programmez!

#192 - Janvier 2016

le magazine du développeur

Drupal™

DOSSIER SPÉCIAL 14 pages

© ISTOCK - DOLGACHOV

Les réalités alternatives

Oculus, Hololens, Google Glass...

*Mon 1er projet **Oculus** et **Unity 5.1***

Les usages, les matériels.

SPECIAL VINTAGE !

Amiga, Amstrad CPC,
Apple II, Atari ST, GameBoy

Redécouvrez ces machines mythiques et comment les programmer

Créer son drone de A à Z

1ere partie

Connecter des objets avec **Node.JS**

Les profils en **Java 8**

M 04319 - 192 - 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

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

programmez!
le magazine du développeur

Tu vas aimer l'informatique vintage.

Cela faisait bien longtemps que l'idée était dans ma tête : faire un dossier vintage où vous verriez enfin ce qu'avaient dans le ventre les ordinateurs des années 80 et 90. Les Atari ST, Amiga, Amstrad CPC, et même les bons vieux Apple II, on en faisait des choses et des prouesses avec quelques octets. Découvrez aussi que ces machines sont toujours recherchées et utilisées, notamment par la démoscène, très active en France ! Et pour ne pas gâcher notre plaisir, un petit émulateur GameBoy en C#. C'est cadeau !

La grande nouvelle de ces dernières semaines est bien entendu la disponibilité de Drupal 8. À force d'en parler dans Programmez !, la plateforme est enfin sortie... Nous vous proposons d'en explorer quelques recoins et nous faisons le point sur la dernière conférence Drupal européenne, la DrupalCon de Barcelone.

Nous nous projetons aussi vers le futur immédiat avec les réalités alternatives. Nous parlerons Oculus Rift, Hololens, Google Glass, usages, modèles de développement. Un dossier passionnant et tout sauf virtuel.

Ce premier numéro de l'année vous propose aussi de nombreux autres articles : Kinect, Java 8, Node.JS, architecture Java, le framework Mean ou encore comment construire son propre drone.

Toute la rédaction de Programmez ! vous souhaite une excellente année 2016.

P.S. : nous remercions tous les auteurs et contributeurs, tous nos partenaires techniques et, vous, fidèles lectrices et lecteurs du magazine.

Le boss
ftonic@programmez.com

À lire dans le prochain numéro n°193 en kiosque le 30 janvier 2016

Les tests et vous
Les tests font partie du développement et du quotidien du développeur. Quels tests faut-il faire ? Quels outils choisir ? Les bonnes pratiques ? Comment intégrer Jenkins et Arduino ?

IDE
Non, Delphi n'est pas mort.

NoSQL
Pourquoi utiliser le modèle orienté document de MongoDB ?

janvier

LavaJUG organise une réunion AngularJS /Angular 2 le 14 janvier.
http://www.lavajug.org/events

JUG Nantes :

soirée continuous delivery avec Jenkins et Docker. Le 19 janvier à partir de 19h. http://www.nantesjug.org

SnowCamp2016 !

Retrouvez +250 développeurs sur de nombreuses thématiques les 21 et 22 janvier à Grenoble. Journée ski le 23... site : http://snowcamp.io/2016/fr/

BeMyApp

organise plusieurs hackathons et concours :

- Vinci Startup Tour : Vinci aide les startups à diffuser leurs productions dans le groupe. Les inscriptions doivent se faire avant le 17 janvier minuit. Les startups présenteront leur projet devant un jury (courant février). Les résultats seront annoncés le 4 mars.
- Challenge Full-stack JS : 10 heures de code, des challenges ! Venez défier les meilleurs experts en JS... à l'école 42. 23 janvier.
- Hackathon e-résidents Dalkia : inventer l'habitation de demain et une nouvelle manière d'habiter et d'interagir. 48 h pour monter le projet et le défendre du 5 au 7 février (Paris). http://hackathon.dalkia.fr

février

DevFest Paris, le 5 février !

Très gros succès à Nantes, la DevFest arrive à Paris. Cette grande journée de développement et de technologies est organisée par le Google Developper Group de Paris. La journée se déroulera de 8h à 20h et l'agenda s'annonce chargé avec 25 conférences et +700 personnes attendues ! Lieu : La Grande Crypte, Paris.
Site officiel : http://devfest.gdgparis.com

mars

NIDays 2016 : 10 mars

National Instruments organise sa journée annuelle à Paris le 10 mars prochain. Occasion pour faire le point sur les différents outils, l'instrumentation, l'informatique industrielle et embarquée. La journée sera rythmée par de nombreuses conférences.
Grand rendez-vous de la journée, deux coupes seront organisées, la Coupe NXT et la Coupe RIO, dont le thème commun est la robotique.
Pour plus de détails : http://france.ni.com/nidays/coupes-robotiques
Site officiel : http://france.ni.com/nidays

AlfrescoDay 2016

L'éditeur Alfresco organise une journée IT/technique le 30 mars 2016 à Paris

MUG Strasbourg : les prochaines réunions

Le Microsoft User Group de Strasbourg organise régulièrement pour la communauté des conférences autour des technologies Microsoft.
Les prochains événements prévus sont :
— Janvier : conférence sur Xamarin
— Février : conférence sur le langage F#
— Mars : conférence sur TypeScript et Angular 2
A l'heure où nous publions, les dates précises ne sont pas encore connues.
— Facebook : https://www.facebook.com/groups/MugStrasbourg/?fref=ts
— Twitter : https://twitter.com/MUGStrasbourg

Quelques dates à retenir

EVOLVE16 :

La conférence développeur de Xamarin se déroulera en Floride du 24 au 28 avril. De nombreux thèmes seront abordés : design, compilation, intégration, sécurité, tests, monitoring... https://evolve.xamarin.com

La conférence **NCRAFTS 2016** aura lieu cette année les 12 et 13 mai

MICROSOFT TECHDAYS 2016 :

Les fameux TechDays se dérouleront cette année en octobre et sur deux jours

Les événements Zenika

Zenika NG2 Tour/Zenika organise le NG2 Tour au sein de ses différentes agences et vous propose de découvrir tous les secrets d'Angular 2. Vous souhaitez commencer une nouvelle application et vous hésitez entre AngularJS et Angular 2 ? Vous désirez avoir un aperçu de la prochaine version du framework ? Ou peut-être faites-vous partie des personnes qui ont des interrogations par rapport à ce qu'elles ont déjà entendu sur l'avenir d'Angular ? Les dates :
Zenika Paris : 12 janvier 2016
Zenika Lyon : 21 janvier 2016
Zenika Lille : 26 janvier 2016
Programme et inscription :
http://www.zenika.com/Evenements/zenika-ng2-tour.html

WEBDEV®

NOUVELLE VERSION 21

CRÉEZ FACILEMENT DES SITES «RESPONSIVE WEB DESIGN»

Fournisseur Officiel de la Préparation Olympique

Rendez vos sites «Mobile Friendly».

WEBDEV 21 vous permet de rendre facilement vos sites «Mobile Friendly».
Les sites que vous créez sont ainsi mieux référencés par Google.
Responsive Web Design et Dynamic Serving sont à votre service dans WEBDEV 21.

DÉVELOPPEZ 10 FOIS PLUS VITE

www.pcsoft.fr

Des centaines de témoignages sur le site

Ecran tactile officiel Raspberry Pi

Voilà il est enfin arrivé ! Raspberry Pi propose son propre écran tactile 7 pouces. Pas toujours facile à trouver, il est vendu généralement aux alentours de 80-90 €. Petite présentation.

François Tonic
Programmez!

Les caractéristiques de l'écran sont plutôt classiques :

- Résolution écran 800x480
- Ecran tactile capacitif
- Compatible Pi 1 et Pi 2 (attention aux trous de fixation qui ne sont pas identiques entre les versions)
- Taille écran physique : 194 x 110 x 20 mm

La construction de l'écran est très propre et de qualité, c'est rassurant. Le pack comprend l'écran en lui-même, le shield dédié, les nappes, les fils de connexion et les vis de fixations. Bien entendu, la Pi n'est pas incluse... Et aucune documentation n'est incluse mais elle est facilement trouvable sur le site element14. L'écran nécessite un shield dédié qui est un adaptateur faisant le lien entre l'écran et la Pi.

Le montage

En lui-même, le montage n'est pas très compliqué et se réalise en quelques minutes :

- Connectez les deux nappes (couleur jaune) de l'écran à la shield. Vous ne pouvez pas vous tromper de sens. Faites bien attention à bien pousser les nappes dans les connecteurs et de bien « verrouiller » l'embout de maintien.
- Prenez la nappe blanche (fournie) et insérez un côté dans le connecteur situé à gauche de la board (côté inscrit en bleu vers vous).
- Connectez la nappe blanche sur la Pi, sur le connecteur Display.
- Prenez les fils rouge et noir. Fixez-les sur les broches 5V (fil rouge) et GND (fil noir) du shield. Puis sur la Pi, connectez les fils sur les broches 5V et GND (broches du haut en partant à gauche : 1ere broche et 3e broche).
- Fixez maintenant la board à l'écran avec les longues vis puis placez la Pi (ici une Pi 2) puis vissez la board.

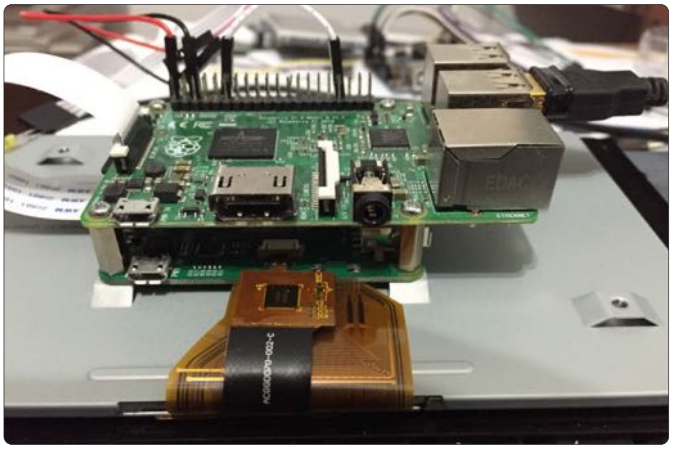
Voilà, le montage est terminé. Si ce n'est pas déjà fait, vous devez installer la distribution Raspbian sur une carte SD (minimum 4 Go, nous conseillons 8 Go) puis l'insérer dans le lecteur SD de la Pi 2. Prenez la distribution la plus récente.

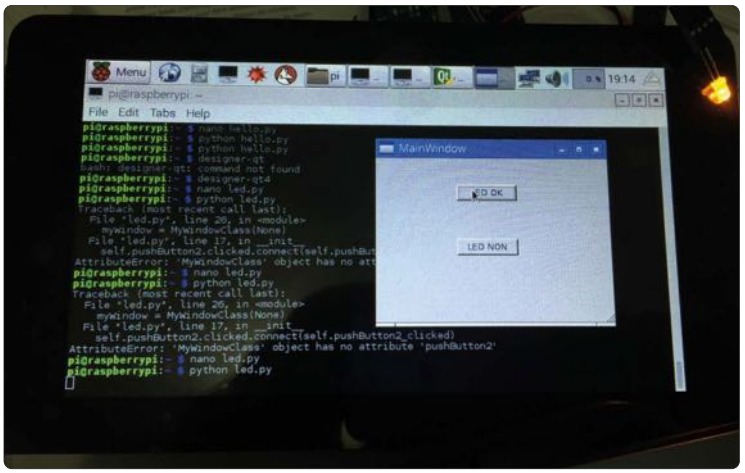
Premier contact

Si le montage est correct, connectez l'alimentation sur le connecteur du shield (et non de la Pi 2). En quelques secondes, l'écran s'allume et le système se charge.

L'écran est très lumineux avec un bon angle de vision. Le tactile est plutôt sensible et efficace même si les gros doigts vont vite avoir quelques problèmes. Vous pouvez alors connecter à votre Pi un clavier et une souris. Car coder avec les doigts ce n'est pas forcément super pratique.

Vous transformerez votre Pi 2 en véritable PC hybride mais pour un prix très raisonnable : — 150 € ! Surtout, vous profitez de tous les avantages du Pi avec l'écran tactile. Vous pouvez ainsi utiliser vos shields et capteurs et continuer à utiliser vos outils de développement... Nous avons essayé de faire fonctionner avec Windows 10 IoT mais nos premiers tests ont échoué (nous n'avions pas les dernières builds IoT).





Codez directement sur l'écran...

Soyons clairs, pour de petits projets, créer des objets directement sur la Pi 2, la taille de l'écran suffira mais pour des projets un peu exigeants, mieux vaut passer par un écran « normal ». Pour notre part, nous avons installé PyQt4 et Qt Designer 4. Qt est une puissante librairie multi-plate-forme pour créer des interfaces et Designer est l'outil graphique. Il suffit de créer l'interface dans le Designer puis de l'implémenter dans le code Python via PyQt qui fait le lien entre Python et Qt (qui est avant tout en C++). Vous importez les fichiers .ui et vous référencez les différents objets d'interface avec les bons events. Le tout fonctionne plutôt bien et vous pouvez prototyper rapidement votre objet IoT et les capteurs. Nous vous conseillons de brancher un clavier et une souris.

Nous le recommandons !

Franchement, malgré un prix un peu élevé, cet écran est réellement intéressant et ouvrira de nouvelles perspectives à votre Pi/Pi 2 et à vos objets connectés. Vous pourrez facilement développer des applications avec interfaces adaptées pour piloter votre domotique et capteurs.

Malgré l'absence de toute documentation dans le carton, celle proposée par Element14 suffit pour être opérationnel en 5-10 minutes, et les tutoriaux devraient rapidement se multiplier; cet écran prend tout son sens pour des projets IoT et les capteurs.

Enjoy !

LES +

- Qualité de l'écran
- Les possibilités du tactile avec la Pi 2
- Un véritable hybride fonctionnel et pas cher
- Fixation rapide et derrière l'écran
- Raspbian compatible (mais pas adaptée)

LES –

- Pas de documentation
- Prix un peu élevé
- Pas toujours facile à trouver
- La colle entre le châssis et l'écran se détache trop facilement

FAÇONNONS ENSEMBLE L'INTERNET DES OBJETS

NIDays

Rejoignez plus de 1 200 innovateurs issus de secteurs industriels variés, venez échanger avec les équipes NI et découvrez comment les avancées technologiques convergent pour créer un monde plus intelligent et plus connecté, reposant sur des systèmes conçus par logiciel.

Paris, Palais des Congrès
Jeudi 10 mars 2016

Inscription gratuite sur **ni.com/nidays.**
Suivez-nous sur Twitter : **@NIFrance** et en live avec **#NIDays.**

NATIONAL INSTRUMENTS

Un développeur au cœur du Vietnam

En 2013-2014, j'ai eu l'opportunité de travailler au Vietnam, à Ho Chi Minh Ville dans le sud du pays sur une période d'un an. J'y ai travaillé 7 mois pour un client français basé en France, Fullsix (http://www.fullsix.fr/f6-havas/fr/), puis 4 mois pour le compte d'une entreprise japonaise, Intelligence Business Solutions (http://www.inte.co.jp/en/).

Fig.A

Guillaume Delattre
SFEIR

Je vais me concentrer sur la première expérience, la seconde étant trop éloignée de mon coeur de compétences. En effet, la culture Japonaise du travail est bien trop différente de la culture française et même les outils informatiques utilisés sont parfois 100% nippons, par exemple le framework DI (Dependency Injection) Seasar Seasar qui a une documentation riche et un support uniquement en Japonais. Avant de commencer à décrire la mission pour laquelle j'ai été amené à travailler à « Saïgon » sur un plan technique, il est nécessaire de décrire le contexte du pays. Le Vietnam fait partie des pays dits émergents et possède un taux de croissance économique soutenu. Le Vietnam a longtemps fermé ses portes au niveau économique pour des raisons historiques et politiques, mais est maintenant largement ouvert. Pour preuve, l'ouverture symbolique d'un 1er McDonald's au Vietnam en 2014, (http://edition.cnn.com/2014/02/10/travel/mcdonalds-opens-in-vietnam/).

Aujourd'hui, de nombreux investisseurs européens (France, Allemagne, etc.), américains, asiatiques (Japon, Corée du Sud, etc.) sont présents au Vietnam. De nombreuses écoles dans le domaine des nouvelles technologies sont implantées et forment chaque année quantité d'ingénieurs, et, parfois les Vietnamiens viennent jusqu'en Europe ou aux États-Unis pour étudier. Et j'ai pu travailler avec quelques Vietnamiens ayant étudié en France.

De plus, Ho Chi Minh, étant située au milieu de la région du delta du Mékong, région productrice d'une quantité importante de denrées alimentaires, possède un port de transit commercial extrêmement important à dimension internationale et donc une activité économique locale très intense. Voilà pour le contexte économique.

En revenant à ma première expérience professionnelle en tant que développeur JAVA et amené à travailler avec un autre collaborateur français sur place, j'ai intégré une équipe dédiée à travailler pour le groupe français Ekino-FULLSIX. L'environnement technique était relativement riche :

- Des postes d'ordinateurs DELL avec une version d'Ubuntu comme OS et donc console d'administration ;
- Une liaison VPN nous permettant de nous connecter à des serveurs distants ;
- Des salles réservées aux vidéoconférences ;
- Environnement Atlassian : JIRA (soft pour le suivi de projet), confluence (solution pour la documentation technique) ;
- Usine de dev au travers d'Artifactory, un soft pour gérer les différents dépôts MAVEN avec un souci d'automatiser les processus au maximum ;
- Un ensemble de scripts de release et de déploiement des applications sur les serveurs ;
- Outil de gestion de versions : GIT avec un niveau de code review via GRIT. Le code pushé est systématiquement revu par une tierce personne validant ou non le commit et qui merge lui-même les différentes branches pushées par les développeurs ;
- JAVA 7, pile de frameworks JAVA modernes et standards (Spring, Hibernate, Guava, WEB-services de type REST, etc.) ;
- Souci d'écriture de tests unitaires (isolés et ne dépendant donc pas de données en base) et tests d'intégration (susceptibles d'être dépendants de la base de données et donc nécessitant régulièrement une maintenance)

Dans un premier temps, j'ai pu aider au recrutement en formant/testant quelques développeurs vietnamiens sur place. Le premier projet auquel j'ai participé au départ a été l'intervention sur le projet en interne de Fullsix, le projet « Webfactory » dont voici l'interface de connections, pour ceux qui auraient travaillé (ou travaillent) pour Fullsix, login – webfactory (la nostalgie du remplissage des CRA mensuels).

Il s'agit de leur outil de gestion en interne Fig.A.

La prise en main de l'outil SCM GIT a été pour moi une première étant donné que je n'avais utilisé dans le passé que SVN. Cette période d'apprentissage a été indispensable pour moi au départ pour pouvoir travailler et être opérationnel rapidement.

J'ai également travaillé sur le projet d'une solution commercialisée par Fullsix, un CMS fait maison extrêmement complet et proposant une solution « clé en main » à l'utilisateur final pour être capable de gérer et publier le contenu, et donc le marketing, de façon entièrement autonome. Quelques clients comme SFR ou Renault poussent à croire que la solution fonctionne très bien et correspond à une réelle demande sur le marché actuel.

Côté technique, j'ai pu apprivoiser un nouveau framework orienté VIEW, le framework Apache Wicket, mais qui m'a un peu dérouté à mes débuts car bien différent de ceux que j'avais pu utiliser auparavant.

Outre l'activité du développement logiciel, la mission de recruter et de former du personnel qualifié sur place a été très vite d'actualité.

De nombreux entretiens techniques ont donc été nécessaires et j'ai pu croiser quelques candidats vietnamiens pour les auditionner. Les différences entre les différents candidats étaient extrêmes. A commencer par l'expression orale en anglais obligatoire. En effet, on remarque que les personnes ayant séjourné/étudié en Europe-Etats-Unis parlent nettement mieux et sortent clairement du lot. Ensuite, la barrière de la langue passée, viennent les tests techniques, et, là aussi, on s'aperçoit qu'il y a de grosses différences. Globalement, une personne intéressée et motivée pour se tenir informée de manière personnelle dans son domaine de compétences est resté le meilleur gage/preuve d'efficacité.

Je souhaite également parler de l'ambiance/environnement de travail à Ho Chi Minh Ville dans les locaux de la société où j'ai travaillé. Pour commencer, les horaires de travail sont relativement identiques à ceux pratiqués en France. Je parle bien d'une entreprise française implantée au Vietnam et non d'une entreprise locale vietnamienne employant du personnel vietnamien. Un petit détail, mais important à mes yeux. Il s'agît de la sieste pratiquée par l'ensemble du personnel et je précise tout le personnel, y compris la direction.

Ensuite, j'ai travaillé dans un bâtiment accueillant 5-6 open-spaces à taille humaine divisés en gros par client cible. Ensuite, pour le reste et les heures de travail effectif, cela ressemble de très près à ce que l'on peut vivre en France.

Pour terminer sur mon expérience avec l'entreprise japonaise, je veux préciser que cela a été un souhait personnel que de travailler pour un groupe nippon. Par curiosité, je pense. Mais 3-4 mois m'ont suffi, car la culture est tellement différente qu'il est très difficile de réellement s'intégrer et de comprendre la mentalité et la méthodologie utilisées. Selon leurs dires, ils souhaitaient un personnel diversifié incluant quelques employés occidentaux et ils appréciaient énormément la France en général.

En conclusion, j'aurais tendance à affirmer qu'une expérience à l'étranger, notamment, dans un pays à forte croissance économique représente un atout et permet également de voir la France de l'étranger, et de s'apercevoir de sa présence à l'international. Et dans le domaine des nouvelles technologies, il est plutôt possible de se lancer facilement dans ce genre de projet professionnel.

Réalités Alternatives et Objets Connectés

Nous disposons tous d'un ordinateur avec un clavier et une souris, et pour les plus chanceux, d'un écran tactile. Avec le Nabaztag, Rafi Haladjian a lancé l'ère de l'internet des objets ; l'idée première étant de ne plus accéder à l'information uniquement via notre ordinateur mais de rendre l'intelligence ambiante.

Jonathan Salamon
Développeur Android pour Accengage, Auto-entrepreneur spécialisé dans les objets connectés et Responsable du pôle développement de l'association Glass Camp; il développe des projets innovants (BikeFinder, CultureGlass, Wearable Intelligent Banking...)

https://fr.linkedin.com/in/jonathan-salamon-35067486
jonathan@glasscamp.org

Valentin Michalak
Microsoft Student Partner Lead, Ingénieur chez SOAT et Responsable technique de la Virtual Association.

https://www.linkedin.com/in/vmichalak
valentinmichalak@gmail.com

Carlos Rodriguez
Ingénieur en électronique et informatique industrielle, est impliqué dans un consortium européen du son immersif (projet Edison3D, Binaural) Responsable du pôle Création Matériel (Hardware) de l'association Glass Camp

www.glasscamp.org
https://www.linkedin.com/in/carlos-rodriguez-b5430241
carlos@glasscamp.org

Les objets connectés, de plus en plus présents dans nos vies, sont une bonne première étape mais aujourd'hui, les industriels réfléchissent déjà à l'étape suivante : la réalité virtuelle / augmentée. Microsoft, Google, Facebook, HTC, etc., tout le monde y va de son produit pour révolutionner les usages via des lunettes, casques et autres périphériques.
Nous commençons déjà à voir ce que cette disruption va nous réserver. Ces nouveaux objets vont aussi révolutionner le monde de la production et la diffusion audiovisuelle, la communication, le divertissement, l'éducation immersive ...
Via ce dossier consacré spécifiquement aux systèmes de réalité augmentée et virtuelle, nous allons tenter de vous montrer ce qui arrivera dans le futur et de comprendre comment tous ces produits vont venir changer notre quotidien.
Êtes-vous prêt à vous passer de votre écran pour vous immerger dans vos informations de manière virtuelle et/ou augmentée ?
Si tout cela vous paraît encore un peu abstrait, nous vous proposons, avant d'aller plus loin, de commencer par préciser ce dont il s'agit.

Aurélie Filippetti, Ministre de la Culture et de la Communication, en plein test de l'application CultureGlass, le 28 mai 2014 lors de la soirée de clôture de l'évènement "Silicon Valois", organisé au ministère.

Utilisation du bracelet Myo afin de contrôler la lecture d'une vidéo

Une utilisatrice d'Oculus Rift dans une montagne russe virtuelle

Différences entre Réalité Augmentée, Virtuelle, et Mixte (RA/RV/RM)

En un clin d'œil, un mouvement de main ou de tête, les objets intelligents nous permettent de se retrouver au milieu d'une plage, au fin fond de l'océan pacifique, en train de marcher le long de la muraille de Chine, ou dans n'importe quel endroit du monde avec la sensation d'être réellement sur place... mais sans y être.

Réalité Augmentée (RA)

La réalité augmentée désigne un procédé permettant de superposer des informations ou des objets qui n'existent pas dans la réalité. Elle vise à donner une plus-value au réel en l'augmentant. Bien que de nombreux exemples soient liés à la vision, la RA peut « augmenter » n'importe lequel de nos 5 sens.
La réalité augmentée est apparue avant la réalité virtuelle.
Un exemple simple : dans un musée, la RA peut permettre d'afficher des contenus contextuels avec une application dédiée, ainsi un visiteur peut faire apparaître des informations sur un tableau donné.

Interaction entre l'utilisateur et un monde virtuel avec le HTC Vive

Réalité Virtuelle (RV)

On appelle réalité virtuelle une forme de simulation dans laquelle le spectateur a l'impression d'être face à une scène réelle, voire dedans.
Elle permet de créer des expériences interactives immersives, qui plongent l'utilisateur au sein même d'autres mondes. Quand l'utilisateur bouge sa tête, la caméra de l'application ou du jeu (développé grâce à des logiciels spécifiques tels que Unity ou Blender 3D) suit le même mouvement. Avec la réalité virtuelle, vous prenez la place du héros de votre jeu vidéo.

DOMAINES | MAIL | HÉBERGEMENT | E-COMMERCE | **SERVEURS**

TESTEZ LE MEILLEUR CLOUD

TESTÉ ET APPROUVÉ PAR CLOUD SPECTATOR

Powered by intel Cloud Technology

1&1 Serveur Cloud : Easy to use – ready to Cloud*

Les performances des nouveaux serveurs Cloud sont imbattables en termes de CPU, RAM et SSD.

Réalisez vos projets dans le Cloud avec la combinaison parfaite entre flexibilité et performances.

- ✓ **Load balancing**
- ✓ **Stockage SSD**
- ✓ **Facturation à la minute**
- ✓ **Intel® Xeon® Processor E5-2660 v2 et E5-2683 v3**

1 mois gratuit !

Puis à partir de 4,99 € HT/mois (5,99 € TTC)*

1 MOIS POUR ESSAYER ★★★★★

1 CLIC POUR CHANGER DE PACK ★★★★★

1 APPEL UN EXPERT VOUS RÉPOND ★★★★★

0970 808 911
(appel non surtaxé)

1&1

1and1.fr

*Facile à utiliser - prêt pour le Cloud. 1&1 Serveur Cloud : 1 mois d'essai gratuit, puis à partir de 4,99 € HT/mois (pour la configuration du serveur Cloud S). Facturation mensuelle en fonction de la configuration choisie. Pas de durée minimum d'engagement, ni de frais de mise en service. Conditions détaillées sur 1and1.fr. Intel et le logo Intel sont des marques commerciales d'Intel Corporation aux États-Unis et/ou dans d'autres pays. 1&1 Internet SARL, RCS Sarreguemines B 431 303 775.

Réalité Mixte avec HoloLens

Réalité Mixte (RM)

C'est en effet Microsoft qui, avec ses lunettes Hololens, présente un nouveau concept de réalité mixte. Cette technologie prend les meilleurs aspects de la réalité virtuelle et de la réalité augmentée.
En fait, ce ne sont pas de simples données digitales qui sont amenées dans la vie réelle, mais des éléments vivants et non vivants de la réalité virtuelle. On parle alors d'hologrammes, et il est possible d'interagir avec eux.
Ce n'est pas tant l'immersion dans un autre monde, mais plutôt l'altération de l'espace autour de l'utilisateur, en faisant croire à notre cerveau que tout ceci est bien réel.

Google Glass, un premier essai grandeur nature

C'est la présentation et la commercialisation des Google Glass qui a le plus marqué les esprits. Objet prometteur et premier sur le marché, il faut avouer que les déceptions furent grandes (initialement bien aidé par une campagne de communication qui s'est révélée finalement balbutiante).
Néanmoins, cela nous a permis d'apprendre beaucoup de choses.

Pourquoi avoir une paire de Glass ?

Tout l'intérêt de ce produit réside dans l'enrichissement du "moment présent" : Prenez n'importe quelle situation où vous avez besoin d'une information rapide et contextuelle comme savoir votre vitesse à vélo, voir le plan d'un magasin, lire une notice de réparation, connaître l'heure de votre prochain rendez-vous, ... Les possibilités sont infinies et particulièrement riches, même dans un contexte professionnel on peut imaginer effectuer un acte chirurgical en étant guidé, obtenir le plan détaillé d'un lieu pendant une intervention dans un bâtiment (pompiers, police..), ... Ce sont de simples exemples de projets déjà réalisés et fonctionnels.
Alors, malgré ses possibilités immenses, pourquoi les Google Glass n'ont pas tenu leurs promesses

Le grand public, une erreur de cible

C'est malheureusement le constat qui ressort de nos expériences : le grand public n'est absolument pas prêt pour ce genre de produit. Une preuve parmi d'autres : les atteintes à la vie privée par des utilisateurs peu scrupuleux qui se sont amusés à utiliser la caméra incluse dans le produit afin de prendre furtivement des photos/vidéos.
En France, suite à un hackathon organisé par l'association Glass Camp, une "GlassWare" (application Google Glass) est née : BikeFinder (http://bikefinder.mobi/).

Aperçu de l'application BikeFinder sur le site MyGlass

Cette application qui est officiellement disponible sur MyGlass (magasin d'applications officiel) à pour but de vous aider à trouver une station avec des vélos en libre service disponibles mais surtout, lorsque vous êtes en train de rouler et que votre temps de location à expiré, de vous proposer la station la plus proche où des places sont disponibles pour garer votre vélo.
Bien qu'elle soit fonctionnelle dans 200 villes et 20 pays, elle totalise environ 300 utilisateurs actifs, ce qui tend à prouver que le grand public n'est pas encore prêt pour ce type d'usage. On peut bien sûr parler également du design, avoir l'air d'un robot ne met personne en valeur et, à force de se faire remarquer en portant ce genre d'objet, on en vient à les laisser de côté pour éviter d'avoir "l'air idiot".

Son prix et son autonomie, des critères décisifs

Le prix un peu excessif à payer au départ pour le premier prototype était de l'ordre de 1500$, ce qui n'est pas donné à tout le monde.
De plus, avoir une batterie qui ne tient que quelques heures, contrainte liée au design de l'objet, ce n'est malheureusement pas suffisant. Tant que ce type d'objet ne tombera pas à un coût abordable et ne tiendra pas au moins la journée en utilisation intensive, les utilisateurs autant grand public que professionnels passeront leur chemin et en concluront inévitablement qu'il ne s'agit que d'un Gadget.
De plus, l'objet a aussi tendance à chauffer, principalement lorsque celui-ci est en charge, ce qui empêche une utilisation prolongée même lorsque les lunettes sont branchées à une source d'énergie.

Google Glass, avant tout un prototype

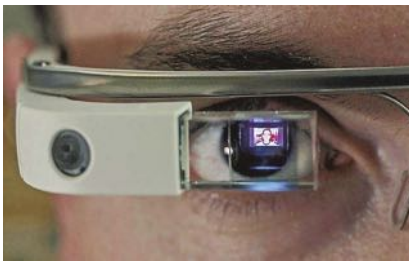
Les Google Glass telles que commercialisées (Explorer Edition, 1500 dollars), n'étaient et ne sont toujours qu'un prototype. Il y a beaucoup de chemin avant d'obtenir un produit final. Ce qui vient confirmer cette hypothèse : Le kit de développement qui est en "preview" (https://developers.google.com/glass/develop/gdk/) , le fait que la version d'Android tournant sur les lunettes est obsolète (4.0.4 puis 4.4) et le fait que ce n'est pas Android Wear qui tourne dessus mais une version allégée de Android (alors que le système Wear serait bien plus adapté et améliorerait probablement l'autonomie).
Il reste encore du chemin mais cela reste une expérience qui fut vraiment intéressante autant en tant que développeur que pour Google qui tirera sûrement les leçons nécessaires de ce semi-échec, et, on comprend pourquoi le projet a été remanié/repensé et s'appelle désormais le Projet Aura (http://www.projectaura.com/en/).
Les Google Glass ne sont que le début d'une nouvelle révolution.

Partenaires du programme "Glass At Work", preuve du changement de cible pour le produit :
Source : https://developers.google.com/glass/distribute/glass-at-work

État des lieux de ce qui existe

Google Glass

A la manière des notifications de nos smartphones, des informations contextualisées peuvent êtres affichées dans les lunettes (SMS, Facebook, Twitter, trouver des avis de restaurants, trouver votre chemin, afficher le dernier titre paru dans la presse, ...). On ne présente plus les Google Glass, qui peuvent être aujourd'hui considérées comme précurseuses. Elles restent la référence des lunettes transparentes monoculaires et l'affichage tête haute (en haut à droite de l'oeil) leur confèrent l'avantage qu'il est facile "d'oublier" qu'on en est porteur. L'information est là quand on en a besoin et la technologie "s'efface" quand elle n'est plus contextuellement nécessaire.



Epson Moverio

Lunettes connectées binoculaires tournant sous Androïd, elles sont principalement réservées aux professionnels.
En effet, elles offrent des normes similaires à celles requises sur des chantiers de construction par exemple.
Elles se pilotent via une télécommande (reliée par un câble aux lunettes) composée d'une batterie, de boutons et d'un touchpad (pour se déplacer dans l'interface). Son autonomie et son champ de vision sont supérieurs à ceux de ses concurrents.

Tango

Projet développé par Google qui consiste notamment à cartographier son environnement, c'est-à-dire la possibilité de recréer à l'identique notre environnement en 360° grâce à des capteurs extrêmement sophistiqués.
Des applications et des jeux sont disponibles qui ont la particularité d'offrir la possibilité "d'ancrer" des objets dans l'espace physique (c'est à dire qu'un objet virtuel reste à sa place dans le monde physique même si l'utilisateur se déplace. Il est donc possible de définir une zone de jeu, de se cacher derrière un objet, etc.)
Il est possible d'y ajouter un "support casque" afin d'ajouter une notion d'immersion similaire à Oculus Rift par exemple (https://www.durovis.com/product.html?id=5)

Oculus Rift

Premier casque de réalité virtuelle sorti ayant connu un réel succès, notamment auprès des joueurs. C'est un casque de réalité virtuelle produit par la société Oculus VR racheté par Facebook. Ces lunettes permettent une immersion totale dans un environnement entièrement virtuel. Elles ont un champ de vision très important et sont très réactives, ce qui en fait une référence pour les jeux vidéos. Disponibilité attendue : courant 2016

Hololens

Casque de réalité augmentée proposé par Microsoft permettant de porter les applications universelles Windows en surimpression, tels des hologrammes. Le casque est proposé en kit de développement au prix de 3000 €. Entièrement autonome, le casque ne nécessite ni téléphone, ni ordinateur pour fonctionner à l'inverse de ses concurrents. Son champ de vision est beaucoup plus grand que celui des Google Glass mais est encore assez limité.

Cardboard

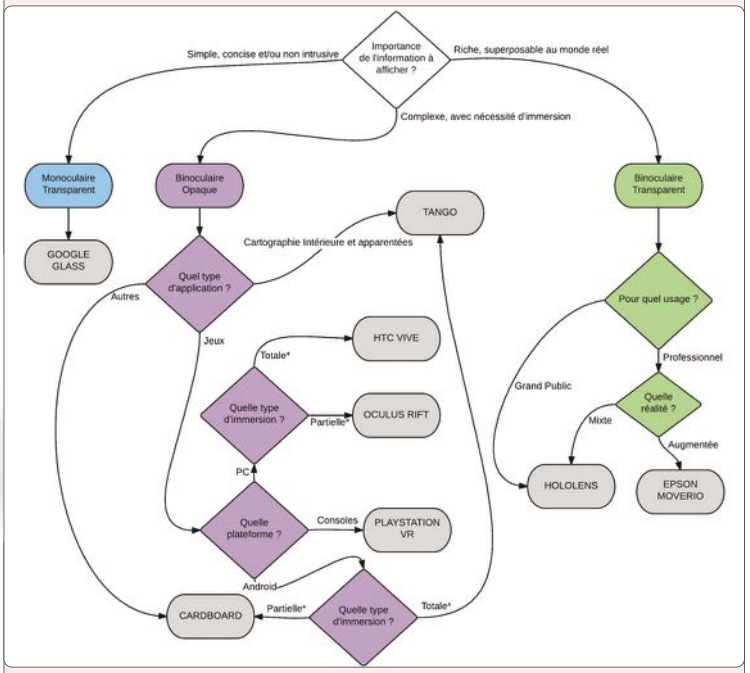
Fonctionne à l'aide d'un téléphone. C'est un casque en carton à fabriquer soi-même pour quelques dizaines d'euros. Il vise à créer un casque de réalité virtuelle fonctionnant sous Android. Google a mis en place un SDK spécifique au Carboard pour développer très rapidement des applications: https://developers.google.com/cardboard/
Il s'agit de la façon la moins onéreuse d'obtenir un casque de réalité virtuelle. Cet avantage leur ont permis d'être choisies pour des opérations marketings temporaires ou pour divers évènements/salons.

Les Autres

D'autres objets sont prometteurs, parmi eux on peut citer le HTC Vive (développé en partenariat avec Valve et dont la commercialisation est prévue à partir de Avril 2016), Magic Leap (qui vient de lever 827 millions de dollars), ainsi que le Playstation VR de Sony.

Comment Choisir ?

Il n'est jamais facile de comprendre et de choisir parmi les produits qui sont proposés à ce jour. Afin de vous aider, nous vous proposons ci-dessous un arbre de décision que nous avons simplifié au maximum. Veuillez noter que bien que cet arbre de décision ait pour but de vous aider à prendre une décision et à y voir plus clair concernant l'objet qui répondra à vos besoins, ce document doit être considéré à sa juste valeur : il s'agit d'un schéma s'ajoutant à une étude de marché que vous devrez avoir menée au préalable (ne pas considérer ce document comme référence absolue). De plus, cet arbre ne prend en compte qu'un nombre restreint de modèles disponibles à ce jour qui pourraient peut être répondre tout aussi bien ou même mieux à vos besoins spécifiques.



***Totale :** Signifie que se déplacer (marcher, courir, etc.) dans le monde physique provoque un mouvement similaire dans le monde virtuel. Un contrôleur supplémentaire reste nécessaire pour interagir à 100% (manipuler des objets) avec l'environnement virtuel.*

***Partielle :** Signifie que certains mouvements physiques sont pris en compte (comme les mouvements de tête, de bras et de mains). Un contrôleur supplémentaire est nécessaire afin d'effectuer des déplacements + interagir à 100% (manipuler des objets) avec l'environnement virtuel.*

Mapper du réel et du fictif avec Unity

Un éditeur pour les regrouper tous

Dans le monde des moteurs de jeu, Unity a su y creuser son petit trou. L'enfant de Monogame (créé en Juin 2005) est depuis devenu la référence des moteurs de jeu pour les créateurs indépendants et est sur le point de devenir (si ce n'est pas déjà le cas) la référence du monde de la RV / VA.
Les raisons de son succès sont très simple :

- Le moteur est compatible avec toutes les plateformes disponibles sur le marché ou presque, du PC à la PS Vita en passant par les smartphones Android et même Windows Phone, aucune plateforme ne lui résiste et aucun casque de réalité augmentée / réalité virtuelle non plus.
- La possibilité d'utiliser du C# ou/et du Javascript. Dans le monde du jeu-vidéo, le C++ à longtemps été la seule vrai solution ; avec Unity, une nouvelle voie s'est proposée et est plus adapté aux projets de petite ou moyenne envergure.

De nombreux jeux que vous connaissez sont conçus avec Unity et ils représentent la vitrine du moteur.

Configurer son environnement

Pour configurer son environnement pour faire de la VR, c'est assez simple. Vous devez disposer au minimum de la version 5.1 d'Unity pour pouvoir suivre ce tutoriel. C'est important car cette version intègre une nouvelle fonctionnalité intéressante autour de l'intégration des casques de réalité virtuelle. Créez un nouveau projet dans l'interface, vérifiez qu'il soit bien en 3D. Pour intégrer votre équipement de RV / RA il existe deux méthodes :

- La méthode utilisant des plugins que nous ne détaillerons pas dans cet article car elle est spécifique à chaque système.
- Et la méthode en utilisant l'intégration d'Unity, beaucoup plus simple et sur laquelle nous allons nous concentrer.

Pour activer l'intégration avec nos casques, il va juste falloir cocher une case Fig.1.

1 - Ouvrez le panneau qui se trouve dans Edit > Project Settings > Player,
2 - Ouvrez l'onglet "Others Settings"
3 - Activez la combobox "Virtual Reality Supported".

Et c'est parti, vous pouvez brancher votre paire d'Oculus !

Créer son premier projet

Pour votre premier projet de VR nous avons choisi de vous faire adapter un grand classique du jeu vidéo : le jeu de jongle.

Fig.1

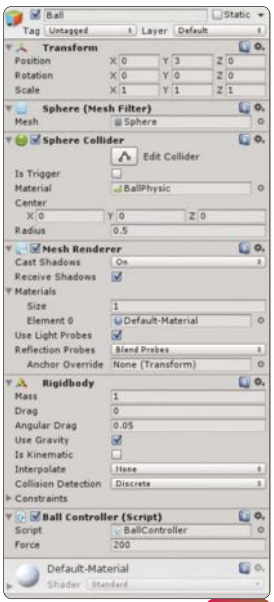


Fig.2

Ce type de jeu va permettre de prendre en main le "positional tracking" (système permettant de vous positionner dans l'espace) disponible notamment avec l'Oculus.
Il est assez simple à prendre en main et facilement adaptable.

Avant de commencer, prenez bien soin d'avoir suivi les étapes précédentes.

Tout d'abord petit point technique : nous allons tout faire comme si nous étions sur l'Oculus Rift DK2 car il s'agit du casque le plus répandu à l'heure actuelle et nous allons donc aussi apprécier ces limitations au niveau du "positional tracking" pour créer notre projet.
Ces limitations sont différentes en fonction du système que vous utilisez, faites-y attention. Dans notre cas nous allons donc être limité par le rayon de vision de la caméra infrarouge qui permet le "positional tracking". Pour commencer nous allons installer notre scène, elle va donc être composée de 4 murs qui seront représentés par 4 cubes, un plan qui servira de sol, une sphère qui servira pour faire des jongles, une lumière et une caméra associée à une zone de collision.
Nous allons donc créer un "GameObject" vide qui va contenir notre environnement.

1 - Faites un clic droit dans votre "hierarchy", "create empty", et nommez le "World" ; faites bien attention à ce qu'il soit positionné en 0,0,0 avec une rotation nulle sur les 3 axes.
2 - A l'intérieur de celui-ci, ajoutez un plan, ajoutez lui le tag "Ground" car cela nous servira par la suite.
3 - De même, ajoutez 4 cubes que vous élargirez afin d'en faire 4 murs (les murs doivent avoir un espacement maximum de 3 unités). Prenez soin que votre lumière et votre caméra soient initialement entre ces quatre murs.
4 - Lorsque vous aurez votre casque sur la tête, vous serez la caméra de votre scène. Positionnez -a donc en fonction, c'est à dire assez bas, proche du sol, dans votre scène.
5 - Nous allons appliquer le tag "Player" à notre caméra et lui accrocher une capsule à laquelle nous désactiverons le "Mesh Renderer" et nous donnerons également le tag "Player", c'est notre zone de collision avec notre balle.
6 - Nous allons créer une sphère que nous allons mettre en haut de notre scène, pour qu'elle soit soumise à la gravité, nous allons lui ajouter un "rigidbody".
7 - Créez un script "BallController" en C# et ajoutez le à votre sphère Fig.2 et 3.

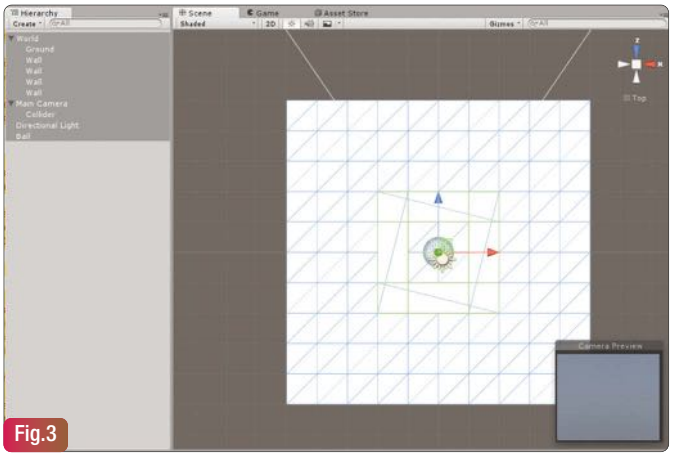


Fig.3

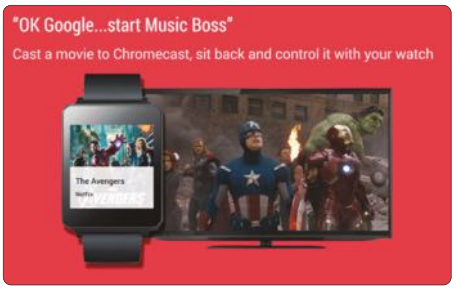
Bravo ! Notre scène est prête à recevoir les premières lignes de code !

