents composants et les relier à la carte. Pratique et rapide, cette méthode permet de limiter le nombre de fils.
- Utiliser des boucliers (shield) de prototypage. Personnellement, j'utilise beaucoup cette méthode pour faciliter le montage. Il en existe de nombreux modèles. Sur l'Arduino et la Pi, nous utilisons la Grove Base Shield. Il en existe aussi pour l'Arduino Nano.

Les shields sont très répandus et permettent d'ajouter des fonctions à votre carte. Il en existe des centaines à des tarifs allant de quelques € à plusieurs centaines.

Les différentes cartes : choisir la bonne plateforme matérielle

Dans le monde Maker, deux plateformes matérielles se distinguent : Arduino et Raspberry Pi. Il en existe d'autres qui peuvent vous intéresser pour construire votre prototype, réaliser votre objet intelligent ou tout simplement, pour bidouiller !





François Tonic,
Maker à Programmez !

Le choix n'est pas toujours facile. On vous conseillera souvent entre Arduino et Raspberry Pi. Le type de projet pourra faire la différence : coût, extensibilité, consommation, puissance brute de la carte. Dans la communauté, on parlera souvent de board et non de carte, carte-mère ou plateforme.

Arduino : la star !

Ce projet un peu fou est né en Italie. Les premières cartes sont sorties en 2005 et depuis, Arduino révolutionne le monde de l'électronique et du « faire soi-même ». Cette board permet à tout le monde de se réapproprier l'électronique pour faire des projets sérieux ou tout simplement pour s'amuser. Arduino est une plateforme ludique. Une des forces d'Arduino est sa simplicité, le faible coût de la board et de très nombreux composants et extensions disponibles ! Arduino est open hardware, c'est-à-dire que le design électronique et les spécifications techniques sont totalement open source et disponibles. De nombreux clones Arduino sont disponibles soit auprès de magasins spécialisés, soit sur Internet. Une carte officielle Arduino Uno est vendue environ 23,90 €. Il est très facile de trouver des clones à partir de 10 – 12 €. En cherchant un peu, il est possible de trouver des boards à -7 € ! En général, ces clones fonctionnent bien. Certaines copies peuvent avoir des soucis de compatibilités avec le code et les modules Arduino. De notre côté, nous avons été très satisfaits des clones Funduino et SanSmart. Il existe plusieurs boards Arduino : Uno, Leonardo, Due, Yùn, Tre, Zero, Mega ADK / Mega 2560, Micro, Robot, Mini, Nano, Fio, Pro, etc. Les différences entre ces différentes boards se font sur le voltage, la vitesse du processeur, les entrées/sorties numériques, les entrées analogiques et le prix. Les 3 best-sellers Arduino :

- Arduino Uno : carte basique, la moins chère et la plus vendue. Très efficace et rapide à utiliser.
- Arduino Leonardo : successeur de la Uno. Le microcontrôleur est un peu plus puissant. Fonctionne comme la Uno
- Arduino Mega (2560) : plus grande, plus puissante et plus extensible que les Uno et Leonardo. La version DUE est une évolution de la Mega. Compatible avec les shields Uno / Leonardo.

On trouve très facilement des dizaines de shields (boucliers) d'extension pour étendre les fonctions de la board (Ethernet, WiFi, écran tactile, carte orientée santé, RFID, etc.) et des centaines de capteurs (température, présence, buzzer, pression, etc.). Pour des prix variant de -1 € à plusieurs centaines d'euros. À vous d'imaginer votre projet !

Raspberry Pi 2 : du costaud et de la mémoire

La Raspberry Pi 2 a fait la couverture de nombreux journaux et magazines. Très attendue, cette version 2 apporte un processeur beaucoup plus puissant et surtout une mémoire vive plus importante. Cette board est désormais un véritable ordinateur à 35 € ! La Pi 2 supporte officiellement Windows 10 alors que jusqu'à présent, seul Linux était accepté. Un usage assez courant de la Pi est de la transformer en média center pour son salon grâce aux excellents outils XBMC (ex. : Raspbmc). On dispose de plus d'interfaces de connexions par défaut que sur les boards Arduino (HDMI, audio, Ethernet, 4 ports USB...). Contrairement à Arduino, Raspberry Pi utilise un système d'exploitation complet et peut exécuter plusieurs applications en même temps, ce qui est très pratique pour des projets nécessitant plusieurs fonctions ou usages simultanés. À la différence d'Arduino, Raspberry Pi n'est pas open hardware, même si les schémas des différentes cartes sont désormais accessibles (exceptés pour la version 2) et actuellement, seule Rapsberry Pi fabrique des cartes Raspberry Pi. Comme pour Arduino, il existe de nombreuses extensions dédiées à Raspberry Pi : shields (shield pour connecter des capteurs, robotique, caméra, écran...), capteurs divers et variés. Des constructeurs proposent des kits

LA DOCUMENTATION : OÙ TROUVER CE QU'IL ME FAUT ?

Bon j'avoue, la documentation officielle est frustrante et pas toujours très complète mais il y a parfois de bonnes choses... Heureusement, la communauté, les constructeurs de capteurs et de composants et plusieurs sites makers nous sauvent notre projet et notre code plusieurs fois par jour, surtout, si comme moi, vous n'avez plus fait de soudure ou de montage électronique depuis au moins 25 ans !

- Instructables : le site incontournable du maker. Des montages par centaines. Les tutos sont très complets et clairs (montages, codes, conseils, etc.). Nous le consultons très régulièrement
- Element14 : une communauté très orientée ingénieur où vous pourrez trouver pas mal de conseils, de projets.
- Grove, adafruit... : certains constructeurs de composants proposent aussi de nombreux usages, montages, exemples de codes et les librairies indispensables.

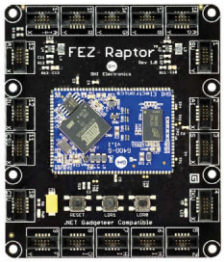
De nombreux livres sont disponibles. Quelques exemples :

- "**Le grand livre d'Arduino**" (Eyrolles) : livre idéal pour démarrer avec Arduino. L'auteur y présente les bases (fonctionnement de la carte, les différentes broches, etc.) et les fondamentaux d'électroniques (ce qui sert rapidement). 20 montages sont proposés avec de nombreux composants et capteurs différents.
- Et aussi : Les capteurs pour arduino et Raspberry Pi (Dunod), Projets créatives avec Arduino (Pearson), Arduino pour les Nuls (First), Raspberry Pi 2 (Eni), Raspberry Pi (Elektor)

complets (Sunfounder, Seeed Studio).

Gadgeteer : un peu de .Net

Moins connu, Gadgeteer est une board qui mérite que l'on s'y intéresse. Il s'agit d'une plateforme complète pour Maker basée sur un .Net Micro Framework (Microsoft). Le projet est open hardware. Il existe différents clones. Les premières cartes compatibles se vendent aux alentours de 25-26 € (ex. : Fez Cerberus), certaines cartes dépassent 80 ou 100 €. On retrouve le même principe de montage que sur les Arduino et les Pi : shields, modules et capteurs. Certaines cartes sont compatibles Arduino et Gadgeteer ! Côté développeur, on utilise les environnements Visual Studio grâce à l'extension Visual Micro. On dispose alors d'un environnement de programmation ultra complet. Le langage utilisé est le C#.



Comment choisir ?

C'est une question qui revient souvent quand je discute Maker et cartes. Il est impossible de donner une réponse unique. Tout va dépendre de ce que vous voulez faire. Raspberry Pi possède une puissance brute largement supérieure aux autres cartes, qui permet de le transformer en véritable PC ou en média center pour le salon; impossible de le faire avec une Arduino. D'autre part, les cartes Arduino ne peuvent exécuter qu'un code à la fois, ce qui est très pénalisant pour certains projets, sauf, à utiliser un Pi pour collecter les données et les traiter, ou une application (ou application Web) pour le faire.

Personnellement, j'apprécie la rapidité et la simplicité de mise en œuvre d'une Arduino (ou une gadgeteer) pour créer une petite station météo, une alarme, pour tester des idées et des capteurs comme j'ai pu le faire pour les TechDays 2015 et Maker Faire Paris 2015. Mais je sais que je serai limité dans les performances de la carte. Pour des projets plus « lourds » et exigeant des ressources (processeur, mémoire, sortie vidéo), une Pi 2 sera le meilleur choix, notamment avec l'utilisation d'un vrai système d'exploitation Linux ou Windows 10 IoT.

Un seul conseil : amusez-vous !

Petit tableau comparatif

	Arduino	Gadgeteer	Raspberry Pi 2
Microcontrôleur /processeur	AtMega (Atmel)	ARM (modèle varie selon la carte et le fabricant)	ARM
Open hardware/disponibilité des cartes compatibles (clones)	oui/oui	oui/oui	non (1)/non
Documentation officielle (qualité, disponibilité, documentation complète, etc.). Sur 5 *	**	**	**
Communauté (dynamisme, projets, documentations, etc.). Sur 5 *	*****	****	*****
Diversités des shields et composants (capteurs, écrans, etc.). Sur 5*	*****	****	****
Compatibilité avec d'autres cartes	Intel Galileo	Arduino (modèle NetDuino)	Raspberry Pi anciennes générations
Quelques usages possibles	prototypage rapide objets connectés projets persos bidouilles	prototypage rapide objets connectés projets persos bidouilles	prototypage rapide objets connectés projets persos bidouilles média center ordinateur
Modèle(s) de programmation (2)	C, C++	C#	C, C++ Python, Java
Système d'exploitation	-	.net micro framework	Linux Windows 10 IoT(3)
Prix de la carte	- 20 € (UNO officielle)	- 30 € (FEZ Cerberus)	35-40 €

(1) les schémas pour les versions antérieures à la Pi 2 sont disponibles.
(2) Nous mentionnons uniquement le ou les principaux langages
(3) Support officiel

LES MINI-BOARDS

Les cartes standards sont idéales pour faire du prototypage rapide d'un projet, d'un objet mais pour peaufiner et optimiser la taille, il faut utiliser d'autres catégories de cartes, en vrac : mini, nano, micro, etc. De notre côté, notamment pour la mini station médicale présentée à Maker Faire Paris, nous avions utilisé une Arduino Uno avant de commencer une migration du projet sur une Arduino Nano pour optimiser la place et la disposition des différents composants.

	Uno	Nano	Mini
Microcontrôleur	ATmega328	ATmega328 (ou 168)	ATmega328
Broches numériques	14 (6 PWM en sortie)	14 (6 PWM en sortie)	14 (6 PWM en sortie)
Broches analogiques (entrée)	6	8	8
Voltage (limites / en entrée)	6-20V	6-20V	7-9V
Voltage de fonctionnement	5V	5V	5V
Dimension (en cm)	6,8 x 5,3	4,5 x 1,8	3x1,8
Poids (en grammes)	25	5	-2

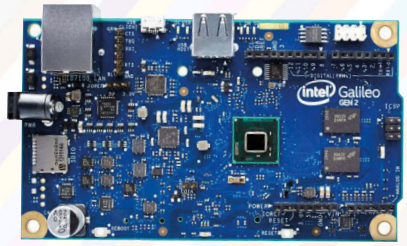
Les principales différences se feront dans les dimensions de la carte, le poids et le voltage autorisé. Pour reste, on peut migrer sans problème d'une Uno à une Nano (codes et composants). Par contre attention à bien vérifier que la puissance de la nano ou de la Mini suffit pour tout alimenter. Il vous faudra souvent passer un module USB externe pour connecter la microcarte. Les boucliers utilisables sur la Uno ne le sont pas sur une Nano ou Mini (pas directement du moins).

Sur les clones Arduino, vous aurez beaucoup de choix et les tarifs sont très variables, souvent de qualité, parfois, médiocres.

Vous pouvez opter pour une nano carte de type ESP8266 qui est un véritable microcontrôleur. Il existe de nombreux modèles. Les premiers prix sont très bas : env. 4 €. Le modèle ESP-07 possède plusieurs broches GPIO, GND et VCC. Pour la programmation, on utilisera le connecteur UART.

QUELQUES AUTRES SOLUTIONS

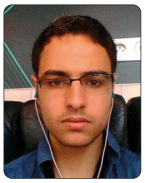
Netduino : carte open source utilise le .Net Micro Framework, comme sur les Gadgeteer, et assure une compatibilité avec les shields et modules Arduino ! Environ 40 € la carte.



Intel Galileo : Intel, célèbre pour ces processeurs, propose une carte bien conçue tout en assurant une compatibilité élevée avec les shields et modules Arduino. On utilise les outils de programmation de la Arduino. 70 – 80 € pour la carte (génération 2).

Impression 3D : Tout ce qu'il faut savoir

L'impression tridimensionnelle (3D) est un autre terme pour désigner la « fabrication additive » qui est un procédé de fabrication d'ajout de matière couche par couche, le tout contrôlé par un ordinateur. Utilisée initialement que pour le prototypage en raison de ses défauts, l'impression 3D est maintenant de plus en plus utilisée pour la fabrication de pièces fonctionnelles.



Chouguiat Aymen
Supinfo International University
aero@vf.vc

Impression 3D ? Oui, mais quel type ?

Il y a plusieurs procédés d'impression 3D existante, des méthodes qui vous permettent d'imprimer en métal ou en céramique en passant par l'impression qui utilise des produits en poudre il existe différentes manières de faire. Toutefois ces procédés sont inabordables et ne seront pas accessible pour tout le monde, dans cette optique on va aborder les deux types d'impressions qui sont disponibles pour le grand public, le « FDM » pour *Fuse Deposition Modeling,* littéralement modelage par dépôt de matière en fusion, et la « SLA » *Stéréolithographie,* qui consiste à solidifier une couche de plastique liquide grâce à une lumière UV Fig.1 et 2.

Impression 3D pour tout le monde

L'impression 3D ne date pas d'hier, même si son existence n'avait pas forcement la forme qu'on connaît actuellement elle était bien présente et remonte aux années 80. Depuis, la technologie a évolué et les possibilités de la dernière génération d'imprimantes 3D sont réalisées à travers le monde commercialement parlant, et pas seulement pour démontrer la faisabilité de la chose.

Par exemple, Airbus a fait appel à la société Stratasys pour l'impression 3D de 1000 pièces pour son A350 XWB jet, de même GE aviation (principal fournisseur mondial de réacteurs d'avion) après l'acquisition de la société Morris Technologies produit pour ses réacteurs de jet un capteur en 3D imprimé avec un alliage cobalt-chrome Fig.3.

Fig.3

Acquérir votre propre Imprimante 3D

Si vous cherchez à acheter votre propre imprimante 3D, votre choix sera conditionné par votre budget ainsi que le genre d'objets que vous souhaitez imprimer Fig.4 et 5.

La plupart des imprimantes relativement abordables (il vous en coûtera quelques milliers d'euros quand même !) sont tous des modèles FDM, qui impriment des filaments de plastique, les réchauffent et qui l'extraient hors d'une buse. Si vous êtes un bidouilleur dans l'âme ou si vous êtes financièrement limité, vous pouvez construire vous-même votre propre imprimante 3D, le projet « Reprap » est la solution qu'il vous faut, le projet visait au départ la création d'une imprimante 3D à faible coût qui pourrait se reproduire en imprimant une version du kit d'elle-même Fig.6.

Avec les imprimantes 3D prêtes à l'emploi, il y a plusieurs kits proposés avec des prix qui diffèrent selon le nombre de composants, ou

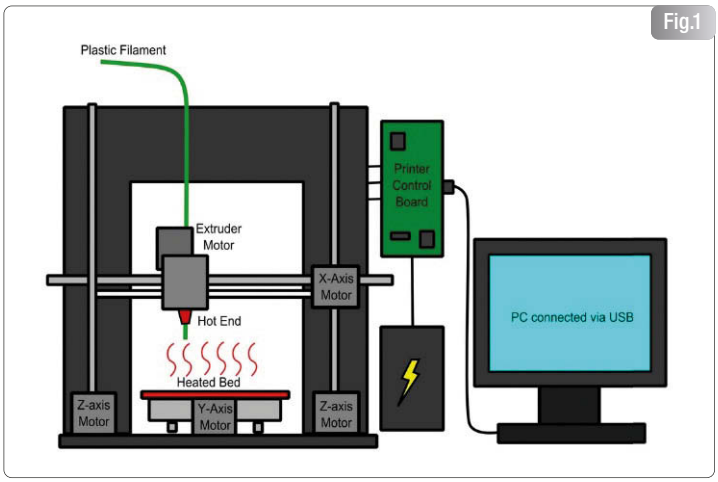


Fig.1

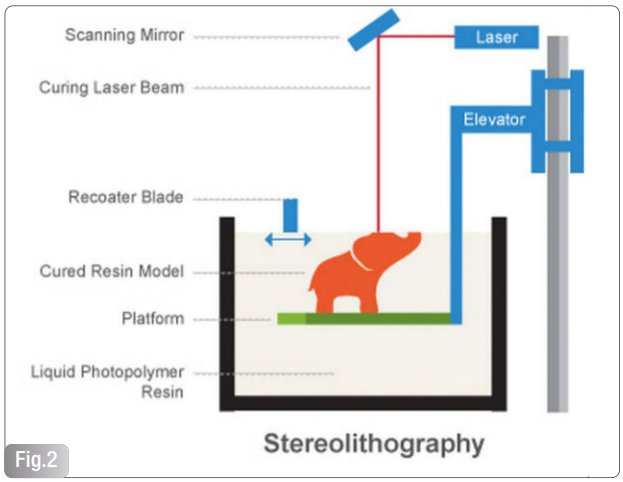


Fig.2

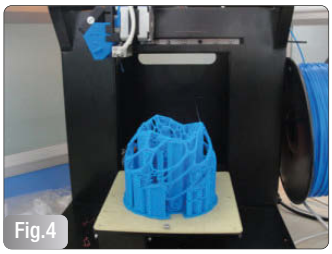


Fig.4

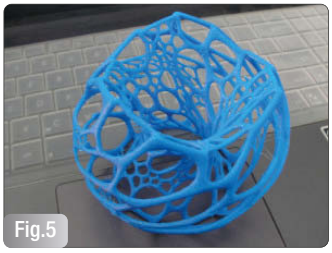


Fig.5

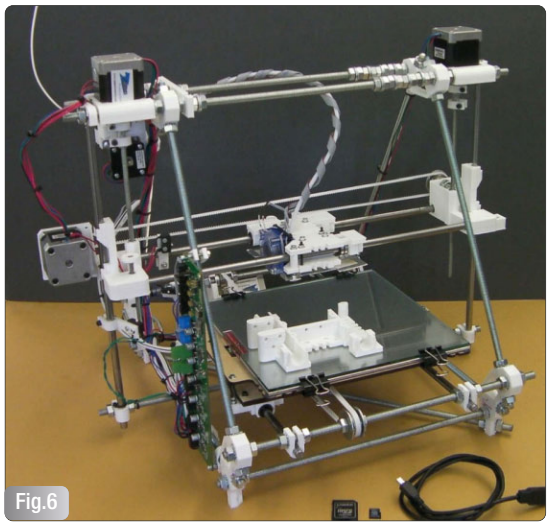


Fig.6

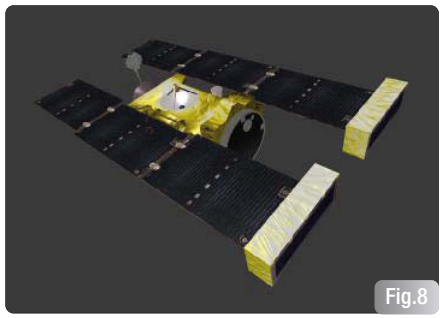


Fig.8

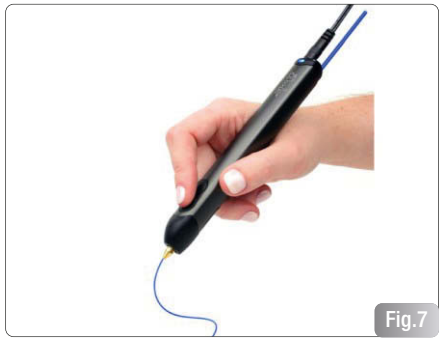


Fig.7

pour les budgets les plus modestes des pièces individuelles pour bâtir vous-même votre imprimante.
Il y a aussi des alternatives beaucoup moins chères à condition d'être habile avec vos mains, comme le 3Doodler qui est un stylo a impression 3D qui agit comme une imprimante, mais sans le mouvement qui va avec, le prix varie entre 100 et 150 euros Fig.7.

Avant d'acheter...

Comme pour toutes les nouvelles technologies, il est important de se poser les bonnes questions et de savoir ce qui est important pour vous avant d'acheter.
Tout d'abord la taille, le volume de construction indiquent les dimensions maximales des objets que vous souhaitez imprimer, plus grand signifie généralement plus cher. Après il y a les matériaux, certains matériaux sont plus coûteux, plus salissants que d'autres, le plastique biodégradable est bon marché, il est fabriqué à partir de fécule de maïs et ne sent pas trop mauvais quand il est chauffé.
D'un autre coté il y a le plastique ABS (l'acrylonitrile butadiène styrene) qui est un thermoplastique, c'est-à-dire qui devient souple et malléable quand il est chauffé, et qui se solidifie quand il refroidit ; en comparaison avec le plastique biodégradable, il est plus dur et peut produire des modèles plus fins et plus détaillés, l'inconvénient c'est qu'il coûte plus cher. Il nécessite aussi des températures plus élevées pour pouvoir le modeler et par conséquent aura besoin de plus de ventilation pour se débarrasser des fumées non désirées.
Pour des résultats encore plus poussés au niveau des détails, la résine utilisée par les imprimantes SLA est conseillée, mais là encore c'est plus cher.
Il existe aussi un certain nombre de nouveaux matériaux émergents qui disposent de propriétés assez intéressantes comme du caoutchouc fluorescent, ou un rendu métallique pour des modèles spécialisés.
Un autre point à considérer avant l'achat étant les buses d'extraction de votre imprimante, certaines imprimantes FDM vous permettent d'imprimer avec une multitude de couleurs, ou plusieurs types de matériaux dans une même impression, pour faire ça ils utilisent plusieurs extracteurs/buses d'extraction.

Que construire ?

Ça y est vous avez craqué et vous vous êtes acheté une imprimante 3D, et maintenant vous vous demandez que construire avec, il est peu probable que vous maîtrisiez déjà la modélisation 3D (sauf si c'est votre vocation !), mais ce n'est pas grave pour autant il y a plein d'endroits où vous pouvez trouver des choses à imprimer sur votre nouveau joujou.
La plupart des gens vont commencer avec des sites tels que Thingiverse (Communauté de partage de fichier de conception 3D), YouMagine, Cubify ou encore GrabCAD.
Même la Nasa a son propre catalogue de modèles imprimables en 3D (en utilisation libre) comme le Stardust Fig.8.

Concevez votre projet

Après avoir pratiqué un peu avec votre imprimante et dès que vous pouvez sortir une impression correcte, vous commencerez à concevoir des modèles qui vous seront propres ; vous vous lancerez dans le peaufinage de modèles déjà existants afin qu'ils répondent un peu mieux à vos besoins, il y a un grand nombre de logiciels de modélisation 3D gratuits, qui vous permettront de faire appel à votre créativité et qui vous cloueront accessoirement sur votre chaise pendant des heures.
À ce moment-là, vous aurez appris un peu plus quels sont les modèles d'imprimantes qui impriment bien et ceux qui ne le font pas pour prendre en compte et intégrer ce fait dans vos propres créations; par exemple, maintenir une épaisseur minimale d'une paroi, pouvoir creuser des objets solides quand c'est possible, prendre en compte les besoins en matériaux..., etc.
TinkerCAD d'Autodesk est un logiciel gratuit et conseillé pour débuter dans la modélisation 3D, il existe aussi des applications plus ludiques comme 123D Catch qui permet de transformer une série de photos en modèle 3D imprimable disponible pour Android, iOS, et PC Fig.9 et 10.

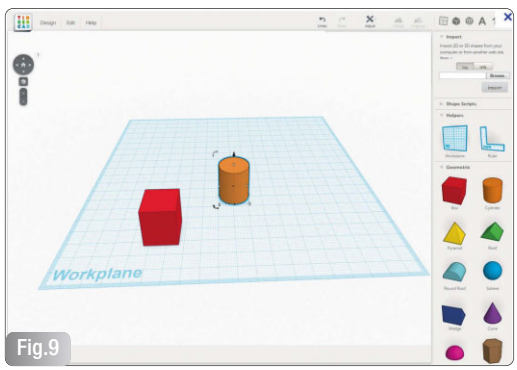


Fig.9


Fig.10

Un éclairage avec capteur de mouvement

Depuis quelques mois, les douilles et ampoules avec capteur de mouvement se multiplient et à tous les prix. Et si nous le faisions à la mode DIY (Faites-le vous-même) ? Amusons-nous un peu.

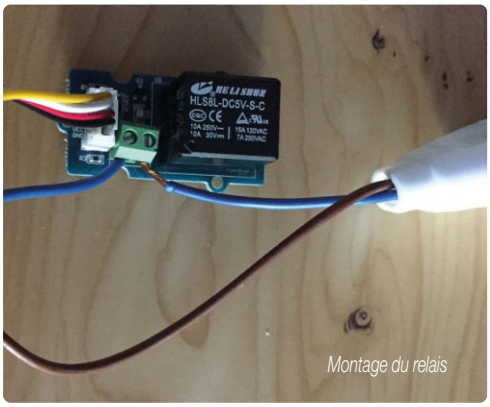
François Tonic
Programmez !

Avertissement : ce montage utilise le courant 220V du réseau électrique. La manipulation du 220V est dangereuse. Respectez rigoureusement les consignes de sécurité. Avant toute intervention sur une prise, une douille, mettez hors tension la zone de travail.

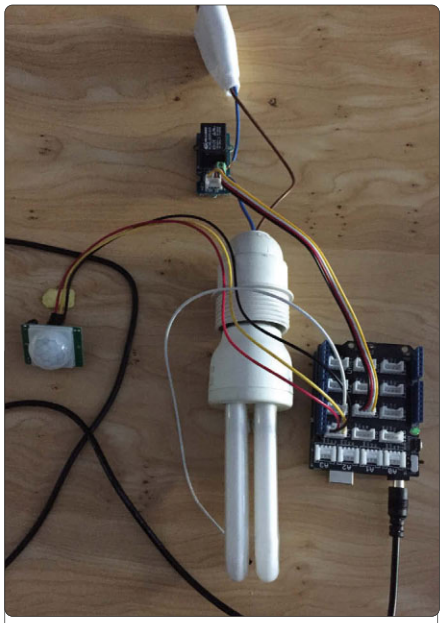
Le projet est très simple : dans le couloir d'entrée, je souhaite que le plafonnier s'allume automatiquement quand il détecte une présence. En absence de toute présence, le plafonnier s'éteint. Cela évite d'actionner manuellement les interrupteurs du couloir.
Code complet sur www.programmez.com

Montage et branchement

Le capteur de mouvement va détecter la présence d'une personne dans le couloir d'entrée. Cette présence va alors actionner l'ampoule du plafonnier qui va s'allumer. Si aucune présence n'est détectée, rien ne se passe. Dans la pratique, on démonte le plafonnier pour installer sur un des fils de la phase 220V le module relais (sur la borne verte). On coupe le fil pour installer le relais comme le montre l'illustration.
Le relais Grove est connecté à la carte Arduino (via le Grove Base Shield). En effet, c'est la carte qui va gérer. Le relais est connecté sur le connecteur D7.
Le capteur PIR est connecté sur le connecteur D4. Le branchement est très simple : VCC (5V), GND et la broche digitale (broche du milieu) qui est connectée à la broche 4. Il est possible de modifier la sensibilité du PIR directement sur celui-ci (2 réglettes sont présentées sur le PIR pour la sensibilité et le temps).
Quand un mouvement est capturé par le PIR, on ordonne au relais de changer d'état (LOW -> HIGH) permettant d'allumer l'ampoule. Quand le PIR ne détecte rien, le relais repasse en état LOW. Le code est très simple : www.programmez.com.



Montage du relais



Liste du matériel

- 1 carte Arduino Uno (mais vous pouvez aussi utiliser une Nano ou une Mini pour le montage final),
- 1 capteur de mouvement de type PIR,
- 1 relais 220V (ici nous utilisons le Grove Relay de Seeed Studio),
- Plusieurs câbles électrique et de connexion entre les capteurs et la carte,
- Pince, tournevis

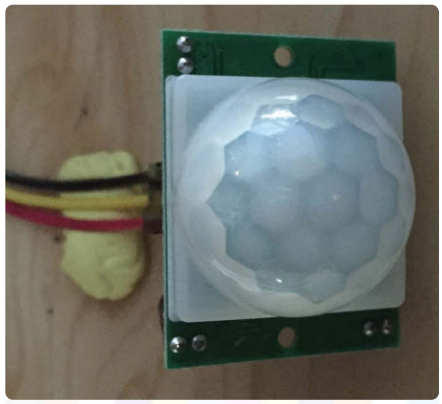
Le code

Il ne comporte aucune difficulté particulière : on déclare l'ensemble des capteurs et des variables puis on traite dans void loop l'état du PIR, et on définit une boucle pour activer ou non le relais selon la détection du PIR.
Pour la durée de l'éclairage, deux solutions :

- La lampe reste allumée tant que le PIR détecte une présence,
- Une temporisation est mise en place pour x secondes. Puis le PIR reprend le dessus. Il suffit d'utiliser : delay([durée]).

Nous avons opté pour la première solution dans notre montage.
Si tout le montage est correct et que le code est bon, il est temps de tester en situation réelle. Une led rouge sur le relais signale le bon fonctionnement de celui-ci, un clic sec se fait en entendre. C'est le relais qui se ferme ou s'ouvre. Enfin, il faut alimenter l'Arduino. Soit vous optez pour une batterie (9 – 12V) totalement autonome, soit vous branchez l'Arduino sur une prise 220V via un bloc d'alimentation dédié (notre option). Bien entendu, pour améliorer l'ergonomie du montage, mieux vaut passer par l'alimentation du plafonnier (dans ce cas, il faut se placer avant le relais) et utiliser impérativement un convertisseur de tension (220V -> 5V).

Avertissement : théoriquement le relais utilisé supporte 10A.
Nous vous déconseillons d'utiliser une ampoule trop forte (100W et +), utilisez plutôt une ampoule de – 60W.



LE CAPTEUR DE MOUVEMENT PIR

PIR signifie Pyroelectric InfraRed (sensor) ou capteur infrarouge pyroélectrique. Il s'agit d'un capteur passif qui va détecter les mouvements par la mesure de la chaleur dégagée. C'est ce rayonnement que le PIR peut détecter. Ces capteurs sont efficaces jusqu'à 6 à 7 mètres, avec un large angle de « vision ».

Prenez Poppy par la main

Nous vivons aujourd'hui dans une société qui adore le progrès technologique mais qui garde encore un œil suspicieux vis-à-vis des robots. Seraient-ce les blockbusters tels que I-robot et Terminator, ou serait-ce la crise "Les robots qui prennent nos emplois" qui font que cette peur a du mal à se transformer en quelque chose de positif.

Mike FAHRASMANE
titulaire d'un master en génie électrique et acteur du mouvement maker et DIY (Do-It-Yourself) passionné d'innovation. Il est le fondateur de GEEKOFYOU (http://www.geekofyou.fr) société de formation en impression 3D FDM (dépôt de fil fondu) et nouvelles technologies.

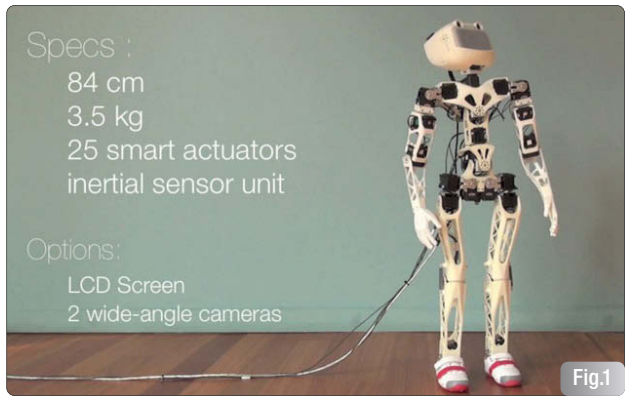


Fig.1

Pour mieux connaître et affronter ses peurs, quoi de mieux que d'apprendre à les apprivoiser. Dans notre cas, pour la robotique, nous parlerons du projet Poppy.
Poppy est un robot humanoïde open-source et open-hardware de 84cm de hauteur pour un poids de 3,5Kg; il dispose d'une structure imprimée en 3D. Tout cela autour d'une plateforme open-source pour la création, l'utilisation et le partage du robot interactif imprimé en 3D Fig.1.
Il a été développé par l'équipe Flowers de l'INRIA de Bordeaux, dirigée par Pierre-Yves Oudeyer, Directeur de recherche connu dans la communauté de la robotique pour ses travaux sur la robotique développementale.
Parmi les utilisateurs de cette plate-forme, nous pouvons retrouver des scientifiques, des laboratoires, des artistes, des universités, des makers et une liste encore plus longue tant le projet est fascinant et complet (informatique embarquée, électronique, impression 3D, mécanique).

Présentation de la partie commande du robot

Pour gérer toute cette structure, Poppy est equipé d'une carte Ordroid U3 alimentée en 5 Volts et permettant de gérer 2 cartes de puissances indépendantes (SMPS2Dynamixel) alimentées en 12 Volts. Elles permettent de contrôler les deux parties du robot à savoir le tronc et les jambes grâce aux servomoteurs Dynamixel Fig.2.
La carte Ordroid U3+ tourne sous Linux avec un processeur 4 cœurs de 1,7Ghz et 2Ghz de RAM. Il dispose de plusieurs périphériques comme un port HDMI, une prise jack 3,5mm, un port Ethernet 10/100, 3 ports USB 2,0 et accepte les slots card comme la microSD.
Les 25 servomoteurs Dynamixel sont branchés en série les uns à la suite des autres. Afin de pouvoir les contrôler individuellement, ils disposent chacun d'une mémoire programmable (EEPROM) ou l'on peut y entrer l'identifiant "id" du moteur. L'identifiant du moteur permettant de sélectionner celui-ci lors de la programmation du robot. Il faut savoir que nos moteurs communiquent grâce aux protocoles série asynchrone RS485. Afin de les faire communiquer avec la carte Ordroid, il va falloir installer les cartes USB/TTL "USB2AX" sur les ports USB de celle-ci. Et ensuite rajouter l'adaptateur SMPS2Dynamixel afin de convertir la partie commande en puissance en branchant celle-ci au 12 volts Fig.3.
Détails sur les cartes USB2AX et les moteurs dynamixels :
Monter l'adaptateur USB2AX :
http://www.xevelabs.com/doku.php?id=product:usb2ax:quickstart
Moteur dynamixel :
http://www.generationrobots.com/en/401088-dynamixel-mx-28r-actuator-robotis.html

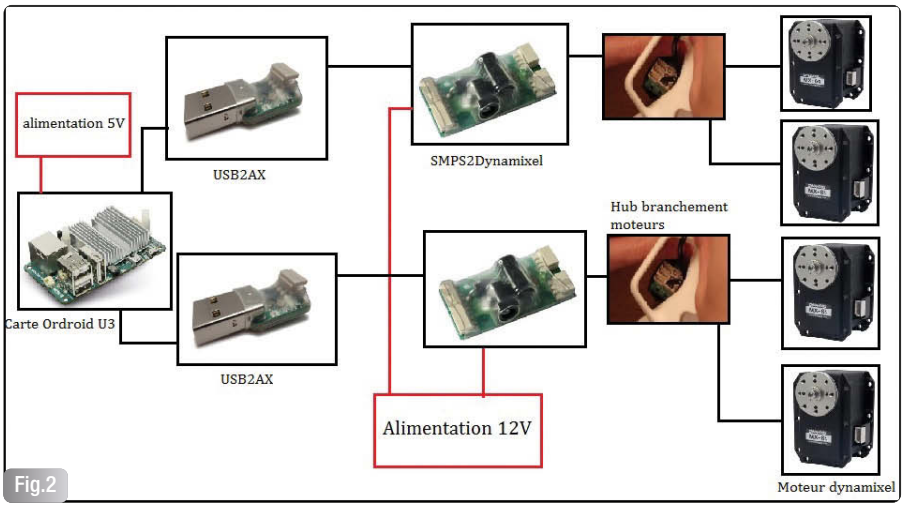
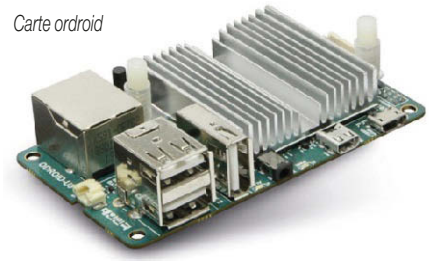


Fig.2



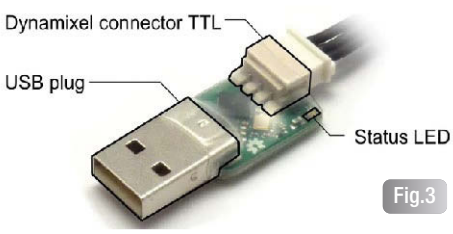


Fig.3

Partie mécanique : montage du bras et de l'avant-bras

Le kit Poppy est vendu avec toutes parties imprimées en 3D, et les parties électriques (moteurs, cables, moteurs...).
Afin de procéder au montage du robot, il nous faut réunir les outils suivants:

- Pince plate
- Cle allen
- Tournevis

- Lubrifiant
- Des vis (fournies)
- Des écrous (fournis)
- Un adaptateur SMPS2Dynamixel
- Un SMPS2Dynamixel (interface de puissance)

Montage de la main et de l'avant-bras

Récupérez la main droite de Poppy, regardez à l'intérieur de celle-ci, vous y verrez 4 placements pour y insérer des écrous. Placez un écrou avec vos doigts dans l'encoche prévue à cet effet et finissez de l'insérer plus profondément avec une pince. Faites de même pour les 3 autres écrous Fig.4. On peut maintenant prendre l'avant-bras et l'insérer dans la main. Ceci fait, il ne nous reste plus qu'à insérer les 4 vis, qu'on aura préalablement lubrifiées, dans les écrous et à les visser afin de solidariser les deux parties Fig.5.

Montage du bras

- 1 bras
- 2 moteurs Mx28T
- 16 écrous
- 16 vis
- 1 câble de 140 mm
- 2 pack HN07-N101
- 1 pack HN07-I101

Prenez un pack HN07-N101 (comprenant une rondelle, une roue d'engrenage et des vis dont on récupérera la plus grande pour cette manipulation) et un moteur Fig.6.

Insertion écrou main Poppy

Montage moteur

Avant de faire cette manœuvre, vous devrez faire attention à mettre la roue d'engrenage à la position zéro par rapport à la roue d'entraînement du moteur. Cette position est indiquée grâce à une entaille que l'on retrouve sur la roue d'entraînement et la roue d'engrenage Fig.7.

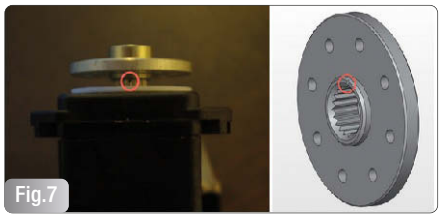
Entaille engrenage

Placez la rondelle, puis la roue d'engrenage comme indiqué, dans la roue d'entraînement. Puis lubrifiez la vis pour ensuite la visser dans la roue d'entraînement du moteur Fig.8.
Répétez maintenant la même opération avec le second moteur. Ensuite récupérer un pack HN07-I101 en prenant la roue d'engrenage, le roulement, la plus grande vis et la rondelle.
Insérez le roulement dans la roue d'engrenage, puis la rondelle à l'intérieur du roulement.
Lubrifier la vis et insérer le tout dans la roue d'entraînement dans l'un des moteurs déjà monté auparavant Fig.9.

Montage seconde roue engrenage

Connecter le câble au moteur.
Vous devrez avant de continuer, configurer le moteur grâce au logiciel **Herborist**. Il est installé directement avec Anaconda (partie programmation du robot). Il est possible de trouver un tutoriel sur ce logiciel et le matériel nécessaire via ce lien : Fig.10.
http://poppy-project.github.io/pypot/herborist.html

Chaque moteur possède son identification selon le membre auquel il est rattaché. Pour notre cas, il portera le numéro 54 pour le **r_elbow_y** et le numero 51 pour le **r_arm_z** Fig.11. Insérez le moteur dans le bras

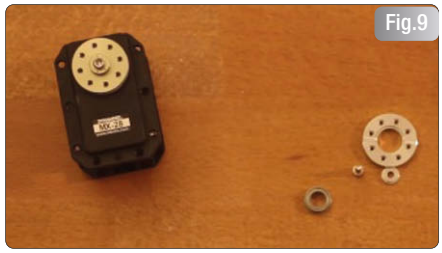


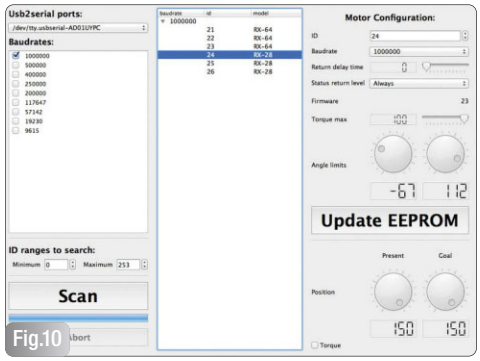
"r_elbow_y". Vous pourrez compter 10 correspondances de trous avec le bras. Retirez le moteur et insérez-y des écrous dans les encoches notées juste avant.
Replacez le moteur dans l'emplacement "r_elbow_y" du bras et solidarisez le tout avec les 10 vis Fig.12.

Faites passer le câble à l'intérieur du bras pour le connecter au second moteur.
Insérez l'autre moteur à l'autre extrémité du bras perpendiculairement à celui-ci. Vous pourrez compter 8 correspondances de trous avec le bras (4 pour le dessus et 4 pour le dessous). Retirez le moteur et insérez-y des écrous dans les encoches notées juste avant.
Replacez le moteur dans l'emplacement "r_arm_z" du bras et solidariser le tout avec les 10 vis Fig.13.

Assemblage du bras et de l'avant-bras

L'avant-bras et le bras étant assemblés avec les moteurs et les parties imprimées en 3D, il reste à assembler ces deux parties Fig.14.
Faites entrer l'emplacement au niveau du coude de l'avant-bras dans le moteur.
Et pour finir, vissez le tout de chaque coté du





bras Fig.15.

Guide de démarrage rapide du robot

Maintenant que vous avez imprimé les différents membres de Poppy grâce aux fichiers 3D que vous aurez téléchargés sur le site **poppy-project.org** et que vous vous êtes procuré les différentes cartes électroniques, servomoteurs et câbles, il reste maintenant à s'initier à la programmation de celui-ci. Pour cela, il existe différentes méthodes qui sont :

- La programmation par le logiciel **SNAP** qui est une interface de programmation graphique,
- La programmation via l'**interface Poppy start-up** qui est faite via un navigateur Web Fig.16

Notons que la programmation via SNAP et l'interface Poppy start-up est rendue possible grâce à une librairie écrite en Python du nom de Pypot et du serveur embarqué "Ipython" sur la carte électronique.

Pour ce poste, nous nous intéresserons à la programmation via l'interface Web.

Pour débuter, il faudra commencer par alimenter le robot avec une alimentation de 12V, le positionner en position de démarrage comme sur la photo ci-dessous (sachant que les servomoteurs peuvent tourner sur eux-mêmes à 360°) et entraîner une casse.

Il faudra ensuite :

- Attendre 2 minutes 30 pour qu'il s'initialise et émette un son,
- Vous connecter à votre ordinateur et au reseau wifi "poppy-network"
- Cliquer ensuite sur l'icone Poppy interface qui ouvrira une page Web http://10.0.0.1:6666/ sous votre navigateur Web Firefox pour vous afficher l'interface Poppy start-up que l'on peut voir juste après.

Via cette interface, il est possible de le programmer de deux façons différentes

- Le bouton "start interface" qui permet de lancer des mouvements préprogrammé via l'interface Web ou de programmer nos propres mouvements : bouger la tête, les bras, les jambes, le torse, parler, etc.
- Le second bouton "notebook" permettant la programmation de Poppy via Ipython.

Ipython étant un environnement interactif pour écrire et faire tourner du code Python. Pour notre cas, il est aidé de la librairie Pypot qui a été développée par l'équipe Flowers de l'Inria permettant une approche plus facile pour la prise en main et le contrôle des différents servomoteurs. Cette pratique ne sera pas traitée ici.

Programmation via start interface

En cliquant sur start interface, on s'aperçoit qu'elle est séparée en deux grandes parties qui sont "Basic behaviour" et "move recorder" Fig.17.

- Basic behaviour est une sélection de mouvements préprogrammés assez rapide à faire reproduire par notre hôte en cliquant sur l'action à exécuter (s'asseoir, stand up...).
- Move recorder permet d'enregistrer des mouvements que nous faisons exécuter au robot en le bougeant manuellement et de les lui faire reproduire par la suite.
- La simulation grâce au logiciel **VREP.** Avec ce simulateur que l'on peut télécharger, on peut avoir une idée de comment évoluera notre robot en fonction du programme qu'on aura chargé. Tout cela de façon virtuelle et en live sur notre écran grâce à un modèle identique exécutant les mêmes mouvements que le vrai robot. Tout cela sans posséder de robot Fig.18.

Vous commencerez par télécharger Anaconda

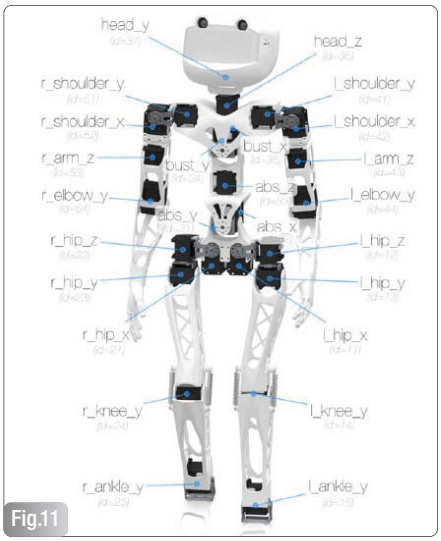


Fig.11


Fig.13

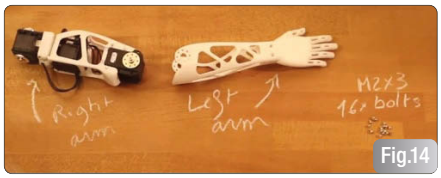


Fig.14


Fig.15

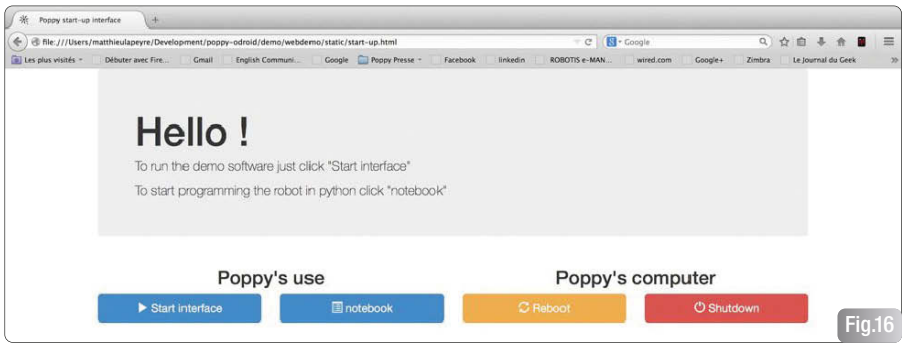


Fig.16

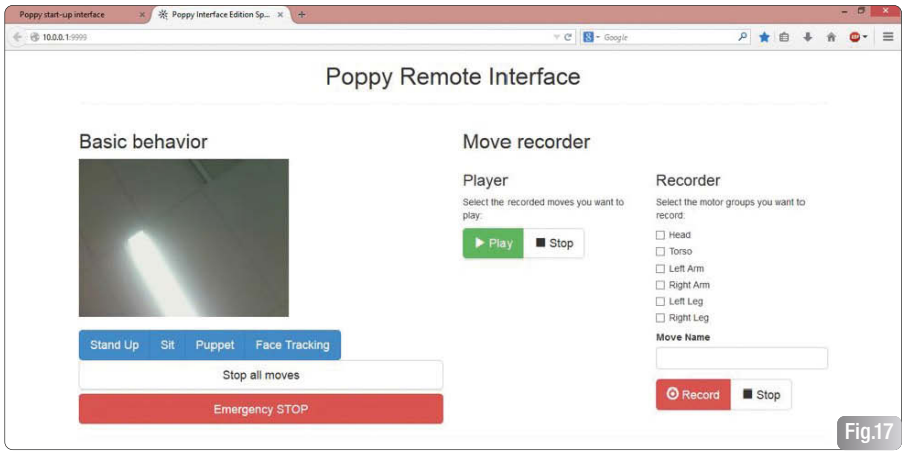


Fig.17

et à l'installer avec les librairies **Pypot** (celle permettant le contrôle des moteurs dynamixels) et la librairie **poppy_humanoid** (pour la simulation). L'installation de ceux-ci étant très bien documentée sur le lien : poppy-project.org
Ceux étant un peu plus à l'aise peuvent installer directement Python 2.7, ou plus, et installer les librairies.
Téléchargez le logiciel de simulation **VREP** sur : coppeliarobotics.com/downloads
La librairie Python s'installera par défaut dans C:\Python27 pour le système d'exploitation Windows.
Ouvrez l'invité de commande Windows et tapez "C:\Python27\Scripts\ipython.exe notebook Fig.19. Cela ouvrira un navigateur pour exécuter les notebooks. Telechargez les notebooks en cliquant sur l'icone comme indiqué dans l'image, puis placez les dans le répertoire C:\Python27\Poppy_notebooks Fig.20.

Retournez dans l'invité de commande et placez vous dans C:\Python27> . Maintenant que nous sommes dans le répertoire Python27, tapez la commande :

```
C:\Python27> C:\Python27\Scripts\ipython.exe notebook
```

Cliquez sur Poppy_notebook dans la liste et ensuite sur "Controlling a poppy humanoid in V-REP using pypot. La page des notebooks apparaît : Lancez les librairies "from pypot.vrep import from_vrep" et "from poppy.creatures import PoppyHumanoid" en cliquant dessus et en appuyant sur play Fig.21.
Instanciez une primitive de Poppy pour la simulation avec VREP :

```
poppy = PoppyHumanoid(simulateur = vrep) .
```

Ouvrez votre simulateur VREP et vous verrez apparaître le robot poppy dans l'interface.
Pour commencer, on lancera un notebook assez simple qui fera tourner la tête de poppy de 90° : Fig.22.

```
poppy.head_z.goal.position= -90 puis on clique sur play.
```

Réduisez la fenêtre du simulateur et du navigateur de façon à voir les mouvements exécutés par le robot quand nous exécutons les notebooks Fig.23. Beaucoup de notebooks sont disponibles dans tous ceux que nous avons téléchargés un peu plus tôt. Vous pourrez faire bouger différents membres de Poppy, et même les associer afin d'avoir des mouvements plus complexes.

Conclusion

Tous cela entre vos mains vous permettra de débuter tout en douceur avec ce nouveau robot. Et pourquoi ne pas vous joindre à la communauté poppy sur le forum poppy-project.org et apporter votre contribution (idée, modification, amélioration, etc).
Si vous ne possédez pas encore votre robot Poppy et souhaitez l'acquérir pour vos projets personnels ou collectifs, vous trouverez le kit complet sur le site :
http://www.generationrobots.com/fr/.
Ces kits pouvant être commandés avec ou sans les parties imprimées en 3D.
Et pour finir, un grand merci à l'INRIA, l'équipe Flowers et à Pierre-Yves Oudeyer pour la belle création mise à notre disposition ainsi que pour les informations disponibles sur leur forum.
A vous de jouer pour un futur plein d'innovation.

Sources :

https://www.poppy-project.org/
http://www.generationrobots.com/fr/
https://www.youtube.com/watch?v=SUlM_mE3plc
http://www.generationrobots.com/en/401088-dynamixel-mx-28r-actuator-robotis.html
http://www.xevelabs.com/doku.php?id=product:usb2ax:quickstart

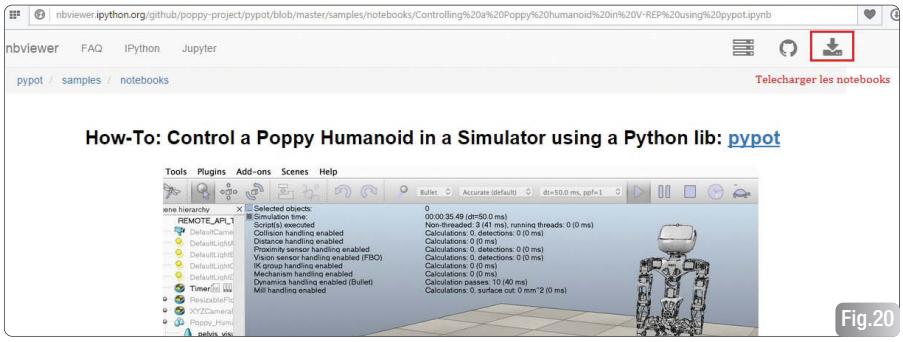


Fig.20

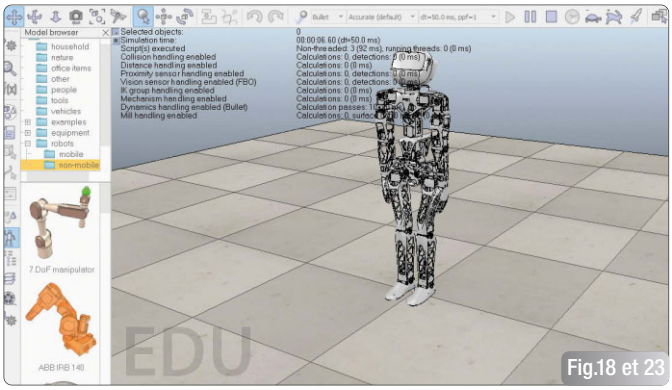


Fig.18 et 23

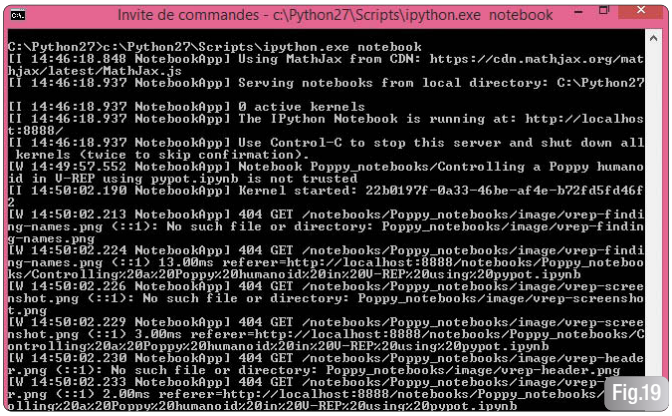


Fig.19

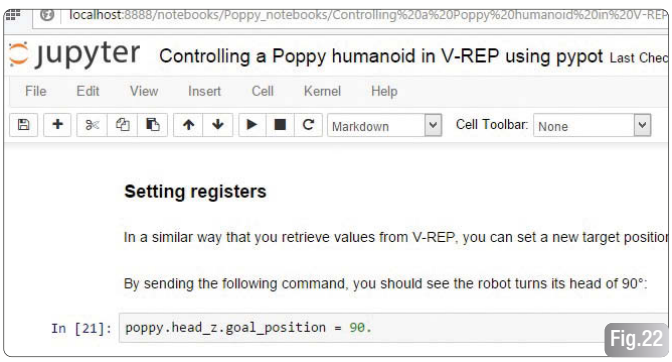


Fig.22

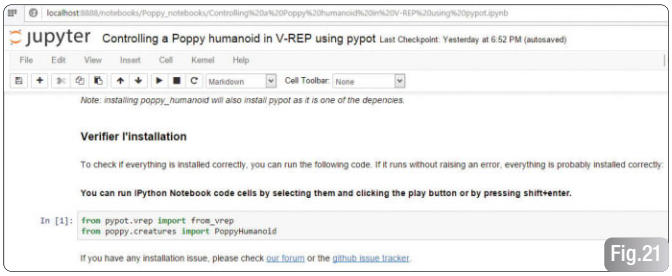


Fig.21

Créer son propre portail avec WEBRTC

Que feriez-vous d'un Portal Gun si vous en aviez un ? Homer Simpson, lui, a parfaitement compris son utilité et peut, grâce à son portail personnel, attraper l'une des bières de son frigo sans bouger de son sofa ! Le rêve de toute une vie devient (presque) réalité ! A défaut de pouvoir se servir directement dans votre frigo depuis votre canapé, vous pourrez le surveiller de près.
Pour « toucher avec les yeux » les bières qui sont au frais, nous allons réaliser ensemble des portails similaires à ceux des célèbres jeux « Portal », tout en se basant uniquement sur des technologies du Web ! Un bout de votre rêve se concrétise...

Jean-François Garreau (jfgarreau@sqli.com)
Responsable IOT & Senior Innovation Developper chez SQLI | Organisateur GDG Nantes
http://jef.binomed.fr

Tout le code source est disponible ici : https://github.com/GDG-Nantes/portal-devfest-2013

LE PROJET PORTAL WEBRTC

Voici le rendu de ce que nous allons réaliser : le portail bleu voit ce qui se passe dans le portail orange et inversement Fig.1.
Avant d'attaquer le code, regardons un peu de quoi nous avons besoin :

- Nous devons être capables de voir ce qui se passe de l'autre côté du portail,
- Un "mur de flammes" entoure notre image,
- Nous voulons nous téléporter de l'autre côté du portail.

Maintenant que notre « wishlist » est prête, attelons-nous à répondre aux différents besoins grâce à des technologies du Web :

- **WebRTC** : il s'agit d'une technologie de visio (mais pas que); c'est donc idéal pour voir ce qui se passe de l'autre côté du portail,
- Les **canvas** nous permettront de jouer efficacement pour simuler de façon performante notre "mur de flammes",
- La **téléportation API**.... euh non rien ne permet de répondre à ce besoin...

Ce projet ne marchera que sous Chrome ou Firefox

ARCHITECTURE DU PROJET Fig.2

- Chaque ordinateur se trouve sur le réseau et se connecte à un serveur Web uniquement pour afficher le contenu de la page. Le serveur est un serveur NodeJS,
- Ce serveur expose aussi une WebSocket dont le rôle sera expliqué plus tard dans l'article,
- L'échange des données vidéo se fait en "direct" entre les 2 ordinateurs via la technologie WebRTC .

WEBRTC WHAT ?

WebRTC (pour Real Time Communication) est l'une des technologies les plus importantes du projet. Grâce à cette API, on peut faire plusieurs choses :

- Obtenir l'audio et la vidéo,
- Etablir une connexion entre 2 hôtes,
- Communiquer de la vidéo et de l'audio,
- Communiquer d'autres types de données,

L'une des forces du WebRTC réside dans le fait que les données s'échangent directement entre les 2 ordinateurs, et que ces dernières ne passent pas par un serveur ! Pour réussir cet exploit, la technologie

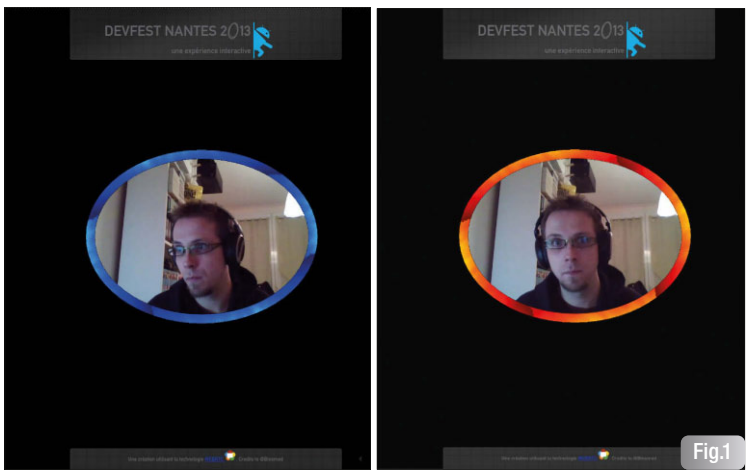


Fig.1

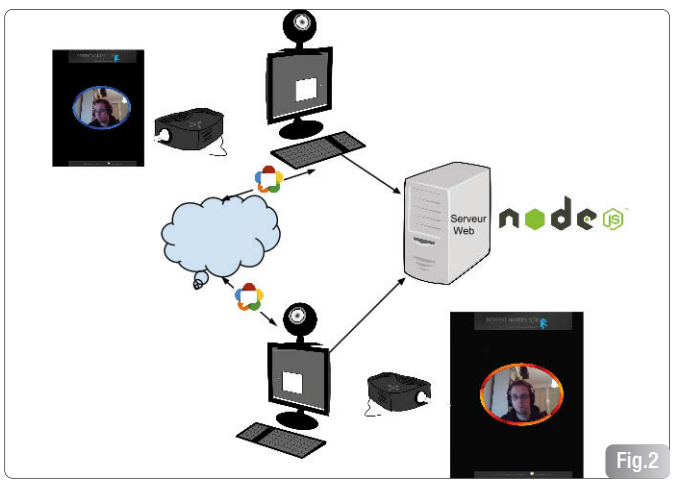


Fig.2

WebRTC repose sur 3 APIS Web :

- **getUserMedia** : cette API permet de récupérer les flux vidéo et audio d'un ordinateur,
- **RTCPeerConnection** : cette API permet de faire communiquer des données entre 2 hôtes en tenant compte de tout un ensemble de contraintes telles que l'adresse IP d'une machine, ses codecs, sa connectivité, ...
- **RTCDataChannel** : cette API permet de faire transiter sur une RTCPeerConnection des données textuelles ou binaires.

Pour notre projet, nous allons uniquement utiliser les API getUserMedia et RTCPeerConnection.

GetUserMedia

Il s'agit d'une API qui permet de récupérer un ensemble de streams de médias synchronisés. Chaque stream peut être vidéo / audio.

```
var constraints = {video: true};

function successCallback(stream) {
  var video = document.querySelector("video");
  video.src = window.URL.createObjectURL(stream);
}
function errorCallback(error) {
  console.log("navigator.getUserMedia error: ",
  error);
}
navigator.getUserMedia(constraints,
    successCallback,
    errorCallback);
```

Dans l'exemple ci-dessus, nous ne récupérons que la vidéo, et nous injectons le résultat de l'appel de getUserMedia dans une balise vidéo.

RTCPeerConnection

La RTCPeerConnection permet de gérer le transport des données. Pour initialiser une RTCPeerConnection, on répond au principe de l'offre et de la demande. Il y a d'une part, une notion d'offre et de demande pour communiquer, mais aussi une notion de chemin à emprunter ! Ces 2 notions s'appellent le "Signaling". Le Signaling a pour objectif de répondre à ces questions :

- Quels types de média et formats je supporte ?
- Que puis-je envoyer ?
- Quel est mon type d'infrastructure réseau ?

Pour réaliser cette étape, il suffit juste de trouver un moyen de passer ces informations à l'hôte distant. Une des technologies préconisées pour faire le signaling est "les WebSockets". C'est donc ici qu'interviendra notre serveur de websockets Voici comment se déroule le signaling :

Gestion de l'offre

- Alice appelle la méthode createOffer(),
- Dans le callback, Alice appelle setLocalDescription(),
- Alice sérialise l'offre et l'envoie à Eve,
- Eve appelle la méthode setRemoteDescription() avec l'offre,
- Eve appelle la méthode createAnswer(),
- Eve appelle la méthode setLocalDescription() avec la réponse envoyée à Alice,
- Alice reçoit la réponse et appelle setRemoteDescription().

Gestion du chemin Ice Candidate (ICE pour Interactive Connectivity Establishement)

- Alice & Eve ont leur RTCPeerConnection,
- En cas de succès de chaque côté les IceCanditates sont envoyées,
- Alice sérialise ses IceCandidates et les envoie à Eve,
- Eve reçoit les IceCandidates d'Alice et appelle addIceCandidate(),
- Eve sérialise ses IceCandidates et les envoie à Alice,
- Alice reçoit les IceCandidates d'Eve et appelle addIceCandidate(),
- Les 2 savent comment communiquer.

RETOUR AU PROJET PORTAL

Après cette rapide introduction sur la technologie WebRTC, revenons à nos moutons, ou plutôt portails, et à la manière de les créer.
Comme tout bon projet, je me suis inspiré de ce que je trouvais sur le Net afin de gagner du temps. Ainsi, plutôt que de vous noyer sous des montagnes de codes compréhensibles et/ou incompréhensibles, je vous donnerai les deltas que j'ai effectués et pourquoi je les ai faits.

Etape 1 : cloner les projets références

La base du projet WebRTC repose sur le codelab initialisé par Sam Dutton : ingénieur chez Google et travaillant sur l'implémentation de WebRTC dans Chrome : https://bitbucket.org/webrtc/codelab.
De façon plus précise notre point de départ sera le step7 de ce codelab.
Pour s'immerger d'autant plus dans l'univers « Portal », la base graphique des flammes repose sur le projet de Chris Longo : https://github.com/chrislongo/html5-canvas-demo
Nous allons donc commencer par cloner les 2 projets afin de récupérer une base de code propre et fonctionnelle que nous allons nettoyer petit à petit pour coller avec notre besoin.

Etape 2 : création du squelette de l'application

L'application est structurée comme suit :

- assets/ : fichiers externes
  - fonts/ : les fonts spéciales utilisées pour le projet
  - images/ : les ressources graphiques utilisées pour le projet
- css/ : le style de notre page
- js/ : les fichiers Javascript utilisés par le projet
- package.json : fichier des dépendances node utilisées pour le serveur node
- server.js : le serveur nodeJS

index.html : notre application
Téléchargement des ressources annexes :

- Télécharger la font Portal : http://fontmeme.com/freefonts/34868/portal.font
- Les images utilisées dans le projet sont disponibles ici :
  - Image du BugDroid Portal : https://goo.gl/vT6svL
  - Image du footer : https://goo.gl/J6CknB
  - Image du header : https://goo.gl/AQnloo -
  - Image du logo WebRTC : https://goo.gl/XeimoA

Etape 3 : écriture du serveur

Comme expliqué précédemment, nous allons baser notre travail sur le step7 du codelab.

./package.json

```
{
  "author": "jefBinomed",
  "name": "protal-devfest-2013",
  "dependencies": {
    "socket.io": "~0.9.14",
```

```
    "node-static" : "~0.6.9"
  }
}
```

./server.js

Nous reprenons le serveur tel qu'il est dans le codelab.
Ce serveur fait donc 2 choses :

- Dans un premier temps, on va définir un serveur http pour servir notre contenu HTML,
- On crée un serveur de webSockets afin d'assurer la partie "Signaling". Un message transféré au serveur sera automatiquement partagé avec l'autre client.

Il ne nous reste plus qu'à récupérer les modules node avec l'instruction :

```
npm install
```

De cette manière, les dépendances nodes seront téléchargées dans notre projet.

Etape 4 : écriture du projet WebRTC

Nous allons poser le style graphique de notre application : voir code complet.

./js/lib/adapter.js

Ce fichier doit être copié tel quel depuis le codelab car il s'agit de la classe Polyfill qui permet d'uniformiser l'API WebRTC entre Chrome & Firefox.

./js/canvasFire.js

Initialisez ce fichier à vide afin d'avoir l'import depuis le fichier html qui fonctionne.

./js/app.js

Nous allons partir du fichier issu du step7 : ./js/main.js. Copiez l'intégralité du fichier et nous allons retirer ce qui ne nous intéresse pas.

LocalVideo

Dans notre projet, nous ne sommes pas intéressés pour afficher le retour de notre webCam à l'écran, nous allons donc supprimer toutes les références à cet élément.

```
var localVideo = document.querySelector('#localVideo');
```

et

```
attachMediaStream(localVideo, stream);
```

dans la fonction handleUserMedia(stream).

DataChannel

De la même façon, tout ce qui concerne le DataChannel ne nous sert pas au début du projet :
https://gist.github.com/jefBinomed/ebcd92cd595efe19ff33

Ensuite :
https://gist.github.com/jefBinomed/f9d021477fbc8e313501

Il faut supprimer également tout ce qui suit et qui se situe au niveau de la fonction createPeerConnection.
https://gist.github.com/jefBinomed/f0f9697b022627b73579

Et enfin, pour finir :

```
var constraints = {'optional': [], 'mandatory': {'MozDontOfferDataChannel': true}};
```

de la fonction doCall() est à remplacer par :

```
var constraints = {'optional': [], 'mandatory': {}};
```

Tester

Nous pouvons à présent tester notre application pour vérifier que la vidéo passe bien à travers l'API WebRTC. Pour ce faire, il suffit simplement de lancer notre serveur à l'aide de la commande :

```
node server.js
```

Notre serveur tourne sur le port 2013. Il faut donc entrer dans notre navigateur l'url : http://localhost:2013. Il est très important d'accepter le partage de vidéo sinon cela ne pourra pas fonctionner.
A ce moment-là, vous devriez avoir un écran noir... à la place de vos potentielles bières ! En effet, comme nous ne faisons pas de retour visuel de notre propre caméra, nous devons ouvrir un deuxième onglet sur la même url pour vérifier le bon fonctionnement. Cependant, il y a une deuxième raison pour laquelle nous ne voyons rien, la balise video remoteVideo a un style « display:none'». Il faudra supprimer ce « display:none » le temps du test. Si tout se passe bien, vous devriez avoir une vidéo sur les 2 onglets correspondants au rendu de votre webcam. Pour chaque futur test, je vous conseille de fermer les 2 onglets à chaque fois car le serveur Node stocke le nombre de clients connectés... et la limite a été fixée à 2 clients maximum !

Etape 5 : ajout du mur de flammes

Maintenant que nous nous sommes occupés de la partie WebRTC, nous allons ajouter un peu de graphisme à tout cela. Pour le moment, notre flux WebRTC arrive directement dans une balise vidéo, mais il se trouve que les canvas et les vidéos fonctionnent très bien ensemble ! En effet, nous allons faire des snapshots de notre balise vidéo que nous allons injecter dans un canvas et ainsi pouvoir commencer à jouer plus sérieusement avec des effets graphiques. Nous allons donc avoir :

- balise vidéo en "display:none",
- canvas restituant la vidéo mais avec un masque,
- canvas affichant le mur de flamme.

display:none

Pour ce faire, il suffit simplement de faire en sorte que dans notre html, nous ayons le code suivant :

```
<video id='remoteVideo' autoplay muted style="display:none;"></video>
```

canvas avec la vidéo

Nous allons maintenant ajouter à notre application (app.js) l'affichage du canvas qui recevra la vidéo :
https://gist.github.com/jefBinomed/f524a35662675351eb2f

Nous utilisons simplement la possibilité de dessiner dans un canvas une image d'une vidéo.

Mur de flammes

Vous devez copier le contenu du fichier canvas.js du projet html5-canvas-demo dans notre fichier ./js/canvasFire.js.
Nous allons maintenant afficher le mur dans un canvas. Nous devons donc éditer notre fichier app.js.

```
var canvasFireElement = document.querySelector('#canvasFireLocalVideo');
var ctxFire = canvasFireElement.getContext('2d');
```

Nous allons aussi modifier la méthode snapshot afin d'y intégrer toute la partie flammes.
https://gist.github.com/jefBinomed/1d548336d9e66db196ba

Le code ajouté nous permet d'initialiser graphiquement le canvas et de demander de piloter les rafraîchissements du mur de flammes à partir de notre application. Nous allons donc devoir faire une modification dans le fichier canvasFire.js : ajouter une méthode de rafraîchissement et supprimer l'appel au requestAnimationFrame :
https://gist.github.com/jefBinomed/d8352bfeb0dbb09c69e0

Etape 6 : ajouter un cercle de feux

Le principe : créer un cercle à partir d'un carré

Le principe pour le mur de flammes est simple : le projet html5-canvas-demo nous fournit un seul mur de flammes, or nous, nous voulons 1 cercle. Nous allons nous y prendre en plusieurs étapes.

- Création d'un mur de flammes sur chacun des axes cardinaux. De cette façon nous aurons des flammes partout autour de notre image,
- Mise en place un masque ovale afin de restreindre la zone affichant les flammes,
- Rotation de ces flammes pour leur donner plus de mouvement et se rapprocher du rendu du jeu Portal,
- Enfin, autorisation d'une deuxième couleur car chaque portail possède sa propre couleur (bleu et orange).

Pour rappel, le projet initial fonctionne de la façon suivante : à chaque fois qu'il peut dessiner (window.requestAnimationFrame), on dessine une image de particules de flammes grâce à la méthode drawImage du context du canvas.

Des flammes selon les axes cardinaux

Afin d'afficher les flammes selon les axes nous allons créer une fonction qui nous permet d'afficher pour un angle donné une image de flamme dans le fichier canvasFire.js. (voir code complet). Cette méthode applique une rotation du contexte du canvas avant de dessiner la flamme. Une des choses importantes dans cette méthode est le retour à la position initiale ! C'est très important pour pouvoir traiter un nouvel angle ! Nous devons maintenant appeler cette méthode là où le drawImage d'origine était effectué, à savoir dans la méthode "draw". Le contenu de cette fonction sera désormais le suivant :
https://gist.github.com/jefBinomed/10c927e157e4ff531cf7

Il faut également ajouter les dimensions de l'image de destination afin de calculer les bons angles.
https://gist.github.com/jefBinomed/2df5ea08063138042cea

Il faut enfin modifier la façon d'afficher les pixels afin de forcer un affichage de pixels transparents ! En effet, actuellement seul le dernier canvas est affiché. Il faut donc modifier la fonction drawPixel comme suit :
https://gist.github.com/jefBinomed/2a401edfc41f48252f51

Comme nous avons changé la méthode init de canvasFire.js. Nous devons donc changer l'appel à cette méthode dans app.js. Ainsi dans la méthode snapshot, il faut remplacer :

```
canvasDemo.init();
```

par

```
canvasDemo.init({width :minVideoWidth, height : idealHeight + 100});
```

Nous constatons que nous allons bien afficher 4 murs de flammes autour de notre vidéo.

Mise en place des masques

Dans l'application finale, la vidéo et les flammes s'affichent dans un ovale. Nous allons utiliser pour ce faire une fonction des canvas : clip. Cette dernière nous permettra de définir une zone ovale dans le canvas et nous y dessinerons notre vidéo et nos flammes.
Plus précisément, nous allons dessiner un ovale de flammes et, par-dessus, nous dessinerons un ovale contenant l'image de la vidéo.
Voici donc la version presque finale de la méthode snapshot dans le fichier app.js. (voir code complet). Il faut ajouter la fonction :
https://gist.github.com/jefBinomed/3b94d59e48de32153f9d

Nous pouvons donc voir que l'on dessine des Ellipses dans lesquelles nous faisons nos affichages de flammes suivis de l'affichage de la vidéo.

Rotation des flammes

La gestion de la rotation des flammes se fait dans le fichier canvasFire.js. Précédemment nous avions défini l'affichage du mur de flammes selon 4 axes cardinaux. Nous allons donc simplement faire évoluer ces angles au fil du temps pour mettre en place la rotation.
Commençons par ajouter de nouvelles variables globales :
https://gist.github.com/jefBinomed/f6826c4fd26661ba12f5

Nous allons simplement mettre à jour le contenu de la fonction draw pour faire évoluer les angles :
https://gist.github.com/jefBinomed/83ab0f24bada20476d2e

Maintenant notre cercle de flammes évolue en tournant.

Ajout d'une autre couleur

Afin de compléter comme il se doit notre *portal*, nous devons donc ajouter une deuxième couleur ! Dans le fichier canvasFire.js, la définition de la couleur utilisée se fait dans la méthode initPalette();. voir le code complet sur www.programmez.com
Nous allons modifier son appel dans la méthode init et ajouter la variable globale color.
https://gist.github.com/jefBinomed/4101de57ab41bbed1037

Comme vous le constatez, nous avons fait évoluer la signature de la méthode init. Ceci veut dire que nous allons faire notre ultime modification dans le fichier app.js dans la méthode snapshot(). Nous allons remplacer :

```
canvasDemo.init({width :minVideoWidth, height : idealHeight + 100});
```

par :

```
canvasDemo.init(isInitiator ? 'red' : 'blue', {width :minVideoWidth, height : idealHeight + 100});
```

Nous en avons fini avec le code !

Etape 7 : Fin

Il ne vous reste plus qu'à projeter le résultat de l'application sur 2 murs différents avec une webcam de chaque côté pour filmer le tout. Le rêve devient réalité, et vous êtes fin prêt à concurrencer Homer Simpson, confortablement installé dans votre salon. A vous de jouer et de créer votre propre Portal !

La BlueFrog
Une mini board pour un maximum de possibilités

La BlueFrog est une nouvelle carte programmable ultra-compacte, pour le prototypage d'objets "intelligents" – connectés ou non. Très complète et nettement plus puissante qu'une Arduino, elle ambitionne d'être facilement exploitable par des néophytes... tout en permettant aux codeurs expérimentés de se faire plaisir. La BlueFrog sera proposée en crowdfunding cet été, en voici un avant-goût.

Xavier Cauchy
Développeur de La BlueFrog. Projet que j'ai initié pour prototyper une idée d'accessoire configurable, ne trouvant pas d'alternative satisfaisante.
la-bluefrog.io.
@labluefrog.

Complète et compacte

Dans 4.2cmx4cmx1.2cm, la BlueFrog contient, une batterie rechargeable par USB, un écran OLED "full color" (optionnel) et une carte électronique programmable très fournie. Celle-ci embarque un processeur 32-bits ultra-basse consommation à coeur ARM Cortex-M4, une mémoire Flash pour stocker des données, de nombreux capteurs, la connectivité BlueTooth Smart ainsi qu'USB, et des broches d'extension largement configurables pour interagir avec du matériel extérieur.
La carte se programme en C (ou C++.) Pour en permettre une exploitation facile, elle est livrée avec un package logiciel open-source (APIs, exemples d'applications, etc.).
Divers design de boitiers imprimables en 3D seront également mis en disposition.
Enfin, pour s'adapter à différentes situations, La BlueFrog se décline en versions "intermédiaires": avec ou sans connectivité BlueTooth Smart, avec ou sans écran OLED, avec ou sans l'ensemble des capteurs.