Code de BallController.cs

```
using UnityEngine;
using System.Collections;

public class BallController : MonoBehaviour {
  [Tooltip("Rebound force.")]
  [SerializeField]
  private int force = 200; // Force de rebond

  private Rigidbody rb; // Physique de notre balle

  private Vector3 initialPosition; // Position initiale pour le reset

  /// <summary>
  /// Factory de notre objet
  /// </summary>
  void Start()
  {
      initialPosition = transform.position; // Sauvegarde la position initiale pour le reset
      rb = GetComponent<Rigidbody>(); // Met à disposition la gestion de la physique de la balle
  }

  /// <summary>
  /// S'exécute à chaque nouvelle frame
  /// </summary>
  void Update()
  {
      // Si une touche est appuyée
      if (Input.anyKeyDown)
      {
      transform.position = initialPosition; // Remettre la balle à sa position initiale
      }
  }

  /// <summary>
  /// S'exécute lorsque l'on rentre en collision
  /// </summary>
  /// <param name="collision"></param>
  void OnCollisionEnter(Collision collision)
  {
      switch (collision.gameObject.tag) // En fonction du tag de l'objet avec lequel on rentre en collision
      {
      case "Player": // Si collision avec un player
          rb.AddForce(0, force, 0); // Ajouter une force vers le haut pour faire rebondir la balle
          break;
      case "Ground": // Si collision avec le sol
          Debug.Log("Game Over"); // Ecrire dans la console de débug: Game Over
          break;
      }
  }
}
```

Une fois ce code ajouté, vous pourrez jouer avec votre Oculus.
Si vous le souhaitez, le projet est aussi disponible au téléchargement à l'adresse suivante :
https://dl.dropboxusercontent.com/u/13395728/BalloonSample.zip

Et après ?

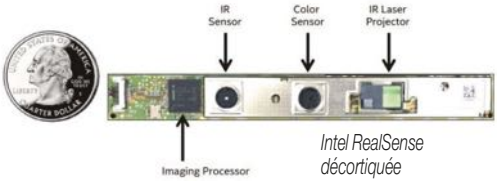


Nous avons presque terminé notre tour d'horizon, et, au delà du présent ou du futur proche, il est temps d'essayer d'entrevoir ce que nous réserve l'avenir. S'il est difficile de savoir ce qu'adviendra ce marché naissant, nous pouvons avoir quelques certitudes : L'avenir passera inévitablement par l'association de plusieurs objets afin de proposer de nouvelles expériences; toujours plus riches, toujours plus efficaces et toujours plus connectées "avec notre corps".

Une montre connectée comme contrôleur

Si vous êtes technophile, cela ne vous a sûrement pas échappé : Certaines montres connectées vous permettent déjà d'agir comme télécommande pour contrôler, par exemple, une vidéo diffusée sur votre télévision via Chromecast.
La montre connectée est appelée à se transformer de plus en plus afin de devenir une source d'information et un contrôleur universel.
On peut facilement imaginer une montre intelligente servant à enrichir l'expérience proposée par d'autres objets.
Un seul exemple pour vous en convaincre : Imaginez un jeu d'horreur sur un casque de réalité virtuelle; la montre pourrait facilement mesurer la fréquence cardiaque de l'utilisateur puis l'envoyer au jeu.
Ceci permettrait d'indiquer au joueur qu'il est temps d'arrêter de jouer ou au contraire, qu'il est temps de le stimuler (ici, lui faire peur) afin qu'il reste immergé au sein du jeu.

Intel RealSense & Kinect



Intel RealSense décortiquée

Kinect

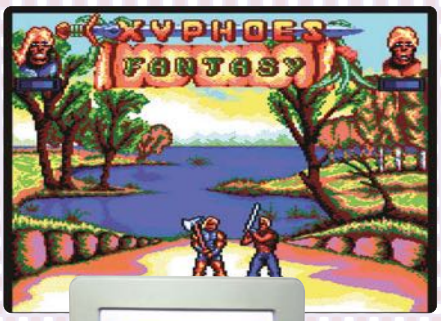
La Kinect de Microsoft et le capteur RealSense d'Intel ont le même objectif et le font de la même manière ou presque, capter votre corps via un système de caméra (notamment infrarouge) afin d'en faire une carte. Une fois que ceci est fait, il est assez simple de s'interfacer avec pour, par exemple, coder un petit jeu et vous immerger entièrement dedans.

Virtualizer

Le Virtualizer est un système permettant de capter vos mouvements pour les intégrer dans votre expérience. Marchez, courrez, sautez, tous vos mouvements seront captés mais vous ne bougerez pas dans la réalité car vous êtes maintenu sur place par le système. Système onéreux pour le moment, son évolution reste à surveiller.

SPÉCIAL VINTAGE

RETOUR VERS LE PASSÉ!



images : D.R.

Amstrad CPC

Atari ST

Amiga 500

Apple II

© Martin Mystère

Amiga 2000

Il est toujours difficile de terminer l'année et de bien débuter la nouvelle. La rédaction s'est beaucoup interrogée et un éclair de génie est arrivé : pourquoi ne pas faire un dossier vintage ? Notre article sur Amiga – Atari ST avait suscité de vives réactions... Cette fois-ci, nous avons décidé d'aller un niveau au-dessus.

Ces machines ont peut-être 30 – 35 ans, elles conservent une magie intacte. Et ce n'est pas un hasard si la démoscène est très active en France. Eh oui, on continue à coder sur ces machines et à rivaliser sur scène pour faire la meilleure démo ! Plusieurs démomakers vont le prouver et vous montrer quelques-unes des possibilités techniques sur Atari ST, Amiga et Amstrad CPC. Nous parlerons aussi Apple II et nous ferons du rétro-gaming avec la Gameboy !!!

Enjoy !

La rédaction

Programmation sur Apple II

Sorti en 1977, l'Apple II fut le roi des ordinateurs 8 bits, et c'est grâce à son succès et à sa très grande longévité qu'Apple a survécu aux échecs de l'Apple III et du Lisa, et au très lent démarrage du Macintosh. Conçu par Steve Wozniak, sa réussite est due principalement à son lecteur de disquettes apparu très tôt, et à son architecture ouverte. En effet, à l'heure où ses principaux concurrents chargeaient péniblement leurs programmes à partir de cassettes lentes et peu fiables, les disquettes de l'Apple II l'ont rendu compatible avec une utilisation professionnelle – Visicalc fut la « Killer Application » - et la possibilité d'insérer des cartes d'extensions lui a allongé sa durée de vie et surtout donné accès à des environnements complémentaires très riches comme CP/M.

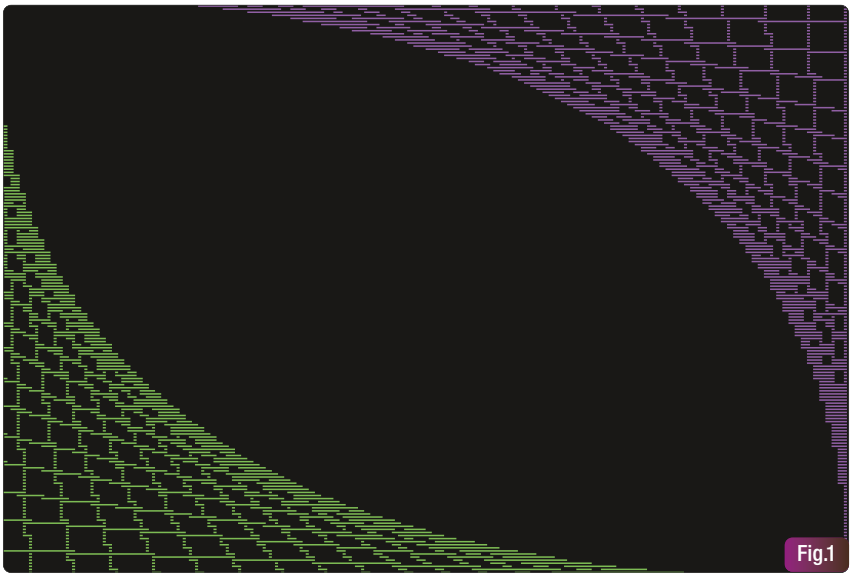


Fig.1

Alain Zanchetta
Développeur dans l'équipe Microsoft Hololens.

Avec ces deux éléments, la palette de langages de programmation de l'Apple II ne cède que devant celle du PC : Basic, assembleur, Pascal, C, logo, Lisp, Forth, Fortran, Modula II, Ada, etc. Il existe même une implémentation de Java, certes limitée, et le petit dernier de cette famille s'appelle Plasma, et est apparu en 2012, soit 35 ans après la machine elle-même.

Basic Applesoft et assembleur 6502

Pour le possesseur d'Apple II de l'époque, les langages rois étaient le Basic et l'Assembleur 6502. Ils étaient en effet disponibles en permanence, et ce, dès l'allumage de la machine. La plupart des jeux du commerce étaient écrits en Assembleur, une exception notable étant Wizardry (« Sorcellerie » dans l'hexagone) écrit en Pascal UCSD. Le Basic Applesoft, disponible à partir de la version II+, a été écrit par Microsoft qui fournissait une adaptation de ce langage à presque tous les constructeurs. Il est assez complet et permet d'accéder à l'ensemble des fonctions de la machine comme les graphismes, les ports de jeux ou encore les fichiers si un lecteur de disquettes est disponible et le système d'exploitation chargé. Lorsque des cartes d'extension sont installées, elles sont généralement accessibles via des appels aux fameux PEEK et POKE dans une plage d'adresse correspondant aux slots dans lesquels elles sont insérées. Voici un exemple de programme affichant des lignes de deux couleurs :

```
10 DX = 278 / 20:DY = 191 / 20
20 HGR : POKE  - 16302,0
30 FOR I = 0 TO 20
40 X = DX * I:Y = DY * I
50 HCOLOR= 1
60 HPLOT 0,Y TO X,191: HPLOT 1,Y TO X + 1,191
70 HCOLOR= 2
80 HPLOT 278,Y TO X,0: HPLOT 279,Y TO X + 1,0
90 NEXT
```

Il utilise la « haute » résolution : 280 x 192 pixels en 6 couleurs. Au passage, on peut noter que l'Apple II possède un mode graphique très astucieux – 280 x 160 pixels en 6 couleurs + 4 lignes de texte en bas de l'écran – très pratique pour certains jeux ou pour les applications professionnelles. Le Basic supporte aussi un mode basse résolution (40x48) mais celle-ci n'est quasiment jamais utilisée. Pour être exhaustif, il faut noter que les derniers modèles de la série supportent des résolutions plus variées comme 140 x 192 ou 560 x 192, mais celles-ci ne sont pas accessibles aux programmes Basic, et elles ne sont donc que rarement utilisées Fig.1.

Les fonctions de dessin du Basic sont assez limitées et consistent essentiellement au tracé de points avec HPLOT ou de lignes avec HPLOT TO. Il existe une notion de SHAPE mais celle-ci est très limitée et n'a rien à voir avec les sprites disponibles sur des ordinateurs à cible essentiellement ludique.

Le développeur de jeux sur Apple II doit donc prendre tout en charge lorsqu'il veut afficher un personnage ou un vaisseau spatial : les opérations logiques sont reines ici, avec les AND pour effacer des parties de l'écran et des OU pour fusionner les bits du pseudo-sprite et ceux du fond d'écran.

La bonne nouvelle est que l'Apple II possède deux pages graphiques identiques qui permettent de réaliser une sorte de double-buffering : le programme dessine la frame suivante dans la page qui n'est pas affichée, puis affiche celle-ci une fois la frame entièrement préparée. Ceci permet d'éviter des clignotements désagréables pour l'utilisateur. Néanmoins, le Basic est trop lent pour réellement tracer des formes à l'écran, et seul le recours au langage machine permet d'obtenir de bons résultats.

En termes de sons, le Basic de l'Apple II n'offre qu'un bip émis lorsque CHR$(7) est envoyé à la routine d'affichage. Jouer des sons plus sophistiqués est possible : accéder à l'adresse mémoire $C030 déplace la membrane du haut-parleur. Des lectures successives permettent donc de jouer une large gamme de notes en contrôlant la fréquence de ces opérations. Néanmoins, le Basic est trop lent et le langage machine est là encore la seule solution.

Graphismes avancés, sons variés, voire vitesse de calcul, le recours à l'assembleur est donc très souvent nécessaire sur l'Apple II. Pour cette raison, cet ordinateur a été conçu pour offrir une parfaite cohabitation entre le Basic et les routines 6502. D'une part, la ROM contient un moni-

teur permettant d'entrer directement des codes hexadécimaux en mémoire ainsi qu'un mini-désassembleur, et d'autre part, les appels entre Basic et langage machine sont possibles dans les deux directions :

- Depuis Basic, CALL adr appelle la routine assembleur située à l'adresse adr. L'organisation de la mémoire de l'Apple II est bien documentée et permet au développeur de savoir où placer ses routines assembleur : une petite fonction est généralement chargée à l'adresse 768 où plus de 200 octets sont inutilisés par le Basic. Si la partie assembleur est plus conséquente, les instructions LOMEM et HIMEM permettent de déplacer la mémoire réservée au Basic (programme et variables).

```
100 POKE 6, X : POKE 7, Y : POKE 8, A : REM X,Y=position
A=numéro sprite
110 CALL 768 : REM trace le sprite
```

- L'instruction Basic & déclenche automatiquement un appel à la routine assembleur dont l'adresse est stockée en 0x3F5: le programme peut alors analyser la suite de l'instruction (qui se finit soit en fin de ligne, soit au prochain « : »), y compris en s'appuyant sur l'interpréteur Basic, par exemple pour lire des constantes ou des variables. Ce simple & permet donc d'étendre le vocabulaire du Basic et un certain nombre de bibliothèques étaient ainsi vendues.

```
100 & DRAWSPRITE X,Y,A
```

- Réciproquement, les adresses en ROM des différentes routines du Basic étaient documentées, permettant aux programmes écrits en assembleur de ne pas tout ré-implémenter.

Un peu de pratique

Pour entrer de petites routines simples, le développeur peut appeler le moniteur de l'Apple II via la commande CALL -151, puis entrer les codes hexadécimaux des instructions 6502 en les faisant suivre l'adresse cible et « : ». Le mini désassembleur, appelé en entrant une adresse suivie de L permet de vérifier le code saisi... ou d'inspecter le code présent dans la ROM de l'Apple II.

- Appel du moniteur :

```
]CALL-151
```

- Entrée de la routine présentée précédemment :

```
*300: A0 00 B1 06 F0 07 20 F0 FD C8 4C 02 03 60
```

- Affichage du code 6502 de la routine COUT du Basic :

```
*FDF0L
```

```
FDF0- 48       PHA
FDF1- C9 A0    CMP  #$A0
FDF3- 4C 95 FC JMP  $FC95
FDF6- 48       PHA
FDF7- 84 35    STY  $35
FDF9- A8       TAY
FDFA- 68       PLA
FDFB- 4C 46 FC JMP  $FC46
FDFE- EA       NOP
FDFF- EA       NOP
FE00- C6 34    DEC  $34
FE02- F0 9F    BEQ  $FDA3
FE04- CA       DEX
FE05- D0 16    BNE  $FE1D
FE07- C9 BA    CMP  #$BA
FE09- D0 BB    BNE  $FDC6
FE0B- 85 31    STA  $31
FE0D- A5 3E    LDA  $3E
FE0F- 91 40    STA  ($40),Y
FE11- E6 40    INC  $40
FE13- D0 02    BNE  $FE17
FE15- E6 41    INC  $41
FE17- 60       RTS
```

Pour les programmes assembleurs plus conséquents, il est préférable de recourir à un macro-assembleur comme Merlin.
Celui-ci permet au développeur de donner des noms aux emplacements mémoire stockant les variables, ainsi qu'aux sous-programmes et différentes branches du programme.
Voici le source Merlin de la routine présentée précédemment :

```
STRING =  $6
COUT   =  $FDF0
*
    ORG  $300
*
    LDY  #$0
LOOP  LDA  (STRING),Y
    BEQ  END
    JSR  COUT
    INY
    JMP  LOOP
END   RTS
```

Merlin est assez compact et se place dans la mémoire à l'emplacement habituel des programmes Basic, ce qui permet au développeur de modifier son source, de l'assembler et de le tester sans avoir à recharger l'environnement depuis le lecteur de disquettes.
Paradoxalement, la programmation en assembleur est réputée difficile mais les processeurs comme le 6502 sont en fait très faciles à programmer car ils ont peu d'instructions différentes : pour l'essentiel, le programme charge une valeur dans un registre, modifie cette valeur via une opération arithmétique élémentaire (addition ou soustraction) ou une opération logique et sauve la nouvelle valeur en mémoire. Les opérations effectuées sur les registres comme l'accumulateur modifient des bits du registre d'état et d'autres instructions effectuent des branchements en fonction de ces bits. Le 6502 possède essentiellement trois registres dans lesquels le programmeur peut enregistrer des valeurs : A, X et Y. En revanche, il possède un grand nombre de modes d'adressage et le minuscule programme présenté plus haut en utilise trois différents :

- LDY #$0 adressage immédiat = le paramètre est une constante
- JSR $FDF0 adressage absolu = le paramètre est une adresse
- LDA ($06),Y adressage indirect indexé : la valeur de Y est ajoutée à la valeur présente aux adresses $06 et $07 pour donner l'adresse qui va être finalement référencée. C'est un mode très utile pour toutes les fonctions graphiques.

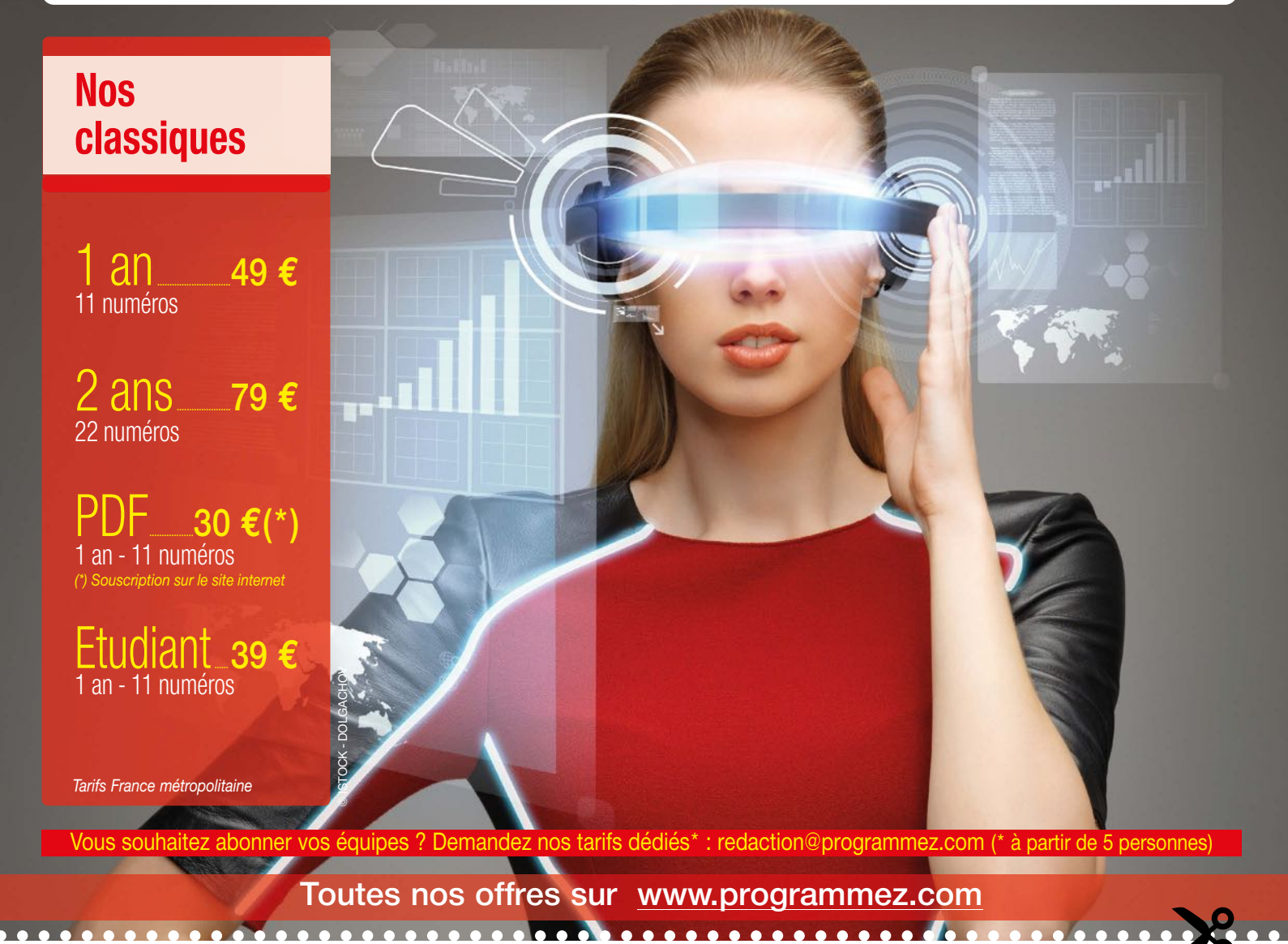
Pour clore ce paragraphe sur le Basic et le langage machine de l'Apple II, il reste à décrire l'espace mémoire de la machine : en effet, sur ces ordinateurs 8 bits, la mémoire virtuelle n'existe pas et le développeur doit gérer précisément où stocker son code et ses données qui doivent cohabiter avec les différents éléments du système comme la ROM ou les zones correspondant aux pages graphiques ou texte. Voici la cartographie d'un système Apple IIe de 64 Ko :

Adresses	Utilisation
$0000-$00FF	Appelée Page 0, cette zone est très utilisée par le système car le 6502 a un mode d'adressage spécifique qui permet d'économiser un octet et un cycle CPU par instruction. Il reste néanmoins des adresses qui ne sont pas utilisées par le système et permettent au développeur d'exploiter ce mode d'adressage.
$0100-$01FF	Pile du 6502.
$0200-$02FF	Buffer d'entrée.
$0300-$03FF	Quelques vecteurs d'interruptions sur le haut de la page mais emplacement idéal pour stocker de petites routines 6502 cohabitant avec le Basic
$0400-$07FF	Page texte ou graphiques basse résolution.
$0800-$0BFF	Page de texte ou graphiques basse résolution ; plus généralement le début des programmes Basic.
$0C00-$1FFF	Disponible.
$2000-$3FFF	Page graphique haute résolution 1.
$4000-$5FFF	Page graphique haute résolution 2.
$6000-$BFFF	Disponible. Le haut de la zone contient généralement le DOS qui se charge en RAM à partir d'une disquette.
$C000-$CFFF	Adresses de pilotage du matériel et des cartes périphériques.
$D000-$DFFF	ROM / Ram Bloc 1 / Ram Bloc 2
$E000-$FFFF	ROM / Ram

Tous les mois, faites le plein de codes

Abonnez-vous à programmez!

le magazine du développeur



Nos classiques

1 an ..... **49 €**
11 numéros

2 ans ..... **79 €**
22 numéros

PDF ..... **30 €(*)**
1 an - 11 numéros
(*) Souscription sur le site internet

Etudiant ..... **39 €**
1 an - 11 numéros

Tarifs France métropolitaine

Vous souhaitez abonner vos équipes ? Demandez nos tarifs dédiés* : redaction@programmez.com (* à partir de 5 personnes)

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

PROG 192
***Offre limitée**, valable jusqu'au 31 janvier 2016*

☐ 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*

Les plages $D000-$DFFF et $E000-$FFFF peuvent correspondre à de la ROM ou de la RAM : en accédant à certaines adresses spécifiques de la zone $C000-$CFFF, on peut rendre l'un ou l'autre de ces blocs visibles. Cette technique – appelée Bank Switching – permet à l'Apple IIe d'avoir 64 Mo de RAM et 12 Ko de ROM alors que le 6502 ne peut adresser directement qu'un espace de 64 Ko.

Par exemple :

- $C082 active la ROM pour l'espace entre $D000 et $FFFF ;
- $C08B active la RAM pour ce même espace ;
- $C083 active le 2e bloc de RAM pour $D000-$DFFF ;
- $C089 active la lecture de la ROM mais l'écriture du premier bloc de RAM.

Les configurations à plus de 64 Ko comme l'Apple IIc ou l'utilisation de cartes d'extensions s'appuient sur un usage extensif de cette technique.

Modern Fun

Bien que l'Apple II approche de ses quarante ans, il est au cœur d'une communauté très active et on continue à voir apparaître pour cette machine du matériel, du logiciel et même quelques livres. Le matériel se décompose essentiellement en deux types : d'une part des adaptateurs permettent de remplacer les éléments fragiles comme les lecteurs de disquettes (cartes mémoire + images disque) ou les vieux moniteurs (adaptateurs VGA), d'autre part de vraies cartes d'extension (parfois des clones d'anciens modèles) étendent les possibilités de l'Apple II comme des cartes mémoire, une carte Ethernet ou encore une carte son. Les logiciels se répartissent de manière habituelle entre jeux (ex Zephyr) et utilitaires (ex Merlin32). Finalement les livres sont souvent des rééditions de livres mythiques (Red Book, Roger Wagner Assembly Lines) voire des vraies nouveautés (The New Apple II User's Guide).

Pour les programmeurs, il est plus facile de développer aujourd'hui pour un Apple II que dans les années 80 en combinant la puissance des machines modernes, quelques outils de développement croisé, les logiciels de manipulation d'images disques et les émulateurs. A titre d'illustration, voici comment développer un classique du langage C et le déployer sur un Apple II.

CC65 est un cross compilateur pour 6502 (http://cc65.github.io/cc65/) qui possède les bibliothèques minimales pour cibler l'Apple II mais aussi les machines Commodore (Vic 20, C64, etc.) ou Atari. Pour éditer un programme C pour Apple II, on peut tout simplement utiliser Visual Studio qui va apporter la coloration syntaxique ainsi que la complétion automatique.

```
#include <stdio.h>
#include <conio.h>

int main()
{
   printf("Hello world\n");

   cgetc();
   return 0;
}
```

La compilation de ce programme s'effectue en ligne de commande avec cl65 :

```
cl65 -O -t apple2  helloworld.c
```

L'étape suivante consiste à ajouter l'exécutable généré à une image disque : cela peut être fait manuellement via CiderPress (http://a2ciderpress.com/) ou en ligne de commande à l'aide d'Apple Commander (http://applecommander.sourceforge.net/).

```
java -jar C:\Download\Apple_II\AppleCommander\Apple
Commander-1.3.5.jar -cc65 prodos.
dsk helloworld bin < helloworld
```

Enfin, on peut lancer un émulateur comme AppleWin (https://github.com/ AppleWin/AppleWin) en lui passant l'image disque ainsi créée :

```
applewin -d1 prodos.dsk
```

Une fois dans l'émulateur, on peut voir l'exécutable et le lancer : Fig.2.

Les différentes commandes ci-dessus peuvent être facilement automatisées et il devient alors possible de lancer directement l'émulateur et le programme compilé en appuyant simplement sur F5 sous Visual Studio.

Le cycle édition / compilation / test ainsi obtenu est bien plus rapide qu'avec le matériel et les logiciels d'époque mais il est possible d'aller encore plus loin car Applewin incorpore un débogueur 6502 : Fig.3.

Les développeurs préférant programmer en assembleur ne sont pas en reste par rapport au C :

- Brutal Deluxe a mis récemment sur son site Web (http://www.brutaldeluxe.fr/) Merlin32 qui est une adaptation de Merlin aux machines modernes et permet l'assemblage de code 6502 ou 65816 (processeur de l'Apple IIgs) depuis Windows, OS X ou encore Linux.
- Olivier Guinart développe un « Language Service » pour Visual Studio apportant la coloration syntaxique et l'auto-complétion au code 6502.

Evidemment, si on dispose d'un vrai Apple II, la dernière étape est de lancer l'exécutable sur celui-ci. Les solutions pour récupérer des fichiers à partir d'un PC sont assez nombreuses aujourd'hui :

- Câble série ou carte Uthernet et logiciel ADT Pro ;
- Simulateur de lecteur de disquette (CFFA 3000, Floppy Emu Disk Emulator, Unisdisk, etc) ;
- Carte Apple II PI et Raspberrry PI servant de passerelle.

Néanmoins, ceci n'est pas indispensable et les différents émulateurs disponibles permettent de retrouver le goût de la madeleine à la pomme sans matériel spécifique...

Fig.2

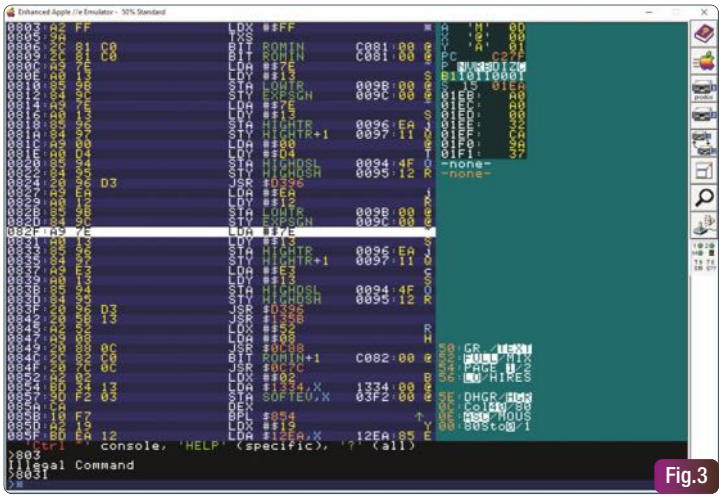


Fig.3

L'Amstrad CPC

Dans les années 80, Amstrad n'est encore qu'une firme britannique spécialisée dans le matériel HIFI et vidéo dont le fondateur, Alan Michael Sugar (AMStrad) ambitionne d'attaquer le marché de la micro -informatique.

Baptiste Bideaux
Consultant en Digital Learning
chez PwC France.

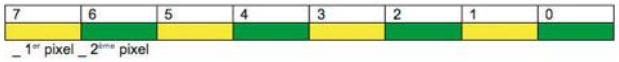
Le projet « Colour Personal Computer » démarre au début de l'année 1983, et en 1984 apparaît sur le marché l'Amstrad CPC 464, ordinateur 8bits avec 64ko de RAM, équipé d'un microprocesseur Zilog Z80 cadencé à 4MHz, d'un magnétophone à cassette et d'un magnifique clavier QWERTY coloré. Il faudra attendre 1 an pour voir arriver la version disquette de la machine : le CPC 664. La même année débarque le CPC 6128, une version du 664 avec une RAM augmentée à 128 ko. La mémoire primaire (64 ko de RAM) est partagée en 4 banques de 16 ko chacune (numérotées de 0 à 3). Le CPC 6128 propose 64ko supplémentaires qui sont eux aussi partagés en 4 banques de 16 ko accessibles en assembleur.

Affichage

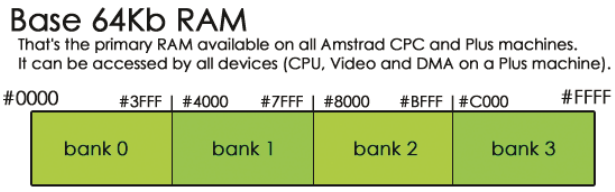
Equipé d'un moniteur monochrome ou couleur (selon les finances de chacun), la gamme CPC propose 3 modes d'affichage :
MODE 0 : 160x200 en 16 couleurs
MODE 1 : 320x200 en 4 couleurs
MODE 2 : 640x200 monochrome
La banque 3 située entre l'adresse &C000 et &FFFF est réservée pour l'affichage écran. Quel que soit le mode d'affichage, il y a toujours 200 lignes. Chaque ligne à une taille de 80 octets. Ainsi la première ligne démarre à &C000 et se termine à &C049. Chaque ligne est dessinée de manière linéaire, de gauche à droite. Cependant, à la fin d'une ligne le tracé effectue un saut de 8 pixels plus bas. Si &C000 correspond aux coordonnées (0,0), &C050 correspond à (0,8). Pour accéder à la ligne directement en dessous d'une autre, il faut incrémenter 1000 (&800) à l'adresse de cette dernière. Donc la seconde ligne sur notre écran (0,1) se trouve à l'adresse &C800.
La synchronisation vidéo est gérée par le CRTC (Cathode Ray Tube Controller), mais les couleurs sont gérées par le Gate Array qui ne raisonne pas en binaire, mais en état (0,1 ou 2). C'est pour cela que nous avons une palette de 27 teintes et non 16 ou 32. Les couleurs utilisables (16 maximum en MODE 0) peuvent être choisies parmi cette palette de teintes. Attention, les teintes de palette sont indexées différemment selon que l'on y accède par le BASIC ou par le Gate Array.

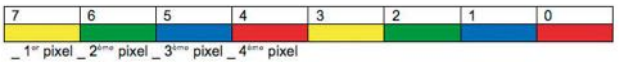
Organisation des pixels

Selon le MODE d'affichage choisi, la manière de coder les pixels est sensiblement différente.
En MODE 0, un octet suffit pour afficher 2 pixels à l'écran, un pixel ayant une longueur de 4 bits. Ainsi, la couleur du premier pixel est codée sur les bits 7, 5, 3 et 1. Le second pixel sur les bits 6, 4, 2, 0. C'est ce que l'on appelle l'entrelacement des pixels.

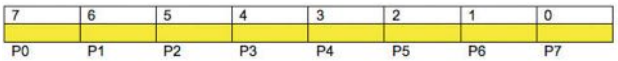


En MODE 1, l'entrelacement est identique au MODE 0. Cependant, le pixel n'est plus codé sur 4 bits, mais sur 2.





En MODE 2, c'est plus simple. Chaque bit d'un octet correspond à 1 pixel.



Comme on peut le voir, le 1er pixel se situe au bit N° 7.

La programmation

Toute la gamme CPC est livrée avec un BASIC intégré dans la ROM (le Locomotive BASIC), qui permet à l'utilisateur de faire ses premiers programmes assez rapidement. Le langage est vraiment très complet pour l'époque et fournit beaucoup de commandes pour dessiner sur l'écran, jouer des sons, gérer les périphériques (comme les 2 manettes de jeu). A noter que le BASIC du CPC 6128 propose des commandes qui n'existent pas dans le BASIC du 464 et 664. A l'époque, certains programmes étaient donc uniquement prévus pour le 6128. Le BASIC reste interprété. Aucun compilateur n'existe à l'époque. On peut toutefois protéger son code contre son listage (afin de garder un droit de propriété) et son interruption d'exécution. Même s'il reste plus lent qu'un programme codé en assembleur, le BASIC de l'Amstrad permet tout de même de produire des programmes de jeux honorables. Les puristes vous diront que pour développer pour l'Amstrad CPC il faut un Amstrad CPC. Mais bon, ceux qui n'ont pas la chance de posséder un vrai CPC peuvent se tourner vers les émulateurs qui feront très bien l'affaire. Le plus utilisé est sans aucun doute WinAPE (http://www.winape.net), qui a l'avantage d'émuler tous les Amstrad CPC et de proposer un assembleur pour le Zilog Z80. Il émule par défaut la gamme CPC+, mais le BASIC reste identique aux versions CPC plus anciennes. Téléchargez l'archive ZIP depuis le lien http://www.winape.net/download/WinAPE20A18.zip et extrayez tous les fichiers dans un dossier (« winape » par exemple). Exécutez le fichier WinAPE.exe. L'émulateur se lance et vous voilà sur l'écran d'accueil du BASIC.

Jouons avec les modes

Par défaut, le mode d'affichage est en MODE 1 (320x200 en 4 couleurs), soit 40 colonnes sur 25 lignes de caractères.
Saisissez le code suivant, en appuyez sur ENTREE pour l'exécuter :

```
MODE 0↵
```

L'affiche change. L'écran est nettoyé et la résolution est passée en 160x200 en 16 couleurs, soit 20 colonnes sur 25 lignes de caractères. Les caractères sont désormais 2 fois plus larges qu'en MODE 1.
Saisissez le code suivant, en appuyez sur ENTREE pour l'exécuter :

```
MODE 2↵
```

L'affiche change à nouveau et la résolution est passée en 640x200 en monochrome, soit 80 colonnes sur 25 lignes de caractères. Les caractères sont désormais 2 fois moins larges qu'en MODE 1.
Saisissez le code suivant, en appuyez sur ENTREE pour l'exécuter :

```
MODE 1↵
```

L'affichage est revenu à son état initial.

Un peu de couleurs

Le BASIC utilise les couleurs indexées de 0 à 15. Chacune est associée par défaut à l'une des 27 teintes de la palette du CPC. Saisissez le code suivant :

```
10 MODE 0↵
20 FOR R=0 TO 15↵
30 PEN R :PRINT « COULEUR » ;STR$(R)↵
40 NEXT R↵
RUN↵
```

Nous venons d'écrire un programme qui sera exécuté à votre demande. Pour exécuter un programme BASIC, vous devez saisir RUN et appuyer sur la touche ENTREE.

Voici les 16 couleurs par défaut que vous pouvez utiliser en BASIC. La couleur 0 est la couleur utilisée pour le fond d'écran (bleu nuit). Les couleurs 14 et 15 sont clignotantes car 2 teintes leur ont été associées. En effet, le BASIC permet de définir 2 teintes pour une couleur afin de créer un effet de clignotement.

Explication du programme :
Ligne 10 : Passage de l'affichage en MODE 0
Ligne 20 : Début de la boucle
Ligne 30 : On utilise PEN pour la sélection de la couleur du texte et PRINT pour afficher le texte.
Ligne 40 : Fin de la boucle
Restez sur cet écran, et saisissez ceci :

```
INK 1,15↵
```

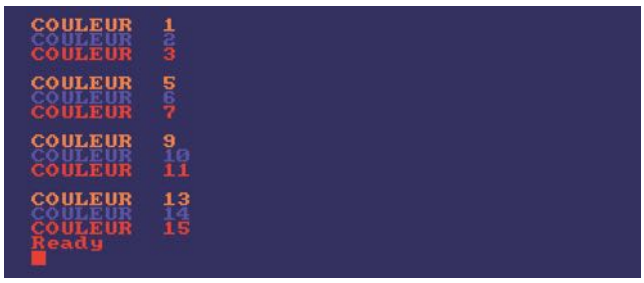
Après exécution, la couleur 1 a changé et est devenue Orange. La commande INK permet d'associer une teinte de la palette à un index de couleur. La teinte doit être comprise entre 0 et 26. Maintenant, saisissez ceci :

```
INK 2, 5 ,4↵
```

La couleur 2 est devenue clignotante car nous avons associé 2 teintes à cette couleur (5 et 4). Que ce passe-t-il si nous changeons de mode d'affichage ?

```
10 MODE 1↵
RUN↵
```

Exécutez à nouveau le programme en saisissant RUN et en appuyant sur la touche ENTREE.



Voilà comment s'affichent nos couleurs en MODE 1. Nous avons bien nos 16 couleurs, mais seulement 4 teintes différentes (les teintes associées aux couleurs 0, 1, 2 et 3). En effet, le MODE 1 ne permet pas d'afficher plus de teintes. Essayons avec le MODE 2, saisissez le code suivant et exécuter le programme :

```
10 MODE 2↵
RUN↵
```

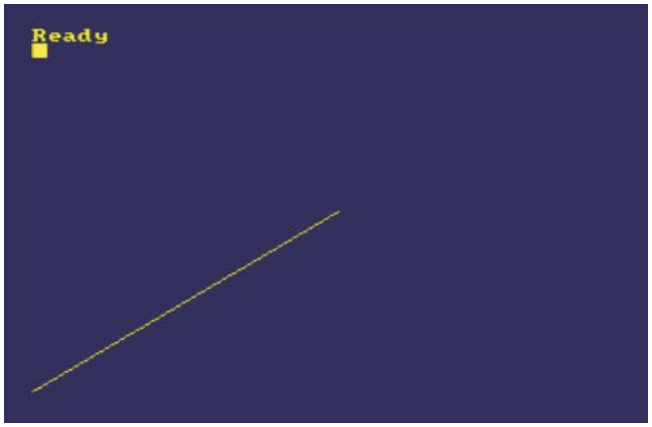
En MODE 2, seules 2 teintes sont affichables (les teintes associées aux couleurs 0 et 1).

Dessinons un peu

L'écran du CPC est orthonormé et a pour origine le coin inférieur gauche. L'axe des abscisses (X) évolue de 0 à 639, l'axe des ordonnées (Y), de 0 à 399. Et cela, quel que soit le MODE d'affichage utilisé. Il faudra donc bien avoir en tête l'organisation de ce plan avant de produire des tracés sur l'écran. Dès le démarrage du CPC, le curseur graphique est placé en coordonnées absolues (0,0). Ce curseur se déplacera en fonction de nos tracés et restera positionné sur le dernier point donné.
Commençons par supprimer le programme précédent :

```
NEW↵
10 MODE 1↵
20 MOVE 0,0↵
30 DRAW 319,199↵
RUN↵
```

Ce programme trace à l'écran une ligne partant du point(0,0) et s'arrêtant au point(319,199). Comme vous pouvez le constater, la ligne part du point inférieur gauche et s'arrête au centre de l'écran. Comme indiqué plus haut, l'axe des abscisses (X) évolue de 0 à 639, l'axe des ordonnées (Y), de 0 à 399. Le point (319,199) correspond donc au centre de l'écran.



Explication du programme :
Ligne 10 : Affichage en MODE 1
Ligne 20 : Place le curseur graphique aux coordonnées absolues (0,0)
Ligne 30 : Trace une ligne depuis la position du curseur graphique jusqu'au point (319,199)

Saisissez désormais ce code :

```
30 DRAW 639,399
RUN
```

La ligne traverse l'écran. En modifiant le MODE d'affichage (ligne 10 du programme), vous constaterez que le tracé reste identique. Seuls les pixels composant le tracé seront différents (plus larges en MODE 0, plus fins en mode 2).

Redéfinir les caractères de l'Amstrad.

L'une des possibilités offerte par le BASIC Amstrad est de pouvoir redessiner la table de caractères du CPC. Saisissez ce programme :

```
NEW
10 MODE 1
20 SYMBOL AFTER 47
30 SYMBOL 48,254,130,130,130,226,226,254,0
RUN
```

Il ne semble rien se produire. Cependant si vous saisissez le texte « 007 », vous constaterez que la forme du caractère « 0 » a changé.

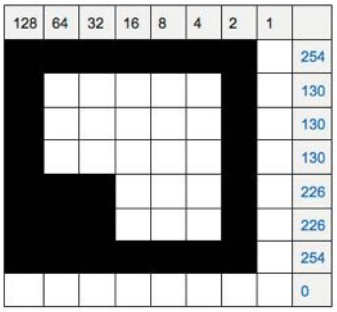
Explication du programme :

Ligne 10 : Affiche en MODE 1

Ligne 20 : SYMBOL AFTER indique que la redéfinition des caractères démarre après l'index 47.

Ligne 30 : SYMBOL définit le caractère 48 (code ASCII du chiffre « 0 »).

Un caractère CPC est contenu dans une matrice 8 x 8. Chaque ligne de la matrice est codée sur 8 bits, le bit 7 correspondant au premier pixel de la ligne. Chaque pixel d'une ligne correspond à une valeur.



Note : *Vous pouvez modifier les caractères de 32 à 255. Les caractères 0 à 31 sont appelés caractères de contrôle et ne peuvent pas être redéfinis par l'utilisateur. Attention, la redéfinition des caractères 32 à 128 aurat un impact direct sur les caractères saisis au clavier et affichés à l'écran (chiffres, lettres, ponctuation...). Si vous ne souhaitez pas modifier la police de caractères, vous pouvez uniquement modifier les caractères à partir de l'index 129.*

Assemblage de symboles

Pour rappel un caractère est contenu dans une matrice de 8x8 pixels. Mais comment faire si l'on veut un graphisme plus grand ? Pour cela, nous utiliserons un assemblage de symboles.

Rond comme un ballon, aussi jaune qu'un citron...

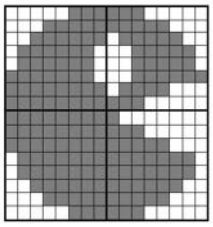
Nous allons réaliser un programme BASIC permettant d'afficher un PAC-MAN de 16x16 pixels en nous servant de l'assemblage de symboles.

Réinitialisez le CPC avec le code suivant :

```
CALL 0
```

La commande CALL permet d'appeler un sous-programme en langage machine à partir de l'adresse mémoire donnée. Ici, l'adresse 0 permet de réinitialiser complètement l'ordinateur.