Positionnement

Pour des dimensions comparables aux plateformes type mini-Arduino, *La BlueFrog* est plus fournie et dispose d'un processeur beaucoup plus rapide – proposant ainsi "de base" nettement plus de possiblités, matérielles et logicielles. Cela en fait un choix pertinent dès lors que l'on s'éloigne d'applications peu exigeantes pour lesquelles Arduino fait merveille.
D'un autre côté, elle ne cherche pas à se comporter en mini-PC, que ce soit en termes de puissance de calcul pure ou de connectivité LAN. Elle fait plutôt le choix de combiner richesse des interactions avec son environnement, et basse consommation énergétique.
En ce qui concerne sa programmation, même s'il n'est pas question, pour l'instant en tous cas, d'IDE Arduino ni de 'sketch', on retrouvera des fonctions familières dans les librairies proposées, par exemple pour allumer une LED ou positionner une pin d'IO. Mais *La BlueFrog* permet d'aller plus loin et d'utiliser toute la puissance du C.

Pour quels types de projets ?

L'utilisation d'un mini-module comme La BlueFrog sera bien adapté aux projets pour lesquels l'encombrement ou le poids du module sont des critères importants : dispositifs à porter sur soi, objets connectés, petits robots autonomes, engins volants, etc. Par exemple, pour prototyper de façon crédible un bracelet connecté, une board de la taille d'une carte de crédit est déjà trop encombrant. Même avec une board de très petite dimension il faudra encore prêter attention à la solution globale, y compris les *shields* ou sources d'alimentation à ajouter.
Pour les projets qui doivent, en plus d'être compacts et peu consommateurs d'énergie, faire tourner des algorithmes relativement évolués, La BlueFrog sera d'autant plus intéressante avec son micro 32 bits. Enfin, La BlueFrog doit permettre aux projets qui l'utilisent de se concentrer rapidement sur leur application, grâce aux librairies fournies pour supporter le hardware pré-intégré.
Quelque exemples de réalisations récentes ou en cours: un punching ball instrumenté, un drône "hacké" et contrôlé par BlueTooth Smart, un démonstrateur d'accessoire de mode personnalisé, une veilleuse pour chambre d'enfants, un robot programmé par langage "custom", un système embarqué de mesure de l'efficacité en cyclisme...

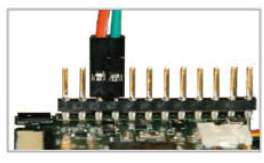
Le matériel

Avec 160x128 pixels en technologie OLED, l'afficheur, d'épaisseur minime, offre une bonne résolution, des couleurs saturées et un large angle de vision. La batterie de technologie LiPo (lithium-polymère) a une capacité de 500mA.h. Cela se traduit par une douzaine d'heures d'autonomie avec un écran affichant en permanence des photos variées, et plusieurs jours à plusieurs semaines (en fonction de l'application et des précautions prises pour limiter la consommation), sans écran actif.
Sur la carte, le micro-contrôleur est un STM32L4 avec 512Ko de mémoire programme et 64K de RAM. Capable d'un cadencement jusqu'à 80MHz, il intègre de multiples périphériques, une FPU et un accélérateur graphique DSP. La mémoire Flash additionnelle de 64MBits peut être vue comme mémoire de masse accessible depuis USB par un PC. De multiples capteurs embarqués fournissent des informations sur le contexte et l'environnement: accélération linéaire et angulaire (gyro), champ magnétique, température, pression/altitude, lumière ambiante, proximité immédiate et distance d'objets, son ambiant. La connectivité est assurée par une liaison BlueTooth Smart (4.x ou "Low Energy") pour communiquer avec un smartphone et par un port USB pour se connecter

Sans écran

Avec écran

Vue de dos



Deux exemples d'utilisation des broches d'extension

à un PC. Si besoin, il est également possible d'interfacer avec un système extérieur via des broches d'extension configurables en GPIO, interruption, bus I2C, SPI ou UART, ou encore en port analogique. La carte est fournie avec une rangée de broches type “press-fit” au pas standard 0.1”/2.54mm. Selon l'usage visé on peut les souder ou simplement les enfoncer dans la carte.

Le Logiciel

Le STM32 qui contrôle La BlueFrog est librement programmable. Cependant, un package software open source est fourni pour faciliter l'exploitation du module. En version bêta aujourd'hui, il évoluera et sera enrichi au cours du temps. Au plus bas niveau, on trouve d'abord une librairie (CMSIS) qui correspond à la description des registres qui contrôlent le STM32 et son coeur ARM Cortex-M. En règle générale, sauf à vouloir travailler à très bas niveau, l'utilisateur n'a pas à s'en soucier, il suffit qu'elle soit dans la liste des fichiers à compiler.

On trouve ensuite la librairie HAL (Hardware Abstraction Layer) de STMicroelectronics qui permet de piloter les périphériques du STM32, par exemple: DMA, UART, I2C etc. Comme sa polyvalence induit une relative complexité d'utilisation, des librairies complémentaires ont été spécifiquement développées pour La BlueFrog pour pouvoir travailler à plus haut niveau, et simplifier encore la programmation, au travers d'une API évoquée ci-dessous.

Les librairies spécifiquement développées pour La BlueFrog incluent :

1) De quoi configurer simplement tout le matériel (horloges, I/Os, périphériques, sources d'interruptions, etc). Deux fonctions y suffisent: *LBF_Board_Fixed_Inits()* et *LBF_Board_Selective_Inits().* Cette dernière utilise des informations fournies par l'utilisateur dans un fichier *User_Configuration.h.* Par exemple, pour utiliser la position 5 du connecteur d'extension en sortie programmable “drain-ouvert” l'utilisateur incluerait simplement dans ce fichier la définition suivante:
   `#define POS5_IS_GPIO_OD`
   Concrètement, un fichier “modèle” contenant toutes les directives supportées est fourni, il suffit d'y “décommenter” les directives souhaitées.
2) Est également fournie une API (Application Programming Interface), c'est-à-dire un ensemble de fonctions prêtes à l'emploi pour faciliter l'exploitation des diverses ressources de la board – LED, capteurs, connecteur d'extension, etc. Dans certains cas il s'agit de fonctions développées spécialement, dans d'autres cas d'une encapsulation et simplification de fonctions plus génériques développées par les fournisseurs des circuits utilisés. Un certain nombre de choix ont été faits pour proposer avec cette API un niveau d'abstraction plus élevé et une syntaxe plus simple que ce qu'offrent les librairies génériques – cependant ces dernières restent bien entendu complètement accessibles.
3) Enfin, trois librairies de haut niveau (“middleware”) sont pré-intégrées. La première d'entre elles permet de voir La BlueFrog comme une clé USB depuis n'importe quel PC – le stockage se fait sur la Data Flash embarquée (8Mo). La seconde (basée sur FatFs de ChaN) permet d'utiliser la Data Flash comme un File System de type FAT: création de fichiers, lectures/écritures, parcours de répertoire, etc. La dernière est une librairie graphique professionnelle, emWin (licenciée par Segger via ST) – qui permet aussi d'afficher de façon simple et flexible du texte à l'écran.

Bien sûr, rien n'oblige à utiliser tout ou partie de ce package software, l'utilisateur reste libre de programmer le STM32 comme il l'entend.

RIOT OS – The friendly operating system for the Internet of Things

RIOT OS est un système d'exploitation à micro-kernel qui vise en particulier les objets connectés. Initié par FU Berlin, l'INRIA et HAW Hamburg, il est maintenant développé en open source par une communauté mondiale. Très économe en ressources, il est bien adapté aux coeurs Cortex-M. Il permet de programmer en C/C++, offre du multi-threading et des capacités temps-réel, et intègre différentes piles réseau dont IPv6 et des protocoles standards tels que CoAP. Enfin, il existe un “portage natif” qui permet d'exécuter RIOT sur un ordinateur en tant que processus Linux ou MacOS. Plus d'informations: www.riot-os.org .

Avec ou sans OS

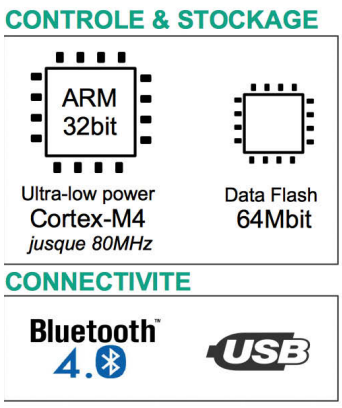
Jusqu'à présent nous n'avons pas parlé d'OS et effectivement toutes les librairies et exemples de code décrits ci-dessus tournent sur le STM32 'bare metal'. Cependant, pour écrire des applications plus complexes, un OS peut s'avérer bien utile. Dans cette optique, le portage d'un OS prometteur, RIOT OS, est en cours (voir encadré). Il reste bien entendu possible d'utiliser un autre OS ou noyau temps réel: il existe de multiples options adaptées à la famille STM32 – par exemple FreeRTOS.

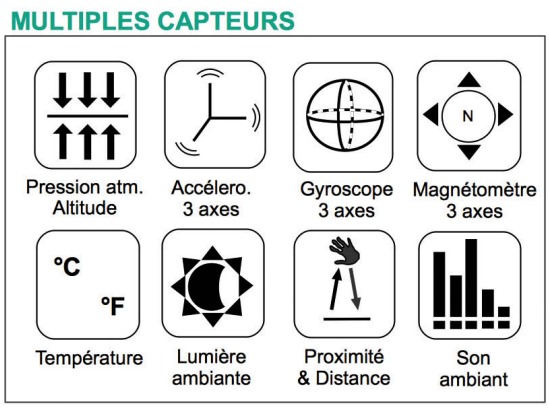
Mise en oeuvre

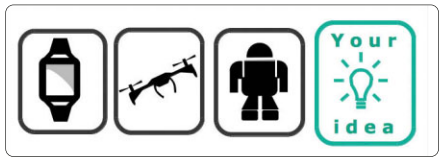
Le code C compilé pour coeur ARM sur PC est chargé dans la mémoire programme du STM32 via un dongle (fourni) qui se connecte d'un côté au port USB du PC, de l'autre au port SWD de la carte. Il permet également de déboguer depuis le PC et, par exemple, d'exécuter le code pas à pas sur le STM32.

Il est également possible de charger le code par USB, moyennant la mise à 3V (“1” logique) d'une broche dédiée accessible en bordure du board – sans possibilité de debug in-situ cette fois.

Pour développer, une première approche possible est de travailler dans un terminal avec un makefile (inclus dans les exemples fournis) qui gère les sources, leur compilation avec gcc (suite *gcc-arm-none-eabi*), et qui peut prendre en charge le chargement dans la mémoire programme du STM32. Une seconde approche



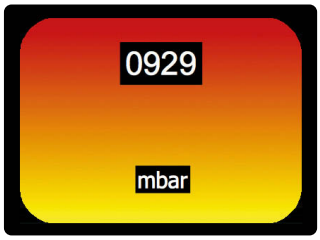




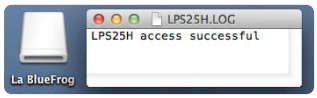
est d'utiliser un environnement de développement intégré (IDE). Il existe de multiples solutions compatibles STM32, commerciales (en général sous Windows) ou libres. En particulier, la société AC6, en partenariat avec ST, propose un IDE “STM32 Workbench” sous Eclipse, librement téléchargeable et disponible en versions Windows et Linux. La chaîne de compilation sous-jacente reste gcc. Un travail en cours permettra d'utiliser le code fourni avec La BlueFrog aussi bien dans la première approche qu'avec cet IDE.

Mini-Tuto : exemple concret de projet

Pour illustrer les possibilités de La BlueFrog, rien de tel qu'un petit exemple concret!
Le mini-projet suivant se propose de vérifier que le capteur de pression est accessible, d'alors écrire un court message de succès dans un fichier créé à cet effet et lisible par USB depuis un PC, et d'effectuer régulièrement une mesure de pression atmosphérique, affichée à l'écran (en mbar). La partie initialisations est standard et suit le modèle suivant. Ici on souhaite utiliser les trois middleware pré-intégrés (USB, FatFs et emWin), ils sont donc activés lors de l'initialisation.



Résultat (écran)



La BlueFrog vue comme stockage USB depuis le PC

```
int main(void)
{

boolean_t  Success = TRUE;

/* ============================================== */
/* Board Initializations and Configurations       */
/* ============================================== */

  LBF_Board_Fixed_Inits();
  LBF_Board_Selective_Inits();
  // At this point, the module is fully configured

  LBF_Led_ON();

/* ============================================== */
/* Optional initialization of Middleware libraries :  */
/* USBD drivers, FatFS File System, STemWin GUI ,     */
/* (Comment out / Uncomment as needed)                */
/* ============================================== */

  /* ... To use La BlueFrog as USB Mass Storage (Full Speed)    */
  Success &= LBF_LaunchUSB_MassStorage();

  /* ... To initialize FatFS and mount Data Flash as File System  */
  Success &= LBF_FatFS_Init();

  /* ... To initialize the STemWin Graphical Library          */
  /*    Caution: reserves some RAM                            */
  Success &= LBF_emWin_Init();

  // ERROR HANDLER  -  Replace by your own as wished */
  Led_StopNBlinkOnFalse (Success);
     // stops here and blink LED if one of above inits has failed

/* ============================================== */
/* Application Code Below */
/* ============================================== */
// .....
}
```

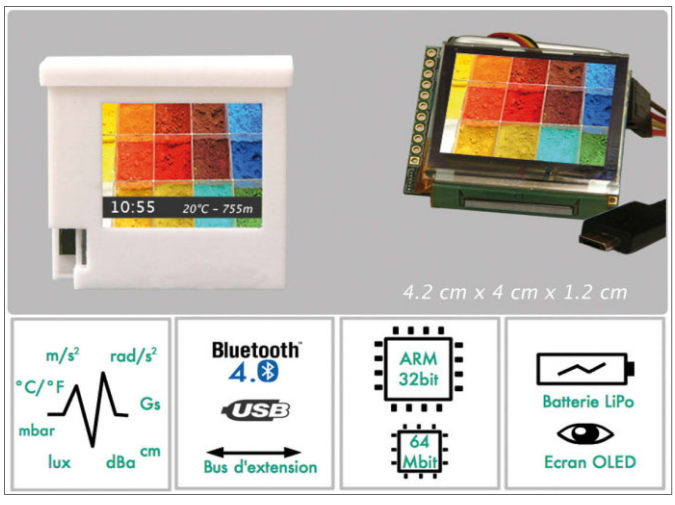
Ce code initialise donc toute la board, allume une LED témoin, puis active les middleware USB, FatFs et emWin.
Passons au code applicatif proprement dit. Il pourrait se présenter comme suit :

```
/* == User Declarations  ================================ */
// Variables
FIL      MyFile;
uint32_t   wbytes_count; /* File write counts */
uint8_t    wtext[] = "LPS25H access successful\r\n"; /* File write buffer */
uint8_t    ReadValue;
int32_t    Pressure_mbar;
// Private functions
int32_t    Get_Pressure_mb(void);


/* == Body  ============================================= */


 /* ---   Check access to Pressure Sensor :  ST LPS25H  -------- */

  ReadValue = I2C2_ReadSingleReg(LPS25H_CHIPID, LPS25H_WHOAMI);
  Led_StopNBlinkOnFalse ( ReadValue == LPS25H_WHOAMI_CONTENTS );
     // to stop here and blink LED if failure

 /* ---   Log success in file created on DataFlash File System -- */

  if(f_open(&MyFile, "LPS25H.LOG", FA_CREATE_ALWAYS | FA_WRITE) == FR_OK)
  {
    f_write(&MyFile, wtext, sizeof(wtext)-1, (void *)&wbytes_count);
    f_close(&MyFile);
  }

 /* ---   Prepare Pressure Sensor   ---------------------------- */

  // Enable single shot mode with internal IT generated when data ready
  I2C2_WriteSingleReg(LPS25H_CHIPID, LPS25H_CTRL_REG1_ADDR, 0x84);


 /* ---   Prepare Display of Results   ------------------------ */

  OLED_Switch_ON();
  // Provide power (13V) to OLED panel, enable display

  // Draw background:
  //rounded rectangle filled with vertical gradient
  GUI_DrawGradientRoundedV(10, 10, X_FULL_SCREEN-10, Y_FULL_SCREEN-10, 10,
              GUI_RED, GUI_YELLOW);
```

```
/* ---   Track  Pressure        ------------------------ */

while(1)
{

    Pressure_mbar =  Get_Pressure_mb();

    // --- Display Result:-----------------------------------------------

    GUI_SetFont(&GUI_Font24B_ASCII);
    GUI_SetTextAlign(GUI_TA_HCENTER | GUI_TA_VCENTER);
    GUI_DispDecAt( Pressure_mbar,  X_FULL_SCREEN/2, Y_FULL_SCREEN/4, 4);
    // Display pressure with 4 digits, centered on specified position

    GUI_SetFont(&GUI_Font16B_ASCII);
    GUI_SetTextAlign(GUI_TA_HCENTER | GUI_TA_VCENTER);
    GUI_DispStringAt( "mbar", X_FULL_SCREEN/2, Y_FULL_SCREEN*3/4 );
    // Display units, centered on specified position

    Delay_ms(1000);


 } //  end while(1)

}
```

La vérification du fonctionnement correct de la communication entre le STM32 et le capteur LPS25H se fait par accès explicite à son registre d'identification WHOAMI. C'est au bus I2C numéro 2 du STM32 que sont connectés les capteurs de La BlueFrog (info disponible dans la documentation hardware), c'est donc via ce bus que les accès aux registres du LPS25H sont faits. Une alternative serait d'utiliser une API plus haut niveau (en cours d'intégration), permettant de simplement demander un statut au capteur, sans même expliciter les accès registres nécessaires.
La création et l'écriture du fichier “LPS25H.LOG” sur la Flash, vue comme FileSystem de type FAT, se font via des fonctions bien documentées dans l'API du File System FatFs, en l'occurence f_open() et f_write(). La syntaxe et les déclarations ncéessaires sont faciles à suivre en se basant sur les exemples de la documentation FatFs.
L'affichage du résultat est réalisé avec les fonctions GUI_xxx fournies par la librairie emWin, pour générer le texte et les graphiques.
L'obtention de la pression se fait par une fonction *Get_Pressure_mb()*. C'est ici une fonction “privée” du main(), mais elle aurait aussi pu être fournie par une API permettant de consulter le capteur avec un plus haut niveau d'abstraction. La fonction *Get_Pressure_mb()* peut se présenter comme suit (note: *I2C2_ReadSingleReg()* fait partie des librairies fournies avec La BlueFrog et encapsule une fonction plus générique d'accès au périphérique I2C fournie par ST) :

```
int32_t  Get_Pressure_mb(void)
{

uint8_t     Pressure_Bytes[4];
int32_t     Pressure_Signed;

    // Launch Single Shot Measurement, bit will auto-cleared when data ready
     I2C2_WriteSingleReg(LPS25H_CHIPID, LPS25H_CTRL_REG2_ADDR, 0x01);

  // Wait until data available from Pressure Sensor
    while ( I2C2_ReadSingleReg(LPS25H_CHIPID, LPS25H_CTRL_REG2_ADDR) != 0x00);


  // Read Results on 3 Bytes (24 bits in 2's complement)
    Pressure_Bytes[0] =
      I2C2_ReadSingleReg(LPS25H_CHIPID, LPS25H_PRESS_OUT_XL_ADDR);
    Pressure_Bytes[1] =
      I2C2_ReadSingleReg(LPS25H_CHIPID, LPS25H_PRESS_OUT_L_ADDR);
    Pressure_Bytes[2] =
      I2C2_ReadSingleReg(LPS25H_CHIPID, LPS25H_PRESS_OUT_H_ADDR);

    // Expand to 4th MSByte according to sign of result on 24 bits
    (Pressure_Bytes[2]&0x80) == 0x80) ?
      (Pressure_Bytes[3] = 0xFF) :  (Pressure_Bytes[3] = 0x00) ;

    Pressure_Signed = (int32_t)( (uint32_t)Pressure_Bytes[3]<<24
                    | (uint32_t)Pressure_Bytes[2]<<16
                    | (uint32_t)Pressure_Bytes[1]<<8
                    | (uint32_t)Pressure_Bytes[0]     );

    return( Pressure_Signed / 4096 );  //in mbar, as per LPS25H datasheet
}
```



Conclusion et Perspectives

Le lancement de La BlueFrog est prévu sur Kickstarter dans le courant de l'été 2015. Dans le même temps, l'environnement software continuera d'être enrichi. Outre l'amélioration du package “C” fourni, le portage de MicroPython ou eLua sur La BlueFrog sont des axes de réflexion intéressants (...avis aux amateurs!) – cela s'est déjà fait sur Cortex-M4.
Le prix visé s'échelonne, en fonction des options retenues, dans une fourchette qui va d'une cinquantaine d'euros pour une version “minimale” à une centaine d'euros pour la version “totale”.
La version “minimale” est une base Cortex-M4 avec FPU et DSP, Data Flash 64Mbit, batterie rechargeable LiPo, USB, ports d'extension configurables, et évidemment le software open source. Les versions intermédiaires correspondent aux options avec/sans BlueTooth Smart, avec/sans écran, avec/sans les capteurs.
Enfin, la roadmap de La BlueFrog intègre la mise à disposition de “micro-cartes d'extension”, une fonction GPS étant l'un des premiers candidats.

Fabriquer son récepteur ADS-B avec un Raspberry Pi

Allongé dans l'herbe par ce beau jour d'été, je contemple les traînées laissées sur l'azur du ciel par les avions qui me survolent... Où peuvent-ils bien aller ? Bon sang, il doit bien y avoir un moyen de le savoir ! Et c'est parti : après quelques recherches sur Internet, je farfouille dans la boîte à bricoles j'en sors une clé TNT avec son antenne et un Raspberry Pi. Prêt pour recevoir les infos ADS-B en direct...

François MOCQ
Créateur du blog www.framboise314.fr
Auteur du livre **Raspberry Pi 2** – *Editions ENI*

Objectif de cet article

Le but de cet article est de fabriquer un récepteur ADS-B avec un dongle TNT et un Raspberry Pi. Avec ce récepteur vous pourrez bien entendu connaître la position mais aussi d'autres paramètres de vol des avions qui circulent dans votre zone. Les données pourront s'afficher en mode console sur votre écran, ou sur une page Web affichée sur un autre poste ou un smartphone. Mais l'histoire ne s'arrête pas là, puisque si vous vous inscrivez auprès de Flightradar24 et que vous leur envoyez les données recueillies, vous bénéficierez du téléchargement gratuit de l'appli – elle ne coûte que 3,59€ - pour votre smartphone ainsi que d'un compte premium dont nous verrons les avantages plus loin.

ADS-B

Chaque avion de ligne embarque aujourd'hui un transpondeur ADS-B (**Automatic Dependent Surveillance-Broadcast**). La position exacte de l'appareil est déterminée grâce à un récepteur GPS. A ces données de position (lattitude, longitude et altitude) sont ajoutés le numéro du vol, l'indicatif de l'avion, les informations de vitesse et de vitesse ascensionnelle, le cap...

Ces données sont transmises sous forme de « burst » binaire (112 bits) non crypté sur 1090MHz. Dans ce mode S, les informations sont envoyées régulièrement en fonction de la phase de vol, par exemple toutes les dix secondes en vol normal et toutes les secondes en phase d'approche. Ces émissions peuvent être reçues par des stations au sol ou par les autres avions. Chaque avion peut ainsi connaître avec précision la position des appareils qui l'entourent et utiliser ces informations pour prévenir les collisions.

L'émission de ces données est omnidirectionnelle et peut être reçue facilement au sol par un système simple, muni d'une antenne et d'un récepteur. Cette installation permet de suivre les avions en vol sans nécessiter un radar beaucoup plus coûteux **Fig.1**.

Récepteur TNT USB

Des récepteurs TNT = Télévision Numérique terrestre (appelés en anglais DVB-T = Digital Video Broadcasting – Terrestrial) sont disponibles sur les sites de vente en ligne (photo 1) pour des prix allant de quelques euros à une vingtaine d'euros. Je disposais de plusieurs exemplaires suite à des essais de réception TV finalement pas très heureux... Mais mes dongles TNT allaient enfin pouvoir reprendre vie !

Pour arriver à construire des récepteurs vendus à un prix aussi bas, les fabricants ont recours à des composants spécialisés et fortement intégrés. C'est ainsi qu'une fois ouvert (photo 2), un dongle TNT ne laisse apparaître que trois circuits intégrés, accompagnés de quelques composants annexes **Fig.2**.

Ces récepteurs sont vendus pour recevoir les bandes radio FM (87,5 à

Fig.1 *Le récepteur TNT et son antenne*

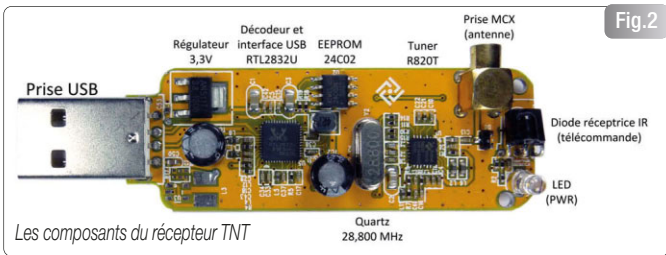


Fig.2 *Les composants du récepteur TNT*

108 MHZ), TV TNT en UHF (470 à 790 MHz). Enfin... ça c'est la version officielle. Parce que dans la réalité, les composants mis en oeuvre permettent de couvrir une gamme de fréquences beaucoup plus large que celle qui est annoncée. Et c'est bien ce qui nous intéresse puisque la fréquence ADS-B à recevoir se situe au delà de 1 GHz.

Avec le tuner R820T utilisé sur ce dongle, la réception des fréquences comprises entre 24 MHZ et 1850 MHz devient possible, ouvrant la voie à de nombreuses applications SDR (Software Defined Radio = Radio logicielle). Le tuner R820T se charge de la réception des signaux qu'il transmet au décodeur RTL2832U, chargé de les numériser et de les transmettre sous forme d'octets via le port USB jusqu'à l'ordinateur (PC ou dans notre cas, Raspberry Pi !). Celui-ci se charge alors du traitement des échantillons en démodulant les signaux et/ou en calculant le spectre des signaux.reçus. Avec cette méthode, l'ordinateur est capable de traiter un spectre (un ensemble de signaux) s'étendant sur plusieurs mégaherz.

Réception des signaux ADS-B avec le Raspberry Pi

Pour recevoir les signaux ADS-B avec le Raspberry Pi et un dongle DVB-T USB, l'antenne livrée avec le dongle suffit. Il ne faudra pas attendre des résultats extraordinaires mais les premiers essais que j'ai réalisés avec l'antenne posée sur un appui de fenêtre au premier étage ont permis la réception d'avions situés à près de 150 Km. Il faut dire que le ciel est particulièrement dégagé vers le sud. Il n'y a ni bâtiment proche, ni obstacle naturel (colline par exemple) qui pourrait empêcher la réception des signaux. A cette fréquence, même si on constate des réflexions et de la dispersion, la meilleure réception a lieu quand les deux antennes

Liste du matériel

1 Raspberry Pi 2
1 boîtier pour le Raspberry Pi 2
1 alimentation 5V / 2A avec connecteur micro USB
1 carte SD 16Go Samsung
1 clé DVB-T modèle SPC-0155
1 antenne (livrée avec la clé USB)

sont à vue. Enfin, quand l'avion est à 100 Km, on ne voit pas grand-chose, mais les ondes, elles, passent bien. Le décodeur RTL2832U transforme les signaux reçus en données de la forme I/Q, I représentant l'amplitude instantanée du signal et Q sa phase. Au risque de gacher vos vacances d'été, je vous rappellerai juste que la porteuse modulée peut s'écrire sous la forme **I cos (f.t) + Q sin (f.t)**. Ça pique un peu non ?

Choisir le système :

Il y a deux possibilités pour préparer la carte microSD. La première est une installation directe de Raspbian. Elle nécessite la mise en oeuvre de Win32DiskImager pour la création de la carte et le transfert d'une image comportant deux partitions, une partition FAT et une partition ext4. Vous choisirez cette option si vous êtes à l'aise avec les outils utilisés. Pour les débutants NOOBS est le bon choix. Il ne nécessite pas de connaissance particulière et s'installe par simple copie des fichiers sur la carte microSD.

Préparer la Carte SD :

Choisissez une carte micro SD de bonne qualité. J'utilise pour ma part des Samsung EVO 16 Go qu'on trouve pour moins de 10¤. Evitez les cartes microSD "pas chères" trouvées sur certains site d'enchères en ligne. De nombreux utilisateurs de ces cartes à la capacité inférieure à la valeur annoncée ont connu des déboires lors de l'installation de NOOBS, celui-ci indiquant que la capacité est insuffisante pour l'installation. Le message d'erreur "Available Space : 0" vous alertera sur ce problème. SI vous récupérez une carte SD précédemment utilisée pour équiper votre récepteur ADS-B, la Fondation recommande de formater la carte avec **SD Formatter 4.0** (https://www.sdcard.org/downloads/formatter_4/) avant d'installer NOOBS **Fig.3**.

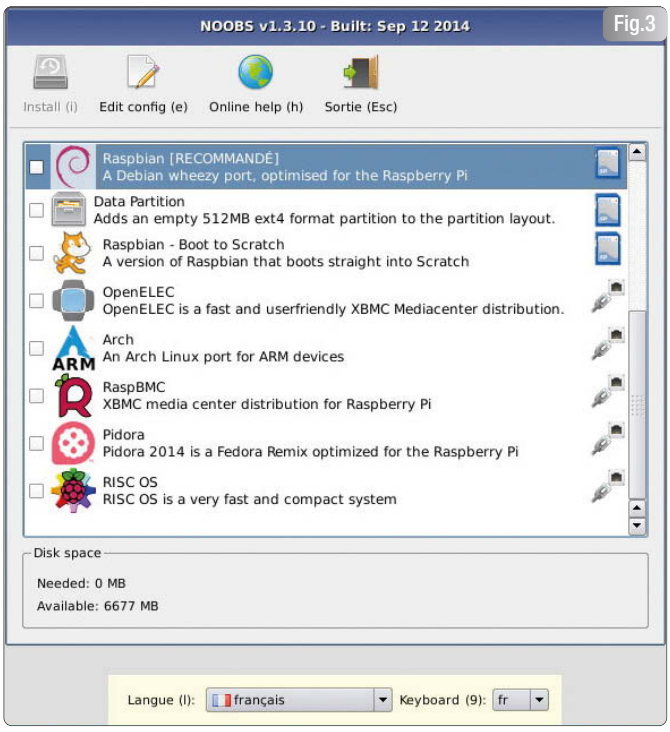


Fig.3

Démarrage de NOOBS

Rendez-vous sur la page de téléchargement de la Fondation (https://www.raspberrypi.org/downloads/). NOOBS existe en deux versions. La version "*Offline and network install*" représente 742 Mo de téléchargement. Elle embarque Raspbian et permet d'installer le système sur un Raspberry Pi non relié à Internet, ou sur un modèle A+ qui ne dispose pas de prise Ethernet. La version "*Network install only*" ne représente que 21,8 Mo de téléchargement, mais elle ne peut être utilisée que sur un Raspberry Pi connecté à Internet car cette image ne contient aucun système et doit télécharger les systèmes à installer. Choisissez la version qui correspond le mieux à vos besoins. Une fois le fichier .zip récupéré, décompressez l'archive dans un dossier. Ouvrez le dossier et copiez les fichiers qui s'y trouvent sur votre carte microSD. Votre système est prêt à démarrer !

Démarrer le Raspberry Pi

Connectez le clavier, la souris et l'écran HDMI. Branchez l'alimentation, le Raspberry Pi démarre et affiche son écran coloré. NOOBS se lance. Vous aboutissez sur la fenêtre de choix du système d'exploitation à installer (photo 3). Cochez la case figurant au début de la ligne de Raspbian, choisissez Français pour la langue du système et du clavier (en bas de l'écran), puis cliquez sur l'icône **Install** en haut à gauche. Servez-vous une bière bien fraîche et installez vous confortablement... Il y en a pour un moment **Fig.4**.

Quand le processus d'installation est achevé, Raspbian démarre et vous atterrissez sur l'écran de configuration de Raspbian, **raspi-config**. La première ligne est sélectionnée. Elle permet à Raspbian d'occuper toute la place disponible lorsque vous l'installez seul sur une carte. Avec NOOBS, cette option n'est pas disponible. Déplacez vous sur **<Finish>** avec la touche de tabulation, puis validez ce choix avec la touche Entrée. Vous arrivez sur la console. Pour ce premier démarrage vous n'avez pas besoin de vous logger. Par la suite vous utiliserez le login **pi** et le mot de passe **raspberry**. Si vous souhaitez relancer l'utilitaire de configuration de Raspbian, exécutez la commande **sudo raspi-config**.

Commencez par mettre votre système à jour:

```
pi@raspberrypi ~ $ sudo apt-get update
pi@raspberrypi ~ $ sudo apt-get upgrade
```

Connecter le Dongle TNT


Fig.4

Le boîtier destiné à abriter le montage et l'alimentation à découpage.

Brancher votre dongle USB TNT sur un des ports USB du raspberry Pi. Pour vérifier que votre dongle est bien détecté, tapez la commande **dmesg**. Parmi les lignes qui apparaissent, repérez les lignes suivantes qui indiquent que le récepteur TNT a bien été détecté, puisque les circuits intégrés RTL2832U et R820T sont mentionnés. Nous allons pouvoir continuer l'installation.

```
[  44.837951] usb 1-1.2: new high-speed USB device number 5 using dwc_otg
[  45.040538] usb 1-1.2: dvb_usb_v2: found a 'Realtek RTL2832U reference design' in warm state
[  45.103415] DVB: registering new adapter (Realtek RTL2832U reference design)
[  45.147362] r820t 0-001a: Rafael Micro r820t successfully identified
[  45.179781] IR RC5(x/sz) protocol handler initialized
[  45.222421] usbcore: registered new interface driver dvb_usb_rtl28xxu
```

En fait Raspbian a installé le pilote permettant de regarder la télévision avec le dongle DVB-T (TNT). Pour qu'il ne vienne pas perturber la réception des signaux ADS-B, nous allons le placer en liste noire (blacklist) : Créez un fichier **rtlsdr.conf** dans le répertoire **/etc/modprobe.d**

```
pi@raspberrypi ~ $ cd /etc/modprobe.d
pi@raspberrypi /etc/modprobe.d $ sudo nano rtlsdr.conf
```

Une fois que **nano** est ouvert ajouter la ligne suivante dans le fichier :

```
blacklist dvb_usb_rtl28xxu
```

Ensuite sauvegardez le fichier **rtlsdr.conf** avec **CTRL X**. Ainsi blacklisté, le module ne sera plus chargé par le système.
Vérifiez si le pilote est chargé avec la commande **lsmod** :

```
pi@raspberrypi ~$ lsmod | grep dvb_usb_rtl28xxu
dvb_usb_rtl28xxu      14195  0
rtl2830             7227  1 dvb_usb_rtl28xxu
rtl2832             8911  2 dvb_usb_rtl28xxu
dvb_usb_v2          13368  1 dvb_usb_rtl28xxu
rc_core          16932  14 ir_sharp_decoder,ir_xmp_decoder,lirc_dev,ir_lirc_codec,dvb_
usb_rtl28xxu,ir_rc5_decoder,ir_nec_decoder,ir_sony_decoder,ir_mce_kbd_decoder,ir
_jvc_decoder,dvb_usb_v2,ir_rc6_decoder,ir_sanyo_decoder
```

Le pilote est effectivement chargé puisque nous avons connecté le dongle TNT. Il faut le supprimer :

```
pi@raspberrypi ~ $ sudo modprobe -r dvb_usb_rtl28xxu
```

Installation du pilote rtl-sdr

Les dongles comportant un décodeur de type RTL2832U sont gérés par le logiciel **rtl-sdr**. Ce logiciel développé par un groupe de passionnés permet de récupérer directement les données du signal I/Q, ce qui transforme votre dongle USB TNT en récepteur radio à large bande au moyen d'un nouveau pilote. En clair, cela signifie qu'un dongle USB TNT peut remplacer pour une vingtaine d'euros un scanner qui coûtait plusieurs centaines d'euros il y a quelques années... La vitesse maximale d'échantillonnage est aux alentours de 2,4 Me/s (méga échantillons par seconde). C'est à partir de ces données que le signal est reconstitué. Pour installer le logiciel, nous avons besoin des outils de construction et de la librairie **libusb** pour que les applications puissent accéder facilement aux périphériques USB.

```
pi@raspberrypi ~ $ sudo apt-get -fym install git cmake build-essential libusb-1.0-0-dev
```

Maintenant nous pouvons récupérer rtl-sdr sur github et le compiler sur notre Raspberry Pi.

```
pi@raspberrypi ~ $ mkdir git
pi@raspberrypi ~ $ cd git
pi@raspberrypi ~/git $ git clone git://git.osmocom.org/rtl-sdr.git
Cloning into 'rtl-sdr'...
remote: Counting objects: 1607, done.
remote: Compressing objects: 100% (701/701), done.
remote: Total 1607 (delta 1174), reused 1213 (delta 898)
Receiving objects: 100% (1607/1607), 344.85 KiB | 10 KiB/s, done.
Resolving deltas: 100% (1174/1174), done.
pi@raspberrypi ~/git $ cd rtl-sdr
pi@raspberrypi ~/git/rtl-sdr $ mkdir build
pi@raspberrypi ~/git/rtl-sdr $ cd build
pi@raspberrypi ~/git/rtl-sdr/build $ cmake .. -DINSTALL_UDEV_RULES=ON
-- The C compiler identification is GNU 4.6.3
-- Check for working C compiler: /usr/bin/gcc
-- Check for working C compiler: /usr/bin/gcc – works
…/…
pi@raspberrypi ~/git/rtl-sdr/build $ sudo make install
Scanning dependencies of target convenience_static
[  5%] Building C object src/CMakeFiles/convenience_static.dir/convenience/convenience.c.o
Linking C static library libconvenience_static.a
…/…
pi@raspberrypi ~/git/rtl-sdr/build $ sudo ldconfig
pi@raspberrypi ~/git/rtl-sdr/build $ sudo cp ../rtl-sdr.rules /etc/udev/rules.d/
```

Le pilote rtl-sdr est installé et nous pouvons passer à l'installation de dump1090.

Installation du logiciel Dump1090

Le logiciel **dump1090** est un logiciel de décodage des signaux ADS-B mode S spécialement destiné à être utilisé avec les périphériques USB gérés par **rtl-dsr**. Il est capable de décoder des signaux faiblement reçus sur 1090 MHz. Il supporte le protocole TCP30003 qui permet de retransmettre les messages à FlightRadar24, par exemple. Il met à disposition de l'utilisateur un serveur HTTP qui affiche les avions détectés sur une carte Google Map mais peut afficher les données en mode console. Dump1090 existe en version Windows et heureusement pour nous, également en version Raspberry Pi !
Pour installer le logiciel dump1090, nous allons le récupérer sur github et le compiler :

```
pi@raspberrypi ~ $ cd ~/git
pi@raspberrypi ~/git $ git clone git://github.com/MalcolmRobb/dump1090.git
Cloning into 'dump1090'...
remote: Counting objects: 1401, done.
remote: Total 1401 (delta 0), reused 0 (delta 0), pack-reused 1401
Receiving objects: 100% (1401/1401), 4.13 MiB | 329 KiB/s, done.
Resolving deltas: 100% (837/837), done.
pi@raspberrypi ~/git $ cd dump1090
pi@raspberrypi ~/git/dump1090 $ make
gcc -O2 -g -Wall -W `pkg-config --cflags librtlsdr`  -c dump1090.c
…/…
```

Tout est prêt pour lancer le logiciel dump1090 et récevoir les avions qui se trouvent dans les parages :

```
pi@raspberrypi ~/git/dump1090 $ ./dump1090 –quiet –net –enable-agc &
[1] 2325
pi@raspberrypi ~/git/dump1090 $ Found 1 device(s):
```

```
0: Realtek, RTL2838UHIDIR, SN: 00000001 (currently selected)
Found Rafael Micro R820T tuner
Max available gain is: 49.60
Setting gain to: 49.60
Exact sample rate is: 2000000.052982 Hz
Gain reported by device: 49.60
```

Connectez-vous depuis un navigateur sur l'adresse de votre Raspberry Pi au port 8080, vous obtenez une carte indiquant les avions en vol près de chez vous (**Fig.4**). Leur position est actualisée à chaque réception de trame ADS-B et si vous cliquez sur un avion, vous affichez les données le concernant. Vous pourrez trouver plus d'informations le concernant en cliquant sur les liens vers FlightRadar24, FlightStats ou FlightAware. **Fig.5.**

Personnaliser dump1090

Si vous souhaitez adapter dump1090 à vos besoins, le fichier ***config.js*** situé dans le répertoire ***public_html*** de dump1090 vous permet de régler successivement:

- l'utilisation des valeurs métriques ou de miles nautiques,
- le centre de la carte (latitude longitude et zoom)
- la couleur des marqueurs (MarkerColor)
- l'emplacement du repère (SiteSHow, SiteLat et SiteLon)
- la présence de cercles indiquant la distance des avions

Le fichier ***config.js*** ci-dessous contient mes propres réglages.

```
// ----------------------------------------------------------
//
// Ce fichier permet de modifier l'affichage sous Dump1090.
// Chargez ce fichier avant script.js dans gmap.html.
//
// ----------------------------------------------------------

// -- Valeurs de sortie ------------------------------------
// Affichage métrique : true pour afficher en km
Metric = true; // true ou false

// -- Configuration de la carte ----------------------------
// Latitude et Longitude en format décimal
// Indique le centre de la carte qui sera affichée
// J'ai choisi de centrer sur mon antenne ADS-B
CONST_CENTERLAT = 46.799107;
CONST_CENTERLON = 4.418899;
// Niveau de zoom google map de 0 à 16, 0 est le plus éloigné
CONST_ZOOMLVL  = 5;

// -- Paramètres des marqueurs -----------------------------
// Couleur par défaut des marqueurs
MarkerColor      = "rgb(127, 127, 127)";
SelectedColor = "rgb(225, 225, 225)";
StaleColor = "rgb(190, 190, 190)";

// -- Configuration du site de réception -------------------
// true affiche une cible aux coordonnées indiquées
// Pour moi : les coordonnées de l'antenne
SiteShow    = true; // true ou false
// Latitude et Longitude en format décimal
SiteLat     = 46.799107;
SiteLon     = 4.418899;
```

```
// -- Dessin des cercles donnant la distance des avions ------
// Les cercles ne sont affichés que si SiteShow est sur true
SiteCircles = true; // true ou false
// En miles nautiques ou en km (selon la valeur de 'Metric')
// J'ai choisi d'afficher des cercles à 50km, 100km, 150km et 200km
SiteCirclesDistances = new Array(50,100,150,200);
```

Protéger l'installation

A défaut de trouver une ancienne boîte de glace à la framboise qui aurait harmonieusement complété l'ensemble, j'ai "investi" dans une boîte étanche achetée un peu plus de 2€ dans une grande surface. Les trous percés pour laisser passer les câbles ont été colmatés avec du silicone utilisé en plomberie. La fixation des différents éléments a été faite avec un adhésif double face prévu pour une utilisation à l'extérieur (**Fig.4**).

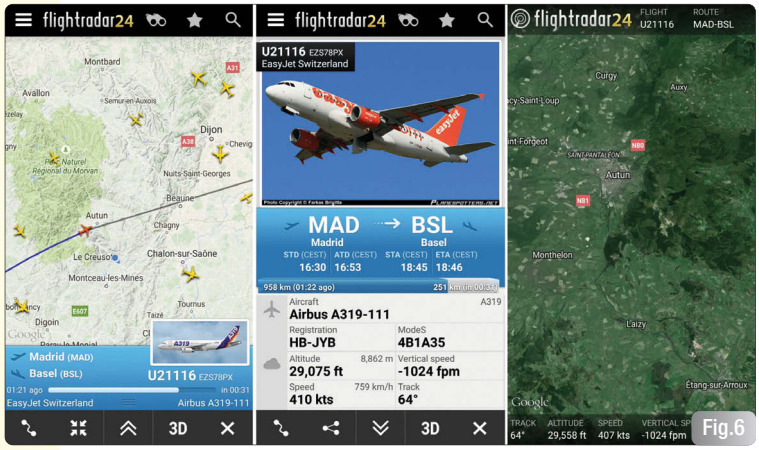
Envoyer des données à Flightradar24

Pourquoi envoyer les données à FlightRadar24 (ou à un autre service de radar virtuel ?) : pour partager les informations recueillies. Elles seront affichées sur le site de FlightRadar24. Vous bénéficierez aussi de l'option premium qui vous permet d'accéder à des données supplémentaires **Fig.6**. Sur la capture d'écran de smartphone (photo 6) vous voyez de gauche à droite : la carte avec les avions en vol, un des avions est sélectionné et sa trajectoire est affichée. Le type de l'avion, sa destination et un certain nombre de paramètres. Par exemple ici, un Airbus A319 de EasyJet va se poser à Bâle et il est en phase de descente (-1024 fpm = - 312 m/minute). Le paysage tel que peut le voir un passager de l'avion (s'il n'y a pas de nuages...).
Pour fournir vos données à FlightRadar24, vous devez envoyer un flux de données compatible TCP30003. Lancez cette fois le logiciel dump1090 avec les paramètres suivants :

```
./dump1090 --quiet --net --net-sbs-port 30003 --enable-agc &
```

Il faut maintenant installer le logiciel de FlightRadar24 pour transférer les données :

Capture d'écran dump1090 Fig.5



Captures d'écran de l'appli FlightRadar24 Fig.6

```
pi@raspberrypi ~ $ mkdir fr24
pi@raspberrypi ~ $ cd fr24
pi@raspberrypi ~/fr24 $ wget https://dl.dropboxusercontent.com/u/66906/fr24feed_arm-rpi_242.tgz
--2015-06-03 20:24:39--  https://dl.dropboxusercontent.com/u/66906/fr24feed_arm-rpi_242.tgz
Résolution de dl.dropboxusercontent.com (dl.dropboxusercontent.com)... 174.129.251.235, 184.73.
223.97, 174.129.255.54, ...
Connexion vers dl.dropboxusercontent.com
(dl.dropboxusercontent.com)|174.129.251.235|:443...connecté.
requête HTTP transmise, en attente de la réponse...200 OK
Longueur: 276631 (270K) [application/x-gtar]
Sauvegarde en : «fr24feed_arm-rpi_242.tgz»
100%[==============================>] 276 631     441K/s   ds 0,6s
2015-06-03 20:24:44 (441 KB/s) - «fr24feed_arm-rpi_242.tgz» sauvegardé [276631/276631]
pi@raspberrypi ~/fr24 $ tar zxvf fr24feed_arm-rpi_242.tgz
fr24feed_arm-rpi_242
```

Le transfert de données vers FlightRadar24 nécessite une clé qui vous sera délivrée après votre inscription :

```
pi@raspberrypi ~/fr24 $ ./fr24feed_arm-rpi_242 –signup
```

Remplissez le questionnaire. Préparez vous à fournir les coordonnées décimales de l'endroit où vous vous trouvez. Une fois que vous aurez reçu la clé (la mienne est remplacée par des **x** ;)), vous pouvez partager vos données :

```
pi@raspberrypi ~/fr24 $ ./fr24feed_arm-rpi_242 –fr24key=xxxxxxxxxxxxxxx –bs-ip=127.0.
0.1 –bs-port=30003
[i]FR24Feed v242 - built on Aug 15 2014/08:32:38
[i]Downloading configuration...OK
[i]Parsing configuration...OK
[c]Interval: 5s
[c]Latitude: 46.8000
[c]Longitude: 4.4200
[c]GND: YES
[c]NonADSB: YES
[i][stats]Stats thread started
[n]defined 1 server
[i]Source defined as Basestation
[n]LFLH1@83.140.21.66:8099/UDP
[n]connecting
[b]connecting to 127.0.0.1:30003
[b]connected
[b]working
[n]connected
[n]switching to UDP
[n]working
[i]Data feed time difference OK abs(20:32:39.080 - 20:32:39.042)=0
[i]sent 1 planes in 1 packets
[i]sent 1 planes in 1 packets
[i]sent 1 planes in 1 packets
[n]pinging the server
[i]sent 1 planes in 1 packets
```

Après une phase d'initialisation votre système est opérationnel et commence à remonter des informations vers FlightRadar24 comme l'indiquent les dernières lignes. Vous bénéficiez de l'option premium à partir de cet instant et elle sera maintenue quelques jours si vous cessez d'envoyer des données au serveur. Si vous souhaitez que votre Raspberry Pi redémarre automatiquement sur dump1090 après sa mise sous tension, il suffira d'écrire un script de lancement de dump1090 et de fr24 et de le faire exécuter lors du démarrage de Raspbian.

Améliorer le système

L'antenne et son câble

L'antenne est un élément clé de ce genre de récepteur. Le petit morceau d'antenne livré avec le dongle TNT permet certes de recevoir quelques avions. Un proverbe radioamateur dit :"Si tu achètes un récepteur à 1000€, achète une antenne à 1000€ !". En effet il ne sert à rien de déployer des trésors de technologie si votre récepteur ne reçoit rien. Le remplacement de cette antenne par une "vraie" antenne 1090 MHz augmentera considérablement le signal reçu et donc la distance à laquelle les avions seront détectés. Ceci explique la présence d'une prise de type "N" sur le câble d'antenne de mon installation (à gauche sur la photo 4). Ce modèle de prise est utilisé sur les antennes professionnelles car il procure des pertes relativement faibles. J'ai opté pour une antenne de type FR24 équipée d'une prise N, le boîtier étant monté sur le mat d'antenne. Ceci élimine pratiquement le problème des pertes dans le câble de liaison entre l'antenne et le récepteur. Si vous choisissez de relier l'antenne au dongle TNT par un câble coaxial, ne lésinez pas sur sa qualité (et son prix) au risque d'être déçu par les résultats ! **Fig.7.**

Antenne FR24 (© FLightRadar24)

Le dégagement

Un autre paramètre important dans la réception des signaux aux fréquences de l'ADS-B est le dégagement de l'antenne. La portée étant principalement à vue, l'antenne doit être bien dégagée. Si vous êtes encaissé entre des immeubles ou situé au fond d'une vallée, la portée sera fortement réduite. Dans certains cas, le montage de l'antenne sur un mât de plusieurs mètres assure un dégagement suffisant pour améliorer la réception. Il faudra juger de la nécessité de surélever l'antenne en fonction votre situation.

Préamplifier le signal

Dans l'arsenal des moyens disponibles pour améliorer la réception, il y a encore la possibilité d'intercaler un préamplificateur de signal entre l'antenne et le récepteur. Ce préamplificateur devra être situé au plus près de l'antenne et être muni d'un filtre pour éliminer les signaux non souhaités susceptibles de provoquer des interférences et de dégrader la réception. C'est une amélioration que je compte tester prochainement.

Conclusion

Au travers de cette description, vous avez pu vous apercevoir que le Raspberry Pi est extrêmement souple et qu'il peut s'adapter à de nombreux projets. Dans le domaine de la réception des signaux ADS-B c'est un auxiliaire précieux qui permet de traiter les signaux reçus 24 heures sur 24, sans monopoliser un PC. Avec un matériel minimum (un Raspberry Pi et son alimentation, une carte SD, une clé USB TNT et son antenne) il est déjà possible d'obtenir des résultats intéressants. J'espère avoir donné envie à certains d'entre vous de se lancer à la chasse aux signaux émis par les avions qui nous survolent en permanence.

Constellation : interconnectez vos objets et services connectés et créez de l'intelligence ambiante !

Autour de nous, de plus en plus d'objets et services sont connectés : bracelets et montres connectés, balances connectées, lampes connectées, TV connectées, thermostats connectés, stations météo connectées, etc. Malheureusement, aucun ne communique avec les autres. C'est ce qui rend difficile la création d'intelligence dite « ambiante ». La plateforme Constellation vise à changer la donne en simplifiant la vie du développeur !

Sébastien Warin
Creative Technologist @ Publicis ETO
http://sebastien.warin.fr

Fig.1

Lorsque j'ai emménagé dans ma nouvelle maison, j'en ai profité pour revoir l'ensemble de mon installation domotique. Plutôt que tout miser sur une seule technologie, j'ai pris le meilleur dans chaque domaine : du thermostat Nest à la domotique Z-Wave pilotée par une Vera, station météo NetAtmo, lampes Hue pour le salon, système de sécurité Paradox, etc. Cela passe aussi par des technologies maison, en bon maker, comme mon système Ambilight nommé « S-light », mon miroir interactif « The Mirror », ou plus récemment mon système de monitoring des ressources énergétiques « S-Energy ».
Mais une maison intelligente ce n'est pas juste avoir un tas d'objets connectés ! Devoir sortir son smartphone ou sa tablette pour allumer des lampes ou monter le chauffage n'a rien d'intelligent !
L'intelligence vient du fait que la « maison » comprend ce que vous faites et pilote l'ensemble des équipements, peu importe comment elle le fait !
Par exemple lorsque je lance un film sur la TV du salon, la maison adapte automatiquement volets et lumières sans avoir à le lui demander ! Idem lorsque je me couche, elle éteint pour moi les lumières, TV, amplis, ferme les volets, baisse le chauffage, etc.
Elle m'indique vocalement si une fenêtre est ouverte lorsqu'il commence à pleuvoir ou m'indique si je consomme trop d'eau lorsque je suis dans la douche !
Elle allume la lumière de mon bureau lorsque mon PC est déverrouillé et, à l'inverse, éteint la lumière lorsque je le verrouille !
Tout cela est relativement simple à mettre en œuvre à partir du moment où les informations circulent, rendant possible la création d'intelligence ambiante.

La plateforme Constellation

Pour connecter tout cela, on a deux approches :

- Créer un « super programme » qui communique avec tous nos objets et services dans une approche *monolithique*,
- Créer une architecture distribuée dans laquelle on peut brancher/débrancher de nouvelles fonctionnalités sans devoir tout arrêter.

Constellation est basée sur cette deuxième approche. Même si le module chauffage « Nest » plante, cela n'empêche en rien le fonctionnement des autres modules. De plus vous pouvez mettre à jour les composants de votre « environnement » sans devoir tout « relancer » !
Dans la Constellation, tout est « package ». Une constellation est un amas de packages, qu'on peut voir comme des services/applications ou des objets.
Chaque package est un processus qui s'exécute sur une machine et qui communique à travers un bus.
L'une des trois missions de la Constellation est de simplifier le développement, le déploiement et l'administration des services et des programmes de votre Constellation.
Pour cela, l'idée est d'installer une sentinelle sur vos machines Windows ou Linux.
Chaque sentinelle communique avec le serveur Constellation.
Grâce au SDK Constellation intégré à Visual Studio, nous pouvons développer des packages en .NET ou en Python, et les déployer directement sur votre Constellation dans un catalogue de packages. Il suffit ensuite de lier les packages aux sentinelles. Vous pouvez ensuite contrôler chaque processus, l'arrêter, le redémarrer, mettre à jour les packages, etc.
Que ce soit à partir du Control Center ou depuis les API Constellation.
L'ensemble des logs produits par les packages sont remontés au serveur en temps réel.

De plus, toutes les configurations des packages sont centralisées sur le serveur. Il n'y a donc plus besoin de se connecter en RDP ou SSH sur chacune de vos machines ; vous installez une bonne fois pour toute une sentinelle et Constellation prend la main !
Une fois les packages déployés et orchestrés par la Constellation, ils sont tous connectés dans la Constellation et peuvent s'échanger des messages. En .NET, le fait d'ajouter l'attribut « MessageCallback » suffit à exposer votre méthode dans la Constellation.
Ainsi chaque package peut envoyer et recevoir des messages (c'est-à-dire invoquer des méthodes) avec les autres packages mais également avec des pages HTML/Javascript connectées à votre Constellation, ou encore avec n'importe quel système grâce à une interface HTTP/REST, comme une Arduino ou même un script Powershell. Une page Web qui pilote une application Python ou une Arduino qui contrôle une lampe Hue ou un thermostat Nest, devient un jeu d'enfant à concevoir ! De plus Constellation apporte un tas de fonctionnalités autour du hub de messaging comme l'auto-description des contrats, la notion de scope, de saga, etc.
Enfin, chaque package dans la Constellation ou système connecté sur l'interface REST de Constellation, peut produire des StateObjects. Un StateObject représente l'état d'un objet, d'une application ou d'un service partagé dans la Constellation.
Par exemple, le volume de mon ampli, la chaîne de la TV, l'état d'une lampe, la T° du jardin, mon agenda Exchange ou le statut de mon alarme (...) sont tous des StateObjects auquels les packages, des pages Web, ou de n'importe quoi connecté à la Constellation, peuvent y accéder ou s'y abonner pour créer des interfaces type dashboard temps réel, ou exécuter des actions en réponse à certaines conditions.
La plateforme Constellation est actuellement en beta-privée, n'hésitez pas à faire un tour sur le site officiel http://www.myconstellation.io pour plus d'information et vous inscrire au programme de test.

Connecter ses volets dans la Constellation avec des Arduino et un Raspberry

Lorsque nous avons acheté notre maison, nous avons entamé de lourds travaux de rénovation ce qui m'a permis de passer du câble multifonction blindé à chaque point de contrôle (interrupteurs) dans l'optique de pouvoir facilement ajouter des fonctionnalités domotiques sans avoir recours à des solutions sans fil comme le Z-Wave assez couteux.

Lucas Dupuis
lucas.dupuis@gmail.com

Après l'installation de pas moins d'une dizaine de volets électriques dans la maison, il me fallait une solution de pilotage à la fois simple, fiable et surtout économique. De plus cette solution devra être ouverte, me permettant de concevoir ma propre interface de contrôle sur tout type de support, et de pouvoir facilement ajouter de l'intelligence. Grâce à la plateforme Constellation, un Raspberry Pi et quelques Arduino, j'ai pu concevoir mon propre système de gestion des volets.

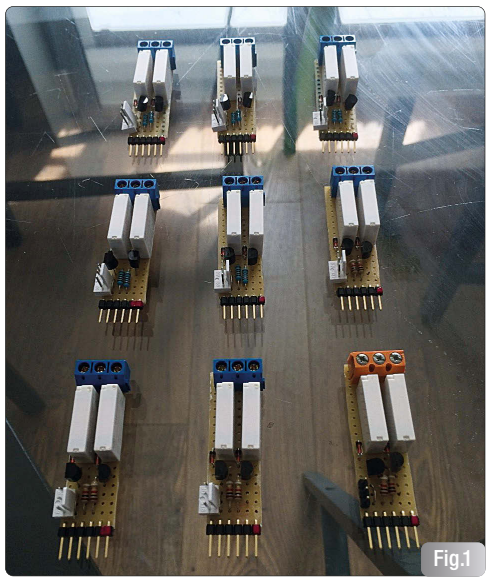


Fig.1

La conception

Concrètement chaque volet dispose de deux phases : une pour la montée et une autre pour la descente.
L'idée est donc de piloter ces deux phases avec des relais. Pour garder le contrôle manuel des volets (c'est-à-dire à partir des interrupteurs encastrés dans le mur), j'ai connecté chaque interrupteur bistable au système pour déclencher les volets (obligatoire pour une certification WAF).
Pour cela derrière chaque interrupteur des volets, j'ai placé une carte de contrôle avec deux relais 220V et deux entrées. Chaque carte de contrôle est connectée sur un câble multifonction qui arrive à chaque volet. On utilise donc 6 fils : deux pour les entrées des interrupteurs (montée et descente), deux pour la commande des deux relais (montée et descente) et deux pour l'alimentation (5V ou 12V en fonction de vos relais).
Côté budget, comptez moins de 3€ par volet Fig.1.
L'ensemble des câbles se rejoignent tous jusque dans ma baie réseau au RDC.
Chaque volet a donc besoin de deux sorties pour les relais et deux entrées pour l'interrupteur bistable. Avec 10 volets, ça nous donne donc 20 sorties et 20 entrées, beaucoup trop pour un Raspberry !
Toujours dans l'optique de baisser les couts, j'utilise un réseau d'Arduino Mini Pro acheté moins de 2€ pièce.
Chaque Arduino peut gérer jusqu'à 3 volets. Ils sont tous installés sur une carte et connectés en série sur un bus I2C sur lequel est également connecté un Raspberry qui sera le cerveau des volets, lui-

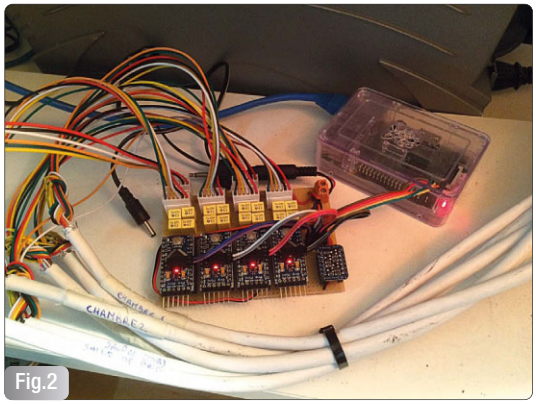


Fig.2

même connecté dans la Constellation Fig.2.
N'importe quel objet, page Web ou programme connecté dans la Constellation pourra envoyer un ordre à chaque volet. Chaque ordre sera traité par le package Constellation qui tourne sur le Raspberry qui commandera l'Arduino associé via le bus I2C qui lui-même activera le relais correspondant à l'ordre de montée ou descente du volet cible !

Conception logicielle

Une fois nos volets connectés sur les cartes relais reliées aux Arduinos, il suffit d'activer les sorties pour basculer l'état des relais et donc piloter la montée ou la descente de chacun de nos volets.
Par exemple pour une montée :

```
    digitalWrite(volets[i].relaisMontee, HIGH);
    digitalWrite(volets[i].relaisDescente, LOW);
```

Pour capter un changement d'état sur les interrupteurs bistables, il suffit de lire l'entrée correspondante configurée en pull-up. Si l'entrée change d'état à l'état LOW, c'est qu'on vient d'appuyer sur le bouton de montée :

```
if (digitalRead(volets[i].buttonMontee) == LOW && volets[i].previousSens != 1)
  {
    // Demande de montée
  }
```

Pour la communication des Arduino avec la base Raspberry, j'utilise le protocole I2C. Chaque Arduino à son adresse 0x11 pour le 1er, 0x12 pour le 2ème, etc. en esclave sur le bus. Le Raspberry est maître.
Lors du setup, j'attribue l'adresse de mon Arduino et configure des callbacks pour la réception et l'émission :

```
Wire.begin(SLAVE_ADDRESS);
Wire.onReceive(receiveData);
Wire.onRequest(sendData);
```

Pour la lecture, dès qu'il y a quelque chose sur le bus, je récupère un Int dont les deux 1ers bits m'indiquent le n° du volet, suivi d'un bit pour savoir si je dois allumer ou éteindre le volet. En cas d'allumage, le dernier bit permet de connaître le sens (montée ou descente) :

```
while(Wire.available())
 {
  int dataReceived = 0;
  dataReceived = Wire.read();
  rawVal += dataReceived;
  }
 int rawInt = rawVal.toInt();
 int numVolet = bitRead(rawInt,2) & bitRead(rawInt,3)*2;
 int allumageVolet = bitRead(rawInt,1);
 int sensVolet = bitRead(rawInt,0);
// Envoyer l'ordre sur les sorties du volet correspondant
```

Pour connaître l'état des interrupteurs depuis le Raspberry, chaque Arduino écrit sur le bus un Int avec le même format pour indiquer pour chaque volet son état :

```
Wire.write(/* int représentant l'état du volet */);
```

Maintenant que je peux piloter l'ensemble de mes volets depuis mon Raspberry, j'utilise la plateforme Constellation pour tout connecter facilement.
Grâce au SDK Constellation dans Visual Studio et aux Python Tools for Visual Studio, j'ai créé un package Constellation en Python : Fig.3.

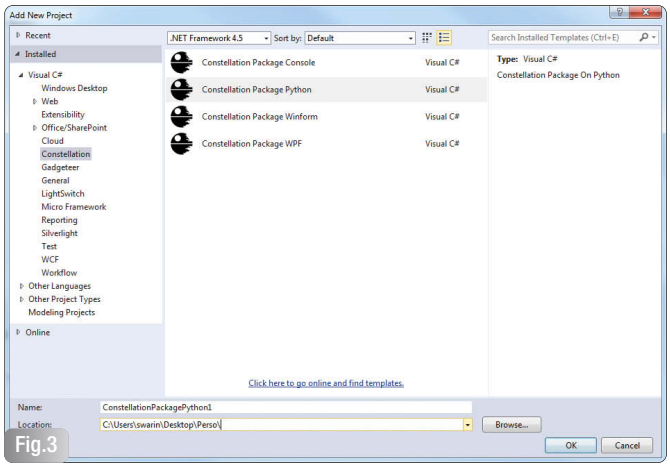


Fig.3

Le package expose des méthodes dans la Constellation pour le contrôle et publie des « StateObjects » sur l'état de chaque volet.

```
def ActionVoletMessageCallback(data):
  # Action à envoyer à un volet.
  try:
    volet = GetShutter(data["shutterName"].lower())
    actionStr = data["actionShutter"].lower()
            if actionStr == "montee":
      volet.Open()
    elif actionStr == "descente":
      volet.Close()
    else:
      volet.Stop()
  Constellation.WriteInfo("ActionVolet: %s => %s ", (volet.nom, actionStr))
```

Les methodes Open/Close/Stop écrivent sur le bus I2C avec la commande I2CSet à destination de l'adresse de l'Arduino cible, l'entier représentant l'ordre à exécuter :

```
commande = 'i2cset -y 1 %s 0x%x' % (self.adresse, i)
os.system(commande)
```

Pour la lecture sur le bus, j'utilise la commande « i2cget ».

```
f = os.popen('i2cget -y 1 %s' % self.adresse, 'r')
retour = f.read()
```

Une fois l'entier représentant l'état de chaque volet décrypté, je publie dans la Constellation un StateObject par volet avec toutes les infos :

```
# Changement, on met à jour le stateobject
 Constellation.PushStateObject(volet.nom, volet.etat, "Shutter",
  {
    "Adresse": volet.adresse,
    "Numero": volet.numero,
    "Statut inversé" : volet.inverseStatut,
    "Commande inversée" : volet.inverseCommande
  })
```

Maintenant mon package développé, je peux directement le publier depuis Visual Studio sur mon serveur et le déployer sur une sentinelle de ma Constellation. Au préalable j'ai installé une Sentinelle Constellation sur mon Raspberry. Il me suffit ensuite de lancer mon package depuis l'interface Web de contrôle : Fig.4. Cette interface me permet également de suivre en temps réel les logs de mes packages : Fig.5.

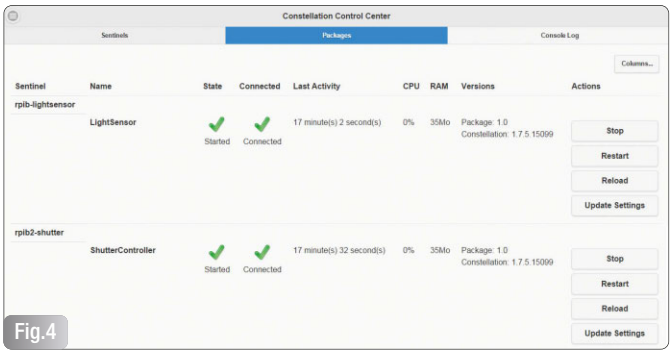


Fig.4

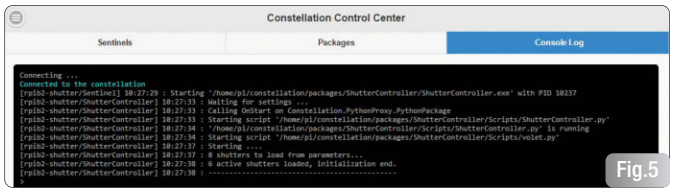


Fig.5

Aller plus loin dans la Constellation

A ce stade l'état de chaque volet de ma maison est publié dans la Constellation sous forme d'un StateObject et je suis capable de piloter chacun d'entre eux en envoyant un message à destination de mon package Python. La suite logique est de créer une interface de pilotage. Pour cela, je vais utiliser l'API Constellation pour AngularJS me permettant de créer une application HTML/JS connectée dans la Constellation avec le framework Angular de Google.
Le code Javascript se résume à cela :

```
app.controller('ShutterAngularController', ['$scope', 'constellation', function ($scope, constellation) {
  constellation.intializeClient("http://constellation.myhome.lan:8088/", "MonAcessKey",
"HomeControl");
  var MesVolets = {};
  constellation.onUpdateStateObject(function (message) {
    $scope.$apply(function () {
      $scope.MesVolets[message.Name] = message;
    });
  });
  constellation.onConnectionStateChanged(function (change) {
    if (change.newState === $.signalR.connectionState.connected) {
      constellation.requestSubscribeStateObjects("*", "ShutterController", "*", "*");
    }
  });
  $scope.actionShutter = function (shutter, action) {
    constellation.sendMessage({ Scope: 'Package', Args: ['ShutterController'] }, 'ActionVolet
MessageCallback', { 'shutterName': shutter.Name, 'actionShutter': action });
  };
  constellation.connect();
}]);
```

Dans mon contrôleur, je me connecte à ma Constellation et je m'abonne aux StateObjects du package « ShutterController ». A chaque réception d'un state object, je l'ajoute dans mon scope Angular. De plus j'ai ajouté une méthode pour envoyer un message de contrôle à mon package Python.
Il ne reste plus qu'à faire une vue HTML en faisant une boucle sur notre variable de scope « MesVolets » :

```
<div class="list-group-item" ng-repeat="shutter in ShutterController">
  <h3>
    {{shutter.Name}}
    <em class="pull-right btn-group" role="group" aria-label="...">
<button type="button"
class="btn btn-default glyphicon glyphicon-chevron-up"
ng-class="{ active: shutter.Value==='Montee' }"
ng-click='actionShutter(shutter, "montee")'>
</button>
<button type="button"
class="btn btn-default glyphicon glyphicon-stop"
ng-class="{ active: shutter.Value==='Arret' }"
ng-click='actionShutter(shutter, "stop")'>
</button>
          <button type="button"
class="btn btn-default glyphicon glyphicon-chevron-down"
ng-class="{ active: shutter.Value==='Descente' }"
ng-click='actionShutter(shutter, "descente")'>
</button>
    </em>
  </h3>
</div>
```

Et voilà comment créer une première interface Web de supervision et de contrôle de l'ensemble des volets à la maison.
Les propriétés des volets étant exposées dans la Constellation, il devient facile d'ajouter de l'intelligence dans la maison.
Par exemple, j'ai également un autre Raspberry avec un capteur de luminosité TSL2561 connecté dans ma Constellation, ce qui me permet d'ouvrir et fermer les volets en fonction du seuil de luminosité ou encore de fermer automatiquement les volets lorsque je lance un film sur mon XBMC (XBMC/Kodi étant lui-même connecté dans ma Constellation).
Grace à la plateforme Constellation, l'interconnexion des objets devient un jeu d'enfant comme l'orchestration et le déploiement de mes programmes à travers les différents devices de la maison.
Ainsi, moyennant un budget inférieur à 100€, j'ai pu connecter et rendre intelligents mes dix volets.

Liste du matériel :

Carte de relais par volet :
- 2x relais 250V/5A (HF49FD)
- 2x transistors BC547 (ou équivalent)
- 2x diodes 1N4007 (ou équivalent)
- 2 resistances 1kOhm
- 1 plaque epoxy, un bornier pour les connections et de la gaine thermorétractable pour isoler la carte dans la boiter d'encastrement

Carte de contrôle :
- 1x Arduino Mini Pro pour 3 volets, soit 4 dans le projet
- 2x condensateurs 1uF par volet (pour absorber les interférences des entrées), soit 16 dans le projet
- 1x convertisseur de niveau logique pour le bus I2C entre les RPi et les Arduino
- 1 plaque epoxy, des borniers et connecteur JST pour les connections, une alimentation 12V
- Un Raspberry avec la sentinelle Constellation
- Un serveur Constellation

Créer son interface de pilotage domotique avancée avec Constellation et AngularJS

La plateforme Constellation permet d'interconnecter des objets et services facilement. Il devient alors possible de créer de l'intelligence ambiante. Par exemple fermer les volets, lumières et thermostats en allant se coucher, tamiser les lumières lorsque je lance un film sur mon media center, etc.

Sébastien Warin
Creative Technologist @ Publicis ETO
http://sebastien.warin.fr

Cela est possible car chaque objet ou service est connecté dans la Constellation pour exposer des méthodes et publier l'état de ses équipements sous forme de StateObject. Grâce à cette notion de StateObject et l'API Constellation pour AngularJS, il devient facile de créer une interface Web de pilotage de ses différents objets ou services connectés, même lorsqu'on ne connaît rien au Javascript. Cela m'a permis de créer une application de supervision et de contrôle de l'ensemble de mes objets et services avec l'interface que je souhaitais ! Au lieu de me retrouver avec 50 applications, je n'en ai plus qu'une seule qui me centralise à la fois ma domotique Z-Wave, thermostat Nest, lampes Hue, mes objets DIY comme S-Light, S-Energy, S-Sound, mais également mes XBMC/Kodi, ma TV, mon ampli Pioneer, les indicateurs hardwares de mes serveurs comme le CPU, la RAM, et j'en passe.

Connecter une page Web dans la Constellation

Pour cela nous avons juste à créer une simple page HTML et à ajouter la librairie « Constellation pour AngularJS ». Dans une balise script de ma page HTML, je vais déclarer mon contrôleur AngularJS et le connecter à ma Constellation en indiquant l'URL et l'AccessKey pour la connexion :

```
var panelApp = angular.module('panelApp', ['ngConstellation'])
  .controller('PanelController', ['$scope', 'constellation', function ($scope, constellation) {
    // Initialisation
    constellation.intializeClient("http://skynet-server.ajsinfo.lan:8888/constellation/",
"monAccesKey", "SPanel");
    // A la reception d'un StateObject, ajouter le SO au scope courant
    constellation.onUpdateStateObject(function (message) {
      $scope.$apply(function () {
        if (typeof $scope[message.PackageName] === 'undefined' || $scope[message
.PackageName] === null) {
          $scope[message.PackageName] = {};
        }
        $scope[message.PackageName][message.Name] = message;
      });
    });
    // Connexion à la Constellation
    constellation.connect();
  }]);
```

A chaque mise à jour d'un StateObject, il sera ajouté au scope courant me permettant de l'exploiter dans ma vue HTML. Pour finir, lorsque que nous sommes connectés à la Constellation il faudra s'abonner aux StateObjects que l'on souhaite exploiter. Dans le cas de mon projet, j'ai besoin des SO de mon projet S-Energy (compteur d'eau, gaz et électricité), des capteurs NetAtmo, de ma domotique Z-wave (box Vera), de mon thermostat Nest, les media center XBMC, mon ampli Pioneer, l'état de mon onduleur APC, mon système d'alarme Paradox, les temps de réponse de mes sites Internet poussés par un script Powershell, les informations du jour (comme la fête du jour), la luminosité ambiante (poussée par un capteur TLS), les prévisions de Meteo France, mon agenda Exchange, le contrôle du son dans la cuisine et salle de bain (projet S-Sound) ainsi que le infos hardware de mon serveur CEREBRUM. Toutes ces informations sont dans la Constellation sous forme de StateObject car j'ai activé les packages (= connecteurs) pour chacun de ces objets ou services dans ma Constellation. Pour m'abonner, il suffit d'ajouter dans mon contrôleur les lignes suivantes :

```
constellation.onConnectionStateChanged(function (change) {
  if (change.newState === $.signalR.connectionState.connected) {
    constellation.requestSubscribeStateObjects("*", "SEnergy", "*", "*");
    constellation.requestSubscribeStateObjects("*", "NetAtmo", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Vera", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Nest", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Xbmc", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Pioneer", "*", "*");
    constellation.requestSubscribeStateObjects("*", "BatteryChecker", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Paradox", "*", "*");
    constellation.requestSubscribeStateObjects("Powershell", "Internet", "*", "*");
    constellation.requestSubscribeStateObjects("*", "DayInfo", "*", "*");
    constellation.requestSubscribeStateObjects("*", "LightSensor", "*", "*");
    constellation.requestSubscribeStateObjects("*", "MeteoFrance", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Exchange", "*", "*");
    constellation.requestSubscribeStateObjects("*", "SSound-Cuisine", "*", "*");
    constellation.requestSubscribeStateObjects("*", "SSound-SalleDeBain", "*", "*");
    constellation.requestSubscribeStateObjects("CEREBRUM", "HWMonitor", "*", "*");
  }
});
```

La méthode requestSubscribeStateObjects permet de récupérer la valeur du dernier StateObject connu dans la Constellation et de s'abonner en temps réel aux mises à jour. La signature permet de spécifier quelle est la sentinelle source, le package source, le nom du StateObject et son type. Le wildcard « * » permet de tout récupérer. Côté Javascript c'est fini !