Pour dessiner notre PAC-MAN, nous avons besoin de redéfinir 4 caractères, qui se présentent ainsi :

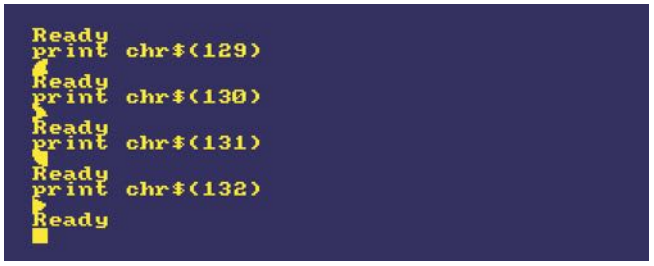


Afin de ne pas modifier les caractères saisis au clavier, nous allons redéfinir les caractères 129, 130, 131 et 132. Saisissez le programme suivant :

```
10 MODE 1
20 SYMBOL AFTER 128
30 SYMBOL 129,7,31,printadr,126,126,254,255,255
40 SYMBOL 130,224,248,124,62,62,63,248,224
50 SYMBOL 131,255,255,255,127,127,63,31,7
60 SYMBOL 132,128,224,248,254,254,252,248,224
RUN
```

Vérifions que nos caractères ont bien été modifiés en saisissant :

```
PRINT CHR$(129)
PRINT CHR$(130)
PRINT CHR$(131)
PRINT CHR$(132)
```



La commande CHR$ permet d'utiliser le caractère N de la table de caractères du CPC. Ici, on affiche nos 4 caractères redéfinis.

Assemblage !

Utiliser le code ci-dessus à chaque fois que l'on veut afficher notre PAC-MAN n'est pas vraiment pratique. Nous allons donc assembler les morceaux de notre PAC-MAN. Saisissez à la suite du programme, les lignes suivantes :

```
70 PAC1$ = CHR$(129)+CHR$(130)
80 PAC2$ = CHR$(131)+CHR$(132)
RUN
```

Les lignes 70 et 80 initialisent 2 variables PAC1$ et PAC2$. Le typage des variables est très simple en BASIC Amstrad. Le « $ » indique que la variable est de type « chaîne de caractères». Pour les nombres, il n'y a pas de symbole. Une variable ne peut pas avoir plus de 8 caractères.

PAC1$ assemble les caractères 129 et 130 l'un à coté de l'autre. PAC2$ fait de même pour les caractères 131 et 132. Saisissez les lignes suivantes :

```
PRINT PAC1$
PRINT PAC2$
```

Nous constatons que nos 2 variables affichent les parties haute et basse

de notre PAC-MAN. C'est mieux, mais on peut faire encore plus avec l'utilisation des caractères de contrôle. Nous allons créer une 3ème variable qui assemblera le tout. Saisissez ces lignes à la suite du programme :

```
90 PACMAN$ = PAC1$+CHR$(8)+CHR$(8)+CHR$(10)+PAC2$
RUN
```

La ligne 90 initialise la variable PACMAN$ avec nos 2 précédentes variables, assemblées grâce à 2 caractères de contrôle 8 (déplacement du curseur texte vers l'arrière) et un caractère de contrôle 10 (déplacement du curseur texte vers le bas). Voyons comment s'affiche désormais notre héros :

```
PRINT PACMAN$
```

L'affichage de notre héros est simplifié par la variable PACMAN$ qui assemble les symboles de caractères 129, 130, 131 et 132 d'une manière précise. Cette technique d'assemblage est très utilisée dans les jeux en BASIC. Mais l'exécution reste cependant lente pour afficher un assemblage plus grand. De plus, nous travaillons avec des caractères et sommes donc contraints d'utiliser les coordonnées texte et non graphique.

PAC-MAN version Sprite

Un sprite est un élément graphique, stocké en mémoire qui est dessiné sur l'écran. Sur CPC, l'affichage d'un sprite doit faire appel à une routine en assembleur afin d'être rapidement exécuté. Il serait possible de le faire en BASIC, en dessinant point par point. Mais la lenteur de l'interpréteur rendrait le programme inexploitable. Saisissons le programme suivant :

```
NEW
10 MEMORY &3FFF
20 REM Données du sprite (16 lignes de 4 octets)
30 DATA 0, 112, 224, 0
40 DATA 16, 240, 240, 128
50 DATA 48, 240, 112, 192
60 DATA 112,224,48,224
70 DATA 112,224,48,224
80 DATA 240,224,48,240
90 DATA 240,240,240,128
100 DATA 240,240,224,0
110 DATA 240,240,128,0
120 DATA 240,240,224,0
130 DATA 240,240,240,128
140 DATA 112,240,240,224
150 DATA 112,240,240,224
160 DATA 48,240,240,192
170 DATA 16,240,240,128
180 DATA 0,112,224,0
```

La ligne 10 réserve la mémoire après l'adresse &3FFF. Les lignes 30 à 180 contiennent les valeurs de chacun des octets de notre sprite. Chaque ligne du sprite est codée sur 4 octets (1 octet = 4 pixels de large en MODE 1). Notre PAC-MAN à donc une taille de 64 octets. Nous allons stocker notre sprite à partir de l'adresse &4000. Ajoutons les lignes suivantes à notre programme :

```
190 RESTORE 30
200 FOR ADR=&4000 TO &4000+63
210 READ A :POKE ADR,A
220 NEXT ADR
RUN
```

Explication du programme :

Ligne 190 : On branche le lecteur de DATA sur la ligne 30 de notre programme

Ligne 200 : Début de notre boucle. La variable ADR contient l'adresse mémoire à alimenter.

Ligne 210 : READ lit une valeur depuis la ligne DATA, la stocke dans la variable A et avance vers la prochaine valeur. POKE attribut la valeur de A à l'adresse mémoire ADR.

Ligne 220 : fin de la boucle.

Il ne se passe rien après l'exécution. C'est normal. Nous avons besoin d'une routine en assembleur pour afficher notre sprite. Pour cela, ouvrez la fenêtre Assembleur de WinAPE (touche F3). Créez un nouveau fichier et saisissez le code assembleur suivant :

```
org &6000
ld hl, &4000
ld de, &c000
ld bc, 16
affiche push bc : push de
ld bc, 4
ldir
pop de
ex hl, de ; call &bc26 ; ex hl, de
pop bc
dec b ; jp nz, affiche
ret
```

Assemblez le code en appuyant sur les touches Ctrl+F9. Notre programme en assembleur peut désormais être utilisé par le BASIC.

Revenez sur l'écran du BASIC et saisissez à la suite du programme BASIC les lignes suivantes.

```
230 CALL &6000
240 GOTO 240
RUN
```

Et voilà. Notre sprite apparait. A l'affichage, il n'y a aucune différence entre la version précédente du programme et la version sprite. Cependant, comme un sprite est un élément graphique, il peut contenir plusieurs nuances de couleurs et surtout être déplacé de manière plus fluide. De même, sur des éléments plus grands, l'affichage d'un sprite est largement plus rapide que l'utilisation de plusieurs symboles. Pour interrompre le programme qui boucle en ligne 240, pressez 2x la touche Echap. Améliorons notre sprite en lui apportant plus de couleurs, et pour le coup nous allons travailler en MODE 0. Saisissons un nouveau programme :

```
10 MODE 0 : MEMORY &3FFF:RESTORE 30
20 FOR R=&4000 TO &4000+ &40:READ A$:POKE R,VAL("&h"+A$):NEXT
30 DATA 00,CF,CF,00,45,CA,C0,80
40 DATA 45,C0,C0,2A,CF,D0,60,95
50 DATA CA,D0,E0,95,CA,D0,E0,80
60 DATA CA,D0,E0,00,CA,C0,80,00
70 DATA CA,C0,00,00,CA,C0,80,00
80 DATA CA,C0,C0,00,CA,C0,C0,80
90 DATA CF,C0,C0,95,45,C0,C0,80
100 DATA 45,CA,95,00,00,CF,CF,00
```

En MODE 0, notre sprite mesure 8x16 pixels. Les pixels étant de largeur double. Nous avons donc besoin de 1 octet pour afficher 2 pixels de couleur. Vous remarquerez que les valeurs contenues dans les lignes de DATA sont exprimées en hexadécimal. On aurait pu les écrire en nombre. C'est juste un confort de lecture.
La ligne 20 charge en mémoire les données à partir de l'adresse &4000. Nous allons à nouveau utiliser le code assembleur précédent pour affichage notre sprite. Ajoutons les 2 lignes suivantes à notre programme :

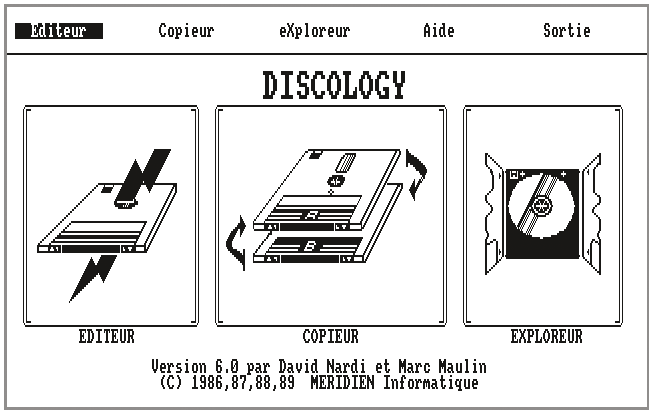
```
110 CALL &6000
120 GOTO 120
RUN
```

Notre PAC-MAN a pris des couleurs. Les pixels du MODE 0 sont plus larges.

Utilisation des différents modes d'affichage

Le MODE 0 permet de profiter de graphismes colorés mais manque de finesse. La plupart des jeux sur CPC utilisent ce mode.

Le MODE 1 offre des graphismes plus fins, mais moins colorés. Pour avoir des effets de dégradé ou donner le sentiment d'avoir plus de nuances de couleurs, le graphiste utilisait très souvent le principe de tramage, combinaison de pixels de couleurs différentes.



Le MODE 2 est surtout utilisé pour les applications (bureautique, utilitaires système...).

Il est possible, à l'aide de routines en assembleur, de partager horizontalement l'affichage en 2 modes différents. Comme ici, l'aire de jeu est en MODE 0 (16 couleurs), et le panneau de score en MODE 1. Il n'est cependant pas possible de superposer 2 modes différents ou de partager l'écran verticalement.
Certains programmes permettent l'affichage plein écran appelé « overscan ». La technique consiste à éliminer la bordure autour de l'écran et ainsi augmenter la résolution. Il s'agit de travailler sur les registres 1, 2, 6 et 7 du CRTC (Cathode Ray Tube Controller ou Contrôle du générateur de signal vidéo). La résolution finale atteint :
192 x 272 pixels en MODE 0.
384 x 272 pixels en MODE 1.
En BASIC il est possible d'obtenir le même effet en saisissant :

```
10 CLS
20 OUT &BC00,1 :OUT &BD00,50
30 OUT &BC00,2 :OUT &BD00,51
40 OUT &BC00,6 :OUT &BD00,34
50 OUT &BC00,7 :OUT &BD00,35
```

Visionnaire

En 1985, Alan Michael Sugar, PDG d'Amstrad explique sa vision de la micro-informatique :
« Les jeux c'est fini. Dans les années qui viennent, c'est sûr que ce marché va chuter et de beaucoup. Les gens cherchent quelque chose de plus sérieux, de professionnel... Je crois énormément au traitement de texte. » .
Heu.... (Source Amstrad Magazine N° 4 - Novembre 1985)

La gamme Plus

En 1990, Amstrad renouvelle la gamme des CPC en lançant la gamme Plus. Il s'agit de la même configuration (464 et 6128 uniquement), mais avec une carrosserie plus actuelle (plastique blanc) et une palette étendue à 4096 teintes (mais toujours 16 couleurs utilisables en MODE 0) et permet la gestion de 16 sprites hard (et zoomables). Cette gamme est aussi l'occasion pour Amstrad d'attaquer le marché du jeu vidéo (tenu par SEGA et Nintendo) en proposant la GX 4000, un CPC 464+ packagé dans une console de salon, auquel on a supprimé le lecteur de K7 et doté d'un support cartouche (support qui équipera également la gamme des CPC+). Les machines auront peu d'intérêt par rapport aux versions précédentes. La palette de 4096 couleurs n'est pas accessible par le BASIC, et le nouveau MODE 3 n'est qu'un MODE 0 en 4 couleurs qui ne sera jamais utilisé par les programmeurs. Malgré le soutien des éditeurs de jeux Européens (français surtout), la gamme disparaît des rayons à peine un an après sa mise sur le marché.

Où voir des CPC en action ?

La scène CPC est très présente en Europe, et plusieurs jeux indépendants sortent chaque année. La scène Espagnole est très active dans le développement de jeux. En France est organisée chaque année la démoparty ReSet (anciennement nommée Amstrad Expo) à Coutances, qui s'est déroulée en Juin cette année. On peut y rencontrer des passionnés du CPC, des concours de démos, des œuvres d'infographistes et des customs de machines. Pour en savoir plus, vous pouvez vous rendre sur cette page : http://reset.pushnpop.net .

Programmer en C pour CPC

Bien que l'assembleur soit le langage favori des CPCistes, il est possible de programmer en langage C et de compiler son code afin de l'exécuter sur CPC (émulateur ou vraie machine). Le compilateur C SDCC (Small Device C Compiler) fonctionne sous Windows et permet (entre autres) de réaliser des programmes pour Amstrad et de générer du code pour le Zilog Z80. (http://www.cpcwiki.eu/index.php/SDCC_and_CPC)

Découvrir ou redécouvrir le CPC

Logiciels d'époque
http://www.cpc-power.com
http://www.cpcgamesreviews.com

Tutorials de programmation assembleur et basic, articles techniques
http://www.cpcmania.com
http://www.cpcrulez.fr

AMSTRAD CPC 464, mon amour !

Après avoir fait mes armes en programmant des calculatrices (TI 57, FX 180P, PB 100), j'ai eu envie, tel l'extraterrestre du quartier, de faire l'acquisition d'un véritable ordinateur. A l'époque, tous ceux qui étaient proposés se ressemblaient beaucoup de par leurs possibilités, leur langage de programmation (le Basic) et leur prix. Compte tenu du fait que je devais le payer avec mes propres deniers, j'ai cherché un bon compromis entre prix et capacités. Mon choix s'est porté sur l'Amstrad CPC 464 couleur, qui à l'époque (1984) m'a coûté un mois de salaire de job d'été (650 euros en gros, oui mes jobs d'été étaient bien payés).

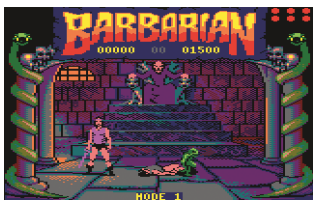
Stéphane Sibué
Directeur Technique chez GUYZMO
Microsoft MVP Windows Platform Development
www.guyzmo.fr
stephane.sibue@guyzmo.fr

C'était une vraie machine de course, processeurs Z80 à 4 Mhz, 64 Ko de Ram (kilo octets oui), clavier mécanique (très rare à l'époque) et lecteur de cassettes intégré, pas pour écouter de la musique, mais pour sauvegarder les programmes écrits en Locomotive Basic (un autre basic que celui de Microsoft). Ecran couleur intégré (une révolution) et un processeur sonore permettant de faire de véritables morceaux de musique par programme sans effort.

Jouer ou programmer ?

Généralement les gens faisaient ce genre d'acquisition pour deux raisons. Tout d'abord les jeux, qui étaient dignes des consoles de l'époque, et la programmation. Perso, vous le savez déjà, c'était pour la programmation, mais je dois avouer que j'ai aussi pas mal joué (j'avais 16 ans alors vous pensez bien). Le plus rigolo avec les jeux, du moins pour moi, ce n'était pas tellement d'y jouer mais plutôt de les pirater. Avec les cassettes, les premiers temps, il suffisait de posséder un bon copieur de cassettes et le tour était joué, les jeux étaient copiés en quelques minutes (oui minutes, l'unité de temps des chargements par cassette). Ensuite les éditeurs ont mis en place des stratagèmes pour éviter ce type de copie et il fallait en passer parfois par quelques lignes d'assembleur pour « casser » la protection, charger en mémoire les différents modules du jeu et les enregistrer ensuite sur une autre cassette, du grand art !

Le jeu Sorcery



Un must, le jeu Barbarian

Locomotive Basic

Coté programmation, l'Amstrad était servi par un basic de très grande qualité. Il était très rapide, et permettait d'utiliser toutes les ressources de la machine (sons, interruptions, ports, vidéo). Sur les 64 Ko de RAM, il ne restait que 42 Ko de disponible pour les programmes en basic, car il fallait enlever la mémoire utilisée par l'interpréteur et aussi la mémoire vidéo de 16 Ko. Coté vidéo c'était assez rustique. 3 modes étaient disponibles. Plus on voulait de couleurs différentes simultanées (dans une palette de 27 couleurs), moins on avait de pixels à l'écran :

Mode	Couleurs	Pixels	Caractères
0	16	160x200	20X25
1	4	320x200	40x25
2	2	640x200	80x25

Chose importante et très vintage, le Locomotive Basic était un Basic avec des numéros de lignes, comme tous les Basic de l'époque d'ailleurs. Quelle galère ces numéros de lignes quand j'y pense !
Mis à part ça, avec ce Basic, il était possible de faire des choses très sympathiques. Par exemple, il était possible de définir des interruptions au sein même du programme. S'il fallait exécuter un sous-programme qui se trouve en ligne 100, 2 fois par seconde, il suffisait d'écrire :

```
EVERY 5,0 GOSUB 100
```

Et toutes les 1/2 seconde (le 5 indique 5/10ème de seconde, soit 1/2 seconde), avec le timer 0, on saute automatiquement à la ligne 100. A la 1ère instruction RETURN rencontrée, le programme reprend là où il a été interrompu. L'Amstrad possédait 4 timers distincts (0 à 3) ce qui permettait de réaliser des choses bien pratiques très simplement. L'instruction AFTER, avec la même syntaxe, permettait de réaliser la même opération mais une seule fois et non cycliquement. Dans tous les cas, Il fallait juste bien penser à suspendre les interruptions (instruction DI) et les réactiver (instruction EI) aux bons endroits pour éviter des effets de bord induits par le fait que le programme pouvait être interrompu n'importe quand, c'est-à-dire dans n'importe quel état. Une très bonne mise en bouche pour la synchronisation d'applications.
Il était aussi possible de mélanger graphiques et caractères ou d'afficher des caractères sur des coordonnées graphiques. On pouvait scruter l'état du clavier, du joystick (indispensable pour jouer), imprimer, créer jusqu'à 8 fenêtres de texte différentes ayant chacune leur propre système de positionnement, lire et écrire des fichiers sur cassette (puis disquettes), charger des blocs de programmes au fur et à mesure des besoins, etc.

Le lecteur de disquettes, indispensable !

Très rapidement un investissement s'est imposé. L'achat d'un lecteur de disquettes, pour accélérer drastiquement les temps de chargement et profiter, oh comme c'est bon, des joies de l'accès aléatoire aux données (contrairement à l'accès séquentiel depuis une bande magnétique). Le lecteur de disquette était fourni avec un système d'exploitation très utilisé à l'époque, le CP/M. En gros, CP/M est l'ancêtre de MS-DOS. Il était utilisé principalement sur des machines professionnelles. L'intérêt a été pour moi de me familiariser en avance aux commandes typiques des systèmes d'exploitation de l'époque (DIR, COPY, REN). Amstrad fournissait aussi le DR Logo (DR pour Digital Research), un langage très riche et bien sympathique permettant de dessiner des figures géométriques (avec la tortue) mais pas que. Il était pourvu d'instructions permettant par exemple d'ajouter des propriétés supplémentaires à des « objets » préexistants et même de créer ses propres objets ; là aussi une mise en bouche pour la programmation orientée objet... avant l'heure. Grâce au CP/M j'ai pu aussi m'initier au Turbo Pascal de Borland qui était disponible (craqué, mais il y a prescription Monsieur le Procureur) pour cet OS.

Le lecteur de disquettes était aussi utilisable depuis le Basic, et franchement ça a changé complètement l'utilisation de mon Amstrad ! C'était de la folie, des disquettes d'une capacité incroyable de 178 Ko par face (eh oui, c'était trop de la balle), et surtout la possibilité de lire et d'écrire des données sans intervention de la part de l'utilisateur (plus besoin d'indiquer de mettre une cassette, de la caler au bon endroit et d'appuyer sur REC). Mon cher CPC devenait presque une machine de pro ! Les disquettes avaient un format qui a été abandonné par la suite, c'était des disquettes 3 pouces, mais elles étaient assez robustes ; 30 ans après j'arrive toujours à les lire !

Du basic à l'assembleur Z80

Très vite une communauté s'est créée autour de cette machine, des journaux se sont spécialisés dans la publication de programmes et d'infos la concernant, des éditeurs ont sorti énormément d'ouvrages permettant d'accéder aux ressources cachées de la bête (utilisation de routines en code machine depuis le basic via les instructions PEEK, POKE et CALL).

Le mensuel Amstrad magazine

La bible du programmeur

Toute cette littérature (seul média disponible à l'époque) a permis à beaucoup d'entre nous de progresser et d'aller de plus en plus loin dans l'utilisation de leur Amstrad.

En fait, mes débuts en programmation ont été un peu atypiques. Contrairement à beaucoup de personnes qui ont commencé à développer sur des calculatrices ou sur des ordinateurs personnels (Amstrad, Comodore 64, Oric, TO7, MO5, Amiga, etc), moi j'ai commencé directement par l'assembleur dès 12 ans, et tout ça à cause d'un concours de circonstances (je ne vais pas entrer dans les détails ici).

C'est tout naturellement qu'après l'utilisation d'un bon Basic ou d'un bon Turbo Pascal (langages de haut niveau), j'ai eu envie de retourner aux sources et de me remettre à développer en assembleur (non, ce n'est pas sale) pour tirer le plus de puissance de la bête. Donc je me suis mis à coder pour le Z80, le processeur de l'Amstrad (le même que celui de la Game Boy). Ce processeur de Zilog était très répandu à l'époque, et pour tout vous dire, on le trouve encore aujourd'hui dans les calculatrices programmables de chez Texas Instrument. Là aussi ce fut une belle expérience qui m'a servi très souvent dans mon travail par la suite.

```
 Program example1.asm      Line 1    Col 1    Free 31909  Insert
        org &4000
        nolist

        jp main

printstr:
        pop bc
        pop hl
        push hl
        push bc
printstrloop:
        ld a, ( hl )
        or a
        ret z
        call &bb5a
        inc hl
        jp printstrloop
L1:
        ret
printstrln:
        pop bc
        pop hl
        push hl
        push bc
```

Un peu d'assembleur ?

30 ans après toujours là

Il y a quelques semaines j'ai ressorti mon Amstrad de sa boîte, il a démarré sans broncher, 30 ans après son achat et au moins 10 ans après sa dernière mise en route. Puis j'ai branché son lecteur de disquettes, et là, catastrophe, impossible de lire la moindre disquette. Après un petit tour sur Internet, j'ai compris qu'il fallait changer la courroie d'entrainement. Un fois chose faite, il est reparti comme en 14 et j'ai pu montrer à mon plus jeune fils (18 ans tout de même) les jeux que j'avais et les programmes que j'avais créés, 15 ans avant sa naissance. Il a adoré et a même joué pendant un moment à certains jeux auxquels je jouais moi même. Si on m'avait dit ça il y a 30 ans...

Si vous aviez un Amstrad et que vous ne pouvez plus l'utiliser, il existe sur le net de très bons émulateurs. Personnellement je vous recommande le site WinAPE (http://www.winape.net/) dans lequel vous trouverez un émulateur Amstrad de grande qualité, pourvu de fonctions très puissantes et pratiques (je vous laisse découvrir).

Le Commodore Amiga

Entre 1985 et 1994, Commodore International commercialise ce qui deviendra une machine de légende : l'Amiga. En l'espace de neuf années de commercialisation, la gamme se verra déclinée en pas moins de 10 versions, dont la plus célèbre reste l'Amiga 500, sorti en 1987.

Baptiste Bideaux
Consultant en Digital Learning chez
PwC France.

L'Amiga est devenu très populaire dans les années 1990. Il séduit les infographistes, devient un outil indispensable pour la TV et le cinéma, et le secteur du jeu vidéo l'adopte complètement, proposant même des titres rien que pour lui. Ses points forts : un affichage hyper-coloré pour l'époque, une fluidité d'animation, un système multitâche et une interface graphique similaire au Macintosh. Si Commodore a disparu, l'Amiga perdure grâce notamment à la société Amiga Inc, qui continue de faire évoluer la gamme. Bien sûr, les machines proposées aujourd'hui n'ont plus rien à voir avec les premières architectures d'époque. D'ailleurs, les anciennes versions sont nommées « Amiga Classics », afin de différencier la vieille gamme de la nouvelle.

Les 3 chipsets de l'Amiga

Selon les versions de l'Amiga, on peut trouver 3 chipsets différents. L'OCS (Original Amiga Chipset) fut le premier chipset proposé avec l'Amiga 1000 et les premières versions de l'Amiga 500 et 2000. Une version améliorée nommée ECS (Enhanced chipset) équipe les Amiga 500+, 600 et 3000. La troisième génération de chipset se nomme AGA (Advanced Graphics Architecture) ; elle équipe les Amiga 1200, 4000 ainsi que la console de jeu Amiga CD32. Les possibilités multimédia (audio et vidéo) sont fournies par la puce « Agnus ». Cette puce évolue en fonction de la génération de chipset. Si la première version (OCS) est capable de manipuler 512ko de mémoire, une nouvelle version appelée « Fat Agnus » et équipant les dernières versions de l'Amiga 500 et 2000 manipule jusqu'à 1Mo de mémoire. La puce du chipset ECS tout comme celle de l'AGA permet de manipuler jusqu'à 2Mo de mémoire et se nomme « Super Agnus ». La vidéo de l'Amiga est proposée en 2 spécifications : PAL(Europe) et NTSC(US). Le nombre de lignes affichables en NTSC étant plus faible qu'en PAL, il n'était pas rare de trouver des jeux en Europe qui n'occupaient pas toute la hauteur de l'écran, enfin de répondre également au marché US.

Les résolutions les plus courantes

Spécifications	Basse résolution	Haute résolution	Haute résolution entrelacée
PAL (50 Mhz)	320x256	640x256	640x512
NTSC (60 Mhz)	320x200	640x200	640x400

L'entrelacement consiste à afficher en alternance une trame constituée des lignes impaires puis celle des lignes paires. L'exploitation de ce procédé vidéo provoque un scintillement de certaines zones de l'écran plus contractées. La rétro compatibilité entre les différents chipsets est gardée. On retrouve donc dans le chipset AGA, toutes les résolutions d'affichage des versions précédentes.

Les couleurs

Chipset	Basse résolution	Haute résolution	Haute résolution entrelacée	Palette
OCS	2 à 32	2 à 16	2 à 16	4096
ECS	2 à 32	2 à 16	2 à 16	4096
AGA	2 à 32	2 à 16	2 à 16	16,777,216

La génération de signaux RVB est prise en charge par le circuit du chipset nommé « Denise ». Le nombre de couleurs à l'écran peut être augmenté en utilisant 2 modes de couleurs :

EHB (Extra Half Bright) qui permet d'afficher jusqu'à 64 couleurs en basse résolution. Les couleurs 32 à 63 sont en fait les nuances en demi-teintes des 32 premières couleurs.

HAM (Hold And Modify) permet d'utiliser la totalité de la palette de 4096 couleurs (OCS et ECS). Le chipset AGA utilise lui le **HAM-8,** permettant l'exploitation de 262 144 couleurs.

Le site **Amiga Graphics Archive** présente en détail l'ensemble des résolutions gérées par les différentes versions d'Amiga. Beaucoup de ces résolutions sont destinées à l'exploitation TV, mais peu sont utilisées dans les applications et jeux vidéo (notamment les hautes résolutions entrelacées) : http://amiga.lychesis.net/knowledge/ScreenModes.html .

Le système vidéo de l'Amiga fonctionne en mode bitplane. Pour une image de 256 couleurs, la machine superpose 8 plans pour afficher l'image. A l'époque, la qualité graphique produite par l'Amiga était supérieure à celle d'un PC VGA. Cependant, pour une telle image, la vitesse d'affichage est divisée par 8.

Programmation

Afin d'animer cet article, et découvrir le potentiel graphique de l'Amiga, vous aurez besoin : d'un émulateur, d'un langage de programmation, d'une disquette de travail.

Emulateur

WinUAE sera utilisé. Il émule parfaitement l'ancienne gamme de l'Amiga et est simple à configurer. Vous pouvez le télécharger sur le site officiel : http://www.winuae.net, dans la rubrique Downloads. Prenez les archives ZIP si vous souhaitez éviter la procédure d'installation. La version de WinUAE qui sera utilisée dans cet article est la **version 3.2.1.**

Les ROMs Amiga étant copyrightées, WinUAE propose par défaut l'utilisation d'une ROM publique nommée AROS. Elle est parfaitement compatible avec les programmes Amiga et on reste dans la légalité.

Langage de programmation

Nous allons utiliser le langage le plus populaire de l'époque : **AMOS Basic**. Il s'agit d'un BASIC développé par François Lionet en 1989. Il facilite le développement de jeux vidéo sur Amiga, mais permet aussi de développer des applications plus classiques. Il a l'avantage d'être indépendant du système d'exploitation de l'Amiga et donc de s'affranchir des contraintes de l'OS. De ce fait, il communique directement avec le processeur 68000 et la ROM, et permet de profiter pleinement du potentiel graphique de l'Amiga. Il a été décliné en une version Pro, apportant plusieurs améliorations. Nous n'utiliserons que la première version, qui sera amplement suffisante pour cet article.

AMOS Basic est disponible aujourd'hui sous licence BSD. Le code source a été mis à la disposition du public par son auteur. Vous pouvez l'utiliser à des fins personnelles. Pour plus d'informations : https://en.wikipedia.org/wiki/AMOS_%28programming_language%29

Je vous propose de télécharger l'archive suivante :
http://www.baptiste-bideaux.com/amosbasic/amosbasic134.zip
Elle contient 2 images de disquette (au format ADF) utilisables sous WinUAE.

- Le logiciel AMOS Basic dans sa version 1.34
- Notre disquette de travail (programmez.adf)

Configuration de l'émulateur

Lancez WinUAE. Après l'affichage d'un message vous informant des ROMs manquantes, le panneau de configuration de WinUAE s'affiche à la rubrique « Quickstart ». La configuration actuelle suffira. Il s'agit d'un Amiga 500 équipé du chipset OCS et d'une RAM de 1Mo. Il faut insérer une disquette dans le lecteur. Pour cela, cliquez sur le bouton « Select image file » de la partie « Emulated Drives » et sélectionnez le fichier « AMOS The Creator v1.34 (1991)(Europress).adf » extrait de l'archive précédemment téléchargée Fig.1. Cliquez ensuite sur le bouton « Start » situé en bas de la fenêtre pour lancer l'émulation. Après quelques secondes de chargement, l'éditeur d'**Amos Basic** apparait Fig.2.

Premier programme

Dans l'éditeur, saisissez ce programme :

```
Screen Open 0,320,200,16,Lowres
Print "Programmez! Le magazine du developpeur"
Wait Key
```

Exécutez votre programme en appuyant sur la touche F1.

Explication du programme

Ligne 1 - Le programme ouvre l'écran N°0 de résolution 320x200 en 16 couleurs, et en basse résolution (mot clé « Lowres »).

Ligne 2 – Ecrit « Programmez ! Le magazine du développeur » sur la 1ère ligne de l'écran.

Ligne 3 – Attend l'appui d'une touche avant de quitter.

Appuyez sur une touche pour quitter. Ensuite appuyez sur la barre ESPACE pour revenir à l'éditeur.

Modifions notre première ligne comme suit :

```
Screen Open 0,640,200,16,Hires
```

Appuyez sur F1 pour exécuter le programme.

Nous avons désormais une résolution de 640x200, toujours en 16 couleurs, mais cette fois-ci en haute résolution (mot clé « Hires »). En haute définition, l'affichage est limité à 16 couleurs.

Appliquons un **entrelacement** à notre système vidéo. L'entrelacement consiste à afficher sur l'écran les lignes paires et impaires de manière alternée. Cela peut provoquer un scintillement de certaines parties de l'écran.

Modifiez la 1ère ligne :

```
Screen Open 0,640,400,16,Hires+Laced
```

Nous augmentons la taille de notre écran (200 à 400) et ajoutons le mot clé « Laced » à la fin de notre ligne. Cette résolution permet d'afficher une image de 640x400 pixels sur la totalité de l'écran. WinUAE, dans sa configuration de base, ne montre pas ce scintillement et le texte apparaît partiellement.

Palette de couleurs

Chaque couleur indexée est définie par 3 composantes RVB, chacune codée sur 4 bits. Voyons comment, avec Amos BASIC, il est possible de manipuler les couleurs. Créons un nouveau programme. Revenez à l'éditeur et appuyez sur les touches Shift+F9. Appuyez sur la touche « F » pour confirmer votre choix. Saisissez désormais le code suivant :

```
Screen Open 0,320,200,32,Lowres
Cls : Flash Off : Curs Off
X=0 : Y=0
For R=0 To 31
Ink R
Box X,Y To X+7,Y+7
Paint X+4,Y+4,1
X=X+8
If X=128 Then X=0 : Y=Y+8
Next R
Wait Key
```

Appuyez sur F1 pour afficher le résultat.

Voyons comment modifier une couleur. Appuyez sur une touche pour sortir du programme et **sur la touche ECHAP pour passer en « Mode direct »** Fig.3**.** Le Mode direct permet de saisir des commandes Amos en dehors du programme lui-même. Ces commandes ne seront pas mémorisées et n'affecteront pas le code du programme. Remarquez que le panneau du mode direct s'affiche en haute résolution, sur un écran en basse résolution. Comme indiqué plus haut, l'Amiga fonctionne en mode bitplane, donc le système peut superposer plusieurs écrans (8 en tout) de tailles et de résolutions différentes. Modifions une couleur en saisissant juste après le libellé « AMOS> », le code suivant. Validez par la touche ENTREE :

```
colour 0,$fff
```

La commande Basic « Colour » permet de modifier les composantes RVB d'une couleur.

La couleur indexée 0 devient blanche ($fff). La bordure étant remplie par la couleur 0 devient blanche également. Nous pouvons avec cette commande modifier chacune des couleurs de l'écran courant. Une erreur surviendra si vous indiquez un N° de couleur en dehors de la palette.

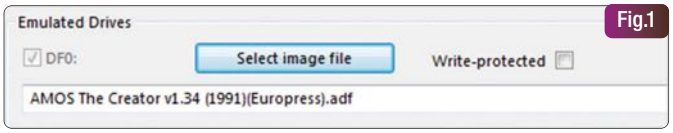


Fig.1

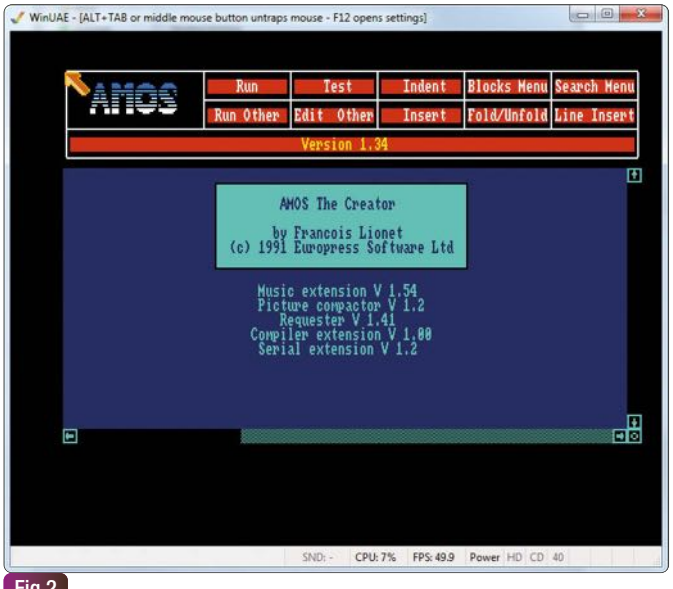


Fig.2

Il est possible de modifier plusieurs couleurs de la palette en une seule fois, grâce à la commande « Palette ». Par exemple pour modifier les 6 premières couleurs:

```
palette $000,$222,$444,$666,$888,$aaa
```

Utilisation du mode bitplane

Nous allons améliorer notre programme. Voir les briques de couleurs s'afficher une à une n'est pas très élégant. Nous allons ajouter un écran supplémentaire qui affichera un message de patience et masquera l'écran de nos couleurs. Lorsque nos couleurs seront totalement tracées, nous les afficherons. Améliorons notre programme comme suit :

```
Screen Open 0,320,200,32,Lowres
Screen Open 1,640,200,2,Hires
Palette $0,$FFF
Cls : Flash Off : Curs Off
Print "Installe les couleurs..."
Screen 0 : Flash Off : Curs Off
X=0 : Y=0
For R=0 To 31
Ink R
Box X,Y To X+7,Y+7
Paint X+4,Y+4,1
X=X+8
If X=128 Then X=0 : Y=Y+8
Next R
Screen To Front 0
Wait Key
```

Explication du programme

Ligne 2 – Ouverture de l'écran N°1, de résolution 640x200, en 2 couleurs et en haute résolution.

Ligne 3 – Définition de la palette de l'écran N°1. Par défaut, toutes les actions de textes, de couleurs et de tracés sont appliquées au dernier écran ouvert.

Ligne 5 – Ecrit le message de patience sur l'écran N°1.

Ligne 6 – On prend la main sur l'écran N°0. L'écran n'apparaît pas, mais les prochains tracés s'effectueront sur cet écran. On désactive le flashing de couleur et le curseur texte pour cet écran.

Ligne 15 – On bascule l'écran N°0 au premier plan. Ainsi, il devient visible.

Jack Vittriano s'installe sur l'Amiga

Pour démontrer des différences graphiques de chaque résolution, j'ai choisi l'image d'une des toiles du célèbre peintre écossais Jack Vittriano : **The Singing Butler** Fig.4.


Fig.4

The Singing Butler, Jack Vittriano, 1992.

Appuyez sur le touche F12 pour revenir au panneau de configuration de WinUAE. Nous allons insérer notre disquette de travail. Cliquez à nouveau sur le bouton « Select image file » et sélectionnez le fichier programmez.adf extrait de l'archive téléchargée. Cliquez ensuite sur « OK » en bas du panneau pour revenir à l'émulation. La disquette contient toutes les images dont nous avons besoin pour cette partie. Sous l'éditeur d'Amos Basic, créez un nouveau programme (Shift + F9) et saisissez le code suivant :

```
Screen Open 0,320,200,32,Lowres
Load Iff "df0:images/LORES200.IFF",0
Wait Key
```

Appuyer sur « F1 » pour exécuter le programme Fig.5.

Explication du programme

Ligne 1 – Ouverture de l'écran N°0 en 320x200, 32 couleurs et en basse résolution

Ligne 2 – Lecture et affichage de notre image depuis la disquette insérée dans le premier lecteur.

Sur Amiga, le format d'images le plus répandu est le format IFF (Interchange File Format) créé par Electronics Arts en 1985. Amos lit parfaitement ce format.

Il est important d'ouvrir un écran de même taille que l'image qui sera affichée. Dans le cas contraire, une erreur se produira. La palette s'ajustera en fonction de l'image chargée.

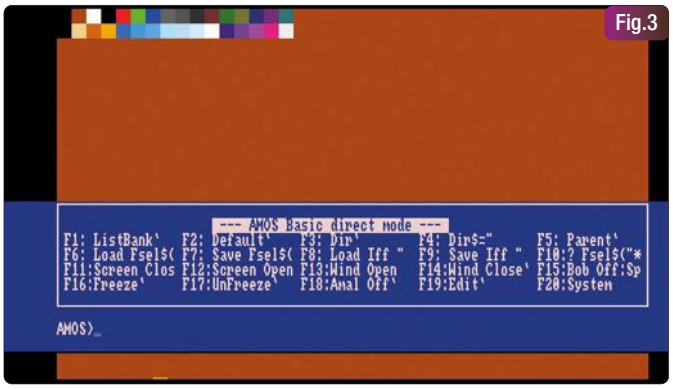


Fig.3

Ligne 3 – Attend l'appui d'une touche pour quitter.

L'image affichée se rapproche de l'image originale.

Mais elle manque de finesse.

320x200, 32 couleurs, basse résolution

Voyons comment elle se présente en haute résolution. Modifiez le programme comme suit :

```
Screen Open 0,640,200,16,Hires
Load Iff "df0:images/hires200.IFF",0
Wait Key
```

L'image affichée en 640x200 est plus fine. La palette est passée à 16 couleurs. Les pixels affichés ne sont plus carrés, mais rectangulaires. L'image met plus de temps à s'afficher.

Affichons la même image, mais en haute définition entrelacée :

```
Screen Open 0,640,400,16,Hires+Laced
Load Iff "df0:images/hires400.IFF",0
Wait Key
```

L'image reste en 16 couleurs mais les pixels redeviennent carrés, ce qui apporte une meilleure précision dans les détails.

Voyons maintenant l'affichage de cette image en 4096 couleurs (mode HAM) :

```
Screen Open 0,320,200,64,Lowres
Load Iff "df0:images/HAM.IFF",0
Wait Key
```

Vous remarquerez que l'écran ouvert définit un affichage en 64 couleurs.

Le mode HAM

Notre image, en basse résolution, affiche 4096 nuances de couleurs. Le tramage de couleurs présent dans les autres résolutions n'apparait plus (ou moins). Ce type d'image a été très peu utilisé dans les jeux vidéo car cela ralentit grandement la machine.

En regardant de plus près, l'image laisse apparaître des bavures de cou-


Fig.5

leurs. Ces bavures, ou « franges » sont dues aux couleurs de la palette. Pour afficher 4096 couleurs, le système se base sur les composantes RVB du pixel précédent, et les modifie pour afficher la couleur du pixel courant. Si la couleur est mal choisie, on voit apparaître ce type de frange. Cela arrive souvent dans les zones de dégradé.

Manipulation d'écrans.

Ce qui suit va vous permettre de comprendre le fonctionnement des écrans et comment, en quelques manipulations, nous pouvons programmer des effets graphiques fun. Vérifiez que la disquette de travail programmez.adf est toujours insérée.

Image : Effet lazer

Ce fût sans doute l'un des tous premiers effets d'apparition d'image, rencontré dans les démos Amiga. Il s'agit d'afficher progressivement une image à l'écran, ligne par ligne, en les projetant du bas vers le haut. Cela donne le sentiment d'un laser qui tracerait l'image. Créez un nouveau programme et saisissez le code suivant dans l'éditeur d'AMOS :

```
Screen Open 0,320,256,32,Lowres
Screen Open 1,320,256,32,Lowres
Cls : Flash Off : Curs Off
Screen 0 : Flash Off : Curs Off
Load Iff "images/dragon32.iff"
Screen 1 : Get Palette(0)
For SY=0 To 256 Step 2
For DY=256 To SY Step -2
Screen Copy 0,0,SY,320,SY+2 To 1,0,DY
Next DY
Next SY
Wait Key
```

Appuyez sur la touche « F1. Voici le résultat :

L'image se dessine ligne par ligne *L'image totalement affichée*

Image : balayage zoomé

Maintenant, nous allons voir comment l'on peut utiliser la superposition d'écran. Toujours avec notre image de dragon, nous allons programmer l'animation d'une barre horizontale, qui balaiera l'écran de haut en bas et donnera l'impression d'un grossissement de l'image lors de son passage. Créez un nouveau programme et saisissez ce qui suit :

```
Screen Open 0,320,256,32,Lowres
Screen Open 1,320,256,32,Lowres
Cls : Flash Off : Curs Off
Screen 0 : Flash Off : Curs Off
Load Iff "images/dragon32.iff"
Screen 1 : Get Palette(0)
Zoom 0,104,45,224,180 To 1,0,0,320,256
DY=56 : OY=0 : SENS=0
Screen To Front 1
Do
Wait 3
Screen Display 1,140,DY,320,83
Screen Offset 1,0,OY+24
If SENS=0 Then DY=DY+4 : OY=OY+4
If DY>204 Then SENS=1
If SENS=1 Then DY=DY-4 : OY=OY-4
If DY<49 Then SENS=0
A$=Inkey$
If A$<>"" Then End
Loop
```

Le résultat :

Animation : images par images

Voyons maintenant comment il est possible d'obtenir, toujours en manipulant plusieurs écrans, une animation images par images. Nous allons programmer une animation de 25 images qui se jouera à l'infini. L'idée est simple : on charge dans un écran en arrière plan, une image IFF contenant l'ensemble des étapes de notre animation. Sur l'écran placé au premier plan, on copie chacune des images, les unes après les autres Fig.6.

Fig.6

Image IFF contenant toutes les étapes de l'animation

Nous allons profiter de cet exemple pour introduire une nouvelle notion graphique bien connue sur Amiga : les Bobs. Un Bob est un élément graphique, similaire à un sprite, et qui peut-être afficher sur n'importe quel écran. Créez un nouveau programme et entrez le code suivant :

```
Screen Open 0,320,256,32,Lowres
Screen Open 1,320,256,32,Lowres
Cls : Flash Off : Curs Off
Screen 0 : Flash Off : Curs Off
Load Iff "images/sw.iff"
Screen 1 : Get Palette(0)
IX=0 : IY=0 : N=0
Locate 11,11 : Pen 2 : Print "Animation 25 frames!"
Screen 0
Do
Get Bob N+1,IX,IY To IX+80,IY+34
IX=IX+80
If IX>240
IX=0
IY=IY+34
End If
N=N+1
If N>24 Then N=0 : Exit
Loop
Screen 1 : Double Buffer
Do
Wait 3
Paste Bob 120,111,N+1
N=N+1
If N>24 Then N=0
A$=Inkey$
If A$<>"" Then Bob Clear : End
Loop
```

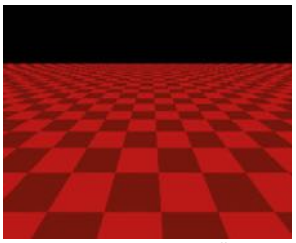
Résultat digne d'un Jedi !

Animation : cycle de couleur

L'une des techniques simples pour créer une animation est de passer par un cycle de couleurs. Ce procédé consiste à isoler quelques couleurs adjacentes dans la palette et d'alterner les valeurs de leurs composants RVB, en boucle. Nous allons voir comment mettre en place ce type d'animation en quelques lignes. Créez un nouveau programme et saisissez ce » qui suit :

```
Screen Open 0,320,256,32,Lowres
Curs Off : Flash Off : Hide
Load Iff "images/demo.iff"
C=16 : SENS=0 : C1=Colour(16) : C2=Colour(20)
Double Buffer
Do
Wait Vbl
If SENS=0
Colour C,C1
Colour C+4,C2
C=C+1
If C=20
C=16 : SENS=1
End If
End If
If SENS=1
Colour C,C2
Colour C+4,C1
C=C+1
If C=20
C=16 : SENS=0
End If
End If
A$=Inkey$
If A$<>"" Then End
Loop
```

Le résultat :



Le cycle de couleurs donne un effet de déplacement du damier vers la droite.

Sur la disquette Programmez!
Retrouvez sur la disquette l'ensemble des programmes AMOS présentés dans cet article. Pour charger un programme AMOS depuis l'éditeur, appuyez sur les touches Shift+F1.

L'Amiga dans tous ses états

Dans ce petit dossier, nous allons survoler 30 ans de machines Amiga, des historiques aux solutions passant par l'émulation. Ce tour d'horizon couvrant une grande période et de nombreux rebondissements. Alors plongeons directement dans le vif du sujet.

Jeffr3y
^Vital Motion ^AmigaVibes ^AmigaImpact
(jeffr3y.vm@gmail.com)

Mon Amie est un Gars !

Nous n'allons pas commencer par tergiverser, la machine est un Amiga, quid de ce dernier ?

Amiga First contact !

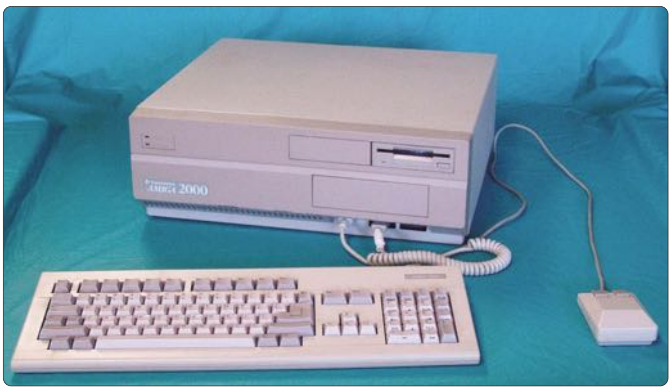
Revenons 30 ans en arrière, en 1985, c'est en cette année bénie, que l'Amiga 1000, première génération de la machine de Commodore va sortir le grand jeu dans la cour des micro-ordinateurs familiaux. Une présentation grandiose et tonitruante au Lincoln Center de New-York le 23 juillet 1985, avec des stars, comme Andy Warhol(1) aidé de Deborah Harry, plus connue comme la chanteuse de Blondie, va démontrer les capacités graphiques de l'Amiga par un travail d'artiste. Lors de cette cérémonie, il y a aussi la mise en avant des capacités de la machine via des démos devenues légendaires comme la Robo City et la mythique Boing Ball. L'Amiga 1000 possède un KickStart 1.0 (KS) sur disquette, rapidement éclipsé par une version 1.1, ainsi qu'un WorkBench 1.0 (WB).

Second Reality !!

La seconde génération d'Amiga arrive en 1987 avec des machines dédiées à la micro-informatique familiale, l'Amiga 500, et, pour les professionnels, l'Amiga 2000 ; ces deux machines embarquant Denise, Paula et Agnus. Ces deux machines ont marqué leur époque, avec des jeux et des démos ayant provoqué de nombreux achats compulsifs. Ainsi Shadow of the Beast (1989), démo technique avant de devenir un jeu. Du côté des démos, les RSI Megademo et autre Scoopex Mental Hangover firent tourner la tête de nombreux jeunes dans leur chambre et certains passèrent du côté demoscene pour créer leur propre groupe de demomakers. Toutefois ce ne fût que lors de la fin des années 1989-90.

Les Amiga Classics de troisième génération, c'est fantastique !

L'Amiga 3000 arrive en 1990 avec WB et KS 2.0, vite suivi par une version 2.04. Mais la véritable troisième génération arrive en 1992 par les 1200 et 600 du côté des familles et l'Amiga 4000 pour les pros. Ces derniers sont



L'Amiga 2000 ou le professionnel

L'Amiga 500, le plus vendu des micros de la gamme

fournis avec les KS et WB 3.0 à leur sortie, puis avec des KS et WB 3.1 après le rachat par Escom. Ces machines firent rêver une grande majorité des utilisateurs de 500 et 2000. Mais si les 1200 sont très complet (le 600 étant un 500 sans le pavé numérique, idéal pour un Amiga transportable). Le 4000 quant à lui fût utilisé par la Nasa jusqu'aux années 2010, sur TF1 pour ses jingles pub et ses incrustations vidéo, ainsi que pour la réalisation des effets spéciaux de Babylon 5 (série de SF des années 90) pour ne citer qu'eux. Mais après les années de faste, l'Amiga à l'instar de l'Atari ST, va connaître la chute avec l'élévation du PC dans les chaumières durant le milieu des années 1990. Les jeux des années 1992 sont Project-X et Pinball Dreams pour ne citer qu'eux, ainsi que Deluxe Paint, du 1 à 5, qui rayonna durant toute la durée de vie de l'Amiga (du 1000 au 4000). Je ne parlerai pas des 3000UX, 1500, 500+, HD et autres variantes qui ne se détachaient pas des Amiga décrits.

D'autres modèles plus atypiques et moins répandus ont été commercialisés comme le 2500, le Walker ou le CDTV qui était une platine multimédia avant l'heure, mise en échec par un positionnement marketing désastreux. Nous ne parlerons pas non plus de la CD32 qui est une console.

AmigaTrek, the Next Generation

Le renouveau arrivera durant le passage dans le nouveau millénaire, avec des solutions pour les accros de l'Amiga OS-like. Le système d'exploitation connut alors une nouvelle distribution avec l'OS4.0 et surtout avec une nouvelle architecture : le Power PC. Cela donna naissance à deux nouvelles machines à destination des utilisateurs : les AmigaOne et les Pegasos. Ces deux machines pouvant faire tourner les OS 4.1 d'Hyperion(2). A savoir que les Pegasos étaient initialement prévus pour faire tourner un autre système d'exploitation alternatif dérivant de l'AmigaOS : MorphOS(3).

Durant ces années noires de l'Amiga, un autre système d'exploitation

La Boing Ball, démo rebondissante créée par RJ Mical et Dale Luck, est une version améliorée de la Spinning Ball de Sam Dicker. Cette balle rebondissante fut présentée lors de démonstrations privées au CES de Las Vegas en 1984. Il est utile de savoir que les premiers Amiga de 1984 furent des Amiga Development System disposant des puces Agnus, Daphne et Portia sur circuits imprimés].

essaie de se créer une communauté, c'est Aros, avec pour leitmotiv d'avoir un déploiement sur des machines PC (x86 donc Intel et AMD) et qui est devenu Icaros(4).
Puis arrivent des cartes mères moins chères avec les Aone500 et SAM d'ACube Systems(5) et depuis 2011, il y a A-eon Computer(6) qui commercialise des Amiga avec les X1000 et X5000, des machines possédant une puce programmable Xena, composée de 2 cœurs à 500 MHz, directement liée au processeur. Back 2 the Roots !

Expansion is Revelation

De nouvelles extensions pour les Amiga Classics voient le jour depuis le milieu des années 2000, avec des cartes comme le HxC emul(7). Mais aussi par le biais de lecteurs USB simulant un lecteur de disquettes original avec la même vitesse de lecture comme le GoTek. Cela permet de redonner une seconde vie aux Classics toujours utilisés par des utilisateurs inconditionnels. Il y a aussi de nombreuses cartes : ACA indivision permettant l'expansion des capacités des Amiga Classics.

Emulation is Salvation

Dans ce paragraphe, nous parlerons des solutions Amiga par le biais de l'émulation qu'elle soit logicielle (software) ou matérielle (hardware).

Soft is Powerful, Soft is Beautiful

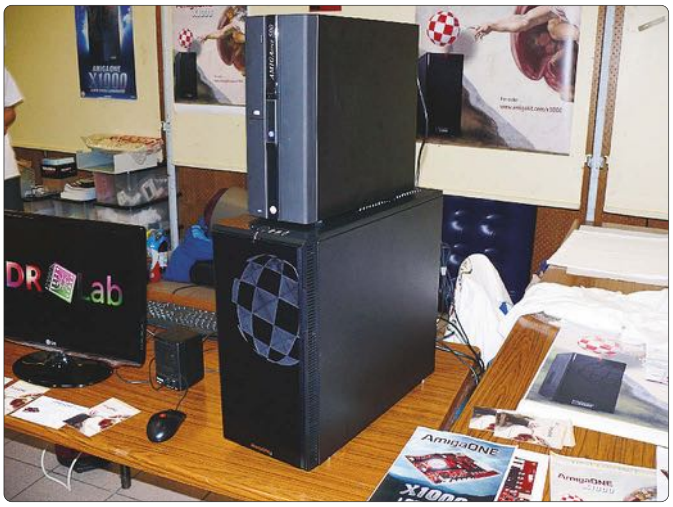
Plusieurs émulateurs Amiga voient le jour sur PC, mais un seul sort du lot après toutes ces années : l'incontournable WinUAE(8). C'est la solution la plus fiable et la plus compatible dans son intégralité avec les Amiga Classics. Pour info, il existe UAE4All qui est moins compatible, mais porté sur de nombreuses plateformes comme la PSP (PSPUAE(9)) ou la GP2x.
Dans le genre plus indépendant en plateforme, il y a la solution du Raspberry 2 Pi avec de l'Amiga émulé dans les distributions Retropie(10) ou RecalBox(11), mais cela nécessitera de bien les paramétrer avec les bons OS et KS. La difficulté venant surtout de cette dernière, car une fois bien configuré, le RPi2 s'avère une belle petite machine émulant l'Amiga avec brio. Si c'est vers les démos que vous lorgnez, préférez-lui les solutions hardware.
Enfin, il y a AmigaForever de Cloanto, solution tout-en-un comprenant WinUAE et intégrant Amikit, Aros et des mini-bases de données, des vidéos pour les éditions DVD. Un must-have pour ceux qui veulent reprendre simplement sur PC.

Hardwired by FPGA

Une autre alternative pour avoir des Amiga est la solution des FPGA initiée par le Minimig d'Acube Systems dans le début des années 2008. Cela permet une émulation du hardware de la machine. Ainsi les successeurs du Minimig sont les MiST et autres FPGA Arcade. La solution ayant le plus le vent en poupe est le MiST avec un développement de cores conséquents, et pas moins de deux dédiés à l'Amiga ; un pour émulation des Classics jusqu'à l'Amiga 1200 de base 68020EC, mais pas au-delà, ce sont les FPGA Arcade qui devraient aller plus loin. Il y a aussi un core qui permet l'émulation de l'affichage AGA. Bref, vous insérez une carte SD, installez le core Amiga et des ADF (image de diskette), HDF (image de disque dur) et même WHDLoad. Un must !

What's the Next Step, Demoscene, Zine, Community, ...

Sur les portails de la demoscene, comme Pouet(12), Scene.org(13), Demozoo(14) ou Demoscene.fr(15), la scene demo Amiga n'est plus aussi active depuis les années 2000 et des scènes comme celle du C64 apparaissent bien plus actives. Mais avec les 30 ans de l'Amiga cette année, elle fût plus



Un X1000 de chez A-eon et Aone500 lors de l'alchimie 2011


Un MiST et un Raspberry par dessus

productive en espérant que cela ne diminue pas l'année prochaine. De plus, une grosse partie de la communauté francophone Amiga se retrouve lors d'événements annuels : l'Alchimie(16) ou la micro-Alchimie, se déroulant dans la Drôme à Tain l'Hermitage dans une ambiance familiale et bon enfant, tout en ayant une coding party et des conférences autour de l'Amiga et de l'informatique alternative en général. On y côtoie aussi de la robotique avec l'association Caliban et le magazine Planète Robot, ainsi qu'un beau stand dédié au retro-gaming en général ;-)

So Far, so Good, so What !

On peut dire : l'Amiga est mort, vive l'Amiga et il continue avec une communauté de développeurs, de sceners et d'applications restreintes. Les temps ne sont plus à des guéguerres Amiga-Atari ST ou pire, OS4-MorphOS-Aros. Alors venez tous à la prochaine Alchimie en 2017 à Tain l'Hermitage, rejoignez les TripleA(17) ou d'autres associations alternatives pour ressentir encore les joies des OS multitâches Amiga. N'hésitez pas à contacter des passionnés de ces micros, ils vous montreront comment exploiter les Amiga Classics comme les NG, voir les émuler, car l'Amiga Spirit, c'est le plaisir et la convivialité avant tout.

Code source d'une démo en Hollywood 6.0 (une partie du scrolltext) :

Le code source sur le site de programmez.com
Retrouvez tous les références du dossier sur programmez.com ⊞

Atari ST, la superstar

L'Atari ST a fêté ses 30 ans cette année ! Machine emblématique des années 80 inventée par le consortium de la famille Tramiel, il sera équipé de son propre système d'exploitation avec un bureau intégré dénommé GEM ; il eu fort à faire avec la concurrence de Commodore et de son Amiga né une année auparavant. Le monde des machines multimédia assemblées avec des composants Motorola et aux architectures décuplées pour l'époque en 16/32 bits permettaient de dépasser en capacité toutes les machines de l'époque telles que l'Amstrad CPC.

Frédéric Sagez
(fsagez@gmail.com)

De cette confrontation entre machines naîtra une « guéguerre » sur le thème de qui a la meilleure bécane entre les possesseurs de l'Amiga et de l'Atari ; même le magazine TILT sortira d'ailleurs un spécial hors série à ce sujet. Les performances graphiques de l'Amiga ne sont pas à remettre en cause car cette machine a des composants dédiés pour, mais *l'Atari fera plutôt le bonheur de beaucoup de musiciens de styles différents comme le célèbre compositeur Jean Michel Jarre ou le groupe Kassav*. L'arrêt de la production de ces machines va s'arrêter brutalement, Commodore va faire faillite en 1994 et Atari Corporation suivra le même chemin quelques années plus tard après quelques rachats malheureux. Et aujourd'hui, que sont devenues nos mythiques machines ?

Le Hardware

Au 21ème siècle, le hardware, vieillissant mal avec le temps, va amener la communauté à trouver un moyen pour changer le lecteur de disquettes qui, lui aussi avec le temps, tombe en panne (sans parler de l'abandon de la production des supports de disquettes double face), ainsi que les gros disques durs.
Dans le cadre du « Software preservation » qui est le principe de restauration, transfert et archivage de disquettes, beaucoup de projet hard et soft verront le jour. L'avantage de l'Atari ST face à ses concurrents de l'époque est de pouvoir lire et écrire sur des disquettes utilisées par un PC. Et l'un des projets les plus aboutis en matière de récupération de données est le projet **Kryoflux** de SPS **:** c'est un dispositif qui permet de relier un lecteur de disquettes traditionnel avec une connexion USB à un PC ; avec son logiciel de lecture des faces de la disquette, il est reconnu pour sa fiabilité et sa précision nécessaire pour acquérir les informations de tout l'ensemble des données de la disquette.

En fait, le but est de créer des fichiers images complets à partir des disquettes d'origine qui soient dans un format relu par différents types de supports comme par exemple les émulateurs qui utilisent des fichiers aux formats ST ou MSA par défaut. Mais l'utilisation d'un lecteur de disquettes de remplacement se fait attendre jusqu'à ce que Jean-*François Del Nero* crée le projet **HxC Floppy Emulator** en 2007 qui va permettre d'utiliser un émulateur de disquettes Hard sur de vraies machines. Il se met tout simplement et assez facilement sur l'emplacement de l'ancien lecteur de disquettes de la machine et a la caractéristique principale de lire des fichiers images via une carte SD. On peut maintenant créer des sauvegardes de ses disquettes en faisant des images disque et ensuite les utiliser sur une vraie machine. Le disque dur de l'époque est un périphérique doté du protocole de transfert ACSI très lent d'accès et limité en quotas disque et partition. Différents projets ont palliés ce type de contraintes : le **SatanDisk** puis l'**UltraSatan** sera un vrai « couteau Suisse » pour remplacer le disque dur en mode natif sur un Atari ST. Créé par le polonais *Miroslav NOHAJ* surnommé « Jookie », il s'agit d'une sorte de petit boîtier incontournable qui se connecte via le port ACSI, et, alimenté par une carte SD, il émule un disque dur avec des partitions. Aujourd'hui, avec ou sans Rasberry Pi, le **CosmosEx** du créateur du **SatanDisk** va au-delà du lecteur de disque dur via une SD Card standard car si on lui rajoute un Rasberry Pi, il permettra de connecter et d'utiliser des périphériques USB et de se connecter à un lecteur réseau pour échanger des fichiers. Les petits plus : téléchargement d'images de disquettes via Internet et utilisation de périphériques tels que le remplacement du clavier/souris via la connectique USB.

Les Émulateurs

Loin des premiers émulateurs PC/DOS tel que **PacifiST** de Frédéric Guidouin qui ont marqué la fin des années 90, aujourd'hui la gamme s'est bien étoffée avec deux émulateurs reconnus dans le monde des utilisateurs d'émulateurs Atari ST : **Steem SSE** et **Hatari**.
Steem SSE est la suite logique de la version créée à la base par les frères Hayward, c'est un fork appelé *Steven Seagal Edition* d'où l'acronyme SSE utilisé par son nouveau contributeur surnommé *Atari Steven* qui a fait une mise à jour complète de l'émulateur qui a été développé à l'origine avec une structure de code en C++. Le logiciel tourne principalement sous Windows mais il existe une version Unix appelée **XSteem** toujours en cours de portage.
Hatari est le deuxième émulateur ultime pour PC, Mac et Linux ! Hormis une interface de configuration un peu archaïque, l'émulateur permet d'émuler la gamme des Atari STF, STE et est ouvert aussi à la gamme 68030 : le Falcon. Comparativement à Steem SSE, c'est un projet Open Source développé en GNU-C, créé par *Thomas Huth* et maintenu par plusieurs contributeurs.
Il existe un émulateur spécifique pour visualiser les premières démos apparues sur Atari ST et entretenu par son créateur *Arnaud Carré* : **Saint**. Il fonctionne sous Windows avec DirectX et son grand intérêt comparativement à ses autres confrères est d'utiliser les images disques « non patchées » et d'origine.
A noter que pour faire fonctionner les émulateurs, il vous faut obligatoirement le **TOS** (surnommé le « The Operating System ») qui est le système d'exploitation de la machine que vous

devez obligatoirement posséder pour l'utiliser avec un émulateur. Il existe différentes versions suivant la machine que vous voulez émuler : une version 1.02 pour simuler l'utilisation d'un simple Atari STF jusqu'à la version 4.92 pour un Falcon et aussi suivant la langue utilisée par la machine. Il existe une autre solution qui est le projet open source **EmuTOS,** il permet de fournir un TOS compatible et gratuit avec l'avantage d'être redistribuable à volonté.

Les Jeux

Beaucoup de jeux ont vu le jour pour le début du 21ème siècle et ce, sur les différentes plateformes qu'offre Atari. (STE, Lynx, Falcon, XL et Jaguar). Le groupe de démo français **Paradize** va produire beaucoup de jeux de réflexion comme **Pooz** ou **Kolmik**. Issus de portage de jeux déjà existants sur des plateformes différentes, des jeux de type « projets » développés par de généreux contributeurs voient le jour comme le jeu **PicrossST** initié par notre ami belge *Shadow272* ; avec la Shadow Team ils vont faire d'un jeu une trilogie : **Saboteur III**.
Le rétro gaming va entrer en force sur l'Atari STE avec l'éditeur de jeux classifiés « vintage » **Retro Gamer CD** (« faux-retro games for modern platforms ») en association avec le groupe de démo **NoExtra** avec le jeu d'arcade **r0x** qui sera principalement développé par *Nicolas Flandin* qui est le musicien du groupe. Rappelons le principe du jeu : avec votre vaisseau spatial vous devez éviter les météorites de différents gabarits tout en prenant des bonus. D'ailleurs le jeu terminera premier de sa catégorie à la **Outline** party en Hollande en 2009. Enfin deux jeux de courses de voitures arrivent en pôle position : **Racer** de *Laurent Sallafranque* du groupe de démo français **Dune** qui propose différents modes de jeu comme le jeu **Out Run** de Sega qui fonctionne uniquement sur Falcon. **Anarcho Ride** de l'Autrichien


Le jeu r0x sur Atari STE

Thomas Ilg, déjà auteur du jeu **Laser Ball** vous propose un système de jeu inversé : percuter un maximum de voiture sans limite de temps sur Atari STE.

Coding Party

C'est un rassemblement qui permet aux personnes de participer à des concours de différentes catégories (code, graphisme et musique) et de voir aussi des groupes s'affronter pour gagner des concours.
Les démos parties ou vous verrez principalement des ataristes se font surtout en Hollande avec la **Outline** ou la **ST News International Christmas Coding Convention** qui revient cette année pour une ultime session avec une rencontre entre les anciens et les nouveaux demosceners. La **SommarHack** en Suède se situe dans un endroit isolé où les barbecues-parties et la franche rigolade en font une CP proche des demosceners. La **SillyVenture** en Pologne permet de voir de belles productions que ce soit sur le ST, FALCON ou sur des consoles comme l'Atari XL.

Le groupe NoExtra encore premier à la VIP2015

Les Démos

Les démos sont des démonstrations graphiques, techniques et musicales montrant la capacité de la machine, et l'année 2003 aura été un cap pour la Démoscène car pendant la démo partie **Error In Line** à Dresden, le groupe suédois **Dead Hackers Society** sort la démo **Moving Into Darkness** sur Atari STE qui est une bombe à l'époque. Elle ne contient que des effets réalisés habituellement sur PC (effets composés principalement avec des objets 3D contenant des textures animées avec reflets et luminosité en « temps réel »)
Habitué à dépasser les capacités de la machine outrageusement, le groupe **Dead Hackers Society** mené par *Anders Eriksson* (Evil) sortira en 2008 la démo **More Or Less Zero**, démonstration contenant beaucoup d'effets mais ayant la particularité unique de fonctionner en mode « FULLSCREEN ». Pour les non-puristes, le mode overscan, c'est-à-dire plus de border haut-gauche-droit-bas à l'écran.
Les démomakers français ne sont pas en reste et en 2011, la démo **STreet Art** du groupe **BlaBla** va aussi dépasser toutes les attentes au niveau des capacités visuelles de la machine à la **SillyVenture.** Aidé par le graphiste de street art *Acet*, *Fabrice Catteau* (Cyg) nous envoie une démonstration de haut niveau avec beaucoup d'effets contenant une multitude de couleurs diverses et variées dépassant les 4096 couleurs à l'affichage. Deux écoles s'affrontent entre les démos dites « old-school » comme le groupe **NoExtra** et les démos dites « new-school » comme le groupe français **Live!**
Groupe qui sortira à la **SillyVenture** en 2012 la démonstration **Muda** avec une musique tonitruante composée par TomChi, *Nicolas Flandin* étant le musicien du groupe **NoExtra** et grand expert logiciel maxYMiser de *Gwem*.
Mais la démonstration « ultime » appelée couramment « colorfull effect » reviendra au groupe **Dead Hackers Society** avec leur démo **Sea of Colour** qui est sortie cette année à la Sommarhack. Cette démo est titanesque de part les effets en tous genres et de la musique qui l'accompagne ; cette démo ne fonctionne d'ailleurs que sur un disque dur, type **Ultra Satan**. Je vous invite à découvrir et à redécouvrir toutes les productions depuis le début du ST sur le site de l'allemand *Stefan Benz* http://no-fragments.atari.org/.

Musique, le son chiptune

Chaque sceners de la scène démo a débuté un jour sur un Amiga ou sur un Atari ST en utilisant un logiciel tel que **Protracker**, un éditeur de pattern qui joue des samples avec des effets sur 4 voies. Et avec le temps ils ont évolué avec un style de musique plus prêt du composant du ST qui est le YM2149 sound chip de Yamaha.
L'intérêt du soundchip 3 voies du ST est qu'il consomme moins de CPU pour un processeur à 8 Mhz en utilisant des sons de synthèses comme samples. Pour mémoire, le célèbre *Jochen Hippel* (Mad Max) du groupe **The Exceptions** a créé un format de musique dit soundchip car il a beaucoup travaillé sur des musiques de jeux tels que **The Great Giana Sisters**, **Enchanted Land** avec des caractéristiques hors du commun pour un jeu sur ST ou encore **Wings of Death** de l'éditeur de jeux allemand **Thalion**. Par la suite il a beaucoup inspiré des sceners tels que *Frank Denis* (Jedi) qui invita son propre format de musique avec son logiciel **Megatizer** ou *Mathieu Stempell* (Dma Sc) du groupe **Sector One**. Un des pionniers de la chipmusic est *Kalle Jonsson* (Dubmood). C'est un compositeur et producteur Suédois provenant de Demoscene Amiga et Atari ; il a sorti dernièrement le vidéoclip **Solitude** où il intègre intelligemment du son provenant de l'Atari comme il le fait à chaque représentation sur scène avec différents DJ de la scène 8bits.
Et pour composer de la musique dite chipmusic via un Atari ST, il vous faut l'outil ultime

Suite page 50

Drupal 8 est (enfin) là !

Le parcours fut long et difficile, avec des reports et des retards. La phase bêta a duré 12 mois, ce qui est très long, puis les versions RC (juste avant la version finale) se sont succédées sur plusieurs semaines. Mais cela ne signifie pas que D7 soit morte dès maintenant. Son support actif est assuré au moins pour 3 ans.
Par contre, la version 6 n'est plus maintenue, excepté pour la sécurité. D'ores et déjà, les équipes Drupal travaillent sur la version 8.1.x dont le processus de développement démarrera réellement en janvier 2016 pour une disponibilité au printemps (courant mars).
D8 propose beaucoup de nouveautés et d'évolutions : nouveaux champs, éditeur de texte revu, fonctions de responsive design (natif à D8), multilangue intégrée et améliorée, module de configuration, utilisation des couches de Symfony, etc.

La rédaction.

Les points majeurs à retenir

Quatre ans après le début de son développement, 3290 contributeurs et 1228 entreprises plus tard, la version stable de Drupal 8 est sortie mondialement le 19 novembre dernier. Très attendue par la communauté, cette nouvelle version du CMS Open Source offre plus de 200 nouvelles fonctionnalités et améliorations. Jetons donc un coup d'œil sur les nouveautés et changements majeurs apportés par Drupal 8.

Chengbo et Morgane,
Feel & Clic
contact@feelandclic.com

Mobile-first (Responsive)

Première évolution notable, une approche « mobile-first » a été adoptée pour cette 8ème version : les principaux thèmes proposés par le CMS sont intégralement responsive et le back office a été repensé (barre d'outils mobile-friendly, tables responsives, bouton « retour au site » sur les pages backend) afin de permettre l'administration de son site depuis des supports mobiles. Le marquage des pages en HTML5, complété par des outils de saisie mobile, permet d'offrir aux internautes en situation de mobilité une expérience utilisateur fluide et ergonomique. Fig.1.

Architecture

- Serveur : Apache 2.0 et +, Nginx 1.1 et +
- Langage : PHP 5.5.9 et +
- Base de données : MySQL 5.5.3 et +, MariaDB 5.5.20 et +, Percona Server 5.5.8 et +, PostgreSQL 9.1.2 et +, SQLite 3.6.8 et +

Des nouveaux modules entrés dans le coeur de Drupal 8

Il y a de nombreux modules qui ont été intégrés dans cette nouvelle version de Drupal. Voici ci-dessous certains qui sont largement utilisés :

- Views, permet de créer, gérer et afficher des listes de contenu. Cette intégration apporte de nouvelles fonctionnalités et permet entre autres la personnalisation de divers éléments (slideshows, galeries, contenu des sidebars, listes des écrans d'administration...) sans avoir recours à du code, le clonage des vues, la modification en série des contenus...
- Multilingue, permet la traduction complète de tous les éléments du site.
- Layouts, permet de créer des affichages différents pour les contenus.
- Web Services, permet d'exposer vos données à un système externe en créant des Web services.

Les champs

De nouveaux champs ci-dessous souvent utilisés sont entrés dans le cœur également, ce qui facilite la tâche de la configuration du site : Date, Email, Link, Reference, Telephone.

Multilangue

Dès l'installation de Drupal 8, il vous sera demandé de sélectionner votre langue de travail

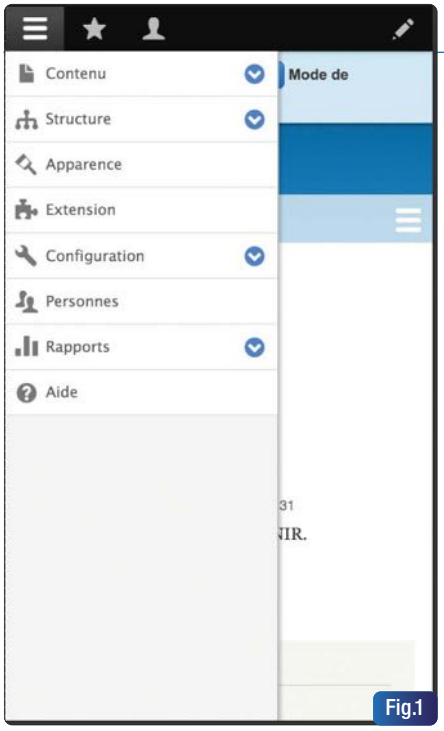


Fig.1

parmi plus d'une centaine fournies par la communauté sans avoir besoin de télécharger un fichier de langue à placer dans un répertoire précis comme sur Drupal 7. De nombreux modules "contribs" ont été intégrés dans le cœur (par exemple i18n, Administration Language...), qui sont réorganisés en 4 modules. Cela permet une traduction complète (l'interface, les configurations, les contenus, etc.) sans avoir besoin d'installer de modules complémentaires.

Les blocs

Le système de gestion des blocs est bien plus évolué dans cette nouvelle version. Ce ne sont que des bonnes nouvelles pour les site-builders. Parmi les nouveautés, il y en a 3 qui nous apparaissent particulièrement importantes :

- La possibilité de positionner un même bloc dans plusieurs régions. C'est ce que Multi-Block propose sur Drupal 7.
- Cette nouvelle version inclut la fonctionnalité du module Bean. Cela veut dire que le bloc devient un entité qui permet de créer des types de blocs et personnaliser les champs pour chaque type de bloc.
- Il est possible de configurer le niveau de menu à afficher pour un bloc de menus. Cela intègre la fonctionnalité du module "menu block".

Gestion de contenu

La création et l'édition de contenu a été globalement facilitée grâce à l'éditeur de texte WYSIWYG intégré CKeditor. On compte donc parmi les nouvelles fonctionnalités : une interface ergonomique répartie en deux colonnes (l'éditeur de contenu à gauche, les informations d'administration à droite), la possibilité de modifier le contenu directement (édition rapide) depuis les pages sans ouvrir le back office, une fonction d'aperçu de la page côté visiteur, une gestion des brouillons améliorée et une sécurité renforcée. Tous les formulaires des entités peuvent donc désormais être personnalisés davantage, tant au niveau de leur apparence que des données qu'ils contiennent. Cela permet surtout de configurer l'affichage du formulaire simplement avec un drag&drop que ce soit le formulaire d'inscription d'un utilisateur ou d'ajout de contenu.

Font-end / Intégration

TWIG : le système de template PHPTemplate a été abandonné au profit de TWIG. Ce dernier a plus d'avantages :

- TWIG est plus sécurisé car il n'est plus possible d'exécuter de code PHP dans ce nouveau type de template.
- La syntaxe de TWIG est simple et plus lisible que l'ancien fichier de template.
- TWIG propose également un système de debuggage. Par exemple, la fonction dump est disponible pour imprimer une variable.
- Le fichier de template est réutilisable car il est possible d'inclure un autre template à l'intérieur du fichier.
- TWIG est bien documenté.
- jQuery2 et Backbone.js : Drupal 6 avait adopté jQuery, Drupal 8 continue en intégrant la dernière version jQuery2 et y ajoute Backbone.js (et de nombreuses autres librairies JS). Les développeurs front-end vont pouvoir nativement créer des interfaces riches utilisant le Javascript.

Programmation Orientée Objet

Dans cette nouvelle version, Drupal quitte son île et s'oriente vers la POO (Programmation Orientée Objet) en intégrant dans son coeur des composants du framework français Symfony comme par exemple Routing, HttpFoundation, HttpKernel, DependencyInjection... On utilise donc maintenant des classes, des interfaces, l'héritage, l'injection de dépendances, l'autoloading, les annotations PHP, etc. Le but poursuivi ici est de rendre Drupal plus flexible et polyvalent ainsi que d'attirer des développeurs venus d'autres horizons qui connaissent déjà les grands principes et pourront donc s'adapter rapidement avec cette standardisation. Pour un développeur Symfony, Drupal 8 sera donc plus rapide à prendre en main. En parallèle, les intégrations des composants extérieurs seront plus faciles, cela permettra de gagner du temps pour se concentrer sur les développements spécifiques.

Cache et performance

Entity cache module est intégré dans le coeur. Le cache et l'agrégation des CSS / Javascript sont désormais activés par défaut dès l'installation du profil standard de Drupal. Nous verrons si cela améliorera la situation actuelle car de nombreux sites Drupal 7 sont en production sans le cache activé. En parallèle, le cache pour les utilisateurs authentifiés est également disponible (et activé dès l'installation), cela permet d'améliorer les performances pour tous les utilisateurs. Du côté front-end, la gestion de Javascript est bien améliorée, le Javascript est désormais chargé dans le footer, et jQuery est chargé uniquement sur les pages qui en ont besoin. Tout cela permet de rendre l'affichage des pages plus rapide.

Gestion des configurations

Le déploiement de nouvelles configurations d'une version de développement vers une version de pré-production ou de production était toujours compliqué sur Drupal 7 même avec des solutions comme le module Features ou les fonctions hook_update_N()...

Le système de configuration de Drupal 8 laisse le choix à l'utilisateur quant à l'emplacement et la manière de stocker les fichiers de configuration. Vous pouvez ainsi stocker vos paramétrages au sein même de votre base de données ou les exporter et les importer sous la forme de fichiers YAML. Vous avez également la possibilité d'importer/exporter en masse des éléments individuels (par exemple, le nom du site, les blocs, les vues, les rôles utilisateurs, les types de contenus, les entités, etc). D'un point de vue technique, cela facilite grandement les mises en production et les déploiements.

Drupal 8 introduit le concept de UUIDs (identifiants uniques universels). Chaque contenu a un identifiant unique (et même d'un site à l'autre), par exemple : 744bdca2-b748-448a-af00-de0ad132e931. Celui-ci est indépendant de l'ID de l'auteur. Il permet de vérifier si un contenu existe déjà ou pas sur le site à migrer même si il n'a pas le même node ID (nid).

Conclusion

Si ces nouvelles fonctionnalités apporteront sans aucun doute plus de confort aux clients et utilisateurs finaux, la majeure partie des évolutions se situe sous l'angle technique. Bien plus que de simples évolutions, certaines de ces fonctionnalités constituent une véritable révolution qui permettront sans aucun doute d'étendre le champ des possibles du CMS et d'attirer de nouvelles compétences au sein de la communauté. Avec des références prestigieuses à son actif telles que La Maison Blanche, gouvernement.fr, Voyages-SNCF, BNP Paribas, ou encore PRR, Drupal n'est pas prêt d'arrêter de sitôt sa croissance.

La révolution du cache

Ce n'est un secret pour personne, l'un des principaux inconvénients de la programmation objet est sa consommation de ressources serveur. Ainsi, lorsque Drupal a fait le choix de passer d'un modèle tout procédural au modèle objet, il était évident qu'il serait nécessaire de trouver une façon d'optimiser un cœur déjà souvent décrié comme étant trop lourd. Comme il n'était pas concevable de transiger sur la souplesse de l'outil en réduisant son spectre fonctionnel, la communauté, en parallèle du travail d'optimisation normal du cycle de développement, s'est intéressée aux problématiques de cache. Le système qui est désormais nativement intégré au cœur de Drupal 8 pourrait bien être répliqué dans tous les autres CMS tant il est malin et efficace.

Edouard Cunibil (DuaelFr)
Happyculture.coop

Constat

Une fois le code optimisé au mieux, il ne reste plus beaucoup de solutions pour réduire l'empreinte d'un logiciel si ce n'est lui éviter de refaire plusieurs fois les mêmes calculs en plaçant leur résultat en mémoire. La méthode est connue depuis toujours ; la difficulté dans le cas d'un outil comme Drupal est de proposer des méthodes pour éviter aux développeurs d'avoir à constamment réinventer la roue.

De plus, lorsque l'on envisage tous les différents types de sites pouvant être conçus avec Drupal, on constate que le système doit répondre à des problématiques très variées.

Drupal 7 dispose déjà d'un module permettant de mettre en cache les pages rendues pour les utilisateurs anonymes. La plupart des sites éditoriaux se contentent très bien de ce seul système mais tous les sites nécessitant une interaction poussée des utilisateurs ne peuvent pas en profiter. Pour ces sites, Drupal 7 contient également dans son API un système de cache de bas niveau permettant de stocker des données en mémoire vive, dans la base, ou dans un cache backend comme memcache ou Redis. L'utilisation de ces systèmes étant à la discrétion du développeur, il n'est pas rare de trouver des sites n'en exploitant pas du tout les capacités.

Démarche

En partant du constat que Drupal 7 ne permettait pas à toutes les typologies de sites de disposer d'un système de cache performant, la communauté s'est demandée comment il serait possible de rendre les technologies utilisées par les plus gros projets accessibles à tous.

En effet, dans le cadre de sites très dynamiques il n'est pas aisé d'aller chercher une couche supplémentaire de cache. Il reste deux problématiques majeures à résoudre qui sont la mise en cache des parties de contenu très spécifique à l'utilisateur (panier de commandes, encart « mon compte », etc.) ou celles variant très souvent dans le temps (derniers commentaires, flux d'actualités entrant, etc.).

Dans la plupart des systèmes, dont Drupal 7, une page étant composée de tous ces éléments ne peut pas être mise en cache sous peine d'être soit invalidée constamment, soit de servir des informations à l'attention d'un utilisateur ou à un autre.

Pour répondre à ces problématiques, des solutions existent déjà mais sont difficiles à mettre en œuvre ce qui explique qu'on ne les retrouve que sur des projets de grande envergure. Qu'il s'agisse d'utiliser des ESI ou un chargement des données via AjaX, cela demande un travail supplémentaire non négligeable.

Solution

Et si Drupal 8 intégrait un système de cache hautement inspiré des ESI pour construire ses pages ? Et si ce système permettait facilement de passer d'un mode de rendu par le serveur à un mode de rendu par le client à la façon de BigPipe ?

Ce sont les deux questions que se sont posées la communauté et qui ont aujourd'hui mené Drupal 8 à ce système novateur, le module Dynamic Page Cache, présent dans le cœur et activé par défaut. Son principe est assez simple : les pages de Drupal étant construites sous forme de blocs de HTML imbriqués, chacun de ces éléments peuvent désormais disposer de métadonnées permettant de décrire comment ils doivent être mis en cache et comment ils doivent être invalidés. Ces métadonnées se propagent des éléments enfants vers les parents, jusqu'à définir la page elle-même. Elles sont de quatre types pour couvrir tous les besoins :

Les clefs de cache (cache keys)

Ce sont un ensemble de valeurs arbitraires permettant d'identifier l'élément dans le cache. Elles sont concaténées pour créer l'identifiant de cache qui pourra être utilisé, de façon complète ou partielle (voir les contextes plus bas), pour l'invalidation.

```
// Exemple de clefs de cache pour une liste d'actualités en page d'accueil.
$build['#cache']['keys'] = ['news_list', 'home'];
```

L'âge maximal (max-age)

La plus simple et la moins utilisée des métadonnées permet de définir à partir de combien de temps le cache de l'élément sera invalidé de toute façon. Généralement, ce type d'information est utilisé lorsque l'on veut faire une remontée d'éléments de façon aléatoire dans une page, sans que soient affichés toujours les mêmes mais sans consommer trop de ressources serveur. Dans la grande majorité des cas, aucun âge maximal ne sera indiqué afin de conserver le cache aussi longtemps que nécessaire.

```
// Cache désactivé.
$build['#cache']['max-age'] = 0;
// Cache d'une durée maximale de 5 minutes.
$build['#cache']['max-age'] = 180;
// Cache permanent. Jusqu'à invalidation manuelle ou via un autre critère.
// Valeur par défaut.
$build['#cache']['max-age'] = Cache::PERMANENT;
```

Les tags de cache (cache tags)

Ils sont utilisés pour définir les dépendances entre le cache et des objets arbitraires. Toutes les entités de contenu et de configuration sont des dépendances valides mais il est également possible de créer ses propres

dépendances en cas de besoin. Si jamais l'un de ces objets est modifié, tous les caches qui lui sont rattachés sont immédiatement invalidés. Certains tags génériques permettent d'invalider automatiquement l'élément lors d'un événement particulier comme la création d'un nouveau contenu.
Par exemple, si la page d'accueil d'un site contient la liste des trois dernières actualités, cette dernière sera marquée par le tag spécifique de chacune des trois actualités affichées (node:ID) et par le tag générique de création d'un nouveau nœud (node_list). De cette manière, si l'une des trois actualités est modifiée ou si une nouvelle actualité est créée, la page (par propagation) et la liste seront invalidées et seront donc reconstruites au prochain affichage.

```
// Tag invalidant la liste lors de la création ou la suppression
// d'un nœud quelconque.
$build['#cache']['tags'] = ['node_list'];
foreach ($nodes as $node) {
  // Tag invalidant la liste lors de la modification ou la suppression
  // du nœud identifié.
  $build['#cache']['tags'][] = 'node:' . $node->id();
}
```

Les contextes de cache (cache contexts)

Ces derniers permettent de définir les diverses variantes du contenu. Ainsi, chaque variante est mise en cache séparément et servie aux utilisateurs qui correspondent aux critères. Il existe un certain nombre de contextes par défaut (langue, permissions, headers HTTP, cookies, route, etc.) et il est possible d'en ajouter au besoin en implémentant un service portant le tag « cache.context ».
Les contextes de cache sont traduits au moment de la génération du contenu et ajoutés à la fin des clefs de cache afin de générer l'identifiant de la variation. Il est donc possible d'invalider toutes les variations d'un élément en se basant uniquement sur ses clefs de cache, ou d'invalider une variation précisée en utilisant l'identifiant complet.
Par exemple, la liste des trois dernières actualités de notre site pourrait dépendre de la langue et des permissions. Cela permettrait à un utilisateur de voir les trois dernières actualités correspondant à ses droits et à la langue dans laquelle il parcourt le site, sans avoir à considérer que cette liste n'est jamais cachée.

```
// Crée des variantes en fonction des permissions des utilisateurs
// et de la langue de la page.
$build['#cache']['contexts'] = ['user.permissions', 'languages'];
```

ESI, BigPipe et CDN

Profitant de la puissance et la souplesse de la programmation orientée objet, ce système de cache peut assez facilement être détourné pour transformer chaque élément de la page par un tag ESI qui sera exploité par un reverse proxy ou alors, comme dit précédemment, en bloc chargé dynamiquement en Javascript à la manière de BigPipe. Cette dernière méthode s'appuie actuellement sur un module contribué qui pourrait bien intégrer directement le cœur dans la version 8.1 ou 8.2 de Drupal, selon l'adhésion rencontrée par la communauté.
Pour couronner le tout, Drupal 8 exploite les métadonnées de cache propagées jusqu'au niveau global de la page pour définir les entêtes HTTP renvoyés et ainsi permettre d'optimiser le cache du navigateur, du reverse proxy ou encore d'un CDN sans configuration supplémentaire ou presque !
En conclusion, à fonctionnalités équivalentes, les sites construits suivant ce principe promettent un temps de rendu bien inférieur à ce que l'on parvenait à faire auparavant, pour peu que les caches aient déjà été générés... ⊞

Drupal 8 multilingue

4 modules du cœur de Drupal remplacent désormais une vingtaine de modules de la communauté

Si vous avez déjà réalisé un projet multilingue avec Drupal 7, vous savez que cela requiert l'installation d'une multitude de modules communautaires. Drupal 8 simplifie grandement la tâche en incluant au cœur du système tout ce qui est requis pour créer des sites en plusieurs langues, ou même simplement des sites dans une langue autre que l'anglais. Pour ce faire, la nouvelle version est livrée avec quatre modules répondant à l'ensemble des besoins que l'on rencontre dans un projet multilingue.