Affichez vos StateObjects

Pour faire une belle interface, j'utilise la librairie bootstrap que j'installe également via NuGet. Côté HTML, si je veux afficher ma consommation instantanée d'électricité poussée par le package S-Energy, il me suffit d'ajouter dans le code HTML :

```
{{ SEnergy.Electricity.Value.WattPerHour }}
```

Si je veux la température de consigne de mon Nest, je peux écrire :

```
{{Nest.LivingRoom.Value.target_temperature_c}}°C
```

Si maintenant je veux afficher l'état d'armement de mon alarme avec une icône associée, je peux écrire le template suivant :

```
<li aria-hidden="true" class="fa fa-{{ (Paradox.AreaInfo1.Value.IsFullArmed || Paradox.AreaInfo1.Value.IsStayArmed) ? 'bell' : 'bell-slash' }}"></li>
<bold ng-if="Paradox.AreaInfo1.Value.IsFullArmed || Paradox.AreaInfo1.Value.IsStayArmed"><bad>ARMÉE</bad></bold>
<bold ng-if="!Paradox.AreaInfo1.Value.IsFullArmed && !Paradox.AreaInfo1.Value.IsStayArmed">OFF</bold>
```

Ici, si la propriété « IsFullArmed » ou « IsStayArmed » mon StateObject « AreaInfo1 » du package « Paradox » est vraie, j'ai l'icône « bell » suivie du texte « ARMEE » ; autrement on aura une icône « bell-slash » suivie de « OFF ». La balise « ng-if » permettant d'ajouter des conditions sur nos balises HTML Fig.1. Pour découvrir l'ensemble des StateObjects de votre Constellation, vous avez dans le « Control Center » un « State Objects Explorer » vous permettant d'introspecter chaque StateObjects. Vous pouvez également faire des itérations sur vos StateObjects dans votre vue HTML grâce à la balise « ng-Repeat ». Par exemple mon package Exchange publie dans un StateObject mon agenda sur les 5 jours à venir, pour les afficher :

```
<table>
  <tr ng-repeat="appointement in Exchange.Appointments4sebastien.Value | limitTo: 5">
    <td>{{appointement.Start.toLocaleDateString ("fr-fr", { weekday: "long", month: "short", day: "numeric" })}} à {{appointement.Start.toLocaleTimeString ("fr-fr", { minute: "numeric", hour: "numeric" })}} : {{appointement.Subject}} {{appointement.Location}}</td>
  </tr>
</table>
```

Fig.2.

Ajouter des composants Angular

Pour rendre la lecture de notre dashboard plus agréable on peut également ajouter des composants à notre module Angular. Pour ma part j'ai ajouté le module Bootstrap-UI pour ces progressbars, Bootstrap-slider pour les sliders, ngPercentDisplay pour affichage circulaire des pourcentages, ou encore AngularChart pour les graphiques. Par exemple mon package « HWMonitor » sur la sentinelle « CEREBRUM » publie différent SO sur des indicateurs hardware. En utilisant ngPercentDisplay, il me reste plus qu'à le lier au StateObject souhaité, ici la consommation du CPU :

```
<percent-display percent="HWMonitor['/intelcpu/0/load/0'].Value.Value" side="80" colors="#383737 #b2c831"></percent-display>
```

Le résultat : Fig.3. Même principe avec le composant "progressbar" pour afficher l'état de mes disques :

```
<progressbar value="HWMonitor['/hdd/1/load/0'].Value.Value" ><b>{{HWMonitor['/hdd/1/load/0'].Value.Value}}%</b></progressbar>
```

Fig.4.

Commander vos appareils

Maintenant que notre vue HTML nous affiche toutes les informations que nous souhaitons, nous allons ajouter des commandes.
Pour cela il suffit d'envoyer un message dans la Constellation au package cible. Par exemple pour contrôler notre thermostat Nest, il faut envoyer le message « SetTargetTemperature » au package Nest en précisant en argument la valeur de la T° de consigne. Sans écrire de Javascript, il me suffit d'ajouter deux liens avec une balise ng-Click pour envoyer ma T° de consigne à mon Nest avec l'API Constellation :

```
<a href="" ng-click="constellation.sendMessage({ Scope: 'Package', Args: [ 'Nest' ] }, 'SetTargetTemperature', Nest['Living Room'].Value.target_temperature_c ) + 0.5)"><img src="imgs/up-small.png" alt=""> UP</a> |
<a href="" ng-click="constellation.sendMessage({ Scope: 'Package', Args: [ 'Nest' ] }, 'SetTargetTemperature', Nest['Living Room'].Value.target_temperature_c ) - 0.5)"><img src="imgs/down-small.png" alt=""> DN</a></p>
```

Fig.5.

Autre exemple pour piloter nos volets. Dans mon cas tous sont connectés en Z-Wave grâce à des micromodules FGRM et pilotés via un box Vera elle-même connectée dans la Constellation. Je vais donc, pour chaque volet, afficher un slider me permettant de définir le pourcentage d'ouverture de mon volet. Cela se décrit par le code HTML suivant :

```
<table>
  <tr ng-repeat="switch in Vera" ng-if="(switch.Type == 'VeraNet.WindowCovering')">
    <td>{{switch.Name}}</td>
    <td>
      <slider onstopslide="SetWindowCovering" data-coverId="{{switch.Value.Id}}" ng-model="switch.Value.Level" min="0" step="1" max="100" value="" class="mySlider" id="windowCover{{switch.Value.Id}}" selection="after"></slider>
    </td>
  </tr>
</table>
```

Le tableau HTML affiche autant de ligne (tr) que j'ai de volets (itération sur mes StateObjects du package Vera où le type est un volet).
Pour chaque ligne j'affiche le nom du volet et un slider dont le modèle est lié à la valeur d'ouverture de mon volet (propriété « Level » de mon StateObject). J'ai ajouté la méthode «SetWindowCovering » dans mon scope Angular qui sera déclenchée dès qu'on change la valeur du slider afin d'envoyer l'ordre à mon package Vera :

```
$scope.SetWindowCovering = function (event) {
  constellation.sendMessage({ Scope: 'Package', Args: [ 'Vera' ] }, 'SetDimmableLevel', { DeviceID: $('#' + event.target.parentElement.id).data('coverid'), Level: event.value });
};
```

Et en quelques lignes j'ai le contrôle de mes volets en temps réel sur mon interface Web : Fig.6.

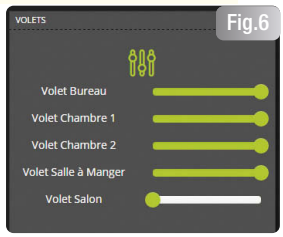


Fig.6

Conclusion

Pour parfaire la solution afin d'éviter d'ouvrir un navigateur, j'ai créé une application Cordova avec un « InAppBrowser » en plein écran qui affiche notre page Web. Cette application est déployée aussi bien sur nos smartphones que sur des tablettes. Afin de retrouver un contrôle immédiat sur la maison, des tablettes ont été installées dans la cuisine et dans la salle de bain pour avoir la main sur l'ensemble des équipements de la maison de façon immédiate. Grâce à la plateforme Constellation, ses nombreuses API et ses connecteurs, interconnecter ses objets ou services et réaliser des interfaces de contrôle est quelque chose de très simple et rapide à faire. De plus avec des frameworks comme AngularJS cela ne nécessite quasiment aucune ligne de code. Que du bonheur !

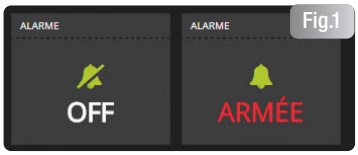
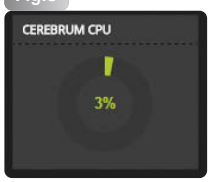


Fig.1

Fig.2

Fig.3



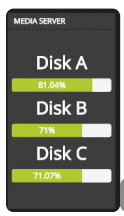
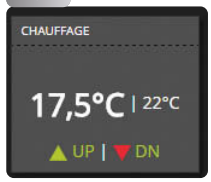


Fig.4

Fig.5



Les 3 objets Programmez! présentés aux TechDays 2015 et Maker Faire Paris 2015

L'idée de départ de ces prototypes est très simple : montrer qu'il est facile et rapide, avec quelques capteurs, un peu d'idées et de codes, de commencer à créer des objets et à démontrer que l'on peut réaliser des projets réels. Enjoy !

François tonic
Maker à Programmez!

Les codes des projets sont disponibles sur le site Web du magazine : www.programmez.com

Pour chaque projet, nous avons testé et codé chaque composant individuellement sur une carte Arduino Uno dédiée aux tests. Puis, nous avons intégré au fur et à mesure chaque composant et code correspondant sur la carte du prototype. Nous avons ainsi pu détecter les problèmes de code et les possibles incompatibilités des librairies et des composants. Pour la programmation, nous utilisons Arduino IDE.

Tube à meuh qui ne fait pas meuh

La veille de Maker Faire Paris, je me suis demandé : bon j'ai un tube à chocolat, quelques composants et une belle NeoPixel de 8 leds multicouleurs que je n'ai pas utilisée ainsi que du Velostat, qu'est-ce que je peux bien faire avec ? Je voulais tout d'abord créer une boîte à meuh avec un capteur d'inclinaison (ou capteur tilt) et un buzzer. Puis j'ai rajouté un capteur de distance (capteur ultrason). Mais finalement, j'ai voulu aussi utiliser le Velostat. Le Velostrat est un film plastique conducteur et sensible à la pression. Et c'est là que l'idée m'est venue de combiner le Velostat et la barre à led NeoPixel d'Adafruit.

1. Préparation du tube : découpe du fond et de la partie supérieure de la vitre plastique. Percement des trous pour fixer les composants et la carte Arduino Uno,
2. Installation de la led sur le couvercle (à la place du bouton en plastique), du capteur tilt et du buzzer, dans la partie haute du tube,
3. Installation du capteur de distance et du NeoPixel,
4. Fixation de l'Arduino et connexion de tous les câbles.

Pour simplifier la connexion, j'utilise une Grove Base Shield (Seeed Studio) :
A0 -> Velostat
D2 -> led (sur socket Grove Led)
D3 -> buzzer
D5 -> NeoPixel
D6 -> capteur de distance
D8 -> capteur tilt

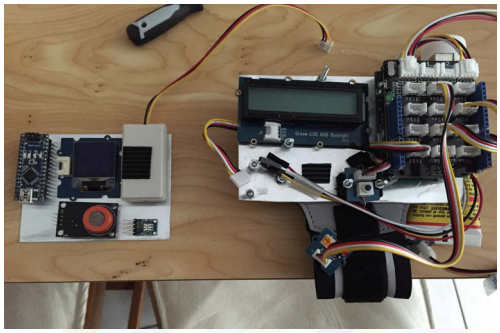
Plusieurs librairies ont été utilisées pour ce projet : Adafruit_NeoPixel (pour contrôler la barre led), pitche (pour le buzzer). Pour simplifier le codage, nous avons utilisé et adapté des exemples pour le buzzer (musique) et le NeoPixel.
Sur le NeoPixel ne pas oublier d'utiliser une résistance de 470 ohms. Sur le Velostat, il faut créer un maillage serré avec du fil métallique sur les deux faces. Pour « alimenter » le Velostat, nous utilisons une pince crocodile. Le tout est branché sur la broche A0 (entrée analogique). Plus le maillage sera de qualité (et bien alimenté), meilleure sera la relecture de la pression. Il suffit d'utiliser analogRead pour lire la valeur du Velostat.
Nous avons eu beaucoup de problèmes avec le capteur tilt. Nous avons finalement opté pour le Grove Tilt, de très bonne qualité.

Liste du matériel

1 AdaFruit NeoPixel Stick Module
1 Grove Socket Led
1 Grove Buzzer
1 carte Arduino Uno
1 Grove Base Shield
1 capteur de distance
1 capteur tilt (capteur d'inclinaison)
1 Velostat
1 pince crocodile
1 résistance 470 ohms (minimum)
1 cylindre
Les fils de connexions

Mini-station médicale « mobile »

On parle beaucoup de santé numérique, de bracelets et montres connectés avec fonctions de santé et sportives. L'idée était donc de préparer un brassard de santé « mobile ». Pour ce faire, nous allons utiliser une carte Arduino Uno pour le prototype montré au Maker Faire. Dans une future itération du projet, nous utiliserons une Arduino Nano (ou Mini) : gain de place de 50 % ! Tous les composants fonctionnent parfaitement, ainsi que le code avec la Nano. Nous disposons d'un capteur de fréquence cardiaque avec clip oreille, capteur de température, capteur de gaz (pour l'alcool). L'affichage de données se fait sur un écran LCD 2 lignes. Un bouton poussoir a été installé. Là encore, peu de difficulté pour ce projet. Pour le capteur de gaz, il ne faut pas se tromper, car la gamme MQ (capteurs chimiques – électroniques) est très large. Le modèle MQ-3 permet de détecter la présence d'alcool par l'haleine et dans son environnement immédiat (l'air). Notre modèle ne nécessite pas de temps de calibration (certains capteurs de gaz l'exigent). Le capteur cardiaque prend une mesure de pulsation cardiaque par seconde, il faut 20 mesures pour obtenir le nombre du pouls. Pour la température, nous utilisons un capteur TMP36, fiable et peu cher.
Comme nous étions limités à 2 lignes sur le LCD, nous affichons en priorité la température et la fréquence cardiaque. Pour la présence d'alcool, le bouton poussoir permet d'afficher le message correspondant. Pour être autonomes, nous avons utilisé une pile 9V branchée à la Uno (attention à la consommation).
Après il possible d'aller bien plus loin avec une connexion sans fil et un composant de type gyroscope ou accéléromètre, qui en cas de chute rapide de la personne portant le



Liste du matériel

1 brassard de type brassard pour iPod
1 plaque pour fixer les différents modules
1 écran LCD
1 bouton poussoir
1 carte Arduino Uno
1 Grove Base Shield
1 capteur MQ-3
1 capteur TMP36
1 câble de connexion batterie — Arduino
Les fils de connexions

brassard, permettra qu'un mail ou un avertissement soit immédiatement envoyé.

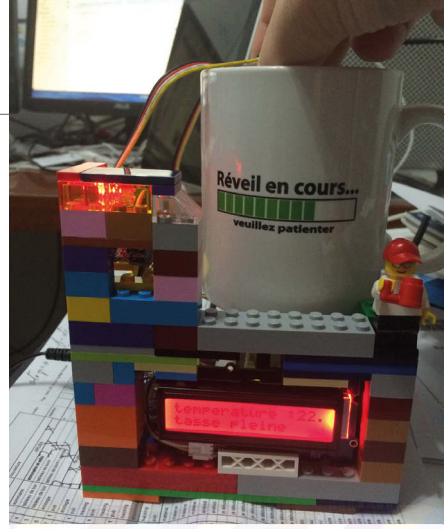
Tasse connectée pour Geek (vNext)

Ce projet a été initialement présenté sur le stand de Programmez !, ainsi que sur le stand Zone61 (merci Cécile) durant les TechDays 2015, puis dans une version modifiée et améliorée durant MakerFaire Paris. Une nouvelle version est en gestation.
La tasse connectée s'articule autour de deux éléments : la tasse en elle-même et le socle d'accueil en lego (c'est l'excuse pour acheter des boites de lego). Une des difficultés est de pouvoir caser l'ensemble des composants. Le design a été modifié à plusieurs reprises. La version actuelle du socle est d'environ 30 % plus petite que la 1ere version des TechDays. Là encore nous avons opté pour une Arduino Uno avec une Grove Base Shield pour connecter les composants. En entrée, nous avons un capteur de force, en A0, (sensible à partir de 50 grammes) pour détecter la présence de la tasse, un capteur de température TMP36, en A1, capteur d'humidité, en A2, et enfin un récepteur RF433, en A3. Nous pouvons ainsi mesurer la température du liquide de la tasse et le niveau de remplissage (pleine, mi-pleine, vide). Le choix du capteur de température a été un peu laborieux. Nous avions initialement opté pour le capteur Grove, puis nous avons envisagé un capteur de température par infrarouge qui s'est révélé difficile à calibrer correctement. Le TMP36 a été choisi pour son prix et sa simplicité d'utilisation.

En sortie, nous avons trois éléments visuels pour informer le Geek : un led bleu (pour dire si une tasse est présente, une barre led pour informer de la chaleur du liquide et un écran OLED 96x96 pour afficher l'ensemble des informations des différents capteurs. La tasse en elle-même possède une carte Arduino Uno, une LED rouge [de contrôle visuel], un capteur tilt et un émetteur RF433. L'objectif était de détecter la chute de la tasse et de la signaler en temps réel au socle connecté par radiofréquence. Au-delà, le scénario est très simple : la tasse connectée est chez une personne âgée, la personne chute [avec la tasse], l'information se répercute au socle qui transmet via un mail ou un SMS la situation. Il est aussi envisageable de faire une analyse de la consommation quotidienne d'une personne avec la tasse... à condition d'optimiser le design et d'intégrer les capteurs directement dans la tasse. Ce projet nous a occupé plusieurs semaines. Une des difficultés a été de stabiliser le code, car plusieurs librairies se sont révélées incompatibles entre elles ou avec d'autres composants. Par exemple, au départ, nous étions partis sur une carte Intel Galileo gen2, mais la Grove Led Bar a refusé de fonctionner correctement. Les modules RF433 ont connu des fonctionnements erratiques. La latence s'est révélée un peu forte à cause de tous les capteurs connectés sur les entrées analogiques. Et comme l'Arduino ne peut faire fonctionner qu'un seul programme à la fois, on boucle constamment sur l'ensemble du code. Et la connexion très proche de nombreux composants peut interférer sur le bon fonctionnement des capteurs. Ainsi, parfois, le TMP36 affiche +300° puis revient à la normale. Et la Led Bar exige d'être isolée des autres composants pour fonctionner correctement [consommation importante]. Si vous chargez trop la carte, vérifiez que nous ne dépassez pas sa capacité électrique...

One more thing : une montre faite maison

Notre dernier projet en cours est la réalisation d'une « montre », ou d'une horloge digitale, le choix final n'est pas encore décidé. Le principe est très simple : afficher la date et l'heure sur un écran ou des matrices de led. Pour ce faire,



Liste du matériel

2 cartes Arduino Uno
2 Grove Base Shield
2 Grove Led Socket
1 Grove Led Bar
1 capteur d'humidité
1 capteur TMP36
1 capteur de force
1 émetteur/1 récepteur RF433
1 boîte de Lego
1 tasse
1 écran OLED 96x96 (Grove)
Les fils de connexions

nous avons opté pour une Arduino Nano, avec possibilité de migrer sur une Arduino Mini. Nous utilisons un module RTC [horloge temps réel] qui fournira l'heure et la date. Pour le moment, nous avons opté pour un affichage sur un écran OLED 96x96, mais nous sommes limités sur les possibilités graphiques de la librairie. Nous avons commencé à tester un écran OLED « standard » avec la librairie graphique U8glib très utilisée dans les objets connectés, mais plus complexe à utiliser. Actuellement, nous rencontrons d'importants problèmes de stabilité d'affichage, par sécurité, nous sommes revenu un écran Grove OLED plus standard mais plus limité.
Un capteur tilt sera intégré pour activer l'affichage quand l'utilisateur lèvera le bras. Un capteur de température sera implémenté. Pour rendre réellement autonome la « montre », nous utiliserons une batterie LiPo 3,7 V avec un module de conversion de tension [3,7V vers 5 V]. La suite à la rentrée...
À noter qu'avec une carte nano, nous devons déconnecter et reconnecter la carte à chaque nouveau chargement du code sur la carte... ⊞

Quelques rappels basiques

- Pour obtenir une température en Celsius, vous devrez faire une conversion des donnés du TMP36.
- Ne pas oublier que certaines données / informations ne s'afficheront pas sur un écran OLED ou LCD de la même manière. Pour des chiffres, sur un OLED, il faut souvent utiliser un putNumber ou convertir les variables d'un type à un autre.
- Respecter la polarité des composants (j'ai déjà cramé un composant comme cela...)

Jubeat Return : une histoire de passionnés

Le projet Jubeat Return est un projet réalisé suite à une année passée en Corée du Sud. Là-bas nous avons découvert la culture Coréenne et ses particularités, notamment concernant l'univers du Jeu Vidéo. Le Jeu Vidéo tel que nous le connaissons principalement en France, s'oriente en forte majorité sur du jeu chez soi sur console de salon, ou bien sur console portable lors de trajets. Ce n'est pas si simple pour la Corée du Sud. Bien que les consoles soient présentes, de nombreuses salles d'arcades et de « PC Bang » (cybercafé coûtant environ 1$ par heure) peuplent les rues de ce pays. Ces bornes qui étaient présentes en Europe et aux Etats-Unis dans les années 80 ont ici toujours beaucoup de succès.

Quentin Guay et Jeremy Dubuc
Managers Innovation à l'Epitech Innovation HUB
5ème année à Epitech

Nous devons l'avouer, nous sommes tombés sous le charme de ces jeux si différents de ce que nous avions l'habitude de voir en France. Les jeux en salle d'Arcade sont tout aussi divers et variés que ceux que l'on peut trouver sur une console de salon, voir plus encore ! De nombreux jeux disposent de leurs propres contrôleurs, leur permettant d'inventer des gameplay tous plus originaux les uns que les autres. C'est le cas du jeu qui nous a le plus marqué parmi toutes ces bornes d'arcades : « Jubeat ».

La naissance de Jubeat Return

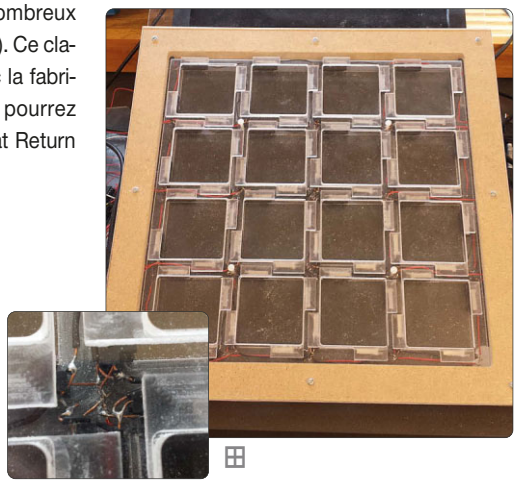
En revenant en France, nous nous sommes rendu compte qu'un jeu développé par Konami en 2008 nous manquait énormément. Ce jeu, connu sous le nom de « Jubeat » en Asie, connait un fort succès là-bas depuis maintenant 7 ans. Ce jeu de rythme musical reprend le même concept que des jeux tels que Guitar Hero, Rock Band, Osu et plein d'autres. La différence se situe principalement au niveau du contrôleur. Jubeat se joue sur un clavier de 4x4 touches. En fonction de la musique jouée, les touches du clavier s'allument afin d'indiquer au joueur d'appuyer dessus. Pour résumer simplement l'idée du jeu, Jubeat est un savant mélange de jeu de rythme musical et de chasse taupe ! Avec l'aide du HUB Innovation d'Epitech nous avons lancé le projet « Jubeat Return ». Le but de ce projet était de permettre à n'importe qui de pouvoir jouer à un jeu de rythme reprenant le même concept que le jeu que nous avions découvert. Nous avons constitué une équipe de 4 personnes motivées et commencé à travailler dessus en Septembre 2014. Neuf mois plus tard la borne d'arcade libre et open-source est sur le point d'être mise à disposition en ligne afin que tout le monde puisse jouer ensemble.

Création d'un clavier de type Jubeat Return

Afin de vous expliquer le fonctionnement du clavier réalisé dans le cadre de notre projet, nous allons vous expliquer comment réaliser un contrôleur de 5 touches pouvant être utilisé sur de nombreux jeux de rythme existants (tel que Frets on Fire). Ce clavier vous permettra de vous familiariser avec la fabrication de contrôleurs personnalisés, vous pourrez ainsi créer vous-même un clavier pour Jubeat Return lors de la sortie du projet !

Matériel nécessaire :

- Par bouton :
  - 4 switchs
  - Une diode
  - Une résistance 330 Ohm
- Un support pour les boutons
- Fer à souder / pinces coupantes
- Étain / câbles
- Vis / écrous
- Arduino / Breadboard
- Résistances / Diodes



Pour commencer la construction de notre clavier, nous avons utilisé une imprimante 3D afin de produire un support et des boutons basiques. Ces boutons réalisés en plastique ABS suffisent largement pour valider le prototype que nous voulions faire. Par la suite le plastique ABS fut remplacé par du PMMA pour des soucis de solidité et de transparence. Le support est composé de 2 parties : supérieure et inférieure. Le bouton réalisé doit venir s'insérer sans soucis entre ces deux parties. La partie supérieure va servir à maintenir en place le bouton. La partie inférieure, elle, sert de base aux switchs. L'idée est de permettre aux excroissances du bouton de venir se poser sur les switchs. Une fois le bouton pris en tenaille entre les 2 parties du support, il ne reste plus qu'à régler la hauteur du support par l'intermédiaire de vis et d'écrous. Il ne vous reste plus qu'à répéter autant de fois que nécessaire l'opération pour obtenir tous vos boutons. Il est aussi possible de prévoir un support contenant plusieurs boutons, et de les regrouper tous ensembles, c'est ce que nous avons fait pour le projet « Jubeat Return ».

Connexion du clavier à l'ordinateur

Maintenant que votre clavier est prêt, il vous reste à le connecter à votre ordinateur pour pouvoir jouer. Pour cela nous allons utiliser une Arduino. Vous allez avoir besoin de souder des câbles directement sur les switchs. Ces câbles vont permettre de brancher les switchs sur une Breadboard. Pour chaque switch, ajoutez une résistance sur la Breadboard puis effectuez la connexion vers l'Arduino. À des fins de tests, nous vous conseillons d'ajouter aussi une Led afin d'effectuer des tests et de voir si tout est fonctionnel. Branchez votre Arduino à l'ordinateur. Il ne vous reste plus qu'à réaliser une application prenant en compte les signaux reçus par l'Arduino. Afin d'être compatible avec la majorité des jeux PC sortis jusqu'à présent, il y a une solution très simple pour utiliser votre contrôleur. Il suffit de faire correspondre les touches du clavier physique à celles du contrôleur. Ainsi sur un jeu tel que Frets on Fire, nos cinq boutons correspondraient aux touches F1 / F2 / F3 / F4 / F5.

Jubeat Return: the game

Pour réaliser « Jubeat Return » nous avons décidé de développer en C++. L'affichage utilise un simple écran d'ordinateur avec une résolution de 1920x1080 incliné à 90 degrés. La librairie SFML est utilisée pour toute la partie graphique. L'écran est séparé en deux parties. La partie supérieure permettant d'afficher toutes les informations importantes de la musique sélectionnée, et la partie inférieure servant d'affichage aux touches transparentes du contrôleur maison que nous avons réalisé. Le contrôleur vient se poser directe-

ment sur l'écran. Une plaque de PMMA sépare ces deux éléments afin d'éviter que l'écran s'abime facilement lors des pressions répétées de boutons tout au long d'une partie. Le fonctionnement d'une borne d'arcade a été entièrement reproduit : affichage des crédits, nombre de parties restantes, choix du mode de jeu, écran de sélection de la musique... Rien n'a été oublié. Au lancement, le joueur est invité à entrer ses identifiants pour que son pseudo soit mis en avant si jamais ses scores entrent dans les Highscore de la borne d'arcade. Chaque morceau jouable existe en plusieurs difficultés, permettant aux apprentis joueurs de débuter en douceur, puis de recommencer leurs morceaux favoris dans une difficulté supérieure. Les morceaux de « Jubeat Return » sont composés de fichiers audio en Wav, ainsi que de fichiers contenant les « Beatmaps » en fonction de la difficulté. Il est possible de trouver des « Beatmaps » sur Internet basés sur les morceaux intégrés dans les jeux de Konami en Asie, mais « Jubeat Return » dispose de son propre système permettant de « mapper » à la volée des morceaux. De ce fait, la liste des morceaux jouables est illimitée et dépend uniquement des décisions de la communauté.

Interpréter les signaux du clavier

Voici le code Processing utilisé pour interpréter les boutons de notre contrôleur maison. Une fois adapté, ce code peut être utilisé sur des claviers de tailles différentes :

```
int inputStates[16];
void setup() { // initialize the buttons' inputs:
 for (int i = 0; i < 16; i++)
 {
  inputStates[i] = 0;
 }
 for (int i = 2; i <= 13; i++)
 {
  pinMode(i, INPUT);
 }
 Serial.begin(9600);
}
void loop() {
 /* Digital reading: PIN2-13 <=> 1 2 3 4 5 6 7 8 9 10 11 12 */
 for (int i = 2; i < 14; i++)
 {
  int buttonState = digitalRead(i);
  if (buttonState != inputStates[i-2])
  {
   if (buttonState == 1) {
    Serial.print("+");
   } else {
    Serial.print("-");
   }
   Serial.print(i);
   inputStates[i-2] = buttonState;
  }
 }
 /* Analog reading: (4) PIN A2-A5 <=> 16 15 14 13 - */
 for (int i = 0; i < 4; i++)
 {
  int sensorValue;
  int buttonState;
  if (i == 0)
   sensorValue = analogRead(A5);
  if (i == 1)
   sensorValue = analogRead(A4);
  if (i == 2)
   sensorValue = analogRead(A3);
  if (i == 3)
   sensorValue = analogRead(A2);
  buttonState = 0;
  if (sensorValue > 50) {
   buttonState = 1;
  }
  if (buttonState != inputStates[i + 12]) {
   if (buttonState == 1) {
    Serial.print("+");
   } else {
    Serial.print("-");
   }
   Serial.print(i + 14);
   inputStates[i + 12] = buttonState;
  }
 }
 delay(20);
}
```

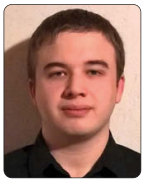
Afin de vous aider à mieux comprendre le fonctionnement d'un clavier réalisé avec une Arduino nous vous conseillons d'aller jeter un œil sur « http://playground.arduino.cc/ ». Vous pourrez y trouver de nombreux exemples et conseils, dont celui fourni plus bas.
La librairie « Keyboard.h » est parfaite pour la réalisation d'un clavier maison en Processing. http://playground.arduino.cc/uploads/Code/keypad.zip
Pour utiliser la librairie, ajoutez celle-ci dans votre dossier « arduino\libraries\ » puis importez-la directement dans votre IDE. Une fois importée, n'oubliez pas d'inclure « Keypad.h » au début de votre code.

```
#include <Keypad.h>
const byte rows = 4; //four rows
const byte cols = 3; //three columns
char keys[rows][cols] = {
 {'1','2','3'},
 {'4','5','6'},
 {'7','8','9'},
 {'#','0','*'}
};
byte rowPins[rows] = {5, 4, 3, 2}; //connect to the row pinouts of the keypad
byte colPins[cols] = {8, 7, 6}; //connect to the column pinouts of the keypad
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, rows, cols );
void setup(){
 Serial.begin(9600);
}
void loop(){
 char key = keypad.getKey();
 if (key != NO_KEY){
  Serial.println(key);
 }
}
```

Nous espérons que cet article vous aura donné envie de vous lancer vous aussi dans l'aventure et de réaliser des montages avec une Arduino. Vous pourrez bientôt essayer Jubeat Return et avoir accès aux plans et au code source du projet. En attendant, expérimentez !

Scratch, un outil de développement pour les enfants

Scratch est un logiciel conçu par le MIT (Massachusetts Institute of Technology) afin d'initier les jeunes enfants à partir de 8 ans à la programmation. Scratch est une interface visuelle permettant de développer des applications avec le langage de programmation Smalltalk.



Kévin Sibué

Fig.1

Scratch possède une particularité bien à lui qui est de pouvoir modifier le code source de l'application développée en cours d'exécution. Scratch traite les notions parfois difficiles d'accès aux débutants comme les tests, les boucles ainsi que les objets et l'affectation des variables.

Le point fort de Scratch, et ce qui le rend accessible à tous, est sa diversité de langages disponibles. Ainsi l'enfant peut coder son application directement dans sa langue maternelle, ce qui facilitera grandement son apprentissage.

Tout le monde peut y avoir accès puisque Scratch est un logiciel Web s'exécutant dans votre navigateur internet (Safari, Chrome 35, Firefox 31 ou Internet Explorer 8+), il nécessite Flash Player pour fonctionner.

Le développement de Scratch n'est pas encore terminé. La version 2 du logiciel est disponible en ligne. Il est distribué sous 2 licences, dont la GPL. Scratch reste une marque déposée dont la redistribution est soumise à plusieurs contraintes.

Néanmoins une version libre, ne contenant pas de marque déposée est disponible en téléchargement et permet la redistribution des modifications apportées avec une licence GPL.

Les développeurs de Scratch ont développé une nouvelle application pour les enfants de 5 à 7 ans : ScratchJr. Vous pourrez en savoir plus en suivant ce lien : http://www.scratchjr.org/

Le slogan de Scratch est « Imagine, Programme, Partage ! » et c'est une réussite puisque près de 6 millions d'utilisateurs sont aujourd'hui enregistrés sur leur site ; et ce nombre ne cesse d'augmenter. Dans ces utilisateurs nous retrouvons bien entendu des enfants souhaitant découvrir l'informatique et notamment le développement, mais également des passionnés, des étudiants et des professeurs qui ensemble forment la grande communauté de Scratch. Scratch est disponible dans 70 langues permettant à n'importe quel enfant ou autre passionné d'utiliser au mieux le logiciel.

Scratch et l'école

Scratch arrive également dans les écoles, où les enseignants peuvent proposer grâce à lui des cours sur la découverte de la programmation dès le plus jeune âge. Ainsi on peut trouver dans certaines écoles primaires de France des enfants jouant avec notre ami le chat virtuel afin de créer quelques histoires dynamiques ou des petits jeux ludiques.

Ce qui permet au jeune esprit de commencer à appréhender ce qu'est le développement et ainsi mieux comprendre toutes les technologies qui les entourent pour enfin démystifier le côté « magique » que possède parfois l'informatique quand on ne connaît pas les rouages internes Fig.1.

L'enseignement se poursuit au collège, où dorénavant Scratch est utilisé pour mieux comprendre les sciences et les appliquer directement dans le monde virtuel.

Ceci offre donc de toutes nouvelles possibilités pour les étudiants qui peuvent dorénavant passer de la théorie à la pratique pour ce qui est des mathématiques et des sciences. Mais le collège est aussi le moment où l'on commence à penser à notre futur métier, à notre avenir : faire découvrir le développement aux jeunes pourrait leur permettre de découvrir un nouveau métier, et, qui sait, peut-être une passion ?

Les fonctionnalités de Scratch

Scratch est un environnement de développement pour les enfants. Néanmoins il comprend toutes les fonctionnalités de base que possède un IDE classique (IDE = environnement de développement intégré). Il permet notamment de sauvegarder directement sur votre machine le ou les projets que vous concevrez, mais également d'importer des projets de votre machine vers le site Web. Une fois que vous serez enregistrés sur le site de Scratch, de nombreuses autres fonctionnalités vous seront alors disponibles. Parmi ces fonctionnalités, la plus intéressante est la possibilité de partager avec l'ensemble de la communauté vos projets (finis ou non). Vous pourrez également créer un nouveau « Studio ». Un studio est un regroupement de Scratchers qui peuvent ensemble partager des projets communs ou ayant une certaine similitude. Dans Scratch, les instructions sont codifiées par couleur suivant leurs actions. On retrouvera par exemple tout ce qui concerne les mouvements en bleu, tandis que les évènements sont en orange foncé et les opérateurs en vert clair.

Bien entendu nous retrouvons toutes les instructions de base que l'on puisse trouver dans n'importe quel langage : les boucles, les tests ou encore la création d'objet.

Elles sont toutes présentes sous la forme de « blocs » comme vous pouvez le voir ci-dessus. Certains comme celui-ci peuvent contenir d'autres blocs (comme les boucles), mais peuvent aussi être précédés ou suivie par un autre. D'autres en revanche peuvent contenir des blocs en paramètres comme les capteurs et opérateurs. Si dans la figure précédente nous souhaitons indiquer, non pas une valeur fixe

comme « 10 », mais le contenu d'une variable ou le résultat d'un calcul, alors nous pourrons y mettre des blocs « variable » ou « opérateur ». Par conséquent lorsque ceux-ci sont assemblés les uns par rapport aux autres, nous obtenons ceci : Fig.2.

L'ordre des différents regroupements de blocs n'est pas important, puisque ceux-ci sont toujours précédés par un évènement faisant office de déclencheur. Il est donc possible d'avoir de nombreux regroupements de blocs au sein d'un même objet (l'objet actuel étant un chat que l'on peut déplacer grâce aux flèches directionnelles et à la barre d'espace).

Dans Scratch les objets sont nommés « lutin ». Il est possible de rajouter des lutins sur votre scène (fenêtre d'action).

Lors de la création d'un lutin, vous devrez lui choisir une apparence, sous la forme d'une image présente dans la collection de Scratch. Cependant si la collection ne vous convient pas ou que vous souhaitez avoir vos propres images, vous pouvez créer un lutin à partir d'un éditeur d'image, et aussi depuis un fichier de votre machine ou de votre webcam.

Il est ensuite possible de modifier les informations de votre lutin comme son nom, sa direction (angle de l'image), son type de rotation, si celui-ci peut être déplacé par l'utilisateur et s'il est visible.

Il en est de même pour votre scène. Il vous est possible d'ajouter un fond contenu dans la collection de Scratchs, mais ici aussi vous pourrez le dessiner vous-même ou l'importer depuis un fichier image ou votre webcam. Ce qui offre une grande liberté aux enfants souhaitant créer leur propre petit jeu.

Scratch intègre également la gestion de son, comme vous pouvez le voir dans l'exemple ci-dessus. Vous pourrez donc importer dans votre projet des sons et musiques de votre propre bibliothèque afin de les ajouter à votre application. Chaque lutin présent dans la bibliothèque de Scratch possède dès sa création un son lui étant propre. Il vous sera aussi possible de sélectionner un instrument contenu dans une liste, où vous retrouverez le piano, la guitare, la flûte et 18 autres instruments. Puis vous pourrez jouer une note ou modifier le tempo.

Une gestion de stylo vous permettant de dessiner sur votre scène dynamiquement est également disponible. Il vous sera possible de changer la couleur, la taille et l'intensité du crayon sélectionné pour ensuite dessiner à l'écran.

L'ensemble des opérateurs est également présent, tel que l'addition, la soustraction, la multiplication et la division. Un générateur de nombres entiers aléatoires est également disponible. Les tests logiques comme « et » « ou » et « non » sont aussi au rendez-vous. Bien entendu il est possible de combiner entre eux l'ensemble de ces blocs afin de former « ou pas », « et pas » et tant d'autres.

Des fonctions de rassemblement de chaînes de caractères et de calculs mathématiques sont présentes, telles que l'arrondi d'un nombre, le modulo de deux nombres ou encore la racine d'un autre.

Cependant, comme vous l'aurez remarqué, je n'ai pas encore parlé des variables, éléments pourtant essentiels de n'importe qu'elle application. Mais ici aussi les développeurs de Scratch ont pensé à tout ! Dans la rubrique « Données », il vous sera possible de créer une variable ou une liste. Il faudra alors lui donner un nom et définir sa portée : si elle est globale à tous les lutins ou seulement pour le lutin sélectionné. Suite à cela, une multitude de blocs seront créés permettant de manipuler la variable ou la liste en question.

Les compléments de Scratch

Si tous ces blocs ne vous suffisent pas, vous aurez la possibilité de créer les vôtres. Vous pourrez donc ajouter en entrée : un nombre, une chaîne de caractères, un booléen. Une fois défini, il sera rajouté dans la liste des blocs disponibles et une nouvelle « fonction » de couleur violette sera également ajoutée dans la liste des blocs. Il faudra alors que vous définissiez son comportement et les actions qu'elle est censée effectuer.

Il vous sera également possible d'ajouter des extensions comme PicoBoard et Lego Wedo qui rajouteront leurs propres blocs afin de contrôler des matériels externes comme les cartes électroniques ou des webservices.

L'environnement de développement

L'environnement de développement est disponible en ligne et hors-ligne. Si vous souhaitez utiliser l'environnement hors ligne, il suffit de télécharger et d'installer Adobe Air (disponible sur toutes les plateformes) puis d'installer l'éditeur Scratch, disponible sur Windows, OS X et Linux. L'environnement de développement que propose Scratch est entièrement visuel.