Maxime Turcotte
Consultant en logiciel libre
Savoir-faire Linux

La langue au premier plan

Le premier module, language, est la base du support multilingue de Drupal 8. Bien que grandement améliorées, ses fonctionnalités faisaient partie du module locale de Drupal 7. Lorsqu'on installe Drupal 8, on remarque l'importance donnée à la langue dès la première question. Avant même de débuter, on nous demande en effet de choisir la langue dans laquelle on désire poursuivre. En plus de la détection basée sur la langue de votre navigateur et des 94 choix disponibles, les fichiers de traductions sont automatiquement téléchargés et le reste du processus se fait alors dans la langue sélectionnée. Auparavant, il fallait suivre une procédure pour récupérer manuellement les traductions et les placer au bon endroit sur le disque.
Sous Drupal 8, tous les éléments sont désormais des blocs qui peuvent être placés dans plusieurs régions d'une même page. Beaucoup de ces blocs et de ces pages sont aussi maintenant des vues. Ces deux nouveautés — bien que n'étant pas en soi des fonctions multilingues — offrent une grande liberté sur le choix de la langue d'affichage du contenu. En effet, les blocs possèdent à présent un paramètre de visibilité basé sur la langue. Les vues ont aussi de nombreuses options de configurations additionnelles par rapport à la langue du contenu à récupérer et dans laquelle celui-ci sera affiché.
Dans un projet multilingue, un contenu peut ne pas nécessiter d'être associé à une langue — une image par exemple — mais aussi ne pas avoir encore de langue associée. Sous Drupal 7, nous n'avions comme seul choix « Indépendant de la

langue », ce qui ne s'applique pas toujours à notre situation. Drupal 8 fournit maintenant deux options de langue génériques — « Non spécifié » et « Non applicable » — et permet même d'en ajouter si celles-ci ne sont pas suffisantes.

Chaque usager possède aussi davantage de liberté sur ses préférences de langue. Il peut sélectionner la langue de son profil, celle dans laquelle il préfère consulter le contenu et, s'il a accès aux pages d'administration, la langue dans laquelle celles-ci devraient lui être présentées. Ces nouvelles options permettent, par exemple, de rendre dynamique la langue d'affichage des contenus selon les préférences de chaque utilisateur.

Une expérience de traduction améliorée

Tout comme son prédécesseur, le module locale de Drupal 8, sert à la traduction de l'interface. Il incorpore maintenant les fonctionnalités de mise à jour du module communautaire i10n_update de Drupal 7. Il récupère donc les fichiers de traductions du cœur et des modules de Drupal et met à jour les traductions. Puisqu'il est souvent nécessaire, dans un projet multilingue, de modifier des traductions venant de la communauté, il est donc possible de les marquer comme étant personnalisées afin d'éviter qu'elles ne soient écrasées par la prochaine mise à jour. Il est aussi possible d'exporter nos traductions personnalisées pour les utiliser dans d'autres projets.

Sous Drupal 7, la fonction de traduction t() ne fonctionne que si la base de données est active. Pour les autres occasions, la fonction st() doit être utilisée. Ainsi, si nous souhaitons signaler du texte à traduire à un moment où la base de données peut ne pas être disponible, nous devons nous rabattre sur une troisième fonction, get_t(), qui se chargera de retourner la bonne fonction à utiliser. Sous Drupal 8, st() et get_t() ont été retirées et la fonction t() se charge d'évaluer son contexte. Si la base de données n'est pas disponible, elle ira chercher ses traductions directement dans les fichiers .po sur le disque.

L'interface a aussi été grandement retravaillée. Précédemment, la page de traduction n'offrait pas la possibilité de traduire les textes directement. Pour chaque chaîne de caractères, on nous présentait l'état des traductions dans toutes les langues disponibles et un seul lien pour les modifier. Ce lien menait au formulaire de traduction qui comportait un champ pour chaque langue — bien qu'il soit peu probable qu'une seule personne se charge de traduire toutes les langues !

Dès la première page, sous Drupal 8, on peut filtrer les traductions par langue et avoir côte à côte la source et un champ pour entrer ou modifier la traduction directement. Il est donc possible d'entrer plusieurs chaînes de traductions avant d'enregistrer. De plus, l'interface de Drupal 8 permet désormais de traduire les textes pluriels, tel que: « 1 commentaire / @count commentaires », ce que l'interface de Drupal 7 ne permettait aucunement.

Du contenu multilingue sans discrimination

Bien que portant le même nom que son homonyme en Drupal 7, le module content_translation est fondamentalement différent. Drupal 8 rend traduisibles toutes les entités de contenus, que ce soient les nœuds, les blocs, les termes de taxonomie, etc. Dans sa version précédente, ce module ne supportait que la traduction des nœuds. Si l'on souhaitait traduire les termes de taxonomie, les blocs ou les menus, il fallait utiliser des modules de la communauté.

Drupal 8 améliore grandement la traduction car tous les contenus sont maintenant des entités traduisibles à l'aide d'un système unique et configurables par type d'entité, par bundle, par champs et parfois même davantage. Pour un champ d'image, par exemple, on peut rendre traduisible le titre et le texte alternatif, tout en conservant la même image pour toutes les traductions.

Toutes les entités de contenu (exceptés les éléments de menu) peuvent être attachées à des champs, et tous ces champs — du titre aux métadonnées — sont traduisibles. On peut ainsi choisir (ou non) d'avoir des métadonnées différentes selon les traductions. Le nouveau système nous donne aussi le choix, concernant les termes de taxonomie, d'avoir soit des ensembles différents par langue (en traduisant le champ attaché au contenu) soit un seul ensemble pour toutes les traductions (en traduisant plutôt les termes eux-mêmes). Il est donc maintenant possible de traduire tout ce que l'on veut, selon le degré souhaité, et rien de plus.

Une autre différence majeure entre Drupal 7 et 8, c'est que cette dernière version ne duplique pas les contenus pour chacune des traductions, mais en traduit seulement les parties nécessaires. Par conséquent, toutes les traductions possèdent le même identifiant que leur source, et la suppression de la source entraîne la suppression de toutes ses traductions (une confirmation est bien sûr demandée).

Configuration dans la langue de notre choix

La configuration de Drupal 8 a été totalement repensée et utilise maintenant un système commun pour toutes ses facettes, que ce soient le nom du site, les rôles, les libellés des champs ou les textes dans les vues. Des fichiers en format YAML sont utilisés afin d'inclure la configuration dans le code, mais aussi pour son déploiement entre les différents environnements. Elle est ensuite transférée dans la base de données. De plus, tout comme pour le contenu, Drupal ne duplique pas les fichiers de configuration pour chaque traduction; il crée plutôt des fichiers avec seulement les textes supplantés.

Si Drupal 8 a été installé dans une langue autre que l'anglais, toute la configuration du site a été importée dans cette langue (grâce au module locale). Chaque traduction est ensuite effectuée depuis cette langue. Il n'est donc plus nécessaire de créer les vues en anglais et de les traduire par la suite. On peut désormais les créer dans la langue de notre choix et les traduire en anglais — ou dans toute autre langue — si nécessaire. On peut ainsi, dans un site multilingue, créer sa configuration dans n'importe quelle langue activée et le système sera capable de la traduire dans toutes les autres langues disponibles.

Toutes ces nouveautés font partie du système d'entité de configuration de Drupal 8. Ce que le module config_translation fournit, c'est une interface utilisateur pour effectuer les traductions. Tous ces textes de configuration sont aussi exposés dans l'interface de traduction du module locale, mais config_translation facilite l'expérience de traduction. Par exemple, il ajoute des onglets de traduction aux côtés des onglets d'édition des vues, des blocs, des menus, etc. Il est en effet plus intuitif de traduire ces groupes de textes sur la même page qu'en effectuant une recherche pour chaque chaîne de caractères depuis l'interface de locale. Le module fournit aussi une page qui liste toutes les configurations traduisibles suivies d'un lien vers leur page de traduction.

La migration

Toutes ces nouvelles fonctionnalités sont très utiles pour un nouveau projet, mais qu'en est-il pour un projet en Drupal 6 ou 7 que l'on souhaiterait migrer ? Là encore, c'est une autre nouveauté. Le module communautaire migrate, bien qu'étant encore au stade expérimental, est désormais incorporé dans le cœur de Drupal 8 et propose des chemins de migration automatique. A ce stade, bien que la migration depuis Drupal 6 soit en grande partie fonctionnelle, il n'existe aucun support concernant le contenu multilingue. Il est évident que beaucoup d'efforts sont toutefois déployés pour augmenter la couverture de ces migrations puisque cette version ne sera bientôt plus supportée... Heureusement, comme sous Drupal 7, il est possible d'utiliser l'API de migrate pour créer soi-même ses migrations multilingues.

Des améliorations en profondeur de la version core

Une des révolutions de Drupal 8 provient de son architecture entièrement repensée et mise au "goût du jour". En effet, avant Drupal 8, le code entier du core Drupal était écrit par la communauté Drupal. Le core Drupal 8 s'ouvre à la communauté Open Source (concept du "proudly found elsewhere") en devenant un best-of-breed de technologies open-source intégrant les meilleures librairies disponibles, et permettant aux développeurs Drupal de se focaliser sur les fonctionnalités CMS de Drupal et non les couches bas niveau (routing, connection à la base de données, ...). Drupal 8 a effectué des évolutions majeures de son API tout en intégrant ce qui se fait de mieux dans l'écosystème PHP.

Angela Byron,
Directrice du developpement communautaire chez Acquia
Angela est l'auteure principale du premier livre sur Drupal d'Oreilly intitule « Using Drupal » (Utiliser Drupal). Le pseudo d'Angela sur Drupal.org est « webchick ».

« Proudly Found Elsewhere » (Fièrement trouvé ailleurs)

Le concept du « Proudly found elsewhere » témoigne d'un changement dans la façon de penser des développeurs Drupal. En effet, trouver les meilleurs outils pour accomplir une certaine tâche et les incorporer au logiciel, au lieu de créer une application particulière appartenant en propre à Drupal, a été le leitmotiv des développeurs.

Vous pouvez constater ce changement de philosophie dans de nombreux composants de Drupal 8. Parmi les bibliothèques externes que nous avons incluses se trouvent PHPUnit pour le test unitaire, Guzzle afin d'effectuer des requêtes HTTP (vers des Web services), plusieurs composants de Symfony (le tutoriel « Create your own framework on top of the Symfony2 Components » est excellent pour en apprendre plus sur ces composants) et Composer comme gestionnaire de dépendances externes et de chargement automatique de classes, entre autres.

Mais ce changement de philosophie se constate aussi dans la base du code lui-même. Nous avons effectué d'importantes modifications de l'architecture dans Drupal 8 afin de nous adapter à la façon dont on organise le code dans l'écosystème PHP : une programmation découplée, orientée objet (OO) et utilisant les éléments du langage PHP dans ses versions les plus récentes, telles que les espaces de nom et les traits. AIN

Prenons quelques exemples d'extraits de codes pour illustrer dans les faits l'architecture « Proudly Found Elsewhere » de Drupal 8.

Drupal 7 : example.info

```
name = Example
description = An example module.
core = 7.x
files[] = example.test
dependencies[] = user
```

Tous les modules dans Drupal nécessitent un fichier .info pour pouvoir être enregistrés dans le système. L'exemple ci-dessus est typique d'un module de Drupal 7. Le format du fichier ressemble à un fichier « INI » facilement éditable, mais il comprend également des « Drupalismes » tels que la syntaxe en array[] qui empêche l'utilisation de fonctions standard PHP de lecture/écriture des fichiers INI. La clé files[], qui déclenche le chargement automatique des classes personnalisées pour ajouter du code dans la base de registre, est particulièrement « drupalienne », et les développeurs de modules qui codent en orienté objet doivent ajouter une ligne files[] pour chaque fichier qui définit une classe, ce qui peut devenir stupide.

Drupal 8 : example.info.yml

```
name: Example
description: An example module.
core: 8.x
dependencies:
  - user
# Note: New property required as of Drupal 8!
type: module
```

Dans la droite ligne du concept « Proudly Found Elsewhere », les fichiers infos dans Drupal 8 sont désormais de simples fichiers YAML, les mêmes qui sont utilisés par d'autres langages et dans d'autres framework. La syntaxe est très similaire (principalement : à la place de =, et des tableaux formatés différemment), et il est toujours aussi facile de lire et d'écrire dans ces fichiers. La clé gênante files[] a été supprimée et remplacée par la norme PSR-4 pour l'auto-chargement des classes par le biais de Composer. En d'autres termes, c'est en suivant une convention spécifique sur le nommage/dossier de la classe (modules/example/src/ Example Class.php), que Drupal peut charger automatiquement le code orienté objet sans nécessiter un enregistrement manuel.

Drupal 7: hook_menu()

Il s'agit d'un module assez basique dans Drupal 7 qui définit une URL sur /hello qui, lorsqu'on y accède, fait des vérifications pour s'assurer que l'utilisateur dispose des permissions d'« accès au contenu » avant d'appeler la fonction _example.page(), qui affiche « Hello world. » à l'écran en tant que page entièrement chartée. Le hook_menu() est un exemple de ce qui est connu de façon péjorative sous le nom de ArrayPI, qui est un modèle d'architecture courant dans Drupal 7 et les versions antérieures.

Le problème posé par les ArrayPI est qu'ils sont difficiles à taper (par exemple, n'avez-vous jamais oublié le return $items, puis passé 30 minutes à chercher comment résoudre le problème?), ils ne disposent pas d'auto-complétion dans l'EDI (environnement de développement intégré) pour les propriétés disponibles, enfin la documentation doit être mise à jour manuellement à chaque fois qu'une clé est modifiée ou ajoutée.

La documentation sur hook_menu() montre que le défaut vient également du fait que trop de choses essaient d'être faites à la fois.

Ce programme est utilisé pour enregistrer les associations/accès entre les chemins et les pages, mais également pour afficher les liens dans l'interface de plusieurs façons, changer de thème, et bien plus.

Drupal 8: Routes + contrôleurs example.routing.yml

```
example.hello:
  path: '/hello'
  defaults:
    _content: '\Drupal\example\Controller\Hello::content'
  requirements:
    _permission: 'access content'
```

src/Controller/Hello.php

```
<?php
namespace Drupal\example\Controller;

use Drupal\Core\Controller\ControllerBase;

/**
 * Greets the user.
 */
class Hello extends ControllerBase {
  public function content() {
    return array('#markup' => $this->t('Hello world.'));
  }
}
?>
```

Dans le nouveau système de routage de Drupal 8, la création des routes chemin/accès est maintenant assurée par un fichier YAML qui utilise la même syntaxe que le système de routage Symfony.

La logique de «fonction de callback» est désormais assurée par une classe « Controller » (comme dans le modèle standard modèle-vue-contrôleur) dans un fichier nommé de façon spécifique, conformément à la norme PSR-4. Dans le module exemple ci-dessus, le « controller » est déclaré dans son propre espace de nom pour permettre au module de nommer ses classes comme il le souhaite, sans se soucier d'éventuels conflits avec d'autres modules.

Finalement, la classe hérite de la logique de la classe ControllerBase donnant ainsi au contrôleur Hello l'accès à toutes les méthodes et capacités du ControllerBase, telles que $this->t() (la façon orientée objet de nommer la fonction t()). Étant donné que ControllerBase est une classe conforme à la norme PHP, toutes ses méthodes et propriétés seront complétées automatiquement dans les EDI ; vous n'aurez donc pas à deviner ce qu'il peut ou ne peut pas faire pour vous.

Drupal 7: hook_block_X()

block.module

```
<?php
/**
 * Implements hook_block_info().
 */
function example_block_info() {
  $blocks['example'] = array(
    'info' => t('Example block'),
  );
  return $blocks;
}

/**
 * Implements hook_block_view().
 */
function example_block_view($delta = '') {
  $block = array();
  switch ($delta) {
    case 'example':
      $block['subject'] = t('Example block');
      $block['content'] = array(
        'hello' => array(
          '#markup' => t('Hello world'),
        ),
      );
      break;
  }
  return $block;
}
?>
```

Voici un exemple typique de la façon dont on définit un nouveau composant dans Drupal (blocs, effets d'image formats de texte, etc.) : une sorte de « hook » _info déclaratif, associé à un ou plusieurs autres « hooks » pour exécuter d'autres actions (voir, appliquer, configurer et bien plus). En plus de ces éléments, qui sont pour la plupart des ArrayPI, on constate qu'il y a des API bien plus difficiles à prendre en main, car l'API générale elle-même ne peut absolument pas être découverte, à moins d'inspecter rigoureusement tous les fichiers .api.php des différents modules à supposer qu'ils existent, ce qui n'est pas garanti) pour découvrir les fonctions « hooks » nommées par « magie » dont vous avez besoin pour définir l'application de tel ou tel comportement. Certains sont nécessaires, d'autres non. Parviendrez-vous à deviner lequel est lequel ?

Drupal 8 : Les blocs (et bien d'autres éléments) en tant que plugins

Dans Drupal 8, ces API mystères sont en majorité transférées vers le nouveau système de plugin, qui ressemble à ce qui suit :

src/Plugin/Block/Example.php

```
<?php
namespace Drupal\example\Plugin\Block;

use Drupal\Core\Block\BlockBase;

/**
 * Provides the Example block.
 *
 * @Block(
 *   id = "example",
 *   admin_label = @Translation("Example block")
 * )
 */
class Example extends BlockBase {
  public function build() {
    return array('hello' => array(
      '#markup' => $this->t('Hello world.')
    ));
  }
}
?>
```

Cet exemple ressemble beaucoup à celui du contrôleur ; un plugin est une classe qui, dans le cas présent, s'étend à partir d'une classe de base (BlockBase) qui prend soin d'implémenter certains mécanismes pour vous. L'API des blocks elle-même est déclarée dans l'interface BlockPluginInterface et implémenté par la classe BlockBase.

Vous remarquerez qu'en règle générale, les interfaces exposent et renseignent plusieurs API de façon à ce que l'EDI soit découverte et agréable à utiliser. Le meilleur moyen de découvrir les nouvelles API de Drupal 8 consiste à naviguer parmi les interfaces proposées.

Les commentaires au-dessus des classes sont appelés « annotations ». À première vue, l'utilisation des commentaires PHP pour spécifier des métadonnées qui affectent la logique applicative semble étrange, mais cette technique est désormais largement utilisée par de nombreuses bibliothèques modernes en PHP et acceptée par la communauté. Cela offre l'avantage de conserver les métadonnées de classe dans le même fichier et juste à côté de la définition de la classe.

Drupal 7 : « Hooks »

Dans Drupal 7 et ses versions antérieures, le mécanisme d'extension utilisé est le concept de « hooks ». En tant que créateur d'une API, vous pouvez déclarer un « hook » utilisant des fonctions telles que module_invoke_all(), module_implements(), drupal_alter(), etc.

Si vous voulez qu'un module réponde à cet évènement, il vous faut créer une fonction nommée modulename_hookname() et déclarer ses résultats d'une façon reconnaissable par l'utilisation d'une fonction « hook ». Par exemple :

```
<?php
/**
 * Implements hook_permission().
 */
function menu_permission() {
  return array(
    'administer menu' => array(
      'title' => t('Administer menus and menu items'),
    ),
  );
}
?>
```

Bien qu'il s'agisse d'un mécanisme astucieux, ce choix d'architecture est surtout dû à l'ancienneté de Drupal (Drupal a démarré en 2001, à l'époque où le PHP3 battait son plein et où le code orienté objet et autres n'étaient pas encore plébiscités), certaines choses ne sont pas si évidentes :

- Ce mécanisme d'extension qui « nomme une fonction d'une façon particulière » tient vraiment du drupalisme, et les développeurs qui s'initient à Drupal peinent à le comprendre au début
- Au moins quatre fonctions différentes peuvent lancer un « hook » : module_invoke(), module_invoke_all(), module_implements(), drupal_alter() et bien d'autres encore. Cela rend les extensions disponibles très difficiles à trouver dans Drupal.
- Il n'y a pas de cohérence entre ce qu'attendent les « hooks ». Certains « hooks » sont de type info et requièrent un tableau (parfois un tableau de tableau de tableau de tableau), d'autres sont des « hooks » de type info qui ne répondent qu'en cas d'évènement particulier, par exemple, le lancement d'une tâche cron ou la sauvegarde d'un nœud. Vous devez lire la documentation relative à chaque « hook » pour comprendre quelles entrées et quelles sorties ils attendent.

Drupal 8 : Évènements (« Events »)

Bien que les « hooks » dominent encore largement dans Drupal 8 dans le cas de comportements commandés par évènements (les « hooks de type info ont été en grande partie transformés en annotations YAML ou en plugins), les portions de Drupal 8 compatibles Symfony (par exemple bootstrap/exit, système de routage) ont pour la plupart été transférées dans le système Event Dispatcher de Symfony.

Dans ce système, les évènements sont distribués lors de l'exécution lorsque certaines logiques sont en cours, et les modules peuvent « abonner » leurs classes aux évènements auxquels ils veulent qu'elles réagissent.

Pour le prouver, regardons de plus près la configuration API de Drupal 8, qui est stockée dans Core/lib/Drupal/Core/Config/Config.php. Elle définit de nombreuses méthodes CRUD telles que save(), delete(), etc. Chaque méthode déclenche un évènement lorsque sa tâche est accomplie, ce qui permet aux autres modules de réagir. Voici par exemple Config::save() :

```
<?php
  public function save() {
    // <snip>Validate the incoming information.</snip>

    // Save data to Drupal, then tell other modules this was
    // just done so they can react.
    $this->storage->write($this->name, $this->data);
    // ConfigCrudEvent is a class that extends from Symfony's
    // "Event" base class.
    $this->eventDispatcher->dispatch(ConfigEvents::SAVE, new
    ConfigCrudEvent($this));
  }
?>
```

Lorsque le processus s'arrête, il y a au moins un module qui a besoin de réagir lorsque la configuration est sauvegardée : le module du core Language. Car si le paramètre de configuration qui vient d'être modifié était la langue du site par défaut, les fichiers PHP compilés doivent être rechargés pour que la modification puisse être effective.

Pour ce faire, le module Language accomplit trois tâches :

1. Il enregistre une classe éligible à un évènement dans son fichier language.services.yml (il s'agit d'un fichier de configuration du Service Container de Symfony pour l'enregistrement des codes réutilisables)

```
language.config_subscriber:
  class: Drupal\language\EventSubscriber\ConfigSubscriber
  tags:
    - { name: event_subscriber }
```

2. Dans la classe référencée, il exécute le EventSubscriberInterface et déclare une méthode getSubscribedEvents() qui énumère les évènements pour lesquels il devrait être alerté. Il fournit également pour chaque évènement une ou plusieurs fonctions de callback qui doivent être lancées lorsque l'évènement débute :

```
<?php
class ConfigSubscriber implements EventSubscriberInterface {
  static function getSubscribedEvents() {
    $events[ConfigEvents::SAVE][] = array('onConfigSave', 0);
    return $events;
  }
}
?>
```

3. Il définit la méthode de callback qui contient le code à exécuter lorsque la configuration est sauvegardée :

```
<?php
  public function onConfigSave(ConfigCrudEvent $event) {
    $saved_config = $event->getConfig();
    if ($saved_config->getName() == 'system.site' &&
     $event- >isChanged('langcode')) {
      // Trigger a container rebuild on the next request.
     PhpStorageFactory::get('service_container')
     ->deleteAl();
    }
  }
?>
```

Pour couronner le tout, nous avons intégré un utilitaire de registre plus explicite pour qu'un seul module puisse souscrire plusieurs classes à des évènements individuels. Cela nous permet d'éviter les situations du passé où il y avait des déclarations conditionnelles (de type switch) dans les « hooks » ou encore de nombreux blocs de code sans rapport et se gênant les uns les autres. Au lieu de cela, on a la possibilité de séparer la logique en plusieurs classes séparées et distinctes. Cela signifie également que notre logique d'évènement est chargée au moment où elle doit être exécutée, et qu'elle ne monopolise pas constamment de la mémoire PHP.

Il est également assez simple de déboguer les évènements et de tracer leur exécution. Au lieu d'avoir toute une flopée de fonctions PHP, de procédures qui peuvent ou non avoir été utilisées pour invoquer votre « hook », c'est le même « Event Dispatcher » qui est utilisé dans tout le système. De plus, trouver les exécutions est aussi simple que d'exécuter une commande grep pour trouver la bonne constante de classe, par exemple : ConfigEvents::SAVE.

Le système d'évènements complète logiquement la transition vers une méthode orientée objet. Les plugins prennent en charge les « hooks » de type info ainsi que les « hooks » invoqués suite à un autre «hook» info. La plupart de nos anciens systèmes sont remplacés par des systèmes de registre explicites en YAML.

Le système d'évènements remplace les « hooks » de type évènement et intègre une méthodologie de souscription performante capable d'étendre des fonctionnalités du core.

...et beaucoup, beaucoup plus !

Vous pourrez trouver une bonne introduction aux modifications de l'API de Drupal 8 sur le site api.drupal.org qui contient également une liste des sujets groupés qui pourront vous guider dans Drupal 8.

Vous pouvez aussi vous rendre sur https://drupal.org/list-changes pour accéder à la liste complète des modifications de l'API entre Drupal 7 et Drupal 8. Toutes les modifications enregistrées de l'API comprennent des exemples de codes avant/après pour faciliter votre migration, ainsi que des conseils qui vous expliquent les modifications et leurs raisons.

Convertir son site Drupal en une App Drupal

Un site Drupal ne se limite pas à un site de gestion de contenus. Il peut être utilisé comme une application Web pour le rendre disponible sur d'autres matériels ou le porter sur des plateformes ouvertes comme Firefox OS de la fondation Mozilla.

Christophe Villeneuve
Consultant IT pour Neuros, Mozilla Reps, auteur du livre "Drupal avancé" aux éditions Eyrolles et auteur aux Editions ENI, Rédacteur pour WebRIVER, membre des Teams DrupalFR, AFUP, LeMug.fr (MySQL/MariaDB User Group FR), Drupagora...

Une source pour des multi-devices

Un site Drupal s'utilise à travers un navigateur. Il est accessible sur n'importe quel terminal ou matériel compatible Web. Ainsi, vous trouverez dans cette catégorie les ordinateurs de bureau ou ordinateurs portables, les tablettes, les smartphones... Et même les objets connectés. Cependant, dès le début de votre projet, il est intéressant de penser que celui-ci est souvent spécifique, unique, et par la même occasion de le proposer dans vos applications mobiles, tout en gardant le contenu disponible à travers votre navigateur. La solution présentée a l'avantage de profiter des spécificités du mobile comme la géolocalisation, ou d'options mobiles. Drupal permet de communiquer avec d'autres sites Internet pour en récupérer du contenu. Ainsi une interaction avec votre site Internet est possible en utilisant les Web Services. Cette technique permet de créer des applications pour les différents mobiles du marché : Android, iOS, Windows Phone / Mobile, Ubuntu, Firefox OS....

Approche générale

Pour mettre en place les Web Services, Drupal a besoin de 2 modules :

- Services (http://drupal.org/project/services)
- Libraries (http://drupal.org/project/libraries)

Il faudra activer les modules Services, Libraries, Rest Server pour son bon fonctionnement. L'étape suivante passe par la création de la structure du Web Service qui s'effectue en 2 temps. Tout d'abord, nous ajoutons un service. L'information principale étant le endpoint, c'est à dire le chemin à partir duquel Drupal sera capable de comprendre que c'est un Web Service et non une page du site classique, et nous renverra les bonnes informations Fig.1. Une fois le Web Service créé, vous pouvez activer certaines fonctionnalités comme l'affichage des nœuds ou la connexion utilisateur Fig.2.
Maintenant que le nœud principal est défini, vous choisissez les ressources qui seront fournies Fig.3. Après la sauvegarde, vous pourrez appeler le Web Service depuis votre mobile, par exemple, l'affiche d'un nœud présenté de la manière suivante : http://votreSite.com/services/node/2

N.B. : *la partie présentée est inspirée du livre « Drupal avancé » aux éditions Eyrolles*

Votre site Web en une application ouverte

Une application ouverte appelée WebApps, est un site Web à part entière et se connectera à distance à votre Site internet. Il appellera le Web Service que nous avons créé précédemment. L'article utilisera comme environnement Firefox. Mais vous avez la possibilité de le rendre compatible sur les autres smartphones.

Une plateforme ouverte

Lancé en 2013 par la fondation Mozilla, Firefox OS est un Web pour smartphone. Il utilise les formats ouverts du Web (HTML5 / JS / CSS3).
Le système possède une architecture à 3 niveaux :

- Gonk : la couche basse du téléphone comme un cœur ;
- Geko : le moteur de rendu HTML5 ;
- Gaia : l'interface utilisateur.

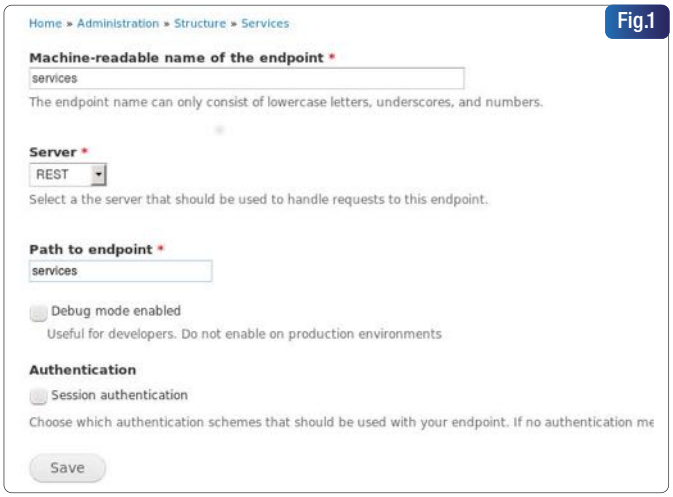


Fig.1

Fig.2

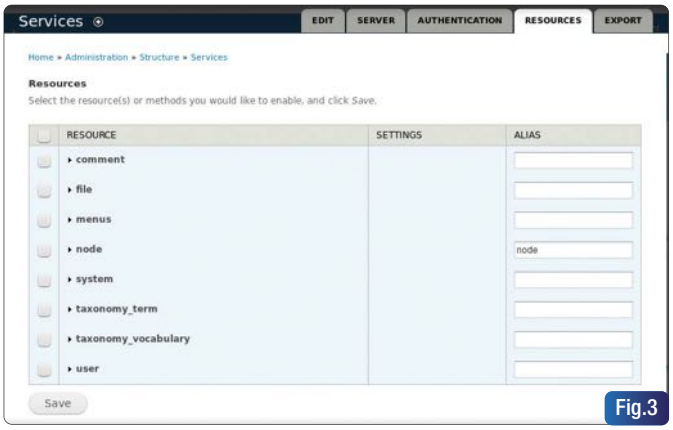


Fig.3

L'interface utilisateur est l'emplacement où les applications seront accessibles à partir du MarketPlace. Ainsi, avec cette approche vous construisez des applications avec seulement du HTML, Javascript et CSS.

Cas pratique

Il existe plusieurs manières pour publier une application Drupal sur un environnement mobile comme Firefox OS :

- Soit de garder sur votre serveur l'application ;
- Soit de la mettre à disposition dans la MarketPlace.

L'article va montrer comment déporter l'application Drupal en une application mobile et comme exemple, nous utiliserons votre magazine préféré « Programmez » pour obtenir le résultat suivant : Fig.4.
Pour rendre le site Internet visible sur le Marketplace de Firefox OS, un certain nombre d'étapes sont nécessaires :

Etape 1 : les icônes

Vous devez créer différents fichiers de différentes dimensions pour rendre votre application visible sur vos différents devices (desktop, ordinateurs portables, tablettes, smartphones...) du marché. Il y a 2 formats obligatoires pour rendre votre application disponible sur cette plateforme ouver-

te. Les dimensions nécessaires doivent être réalisées aux formats PNG et de dimensions (en pixels) : 128x128 et 512x512. Il est possible de prévoir des icônes supplémentaires pour 16x16, 48x48, 128x128, 256x256 pour afficher une icône de bonne qualité. Pour faciliter la gestion de vos fichiers, nous préconisons de ranger ces fichiers dans un dossier images et icons Fig.5.

Etape 2 : fichier Index.html

Le fichier index.html est un fichier au format HTML5. Il s'agit du premier fichier lancé par l'application, et appelé par le manifest (voir plus bas).

Fichier : index.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" manifest="manifest.appcache">
<head>
<meta name="viewport" content="width=device-width" />
<title>Programmez, le magazine du développeur</title>
<link rel="stylesheet" type="text/css" href="css/programmez.css" />
</head>
<body>
  <article>
   < !-- le code source de la page →
  </article>
</body>
</html>
```

Il est important d'utiliser les balises HTML5, comme ceci vous utilisez la meta « Viewport » pour définir la largeur de l'écran. Ensuite, nous prévoyons d'embarquer dans notre application le logo de votre magazine en modifiant sa taille et nous le centrons en utilisant les balises CSS.

```
<header>
<img src="img/logo.png" id="logo" alt="Programmez, le magazine du développeur" />
</header>
```

L'étape suivante affiche la couverture de votre magazine en cours Fig.6. Comme nous ne pouvons pas embarquer l'image dans notre application, nous insérons un lien pour là télécharger Fig.7 et 8.

```
<p>
<a href="http://www.programmez.com/magazine">
```

Fig.5


Fig.8

Fig.4

Fig.6

Fig.7

```
<img width="167" height="230" alt="couverture programmez"
src="http://www.programmez.com/sites/default/files/styles/magazine_couver_home/public/img_magazine/couv-188.jpg"
typeof="foaf:Image">
</a>
<br />
<a href="http://www.programmez.com/magazine/">Consultez le contenu</a>
</p>
```

Pour afficher le contenu publié venant du site Internet, vous utiliserez les Web Services avec le nœud créé précédemment. Comme c'est expliqué dans l'ouvrage « Drupal Avancé » aux éditions Eyrolles.

Cependant, le script du cas pratique présenté aujourd'hui sera un script générique pour gagner de la place.

Etape 3 : fichier manifest.webapp

Le fichier manifest.webapp est la nouveauté dans la réalisation d'un projet Web. Il s'agit d'un connecteur qui utilise le format ouvert jSON.

Ce fichier se décompose de la façon suivante :

- Version : est le contrôle du logiciel pour l'application ;
- Nom : le nom de votre application ;
- Chemin de lancement : il doit pointer sur votre fichier index.html ;
- Icônes : définir le chemin des icônes ;
- Développeur : votre nom de développeur et un lien.

Bien entendu, de nombreuses autres options sont disponibles à la page suivante : https://developer.mozilla.org/fr/Apps/Manifeste.

Elle vous propose la possibilité de définir plusieurs langues, permissions, l'orientation, le serveur, le mode plein écran...

```
{
 "version": "1.0",
 "name": "Programmez",
 "description": "Programmez, le magazine du développeur",
 "launch_path": "/index.html",
 "icons": {
  "16": "/img/icons/programmez-16.png",
  "48": "/img/icons/programmez-48.png",
  "128": "/img/icons/programmez-128.png",
  "512": "/img/icons/programmez-512.png"
 },
 "developer": {
  "name": "Christophe Villeneuve",
  "url": "http://www.hello-design.fr"
 },
 "installs_allowed_from": ["*" ],
 "appcache_path": "/cache.manifest",
 "locales": {
  "fr": {
   "description": "Programmez, le magazine du développeur",
   "developer": {
    "url": "http://www.hello-design.fr"
   }
  }
 },
 "default_locale": "en"
}
```

Etape 4 : fichier manifest.appcache

La fichier manifest.appcache a pour rôle de stocker en mémoire certains fichiers pour éviter de les recharger ou de les uploader lors du prochain lan-

cement de votre application. Une page composée de tous les détails se trouve à l'adresse suivante : https://developer.mozilla.org/fr/docs/Utiliser_Application_Cache

```
CACHE MANIFEST

# Version 0.8

#NETWORK:
CACHE:
css/programmez.css
img/logo.png
img/icons/programmez-16.png
img/icons/programmez-128.png
img/icons/programmez-256.png
img/icons/programmez-512.png
js/flux.js
index.html
```

Comme le montre le script, nous listons l'ensemble des fichiers qui ne changent pas car ils sont disponibles dans le téléphone et vous n'avez pas besoin de recharger les fichiers stockés à distance.

Attention, *il est impératif d'avoir le même numéro de version entre les 2 fichiers manifest (étape 3 et 4) qui est utile si vous modifiez un des fichiers listés.*

Pour que le fichier soit pris en compte, il faut ajouter une ligne supplémentaire dans l'étape 3 :

```
"appcache_path": "/cache.manifest",
```

Fig.9

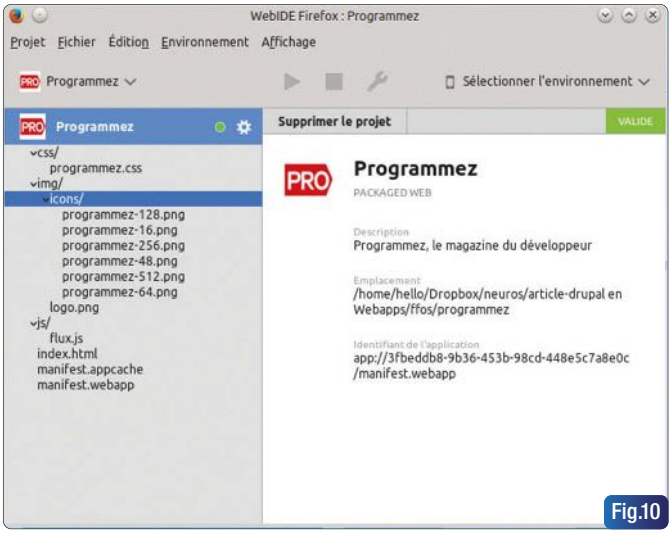


Fig.10

Etape 5 : validation

La première des validations passe par le WebIDE disponible par défaut dans le navigateur Firefox. Pour cela, vous trouverez l'outil à partir de l'icone correspondante (icône entourée) ou à partir du raccourci clavier [shift] + [F8]. Fig.9. Ainsi, vous obtenez une interface permettant de sélectionner votre webApp et la version de Firefox OS Fig.10.

L'interface proposée est une console qui regroupe un player pour exécuter le programme, mais aussi une boite à outils pour corriger les éventuels bugs obtenus. La partie droite de la fenêtre propose de tester votre application à travers différentes versions de votre navigateur en mode simulateur ou de connecter directement un téléphone.

Il est possible d'aller plus loin que dans ce WebIDE directement à la page suivante : https://developer.mozilla.org/fr/docs/Outils/WebIDE

Etape 6 : proposer son APP

Avant de proposer son application sur la marketplace de FirefoxOS, vous devez penser où celle-ci sera distribuée pour permettre sa diffusion et son téléchargement. Les possibilités sont variées car vous pouvez la mettre à disposition :

- Sur le Marketplace de Firefox OS ;
- La stocker sur votre serveur ;
- La proposer sur une plateforme collaborative (ex : Sourceforge, GitHub...)

Le choix d'aujourd'hui va se porter sur le MarketPlace de Firefox OS et nous compressons l'application au format ZIP, qui sera la dernière étape pour l'envoyer sur le MarketPlace (https://marketplace.firefox.com/).

Rendez-vous à la page suivante (Developper \ proposez une application) ou directement à partir du lien suivant : https://marketplace.firefox.com/developers/submit/

Une fois que vous remplissez le formulaire et téléchargez le fichier zip, vous devez attendre l'examen et l'approbation. Cela peut prendre de quelques heures à plusieurs jours Fig.11. Lors de la soumission, vous déterminez la compatibilité de votre application, comme ceci, vous n'avez pas l'obligation de la refaire comme ici pour Android. Ensuite, vous devez suivre la procédure étape par étape pour uploader votre fichier. Lorsque l'ensemble des étapes se sont correctement déroulées, vous devrez patienter au maximum quelques jours pour l'approbation finale https://marketplace.firefox.com/developers/

Enfin...

Comme vous le voyez, votre application Drupal peut communiquer avec différents devices pour partager du contenu.

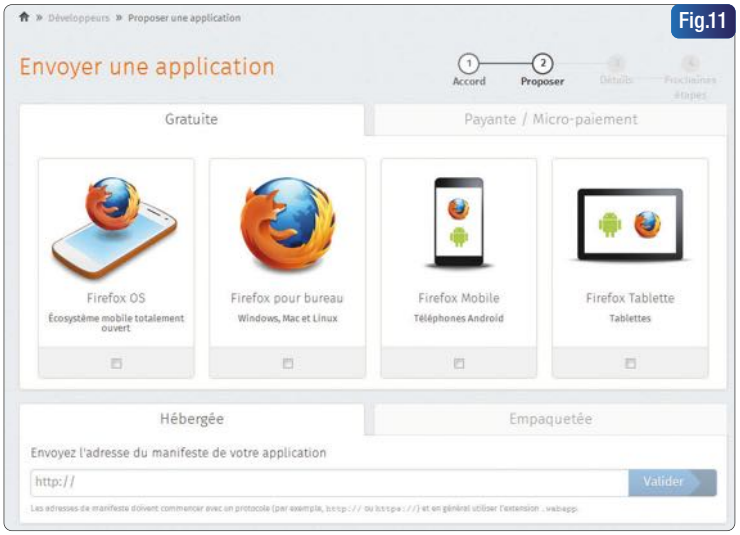


Fig.11

Retour sur la Drupalcon de Barcelone

Comme à chaque fois qu'elle a lieu, la Drupalcon déchaîne les passions, mais aussi, avouons-le, quelques déceptions en fonction des sessions auxquelles les utilisateurs participent.

Emmanuel Quedville
Core-Techs

(Source : http://www.appsdev.is.ed.ac.uk/blog/wp-content/uploads/2015/09/DrupalCon_Driesnote-1024x463.jpg)

Mais s'il y a bien un moment solennel apprécié de tous, c'est bien la grande messe du 1er jour, présentée pardon... célébrée par Dries Buytaert, le grand gourou de Drupal. Cette conférence, souvent de haut vol, est un moment fort de la convention. Dries Buytaert y partage sa vision sur l'évolution de Drupal mais aussi, de manière plus générale sur les valeurs de l'Open-Source comme il l'avait si bien fait lors des précédentes DrupaCon. Devant une assemblée sans doute déjà acquise, la Keynote de Barcelone était surtout l'occasion pour le fondateur de Drupal de s'expliquer sur le retard prolongé de D8, de crever l'abcès sur l'éventuelle « perte de vitesse » de Drupal face à ses concurrents, de rassurer une communauté inquiète de certains choix technologiques. Découpées en trois grandes thématiques et martelées à chaque fois par « We need to talk about it », nous évoquerons donc les aspects suivants :

Processus de développement :

- Pourquoi la solution Drupal est-elle en perte de vitesse ?
- Pourquoi Drupal 8 n'est-il pas sorti en temps et en heure ?

Positionnement sur le marché :

- Est-ce que Drupal peut encore concurrencer les solutions CMS actuelles ?
- Drupal 8 peut-il mieux répondre à des profils non techniques ?

Pertinence de la technologie :

- Quelle est la pertinence technologique de Drupal 8 face aux frameworks « mastodontes » et incontournables du moment ?
- Faut-il considérer ces Frameworks comme une menace, comment réagir ?

Processus de développement

Même son créateur admet sans détour : Drupal a perdu de son dynamisme.

Cette constatation n'a finalement rien d'alarmant et s'explique par ce qu'on appelle « L'effet Osborne », du nom de l'auteur qui, ayant annoncé trop tôt une nouvelle version de son micro-ordinateur, a conduit son entreprise à la faillite.

L'annonce trop précoce de Drupal 8 a provoqué une adoption plus lente de Drupal 7. Mais il n'y a pas de craintes à avoir, cette défection pour la solution est cyclique. Le même phénomène s'était produit mi-2010, lorsque Drupal 7 était en chantier : l'adoption de Drupal 6 avait alors chuté de 28%. Il est par ailleurs démontré que cette baisse engendre par la suite un réel engouement, qui devrait se reproduire lors de la sortie de Drupal 8.

Pourquoi Drupal 8 n'est-il pas sorti en temps et en heure ?

On a tendance à l'oublier mais Drupal existe depuis presque 15 ans. 5 ans ont été nécessaires pour développer Drupal 8, ce qui représente un cycle de développement particulièrement long, surtout lorsqu'on le compare aux précédentes « release » (8 mois entre Drupal 4 et 5, 13 mois entre Drupal 5 et 6, 3 ans pour passer de Drupal 6 à 7) et de manière plus générale aux autres solutions du marché.

Ce retard peut sembler d'autant plus incompréhensible que la communauté elle, n'a cessé de croître chaque année. Plus spécifiquement sur Drupal 8, on compte quand même plus de 3.000 contributeurs, 15 000 patches, 1300 bugs corrigés. Rappelons aussi que la mise en œuvre de cette nouvelle version suit les mêmes péripéties que la mise en place de n'importe quel projet que nous, agences, intégrateurs de solutions Internet, mettons en œuvre : une fonctionnalité plus complexe qu'il n'y paraissait, une charge sous-évaluée en développement, une livraison qui déclenche des effets de bord non prévus... Bref, il est difficile d'anticiper toutes les difficultés inhérentes à un projet, et finalement, sain de se rappeler que le développement de Drupal reste aussi, et avant tout, un projet humain Fig.1.

A qui imputer ce retard ? « Il n'y a pas de coupable » comme le soulignait Dries, même s'il reconnaît que le développement de Drupal 8 a été par certains côté un peu « chaotique ». Le découpage de Drupal 8 en plusieurs initiatives, elles-mêmes découpées en plusieurs fonctionnalités a morcelé les développements, provoquant des régressions entre des modules et le noyau Drupal mais aussi entre les modules entre eux.

Fig.1

C'est d'ailleurs ce qui s'est passé lors de la phase « Alpha » de Drupal 8. Pendant 1 mois, aucun bug critique n'avait été remonté par la communauté jusqu'au jour où une simple « release » - sans doute de module - a provoqué une série noire de plusieurs bugs majeurs, repoussant de plusieurs semaines une éventuelle sortie « RC » qui semblait pourtant très proche.

Afin de pallier cette problématique, il a été décidé de « merger » les développements une fois la fonctionnalité terminée. Comme certains membres l'ont remarqué, la terminologie de Drupal change en version 8. D'une version « 7.x », on passe à une version « 8.x.x »

- Le 1er « x » correspond à la mise en place de nouvelles fonctionnalités dans le noyau dont le très attendu « Migrate Update » en V8.1 et qui permettra de simplifier la migration des sites Drupal 7 vers Drupal 8)
- Le 2e « x » appelé « Semantic versionning » correspond à la montée de version mineure, notamment en cas de « Bug fixe ». Son fonctionnement est aujourd'hui similaire à celui des versions précédentes de Drupal, notamment Drupal 6 et 7.

Depuis cette conférence une RC2 (21 octobre 2015) puis une RC3 (4 novembre 2015) sont sorties. La version « finale » dite drupal-8.0.0 a été annoncée.

Positionnement sur le marché

Souvent comparé à ses homologues WordPress et Joomla!, Drupal fait partie des 3 solutions CMS Open Source les plus populaires et utilisées dans le monde. Les 2 premières solutions ont par ailleurs profité de la « léthargie » de Drupal pour grandir vite, nous y reviendrons un peu plus tard.

Il est souvent dit que WordPress est la solution la plus utilisée dans le monde, suivie de Joomla! puis de Drupal. Chiffres de la société Alexa à l'appui - qui rappelons une société reconnue dans son rôle de fournisseur de statistiques sur le trafic Web mondial -, Dries explique que Drupal est la plateforme dominante pour des sites « conséquents » et « complexes ». Ceci s'explique notamment par les outils dont Drupal dispose en termes de modélisation de contenus. Certains CMS commencent à suivre cette même logique, mais restent encore très loin de ce que Drupal propose maintenant depuis des années.

Ce qui est en revanche plus surprenant, c'est de découvrir qu'en réalité, le CMS Joomla! semble avoir disparu de ce classement au profit de CQ5 (renommé désormais en Adobe Experience Manager), la solution développée par Adobe. Concurrent de plus en plus sérieux des CMS du marché, c'est un acteur inattendu qui a su rapidement s'imposer et grappiller des parts de marché. D'autre part, même si Drupal est la solution la mieux placée sur les sites générant plus de 10.000 et 100.000 visiteurs, WordPress reste la solution N°1 pour les sites générant plus d'un million de visiteurs. Quel que soit notre niveau d'empathie pour cette solution, Il y a forcément certaines choses que la communauté Drupal doit aussi apprendre de la communauté WordPress Fig.2.

Drupal est-il un concurrent de Wordpress ?

On compare souvent les solutions « WordPress » et « Drupal ». Pour Dries, c'est une erreur car ces solutions sont différentes : elles s'adressent à des audiences et à des cibles qui ne sont pas les mêmes. On remarque aussi que la philosophie de ces 2 CMS n'évolue pas forcément dans le même sens. Les dernières versions de WordPress sont orientées « Site Builder Experience » et offrent des facilités d'usage pour les personnes en charge des contenus. Ces profils pas ou peu techniques bénéficient depuis quelques versions de nouveautés intéressantes : nouvelle médiathèque avec Drag & Drop, édition des thèmes, gestion de playlist audio et vidéo, nouveau tableau de bord d'administration, prévisualisation améliorée... Quant à Drupal 8, il est pour le moment clairement orienté « Developer Experience ». Les nouvelles fonctionnalités restent aujourd'hui très techniques puisqu'elles concernent l'intégration de Symfony2, de Twig, l'amélioration des Web Services et des performances générales, la présence de Views directement dans le noyau, la gestion des modules ...

Drupal 8 peut-il mieux répondre à des profils non techniques ?

Mêmes si certaines des fonctionnalités « Site builder experience » n'ont pas attendu Drupal 8 pour exister (la médiathèque par Drag and Drop est possible par exemple avec le module ScalD), il faut quand même avouer que Drupal 8 peut sembler complexe pour certains profils pas ou peu technophiles. De son propre aveu, Dries explique que c'est un point de vigilance auquel Drupal 8 doit répondre, non pas que pour « grappiller » de nouveaux sites mais aussi pour séduire aussi une nouvelle cible, celle qui n'utilise pas de CMS actuellement sur son site (estimé à 80% du monde !).

Pertinence de la technologie

Petite histoire des CMS

Flashback !

Il y a presque 20 ans, les notions de pages et de contenus étaient très liées : le webmaster disposait d'un unique fichier HTML dans lequel figuraient à la fois le fond (le contenu) et la forme (la page). Lorsqu'il était amené à faire évoluer le graphisme de son site, le webmaster devait modifier chaque page HTML... qu'il devait ensuite télécharger sur un serveur distant pour qu'il puisse être lu par un navigateur Web Fig.3.

Grâce aux CMS, ce temps est révolu : l'arrivée des CMS, il y a environ 15 ans, a rendu possible dans un premier temps la séparation du conteneur (la page) et du contenu (l'article) via une interface, appelée Back-Office. Le contenu saisi – obligatoirement en HTML à l'époque – s'insérait par la suite dans un gabarit de page prédéfini, dont le rendu visuel était toujours généré par le serveur, s'affranchissant ainsi des contraintes précédemment évoquées.

Par la suite ces mêmes CMS ont bénéficié de nouvelles fonctionnalités simplifiant grandement le travail des contributeurs : mise en place d'un éditeur WYSIWIG pour saisir du texte, séparation des rôles et des droits permettant une décentralisation des tâches au niveau du Back-Office, création de workflow de validation des contenus ...

Ce qui est intéressant de retenir de ce petit rappel historique, c'est que les CMS et plus généralement les langages de programmation comme le PHP ont permis de propulser nos sites dans une nouvelle ère, facilitant d'une part le travail des équipes éditoriales et techniques mais aussi d'autre part ouvrant de nouvelles perspectives pour nos sites Web avec l'apparition de la vidéo, mais aussi de nouveaux usages (RSE, e-commerce...).

Disons-le clairement : les CMS ont « ringardisé » nos vieux outils de création de sites Web « classiques », mais se sont reposés pour beaucoup sur leurs acquis en valorisant un nombre de plus en plus conséquent de modules sans forcément investir dans des contrées plus sinueuses (outils Marketing, Expérience User-centric...).

Dries reconnaît lui-même que Drupal 8 a mis le paquet sur « l'expérience développeur » au détriment de « l'expérience utilisateur » (UX). Cette confidence est d'autant plus étonnante qu'il existe aujourd'hui sur le marché des frameworks qui répondent à ces 2 aspects.

CMS versus Framework : la nouvelle bataille ?

En clair, est-ce que les choix stratégiques de Drupal sont les bons ? L'arrivé des frameworks Web répondant à la fois à des considérations techniques et comportementales n'ont-ils pas eu raison de nos bons vieux CMS ? Sont-ils amenés à disparaître ?

Fig.2

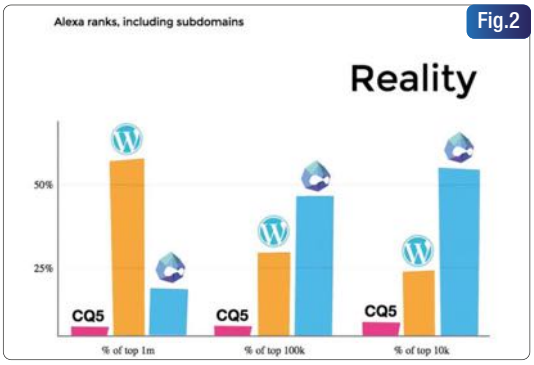
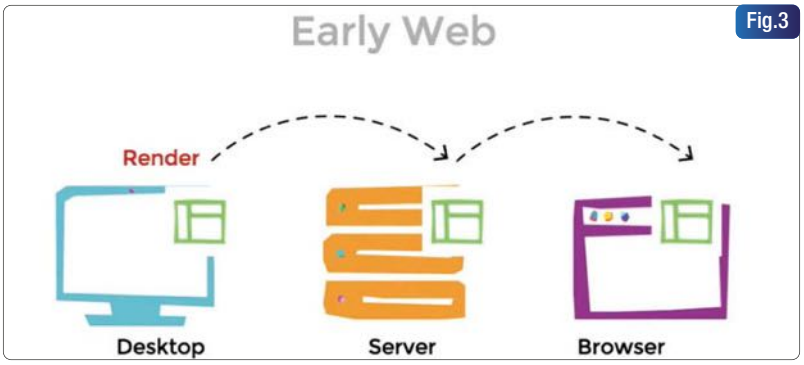


Fig.3



Avant de répondre à cette question, rappelons tout d'abord qu'un framework est une bibliothèque logicielle intégrant un ensemble de fonctionnalités préexistantes permettant de faciliter et d'accélérer le travail des développeurs. Il existe sur le marché différents types de frameworks applicables à différents usages (application Web, logiciel, application mobile, orienté gestion de contenus, orientés design...).

Aujourd'hui, de nombreux frameworks peuvent, en fonction des modules qui sont installés, se réorienter vers différentes activités informatiques ou usages... à la manière de solutions CMS, qui, par des modules ou des distributions spécifiques, peuvent répondre elles aussi à différents types d'usages.

Ces frameworks disposent de librairies ou de composants pré-utilisables, permettant de travailler dans un « cadre de travail », tout comme Drupal le fait aussi via la disposition de modules permettant d'accélérer les développements. C'est d'ailleurs pour cette raison que beaucoup considèrent Drupal comme un CMF et non comme un CMS, car son ambition va au-delà de la gestion de contenus classique chère à certaines autres solutions.

Il y a donc des similitudes entre les frameworks et Drupal, même si on a affaire à deux approches différentes :

- D'un côté, une architecture traditionnelle dite « monolithique » où le front-end est partie intégrante du CMS,
- De l'autre la mise en place d'une architecture découplée dite aussi « sans tête » où le front-end est décorrélé du CMS.

L'architecture découplée a plusieurs avantages :

- Elle permet tout d'abord, la mise en place de ce qu'on appelle l' « optimistic Feedback » (une réponse apparaît avant que le serveur traite la requête d'un utilisateur). Il n'y a plus d'appel serveur à chaque requête formulée par l'utilisateur en front et donc pas / plus de temps d'attente à l'affichage d'une information pourvue qu'elle soit stockée comme composant de la page.
- Elle rend réalisable la mise en place d'interfaces utilisateur non-bloquantes (l'utilisateur peut continuer à interagir avec l'application tandis que des portions de pages sont toujours en cours de chargement). C'est notamment ce que propose Facebook lorsque vous visionnez une vidéo pendant que la page continue à se charger.
- Elle facilite la mise en place d'applications web légères et globalement très ludiques, comme c'est par exemple le cas de Trello, un outil de gestion de projet en ligne qu'il est conseillé d'aller voir.

Au-delà de la prouesse technique, il est surtout intéressant de constater que ces « fonctionnalités » permettent de répondre parfaitement à des problématiques UX (expérience utilisateur) et UI (Interface utilisateurs). Certains comportements UX et UI comme le menu « burger », « le scroll infini », la possibilité de trier des résultats de recherche sans recharger la page restent possibles dans Drupal mais nécessitent du code là où les frameworks Front-end disposent déjà d'une librairie / d'un composant pré-utilisable.

L'approche découplée a cependant quelques inconvénients. Cela concerne surtout la perte des fonctionnalités majeures intégrées à Drupal et qui justifie finalement l''utilisation d'une solution CMS : l'absence de la barre d'outil éditoriale, la disparition de la gestion des contenus « in-line » (avec Quick Edit) ou de la mise en page (avec Panels), la prévisualisation, le RDFa (syntaxe permettant d'ajouter des données structurées dans une page HTML ou document XML) ...

L'une des problématiques que nous avons précédemment abordée et qui constitue elle aussi un critère d'UX est le temps de chargement des pages. Par son approche « client Side Apps », il s'agit d'un avantage non négligeable pour les frameworks alors que la « gestion du cache » a toujours été un point faible de Drupal 6 et dans une moindre mesure Drupal 7 sur la partie connectée.

Actuellement, Drupal 8 est la seule solution CMS compatible avec BigPipe.

Cette solution offre des performances assez étonnantes comme le démontre la vidéo postée sur https://youtu.be/JwzX0Qv6u3A

Même si les deux approches sont différentes, et que, disons-le clairement, Drupal 8 est à la traîne sur cet aspect, il est toujours réconfortant de constater que Dries a conscience de cette lacune. Pour y remédier, plusieurs solutions sont actuellement en chantier : utilisations d'une API unique limitant ainsi les requêtes serveurs, mais aussi de GraphQL, un langage de requête de données pour décrire des requêtes complexes dont une présentation complète est disponible ici : https://www.youtube.com/watch?v=p3gMSnaZrp8

Dries estime que, Drupal 8 couplé à GraphQL est LA solution idéale pour la construction d'applications et de sites Web. Elle constitue une réponse sérieuse à ce que propose les frameworks comme il le souligne en conclusion de sa démonstration.

Faut-il considérer ces Frameworks comme une menace ?

C'est un fait, l'arrivée des frameworks Front-End, appuyés par de grands acteurs du Web (Google pour AngularJS, Facebook et Instagram pour React.js) a chamboulé un certain nombre de nos habitudes. Les frameworks sont désormais une alternative sérieuse à nos outils actuels. Ont-ils signé pour autant la mise à mort de Drupal et de manière plus générale la fin des solutions CMS ?

Pour Dries, la réponse est « NON » d'autant plus que Drupal, contrairement à d'autres solutions CMS n'est pas fermé à la « complémentarité » éventuelle de ces solutions.

Son API complètement redéfinie en D8 permet de fournir aux frameworks utilisés en Front, les données exactes attendues. Drupal peut alors servir d'interface de gestion et de stockage de l'ensemble des données qui sont par la suite alimentés via une API sur n'importe quelle autre solution permettant cet interfaçage, frameworks compris.

C'est d'ailleurs ce qu'a démontré quelques jours plus tard John Ennews, en expliquant dans le cadre d'une des sessions comment il est possible de créer un site utilisant Drupal au niveau Back-end et AngularJS au niveau Front-end. Cette session, captée en vidéo est disponible sur https://www.youtube.com/watch?v=WkzrPMD9SqY et fera l'objet d'un futur article sur le blog de Core-Techs. Cette même technologie est d'ailleurs retenue pour tous les projets de refonte des sites de Radio France.

Il n'y a donc pas de crainte à avoir sur l'éventuelle pérennité de la solution Drupal 8 qui, comme nous venons de le dire est ouvert à plusieurs approches.

Quel futur pour Drupal ?

Il reste à combiner pour le futur de Drupal le meilleur de ces deux approches, celles que Dries définit comme étant la 3e approche, « l'architecture progressive », à mi-chemin entre l'approche traditionnelle et moderne et sur laquelle Drupal 8 se base mais qui nécessite encore de nombreuses améliorations. Cette vision est détaillée sur son blog : http://buytaert.net/the-future-of-decoupled-drupal) **Fig.4.**

Fig.4

Pour Dries, c'est cette 3e approche, couplée à celle de GraphQL qui font de Drupal le principal compétiteur face à des solutions CMS, innovantes ou frameworks MVC.

L'avenir nous dira s'il a été une fois de plus visionnaire... ⊞

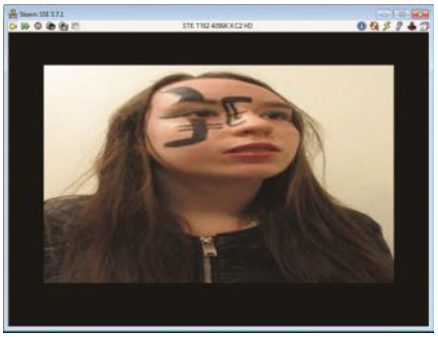
Suite de la page 35

appelé **maxYMiser**. Cet outil créé par *Gareth Morris* (Gwem) permet de gérer des patterns et de créer des sons spécifiques au composant Yamaha.
Aujourd'hui toutes les musiques provenant de l'Atari ST sont archivées principalement sur le site http://sndh.atari.org/, maintenu pour le fameux *Phil Graham* (Grazey), les fichiers sont au format unique **SDNH** qui a été inventé par *Jochen Knaus* et qui peut être joué via le logiciel **JAM** du groupe Cream pouvant être écouté sur PC comme sur ST.

Voici un bout de code en assembleur 68k du programme **Plugin-based multipalette picture viewer** (MPP VIEWER) développé par *Francois Galéa* (Zerkman) du groupe Sector One. Ce programme permet d'afficher des images sur un Atari STF et sur un Atari STE où les performances de celui-ci rajoutent plus de nuances au niveau des couleurs avec en plus une utilisation du Blitter permettant le mode overscan pour dépasser le format traditionnel d'affichage. Le format des images est en 320 par 200 pour un affichage de 56 couleurs par ligne, soit plus de 10000 couleurs affichées sur un écran standard !

```
;--------------------------------------------------------------------
;  Multipalette routine.
;  by Zerkman / Sector One
;  mode 3: 416x273, CPU based, displays 48+6 colors per scanline
;       with overscan and non-uniform repartition of color changes.
;--------------------------------------------------------------------
; Init routine.
; a0: file structure address
; a1: destination palette
m3_init:  move.b#2,$ffff820a.w ; 50Hz
   clr.b   $ffff8260.w     ; Low Resolution
   rts

m3_timera1:
   move #$2100,sr
   stop   #$2100
   move #$2700,sr
; top border HBL=33, LineCycles=452~460
   naupe    90
   clr.b   $ffff820a.w
   naupe    11
   move.b   #2,$ffff820a.w
   move.l   a7,usp
m3_tstsync0:
   move.b   $ffff8209.w,d0
   beq.s m3_tstsync0
   neg.b d0
   lsr.l    d0,d0
   move.l   m3_pal(pc),a0
   lea$ffff8240.w,a1
   lea$ffff820a.w,a2
   lea$ffff8260.w,a3
   lea$ffff824c.w,a7
   naupe    36
   movem.l (a0)+,d1-d5
   movem.l d1-d5,(a7)
   rept   227
   move a3,(a3)       ; LineCycles = 508 -> 4
   nop
   move.b   d0,(a3)       ; LineCycles = 8 -> 16
   movem.l (a0)+,d2-d7/a4-a5   ; 19
   movem.l d2-d7/a4-a5,(a1) ; 18
   movem.l (a0)+,d0-d7/a4-a6   ; 25
   movem.l d0-d7,(a1)      ; 18
   movem.l a4-a6,(a1)      ; 8
   moveq    #0,d0     ; LineCycles = 368 -> 372
   move.b   d0,(a2)       ; LineCycles = 372 -> 380
   move a2,(a2)       ; LineCycles = 380 -> 388
   movem.l (a0)+,d1-d5    ; 13
   move a3,(a3)       ; LineCycles = 440 -> 448
   nop
   move.b   d0,(a3)       ; LineCycles = 452 -> 460
   movem.l d1-d5,(a7)  ; 12
   endr
   move a3,(a3)       ; LineCycles = 508 -> 4
   nop
   move.b   d0,(a3)       ; LineCycles = 8 -> 16
   movem.l (a0)+,d2-d7/a4-a5   ; 19
   movem.l d2-d7/a4-a5,(a1) ; 18
   movem.l (a0)+,d0-d7/a4-a6   ; 25
   movem.l d0-d7,(a1)      ; 18
   movem.l a4-a6,(a1)      ; 8
   moveq    #0,d0     ; LineCycles = 368 -> 372
   move.b   d0,(a2)       ; LineCycles = 372 -> 380
   move a2,(a2)       ; LineCycles = 380 -> 388
   movem.l (a0)+,d1-d5    ; 13
   move a3,(a3)       ; LineCycles = 440 -> 448
   nop
   move.b   d0,(a3)       ; LineCycles = 452 -> 460
   move.b   d0,(a2)
   movem.l d1-d3,(a7)  ; 8
   move a2,(a2)
   rept   44
   move a3,(a3)       ; LineCycles = 508 -> 4
   nop
   move.b   d0,(a3)       ; LineCycles = 8 -> 16
   movem.l (a0)+,d2-d7/a4-a5   ; 19
   movem.l d2-d7/a4-a5,(a1) ; 18
   movem.l (a0)+,d0-d7/a4-a6   ; 25
   movem.l d0-d7,(a1)      ; 18
   movem.l a4-a6,(a1)      ; 8
   moveq    #0,d0     ; LineCycles = 368 -> 372
   move.b   d0,(a2)       ; LineCycles = 372 -> 380
   move a2,(a2)       ; LineCycles = 380 -> 388
   movem.l (a0)+,d1-d5    ; 13
   move a3,(a3)       ; LineCycles = 440 -> 448
   nop
   move.b   d0,(a3)       ; LineCycles = 452 -> 460
   movem.l d1-d5,(a7)  ; 12
   endr
   move a3,(a3)       ; LineCycles = 508 -> 4
   nop
   move.b   d0,(a3)       ; LineCycles = 8 -> 16
   movem.l (a0)+,d2-d7/a4-a5   ; 19
   movem.l d2-d7/a4-a5,(a1) ; 18
   movem.l (a0)+,d0-d7/a4-a6   ; 25
   movem.l d0-d7,(a1)      ; 18
   movem.l a4-a6,(a1)      ; 8
   moveq    #0,d0     ; LineCycles = 368 -> 372
   move.b   d0,(a2)       ; LineCycles = 372 -> 380
   move a2,(a2)       ; LineCycles = 380 -> 388
   movem.l (a0)+,d1-d5    ; 13
   move a3,(a3)       ; LineCycles = 440 -> 448
   nop
   move.b   d0,(a3)       ; LineCycles = 452 -> 460
   rept   5
   clr.l    (a1)+
   endr<<<<<<<
   move.l   usp,a7
   move #$2300,sr
   rts
```

Le source sera principalement utilisé dans la démo Antiques des groupes Dune et Sector One sortie en 2011 où tout le talent du graphiste Michel Savariradjalou (Mic) du groupe Dune est impressionnant dans cette démo.

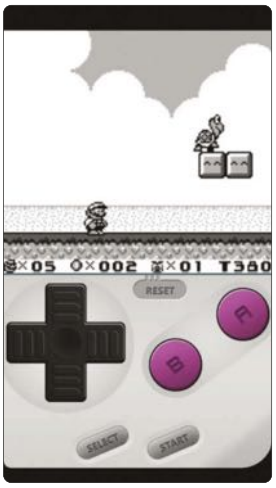


Juliane : Generation Ahead

Conclusion

La communauté Atari est toujours vivace grâce aux différents projets qui l'entourent : d'un côté le portage de l'émulateur **Hataroid** sous Android ou l'émulateur JavaScript **EstyJS,** ou encore le portage de jeux Atari sur Smartphone comme les jeux **Another World** d'Éric Chahi ou **North & South** d'Infograme. Il existe un clone de l'Atari ST : **MIST** fabriqué par *Lotharek* qui permet d'émuler un Atari ST mais reste encore en cours de développement car les innombrables contraintes techniques utilisées dans les jeux et les logiciels ne permettent pas d'émuler un Atari ST à 100%.
N'hésitez-pas à trouver toutes les informations et l'actualité de la communauté Atari sur http://www.atari-forum.com/ ou j'officie comme administrateur.

Introduction à l'émulation Gameboy en C#



En 1989 sort une console qui va révolutionner toute l'industrie du jeu vidéo : La Gameboy. Rapidement devenue culte par la qualité de ses jeux et ses nombreuses déclinaisons c'est aussi l'une des premières consoles à être émulée sur PC. Je vous propose dans cet article de découvrir quelques clés qui vous permettront de comprendre comment fonctionne cette merveilleuse petite console et peut-être vous donner envie de créer votre propre émulateur Gameboy.

NB : *J'ai simplifié volontairement certains aspects de l'architecture afin qu'elle soit plus facilement compréhensible par le lecteur.*

Samuel Blanchard
Responsable Développement pour NavisoDev
MVP Windows App Dev

Architecture simplifiée de la Gameboy

La Gameboy est constituée de trois éléments majeurs :

- De la mémoire (RAM et ROM) : incluse dans la Gameboy et sous forme de cartouche de jeu à insérer.
- Un CPU : un processeur 8bits de type Z80 modifié.
- Un GPU : un coprocesseur vidéo capable de gérer un fond (avec scrolling câblé), des sprites et une fenêtre.

Plan d'adressage

Ces éléments sont soudés sur une carte mère et câblés afin de pouvoir discuter entre eux. Ces câblages sont appelés des bus. A l'aide de ces bus, le CPU va pouvoir lire et écrire dans la mémoire et dans le processeur vidéo.
Il existe 3 types de bus :

- Un bus d'adressage permettant de positionner une adresse.
- Un bus de données (data) capable de lire ou d'écrire à cette adresse.
- Un bus de contrôle capable de donner un sens au bus de données (écriture ou lecture) mais également de lever des interruptions (le CPU exécute alors du code à une adresse particulière).

Plan d'adressage

Grâce aux bus, le CPU est capable d'accéder à des emplacements qui ressemblent à un tableau d'octets. Chaque octet est accessible par une adresse de 16 bits soit 65536 emplacements accessibles de 0x0000 à 0xFFFF en hexadécimal.
Lorsqu'une cartouche de jeu est insérée dans la Gameboy, celle-ci accessible dans le plan d'adressage.
Selon les adresses on accédera donc à des emplacements de la cartouche, de la Gameboy ou du coprocesseur vidéo.

Plan Adressage		
0x0000	Rom Bank #0 16 KO	Cartouche
0x4000	Rom Banks 16 KO	Cartouche
0x8000	Vidéo Ram	Gameboy
0xA000	Ram Banks	Cartouche
0xC000	Internal Ram	Gameboy
0xFF00	I/O ports	Gameboy

La ROM de la Cartouche contient le code et les données du jeu (de 0x0000 à 0x8000). La RAM vidéo contient les données qui seront affichées à l'écran (de 0x8000 à 0xA000). Ces données ne sont pas accessibles par le CPU en permanence car le GPU les utilise pour afficher l'écran de la Gameboy. Le GPU prévient le CPU que l'accès de la RAM vidéo est possible en générant une interruption par le biais du bus de contrôle (IRQ HBL & VBL). Ces interruptions seront détaillées plus loin.
La cartouche peut contenir sa propre RAM (0xA000-0xC000) afin de pallier la quantité de RAM limitée de la Gameboy (0xC000-0xFF00). Notez que cette RAM peut être sauvegardée par le biais d'une batterie (comme dans le jeu Zelda par exemple).
Les adresses 0xFF00 à 0xFFFF contiennent des accès à des registres externes. Ainsi l'adresse FF43 permet de contrôler la position horizontale du scrolling du GPU.

La cartouche Gameboy

Une cartouche est composée d'une ROM et éventuellement d'une RAM.
La ROM contient les informations du jeu (code, graphisme et son) mais également une description d'elle-même (son nom interne, sa capacité en terme de ROM et de RAM) et un header comprenant le logo Nintendo (sous forme de pixel). Au lancement de la Gameboy, ce header sera vérifié et affiché à l'écran. Cette vérification ne sera pas prise en compte par notre émulateur.
Le code contenu dans la ROM de la cartouche démarre à l'adresse 0x0100.
Les cartouches sont utilisées par les émulateurs sous forme de fichier contenant la ROM complète du jeu.

Emulation du plan d'adressage

Le plan d'adressage se comporte comme un tableau d'octets de 16 bits. On va donc instancier un tableau de 65 Ko puis charger le fichier de la ROM dans ce tableau à l'adresse 0x0000 à 0x8000.
Mais attention, certaines cellules de ce tableau ne seront pas modifiables, notamment la ROM. De même, la lecture de certains registres de 0xFF00 – 0xFFFF devra être relayée par des puces externes. C'est le cas du registre 0xFF43 du scrolling vidéo qui devra être relayé par la puce vidéo.
Voici un exemple de code permettant d'émuler une lecture de données 8 bits du plan d'adressage :

```
public byte Lecture8Bits(int adresse)
{
   switch( adresse)
   {
     case 0xFF43 :
```

```
        // Lecture du registre de scrolling de l'écran
        // en X câblé sur le processeur Vidéo
        return this.Video.ScrollX;
    }
    // tableau de 65ko contenant la ROM
     return this.MemoireInterne[ adresse ];
}
```

Il sera également utile de créer des méthodes d'écriture de données 8 bits et lecture/écriture de données 16 bits.

Le CPU : un cerveau pour la Gameboy

Maintenant que l'on sait lire et écrire des données en provenance de la mémoire et du GPU, on peut commencer à s'intéresser au chargement de ces données et à leur manipulation.

A cette fin, le CPU utilise des registres internes qui se manipulent comme des variables mais sont extrêmement plus rapides qu'un accès mémoire classique car intégré au CPU.

Les registres 8 bits permettent de manipuler des données tandis que les registres 16 bits des adresses.

Les registres 16 bits sont des concaténations de registres 8 bits, excepté pour les registres SP et PC. Ainsi les registres 8 bits B et C forment le registre 16 bits BC.

Registres		
8 Bits		16 Bits
A	F	AF
B	C	BC
D	E	DE
H	L	HL
		SP
		PC

Le CPU est capable d'exécuter des opérations à l'aide de ces registres.

Le registre F (Flag) est un peu particulier puisqu'il sert de stockage pour des bits contenant des informations sur la dernière opération effectuée par le processeur. Par exemple si le résultat d'une opération est différent de 0, le bit C du registre Flag sera fixé à 1. En testant les différents flags disponibles de ce registre, on est capable d'effectuer des branchements conditionnels (des if-then-else).

Le registre PC (Program Counter) permet au CPU de connaître l'adresse où le code doit être exécuté.

Le registre SP (Stack Pointer) contient une adresse (un pointeur) vers de la mémoire permettant de mettre en place une pile. Cette pile est importante pour le développeur de jeu, notamment pour stocker les paramètres d'une méthode que l'on souhaite appeler, ou pour empiler une adresse permettant à la méthode de retourner à son point d'appel.

Exécution d'une opération par le CPU

Lorsque la Gameboy démarre le registre PC du CPU est fixé à 0x0100. C'est la première adresse exécutée par celui-ci.

Dans l'exemple on voit que la lecture de l'adresse 0x0100 renvoie la donnée 0x00. Cette donnée est en fait un index correspondant à une opération à effectuer. Ces opérations sont appelées code machine et sont écrites généralement à l'aide du langage Assembleur. L'opération représentée par 0x00 est un NOP c'est-à-dire « No-Opération ».

Cette instruction n'effectue pas d'opération particulière à part incrémenter la valeur du Registre PC de sa taille (soit un).

Une fois le NOP exécuté le CPU est prêt à décoder une nouvelle instruction. L'adresse 0x0101 est lue. L'opération à exécuter est, cette fois, 0xC3 ce qui correspond à un JP – « Jump ».

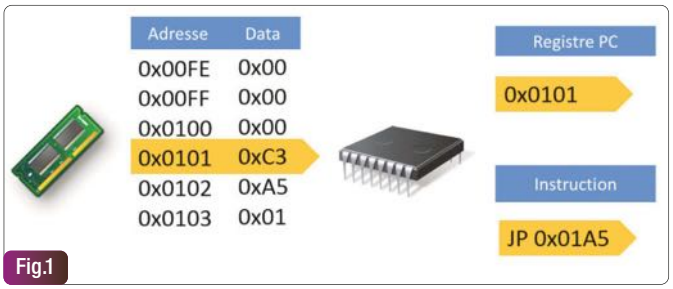


Fig.1

Le CPU va effectuer un saut à la manière d'un GOTO en Basic. Le CPU sait que l'instruction 0xC3 nécessite la lecture de l'adresse de saut, il va donc lire les deux octets suivants (0xA5 et 0x01) afin de constituer cette adresse 16 bits (0x01A5). Il ne reste plus, pour le CPU, qu'à modifier son registre PC avec la nouvelle adresse.

La prochaine opération à exécuter par le CPU sera donc à l'adresse 0x01A5 Fig.1.

Quelques instructions

Le CPU est capable d'exécuter des centaines d'instructions différentes que nous ne pouvons aborder dans le cadre de cet article.

Néanmoins vous trouverez dans ce tableau quelques instructions classiques du Z80.

Instructions			
Nom	Paramètres	Opcodes	Cycles
LD	A, #	0x3E	4
CP	A, C	0xB9	4
ADD	A, #	0xC6	8
JP	##	0xC3	12

L'instruction LD pour « Load » permet de charger une valeur dans un registre. Dans le cas de l'instruction 0x3E, soit par exemple LD A,#42, le registre A sera rempli avec la valeur immédiate #42 (écrite en en dur dans le code à la suite de 0x3E).

Notez que l'exécution de cette instruction durera 4 cycles soit 4 oscillations du quartz de la Gameboy. Le quartz permet de donner la cadence d'exécution du CPU afin qu'elle soit régulière. L'oscillation du quartz est également utilisée dans le cadre du GPU afin qu'il soit synchronisé au CPU.

L'instruction CP pour « Compare » permet d'effectuer une comparaison entre deux valeurs (registres) à l'aide du registre F dont nous avons parlé plus en amont. Elle prend 4 cycles.

L'opération ADD pour « Add » a pour fonction d'ajouter une valeur à une autre. Son résultat affecte le registre F. Cette instruction prend 8 cycles.

La dernière instruction est le saut que nous avons décortiqué ensemble dans le paragraphe précédent. Elle prend 12 cycles.

Emulation du CPU

Pour réaliser l'émulation de notre CPU, on va s'appuyer sur la méthode Lecture8bits créée précédemment. L'instruction Jump est prise comme exemple. Toutes les instructions du CPU devront être implémentées.

```
Public int RunOneInstruction()
{
Int cycle = 0;
byte opcode = this.Memoire.Lecture8Bits(this.PC);
Switch( opcode )
{
   // Jump
   case 0xc3 :
```

```
    this.PC = this.Memoire.Lecture16Bits(this.PC+1);
        cycle = 12;
    break;
    // Autres instructions
    case ...
}
Return cycle;
}
```

Le processeur Vidéo

Le processeur 8 bits utilisé par la Gameboy n'est pas assez puissant pour gérer l'affichage nécessaire à la réalisation d'un jeu vidéo. Il est donc épaulé par un processeur Vidéo ou GPU permettant l'affichage simultané d'un fond scrollable (déplaçable), de sprites et d'une fenêtre Fig.2.

Pour envoyer les graphismes du jeu à l'écran on utilise un emplacement mémoire particulier : La RAM vidéo (VRAM).

Afin que le CPU puisse modifier rapidement l'écran, celui-ci ne permet pas un accès direct par pixel.

Au lieu de cela il propose un système de tuiles et de cartes.

Une tuile (ou tile) est un bloc de 8x8 pixels.

Chaque tuile est rangée dans la VRAM à une adresse comprise entre 0x8000 et 0x9000. La carte (ou map) va récupérer l'index de ces tuiles et les afficher à l'écran Fig.3.

Cette utilisation de blocs de 8x8 pixels est particulièrement visible dans l'illustration du jeu Zelda sur les poteaux par exemple.

La carte est rangée dans la VRAM à une adresse comprise entre 0x9800 et 0xA000.

Concrètement un jeu chargera ses tuiles au démarrage du niveau, puis mettra en place sa carte. La carte sera modifiée régulièrement durant la partie alors que les tuiles le seront très rarement.

Un écran fait 160 par 144 pixels soit 20 tuiles par 18.

Les pixels de chaque tuile sont stockés dans un format particulier. Les tuiles sont capables d'afficher 4 couleurs simultanément constituées de 2 bits :

00 = blanc
01 = gris clair
10 = gris foncé
11 = noir

Les deux bits constituant une couleur sont séparés en deux octets comme dans l'illustration suivante. Les deux octets contiennent donc 8 couleurs simultanément Fig.4.

Le fond (Background)

Le fond d'écran est constitué 32x32 tuiles soit 256x256 pixels. Il est donc plus grand que l'écran.

Fig.2

Le fond est scrollable en X et en Y et boucle sur lui-même.

Un jeu nécessitant un scrolling horizontal infini à la Super Mario Land peut être développé en utilisant conjointement le registre de scrolling en X (adresse 0xFF43) du GPU et la mise à jour des tuiles aux endroits non visibles de l'écran pour constituer les nouveaux décors du jeu.

Les sprites

Les sprites sont des éléments graphiques dont la première couleur est transparente. Cela permet la création de personnages se détachant du fond. Les sprites sont constitués de tuiles et peuvent être inversés horizontalement ou verticalement afin de minimiser le nombre de tuiles employées. Deux tuiles peuvent être également associées pour créer un sprite ayant une hauteur double. Chaque sprite peut être positionné indépendamment.

La fenêtre (Window)

La fenêtre est souvent utilisée pour afficher des inventaires dans les jeux.

Elle est constituée de 20x18 tuiles soit la taille exacte de l'écran (160x144 pixels).

Cette fenêtre peut être déplacée à l'aide de deux registres du GPU représentant les positions X et Y.

Quelques registres du GPU

Les registres du GPU permettent de paramétrer le fonctionnement de la puce Vidéo.

Les registres suivants sont les plus communs :

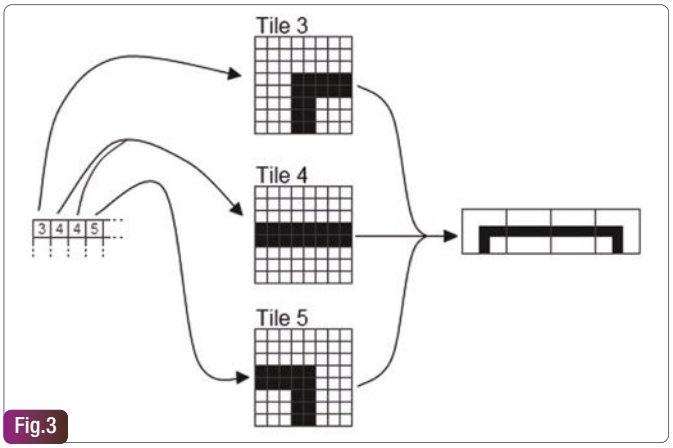


Fig.3

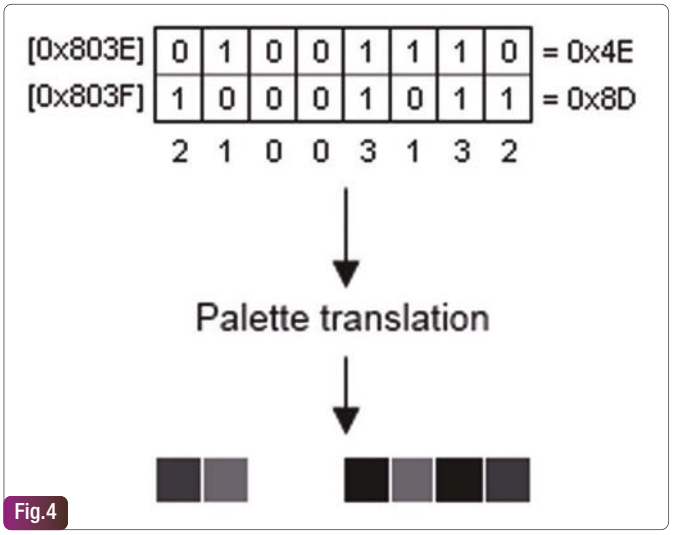


Fig.4

- LCDC : sélectionne l'affichage du fond, des sprites, de la fenêtre et de l'écran LCD complet.
- SCX/SCY : valeur du scrolling du fond
- WX/WY : position de la fenêtre
- BGP : palette fond et fenêtre (4 niveaux de gris)
- OBP0/OBP1 : 2 palettes pour les sprites (3 niveaux de gris car une couleur transparente)

Dessiner l'écran

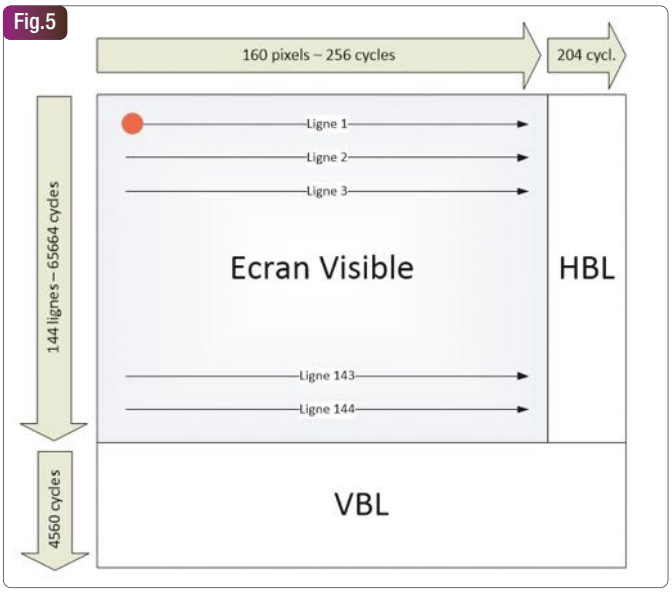
La VRAM est accessible à la fois par le GPU et le CPU mais jamais au même moment.
Lorsque le GPU dessine l'écran, le CPU ne peut y accéder.
C'est le GPU, par le biais d'interruptions (control bus) qui autorisera le CPU à accéder à la VRAM.
Le CPU sera alors stoppé dans son fonctionnement classique et sautera à l'adresse de l'interruption.
Deux interruptions sont utilisées par le GPU :

- La HBL : interruption horizontale pour chaque ligne de l'écran affichée.
- La VBL : interruption verticale lorsque l'écran est complétement affiché Fig.5.

L'émulation de cet écran s'effectuera en comptant les cycles CPU effectués. Au bout de 256 cycles, on lèvera l'interruption HBL au CPU qui changera son registre PC pour sauter à l'adresse du code HBL. Ce code ne peut excéder 204 cycles. A la fin de ces cycles le CPU retourne à son adresse originale et la ligne de l'écran est prête à être affichée par le GPU émulé.
L'émulation de cet affichage consistera à récupérer les valeurs de chaque pixel constituant le fond, les sprites et la fenêtre en prenant en compte les valeurs des registres du GPU.
// CountHBL = compteur de position de la HBL de 0 à 143 car l'écran fait 144 pixels de hauteur

```
protected void DrawScreenLine(int countHBL)
{
  int lcdc = this.memory.Lecture8Bits( REGISTER_LCDC );
  if( (lcdc &  VideoGB.MASK_LCDC_DISPLAY) == VideoGB.MASK_LCDC_DISPLAY )
  {
    this.DrawBackgroundLine( lcdc, countHBL );
    this.DrawWindowLine( lcdc, countHBL );
    this.DrawSpriteLine( lcdc, countHBL );
```

Fig.5




```
  }
}
```

La méthode DrawBackgroundLine, par exemple, sera chargée de récupérer l'emplacement visé par le compteur HBL dans la carte en intégrant les registres SCX et SCY dans son calcul. Une fois l'index de la tuile récupéré, on lit les octets contenus dans la tuile, et, en les associant par deux on est capable de retrouver la couleur des pixels à afficher (de 0 à 3). Il ne reste plus qu'à convertir cette valeur en couleur compréhensible par notre système (du RGB en l'occurrence).
Toutes ces valeurs de couleur sont stockées dans un tableau représentant la totalité de l'écran.
Une fois toutes les lignes dessinées (au bout de 65664 cycles), le GPU lève l'interruption VBL. Le CPU se positionne alors sur l'adresse du code VBL. Durant cette période, Le CPU a accès à la VRAM et en profite souvent pour modifier les valeurs de la carte et parfois même des tuiles. En effet, le temps d'exécution autorisé de 4560 cycles est beaucoup plus long que pour les HBLs.
A la fin de la VBL, nous possédons un écran complet du jeu.

Synchronisation de la Gameboy et affichage écran

Sur un PC actuel, notre Gameboy risque d'être émulée extrêmement rapidement. Si nous ne la synchronisons pas avec notre PC, elle risque d'être beaucoup trop rapide pour être jouable.
On va donc synchroniser la Gameboy et le PC en prenant un de leurs points commun : La VBL.
L'écran de notre PC dispose également d'une VBL.
Pour afficher nos pixels à l'écran, on utilise un framework graphique (par exemple SharpDx).
Les frameworks graphiques actuels proposent pratiquement toujours une méthode pour accéder à cette VBL.
Ils permettent également d'envoyer le tableau contenant les valeurs RGB dans une texture. Cette texture sera ensuite affichée à l'écran.
Maintenant que l'affichage écran est terminé on peut reprendre l'exécution de la Gameboy pour afficher un nouvel écran.