L'environnement se découpe en trois zones distinctes. La première et la principale étant la zone de droite qui affichera suivant l'onglet que vous sélectionnerez, soit les scripts rattachés au lutin sélectionné, soit son costume (l'éditeur d'image), ou alors le gestionnaire des sons.

Quant à la partie de gauche, elle est elle-même divisée en deux. La partie haute étant la scène de votre application. C'est ici que vos lutins seront placés, et c'est dans cet environnement qu'ils évoluent. C'est également dans cette partie que vous pourrez nommer votre nouvelle création et que les deux boutons de base (le drapeau vert et le stop rouge, le premier étant un bouton pouvant être utilisé comme déclencheur pour commencer le jeu, et le second permet de stopper le jeu) sont présents. La partie basse, en revanche, vous servira à modifier l'arrière-plan de la scène, mais également la liste de l'ensemble de vos lutins déjà créés, et, enfin, vous affichera toutes les options pour en créer de nouveaux.

Vous aurez pu le constater tout au long de votre lecture que Scratch est un outil qui semble avoir un bel avenir, et qui remplit correctement son rôle auprès des jeunes. Ayant dû moi-même me former à Scratch, je peux vous le dire qu'il est extrêmement facile de commencer, et qu'on y prend goût rapidement.

Il ne vous reste plus cas essayer vous-même ou demander à vos enfants de jouer avec :

https://scratch.mit.edu/

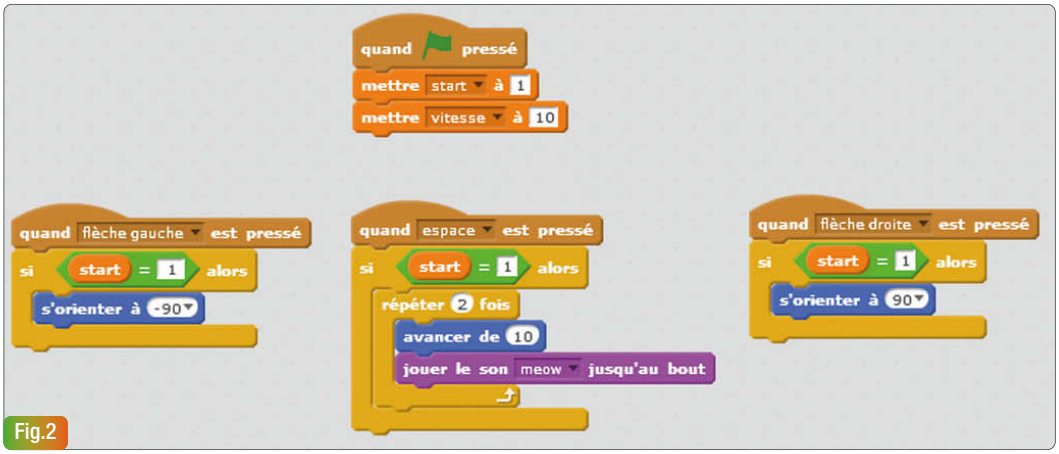


Fig.2

Je démarre avec Scratch

Lorsque Scratch démarre sur un ordinateur, cela accroche tout de suite le regard des enfants comme des plus grands ! Scratch donne envie de tout essayer, de tout toucher. Cependant, si le ticket d'entrée n'est pas très élevé, quelques pistes pour démarrer peuvent s'avérer utiles. Ce qui suit est issu de l'expérience acquise durant deux semestres d'animation auprès des enfants du Centre de Loisirs de l'Etang de la ville de Nanterre pendant l'année scolaire 2013-2014, dans le cadre du projet Programatoo.

Aline Paponaud

A chaque enfant sa participation !

Seul, accompagné d'un adulte, avec d'autres enfants, toutes les configurations sont possibles, il n'y a pas de recommandation particulière pour profiter au mieux de Scratch, qui est avant tout un jeu.

Il est possible de commencer sur Scratch dès le plus jeune âge. Savoir lire est tout de même indispensable pour bien saisir ce qui se passe dans chaque bloc.

Avec l'aide d'un adulte, un enfant qui ne sait pas encore lire peut quand même comprendre certaines choses car chaque type de bloc possède sa couleur.

Le mieux est de le faire participer dans l'équipe du jeu, avec les plus grands : dans les ateliers collectifs comme le coding goûter(1), les plus petits font des dessins de personnages avec plusieurs costumes, qu'un adulte scanne et intègre dans Scratch pour qu'ils soient utilisés par les plus grands.

Gérer le temps

Lors d'un atelier collectif ou à la maison, il faut isoler différents moments importants de la découverte de Scratch : un temps où l'enfant essaye des choses, et un temps où il faut montrer ce qui est fait et le tester.

Cela permet de se recentrer sur l'objectif en cours, de valider ce qui est réalisé, et surtout de ne pas passer trop de temps sur des choses trop compliquées sans les voir terminées, ce qui peut créer des frustrations.

Faire fonctionner un premier programme au plus vite

Le plus souvent, l'enfant voit du premier coup d'œil qu'il va être possible de tout faire avec Scratch. Et au lieu de se dire « je vais faire se déplacer le petit chat de gauche à droite », il se dit plutôt « je veux refaire les graphismes de

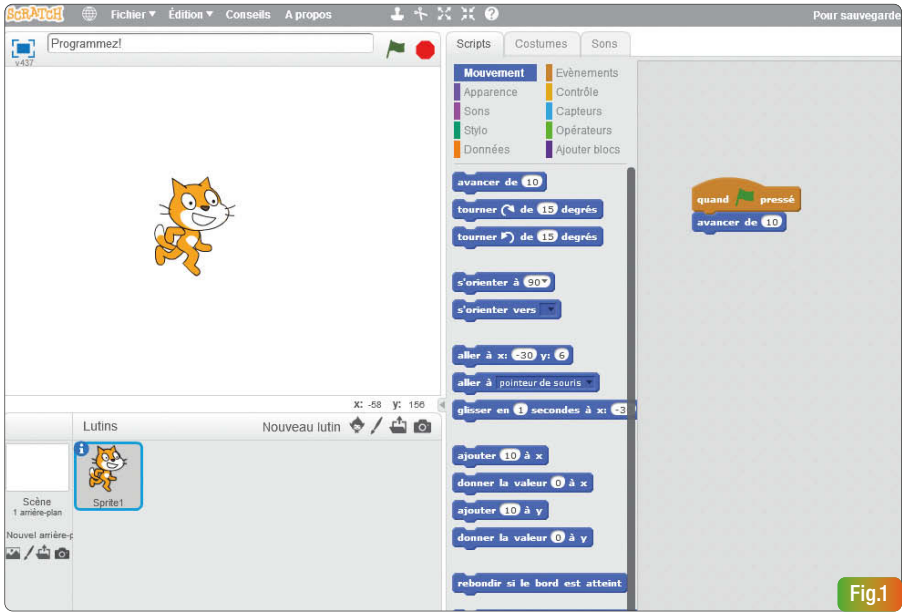


Fig.1

Fig.2

Fig.3

FarCry 4 que vient d'acheter mon grand frère ! »... Vous voyez le souci.

L'idée est donc d'encourager l'enfant à se fixer un premier objectif atteignable, qui peut constituer une étape vers quelque chose de plus ambitieux. Ainsi, il ou elle pourra découvrir les fonctionnalités principales de Scratch et les premiers blocs de base, le tout par des étapes très simples.

Le chat qui se déplace de gauche à droite

Garder le personnage d'origine, le fond blanc, et simplement, faire avancer le chat. Pour cela, rien de plus simple, c'est un bloc d'action associé à l'événement « départ » : Fig.1.

Ainsi lorsque je clique sur « départ », le chat bouge ! Magique. Pour aller plus loin, il suffit de cliquer plusieurs fois. Cela me fatigue de cliquer plusieurs fois, je vais donc réaliser ma première boucle. J'insère l'action dans un bloc : Fig.2. Cette fois quand je clique sur « départ », le chat part sur la droite... Mais se prend le mur ! Comment faire ? Scratch a tout prévu. Cette action est un peu difficile à trouver la première fois, il s'agit de « rebondir si le bord est atteint ». Ce n'est pas encore parfait : le chat se retourne complètement. Il y a une action qui permet de ne se retourner que de gauche à droite.

Il suffit de l'exécuter une fois dans le programme, au début : Fig.3. Ca y est : le chat peut se déplacer.

Animer la marche

Pour l'instant on a l'impression que notre chat fait du patin à glace, non ?

Afin d'animer un peu plus notre personnage, il est possible de changer son « costume » à chaque étape, pour mimer ses pas. Le chat de Scratch (ce n'est pas le cas de tous les personnages mis à disposition dans l'application) dispose de deux costumes mimant un pas que vous pourrez voir dans l'onglet « costumes » Fig.4.

Il est possible dans cet onglet d'éditer ou de des-

siner soi-même ses costumes. Un troisième onglet « sons » permet de faire miauler le chat, ou même d'enregistrer ses propres sons. Mais ne nous éparpillons pas, la découverte se fera plus tard, nous avons un objectif ! Nous allons ajouter une action « costume suivant » à la boucle. Cette action est dans la partie « Apparence » Fig.5.

Essayons : le chat ne marche pas : il court ! Il est possible de temporiser en faisant attendre le chat un tout petit instant, avec l'action de contrôle « attendre __ secondes ».

Il est possible d'y insérer un nombre à virgule, sinon c'est trop lent.

Il est intéressant de voir la vitesse changer quand on modifie le nombre correspondant au temps d'attente dans la boucle Fig.6.

Et voilà, notre première étape est atteinte. Voici les notions de programmation qui auront été apprises rien qu'en faisant ce premier pas dans Scratch :

- La différence entre « Action » et « Evènement » : un verbe, « quand... »,
- Les boucles, qui permettent de répéter toujours la même action,
- L'ajout d'un bloc d'actions à une boucle,
- La notion de temps et de séquence,
- Les éléments variables.

Pour aller plus loin, il faudra passer aux tutoriaux plus avancés, ou explorer soi-même.

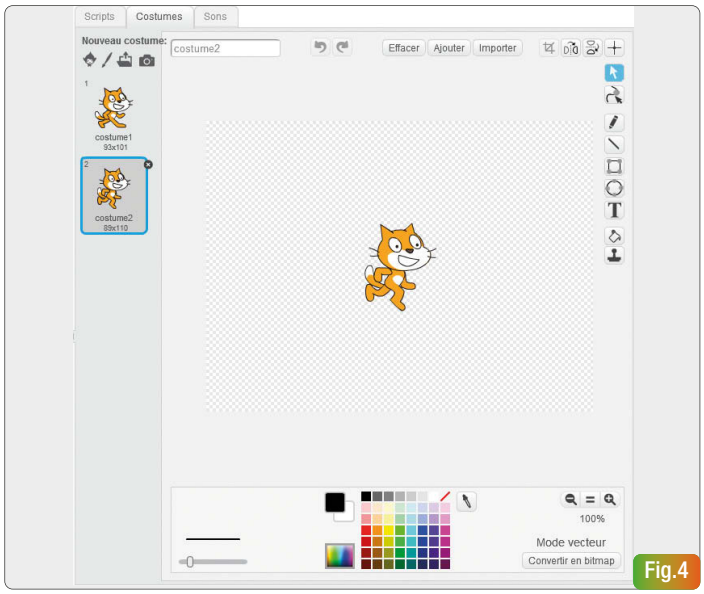


Fig.4

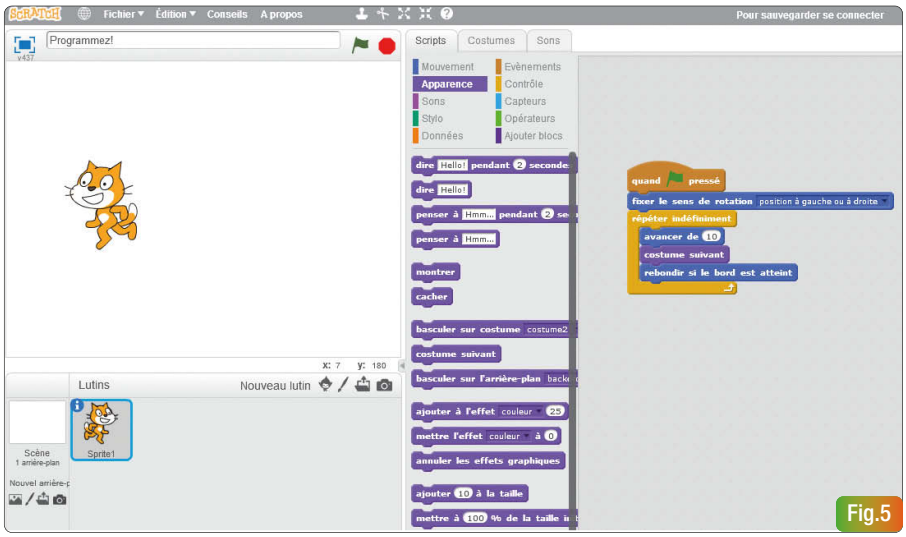


Fig.5

Un atelier près de chez vous

Parmi les nombreuses activités qui existent, la *Codeweek*(2) européenne aura lieu du 10 au 18 octobre 2015.

L'occasion pour votre enfant de découvrir la programmation avec Scratch ou d'autres outils. Le site de la Codeweek est celui qui regroupe toutes les initiatives en Europe, dont un très grand nombre en France.

La liste est en train d'être mise à jour en ce moment, n'hésitez pas à la consulter régulièrement ou à contacter un des ambassadeurs français (bientôt disponible sur le site).

Devoxx4Kids(3) est aussi une association très active en France, en Belgique et dans d'autres pays. Le Coding Goûter était également cité précédemment, toutes les infos sont sur leur site Web.

Fig.6

(1) http://codinggouter.org
(2) http://codeweek.eu
(3) http://devoxx4kids.org

Exemple 1 - Paf et La Balle

par Audrey Neveu

L'objectif : faire un jeu avec une balle. Quand Paf atteindra la balle, il gagnera un point...

Ce qu'on va apprendre :

- Créer plusieurs objets dans Scratch, les nommer,
- Définir la position d'un objet,
- Diriger des objets avec les flèches,
- Faire rebondir des objets,
- Utiliser l'opérateur "nombre aléatoire",
- Utiliser des variables et faire changer leur valeur,
- Envoyer et recevoir un évènement.

Etape 1 : Choisir un objet qui va rebondir

Scratch s'ouvre avec le chat Scratch en tant qu'objet principal. Mais tu peux en choisir un autre ! Choisis n'importe quel objet qui fera office de balle. Renomme-le en "Ballon" Fig.1.

Etape 2 : Faire rebondir le ballon

Fais en sorte que la balle rebondisse tout le temps, partout ! Elle doit continuellement avancer, et rebondir si le bord est atteint. Tu peux lui donner un angle de départ si tu veux qu'elle fasse des diagonales.

Etape 3 : Choisir Paf

Ajoute un animal que tu nommeras Paf.

Etape 4 : Diriger Paf

Paf sera dirigé avec les flèches du clavier. Fais en sorte que quand tu appuieras sur la flèche du haut, il aille vers le haut, sur la flèche de gauche, vers la gauche, etc., pour le diriger complètement Fig.2.

Etape 5 : Créer un compteur de points

Crée une nouvelle variable appelée Points.

Etape 6 : Quand la balle est touchée par Paf, on gagne un point !

Maintenant quand la balle est touchée par Paf, incrémente le compteur de Points de 1 Fig.3.

Etape 7 : Quand la balle est touchée par Paf, elle disparaît

En plus d'augmenter les points de 1, il faut maintenant que tu fasses disparaître la balle quand Paf la touche, comme si il l'avait faite éclater. Mais ne t'inquiète pas, dans la prochaine étape nous allons la faire réapparaître.

Etape 8 : Faire réapparaître la balle... N'importe où

Et maintenant que la balle a disparu, il faut la faire réapparaître pour un nouveau tour ! Pour cela, en plus d'augmenter les points et de disparaître, ajoute une action qui permettra à la Balle de réapparaître, en allant à une position aléatoire. Il faut que la balle change de position X et Y, en attribuant à ses X et Y des valeurs aléatoires. Attention à ne pas sortir de l'écran ! La valeur minimum de X est -240, son maximum est 240. Pour Y, son minimum est -180, son maximum est 180 Fig.4.

Etape 9 : Mais en fait... Quand est-ce qu'on gagne ?

Eh oui, toutes les bonnes choses ont une fin ! Le jeu se termine quand Paf a récolté 100 points. C'est à dire, quand il a touché 100 fois la balle. Fais dire à Paf "Gagné" quand le nombre de points arrive à 100, et arrêter tout. Attention, quand le jeu s'arrête puis redémarre, il faut remettre les points à Zéro ! Fig.5 et 6.

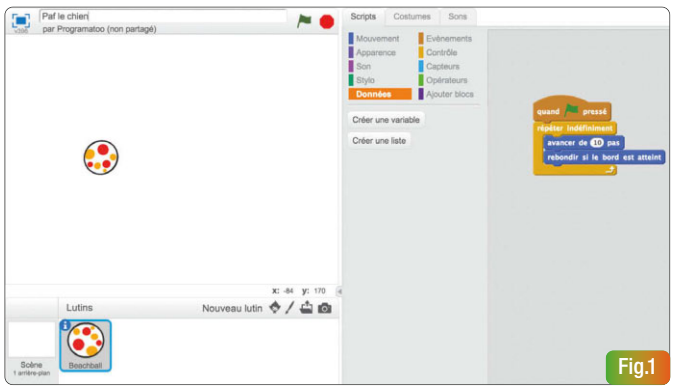


Fig.1

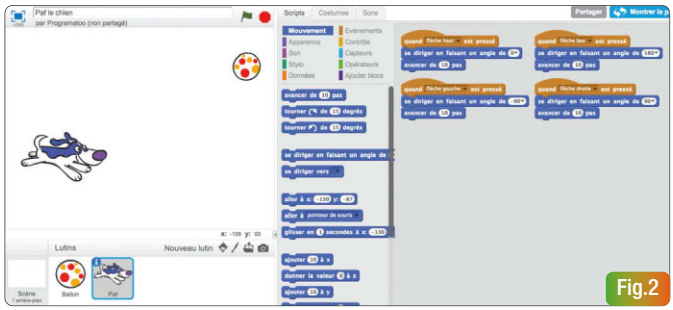


Fig.2

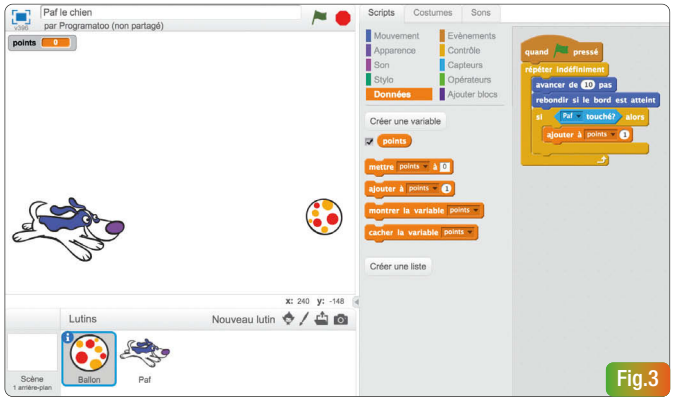


Fig.3

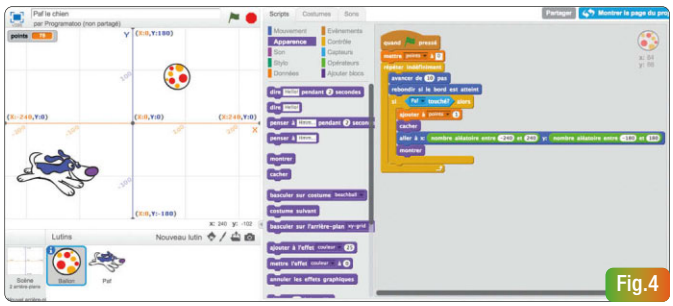


Fig.4

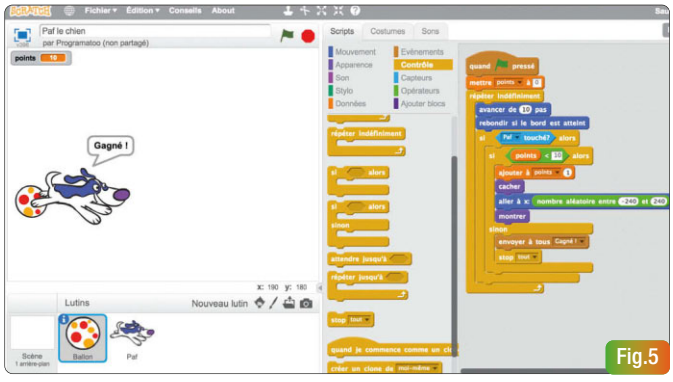


Fig.5

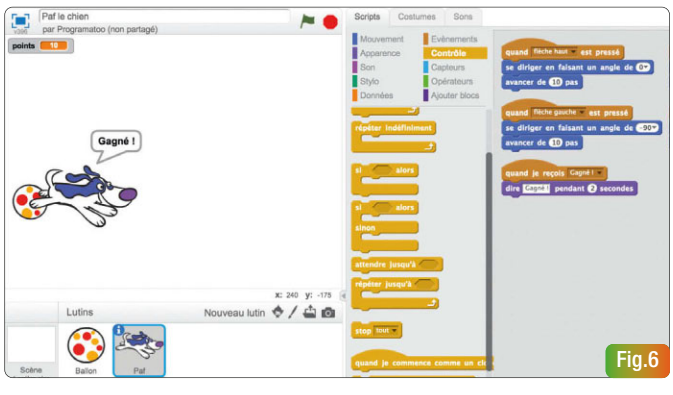


Fig.6

Exemple 2 - Pico et Giga

par Audrey Neveu

L'objectif : inventer une histoire entre deux personnages, les faire se déplacer et dialoguer.

Ce qu'on va apprendre :

- Créer plusieurs objets dans Scratch, les nommer,
- Diriger des objets, en leur donnant des directions,
- Initialiser un objet avec des valeurs,
- Animer des objets, avec des costumes,
- Faire un dialogue entre deux personnages.

Etape 1 : Créer les personnages, le fond

Choisis deux personnages qui devront discuter ensemble et donne leur un nom : dans notre exemple nous prendrons Pico et Giga, mais tu peux choisir ceux que tu veux ! Choisis une scène, qui sera l'endroit où se passe ton histoire. Choisis enfin un troisième objet, qui sera un accessoire pour ton histoire (arbre, fusée, coquillage, voiture...) Place un des deux personnages dans un coin, et l'autre en dehors de l'écran, dans le coin opposé Fig.1.

Etape 2 : Les personnages se rencontrent !

Par exemple Giga est en train d'attendre Pico qu'elle guette en regardant à droite alors que Pico se dirige vers elle depuis la gauche. Fais marcher Pico vers Giga Fig.2.

Faire marcher un personnage

Pour faire marcher ton personnage, il faut qu'il ait plusieurs costumes. Tu pourras vérifier que c'est le cas dans l'onglet "Costumes". Si cela est possible, tu peux aussi en importer des supplémentaires. Le principe, c'est de faire des pas. Quand on répète indéfiniment l'instruction "Avancer de x pas", il faudra aussi changer de costume pour que le personnage change de pied, à l'aide de l'action "costume suivant". Il faudra aussi que tu dises à l'ordinateur d'attendre un peu entre chaque répétition car sinon, il ira tellement vite que tu n'auras pas le temps de le voir ! Fig.3.

Etape 3 : Initialiser un objet

Chaque fois que ton personnage se déplace, tu dois le remettre à sa position de départ si tu veux conserver ton animation ... un peu casse-pieds non ? C'est pourquoi nous allons l'initialiser. Pour cela nous allons lui donner toutes les informations sur l'aspect qu'il doit avoir au lancement du programme : à quel emplacement est-il ? Quel est son costume ? Est-il visible ou pas ? Une fois que tout cela est fait, relance le programme, arrête-le, puis relance -le à nouveau : les objets reviennent à leur place de départ.

Etape 4 : Les personnages se parlent

Engage le dialogue entre les deux personnages ! On va faire comme une BD, en affichant le texte dans des bulles. Pour cela, utilise l'action "Dire" et donne la parole à ton personnage, puis recommence avec l'autre personnage pour la réponse Fig.4. Pour construire le dialogue tu vas devoir ajouter des temps de pause entre chaque réplique, sinon les personnages parleront en même temps ! Fig.5.

Etape 5 : Les personnages interagissent avec l'accessoire

Après avoir bien discuté, tes personnages ont sûrement envie de bouger un peu ! Peut-être qu'ils décident de prendre la voiture pour aller faire un tour, de jouer avec un ballon ou de grimper à un arbre ... Change la direction de tes personnages pour qu'ils se tournent vers l'accessoire, et fais-les se déplacer ensemble vers lui. Une fois que c'est fait, tu peux en faire tout ce que tu souhaites ! Ici nous ne verrons qu'un exemple, mais les possibilités sont bien plus grandes...

En voiture Simone !

Pico et Giga se déplacent en fusée. Pour donner l'illusion qu'ils entrent dedans, une fois que chacun des personnages touche la fusée, fais-le disparaître. Une fois que c'est fait, on a l'impression qu'ils sont tous les deux dans la fusée. Et maintenant on peut partir... Fais se déplacer la fusée verticalement, tu peux aussi diminuer sa taille progressivement jusqu'à la faire disparaître aussi dans le ciel Fig.6.


Fig.1

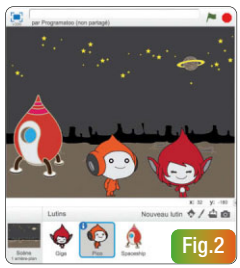


Fig.2

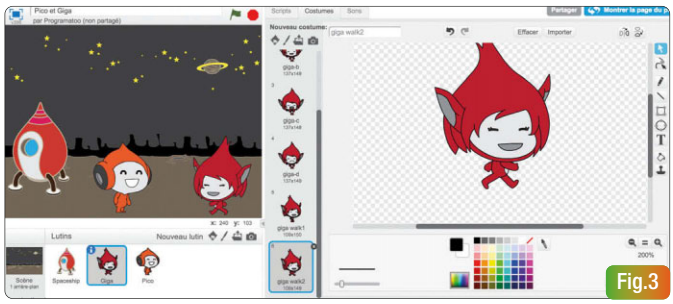


Fig.3

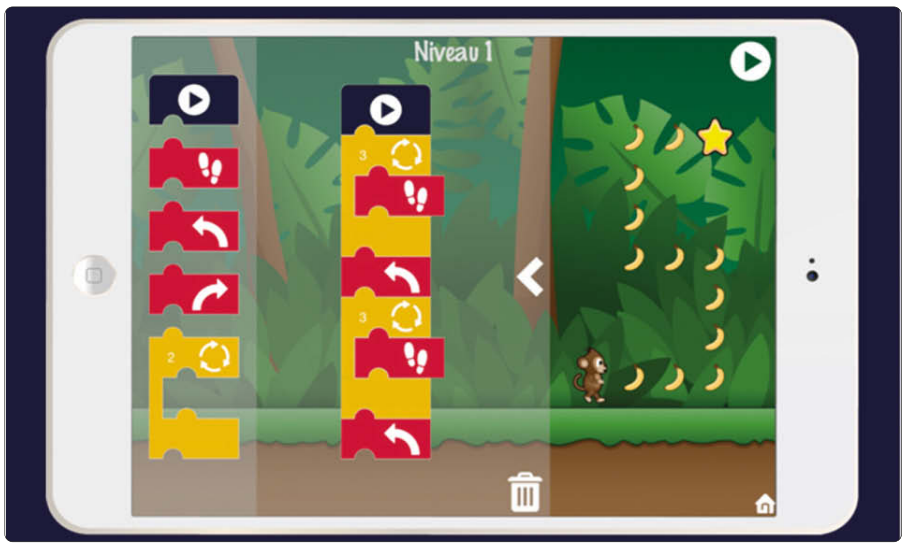

Fig.5

Fig.4


Fig.6

Loliplop : l'autre environnement de programmation pour les enfants

La toulousaine Lisbé Juin a créé Loliplop, une application ludo-éducative, à partir d'un constat simple : plus de 20% des enfants de moins de 12 ans ont déjà leur propre tablette, et très rares sont les applications vraiment adaptées proposant une initiation complète à la programmation informatique.



C'est durant l'activité "Crée ton jeu numérique" mise en place dans les accueils périscolaires de la Communauté de Communes de la Piège-Lauragais-Malepère, dans l'Aude, que la jeune start-up teste, depuis février, son prototype.
Cette application, qui s'adresse aux enfants entre 5 et 11 ans, est multi supports : tablettes, ordinateurs et tableaux interactifs. Elle a spécialement été conçue pour correspondre au mieux aux besoins et attentes des enfants. Facile d'accès et attrayante, elle leur propose un apprentissage du code ludique.

Comprendre l'informatique

Loliplop permet un apprentissage non pas de la programmation classique mais plutôt de la pensée informatique dès le plus jeune âge en éveillant, en particulier, la logique et la créativité de l'enfant.
Aujourd'hui apprendre la logique de la programmation c'est primordial pour comprendre le monde qui nous entoure, car on retrouve de la programmation dans tous les secteurs, de la simple machine à laver jusqu'à nos smartphones; tous les outils que nous utilisons chaque jour ont un rapport à l'informatique.

C'est grâce au concept de la programmation visuelle, qui représente le code sous la forme de blocs de couleurs à imbriquer ensemble pour créer une histoire, que Loliplop permet une initiation pas à pas des concepts de base de la programmation. Le but étant de déduire les instructions logiques que l'on retrouve dans la programmation classique.
Comme par exemple faire avancer le personnage, le faire tourner, le faire sauter, lui faire jouer des sons, changer d'apparences, tout ceci de façon simple, intuitive et sans une ligne de code.

Apprendre à coder devient un jeu d'enfants

Aujourd'hui, il existe plusieurs plateformes pour s'initier à la programmation, la plus connue étant Scratch « Imagine, Programme, Partage » développée par le MIT. Elle regroupe plus de 9 millions de projets développés et partagés par de jeunes utilisateurs.
Scratch a pour objectif l'enseignement de l'univers informatique aux enfants, de la même façon que Loliplop, il aborde les concepts de base de la programmation, tels que : les boucles, les tests, les affectations de variables, etc.
Son mode créatif permet aux enfants de créer leur propre contenu interactif, comme par exemple, une carte d'anniversaire ou un jeu de labyrinthe, et de le partager directement depuis la plateforme à leurs amis ou à la communauté.
L'application Loliplop offre une prise en main rapide pour tout type de profil, même les non-initiés à la programmation, grâce à son interface facile et intuitive.
Enfin, grâce aux ressources pédagogiques et à l'assistance pour les élèves, les parents, professeurs ou animateurs peuvent suivre en temps réel les progrès de l'enfant, à l'aide du dashboard de suivi des progrès
Utilisables à l'école, en périscolaire et à la maison, ces deux outils modifient la perception qu'ont les enfants de l'écran pour qu'ils ne soient plus seulement de simples consommateurs, mais plutôt pour qu'ils deviennent acteurs de leur environnement numérique.
C'est grâce à son deuxième mode, le mode défis, que Loliplop marque sa différence en intégrant certaines matières scolaires (français, anglais, mathématiques), et en s'adaptant au niveau de chaque enfant.
Sa prise en main est interactive car, elle immerge l'enfant dans le monde imaginaire de Loli et ses amis, dans lequel Loli introduit chaque nouveau bloc ou élément et explique comment l'utiliser. Loliplop c'est aussi un pack de ressources pédagogiques, de tutoriels et tableaux de suivi des progrès pour les parents, les professeurs et animateurs.
Enfin, avec son système de « rewarding » qui récompense les progrès des enfants après chaque concept acquis, Loliplop permet de développer une certaine implication dans leur apprentissage.
Loliplop a mis à disposition et en téléchargement gratuit, sur son site : www.loliplop.com une version Beta, qui ne cessera d'évoluer au fil des semaines, notamment grâce à vos retours.
Au début du mois de septembre Loliplop lancera une campagne Kickstarter pour sortir l'application, ce sera aussi l'occasion d'annoncer l'intégration de plusieurs nouvelles fonctionnalités.

Intégration du DataTables Jquery dans ASP.NET MVC 5

2e partie

Le « DataTables Jquery » est un outil open source JavaScript permettant l'affichage de données sous forme de tableau HTML au sein de votre application Web. (Suite de la 1ère partie)

Georges DAMIEN
Consultant .NET/WEB chez
Cellenza
Blog : www.georgesdamien.com

LE MODÈLE MVC MÉTIER

Nous créons donc, côté serveur, notre modèle "Browser" utilisé lors du premier article :

```
public class BrowserModel
{
    public string Engine { get; set; }
    public string Browser { get; set; }
    public string Platform { get; set; }
    public string Version { get; set; }
    public string Grade { get; set; }

    public BrowserModel() { }
    public BrowserModel(string engine, string browser, string platform, string version, string grade)
    {
        Engine = engine;
        Browser = browser;
        Platform = platform;
        Version = version;
        Grade = grade;
    }
}
```

A partir de ce modèle, nous allons voir comment interagir avec notre plugin afin de l'alimenter côté client.

LE MODÈLE DU DATATABLES

Côté serveur, le « DataTables Jquery » possède également un modèle, permettant de gérer la majorité des scenarii (filtre, pagination, etc). Vous trouverez via le lien ci-dessous les informations relatives à ce modèle :
http://legacy.datatables.net/usage/server-side
Côté C#, voici un exemple de modèle Mvc contenant quelques propriétés que nous jugeons utiles pour cette démo :

```
/// <summary>
/// Class that encapsulates most common parameters sent by DataTables plugin
/// </summary>
public class DataTableParamModel
{
    /// <summary>
    /// Request sequence number sent by DataTable,
    /// same value must be returned in response
    /// </summary>
    public string sEcho { get; set; }

    /// <summary>
    /// Text used for filtering
    /// </summary>
    public string sSearch { get; set; }

    /// <summary>
    /// Number of records that should be shown in table
    /// </summary>
    public int iDisplayLength { get; set; }

    /// <summary>
    /// First record that should be shown(used for paging)
    /// </summary>
    public int iDisplayStart { get; set; }

    /// <summary>
    /// Number of columns in table
    /// </summary>
    public int iColumns { get; set; }

    /// <summary>
    /// Number of columns that are used in sorting
    /// </summary>
    public int iSortingCols { get; set; }

    /// <summary>
    /// Comma separated list of column names
    /// </summary>
    public string sColumns { get; set; }
}
```

Nous verrons plus loin qu'il existe une quantité assez large de propriétés liées au « DataTables Jquery ».

CHARGEMENT ASYNCHRONE

Côté client

Côté client (HTML), nous allons encore une fois utiliser l'appel asynchrone intégré à la librairie afin de charger notre « DataTables » :

```
$(document).ready(function () {

    var oTable = $('#exampleWithAjax');

    oTable.dataTable({
        "oLanguage": {
            "sProcessing": "Traitement en cours...",
            "sSearch": "Rechercher :",
            "sLengthMenu": "Afficher _MENU_ &eacute;l&eacute;ments",
```

```
        "sInfo": "Affichage de l'&eacute;lement _START_ &agrave; _END_ sur _TOTAL_ &eacute
;l&eacute;ments",
        "sInfoEmpty": "Affichage de l'&eacute;lement 0 &agrave; 0 sur 0 &eacute;l&eacute;ments",
        "sInfoFiltered": "(filtr&eacute; de _MAX_ &eacute;l&eacute;ments au total)",
        "sInfoPostFix": "",
        "sLoadingRecords": "Chargement en cours...",
        "sZeroRecords": "Aucun &eacute;l&eacute;ment &agrave; afficher",
        "sEmptyTable": "Aucune donn&eacute;e disponible dans le tableau",
        "oPaginate": {
          "sFirst": "Premier  ",
          "sPrevious": "Pr&eacute;c&eacute;dent  ",
          "sNext": "Suivant",
          "sLast": "  Dernier"
        },
        "oAria": {
          "sSortAscending": ": activer pour trier la colonne par ordre croissant",
          "sSortDescending": ": activer pour trier la colonne par ordre d&eacute;croissant"
        }
      },
      // activate Ajax call
      "bServerSide": true,
      // show loader
      "bProcessing": true,
      // Ajax call
      "sAjaxSource": "/Home/LoadBrowsers",
      "columns": [
        { "title": "Engine" },
        { "title": "Browser" },
        { "title": "Platform" },
        { "title": "Version", "class": "center" },
        { "title": "Grade", "class": "center" }
      ]
    });
});
```

Ici, on a spécifié que l'on souhaite charger les données depuis le serveur via la propriété "bServerSide", et nous spécifions la méthode « serveur » à appeler : "sAjaxSource": "/Home/LoadBrowsers".

FONCTIONNEMENT

Une fois le « DataTables » instancié, un appel asynchrone sera émis vers le serveur afin de charger les données dans notre Tableau. Chaque évènement observé engendrera également un appel vers le serveur afin de rafraîchir notre tableau en conséquence (par exemple lors d'une recherche, d'un tri, d'une pagination, etc).