```
public byte[] RunOneFrame()
    {
      int cycles = 0;
      do
      {
   cycles += this.processor.Run();
 this.video.Run(cycles);
      }
      // 70800 cycles pour 1 frame
      while (this.Video.HaveNewFrame == false);
      return this.video.Screen;
    }
```

Conclusion

Il reste beaucoup de choses à mettre en place avant que notre Gameboy soit pleinement fonctionnelle (Joypad, Timer, DMA) mais globalement vous pouvez commencer à émuler vos premiers jeux.
Si vous désirez des informations complémentaires sur ce sujet, vous pouvez visiter le site de Imran Nazar dont sont tirées deux illustrations de cet article avec son aimable autorisation.
http://imrannazar.com/GameBoy-Emulation-in-JavaScript:-The-CPU
Un grand merci à Laurent pour sa relecture attentive.

11 outils qu'un développeur Web doit connaître

Un développeur doit avoir comme obsession de ne pas réinventer la roue à chaque fois. En cela se former aux différents Frameworks afin de s'économiser du code est indispensable. De plus, il est très utile de connaître les outils qui effectuent de manière graphique et rapide une série d'opérations que sont amenés à faire les développeurs Web. C'est l'objet de cet article. Il présente des outils que son auteur apprécie pour leur simplicité et le gain de temps qu'ils apportent. Ils sont souvent disponibles en ligne, ce qui ne nécessite pas l'installation d'applications sur son poste de travail. Tous ces produits sont gratuits, et leur prise en main est quasi immédiate, le principal obstacle à leur utilisation étant de savoir qu'ils existent.

Gautier Deruette,
ingénieur développement,
Osaxis

1 Open httpRequester : tester ses Web Services Fig.1.

Un des moyens les plus simples de tester ses Web services est de s'installer un plugin permettant de lancer des requêtes POST (pour des requêtes GET, le navigateur seul suffit). Il est alors facile de remplir le corps en JSON ou en XML, de modifier le content type et d'envoyer sa requête en POST ou en GET. On peut également modifier le header ou bien ajouter, de manière propre, des paramètres à la requête. Mais ce qui est surtout utile, c'est de pouvoir enregistrer ces requêtes de tests afin de ne pas reconstruire via des copier-coller hasardeux son jeu de tests entre chaque modification de code. Si l'on veut pousser plus loin, on utilisera SOAP UI qui a des fonctionnalités avancées de tests, d'appels à la chaîne, etc Fig.2.

2 JsonParser : construire un JSON en un clin d'œil

En lien avec le plugin précédent, qui envoie des requêtes dont le corps est souvent en JSON, il n'est pas rare alors de voir des développeurs compter du bout des lèvres leurs parenthèses, accolades et crochets, ouvrants et fermants, cherchant qui est coupable du JSON mal formé. Voici une application Web qui permet de construire et modifier des objets JSON. L'application va mettre en forme, colorer et indenter à la volée notre objet permettant ainsi une édition sereine du JSON. Il existe des équivalents pour le XML comme codebeautify.org/xmlviewer. Ils font le travail de manière aussi efficace et proposent souvent d'indenter tous types de code mais JsonParser, qui ne se concentre que sur le JSON, a le mérite d'être le plus épuré et de faire son indentation à la volée Fig.3.

3 "Dust me selector" : faire du tri dans ses classes CSS Fig.4.

Ceci est un plugin Firefox, qui en un rien de temps va vous permettre de faire un grand nettoyage dans vos fichiers CSS. Il peut scanner une page

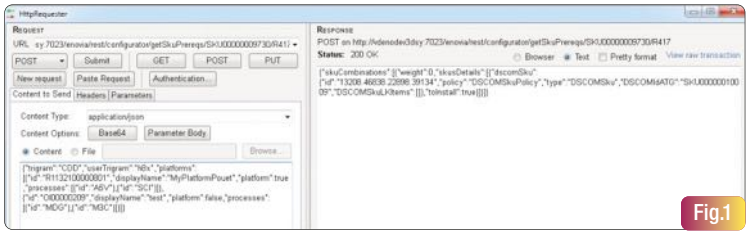


Fig.1

Interface de HttpRequester : simple et fonctionnelle

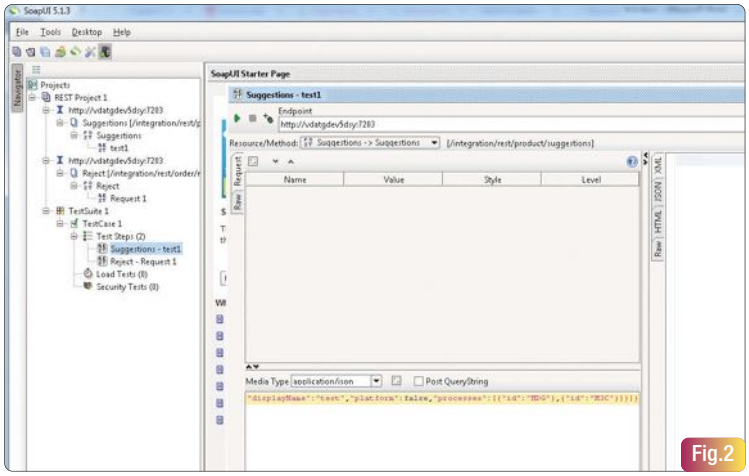


Fig.2

SOAP UI est un logiciel avancé d'appels de Web service

Fig.3

JsonParser indente et colorie le JSON en entrée. Ici le JSON est mal formé, il manque une accolade.

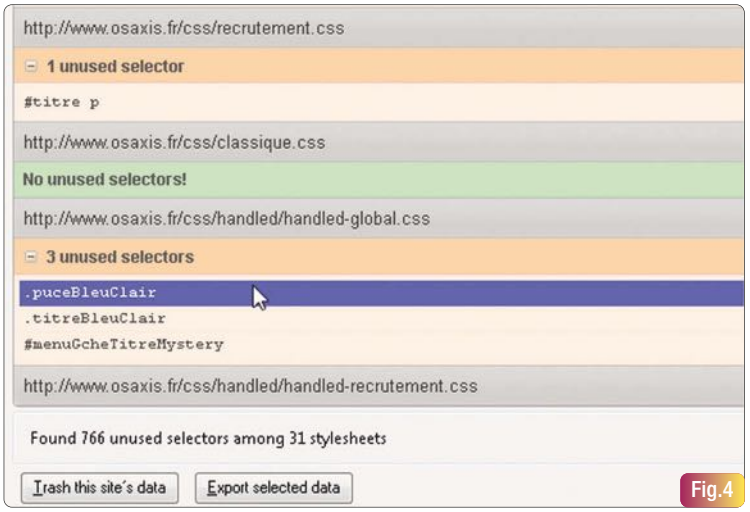


Fig.4

Liste des classes CSS inutilisées, classées par fichier de Dust me Selector

mais surtout un site entier, à la recherche de classes CSS inutilisées. Mais il ne s'arrête pas là : il va, à la demande, générer les fichiers CSS épurés de tout code inutile. Ceci est précieux aujourd'hui alors que chaque kilooctet est à considérer afin de diminuer le poids de nos pages Internet qui sont de plus en plus lues via un smartphone.

4 Quick accept language switcher : changer la langue du navigateur en un clic

Les sites sont de plus en plus internationaux et proposent donc de plus en plus différentes langues. Par défaut, il est donc préférable de choisir la langue du navigateur comme langue d'affichage quand l'utilisateur arrive pour la première fois. Lors de nos tests, ce plugin pour Firefox permet en un clic d'éditer la langue du navigateur. Encore un outil qui nous fait gagner du temps... Fig.5.

5 La boîte à couleurs : une pipette à couleurs sur le bureau Fig.6.

La boite à couleur est un outil qui n'a qu'une seule fonctionnalité, mais c'est incroyable ce qu'elle peut servir : lorsque vous souhaitez récupérer une couleur précise sur une image fournie par un designer, vous récupérez en un clic le code hexadécimal à ajouter dans vos classes CSS. Indispensable.

6 jsfiddle.net : écrire son code statique et tester en ligne

Cette application Web est tout simplement magique. Elle sert principalement à développer du JavaScript et à pouvoir valider les modifications directement dans le navigateur. Cela permet de s'abstraire de son site Internet, de tester une fonctionnalité avant de l'implémenter réellement. Cette application se compose de quatre cases : la première permet d'entrer du HTML, la seconde du CSS, la troisième du JavaScript et enfin la dernière permet de visualiser le résultat de ce code après un clic sur « run ». Il est bien sûr possible d'appeler des bibliothèques JavaScript et CSS externes. Une autre fonctionnalité extrêmement utile est celle de pouvoir enregistrer son code, que l'on peut alors retrouver ou partager sur les forums grâce à une URL générée Fig.7.

Fig.5

Aperçu du plugin de changement de langue de Firefox

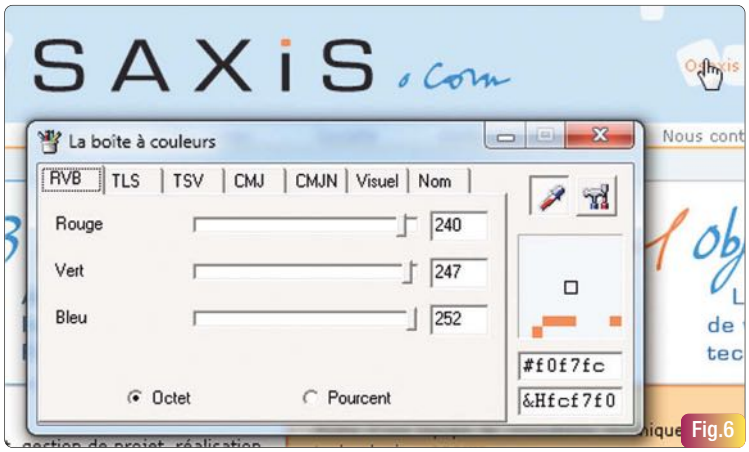


Fig.6

La boîte à couleurs, un outil mono fonctionnel, mais à connaître

7 refresh-sf.com : minimiser la taille de ses fichiers statiques

On ne le dira jamais assez : le mobile va dépasser l'ordinateur de bureau en termes de connexion Internet. Ce dernier, malgré l'arrivée du 4G, conserve souvent une mauvaise connexion. Il faut donc à tout prix réduire le poids de nos pages. Il existe heureusement une multitude de compresseurs, en ligne qui plus est.

Pour le CSS, ce type d'outil va rassembler les sélecteurs récurrents et supprimer certains éléments de syntaxe. Ceux-ci sont utiles à la lecture, mais inutiles au moteur de rendu.

Pour le JavaScript, certains éléments de syntaxe vont être supprimés également et les variables vont être renommées en des variables d'une lettre.

Dans les deux cas, les commentaires, espaces inutiles et retours à la ligne vont être supprimés. L'ensemble de ces traitements rend bien sûr le code illisible pour un développeur qui doit du coup conserver une version propre de son code pour les corrections, puis repasser son code dans la moulinette qui réduit le code.

Internet regorge d'outils en ligne gratuits pour effectuer ces traitements. Il vous est ici proposé refresh-sf.com, d'une part parce qu'il compacte le code aussi bien CSS que JavaScript (qu'il détecte automatiquement), d'autre part parce qu'il est aussi efficace voire un peu plus que les autres. Notons néanmoins que dans ce domaine, Google semble comme souvent surpasser tout le monde avec son outil réduisant le JavaScript : closure-compiler.appspot.com/home

Il est possible une fois le fichier minimisé de le copier dans le presse-papier ou bien de le télécharger. Il existe tout un tas d'options intéressantes à parcourir mais celles-ci sont configurées pour l'utilisateur dans 99% des cas Fig.8.

8 WinSCP : un client SFTP libre sous Windows

Certains développeurs ne jurent que par leur terminal de commande, d'autres préfèrent les interfaces graphiques agréables. WinSCP en est une permettant de se connecter à une Machine distante et d'y déposer ou d'y récupérer des fichiers et même de les modifier avec un éditeur simple (ce que ne peut pas faire le tout aussi connu FileZilla). L'interface est moderne et la prise en main évidente. Pensez à sauvegarder vos connexions pour ne pas retaper vos identifiants à chaque fois, l'objectif de ces outils étant toujours de gagner du temps Fig.9.

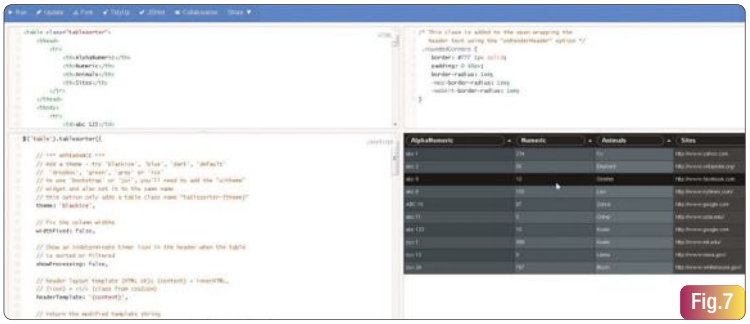


Fig.7

JSFIDDLE se décompose en quatre cadrans : HTML, CSS, JavaScript et rendu

Fig.8

Refresh-CSS est en train de transformer notre feuille CSS afin qu'elle prenne le moins de poids possible

9 www.askapache.com/online-tools/base64-image-converter : convertir des images en base 64

Les data Uri permettent de stocker des informations dans le champ plutôt que d'aller chercher la ressource à l'extérieur. Concrètement, dans une page HTML référençant des images, les balises « img » ont dans leur champ une image encodée en base 64 plutôt qu'une URI pointant vers une image externe. Cette méthode présente l'avantage de supprimer des requêtes. En revanche, une image encodée est généralement plus lourde que l'image elle-même. À chacun de décider s'il préfère privilégier le nombre de requêtes au poids de ses pages ou l'inverse.

```
<img
src="
akyH5BAQUAP8ALAAAHJHUGBQAA8AAAPb
WLrc/jDKSVe4OOvNu/9gqARDSRBHegyGMahqO4R0bQcjIQ8E4BMCQc930JluyGRmdAAcdiigMLVr
ApTYWy5FKM1IQe+Mp+L4rphz+qIOBerIUYeCY4p2tGrJZeH9y79mZsawFoaIRxF3JyiYxuHi
MGb5KTkpFvZj4ZbYeCiXa0iKBwnxh4fnt9e3ktgZyHhrChinONs3cFAShFF2JhvCZlG5uchYNun5eedRxM
AF15XEFRXgZWWdciuM8GCmdSQ84ILQfY5R14wDB5Lyon4ubwS7jx9NcV9/j5+g4JADs=
" />
<!-- exemple d'image en base 64 insérée dans le code HTML -->
```

Il est important de connaître au moins cette possibilité. Certains plugins JavaScript demandent pour être customisés de modifier des fichiers XML, et notamment pour modifier une image par défaut d'y inclure une image encodée en base 64. Askapache va convertir une image donnée en base 64, sachant qu'il est tout à fait capable de faire l'inverse Fig.10.

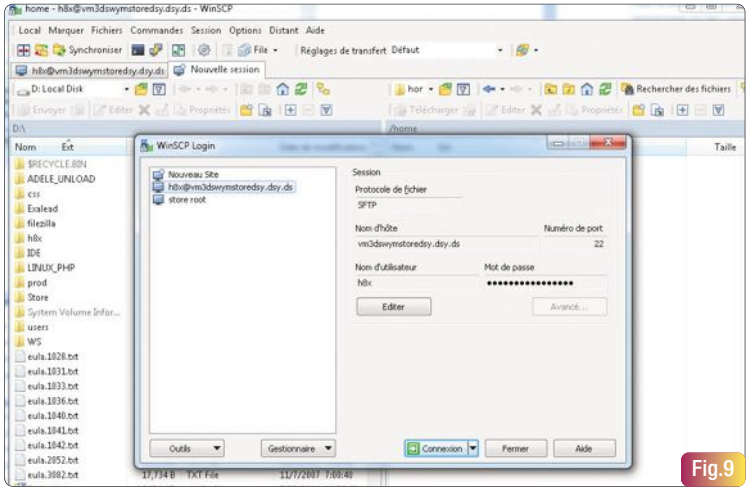


Fig.9

Interface de connexion de WinSCP

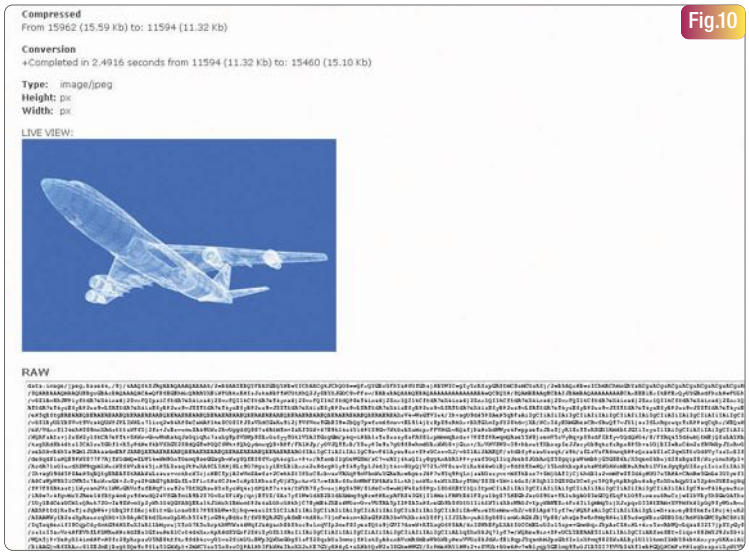


Fig.10

Exemple d'une image convertie avec askapache en base 64

10 Meld : comparer fichiers et dossiers

Il existe beaucoup d'éditeurs et de comparateurs de fichiers ou de dossiers. Ils ont chacun leurs avantages et on n'aime pas forcément en changer. C'est justement pour cela que nous vous proposons Meld. Celui-ci a la particularité d'être disponible sous Linux, Mac et depuis peu, sous Windows. Il possède des options sympathiques qui font que c'est un outil agréable à utiliser comme la coloration syntaxique et le changement de la tabulation pour y voir plus clair. Meld compare aussi bien les fichiers que les dossiers. Fig.11.

11 Xenu : vérifier les liens de son site

Il est de temps en temps utile de vérifier l'ensemble des liens de son site afin de prévenir les erreurs 404 (fichier non trouvé). Pour cela, Xenu est un logiciel très simple d'utilisation qui va rechercher tous les liens et tester toutes les requêtes que propose le site. Concrètement, Xenu va faire une requête GET sur chacun des liens dans les balises <a>, <link>, <img>, <script>... et ainsi tester la présence des fichiers. Au fur et à mesure qu'il avance, Xenu nous affiche les liens qu'il a testés et l'état de réussite du test. Ainsi en quelques instants, on peut dénicher les erreurs de frappes, et nettoyer un peu son code d'appels obsolètes Fig.12.

Conclusion

Cette liste est bien sûr incomplète. Vous avez, espérons-le, découvert quelques applications qui vous feront gagner un peu de temps. Bien sûr, pour chaque application présentée, il en existe toujours d'autres qui ont leurs avantages. Mais l'objectif est de garder en tête l'existence de ces applications, de connaître les possibilités qu'elles offrent et d'y penser au bon moment.

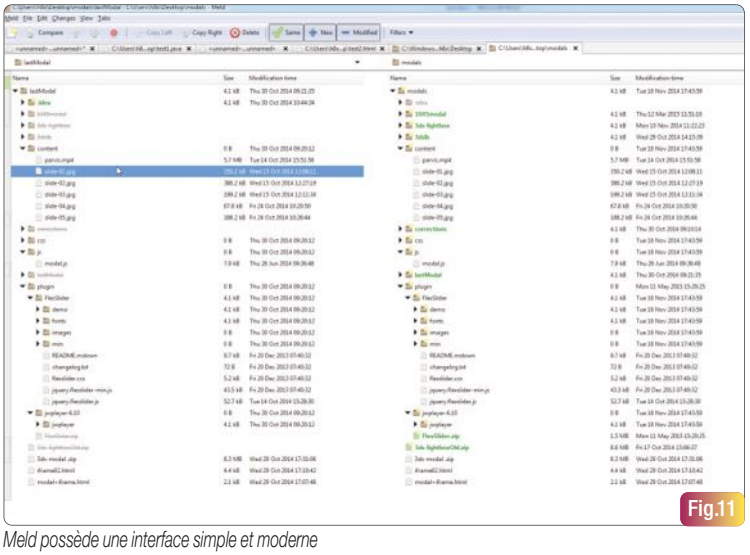


Fig.11

Meld possède une interface simple et moderne

Xenu nous remonte la liste des liens cassé qui apparaissent ici en rouge

Fig.12

Basculer son code vers Java 8 sur un projet à grande échelle

J'accompagne Voyages SNCF (VSCT) depuis plus de deux ans en tant qu'Architecte de suivi de production chez PALO IT. J'interviens notamment sur les problématiques liées à la mise en production de nouvelles versions des applications. Je vais vous faire part de mon retour d'expérience suite à ma participation au hackathon intitulé « Faire basculer VSCT vers Java 8 », ayant eu lieu en janvier 2015.

Nadhem LAMTI,
Architecte Java chez **PALO IT**

Expert Java chez PALO IT, Nadhem accompagne Voyages SNCF (VSCT) depuis bientôt 3 ans en tant qu'Architecte suivi production. Il intervient sur des incidents pouvant toucher des applications Web Front End & Mobile en production.

Alors que la version 5 de Java avait permis l'introduction de classes/types génériques et d'une nouvelle API, cette évolution avait en réalité eu peu d'impact sur la manière de développer, et le temps d'adaptation pour une équipe était négligeable. Java 8 n'est pas une simple nouveauté : c'est une véritable révolution ! En effet, c'est une logique toute entière de conception et de développement qui change, en introduisant principalement la logique de programmation fonctionnelle (expression Lambda) comme les « functional interfaces » et les nouvelles API de collection « Stream ».

Cette nouveauté remet en question toute une pratique de développement (de plus de 10 ans pour moi !) et si le passage à Java 8 peut s'effectuer sans difficultés pour certains (en particulier pour les jeunes développeurs fraîchement diplômés), il pourrait s'avérer plus complexe pour ceux qui n'auraient pas encore eu l'occasion de découvrir les principaux langages fonctionnels tes que « Scala » et/ou « Clojure ».

Contexte et enjeux du hackaton

Dans une démarche d'expérimentation, VSCT organise chaque année un hackathon auquel tous les collaborateurs peuvent participer. Une présélection des sujets de tous types est organisée par un jury, qui identifie ensuite les équipes participant à cet événement. J'ai alors proposé l'étude de la migration du code existant vers Java 8, en appliquant la nouvelle syntaxe des expressions Lambda sur certaines classes cibles.

Le site Voyages SNCF est une application critique de haute disponibilité, devant offrir une qualité et un délai de service irréprochables pour ses utilisateurs. Dans ce contexte, les améliorations vendues par Java 8 en termes de qualité de code et de performance étaient intéressantes. Cependant, le lancement d'un plan de migration sans stratégie préalable aurait pu s'avérer risqué, les équipes travaillant en mode Agile et les demandes étant gérées par ordre de priorité. Ajouter à cela un tel chantier aurait forcément nécessité un nombre d'homme/jour supplémentaire, sans compter le délai de montée en compétences, ainsi que le risque de répercussions négatives sur les performances de l'application si le code design de la nouvelle implémentation n'est pas soigné.

Sans oublier qu'il ne s'agit pas d'un développement « from scratch » : on parle bien de code existant qu'il faudrait reprendre en Java 8. Si ce dernier est complexe et/ou mal développé, le « Refactoring » en serait d'autant plus délicat. Mises à part les contraintes de délai, on risquerait de se retrouver avec des performances dégradées malgré les promesses vendues par Java 8 et, dans ce cas de figure, peu d'espoir que le sujet ait encore un grand intérêt.

Conscients des possibilités offertes par ce nouveau style de développement et prêts à relever le challenge, les développeurs des différentes équipes étaient déjà très motivés à l'idée de basculer vers Java 8. Ce ne sont pas les compétences qui manquent chez VSCT mais ce projet de migration ne remportait pas l'adhésion du Management, pour les raisons citées précédemment. Le hackathon représentait cependant une belle opportunité pour expérimenter cette idée et obtenir un premier résultat qui apporterait une meilleure visibilité quant à la priorisation de ce chantier de migration.

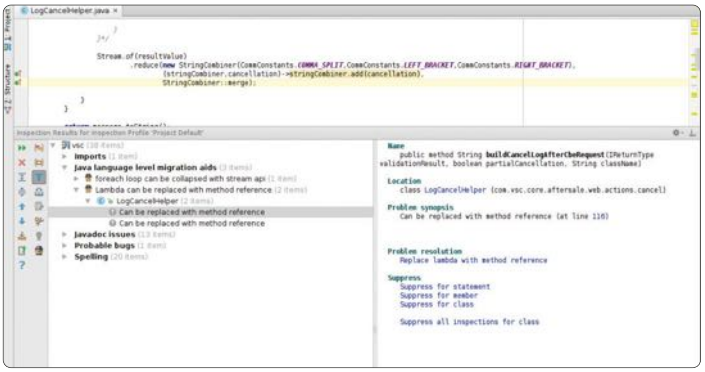
Dans le cadre de ce hackaton, nous avons opté pour le déroulement des étapes suivantes :

- Constitution de l'équipe ;
- Préparation de l'environnement de travail ;
- Transformation du code en Java 8 ;
- Exécution des tests unitaires ISO sans Java 8 ;
- Tests de performance.

Etape 1 : constitution de l'équipe par niveaux

Le hackathon permettrait en premier lieu la montée en compétences des équipes internes. Nous avons donc formé une équipe hétérogène, en nous basant sur nos différents niveaux de maîtrise de Java 8 :

- Moi-même au niveau avancé : dans le cadre de mon expérience personnelle, je rassemblais des connaissances depuis 5 mois grâce au déve-



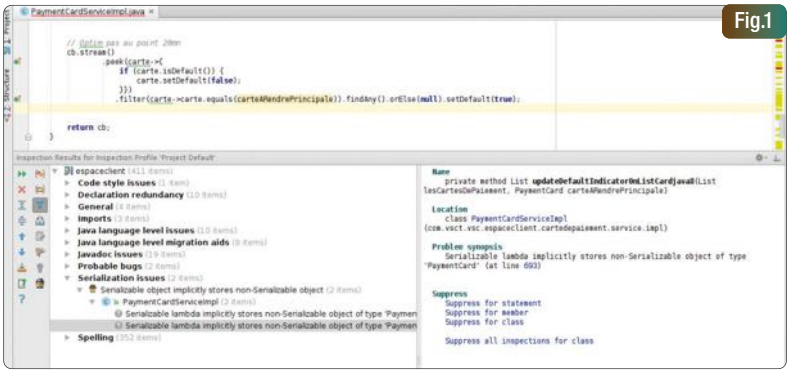


Fig.1

loppement de mini-exemples, la lecture d'ouvrages et d'articles sur les bonnes pratiques de développement, la participation aux événements Java 8, etc. ;

- Hichem au niveau moyen : il avait commencé à s'approprier les notions de base des nouveautés quelques jours avant;
- Dhia au niveau débutant : il n'avait aucune connaissance Java 8 préalable.

NB : *Pour rappel, les membres de l'équipe sont tous des Architectes de suivi de production et des Experts Java.*

Nous avons ensuite identifié des classes significatives, qui sont souvent appelées lors de l'exécution, et sur lesquelles nous pouvions appliquer les transformations issues des nouvelles API « Stream » ou « Collectors ». Nous avons fini par identifier 8 classes simples, 8 classes moyennes et 8 classes complexes réparties sur 3 projets distincts. Leur attribution s'est faite de manière aléatoire, ce qui nous a permis de dresser 3 tableaux que nous analyserons plus bas.

Etape 2 : migration et exécution de tests unitaires

Une fois les rôles définis et les classes identifiées, nous avons débuté la phase de migration :

- Préparation des environnements de travail : mise en place des projets en local + compilation Java 8 + configuration de l'IDE + création des projets sur Git [durée : 2H30] ;
- Présentation de Java 8 axée autour des grandes nouveautés et des best practices à prendre en compte [durée : 1H30] ;
- Début du développement : j'ai dédié 70% de mon temps au développement et 30% au support des autres membres de l'équipe. Chaque classe redéveloppée n'est considérée finie que si le Code Design est respecté (éviter les « Foreach » au maximum) et que les résultats des tests unitaires sont ISO code sans les expressions Lambda [durée : 1 jour] ;
- Pour assurer la qualité du code développé, nous avons utilisé les plugins suivants sur « IntelliJ» : « QAPlug», « FindBugs » et « CheckStyle », qui sont compatibles avec les nouveautés Java8. Ci-dessous quelques résultats d'analyses appliquées sur du code réécrit en Java 8 affichant des recommandations d'amélioration sur l'utilisation de l'API « Stream » : Fig.1.

Il n'était pas possible de mettre un point d'arrêt, ce qui est extrêmement contraignant pour deboguer le code.

Nous avons souhaité utiliser « SonarQube », mais pour des raisons de Timing, nous n'avions pas pu le mettre en place vu le coût d'installation et de configuration que ça pouvait engendrer. Le résultat de notre avancement est récapitulé dans les 3 tableaux ci-dessous. La ligne Equipe récapitule le pourcentage du nombre de méthodes migrées par rapport à la cible et le temps passé en moyenne par personne et par méthode transformée (tests unitaires inclus) : Tableau 1, 2 et 3.

Ci-dessous quelques exemples de bouts de code changés en Java 8 en utilisant les expressions Lambda en notant à chaque fois le temps passé et des remarques sur les difficultés trouvées :

		Classes simples		
	Niveau	Nom	Nbr méthodes migrées + test unitaires	Tps de dev
Nadhem	Avancé	PayCardServiceImpl	13/13	3h30
		SearchViewBeanHelper	2/2	30min
		ExposablePropertyHolder	1/1	20mn
Hichem	Moyen	CbeFormatterHelper	3/3	2h
		ValidBeforeCommand	1/1	30 min
		PaymentHelper	2/2	1h
Dhia	Débutant	SaveCommBasket	2/2	2h10
		FinalizationServiceImpl	2/2	3h
Equipe			100%	29min/méthode

Tableau 1 : « Statut avancement refactoring Java 8 des classes simples »

		Classes moyennes		
	Niveau	Nom	Nbr méthodes migrées + test unitaires	Tps de dev
Nadhem	Avancé	PrnHelper	0/3	0
		ASLogCancelHelper	0/3	0
		ASDeadlineBuilder	0/2	0
Hichem	Moyen	TravelViewBeanBuilder	0/2	0
		CustomerAccountHelper	0/3	0
		FrequentTravellerCardTypeServi	3/4	2h
Dhia	Débutant	TravelCardTypeImpl	0/1	0
		CancelActionHelper	0/2	0
Equipe			15%	40mn/méthode

Tableau 2 : « Statut avancement refactoring Java 8 des classes moyennes »

		Classes complexes		
	Niveau	Nom	Nbr méthodes migrées + test unitaires	Tps de dev
Nadhem	Avancé	LogCancelHelper	1/1	1h
		ProposalConverter	0/3	0
		CcoutputTransformer	0/2	0
Hichem	Moyen	AddSupplServiceHelper	0/2	0
		HpGetPrnService	0/4	0
		AqDayViewBeanBuilder	0/3	0
Dhia	Débutant	WishesHelper	0/2	0
		ExceptionInterceptor	0/3	0
Equipe			6%	1h/méthode

Tableau 3 : « Statut avancement refactoring Java 8 des classes complexes »

Exemple 1

```
public List<FrequentTravellerCardTypeEnum> getFrequentTravellerAdvantage() {
  final List<ProgrammeDeFidelite> fids = getProgrammesDeFidelite();

 /* Ancien Code

  final List<FrequentTravellerCardTypeEnum> result = new          ArrayList<FrequentTraveller
CardTypeEnum>(fids.size());
  for (final ProgrammeDeFidelite f : fids) {
    final Integer id = Integer.valueOf(f.getCode());
    if (FrequentTravellerCardTypeEnum.getById(id) != null) {
      result.add(FrequentTravellerCardTypeEnum.getById(id));
    } else {
      logCarteInconnu(f)
    }
  }
  return result;
  */

  // nouveau code Java8

  // 20 mn
   return fids.stream()
                .mapToInt(fid -> Integer.valueOf(fid.getCode()))
                .mapToObj(FrequentTravellerCardTypeEnum::getById)
                .peek(e-> {
            if (e.getCode()) logCarteInconnu(e);
          })
                .filter(e -> e != null);
}
```

Exemple 2

```
/* Ancien code

PaymentCard carteARendrePrincipale = null;
for (final PaymentCard carte : lesCartesDePaiement) {
  if (!carte.isExpired()
      && !carte.isDefault()
      && (carteARendrePrincipale == null || carteARendrePrincipale.getCreationDate().before(
```

```
        carte.getCreationDate())) {
    carteARendrePrincipale = carte;
  } */

// java8 20 mn
PaymentCard carteARendrePrincipale = lesCartesDePaiement.stream()
    .filter(carte-> !carte.isExpired() && !carte.isDefault())
    .min(Comparator.comparing(carte->carte.getCreationDate().getTime()))
    .orElse(null);
```

Exemple 3

```
/*public ReturnEnumeratedType execute(final IContext vscContext, final FinalizationService
ValueObject vo,final boolean moduleDisabled) {

    final CommandContext context = createContext(vscContext, vo, moduleDisabled);

    for (int i = 0; i < commands.size(); i++) {        final boolean result = commands.get(i).
execute(context);        if (!result) {        commands.get(i).cleanup(context);
  for (int j = i - 1; j > -1; j--) {

          commands.get(j).rollback(context);

        }

        return context.getStatus();

      }

    }

    return context.getStatus();

  }*/

/*-----------------------------------------Java8-----------------------------------------
                    Nous avons passé 1h30 sur cette méthode
Difficulté de transformer les boucles imbriquées avec les expressions Lambda sans passer
 par Foreach
---------------------------------------------------------------------------------------*/