Côté Serveur

Au niveau serveur (Controller MVC), voyons à quoi ressemble notre fonction de chargement du « DataTables » :

```
    [HttpGet]
    public JsonResult LoadBrowsers(DataTableParamModel param)
    {
      // Browser list
      var totalBrowserList = SrvBrowser.ListOfBrowsers();

      var filteredBrowserList = totalBrowserList.Skip(param.iDisplayStart).Take(param.i
DisplayLength);
      var filteredResult = from b in filteredBrowserList
                   select new[] { b.Engine, b.Browser, b.Platform, b.Version, b.Grade };

      return Json(new
      {
        sEcho = param.sEcho,
        iTotalRecords = totalBrowserList.Count(),
        iTotalDisplayRecords = totalBrowserList.Count(),
        aaData = filteredResult
      },
       JsonRequestBehavior.AllowGet);
    }
```

Nous considérons ici que la liste d'éléments à afficher (les browsers) nous est fournie par un service tiers SrvBrowser (instance de ServiceBrowser (cf projet Git)) via la méthode « ListOfBrowsers ».
Si nous observons le contenu de l'objet « DataTableParamModel » passé en paramètre, nous constatons qu'il contient bien les données du modèle associé sans que nous n'ayons eu à renseigner ces éléments côté client.
Notre méthode retourne donc un objet Json contenant :

- aaData : la liste des éléments à afficher par page
- sEcho : le numéro de la page courante
- iTotalRecords : le nombre total d'éléments à afficher
- iTotalDisplayRecords : le nombre courant d'éléments à afficher par page
- sSearch : le mot clé du champ de recherche

A partir de ces propriétés, nous pouvons gérer notre chargement côté serveur, ainsi que la pagination et le filtre d'affichage de lignes prédéfinies. Cependant, dans cet exemple, nous n'effectuons aucun filtre de recherche ni de tri par colonne. Il faut également gérer ces fonctionnalités côté serveur. Voyons le code correspondant pour pallier cela :

```
[HttpGet]
    public JsonResult LoadBrowsersFull(DataTableParamModel param, string bwsrCreatedDate)
    {
      // Browser list
      var totalBrowserList = SrvBrowser.ListOfBrowsers();
      IEnumerable<BrowserModel> filteredBrowserList;

      // filter keyword search
      if (!string.IsNullOrEmpty(param.sSearch))
      {
        // limit search in column 1 and 2
        var isBrowserSearchable = Convert.ToBoolean(Request["bSearchable_1"]);
        var isEngineSearchable = Convert.ToBoolean(Request["bSearchable_0"]);

        filteredBrowserList = totalBrowserList.Where(bwsr =>
                isBrowserSearchable && Convert.ToString(bwsr.Browser).Contains(param.
sSearch.ToLower()) ||
                isEngineSearchable && bwsr.Engine.ToLower().Contains(param.sSearch.
ToLower()));
      }
      else
      {
        // No search filter
        filteredBrowserList = totalBrowserList;
      }

      var isBrowserPlatformSortable = Convert.ToBoolean(Request["bSortable_2"]);
      var sortColumnIndex = Convert.ToInt32(Request["iSortCol_0"]);
```

```
        Func<BrowserModel, string> orderingFunction = (bwsr =>
                    sortColumnIndex == 2
                    && isBrowserPlatformSortable ? Convert.ToString(bwsr.Platform) : "");


        var sortDirection = Request["sSortDir_0"];

        // asc or desc ofr column Platform
        filteredBrowserList = sortDirection == "asc"
          ? filteredBrowserList.OrderBy(orderingFunction)
          : filteredBrowserList.OrderByDescending(orderingFunction);

        var displayedFilteredBrowserList = filteredBrowserList.Skip(param.iDisplayStart).Take
(param.iDisplayLength);

select new[] { b.Engine, b.Browser, b.Platform, b.Version, b.Grade };

        return Json(new
        {
          sEcho = param.sEcho,
          iTotalRecords = totalBrowserList.Count(),
          iTotalDisplayRecords = totalBrowserList.Count(),
          aaData = filteredResult
        },
         JsonRequestBehavior.AllowGet);
    }
```

Au passage, nous constatons que nous pouvons récupérer d'autres champs du "DataTable Param" dans la "Request" comme, par exemple, les colonnes à filtrer, à trier, les index des colonnes, etc. Bien que la liste des éléments paramétrables du « DataTables Jquery » soit longue et assez complète, nous pourrions avoir envie d'ajouter des filtres personnalisés : là encore le plugin est assez souple pour le permettre. Il suffira d'ajouter côté client les paramètres voulus via la propriété du DataTables "fnServerParams". Imaginons un filtre global extérieur à notre DataTables, comme, par exemple, une liste de dates de création des navigateurs que nous ne souhaitons pas voir apparaître dans notre tableau; cela nous donnera côté Js :

```
// ---------------
    'fnServerParams': function (aoData) {
      aoData.push({ 'name': 'bwsrCreatedDate', 'value': $('#bwsrCreatedDate').val() }
      );
    },
```

Côté Serveur, il suffira de rajouter le paramètre en entrée de la méthode de filtre et d'y appliquer ce que nous souhaitons comme filtre :

```
[HttpGet]
public JsonResult LoadBrowsersFull(DataTableParamModel param, string bwsrCreatedDate)
```

OPTIMISATION SQL

Nous avons allégé le chargement de notre tableau en affichant un nombre limité d'enregistrements notamment via les filtres de lignes d'affichage et la pagination du « DataTables ». Mais imaginons des données provenant d'une table en base de données contenant des millions de lignes... On pourra, dans ce cas, déléguer à une procédure stockée par exemple tout ou partie du filtre et chargement de notre « DataTables ». Les paramètres de filtre étant déjà passés en paramètre côté serveur, il nous suffira de les relayer à notre procédure stockées afin d'effectuer les filtres/tri/recherche par mot clé (recherche avancée si besoin), etc .
Voici à quoi pourrait ressembler notre requête SQL sous SQL SERVER :

```
CREATE PROCEDURE [dbo].[Browser_Search]
   @searchKey varchar(256) = NULL,
   @skipRows int = 0,
  @takeRows int = 10,
  @count int = 0

AS

BEGIN
   SET NOCOUNT ON;

   DECLARE @TotalCount INT
   SELECT  @TotalCount = COUNT(BrowserId)
   FROM TblBrowser

   SELECT *
   FROM
   (
      SELECT   ROW_NUMBER() OVER(ORDER BY BrowserId) AS NUMBERROW,
            Engine,
            Browser,
            Platform,
            Version,
            Grade,
            @TotalCount as TotalCount
      FROM TblBrowser
      WHERE
        (
            @searchKey IS NULL
            OR
            (
                LOWER(Engine) LIKE '%' +@searchKey + '%'
              OR LOWER(Browser) LIKE '%' +@searchKey + '%'
              OR LOWER(Platform) LIKE '%' +@searchKey + '%'
              OR LOWER(Version)  LIKE '%' +@searchKey + '%'
              OR LOWER(Grade)LIKE '%' +@searchKey + '%'
            )
        )
      ORDER BY BrowserId DESC

   ) AS TBL
   WHERE NUMBERROW BETWEEN (@skipRows + 1) AND (@skipRows + @takeRows)

END
```

Conclusion

Le « DataTables Jquery » est un plugin complet qui nous permet d'afficher nos données à exploiter sur le Web sous forme de tableau. Il nous offre une certaine souplesse et beaucoup de facilités notamment avec la gestion native de filtres - tri - pagination - internationalisation - recherche par mot clé - appel asynchrone - etc.
Il reste également paramétrable, et « surchargeable » en termes de fonctionnalités. Une fois pris en main, il peut s'avérer être un réel atout pour les développeurs Web, en leur permettant de rendre disponible, ou non, tout ou partie d'un ensemble de fonctionnalités déjà embarquées. ⊞

Vous trouverez le code source de cet exemple sur mon git :
https://github.com/Djdodjes/DataTablesAspNetMvcDemo

Mettre en oeuvre des tests efficaces sous Android

Pratique essentielle de tout projet informatique, les tests doivent également occuper une place prépondérante dans tout processus de développement d'applications Android. Si, de par sa nature spécifique, la plateforme présente certaines difficultés pour la mise en place de tests, des solutions existent afin de mettre en oeuvre une réelle stratégie de tests. Des tests unitaires aux tests fonctionnels, passage en revue des principaux outils disponibles à l'heure actuelle.

Sylvain SAUREL
Ingénieur d'Etudes Java / Java EE
sylvain.saurel@gmail.com

Garantir le bon fonctionnement d'une application au cours du temps tout en évitant les régressions a toujours été une problématique centrale de l'informatique. Le domaine des smartphones n'échappe évidemment pas à la règle et compte tenu des multiples contraintes de cet univers, les tests y jouent un rôle plus important encore. Si les tests fonctionnels ont toujours été un passage obligé, l'essor des méthodologies agiles a remis sur le devant de la scène les tests unitaires en faisant un passage obligé. Que les applications soient développées en TDD ou non d'ailleurs. Bien souvent, les développeurs souhaitant utiliser massivement les tests unitaires ou d'intégration se heurtent à la même problématique : l'API android.jar récupérée sur son poste développement est vide ! En effet, plutôt que de fournir le code en l'état, ce JAR ne propose que les classes et leurs méthodes sans aucun code fonctionnel à l'intérieur. Ainsi, tous les constructeurs et corps de méthodes sont des stubs de cette forme :

```
public class View {
 public View(Context context) {
  throw new RuntimeException("Stub!");
 }
...
}
```

Le contenu réel de l'API étant présent directement sur les smartphones.

JUnit

Dès lors, la mise en place de tests unitaires s'appuyant sur le framework JUnit et héritant de la classe TestCase produira une exception à l'exécution. Afin de prendre en compte les spécificités d'Android, le SDK fournit des implémentations de TestCase permettant l'exécution de tests unitaires, mais nécessitant le lancement d'un émulateur ou la connexion à un terminal physique. Ces implémentations vont permettre le test d'une activité via ActivityInstrumentationTestCase2, celui d'un service via ServiceTestCase, ou encore d'un contre provider via ProviderTestCase2. Pour mettre en place ces tests, nous utilisons une application réalisant la multiplication entre 2 nombres saisis par l'utilisateur (figure 1).

Fig.1 *Application à tester*

Séparer le code de l'interface utilisateur du code métier est quelque chose d'ardu sous Android du fait qu'une activité est attendue pour jouer le rôle d'un contrôleur et d'une vue à la fois. Autant que faire se peut il faut néanmoins bien séparer le code métier du code IHM. Ainsi, pour notre application, la logique de calcul est déportée au sein d'une classe Calculator proposant 2 méthodes multiply : une multipliant 2 nombres flottants, et l'autre multipliant 2 nombres flottants entrés en paramètres comme des chaînes de caractères.
Sous un IDE tel que Eclipse, il faut tout d'abord créer un projet Android de test en précisant qu'il s'appuie sur le projet SimpleCalculator qui est à tester. Reste ensuite à créer un test pour la méthode multiply :

```
public class CalculatorTest extends TestCase {
 private Calculator calculator;

 @Override
 protected void setUp() throws Exception {
  calculator = new Calculator();
 }

 public void testMultiply() {
  assertEquals(6.25f, calculator.multiply(1f, 6.25f));
 }
}
```

Aucun ajout de dépendances n'est nécessaire puisque le SDK embarque directement JUnit. Le lancement du test via le lanceur classique de JUnit provoque un crash, et il est donc impératif d'utiliser le lanceur Android spécifique pour l'exécuter. Las, l'exécution du test nécessite alors le lancement d'un émulateur à défaut d'une connexion avec un terminal ! En outre, l'APK de l'application est généré et installé sur le terminal avant de pouvoir exécuter le test qui utilise pourtant un objet indépendant d'Android.
Souhaitant aller un peu plus loin en réalisant un test d'intégration avec JUnit, nous choisissons de tester l'activité principale de l'application en laissant le soin au test automatisé de rentrer les nombres à multiplier avant de vérifier le résultat. L'API de test du SDK permet ces tests mais le code suivant montre clairement les difficultés de l'approche :

```
public class MainActivityTest extends ActivityInstrumentationTestCase2<MainActivity> {
 private MainActivity mainActivity;
 private EditText editText1;
 private EditText editText2;
 private Button button;
 private TextView result;

 public MainActivityTest() {
  super("com.ssaurel.calculator", MainActivity.class);
 }

 @Override
 protected void setUp() throws Exception {
  super.setUp();
  mainActivity = getActivity();
```

```
 editText1 = (EditText) mainActivity.findViewById(com.ssaurel.calculator.R.id.EditText01);
 editText2 = (EditText) mainActivity.findViewById(com.ssaurel.calculator.R.id.EditText02);
 button = (Button) mainActivity.findViewById(com.ssaurel.calculator.R.id.Button01);
 result = (TextView) mainActivity.findViewById(com.ssaurel.calculator.R.id.TextView01);
}

public void testMultiply() {
 mainActivity.runOnUiThread(new Runnable() {
  @Override
  public void run() {
   editText1.setText("");
   editText1.requestFocus();
  }
 });

 getInstrumentation().waitForIdleSync();
 sendKeys(KeyEvent.KEYCODE_1);

 mainActivity.runOnUiThread(new Runnable() {
  @Override
  public void run() {
   editText2.setText("");
   editText2.requestFocus();
  }
 });

 getInstrumentation().waitForIdleSync();
 sendKeys(KeyEvent.KEYCODE_2);

 mainActivity.runOnUiThread(new Runnable() {
  @Override
  public void run() {
   button.requestFocus();
   button.performClick();
  }
 });

 getInstrumentation().waitForIdleSync();
 assertEquals("2.0", result.getText());
 }
}
```

Ce code se révèle affreusement verbeux pour le peu de traitements réalisés. La contrainte principale étant la nécessité d'exécuter les actions IHM au sein de l'UI Thread via l'appel à la méthode runOnUiThread de l'activité instrumentalisée puis d'attendre son exécution. On se doute également que le code pour des interactions entre activités est encore plus lourd. Bref, JUnit à la sauce Android n'est pas vraiment la meilleure solution pour réaliser des suites de tests performantes compte tenu de la lenteur d'exécution et de la lourdeur de l'API.

Robolectric

Afin d'adresser cette problématique et autoriser le TDD sous Android, le framework Robolectric a été mis au point. Disponible en version 2.2 ici : http://robolectric.org, il rend possible l'exécution de tests unitaires sous Android réellement indépendants de la plateforme. Pour réaliser cette prouesse, Robolectric redéfinit l'ensemble des classes du SDK Android permettant ainsi leur exécution au sein d'une JVM classique. Les méthodes étant mockées ou implémentées de façon à reproduire un comportement identique. L'idéal pour utiliser la bibliothèque sur un projet est d'utiliser Maven pour le build et d'ajouter les dépendances suivantes à son POM :

```
<dependencies>
 <dependency>
  <groupId>junit</groupId>
  <artifactId>junit-dep</artifactId>
  <version>4.8.2</version>
  <scope>test</scope>
 </dependency>

 <dependency>
  <groupId>org.robolectric</groupId>
  <artifactId>robolectric</artifactId>
  <version>2.2</version>
  <scope>test</scope>
 </dependency>

 <dependency>
  <groupId>android</groupId>
  <artifactId>android</artifactId>
  <version>4.3_r2</version>
  <scope>provided</scope>
 </dependency>
</dependencies>
```

Une fois les dépendances ajoutées, nous pouvons passer à l'écriture d'un test d'intégration pour la calculatrice :

```
@RunWith(RobolectricTestRunner.class)
public class IntegrationTest {

 @Test
 public void testMultiply() throws Exception {
  MainActivity mainActivity = Robolectric.buildActivity(MainActivity.class).create().get();
  EditText editText1 = (EditText) mainActivity.findViewById(R.id.EditText01);
  EditText editText2 = (EditText) mainActivity.findViewById(R.id.EditText02);
  Button button = (Button) mainActivity.findViewById(R.id.Button01);
  TextView result = (TextView) mainActivity.findViewById(R.id.TextView01);
  editText1.setText("1");
  editText2.setText("2");
  button.performClick();
  Assert.assertEquals("2.0", result.getText());
 }
}
```

La classe est annotée avec @RunWith afin de spécifier l'utilisation du runner JUnit RobolectricTestRunner pour le lancement des tests. Au sein de testMultiply, on instancie l'activité via Robolectric et effectue les actions IHM avant de vérifier la conformité du résultat obtenu. L'exécution des tests se faisant via Maven avec un classique *mvn* clean test. Simple d'emploi, Robolectric sélectionne le bon runtime Android à l'exécution. En sus, ses possibilités sont nombreuses puisqu'il permet le chargement de layouts de même que celui des vues ou la création d'activités. Supprimant le recours à un émulateur, Robolectric donne la possibilité aux développeurs de mettre en place des suites de tests réellement performantes s'exécutant en quelques secondes ouvrant de fait la voie au TDD sous Android. Il est bon de noter que des bibliothèques de mock classiques du

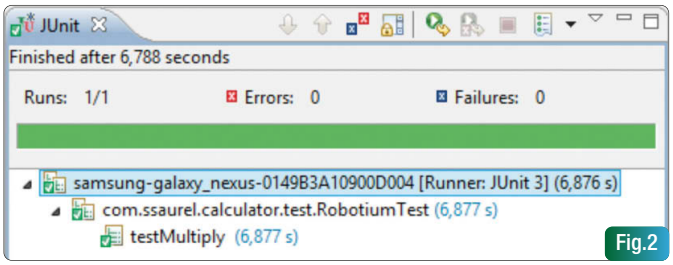
monde Java telles que Mockito ou Powermock peuvent être employées mais elles ne vont pas aussi loin dans l'intégration avec Android que Robolectric.

Robotium

Autre outil de choix dans le domaine des tests sous Android, le framework Robotium permet de réaliser des tests d'intégration boîte blanche mais également boîte noire. Ce dernier point est intéressant puisque cela permet d'automatiser le test d'une application à partir de son APK. Là encore, Robotium est un produit open source en version 4.3 récupérable ici : http://code.google.com/p/robotium/. Robotium s'appuie sur les classes d'instrumentation proposées par le SDK avec une simplicité d'écriture apportant un net gain de productivité. Tous les scénarii de tests sont envisageables et il est aisé d'implémenter des scénarii complexes mettant en jeu des enchaînements d'activités. Une fois la bibliothèque ajoutée à notre projet, nous écrivons l'équivalent du test proposé précédemment :

```
public class RobotiumTest extends ActivityInstrumentationTestCase2<MainActivity> {

 private Solo solo;

 public RobotiumTest() {
  super("com.ssaurel.calculator", MainActivity.class);
 }

 protected void setUp() throws Exception {
  super.setUp();
  solo = new Solo(getInstrumentation(), getActivity());
 }

 protected void tearDown() throws Exception {
  solo.finishOpenedActivities();
  super.tearDown();
 }

 public void testMultiply() {
  solo.assertCurrentActivity("Main Activity", MainActivity.class);
  solo.clearEditText(0);
  solo.typeText(0, "1");
  solo.clearEditText(1);
  solo.typeText(1, "2.65");
  solo.clickOnButton(solo.getString(R.string.multiplication));
  assertEquals("2.65", solo.getText(5).getText().toString());
 }
}
```

Au coeur de Robotium se trouve la classe Solo permettant d'interagir avec l'application et ses composants UI. A partir d'elle, il est possible de simuler n'importe quel évènement IHM de manière aisée. L'accès aux éléments UI se fait par index ou par libellé en utilisant les constantes strings par exemple. Puisque basé sur la classe ActivityInstrumentationCase2 de l'OS, le test nécessite un émulateur ou un terminal pour s'exécuter, ce qui implique un temps d'exécution de l'ordre de quelques secondes (figure 2). D'autre part, Robotium se révèle être le compagnon idéal pour la mise en place de tests fonctionnels boîte noire. En effet, si les tests réalisés ici s'appuyaient sur les sources du projet, il est possible de réaliser le même type de tests avec comme point d'entrée l'APK d'une application à condition d'utiliser la même clé que celle-ci. Bien entendu, les actions sur les éléments IHM ne peuvent s'effectuer que via leur index. Attention dès lors à ne pas superposer des composants, ce qui les rendrait inaccessibles.



Exécution du test avec Robotium

Monkey

Toujours au rayon tests fonctionnels boîte noire, le SDK met à disposition l'utilitaire Monkey scindé en 2 parties : l'outil ligne de commande Monkey conçu pour stresser une application via l'envoi d'évènements UI aléatoires à intervalles de temps donnés ou aussi vite que possible. L'outil se révèle intéressant pour mettre en lumière des bugs bien particuliers tels que les ANR (Application Not Responding). La seconde partie est le MonkeyRunner qui est une version plus sophistiquée de l'outil autorisant la création de scénarii de tests via des scripts Python. Concrètement, le MonkeyRunner se compose de 3 classes :

- MonkeyRunner, point d'entrée de l'outil pour interagir avec le périphérique
- MonkeDevice proposant les méthodes d'entrée pour agir sur le terminal
- MonkeyImage, classe spécifique pour obtenir des images de type screenshot et les comparer

Un script MonkeyRunner a l'allure suivante :

```
from com.android.monkeyrunner import MonkeyRunner, MonkeyDevice

monkeyDevice = MonkeyRunner.waitForConnection()
monkeyDevice.installPackage('bin/SimpleCalculator.apk')
package = 'com.ssaurel.calculator'
activity = 'com.ssaurel.calculator.MainActivity'
runComponent = package + '/' + activity
monkeyDevice.startActivity(component=runComponent)
```

Ici, on réalise l'installation de l'APK sur le terminal courant. Notons qu'il est possible de spécifier un terminal en particulier. Une fois l'application chargée, on lance l'activité MainActivity de SimpleCalculator. Reste ensuite à écrire les scénarii de tests. L'outil n'ayant pas accès aux ressources de l'application, ceux-ci se font en naviguant au sein de l'IHM à l'aide des commandes DPAD_UP, DPAD_DOWN, DPAD_LEFT et DPAD_RIGHT avant de cliquer sur l'élément désiré via DPAD_ENTER. Ainsi, pour sélectionner le premier élément de l'UI, les lignes suivantes sont nécessaires :

```
monkeyDevice.press("DPAD_DOWN", MonkeyDevice.DOWN_AND_UP);
monkeyDevice.press("DPAD_ENTER", MonkeyDevice.DOWN_AND_UP);
monkeyDevice.type("1");
```

L'outil permet également de cliquer en un point particulier de l'écran via la commande suivante :

```
monkeyDevice.touch(x, y, MonkeyDevice.DOWN_AND_UP);
```

Malgré d'évidentes possibilités, cet outil se révèle difficile d'accès de par une API contraignante pour mettre en place des actions simples. Néanmoins, il peut être intéressant de le considérer pour certains cas d'usages.

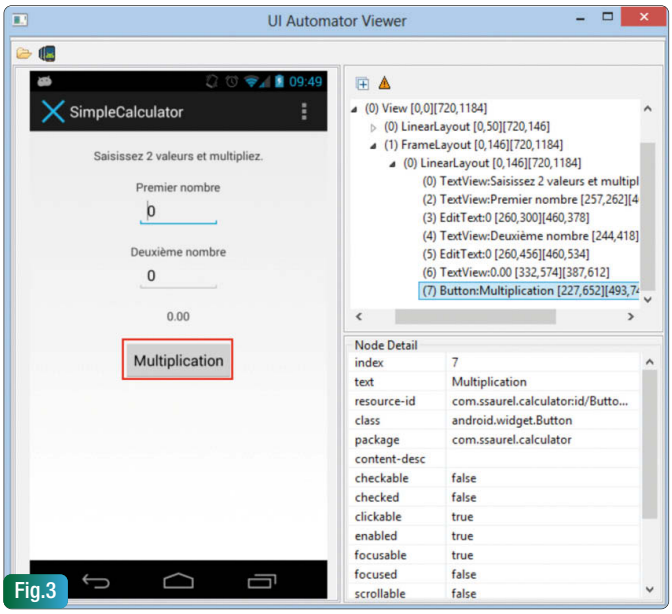


Fig.3

UI Automator Viewer

UI Automator

Ajouté aux outils standards du SDK bien après l'outil Monkey, le framework de test UI Automator sert à implémenter des tests IHM efficaces. La première étape consiste à analyser le contenu de l'IHM cible pour les tests. Pour ce faire, le SDK propose l'UI Automator Viewer qui est un petit utilitaire présent au sein du dossier tools/ du SDK. Une fois lancé (figure 3), il permet d'accéder à la hiérarchie des composants de l'UI en ayant pour chacun d'entre eux tout un tas de détails essentiels lors de l'écriture des tests.

L'outil sert de support lors de l'écriture des tests fonctionnels. Une autre étape importante consiste à vérifier que l'application à tester est bien accessible, c'est-à-dire qu'elle définit les éléments *android:contentDescription* pour des vues telles que ImageButton, ImageView ou Checkbox par exemple. En outre, chaque élément doit être accessible via un trackball puisque l'UI Automator se déplacera ainsi entre chaque composant.

Ces vérifications réalisées, il faut créer un projet Java sous Eclipse en ajoutant la dépendance vers JUnit 3 ainsi que l'android.jar et l'API de l'UI Automator. Nommée uiautomator.jar, elle est disponible à la racine du répertoire cible du SDK considéré en bonne place à côté de l'android.jar . Le coeur de l'API tourne autour de 3 classes : UiDevice modélisant l'état du périphérique, UiSelector représentant un critère de recherche sur l'écran courant et UiObject représentant un élément de l'IHM. Le test simulant le lancement de l'application SimpleCalculator a ainsi l'allure suivante :

```
public class UiAutomatorTest extends UiAutomatorTestCase {
 public void testDemo() throws UiObjectNotFoundException {
  getUiDevice().pressHome();
  UiObject allAppsButton = new UiObject(new UiSelector().description("Apps"));
  allAppsButton.clickAndWaitForNewWindow();

  UiObject appsTab = new UiObject(new UiSelector().text("Apps"));
  appsTab.click();

  UiScrollable appViews = new UiScrollable(new UiSelector().scrollable(true));
  appViews.setAsHorizontalList();

  UiObject settingsApp = appViews.getChildByText(new UiSelector().className(android.widget.
TextView.class.getName()), "SimpleCalculator");
  settingsApp.clickAndWaitForNewWindow();

  UiObject appValidation = new UiObject(new UiSelector().packageName("com.ssaurel.
calculator"));
  assertTrue("Unable to detect Simple Calculator", appValidation.exists());
 }
}
```

Cette approche se révèle assez puissante en permettant de simuler le comportement sur un terminal. La difficulté majeure résidant dans le build et le déploiement. En effet, il n'est pas possible d'exécuter ces tests directement via Eclipse par exemple. Il faut d'abord créer le projet de tests UI embarquant les classes de tests, le construire avec ant, puis pousser vers le terminal cible le JAR correspondant au projet de test :

```
cd <your/project/path>
android create uitest-project -n UiAutomatorTest -t 1 -p .
ant build
adb push ./bin/UiAutomatorTest.jar /data/local/tmp/
```

Enfin, il ne reste plus qu'à exécuter les tests sur le périphérique cible en lançant la commande suivante :

```
adb shell uiautomator runtest UiAutomatorTest.jar -c com.ssaurel.calculatortest.UiAutomatorTest
```

Conclusion

En présentant plusieurs outils, cet article aura mis en avant les différents tests qu'il est possible d'implémenter sous Android. Aucun outil n'est meilleur qu'un autre, mais chacun apporte son lot d'avantages et d'inconvénients qu'il conviendra donc de bien mesurer pour adopter l'outil le plus adéquat selon son contexte. Ainsi, s'appuyer sur Robolectric pour la mise en place de tests unitaires ou d'intégration réellement indépendants de l'OS est une bonne pratique ouvrant la voie au TDD sous Android. Rajouter en sus des tests d'intégration boîte blanche ou fonctionnels boîte noire en tirant profit de la puissance de Robotium. Et enfin s'appuyer sur un framework comme l'UI Automator pour des scénarii de tests spécifiques. En outre, utiliser un outil de build tel que Maven ou Gradle est une bonne pratique qui permettra d'automatiser l'exécution des tests au sein d'une plateforme d'intégration continue. Le tout étant de concilier pertinence des tests et performance afin d'augmenter la qualité des applications Android.

Répondre à un appel d'offres avec une approche par les modèles

Le module eaXL d'eaDocX permet de synchroniser des informations stockées au format Excel avec un projet de modélisation Sparx Systems Enterprise Architect, par exemple pour importer et exporter des exigences, classes métiers, cas d'utilisations, etc.

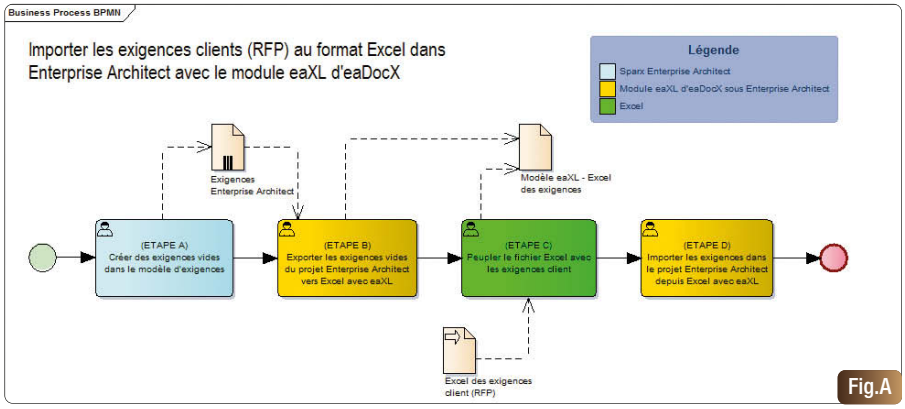


Fig.A

Guillaume FINANCE
VISEO
guillaume.finance@viseo.com

Cet article présente un retour d'expérience (REX) sur l'utilisation d'eaXL dans le cadre d'une réponse à appel d'offres, où les nombreuses exigences du client ont été importées dans un projet Enterprise Architect afin de faciliter l'analyse des besoins et l'élaboration d'une réponse avec une approche par les modèles.

Contexte

J'ai récemment travaillé sur l'appel d'offre d'un client comportant plus d'une centaine d'exigences organisées par catégories. La description, référence, et catégorie de chaque exigence ont été fournies au format Excel dans le but de compléter ce document avec nos réponses et propositions (livrable pour le client). Ayant la tâche d'étudier chacune des exigences vis à vis de la faisabilité, des efforts nécessaires, et des solutions à proposer, j'ai naturellement choisi d'effectuer ce travail en m'appuyant sur un projet de modélisation Enterprise Architect. Une réponse à appel d'offres nécessite de travailler sur un large périmètre de demandes et d'attentes ; une approche par les modèles (model-driven) présente l'avantage de faciliter cette tâche fastidieuse en centralisant l'ensemble des informations, diagrammes et réponses dans un référentiel **Fig.A**.

Etape A : créer des exigences vides dans le modèle

A1. Après avoir initialisé un projet Enterprise Architect adapté aux besoins de cette étude, le modèle des exigences est rempli avec quelques exigences vides (cette étape est nécessaire afin de permettre à eaXL de générer le modèle Excel par l'export des exigences présentes).

A2. Des informations complémentaires sont créées sur les exigences via les « tagged values » d'Enterprise Architect : Feasible (réalisable : oui/non), Priority, et Category.

Etape B : exporter les exigences du modèle dans Excel avec eaXL

B1. Afin de générer un fichier Excel compatible avec eaXL pour le compléter avec les exigences du client, les exigences du modèle Enterprise Architect sont exportées vers Excel avec eaXL (clic droit sur le paquetage cible > Extensions > eaDocX > Open in Excel).

B2. Des propriétés supplémentaires sont rajoutées dans l'export en cochant les options suivantes sous l'onglet Columns d'eaXL : Name, Alias, Description, et les "Tagged Values" Feasible, Priority, et Category.

B3. Le contenu du modèle sous Excel est mis à jour en cliquant sur le bouton « Export data from EA to current worksheet ».

Etape C : remplir le fichier Excel avec les exigences client

C1. A partir du fichier Excel fourni par le client, l'ensemble des exigences est copié dans une nouvelle feuille de calcul du fichier Excel généré avec eaXL.

C2. En utilisant des références entre les deux feuilles de calcul, la première feuille de calcul, utilisée pour l'import/export entre Excel et EA, est peuplée avec les exigences client dont les informations sont réparties sous les colonnes définies par eaXL.

Etape D : importer les exigences du client dans Enterprise Architect

D1. Sous Enterprise Architect, eaDocX/eaXL permet en premier lieu de lancer une comparaison avant de valider les exigences à importer dans les modèles.

D2. L'import depuis Excel vers le modèle Enterprise Architect est alors lancé via le bouton « Import contents of worksheet into EA ».

Résultat : eaDocX a importé les exigences dans le modèle avec l'ensemble des informations :

D3. A ce stade, les exigences vides initialement créées dans l'étape A peuvent être supprimées.

Etapes suivantes : réponses et propositions aux exigences, génération du livrable Excel

Disposant alors de toutes les exigences du client dans un référentiel de modélisation, l'analyse des besoins et l'élaboration de la réponse à appel d'offres peuvent être effectuées :

- Création d'exigences stéréotypées "réponse" et d'éléments de type Feature pour les fonctions du système. Association de ces éléments aux exigences.
- Création de diagrammes UML et BPMN pour proposer une architecture ou illustrer des solutions pour une ou plusieurs exigences.
- Saisie d'une valeur dans la tagged value "Feasible" de chaque exigence afin de définir la faisabilité.
- Export des résultats dans un fichier Excel avec eaXL.
- Génération d'un document "réponse à appel d'offre" au format Word avec eaDocX.

Conclusion

L'utilisation d'Enterprise Architect pour analyser une longue liste d'exigences a permis de construire un modèle structuré avec les éléments de réponse et les liens entre exigences comportant des similarités ou des dépendances. Une fois toutes les exigences analysées, générer le tableau Excel pour le client ne nécessite que quelques minutes!

Ce document Excel a été envoyé en annexe du document Word généré avec eaDocX, comportant notamment une matrice des associations entre les exigences et les fonctions du système à réaliser.

Dans le cas où la réponse à appel d'offres est retenue, cette approche présente l'avantage de disposer dès le début d'un modèle Enterprise Architect avec l'ensemble des exigences initiales du client, ainsi que les diagrammes et éléments de réponse élaborés.

Je pratique le C++

Partie 1/5

Nous vous proposons une série d'articles sur la pratique de C++ pour que vous puissiez tous vous y mettre. Pas besoin de débourser un centime pour pratiquer car tout est gratuit. Que ce soit l'ancien Visual Studio Express, Visual Studio Community Edition 2013 ou GCC, ou CLang via Linux ou MinGW, ça ne coûte rien. L'essentiel est d'avoir un compilateur récent pour bénéficier du C++ 11 alias le C++ Moderne. C++ 11 apporte énormément de changement à C++ 03 et C++05TR1, et c'est la raison pour laquelle il y a un engouement énorme autour de C++ 11. C'est la renaissance de C++. C++ est le langage incontournable pour faire des applications sur téléphone Android, iOS, Windows Phone, mais aussi pour faire des applications sur Windows, Linux et Mac. Et pourtant on n'en fait pas beaucoup de publicité.

Christophe PICHAUD | .NET Rangers by Sogeti
Consultant sur les technologies Microsoft
christophepichaud@hotmail.com | www.windowscpp.net

C++ est un langage normalisé par un comité ISO qui améliore constamment le langage et la librairie standard STL (Standard Template Library). Le comité ISO est en préparation de la version C++17 (qui fait suite aux versions C++11, C++14).

Les types prédéfinis

Il existe de nombreux types prédéfinis :

Type	Sens	Taille minimum
bool	Booléen. true ou false.	NA
char	Charactère	8 bits
wchar_t	Charactère large	16 bits
char16_t	Charactère Unicode	16 bits
char32_t	Charactère Unicode	32 bits
short	Entier court	16 bits
Int	Entier	16 bits
long	Entier long	32 bits
long long	Entier long	64 bits
float	Flottant à précision simple	6 digits
double	Flottant à double précision	10 digits
long double	Flottant à double précision	10 digits

Les types peuvent être signés ou non via le specifier signed ou unsigned. Les types sont tous signés par défaut. La taille d'une variable peut être obtenue via sizeof(t).

Les variables

Une variable se déclare comme suit :
<type> nom_de_variable = valeur_d'initialisation ;
L'initialisation n'est pas obligatoire mais si elle est manquante, le contenu de la variable est indéfini. En mode DEBUG, ça vaudra 0 ou null mais en RELEASE c'est indéfini. Conseil : il faut initialiser ses variables.

Les références

Une référence pointe sur un objet. C'est une référence vers quelque chose. C'est un alias.

```
void Sample_References()
{
   int a = 10;
   int &b = a;
   a++;
   cout << "b=" << b << endl;
}
```

L'affichage sera : b=11. Les références sont comme des pointeurs mais avec un pouvoir limité. (ça gratte ?).

Les pointeurs

Un pointeur sert à désigner un espace mémoire qui pointe vers quelque chose, un objet ou une variable. C'est comme une référence sauf qu'il n'est pas obligé d'être initialisé à la déclaration.

```
void Sample_Pointer()
{
   int a = 10;
   int * ptr = nullptr;
   ptr = &a;
   a++;
   cout << "ptr adress=" << ptr << " value=" << *ptr << endl;
}
```

L'affichage sera : ptr adress=0x0062FE20 value=11
Dans cet exemple, on initialise le pointeur à nullptr. Ensuite on indique que ptr vaut l'adresse (via &) de a. La valeur de a est son emplacement mémoire et le contenu du pointeur est accessible via *.

const

Utiliser const sur une variable empêche cette variable d'être modifiée. Cependant, elle doit être initialisée à la déclaration.

```
void Sample_Const()
{
   const int a = 100;
   a++;    // error
   a = 200;// error
}
```

Par défaut, les objets const sont locaux à un fichier .cpp pour la compilation. Pour définir et partager un objet const à plusieurs, il faut déclarer cette variable comme extern sinon le compilateur indiquera ne pas trouver la variable avec un message du style « var is undefined. ».

Références const

Il existe aussi des références const. Ce sont des références que l'on ne peut pas changer.

```
void Sample_ConstRef()
{
   int a = 10;
   const int & aref = a;
   int b = 20;
   aref = b; // error aref est const
}
```

Alias de types

Il est possible de définir un alias sur un type. Pourquoi ? C'est pour rendre le programme plus lisible.

```
typedef std::string String;
typedef vector<string> MesFilles;
typedef vector<string>::iterator IT;

void Sample_Typedef()
{
   String maggie = "je m'appelle Maggie et j'ai bientôt 5 ans et j'aime Papa";
   cout << maggie << endl;

   MesFilles girls;
   girls.push_back("Edith");
   girls.push_back("Lisa");
   girls.push_back("Audrey Maggie");
   for (IT it = begin(girls); it != end(girls); ++it)
   {
      cout << *it << endl;
   }
}
```

Dans cet exemple, j'ai créé 3 alias : sur std ::string, sur le vector<string> et sur son iterator.

Le mot-clé auto

Disponible depuis C++ 11, le mot clé auto permet de laisser le compilateur déduire le type d'un objet. C'est le var du C# pour ceux qui connaissent.

```
void Sample_auto()
{
   MesFilles girls = { "Edith", "Lisa", "Audrey" };
   for (auto it = begin(girls); it != end(girls); ++it)
   {
      cout << *it << endl;
   }
}
```

Vous remarquez que j'ai initialisé le vector de string (typedef MesFilles) avec des {}. C'est le C++ 11 qui permet cela.

Les structures

Pour créer un type utilisateur qui contient des variables, le plus simple c'est de créer une structure.

```
struct Book
{
   string name;
   int price;
   string comments;
};

void Sample_Book()
{
   Book CPPPrimer5ThEdition = { "C++ Primer", 50, "the bible" };
   cout << "book :" << CPPPrimer5ThEdition.name << endl;
   cout << "price : " << CPPPrimer5ThEdition.price << endl;
   cout << "comments : " << CPPPrimer5ThEdition.comments << endl;
}
```

Dans cet exemple, la structure est déclarée en dehors de la fonction et la définition d'une variable de ce type se fait avec les {}. On aurait pu aussi initialiser les membres un par un.

```
   Book ref;
   ref.name = "The C++ language";
   ref.price = 50;
   ref.comments = "by Bjarne Stroustrup, the creator of C++";
```

Une structure est comme une classe dont tous les membres sont "public" alors qu'ils sont "private" pour une classe.

Les entêtes .h (ou .hpp)

Un programme peut contenir plusieurs fichiers .cpp. Et il est nécessaire de partager des informations sachant que lorsque le compilateur prend un fichier, il faut que toutes les déclarations de types soient connues. Donc, on écrit des fichiers header (.h) qui contiennent toutes déclarations que nous allons utiliser. Si je veux que la structure Book soit accessible par plusieurs fichiers .cpp, il faut que je la définisse dans un header .h et que je fasse un #include au début du fichier .cpp qui veut l'utiliser pour que le compilateur puisse faire son job. Voici donc la version « propre » du sample qui utilise la structure book :

```
#include "Book.h"

void Sample_Book()
{
   Book CPPPrimer5ThEdition = { "C++ Primer", 50, "the bible" };
   ...
```

Les entêtes précompilées

Par convention, on met les #include au début du fichier .cpp. Pour optimiser le temps de compilation, il existe quelque chose qui se nomme les entêtes précompilées. Il suffit de mettre tous ses fichiers d'entêtes dans un seul header (stdafx.h pour le Visual C++) et de paramétrer le projet pour dire que ces entêtes doivent être précompilées ainsi, le fichier stdafx.cpp contient le #include stdafx.h et c'est tout. Donc, le fichier qui contient toutes les entêtes en parsé une fois et c'est tout.

Le préprocesseur

Hérité du langage C, le préprocesseur permet de changer le contenu du programme. #include ne fait qu'injecter le source dans le programme, et pour inclure plusieurs fois le même fichier dans un source, nous utilisons des barrières. Je m'explique... En réalité on ne met pas tous les headers dans les entêtes précompilées, on met juste les entêtes pour les fichiers .h définies par le système d'exploitation et les librairies tierces. Les structures que nous définissons doivent comporter une structure qui ressemble à cela :

```
// Book.h - this header contains the definition of an entity Book

#ifndef BOOK_HEADER
#define BOOK_HEADER
#include <string>

struct Book
{
   string name;
   int price;
   string comments;
};
#endif
```

Pour un compilateur qui ne supporte pas les entêtes précompilées, le fichier peut être inclus plusieurs fois, il ne sera parsé qu'une seule fois juste avec l'astuce d'un ifndef.

Le namespace std

Il est possible de définir des types dans des espaces de nom. Pour ce, il suffit de déclarer un bloc namespace <nom> {...} et tous les types seront dedans. C'est utile pour séparer les choses. La librairie standard du C++ (STL alias Standard Template Library) définit tous ses types dans le namespace std (standard). Pour inclure la résolution des noms dans un programme, il suffit d'écrire using namespace <nom> ;

Le type string

La STL définie le type string. C'est une classe template qui permet de gérer les chaînes de caractères. Avant de pouvoir utiliser string, il faut ajouter l'instruction #include <string>. string est définie dans le namespace std. Le type string est une classe donc il y a des méthodes pour manipuler les chaînes. On peut même utiliser l'opérateur + pour concaténer des chaînes. On peut utiliser la méthode c_str() pour obtenir le pointeur const sur la chaîne de caractères comme en C.

```
void Sample_String()
{
   string audrey = "je suis une coquine";
   string maggie = audrey + string(" a sa maman");
   if( !maggie.empty() )
   {
      cout << "taille: " << maggie.size() << endl;
   }
   for (auto c : maggie) // pour chaque charactère
   {
      cout << c;
   }
   cout << endl;

   const char * pChar = maggie.c_str();
   printf("%s\n", pChar);
}
```

La classe string est simple à utiliser.

Le type vector

C'est le type le plus connu de la STL avec string. vector<T> permet de stocker des éléments contigus. Si vous cherchez à stocker des éléments en mémoire, il faut utiliser le vector<T>

```
void Sample_Vector()
{
   vector<int> v = { 1, 2, 3, 4, 5, 6 };
   vector<string> girls;
   girls.push_back("Edith");
   girls.push_back("Lisa");
   girls.push_back("Audrey Maggie");

   string lacoquine = girls[2];
   cout << "la coquine c'est " << lacoquine << endl;

   for (auto & f : girls)
   {
      cout << f << endl;
   }
}
```

Les Itérateurs

La STL introduit la notion d'itérateurs qui permet, comme un pointeur, d'avoir un accès indirect à un objet. Dans le cas d'un itérateur, l'objet est un élément dans un container ou dans une chaîne string. Un itérateur permet de récupérer un élément et de passer la position suivante. Comme un pointeur, un itérateur peut être valide ou non. La fonction begin() permet d'obtenir la première position de l'itérateur et end() indique la dernière position invalide. Si le container est vide, begin() et end() auront la même valeur.

```
void Sample_Iterator()
{
   vector<string> girls = { "didou", "mini-mini", "méga teuteute" };
   vector<string>::const_iterator cit = begin(girls);
   cout << "my 12y baby is " << *cit << endl;
   ++cit;
   ++cit;
   cout << "super maggie is " << *cit << endl;
   ++cit;
   if (cit == end(girls))
   {
      cout << "iterator is end..." << endl;
   }

   // it est un vector<string>::iterator
   for (auto it = girls.begin(); it != girls.end(); ++it)
   {
      cout << *it << endl;
   }
}
```

Dans cet exemple, j'utilise un iterateur const et un itérateur normal. Voici les opérations possibles sur un itérateur de container :

Opération	Explication
*iter	Retourne une référence vers l'objet pointé par iter
Iter->mem	Accède à l'objet memoire
++iter	Avance d'une position pour aller sur le prochain objet
–iter	Recule d'une position pour avoir l'objet précédent
Iter1==iter2	Compare deux itérateurs
Iter1 !=iter2	Compare deux itérateurs

Il existe deux types d'itérateurs: const_iterator et iterator. Les operations begin() et end() peuvent s'écrire de deux façons :

```
   auto it1 = begin(girls);
   auto it2 = girls.begin();
   if (it1 == it2)
   {
      cout << "it1 == it2" << endl;
   }
```

Les opérations (on passe vite)

Un bloc en C++ commence par { et se termine par }. Il existe les opérations suivantes: if, switch, while, do while et for. Le C++ 11 introduit le for simplifié alias range-for :

```
void Sample_For()
{
   vector<string> girls = { "didou", "mini-mini", "maggie" };
   for (auto & girl : girls)
   {
```

```
        cout << girl << endl;
    }
}
```

D'autres opérations existent comme break, continue ou goto.

La gestion des exceptions

Les exceptions sont un bloc try..catch avec le type d'exception à catcher. Il est possible de lancer une exception via un throw. La STL définit la classe exception pour les problèmes principaux. Les exceptions sont définies dans std::except.

Les fonctions et le linker (édition de lien)

Pour définir une fonction, il faut un nom, un type de retour et éventuellement des arguments. Comme les programmes peuvent devenir complexes, on crée des fonctions dans des fichiers .cpp et on les utilise dans d'autres fichiers. C'est le principe de la compilation séparée. Pour pouvoir utiliser ce mécanisme, il faut que la fonction soit définie dans un point .h ou .cpp avec juste sa déclaration avec un ; (qu'on appelle le prototype). Cela permet au compilateur de savoir qu'un appel est fait à une fonction et c'est l'éditeur de lien (le link) qui va se charger de faire la résolution des trucs pour construire un exécutable.

Passage d'argument par références

Les habitués du C passent des pointeurs en paramètres alors que les habitués du C++ passent leur paramètre par références.

```
void Reset_Int_Ptr(int * i)
{
    *i = 0;
}

void Reset_Int_Ref(int & i)
{
    i = 0;
}

void Sample_Function()
{
    int a = 10;
    Reset_Int_Ptr(&a); // on passe l'adresse de a
    cout << a << endl;
    int b = 200;
    Reset_Int_Ref(b); // on passe b par reference
    cout << b << endl;
}
```

Gérer la ligne de commande

La ligne de commande du "main" est simple à gérer. Le premier argument donne le nombre d'éléments et le deuxième est un tableau de chaînes contenant les paramètres.

```
int main(int argc, char **argv) { ... }
```

Des fonctions avec des paramètres variables

Introduit avec le C++ 11, les fonctions à paramètres variables tirent parti des initializer_lists. Un type initializer_list est un type de a STL qui représente un tableau. Il est présent dans le header initializer_list.

```
void error_msg(initializer_list<string> il)
{
    for (auto beg = il.begin(); beg != il.end(); ++beg)
        cout << *beg << " ";
    cout << endl;
}

void Sample_Initializer_List()
{
    initializer_list<string> params = { "I am happy", "with a beer", "at 6PM" };
    error_msg(params);
}
```

Arguments de fonctions par défaut

Il est possible de donner une valeur par défaut aux paramètres d'une fonction. Ceci doit être spécifié dans la déclaration de fonction dans un fichier d'entête spécifique.

Fonctions inlines

Héritée du C, une fonction peut être annotée inline. Cela veut dire qu'à chaque appel dans le code source, la fonction sera éclatée en vrac avec son code. C'est pour gagner des pouillèmes en performance mais c'est très souvent utilisé. Ne méprisez pas.

Les macros du préprocesseur

Il existe 4 macros qui sont super utiles en debugging:

__FILE__ : nom du fichier source
__LINE__ : numéro de ligne
__DATE__ : date de compilation
__TIME__ : heure de compilation

Les pointeurs de fonctions

Un pointeur de fonction est un pointeur standard sauf qu'au lieu de pointer sur un objet, il pointe sur une fonction. Plus précisément, il pointe sur une fonction qui possède un type retour et des arguments d'un certain type. Le nom de la fonction n'entre pas en ligne de compte pour le pointeur.

```
bool CompareInteger(const int & a, const int & b)
{
    if (a > b)
        return true;
    else
        return false;
}

void Sample_FunctionPointer()
{
    // déclaration du pointeur de fonction
    bool(*pFn)(const int&, const int&);

    // association ptr
    pFn = &CompareInteger;

    if( (*pFn)(100, 99) == true )
    {
        cout << "superior !" << endl;
    }
}
```

La suite, c'est du lourd...

Dans l'article suivant, nous aborderons les classes qui sont l'essence de C++. Nous verrons les types simples, les constructeurs, les destructeurs, les méthodes virtuelles, l'héritage simple et multiple, les classes amies et peut-être les templates - ce sera peut-être un article à part – donc un beau programme en perspective !

La nouvelle API OneDrive

OneDrive est l'offre Cloud de stockage de fichiers de Microsoft pour le grand public. Ce service offre un espace en ligne gratuit qui peut être étendu via différents types d'offres. Il possède une intégration forte au sein de Windows et propose une application mobile sur les principales plateformes (iOS, Android, Windows et Windows Phone).

Arnaud Auroux et Daniel Djordjevic
Consultants **Infinite Square**

Il peut être intéressant pour un développeur de fournir aux utilisateurs de son application un accès à leur espace de stockage afin de l'intégrer aux fonctionnalités offertes par leur application. Pour cela, une API était déjà accessible via le *Live SDK* permettant un accès en lecture/écriture sur le service. Fin février, une nouvelle version de l'API OneDrive (décorrélée du Live SDK cette fois) est disponible. Nous verrons dans cet article les nouvelles fonctionnalités proposées par cette version. Cette nouvelle API respecte l'architecture REST (communication client / serveur, séparation des responsabilités, pas de persistance du contexte, etc.) et se base sur OAuth 2.0 pour l'authentification. Elle offre principalement :

- Un accès au listing détaillé du contenu OneDrive et à des opérations de modification,
- Un accès à la liste des modifications effectuées sur un élément et ses enfants depuis une date,
- La gestion d'envoi de fichiers volumineux,
- L'envoi de fichiers depuis une URL,
- Un système de recherche.

Authentification

L'accès à API OneDrive est contrôlé via le protocole OAuth 2.0. Ceci a, entre autres, l'avantage de permettre l'accès au stockage d'un utilisateur depuis une application tierce sans aucune communication des identifiants de l'utilisateur à cette application. L'API OneDrive propose 2 modes d'authentification via ce protocole :

- Token flow : ce mode permet d'obtenir rapidement, après consentement de l'utilisateur, un jeton d'accès à l'API limité dans le temps,
- Code flow : ce mode permet, après autorisation de l'application par l'utilisateur, d'obtenir un jeton d'accès à l'API ainsi qu'un jeton pour rafraîchir le jeton d'accès lorsque ce dernier a expiré. Ce mode permet un accès permanent à l'API. Dans le cas général, c'est ce mode qui sera utilisé.

Pour plus d'information sur le protocole OAuth 2.0, nous vous conseillons de jeter un œil à la RFC qui est très bien faite : https://tools.ietf.org/html/rfc6749
Intéressons-nous au mode « Code flow ». Tout d'abord, il est nécessaire d'enregistrer votre application auprès de Microsoft sur le Microsoft Account Developer Center (https://account.live.com/developers/applications). Ce portail Web vous permettra de spécifier les informations propres à votre application (nom, Web ou mobile, etc.) ainsi qu'une url de retour vers laquelle le serveur d'autorisation renverra le code d'autorisation de l'application ou le jeton d'accès à l'API. La configuration terminée, vous aurez accès à un « client ID » et à un « client secret », l'équivalent d'un login/mot de passe qui vous sera utile pour les différents échanges OAuth. 1ère étape dans ce flux d'authentification OAuth : récupérer un code d'autorisation de l'application. Cela se fait via un simple HTTP GET :

```
GET https://login.live.com/oauth20_authorize.srf?client_id={client_id}&scope={scope}
 &response_type=code&redirect_uri={redirect_uri}
```

Cet appel va rediriger l'utilisateur vers une page de demande d'autorisation de son espace de stockage OneDrive par votre application. Notez l'utilisation d'un paramètre « scope » qui va définir les types d'accès de OneDrive requis par l'application. Ceux-ci seront présentés à l'utilisateur sur la page de demande d'autorisations afin que ce dernier ait conscience de ce qu'il autorise. Une fois cette étape terminée, le serveur d'autorisation effectue une redirection HTTP vers l'url de retour (redirect_uri) que vous avez définie lors de la configuration de votre application sur le portail avec en paramètre le code d'autorisation :

```
[REDIRECT_URI]?code=df6aa589-1080-b241-b410-c4dff65dbf7c
```

Ce code va vous être utile pour la récupération du jeton d'accès. Pour cela il vous suffit d'envoyer en POST un formulaire contenant le client ID, le client secret, l'url de retour, le code d'autorisation et un dernier paramètre définissant le type de méthode d'authentification utilisé (OAuth définissant plusieurs types de méthodes d'authentification selon la nature du client) :

```
POST https://login.live.com/oauth20_token.srf
Content-Type: application/x-www-form-urlencoded

client_id={client_id}&redirect_uri={redirect_uri}&client_secret={client_secret}
&code={code}&grant_type=authorization_code
```

En réponse, vous recevez alors vos informations d'authentification au format JSON, dont le jeton d'accès et le jeton de rafraîchissement :

```
{
 "token_type":"bearer",
 "expires_in": 3600,
 "scope":"wl.basic onedrive.readwrite",
 "access_token":"EwCo...AA==",
 "authentication_token":"eyJh...93G4",
 "refresh_token":"eyJh...9323"
}
```

C'est ce jeton d'accès que vous allez utiliser pour chaque appel à l'API, sa durée de validité est limitée (ici définie par le paramètre « expires_in » en secondes, le jeton est ici valide 1 heure). Le jeton de rafraîchissement vous permettra lui, conjointement au client id et client secret, de récupérer un nouveau jeton d'accès lorsque ce dernier aura expiré. Il peut donc être intéressant de faire persister une table de correspondance entre un jeton d'accès et un jeton de rafraîchissement pour facilement récupérer un nouveau jeton d'accès lorsque celui en cours a expiré.

Accès à l'API

Avec ce jeton d'accès, vous pouvez accéder à l'API via l'url racine suivante : https://api.onedrive.com/v1.0/
A partir de cette url, il est possible d'accéder à 2 types de ressources :

- Drive : élément racine de l'espace de stockage (dans le cas général, le drive par défaut est le seul présent),
- Items : enfants d'un Drive.

Ainsi pour accéder à la liste des éléments d'un répertoire, il suffit d'en-

voyer la requête suivante avec le chemin relatif du répertoire (notez l'utilisation du jeton d'accès qui est passé via le header HTTP « Authorization ») :

```
GET /v1.0/drive/root:/repertoire/sous_repertoire:/children HTTP/1.1
Authorization: Bearer EwBgAq1DBA…cgjVOubZfVAE=
Host: api.onedrive.com
X-Target-URI: https://api.onedrive.com
Connection: Keep-Alive
```

Vous recevez alors une réponse au format JSON contenant toutes les métadonnées des éléments enfants du répertoire : /répertoire/sous-répertoire :

```
{
 "@odata.context": "https://api.onedrive.com/v1.0/...",
 "value": [
   {
    "@content.downloadUrl": "https://public-dm2305.files.1drv.com/y2...-Yi-M",
    "createdBy": {
     "user": {
      "displayName…
…
}
```

Ce JSON contient par exemple les données liées à la dernière modification apportée à un élément, son identifiant (utile pour récupérer les informations spécifiques à cet élément via l'API **/drive/items/[ITEM_ID]**), son url, une url d'accès publique à cet élément, etc.

Modification

En plus d'un accès en lecture au contenu de l'espace de stockage et à ses métadonnées, l'API fournit également une liste d'API pour la création, l'envoi, la suppression, le déplacement ou la copie d'un élément. Voyons quelques exemples : Création d'un sous-répertoire (spécifié par l'élément JSON vide *folder*) « Test » dans un répertoire donné :

```
POST /drive/root:/REPERTOIRE_PARENT/
Content-Type: application/json

{
 "name": "Test",
 "folder": { }
}
```

Mise à jour du nom d'un répertoire :

```
PATCH /drive/root:/REPERTOIRE
Content-Type: application/json

{
   "name": "NewFolderName"
}
```

Partage

L'une des fonctionnalités les plus utiles de OneDrive, et de n'importe quel système de stockage en ligne, est le partage. La nouvelle API de OneDrive vous permet de créer simplement un lien de partage, i.e. un lien contenant une clé de sécurité pour l'accès à un élément donné par un autre utilisateur. Pour cela, il suffit d'utiliser l'action *action.createlink* à laquelle vous spécifiez le type d'autorisation fournit par le lien de l'élément (read-only : *view* ou read-write : *edit*): Ex :

```
POST /drive/items/{item-id}/action.createLink
Content-Type: application/json

{
 "type": "view"
}
```

Vous obtenez alors dans la réponse ce fameux lien :

```
…
https://onedrive.live.com/redir?resid=5D33DD65C6932946!70859&authkey=!AL7N1QAf.....
…
```

Détection des modifications

Une application accédant à un espace de stockage OneDrive peut récupérer le contenu sous-jacent afin de permettre un accès en mode déconnecté. Il peut être utile dans ce cas de détecter les modifications apportées à un élément depuis sa dernière récupération afin de mettre à jour l'entrepôt local. Sur chaque ressource, il est ainsi possible de récupérer les changements effectués depuis une certaine date via un jeton de modification. Pour cela, une action est disponible sur chaque ressource : *view.changes.* Ex :

```
GET /drive/root:/CHEMIN_ELEMENT/view.changes
```

Cette requête renvoie un objet au format JSON contenant, entre autres, la liste des modifications apportées aux éléments de la ressource et le jeton de modification utile pour un prochain appel à cette API afin de spécifier le temps référence à partir duquel récupérer les modifications,

```
HTTP/1.1 200 OK
Content-type: application/json
Content-length: length

{
  "value": [
  {
    "id": "0123456789abc",
    "name": "folder2",
    "folder": { }
  },
  {
    "id": "123010204abac",
    "name": "file.txt",
    "file": { }
  }
  ],
  "@odata.nextLink": "https://api.onedrive.com/drive/root/view.changes?token=1230919asd190410jlka",
  "@changes.hasMoreChanges": true,
  "@changes.token": "1230919asd190410jlka"
}
```

Recherche

Dans le même esprit que la détection des modifications, il est possible d'utiliser sur chaque ressource une action *view.search* pour effectuer une recherche à la fois sur le nom des fichiers contenus dans cette ressource mais également dans le contenu de ces fichiers : Ex :

```
GET /drive/root/view.search?q=stockage
```

Le résultat contient les différentes métadonnées des éléments concernés par le modèle de recherche envoyé.

Upload d'un fichier volumineux

Lorsqu'on veut uploader des fichiers volumineux sur OneDrive, il est important de bénéficier d'une méthode de téléchargement qui peut être

interrompue, et reprise à tout instant. Pour avoir un tel comportement, une méthode permet d'envoyer le fichier par fragments pour permettre de relancer plus facilement le téléchargement à partir du dernier fragment bien transmis. Avant de pouvoir envoyer ses fragments, il va falloir créer une session d'upload OneDrive.

Créer la session d'upload

Cette session va avoir pour but de recueillir les données du fichier en attendant qu'il soit complètement transféré sur OneDrive. Pour cela, il suffit d'envoyer une requête POST

```
POST https://api.onedrive.com/v1.0/drive/root:/theBigFile.txt:/upload.createSession
```

Cette requête, si elle réussit, contient la représentation JSON des données nécessaires à l'upload des fragments.

```
{
  "uploadUrl": "https://api.onedrive.com/up/eyJSZXNvdXJjZUlEIjpudWxsLCJSZWxhd...",
  "expirationDateTime": "2015-03-24T22:46:19.616623Z",
  "nextExpectedRanges": ["0-"]
}
```

Uploader les fragments

Une fois la session créée et les données récupérées, on va pouvoir uploader le fichier en question. La limite d'un fragment étant de 60 MB, si le fichier ne dépasse pas cette taille, il est possible de l'envoyer en une seule fois. Cependant, si le fichier excède les 60 MB, le développeur doit fragmenter son fichier lui-même et envoyer les fragments un à un, et surtout dans l'ordre. Plus un fragment est petit, moins il y a de risque d'erreurs pendant qu'on le charge. Pour cela, on va tout simplement pouvoir utiliser l'uploadUrl récupérée au préalable et y envoyer une requête PUT.

```
PUT https://api.onedrive.com/up/eyJSZXNvdXJjZUlEIjpudWxsLCJSZWxhd...
Content-Length: 26
Content-Range: bytes 0-25/48

<bytes 0-25 du fichier>
```

En header de la requête, il faut spécifier, en byte, la plage du fragment envoyée (Content-Range) qui est ici de 0 à 25, et la taille totale du fichier (Content-Length) qui est de 48. Il est primordial de spécifier la même longueur de fichier dans toutes les requêtes PUT à venir, car dans le cas contraire, le téléchargement viendrait à échouer.
Si la requête réussit, on reçoit un HTTP 202 'Accepted' et on peut continuer à uploader ses fragments.

Reprise et annulation

Là où l'API devient vraiment intéressante, c'est avec la reprise d'un téléchargement. En effet, dans le processus d'upload des fragments, l'API va retourner au développeur les éventuelles plages manquantes qui correspondent tout simplement aux fragments manquants.
Cela va être important lorsqu'une erreur HTTP 503 'Service Unavailable' apparaît. Il suffit de GET l'uploadUrl pour récupérer ces plages.

```
GET https://api.onedrive.com/up/eyJSZXNvdXJjZUlEIjpudWxsLCJSZWxhd...

HTTP/1.1 200 OK

{
  "expirationDateTime": "2015-03-24T22:46:19.616623Z",
  "nextExpectedRanges": ["12345-"]
}
```

Au-delà de la reprise, l'annulation d'un transfert est aussi une fonctionnalité dont il est difficile de se passer. On reste, là aussi, dans la facilité avec une simple requête DELETE sur l'uploadUrl.

```
DELETE https://api.onedrive.com/up/eyJSZXNvdXJjZUlEIjpudWxsLCJSZWxhd...
```

Les bonnes pratiques d'utilisation de l'API

Même si elles semblent quelque peu évidentes, des bonnes pratiques sont mises en avant pour l'utilisation de l'API OneDrive :

- Reprendre le téléchargement du fichier sur les erreurs suivantes :
  - 500 'Internal Server Error'
  - 502 'Bad Gateway'
  - 503 'Service Unavailable'
  - 504 'Gateway Timeout'
- Lorsqu'on est confronté à une erreur 404 'Not Found', reprendre le téléchargement du début.
- Ne pas utiliser ce système d'upload pour des fichiers inférieurs à 10 MB mais faire un upload « direct ».
- La taille optimale d'un fragment varie en fonction de la connexion. Environ 4 MB pour des connexions lentes ou moyennes et environ 10 MB pour des connexions rapides.
- Utiliser des tailles de fragment qui sont des multiples de 320 KB.

Uploader depuis une URL

Tout est dans le titre, depuis une application, il est possible d'envoyer un fichier sur OneDrive en lui indiquant qu'il se trouve sur un autre serveur à une URL donnée. OneDrive va automatiquement télécharger le fichier depuis un serveur distant et éviter à l'application de devoir le faire. Cela va être un atout non négligeable lorsqu'on développe une application mobile, ou lorsqu'on traite des fichiers volumineux et que l'on désire externaliser leur transfert. Pour cela, nous allons construire une requête POST et fournir en body de la requête la représentation JSON du fichier à créer.

```
{
  "@content.sourceUrl": "http://www.infinitesquare.com/Content/images/theme/logo.png",
  "name": "logo.png",
  "file": {}
}
```

Il est important que cette url soit publique, étant donné que le serveur distant ne va pas utiliser l'authentification du compte OneDrive pour télécharger le fichier. Le service va immédiatement retourner, si la requête passe, un HTTP 202 'Accepted' avec une url permettant de monitorer le statut du transfert. Pour cela il suffira de GET l'url tout juste récupérée. Si le retour est un HTTP 202, le corps de la réponse est alors un AsyncJobStatus qui contient des données telles que le pourcentage effectué du transfert et le statut de l'opération. Si le transfert est terminé, un HTTP 303 'See Other' est retourné avec une nouvelle url menant au fichier transféré.

Conclusion

La nouvelle API de OneDrive permet un accès riche, simple et efficace à l'espace de stockage OneDrive d'un utilisateur par vos applications. Elle est de plus très bien documentée, alors pourquoi s'en priver ? Rendez-vous sur http://onedrive.github.io/ ;)

Genymobile : le génie français d'Android

Si vous êtes un habitué de la DroidCon Paris, ou que vous lisez tous les mois Programmez, Genymobile est synonyme d'Android et d'outils dédiés dont un émulateur Android qui a fait beaucoup parler de lui. Nous avons rencontré dans les locaux parisiens Arnaud Dupuis, co-fondateur.

« Quand on crée, on cherche à trouver le « la ». Beaucoup de Web agencies sont allées vers le mobile, avec l'explosion du smartphone » introduit Arnaud. Dès 2010, l'aventure a commencé. *« Au départ, nous voyions beaucoup de développements autour de l'affichage de flux, c'était l'époque du « le web c'est mort, si tu n'as pas ton appli tu n'existe pas. »* poursuit Arnaud. Mais les entreprises et le DSI étaient peu ciblés. *« Nous voulions les aider à passer ce cap ».* Un déclic arriva avec l'explosion de l'iPhone, dès alors, on a demandé aux DSI d'ouvrir les vannes des serveurs. *« Tout le monde avait ou voulait un smartphone. C'était en quelque sorte un Far West. C'est là que nous nous sommes dits, il faut que nous proposions des solutions simples et efficaces. Dès lors, nous nous sommes peu à peu orientés vers l'édition logicielle et non plus le développement sur mesure. »* précise Arnaud.

Un passage initiatique pas simple mais excitant

Mais la mutation de Genymobile ne fut pas simple. Le changement de stratégie avait un écueil non négligeable : pour être éditeur, il faut des équipes, des budgets, du temps. Un point délicat est la levée de fonds pour financer la structure et les développements qui peuvent prendre plusieurs années. « Pour assurer le quotidien et avoir du cash, nous étions une SSII de développement avec une forte expertise Android » poursuit Arnaud. La jeune société a toujours misé sur l'expertise et le très haut niveau de compétences des développeurs. Nous sommes alors en 2011 et 2012.

Genymotion arrive !

Fin 2012, un des grands moments des équipes fut la présentation officielle de Genymotion, une émulation Android pour les développeurs et les entreprises, qui offrait des performances et des fonctions jusqu'alors inconnues dans ce genre d'outils Android... Il fallut attendre quelques mois pour voir apparaître la première version commerciale de l'émulateur, créé dans les locaux parisiens. L'accueil a été plus que favorable. Mais les équipes ne se relâchent pas et la société regarde de près Android et elle se dit : il y a quelque chose à faire sur la partie déploiement et industrialisation des déploiements des environnements Android et des applications. Coup sur coup, 2 gros développements sont lancés : Genydeploy et Genymaster. Il existe une vraie demande sur ces types d'outils notamment quand les entreprises doivent déployer et configurer des centaines de terminaux. Mais ce n'est pas pour autant que Genymobile va fondamentalement changer. « nous serons toujours une SSII même si dans le chiffre d'affaire, l'édition logicielle fait aujourd'hui le gros de l'activité. » recadre Arnaud. L'édition logicielle permet ainsi de soutenir l'internationalisation de Genymobile qui décroche régulièrement de gros contrats aux Etats-Unis où Angy Zettor (cofondatrice de la société) et Arnaud passent de plus en plus de temps et y vivent même.

Genymobile = boîte à développeurs

Aujourd'hui, Genymobile c'est 48-49 personnes dont 42 développeurs ! Si ce n'est pas une boîte de développeurs, qu'est-ce que c'est ? La société fait environ 4 millions €. « Pour 2015, nous atteindrons 7 à 10 millions € ! » espère Arnaud. Trois bureaux sont ouverts : Paris, Lyon et San Francisco. « *Pour soutenir notre croissance et nos projets, nous allons fortement recruter, 20 à 30 personnes: des développeurs ! Nous avons une grosse roadmap étalée sur 18 mois...* » indique Arnaud.

La passion du code

Arnaud est un passionné, un geek dans l'âme. Pour lui, le dév de Genymobile est avant tout un vrai passionné qui apporte quelque chose « de + » à la société. « Le diplôme n'est pas notre critère pour choisir telle ou telle personne. Il doit aimer aller plus loin, apprendre, être capable de chercher des solutions, d'en proposer. » tient à préciser Arnaud. La technologie reste au cœur de la société et la R&D est fondamentale mais délibérément, les équipes bossent sur un nombre restreint de technos : Qt, C++, Android. L'âge moyen des dévs est de 30 ans, avec quelques stagiaires (triés et sélectionnés). Trois développeuses ont intégré les équipes.
Et les salaires ? « Ils sont au-dessus de la moyenne mais il n'y a pas que l'argent qui motive. Si l'année est bonne, on augmente tout le monde. Pour nous, le développeur n'est pas un sous-salarié, c'est notre ressource ! » s'exclame Arnaud. Et si vous tentiez l'aventure Genymobile ?

L'art de cultiver une culture d'entreprise et le bien-être

Pour Angy et Arnaud, Genymobile est une culture d'entreprise bâtie depuis les débuts. Angy est le lead de la culture d'entreprise. Les trois bureaux sont connectés pour pouvoir organiser rapidement des réunions et des points techniques. Il s'agit aussi de faire participer tout le monde à la vie de la société et aux grands choix stratégiques des projets. Il y a aussi un comité open source. Car une des essences de Genymobile est l'open source. « On passe beaucoup de temps dans les locaux, avec les équipes. Il ne faut pas que l'on s'y emmerde. On y joue, on y mange, on y dort parfois. Il faut que tout le monde s'y sente bien. Tout le monde est très geek ici mais nous n'imposons rien tant que le travail est réalisé et bien fait. Parfois, nous avons des horaires décalés, certains sont plus du matin, d'autres du soir. Nous avons des règles de bonnes conduite. » explique Arnaud. « il faut vivre en bonne intelligence. La culture d'entreprise est très forte à Genymobile. Nous avons un séminaire tous ensemble d'une semaine, chaque année. Chaque nouvel arrivant reçoit un welcome Pack, avec plein de surprises ». La culture d'entreprise est un élément crucial dans la vie quotidienne de Genymobile. Angy et Arnaud se définissent comme les hémisphères d'un même cerveau dans un management collaboratif : cet esprit est insufflé auprès de tous les salariés. Ce modèle de fonctionnement est encore peu répandu. Bref, il y a de la zenitude ! Et si les étages sont studieux, le rez de chaussée est le lieu d'échange, de relaxation.

Plankalkül : le premier langage de programmation moderne !

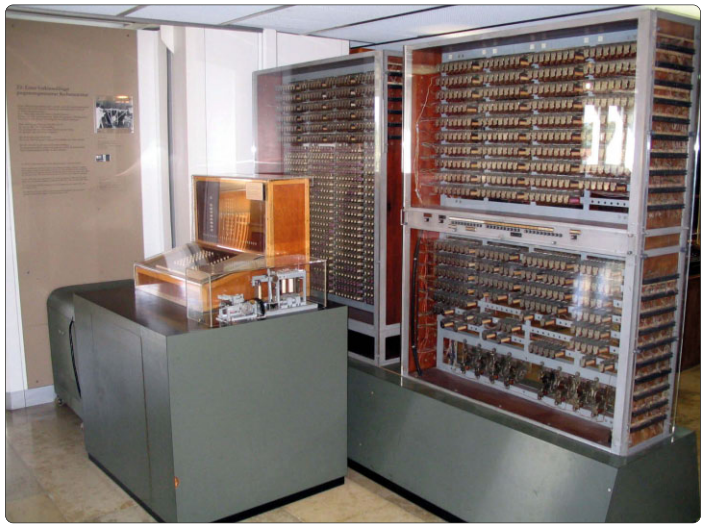
Timeline : 1943
Objet : programmation

François Tonic

Où et quand est né le premier véritable langage de programmation au sens moderne du terme ? Méconnu du public et même de beaucoup de développeurs, l'inventeur de cette merveille est l'Allemand Conrad Zuse dès le début des années 1940, en pleine Seconde Guerre Mondiale.

Les ordinateurs Z

Zuse va conceptualiser le premier langage complet de haut niveau : PlanKalkül. Zuse est un inventeur de génie. Dès les années 1930, il imagine et prototype son premier ordinateur : le Z1. Il conçoit ainsi les premiers ordinateurs à relais. Le Z1 est massif : 1 tonne, mais il s'agit d'une machine programmable. Il comporte plusieurs unités : contrôle, arithmétique (contenant 2 registres), Entrée/Sortie, mémoire, sélecteur de mémoire. L'ensemble fonctionne à une vitesse d'un cycle par seconde (1 Hz), le fameux Hertz. La mémoire est un élément important des ordinateurs de Zuse. Il consiste en une mémoire de 64 mots. Des instructions Entrée/Sortie permettaient de manipuler les fonctions du Z1 et de le programmer, par exemple, l'instruction Pr Z pour lire le contenu des deux registres de la mémoire...
Le Z3 (construit entre 1939 et 1941) sera le premier ordinateur capable d'exécuter des programmes et de « travailler », même s'il s'agit d'un ordinateur électromécanique. Le Z3 est une avancée considérable : horloge de 5,33 Hz, nouvelles mémoires, nouveaux relais, une unité de calculs flottants, clavier spécifique. Il supporte la racine carrée dans l'unité de calcul arithmétique. Il faut environ 3 secondes pour une multiplication. Cependant, le Z3 ne peut pas stocker des programmes en mémoire, mais il contenait déjà les modules nécessaires des futurs ordinateurs...
Zuse travaille à la génération suivante, le Z4 qui sera opérationnel en 1950.



Plankalkül : un langage en avance sur son temps !

Zuse travaille parallèlement aux ordinateurs Z sur la notion de programmation et donc du langage de programmation. Il comprend très vite qu'il faut un langage informatique pour faire travailler l'ordinateur. Il va alors imaginer et théoriser le langage Plankalkül.
Zuse imagine déjà de nombreuses instructions et des principes de programmation qui ne vont réellement apparaître que dans les années 1950 et 60 : les types de données, les variables locales, les constantes, les résultats, les entiers, les sous-programmes, les boucles, les opérateurs logiques, les listes... Les programmes sont réutilisables.
Plankalkül n'a jamais pu être implémenté et Zuse publia le concept complet en 1948 (même si les bases du langage datent de 1943).

John Von Neumann : le Conrad Zuse américain

Von Neumann, devenu Américain dans les années 1930, travaille avec l'armée américaine et participera au projet Manhattan. Il travaille activement au projet ENIAC, une machine à calculer électronique, lancé en 1943, mais ne jouera pas de rôle important dans la guerre.

En 1945, Von Neumann imagine une nouvelle architecture pour créer un ordinateur électronique avec une mémoire capable de stocker les instructions et les données, un stockage externe : l'architecture de Von Neumann (même si un débat existe sur ce terme et la paternité réelle du concept). Zuse avait lui aussi imaginé un stockage des programmes sur une mémoire dès 1936, mais sans pouvoir aller au-delà.

Abonnement : Service Abonnements PROGRAMMEZ, 4 Rue de Mouchy, 60438 Noailles Cedex. - Tél. : 01 55 56 70 55 - *abonnements.programmez @groupe-gli.com* - Fax : 01 55 56 70 91 - du lundi au jeudi de 9h30 à 12h30 et de 13h30 à 17h00, le vendredi de 9h00 à 12h00 et de 14h00 à 16h30. **Tarifs** abonnement (magazine seul) : 1 an - 11 numéros France métropolitaine : 49 € - Etudiant : 39 € CEE et Suisse : 55,82 € - Algérie, Maroc, Tunisie : 59,89 € Canada : 68,36 € - Tom : 83,65 € - Dom : 66,82 € Autres pays : nous consulter.
PDF : 30 € (Monde Entier) souscription sur www.programmez.com

Une publication **Nefer-IT**
7 avenue Roger Chambonnet
91220 Brétigny sur Orge
redaction@programmez.com
Tél. : 01 60 85 39 96
Directeur de la publication & rédacteur en chef : François Tonic

Secrétaire de rédaction : Olivier Pavie

Ont collaboré à ce numéro : F. Mocq, S. Saurel.

Experts : R. Degret, B. Doolaeghe, K. Aldebert, A. Guelfi, R. Jeannin, T. Larbi, N. Loye, G. Damien, K. Situé, A. Paponaud, A. Neveu, X. Cauchy, S. Warin, L. Dupuis, Q. Guay, J. Dubuc, J-F Garreau, G. Finance, C. Pichaud, A. Auroux, D. Djordjevic

Photos/illustrations : CommitStrip

Maquette : Pierre Sandré

Publicité : PC Presse,
Tél.: 01 74 70 16 30, Fax : 01 41 38 29 75
pub@programmez.com

Imprimeur : S.A. Corelio Nevada Printing, 30 allée de la recherche, 1070 Bruxelles, Belgique.

Marketing et promotion des ventes :
Agence BOCONSEIL - Analyse Media Etude

Directeur : Otto BORSCHA oborscha@boconseilame.fr
Responsable titre : Terry MATTARD
Téléphone : 09 67 32 09 34

Contacts
Rédacteur en chef : ftonic@programmez.com
Rédaction : redaction@programmez.com
Webmaster : webmaster@programmez.com
Publicité : pub@programmez.com
Evenements / agenda : redaction@programmez.com

Dépôt légal : à parution - Commission paritaire : 1215 K 78366 - ISSN : 1627-0908

Sur abonnement ou en kiosque

Le magazine des pros de l'IT

Mais aussi sur le web

Ou encore sur votre tablette

L'INFORMATICIEN

ikoula
HÉBERGEUR CLOUD

LE CLOUD GAULOIS, UNE RÉALITÉ ! VENEZ TESTER SA PUISSANCE

EXPRESS HOSTING

Cloud Public
Serveur Virtuel
Serveur Dédié
Nom de domaine
Hébergement Web

sales@ikoula.com
01 84 01 02 66
express.ikoula.com

ENTERPRISE SERVICES

Cloud Privé
Infogérance
PRA/PCA
Haute disponibilité
Datacenter

sales-ies@ikoula.com
01 78 76 35 58
ies.ikoula.com

EX10

Cloud Hybride
Exchange
Lync
Sharepoint
Plateforme Collaborative

sales@ex10.biz
01 84 01 02 53
www.ex10.biz