public ReturnEnumeratedType executeJava8(final IContext vscContext, final FinalizationService
ValueObject vo, final boolean moduleDisabled) {
    final CommandContext context = createContext(vscContext, vo, moduleDisabled);
    List<FinalizationCommand> finalizationCommands = new ArrayList<>();
    commands.stream()
        .filter(com->!com.execute(context))
        .forEach(e->{
          finalizationCommands.add(e);
          e.cleanup(context);
          finalizationCommands.
              stream().
              forEach(ep->ep.rollback(context));
        });
    return context.getStatus();
  }
```

Etape 3 : tests de performance

Notre collègue « Hichem» nous a proposé d'utiliser « Contiperf » (http://databene.org/contiperf), un utilitaire qui nous a permis de dérouler des tests de performance sur certains tests unitaires Java 8 vs sans Java 8. Cet outil permet de générer un rapport contenant le temps d'exécution moyen, en percentile et le temps de latence maximum de chaque test.

NB : *Nous avons configuré un comportement du « StressTest » représentatif de la charge de production, ce qui consiste à injecter un grand nombre d'itérations en parallèle et non de gros volumes de données.*

« Profiling » du code avec « Jprofiler » afin de mesurer la quantité mémoire consommée pour chaque test déroulé. Les illustrations suivantes présentent un rapport du résultat d'un échantillon de « StressTest » tel que le génère « Contiperf » : Fig.2.

1. Rapport du résultat : ContiPerfReport

Comme vous pouvez le constater, les performances du code Java8 sont deux fois meilleures sur ce « StressTest ». Fig.3.

2. Rapport du résultat : ContiPerfReport

Contrairement à la première illustration, sur ce test le code Java8 est deux fois moins performant. Conclusion : le code est à revoir !

Résultats et interprétation

Nous avons réussi à migrer la totalité des classes simples avec une moyenne de 29 minutes par service, y compris les tests unitaires qui vont avec.

> C'est un résultat encourageant pour une équipe composée des niveaux mixtes. Le partage de connaissances et l'accompagnement des autres membres semblent avoir été efficaces.

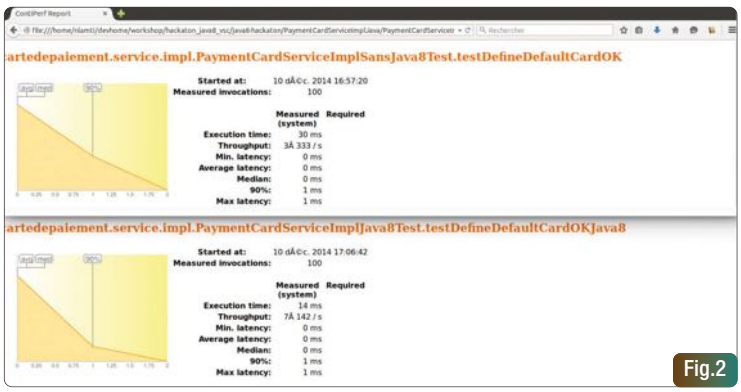


Fig.2

Comme vous pouvez le constater, les performances du code Java8 sont deux fois meilleures sur ce « StressTest ».

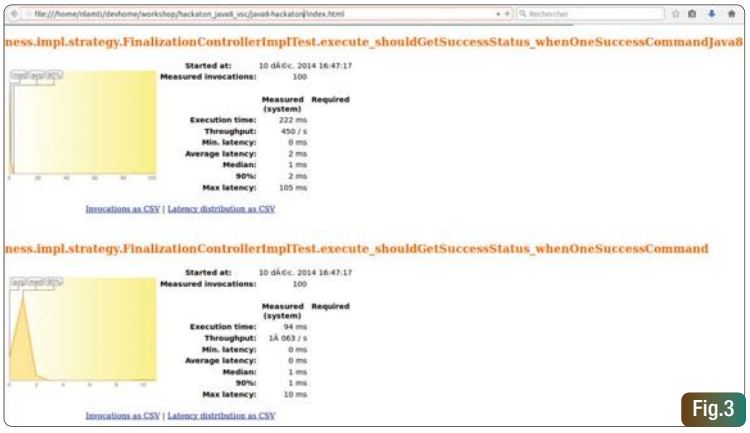


Fig.3

Contrairement à la première illustration, sur ce test le code Java8 est deux fois moins performant. Conclusion : le code est à revoir !

Seules les classes moyennes et complexes ont été modifiées. Par rapport aux classes simples, le temps moyen de transformation est estimé au double (1H00 / méthode) pour les classes moyennes et, au minimum, au triple (<1h30 / méthode) pour les classes complexes.

> Une connaissance plus approfondie sur les pratiques avancées des nouvelles APIs Java 8 est nécessaire. Si le graphique de dépendance des classes est de plus en plus complexe, l'impact du « Refactoring» peut facilement dépasser le changement du code lui-même.

Au niveau des performances, certains tests ont donné de bons résultats en termes de temps d'exécution et de consommation mémoire, quand d'autres sont moins bons (surtout pour le cas de classes complexes). Globalement, les performances sont équivalentes. Notez que nous n'avons pas testé les « ParallelStream », ceux-ci étant peu adaptés à nos volumes de production.

> Les conclusions restent à confirmer dans le cadre d'un vrai test de charge. Il est aussi probable que certains « Refactoring » et transformations devraient être revus compte tenu de leurs résultats insatisfaisants, et ce indépendamment de la performance Java 8.

Les tableaux suivants récapitulent le résultat comparatif des temps d'exécution et de consommation mémoire sur un cas de « StressTest » d'une classe simple, moyenne et complexe : résultat concluant pour les classes simples, mais le code est à revoir pour des cas de classes complexes. Fig.4, 5 et 6.

Classe simple	Ancien code	Nouveau code
Temps d'exécution (100 itérations, 1 thread)	203 ms	194 ms
Max latence (100 itérations, 1 thread)	64 ms	91 ms
Mémoire	17,2 MB	18,4 MB

Fig.4

Classe moyenne	Ancien code	Nouveau code
Temps d'exécution (1K itérations, 10 threads)	290 ms	313 ms
Max latence (1K itérations, 10 threads)	177 ms	193 ms
Mémoire	19,5 MB	20,8 MB

Fig.5

Classe complexe	Ancien code	Nouveau code
Temps d'exécution (10K itérations, 10 threads)	106 ms	206 ms
Max latence (10K itérations, 10 threads)	462 ms	624 ms
Mémoire	36,3 MB	36,9 MB

Fig.6

Conclusion

L'exercice de « Refactoring » que nous avons appliqué vient confirmer l'avantage d'utiliser les expressions Lambda de Java 8 : un code plus simple, avec moins de typages inutiles, plus maintenable et moins verbeux. Cet exercice de « Refactoring » sur une « Codebase » réelle est très intéressant : il permet une courbe d'apprentissage extrêmement rapide pour un investissement somme toute relatif. Pas de dégradation des performances sur les transformations simples grâce à une pratique correcte des expressions Lambda.

Cela n'a pas été le cas sur les classes complexes, montrant un risque de dégradation lié d'une part à des « Refactoring » importants sur du code complexe et d'autre part à une mauvaise utilisation des expressions Lambda. Par rapport aux résultats attendus de ce hackaton, nous n'avons pas pu transformer la totalité des classes et les performances n'étaient pas au rendez-vous, surtout sur les classes moyennes et complexes. Nous n'avons donc pas pu diffuser un message rassurant tel que nous l'avions souhaité au début de cet article.

Développer avec les expressions « Lambda » et les nouvelles API Java 8 ne se fait pas du jour au lendemain, surtout quand il s'agit d'une réécriture de code dans le cadre d'un grand projet à haute disponibilité. Une bonne approche pour poursuivre cette migration serait donc :

D'adopter une stratégie de migration progressive en ciblant un projet pilote ;

De faire émerger de ce projet un pôle de compétences Java8/Lambda ;

D'utiliser ce pôle pour permettre un transfert de connaissances plus rapide et un retour d'expérience plus réaliste aux autres équipes ;

D'assurer la montée en compétences générale en multipliant les workshops et les évènements autour du sujet ;

D'inciter à la bonne pratique d'utilisation des outils de code qualité existants et compatibles Java 8 tels que « PMD», « FindBugs », « CheckStyle » et « SonarQube ».

Ensuite, comme souvent pour ce type de migration, il sera plus efficace de les inclure dans des projets de refonte déjà planifiés dans les roadmaps projets, voire des projets métiers, plutôt que de les gérer en « standalone ». Aujourd'hui, VSCT a choisi de s'inscrire dans cette optique par le biais du lancement de ses applications « NextGen ».

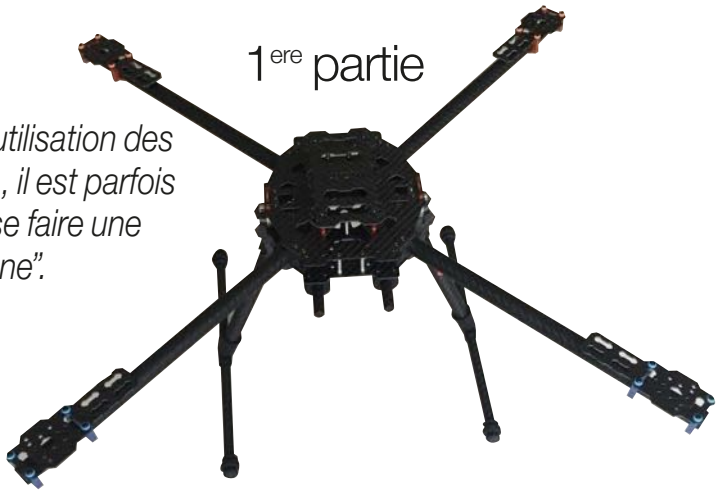
Pour finir, nous n'en sommes aujourd'hui qu'au début de cette nouvelle génération du langage Java.

Ce dernier demeure en mode expérimentation : le décollage s'effectue donc en douceur et le code reste toujours rétro-compatible avec les anciennes versions. Ce nouveau mode de programmation et les langages fonctionnels semblent être la nouvelle orientation à prendre. On risque cependant d'attendre longtemps avoir de le voir généralisé sur tout le « Core Java ». En attendant, voyons ce que donnera le projet « Jigsaw » sur la version 9.

- D'adopter une stratégie de migration progressive en ciblant un projet pilote ;
- De faire émerger de ce projet un pôle de compétences Java8/Lambda ;
- D'utiliser ce pôle pour permettre un transfert de connaissances plus rapide et un retour d'expérience plus réaliste aux autres équipes ;
- D'assurer la montée en compétences générale en multipliant les workshops et les évènements autour du sujet ;
- D'inciter à la bonne pratique d'utilisation des outils de code qualité existants et compatibles Java 8 tels que « PMD», « FindBugs », « CheckStyle » et « SonarQube ».

Construire son drone

1ere partie



Bien que le législateur ait commencé à se pencher sur l'utilisation des drones de loisirs (ou UAV pour Unmanned Aerial Vehicle), il est parfois difficile pour les profanes comme pour les amateurs de se faire une idée sur ce qui se cache exactement derrière le mot "drone".

François Chevalier
Concepteur développeur chez SQLI Enterprise à Nantes depuis 2011. Passionné par les nouvelles technologies et l'aéronautique, il s'intéresse au concept du Do It Yourself et essaye de l'appliquer dans son quotidien.

Il existe une communauté grandissante en France autour du drone et beaucoup de passionnés de modélisme se tournent également vers cet engin. Parmi les disciplines qui se pratiquent avec un drone, la course de drone (FPV Racing) connaît un fort essor : elle consiste à mettre au défi des pilotes de petits drones légers munis d'une caméra. Le pilote dirige son drone à l'aide de lunettes qui retransmettent l'image de la caméra (First Person View). La prise de vue est l'autre discipline majeure chez les pilotes de loisirs. Dans ce cas, la stabilité du drone prime sur sa vitesse et l'on a souvent affaire à des drones de tailles beaucoup plus importantes. Il existe de nombreuses autres applications à l'utilisation des drones dans le monde professionnel mais je ne m'attarderai pas sur ce point ici.

Mon objectif aujourd'hui est de vous faire partager les différentes étapes dans la fabrication d'un drone et vous aider, grâce à mon expérience, à vous éviter quelques écueils auxquels j'ai été confronté. Étant développeur de métier, passionné par tout ce qui vole et un peu débrouillard en électronique, il ma semblé naturel de me lancer dans la fabrication d'un drone! On voit un peut partout les mentions Do It Yourself alors pourquoi ne pas le faire ?

Quel usage pour ce drone ?

Avant de se lancer dans la fabrication d'un de ces engins volants, il faut bien identifier l'usage que l'on va en faire car cette finalité va conditionner toute la suite du processus. Pour ma part, j'ai eu envie de réaliser un drone sur lequel j'allais pouvoir installer une caméra de type GoPro afin de pouvoir réaliser des prises de vue aériennes. Dans un premier temps, je me contenterai de le diriger par le biais d'une radio-commande, ou via un logiciel dans lequel je peux lui donner des points GPS à suivre et le laisser en autonomie. Mais mon objectif final est de pouvoir le rendre plus autonome en lui permettant de faire des prises de vue en me suivant grâce à une balise GPS que je porterai sur moi.

Avec ces objectifs en tête, je sais maintenant que je vais devoir réaliser un drone capable de soulever un certain poids et qui soit relativement stable. Après avoir parcouru les différents forums et sites spécialisés, j'ai relevé qu'il existe un grand nombre de drone différents : avion, quadcopter, hexacopter, octocopter,... Après avoir comparé ces différents modèles, j'en suis venu à la conclusion qu'un quadcopter remplira les critères que je me suis fixé. En effet, un quadcopter (engin à quatre hélices) permet de faire du vol relativement stationnaire, d'avoir une stabilité correcte et surtout -ce paramètre n'est pas négligeable- il devrait rentrer dans mon budget. J'insiste sur ce point car faire moi-même un drone me permet d'étaler les dépenses et de pouvoir gérer un budget serré.

Le matériel

Je résume le matériel nécessaire pour réaliser un drone de type quadcopter :

- 1 Châssis (Frame)
- 4 Moteurs brushless
- 4 Hélices (2 sens horaire + 2 sens anti-horaire)
- 4 Contrôleurs de vitesse (ESC)
- 1 Contrôleur de vol
- 1 Module GPS
- 1 Récepteur RC
- 1 Batterie Lipo
- 1 Radio-commande

Budget minimum estimé : entre 600 et 800 euros

Auquel s'ajoute le matériel nécessaire pour la programmation :

- 1 Module de télémétrie (environ 80 euros)
- 1 Raspberry Pi (facultatif)

Et enfin, le matériel pour la prise de vue :

- 1 Nacelle (Gimbal) (environ 80 euros)
- 1 Caméra type GoPro

Un chassis : et si on jouait au "Tarot"?

Le châssis de mon futur quadcopter devait répondre à deux exigences : avoir des pieds suffisamment long pour pouvoir installer une nacelle sur laquelle je fixerais une caméra et être léger pour ne pas handicaper le drone en terme d'autonomie, car il est évident que plus un drone est lourd, moins il est autonome. Il s'agit donc de trouver le juste milieu en espérant faire le bon choix. Dans mon cas, j'ai décidé d'acheter un châssis de la marque Tarot : le modèle FY650 IRON MAN 650 Quad-Copter. C'est un châssis en carbone équipé d'un rail sur lequel on peut monter facilement une nacelle. Il peut accueillir quatre moteurs et a la forme d'un X. Cette forme est très commune pour les drones à quatre hélices. L'inconvénient de ce châssis est que le manuel de montage est très succinct et donc pas évident au premier abord. Heureusement il existe sur Internet de nombreuses vidéos de tutoriels qui expliquent les différentes étapes de montage de ce châssis. Prévoyez d'investir dans des clés « allen » avec des diamètres très petits si vous n'en avez pas déjà. Pour ce modèle Tarot, il m'a fallu des clés de 1,5 mm et 2 mm et quelques acrobaties pour pouvoir atteindre certaine vis, et bien sûr l'indispensable smartphone en mode "niveau" pour que tout ce montage soit d'aplomb!

Quatre moteurs pour un décollage

Le choix des moteurs ne peut être dissocié du choix des hélices car c'est l'association de ces deux éléments qui va faire que votre drone va décoller ou pas ! Pour faire simple, si votre hélice est trop grande et que votre moteur n'a pas assez de couple alors vous risquez de griller votre moteur car l'effort à fournir par celui-ci sera trop important. A l'inverse un moteur avec beaucoup

de couple et une petite hélice sera inefficace et dans ce cas votre drone restera sur le plancher des vaches, faute de générer suffisamment de portance. Partant de ce principe, quel choix doit-on faire ? Là encore, le choix dépend de l'utilisation que vous voulez faire de votre drone : si vous avez besoin qu'il soit rapide et très réactif, voire si vous souhaitez faire de l'acrobatie, alors dans ce cas il faudra choisir des moteurs ayant une vitesse de rotation très importante et de petites hélices. Dans le cas où la stabilité est primordiale, alors on s'orientera vers des moteurs à fort couple mais avec une faible vitesse de rotation sur lesquels on montera des hélices de grande taille. Pour ma part c'est vers cette deuxième solution que je me suis tourné. Le moteur est défini par plusieurs caractéristiques :

- Le KV, le voltage qu'il peut supporter et le nombre d'ampères qu'il va consommer au maximum de sa charge. Le KV est ce qui nous intéresse dans un premier temps : c'est le nombre de tours par minute par Volt. Un moteur avec un KV très élevé va entrer dans la catégorie des moteurs à très grande vitesse mais à faible couple, alors qu'un moteur avec KV faible entrera dans la catégorie inverse. Si je prends l'exemple du moteur que j'ai choisi : un T-Motor avec un KV de 380 alimenté par une batterie qui fournira environ 14,8 V, mon moteur tournera ainsi à 14,8 * 380 = 5 624 tours/min. Ce nombre peut paraître élevé, mais sachez que pour les drones utilisés pour les courses de FPV par exemple, les moteurs ont des vitesses de rotation supérieures à 13 000 tours/min.
- Un autre élément à prendre en compte est le nombre d'ampère que le moteur va consommer à pleine charge, car cette valeur va nous servir à choisir l'ESC (Electronic Speed Control). Cet élément qui, lié au moteur devra pouvoir supporter la charge maximale demandée par le moteur (nous reviendrons sur ce point en détail dans une autre partie).
- Enfin, le Voltage que le moteur peut supporter est en général une fourchette exprimée en nombre de S, 1 S étant équivalent à 3,7V. Dans le cas du moteur que j'ai choisi, son fonctionnement doit se faire avec des batteries de type 4S ou 6S (nous détaillerons ce point dans la partie consacrée à la batterie).

Les moteurs sont en général livrés avec la visserie nécessaire pour les fixer au châssis, mais il m'a fallu adapter la longueur des câbles pour que le montage moteur/ESC soit propre et corresponde aux caractéristiques de mon châssis.

Hélice cherche moteur pour s'envoyer en l'air

L'hélice a, quant à elle, deux valeurs qui vont la définir : la longueur et le pas. Ces valeurs sont exprimées en pouces (“). Le moteur que j'ai choisi ayant un fort couple, mon choix d'hélice se portera vers des hélices de 13” ou 14” de longueurs avec un pas de 4,5” ou 5”. Une grande longueur associée à un pas important permettent d'avoir une grande portance même à faible vitesse, ce qui devrait, en associant cette configuration à mes moteurs, produire l'effet escompté : décoller! Il ne faudra pas oublier de vérifier lors de l'achat de ces éléments, s'ils sont compatibles entre eux. J'entends par là que pour qu'une hélice puisse se monter sur un moteur, il faut vérifier que l'adaptateur existe. Et vous devrez prendre en compte que pour un multicopter, vous allez avoir besoin d'hélices conçues pour tourner dans le sens des aiguilles d'une montre et d'autres dans le sens inverse. En effet si toutes vos hélices tournent dans le même sens, votre drone a de grandes chances de tourner sur lui même et de devenir incontrôlable!

Electronic Speed Control, le compagnon du moteur brushless

Contrairement à un moteur classique, un moteur brushless a besoin d'un contrôleur pour le faire fonctionner, ce contrôleur permet de gérer la vitesse de rotation du moteur. Il est alimenté par la batterie (par les câbles rouge et noir sur la photo) et alimente lui même le moteur via trois câbles d'alimentation (3 câbles noirs dans mon cas). Il est aussi muni d'un câble qui le relie au contrôleur de vol. C'est via celui-ci qu'il reçoit les ordres en termes de vitesse de rotation du moteur. L'ESC contient un micro-contrôleur et peut être pré-chargé avec un firmware. Il existe plusieurs types de firmware dont le plus connu est SimonK. C'est ce firmware qui nous intéresse car il est particulièrement bien adapté aux multicopters et permet d'avoir des moteurs très réactifs. Je suis donc parti à la recherche d'ESC ayant le firmware SimonK déjà pré-chargé, et pouvant encaisser les 20A que chaque moteur peut utiliser à pleine charge. J'ai fait le choix d'utiliser des ESC T-Motor 35A avec le bon firmware chargé. J'ai une sécurité de 15A supplémentaire, ce qui devrait éviter les problèmes de surchauffe et les éventuelles surcharges. En parcourant les différents sites de vente de matériel pour drones, vous noterez parfois que certains ESC ont une mention BEC. Cette mention signifie que l'ESC est capable de fournir un courant supplémentaire, de 5V en général, vers le contrôleur de vol, et donc permet d'alimenter ce dernier. Le contrôleur de vol que j'ai choisi dispose d'un système d'alimentation qui passe par la batterie directement, donc je n'ai pas besoin de ce dispositif sur mes ESC. J'ai décidé de les installer au plus prêt des moteurs car mon châssis s'y prêtait, et j'ai utilisé du scotch d'électricien de couleur bleue et rouge pour les fixer. Cela me permettra aussi en vol de différencier plus facilement l'avant et l'arrière de mon drone.

Je me trouve maintenant avec un châssis, sur lequel j'ai monté quatre moteurs brushless auxquels j'ai relié quatre ESC. Il m'a fallu user de pinces, d'un fer à souder ainsi que de gaines thermo-rétractables pour pouvoir adapter et relier les différents éléments entre eux.

La suite au prochain épisode... ⊞

Connectez votre rameur d'appartement avec Chrome

2e partie

WiiFit, AppleHealth, Google Fit, tout ça c'est du passé ! Place à SkiffSimulator !! Ça va ramer dans les salons !

Jean-François Garreau
SQLi

Gestion du moteur

Un des éléments clés du programme est l'alimentation du modèle depuis l'application Chrome. L'application Chrome va alimenter le modèle et déterminer un certain nombre d'éléments en lien avec le rendu souhaité, puis appeler le moteur de calcul pour terminer les traitements. On vérifie par exemple dans quel sens va le joueur, s'il y a eu un déplacement, etc.

```
// Calcul
function setDistance(distance) {
  if (gameModel.distanceArduino === distance) {
    gameModel.direction = 0;
    }
  else if (gameModel.distanceArduino > distance) {
    gameModel.direction = 1;
    }
  else {
    gameModel.direction = - 1;
    }
  // Vitesse en cm / ms
  var deltaCM = Math.abs(gameModel.distanceArduino - distance);
  if (deltaCM > ConstSAS.MIN_DELTA_CM) {
    gameModel.speed = deltaCM / ConstSAS.DELAY;
    }
  else {
    gameModel.speed = Math.max(gameModel.speed - ConstSAS.FACTOR_SPEED,
    0);
    }
  gameModel.distanceArduino = distance;
  engineSkiff();
  }
```

Le coeur du moteur quant à lui, s'occupe uniquement de lire les données issues de l'Arduino et de calculer la distance globale parcourue, ainsi que le pourcentage de déplacements dans l'écran.

```
function engineSkiff() {
  if (gameModel.speed > 0 && gameModel.stateGame === constState.STATE_RUNNING) {
    var distanceSpeed = gameModel.speed * ConstSAS.DELAY;
    // On incrémente la distance
    gameModel.distanceSkiff += (distanceSpeed * ConstSAS.FACTOR_DISTANCE);
    // On gère l'effet de déplacement des bords via un pourcentage
    gameModel.percent = (gameModel.distanceSkiff % 100) / 100;
    }
  }
```

SPÉCIFIQUE CHROME

Depuis le début nous parlons d'application Chrome et de l'API Serial, il est désormais temps de voir comment nous implémentons cette partie.

Interaction avec l'Arduino

Toute l'interaction avec l'Arduino se fait directement via le port série. Mais il faut pour cela passer par des étapes clés :

- Récupérer les appareils connectés sur les ports série "chrome.serial.getDevices";
- Une fois un appareil trouvé, on s'y connecte "chrome.serial.connect". Dans notre exemple, on veille à ce qu'il n'y ait qu'un appareil de connecté au moment du lancement de l'application;
- Lire le port série "chrome.serial.onReceive".

```
function initArduino() {
  chrome.serial.getDevices(function(ports) {
    if (ports && ports.length == 1) {
      chrome.serial.connect(ports[0].path,
      onOpenArduino);
      }
    }
  );
  }
function onOpenArduino(openInfo) {
  connectionId = openInfo.connectionId;
  console.log("connectionId: " + connectionId);
  if (connectionId == - 1) {
    console.log('Could not open');
    return;
    }
  console.log('Connected');
  chrome.serial.onReceive.addListener(onReadArduino);
  }
function convertArrayBufferToString(buf) {
  return String.fromCharCode.apply(null,
  new Uint8Array(buf));
  }
function onReadArduino(readInfo) {
  if (readInfo.connectionId == connectionId && readInfo.data) {
    var str = convertArrayBufferToString(readInfo.data);
    if (str.charAt(str.length - 1) === '\n') {
      value += str.substring(0,
      str.length - 1);
      if (regExp.test(value)) // Light on and off
      {
        var distanceTmp = + regExp.exec(value)[1];
        if (distanceTmp < ConstSAS.DISTANCE_MAX && Math.abs(distance - distanceTmp) < (Const
SAS.DISTANCE_MAX * 1.5) ) {
          AppSAS.setDistance(distanceTmp);
          }
        distance = distanceTmp;
        }
      value = "";
      }
```

```
    else {
      value += str;
      }
    }
  }
```

Il faut noter une petite particularité lors de la lecture : nous lisons le port série et nous attendons une chaîne de caractères. Aussi, il est important de penser à convertir les données issues du port série en données exploitables sous forme de chaîne de caractères.

GESTION DES ÉCRANS ET CAS PARTICULIERS

Maintenant que nous avons vu la mécanique sous le capot, regardons de plus près quelques cas particuliers qui méritent un peu d'attention.

Déplacement du décor

Afin de donner une sensation de déplacement, il nous faut bouger nos rives. Pour ce faire, on va simplement fonctionner avec un indicateur en pourcentage en rapport avec le déplacement global du rameur. Ainsi pour afficher correctement nos 2 rives qui bougent, il nous suffit juste de dessiner 2 fois chaque rive de chaque côté afin de gérer les dépassement d'écran et de les positionner en fonction d'un pourcentage résultant d'un modulo de la distance du rameur :

```
// Affiche le rivage en fonction de la rive souhaitée et de la progression du rameur
function paintRive(riveDroite) {
  var rive = ui.resources.images[(riveDroite ? 'rive_droite' : 'rive_gauche') + getSuffix()];
  var finalHeight = rive.height * ui.ratio,
  finalWidth = rive.width * ui.ratio;
  ui.context.drawImage(rive , 0 //sx clipping de l'image originale
  , 0 //sy clipping de l'image originale
  , rive.width // swidth clipping de l'image originale
  , rive.height // sheight clipping de l'image originale
  , riveDroite ? ui.canvas.width - finalWidth : 0 // x Coordonnées dans le dessing du canvas
  , 0 - (finalHeight * gameModel.percent) // y Coordonnées dans le dessing du canvas
  , finalWidth // width taille du dessin
  , finalHeight // height taille du dessin
  );
  ui.context.drawImage(rive , 0 //sx clipping de l'image originale
  , 0 //sy clipping de l'image originale
  , rive.width // swidth clipping de l'image originale
  , rive.height // sheight clipping de l'image originale
  , riveDroite ? ui.canvas.width - finalWidth : 0 // x Coordonnées dans le dessing du ui.canvas
  , finalHeight - (finalHeight * gameModel.percent) // y Coordonnées dans le dessin du canvas
  , finalWidth // width taille du dessin
  , finalHeight // height taille du dessin
  );
  }
```

L'affichage du rameur

L'affichage du rameur comporte 2 parties à prendre en compte :

- L'affichage du rameur en fonction de la position du joueur;
- L'avancée du rameur quand le mode ghost est activé.

Position du rameur en fonction du joueur

Afin de restituer au mieux les gestes effectués par le joueur, il a fallu réfléchir à une façon d'afficher la bonne image de rameur, en fonction de sa position sur le rameur. La réponse était relativement simple : il suffit de connaître à chaque instant la position du joueur sur le rameur et sa direction, puis d'appliquer la bonne image Fig.1.

```
function indexToUse(direction, distance) {
  var arrayToUse = direction >= 0 ? mappingPositonRameurFront : mappingPositonRameurBack;
  for (var i = 0; i < arrayToUse.length; i++) {
    var minMax = arrayToUse[i];
    if (distance > minMax.min && distance <= minMax.max) {
      return minMax.indexSprite + AppSAS.getSuffix();
      }
    }
  return mappingPositonRameurBack[0].indexSprite + AppSAS.getSuffix();
  }
// Affiche le bon sprite du bateau
function paintBoat() {
  //var ratio = 0.05;
  var image = AppSAS.ui.resources.images[AppSAS.gameModel.indexSprite];
  AppSAS.ui.context.shadowOffsetX = 0;
  AppSAS.ui.context.shadowOffsetY = 0;
  AppSAS.ui.context.shadowBlur = 0;
  AppSAS.ui.context.drawImage(image , 0 //sx clipping de l'image originale
  , 0 //sy clipping de l'image originale
  , image.width // swidth clipping de l'image originale
  , image.height // sheight clipping de l'image originale
  , (AppSAS.ui.canvas.width / 2) - ((image.width * AppSAS.ui.ratio) / 2) // x Coordonnées dans le dessin
du AppSAS.ui.canvas
  , (AppSAS.ui.canvas.height / 2) - ((image.height * AppSAS.ui.ratio) / 2) + 100// y Coordonnées dans
le dessin du AppSAS.ui.canvas
  , image.width * AppSAS.ui.ratio // width taille du dessin
  , image.height * AppSAS.ui.ratio // height taille du dessin
) }
```

De cette manière nous affichons toujours la bonne image. Puis, afin de faciliter le déplacement du bateau, il a été considéré qu'il était en position fixe sur l'écran et que l'illusion du déplacement se fait uniquement à travers le déplacement du décor.

Gestion du ghost

L'affichage du ghost doit faire face à un problème. Contrairement au bateau, ce dernier se déplace sur l'écran. Le problème est que ce déplacement vient surtout du fait qu'il a fallu gérer le cas d'un ghost allant plus vite que le joueur courant. En effet, si le ghost est meilleur que le joueur, alors, il sera vers le haut de l'écran ce qui veut dire que sa position va s'approcher de l'axe, voire aller dans les négatifs. Or dans un canvas, si un élément est dessiné avec des coordonnées de destination dans le négatif, l'élément n'est tout simplement pas peint ! Il a donc fallu tronquer l'image source

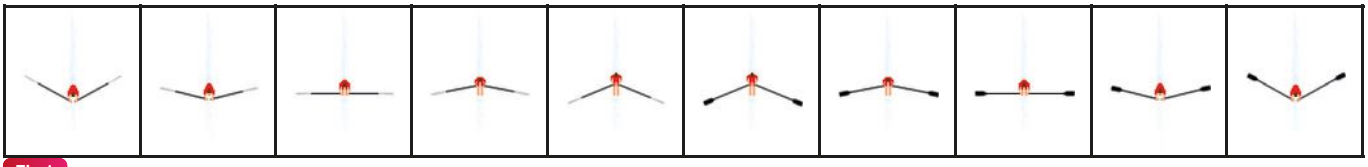


Fig.1

pour donner l'illusion que le dessin du ghost parte vers le haut de l'écran.

```
// Affiche le bon sprite du bateau du mode Ghost
function paintGhost() {
  //var ratio = 0.05;
  var image = AppSAS.ui.resources.images[AppSAS.gameModel.indexSpriteGhost];
  ...
  // Le fantôme doit être dessiné là où il est au niveau de sa distance globale par rapport au bateau actuel
  // => On affiche là où est son delta en distance par rapport à bateau actuel
  var stateGhost = AppSAS.gameModel.ghost[AppSAS.gameModel.step];
  var deltaGhost = AppSAS.gameModel.distanceSkiff - stateGhost.distanceSkiff;
  // On doit tronquer le ghost s'il dépasse de l'écran
  var yGhost = (AppSAS.ui.canvas.height / 2) - ((image.height * AppSAS.ui.ratio) / 2) + 100 + (deltaGhost *
ConstSAS.FACTOR_GHOST);
  var heightSpriteGhost = image.height;
  if (yGhost < 0) {
    heightSpriteGhost = image.height + yGhost;
    yGhost = 0;
    }
  AppSAS.ui.context.drawImage(image , 0 //sx clipping de l'image originale
  , image.height - heightSpriteGhost //sy clipping de l'image originale
  , image.width // swidth clipping de l'image originale
  , heightSpriteGhost // sheight clipping de l'image originale
  , (AppSAS.ui.canvas.width / 2) - ((image.width * AppSAS.ui.ratio) / 2) // x Coordonnées dans le dessin
du AppSAS.ui.canvas
  , yGhost // y Coordonnées dans le dessin du AppSAS.ui.canvas , image.width * AppSAS.ui.ratio // width
taille du dessin
  , heightSpriteGhost * AppSAS.ui.ratio // height taille du dessin
  );
  ...
}
```

Fig.2

Affichage des écrans de login & de scores

Ces 2 écrans sont différents car on y affiche non pas une animation, mais du texte Fig.2.

Ecran de Login

Concernant l'écran de login, il n'existe pas d'équivalent du champ input dans un canvas. Aussi, il a fallu intégrer à notre html une balise input que l'on affiche ou cache en fonction du besoin.

Ecran de scores

L'affichage du score est simple car il ne s'agit que d'afficher du texte :

```
AppSAS.ui.context.fillText(MyText', x, y);
```

La taille et la couleur du texte sont définis par des propriétés appliquées directement sur le context du canvas.

Persistance des données

Pour sauvegarder les données d'une partie à l'autre, ou même d'un démarrage d'application à l'autre. J'ai fait le choix le plus simple : le LocalStorage. Le seul hic avec le LocalStorage et les ChromeApps, c'est que l'API telle qu'elle est disponible en html5 n'existe pas sur une ChromeApp. En effet, l'api localStorage étant syncrhone, Google a préféré mettre en place une solution asyncrhone : https://developer.chrome.com/apps/storage. J'ai donc mis en place la solution de Google et j'en ai profité pour prévoir une api uniforme entre localstorage & chrome.storage au cas où vous partiriez sur une solution native html5.

Placer l'Arduino

Afin de mesurer au mieux les données de distance, j'ai placé l'Arduino au dos de l'utilisateur et j'ai fabriqué une petite boîte pour packager un peu tout ça : Fig.3.

ANNEXES

Le code complet est disponible https://github.com/sqli-nantes/skiff-simulator

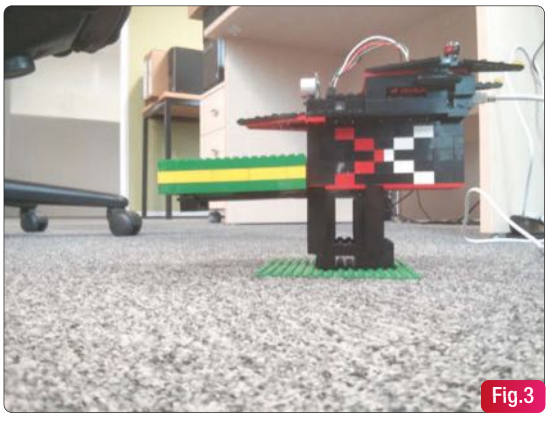


Fig.3

A la découverte des profils Java 8

Sortie en Mars 2014, la nouvelle mouture de Java a été l'évènement du début d'année dans le monde informatique. Les articles et les sujets de blogs ont foisonné mettant en exergue la révolution des Lambdas expressions ou encore le soulagement de bénéficier enfin d'une API de gestion du temps digne de ce nom. A contrario, l'introduction des profils au sein de Java 8 est restée un sujet plutôt confidentiel alors qu'il constitue un grand pas pour le futur de la plateforme. Tentative d'explications.

Sylvain SAUREL
Ingénieur d'Etudes Java / Java EE
sylvain.saurel@gmail.com

Un peu de théorie

Au fil du temps, la plateforme Java a gagné en maturité mais également en fonctionnalités devenant la solution de référence pour les développements informatiques en entreprise dans de nombreux domaines. L'ajout de fonctionnalités aura au final conduit à augmenter sans cesse la taille de l'environnement d'exécution Java plus connu sous le nom de JRE (Java Runtime Environment). Avec Java 7, le JRE complet tournait ainsi autour des 140 Mo ! Cette taille pose d'autant plus de problèmes que pour beaucoup d'applications une grande partie des fonctionnalités du JRE ne sont pas utilisées (JMX ou Corba par exemple).

Réduire l'empreinte mémoire de la plateforme Java est donc devenu une priorité absolue pour les équipes en charge de Java chez Oracle. Sans prendre en compte le code natif chargé par les bibliothèques dynamiques, le JRE se compose d'un bloc monolithique de près de 60 Mo avec la bibliothèque rt.jar . Découper ce bloc imposant en sous blocs est une nécessité afin de modulariser le JRE. Cette modularisation devant notamment amener les gains suivants :

- Un temps de démarrage réduit pour la JVM ;
- Une consommation mémoire réduite ;
- Un nettoyage du coeur de la plateforme en retirant des packages du coeur les APIs n'ayant rien à y faire et les classes non utilisées ;
- Une sécurité accrue du fait de la réduction du coeur de la plateforme et donc de la surface d'attaque potentielle ;
- Une convergence accrue entre Java ME (version destinée à l'embarquée) et Java SE.

Ce projet de modularisation ne date pas d'hier et les développeurs Java ont tous déjà entendu parler du projet Jigsaw. Projet au combien ambitieux, Jigsaw visait une modularisation complète de la plateforme tout en prenant en compte également les besoins suivants :

- Gestion des dépendances en s'appuyant sur les bonnes pratiques et les erreurs de solutions comme Maven, Apache Ivy ou OSGI notamment ;
- Isolation des dépendances afin de résoudre le problème de versionning des bibliothèques ;
- Autoriser le packaging d'applications sous la forme de modules plutôt que sous la forme de JAR.

Cette approche nécessitait d'énormes modifications en profondeur au sein de la plateforme Java. En particulier, elle implique la mise en place d'un classloader lui aussi modulaire ce qui induit une grande complexité pour garantir une compatibilité ascendante.

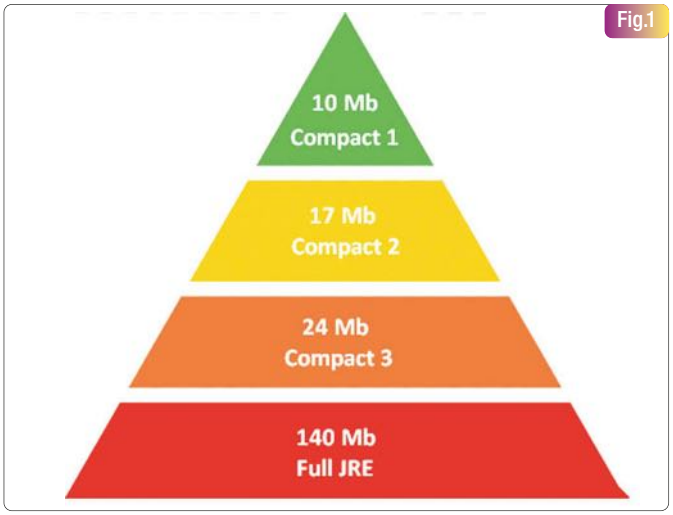
Devant l'ampleur de la tâche et afin de livrer enfin la version 8 de Java, Oracle a décidé de repousser ce vaste chantier de modularisation à Java 9 attendu pour 2016. Néanmoins, les travaux initiaux n'auront pas été vains puisque Java 8 décide de franchir un premier pas dans ce chantier de modularisation en proposant les profils Compact. Si, à première vue, la montagne a accouché d'une souris, la réalité est tout de même bien plus rose. En effet, les profils Compact de Java 8 s'approprient une partie, somme toute basique, des avantages attendus du projet Jigsaw :

- Une compatibilité totale avec la JVM et les spécifications du langage Java ;
- Une réduction substantielle de l'empreinte mémoire de la plateforme ;
- Un retrait de fonctionnalités qui ne sont plus toujours utiles (CORBA par exemple) ;
- Un fonctionnement pour la plupart des applications et en particulier les applications serveurs.

Concrètement, les profils Compact de Java 8 se basent sur les packages. Ainsi, un profil contient obligatoirement des packages complets. Il n'y a donc pas de support pour des packages partiels. En sus, un profil doit être un ensemble fermé ce qui implique que les références à des classes non présentes au sein du profil sont interdites. De fait, si un profil contient des classes d'un autre profil plus petit, il doit les contenir entièrement ce qui induit donc que les profils sont additifs.

Les profils sont au nombre de 3 auxquels il faut ajouter le JRE complet (Fig.1).

Fig.1



Les 3 Profils Compact

On remarque que le profil compact1 est le plus petit avec une taille légèrement en-deçà des 10 Mo, ce qui constitue un gain d'espace assez drastique. Il correspond au coeur de la plateforme Java et contient notamment les packages java.io, java.lang, java.nio, java.math, java.text, java.util, java.security, javax.security ainsi que java.text. Le profil compact2 correspond au compact1 auquel sont ajoutés les packages liés aux invocations RMI, à SQL et XML. Enfin, le compact3 comprend tout le compact2 auquel sont adjoints les outils de management liés à JMX ou encore les bibliothèques de cryptographie additionnelles. Les développeurs d'IHM client lourd auront tout de suite noté que ces profils ne contiennent pas de classes liées aux interfaces utilisateurs. Ainsi, pour les applications recourant à des classes de Swing ou de AWT, il faudra passer par le JRE complet. La Fig.2 donne un aperçu plus détaillé du contenu des différents profils tout en gardant à l'esprit qu'ils sont additifs.

Place à la pratique

La théorie autour des Profils Java 8 assimilée, les développeurs ont besoin d'outils pour pouvoir développer des applications s'appuyant dessus. En effet, il est difficile de se rappeler quelle API se trouve dans quel Profil Compact. Pour aider les développeurs à tirer partie des profils et favoriser au mieux leur adoption, Java 8 met à disposition 1 outil en ligne de commande spécifique nommé jdeps et rajoute des options dédiées au compilateur javac. A la compilation d'un programme il est désormais possible d'utiliser l'option -profile afin de spécifier au compilateur de générer une erreur si le programme utilise des classes d'un profil supérieur. Ici, on peut préciser que notre programme MyProg se limite à utiliser le profil compact1 :

```
javac -profile compact1 MyProg.java
```

Nouvellement ajouté avec Java 8, jdeps est un outil d'analyse statique qui réalise une analyse des dépendances d'une classe spécifique ou d'un fichier JAR. Outre le fait qu'il permet de connaître les dépendances en regard d'un profil, l'outil se révèle également très utile avec son option -profile (ou son raccourci -P) indiquant de quel profil dépendent des packages.
Nous pouvons par exemple analyser les dépendances de l'IHM de démo Notepad.jar inclus avec le JDK :

```
$ jdeps demo/jfc/Notepad/Notepad.jar

demo/jfc/Notepad/Notepad.jar -> /usr/java/jre/lib/rt.jar
  <unnamed> (Notepad.jar)
    -> java.awt
    -> java.awt.event
    -> java.beans
    -> java.io
    -> java.lang
    -> java.net
    -> java.util
    ...
```

Afin de savoir à quel profil appartiennent les dépendances de Notepad.jar, la ligne de commande suivante sera employée :

```
$ jdeps -profile demo/jfc/Notepad/Notepad.jar
demo/jfc/Notepad/Notepad.jar -> /usr/java/jre/lib/rt.jar (Full JRE)
  <unnamed> (Notepad.jar)
    -> java.awt        Full JRE
    -> java.awt.event      Full JRE
    -> java.beans    Full JRE
    -> java.io         compact1
    -> java.lang       compact1
    -> java.net          compact1
```

Fig.2

Full JRE	Beans Preferences RMI-IIOP Sound AWT Image I/O	Input Methods Accessibility CORBA Swing Drag and Drop JAX-WS	IDL Print Service Java 2D
compact3	Security XML JAXP	JMX Management	JNDI Instrumentation
compact2	JDBC	RMI	XML JAXP
compact1	Core (`java.lang.*`) Networking Date and Time Logging JAR Internationalization Extension Mechanism	Security Ref Objects Input/Output Concurrency ZIP JNI Scripting	Serialization Regular Expressions Collections Reflection Versioning Override Mechanism

Contenu des Profils

```
    -> java.util          compact1
    -> java.util.logging compact1
    -> javax.swing        Full JRE
    ...
```

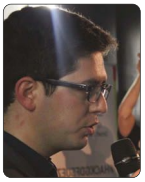
L'utilitaire jdeps prend un grand nombre de paramètres en entrée lui permettant d'être souple et de s'adapter à un grand nombre de cas d'utilisations. Les développeurs désireux d'utiliser les profils Compact ne pourront rapidement plus s'en passer.
Pour terminer un mot sur le support par les IDE de ces outils tournant autour des profils de Java 8. Développé par Oracle, l'IDE NetBeans en version 8.0 supporte fort logiquement un large éventail des nouvelles fonctionnalités de Java 8. Ainsi, les profils Compact ne font pas exception à la règle. Lors de la configuration d'un projet, l'IDE permet de sélectionner en regard de quel profil il devra être compilé. Il devient alors plus facile de cibler un profil en particulier et de vérifier les classes ayant des dépendances non souhaitées vers d'autres profils. Bien entendu, il reste possible de sélectionner une compilation standard en ciblant un JRE complet. Un support du même type est planifié dans les futures releases des outils Java pour Eclipse.

Conclusion

Premier arrêt sur le long chemin menant à la modularisation du JDK, les profils Compact de Java 8 représentent un pas significatif pour le futur de la plateforme. Réduire la taille du JDK et son empreinte mémoire s'avère fondamentale pour l'amélioration des performances des applications Java que ce soit en milieu embarqué, desktop ou encore serveur. La prochaine étape sera celle de la mise en place du projet Jigsaw au sein de Java 9 avec une modularisation pleinement supportée. Néanmoins, l'heure tourne et 2016 est là. Les récentes hésitations quant à la direction que doit prendre le projet laissent planer un doute certain sur une prise en compte dès Java 9 ! Affaire à suivre ...

Interconnecter vos objets avec Node.js 1ère partie

En permettant le développement d'applications côté serveur en Javascript, langage connu pour sa grande souplesse, Node.js offre la possibilité de manipuler les couches hautes comme les couches basses. Ce sont ces dernières qui permettent de prendre le contrôle des objets et de créer entre eux des interactions. Après avoir défini un environnement adéquat, on peut alors obtenir toutes les interactions induites par la potentialité des objets.



Matthieu Bouilloux
Développeur web et applicatif. Intéressé par tout type de nouvelle technologie tel que l'Iot ou encore l'analyse des marchés boursier.
http://www.katlea-edition.com
matthieu@katlea-edition.com

Il s'ensuit que l'on peut prendre le contrôle du protocole Bluetooth dans sa version Low Energy (4.0) qui tend à devenir omniprésente. Mais on peut aussi envisager de créer des applications plus exotiques qui permettront d'accéder à la sortie audio, au contrôle de l'affichage vidéo par Javascript, ou encore, là où vous mènera votre imagination.

En Node.js, on peut donc créer de la communication entre objets connectés. J'ai ainsi interfacé plusieurs cartes de prototypage via les couches basses du Bluetooth à un bracelet Myo (qui n'est autre qu'un électromyogramme). Mais les possibilités d'interactions ne sont pas limitées, car on peut déjà échanger entre objets connectés, ordinateurs, ou périphériques, voire même synchroniser le tout sur un serveur Web.

Dans cet article

Les trois premières parties expliquent comment prendre en main les cartes Arduino Yùn, Intel Edison et Raspberry Pi, avec Node.js et un peu de Bluetooth. Ensuite un premier DIY pour s'approprier la sortie vidéo du Raspberry Pi 2. Suivi d'une initiation au mappage Bluetooth à l'aide d'un bracelet Myo. Pour terminer un DIY pour créer des modules multi-room pour enceintes avec télécommandes à l'aide d'un Raspberry Pi 2 et d'un Arduino Yùn.

MISE EN ROUTE DE L'ARDUINO YÙN

L'Arduino Yun est une carte de développement / prototypage qui supporte une distribution Linux basée sur OpenWRT et se nommant Linino.

Cette carte embarque un microcontrôleur basé sur une ATmega32u4 offrant toutes les possibilités d'interaction propres aux cartes Arduino standards (capteurs ...). Elle intègre aussi un processeur Atheros AR9331 sur lequel tourne une distribution Linux. Elle constitue ainsi un outil de prototypage parfait pour l'internet des objets !

Mise en route

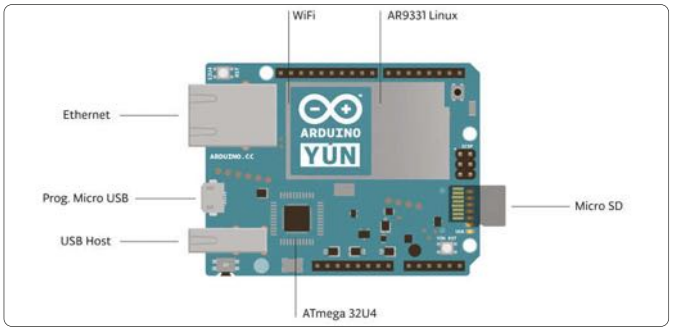
Pour exploiter au mieux le Yùn, il est nécessaire de faire quelques manipulations pour étendre l'espace disque et la mémoire vive. En effet le Yùn ne dispose que de 16 Mb de mémoire flash et 64 Mb (DDR2) de mémoire vive.

Prérequis

- Une Arduino Yùn avec la dernière image de OpenWrt (1.5.3)
- Le dernier logiciel Arduino : https://www.arduino.cc/en/Main/Software
- Une carte Micro-SD
- Le logiciel Putty http://www.putty.org/

Trouver le port du Yùn

Une fois le Yùn connecté en USB à l'ordinateur, il faut ouvrir le "Gestionnaire de périphériques", et se rendre sur la section "Ports (COM et LPT)". Vous devriez voir une entrée : "Arduino Yun COMX" où X est le port du Yun.



OpenWrt upgrade

Dans le logiciel Arduino, il faut choisir l'Arduino Yùn ainsi que le port correspondant.

- Outils > Type de carte > Arduino Yùn
- Outils > Port > COMX (Arduino Yùn)

Ensuite il faut charger l'exemple YunSerialTerminal sur le Yun pour éviter les conflits.

- Fichier > Exemples > Bridge > Yùn Serial Terminal

Puis téléverser :

- Croquis > Téléverser

Pour la suite il suffit de copier l'image de OpenWrt à la racine de la micro-sd et redémarrer le Yùn. Pour upgrader le Yùn il suffit de se connecter via WiFi à l'Arduino pour accéder à l'interface Web du Yùn : http://arduino.local (mot de passe par defaut : "arduino"). Une fois entrée sur l'interface, une option upgrade (reset) apparaît ! Le processus dure environ 3 minutes. Une fois terminée, la led WLAN arrête de clignoter.

Etendre l'espace disque à la Micro-SD

Attention l'Arduino doit être connectée à Internet !

Il nous faut le sketch correspondant pour commencer :

https://www.arduino.cc/en/uploads/Tutorial/YunDiskSpaceExpander.zip

Une fois le sketch téléversé, ouvrez le terminal : Outils > Moniteur série

Vérifiez que l'option "Nouvelle ligne" et "115200 Baud" sont sélectionnées.

Un message doit apparaître, répondre "yes" trois fois puis entrez la taille en Mo de l'espace de données; le reste sera attribué au système Linino.

Appuyez sur le bouton Yun RST pour redémarer Linino, lorsqu'on vous le demande dans le terminal. Commande pour vérifier l'espace disque :

```
df -h
```

Etendre la mémoire vive en créant un swap

Connectez-vous au Yùn via SSH par WiFi ou rechargez l'exemple Yùn Serial Terminal.

- Par SSH connectez-vous sur arduino.local (via putty).
- Par USB choisir "serial" dans Putty, mettre "speed" à 115200 et Serial line à COMX où X est le port du Yùn.

Créer le Swap

Ici nous créons un swap de 512 Mo que Linino pourra utiliser en cas de dépassement de la mémoire vive.

```
mkdir swap
dd if=/dev/zero of=/swap/yunswapfile bs=1M count=512
```

```
mkswap /swap/yunswapfile
swapon /swap/yunswapfile
```

Configurer le swap au démarrage

On ajoute le swap au système de fichiers :

```
uci add fstab swap
uci set fstab.@swap[0].device=/swap/yunswapfile
uci set fstab.@swap[0].enabled=1
uci set fstab.@swap[0].fstype=swap
uci set fstab.@swap[0].options=default
uci set fstab.@swap[0].enabled_fsck=0
uci commit
```

On édite le fichier de démarrage :

```
nano /etc/rc.local
```

Puis on rajoute avant "Exit 0" :

```
/etc/init.d/fstab start
```

Et voilà pour le swap. Pour tester il suffit de taper la commande suivante :

```
free -m
```

Installation de Node.js et de sftp

Systématiquement mettre à jour les paquets à chaque redémarrage :

```
opkg update
```

Puis installer les paquets pour Node.js et pour le serveur sftp afin de faciliter les transferts.

```
opkg install openssh-sftp-server
opkg install node
opkg install node-ws
opkg install node-serialport
```

Cylon.js

Le paquet Cylon.js permet de dialoguer en Javascript avec les composants de l'Arduino. Créer un dossier pour y installer cylon.js

```
npm install cylon cylon-gpio cylon-i2c
```

Installer cylon-firmata sur votre ordinateur puis copier le dossier sur le Yùn dans le répertoire node_modules de votre application.

```
npm install cylon-firmata
```

Une fois copié, il faut retirer le module serialport déjà installé sur le système.

```
rm -rf ./node_modules/cylon-firmata/node_modules/firmata/node_modules/serialport
```

Puis désactiver le bridge :

```
nano /etc/inittab

> commenter avec # la ligne:
> ttyATH0::askfirst:/bin/ash --login
```

Puis installer via l'application Arduino le sketch suivant pour dialoguer avec Cylon : https://gist.github.com/edgarsilva/e73c15a019396d6aaef2
Il ne vous reste plus qu'à tester :

```
var Cylon = require('cylon');

Cylon.robot({
  connections: {
    arduino: { adaptor: 'firmata', port: '/dev/ttyATH0' }
  },

  devices: {
    led: { driver: 'led', pin: 4 },
    button: { driver: 'button', pin: 3 }
  },

  work: function(my) {
    my.button.on('push', function() {
      my.led.toggle()
    });
  }
}).start();
```

A propos du Bluetooth Low Energy

A l'heure actuelle l'installation d'un dongle Bluetooth est possible (reconnu par hciconfig après quelques manipulations) mais l'établissement de la connexion entre appareils Bluetooth ne se fait pas malgré le fait que le scan fonctionne sans problème.

MISE EN ROUTE DE L'INTEL EDISON

La mise en route de l'Edison est beaucoup plus aisée, car Intel l'a mise à disposition à l'adresse suivante :
https://software.intel.com/fr-fr/articles/assemble-intel-edison-on-the-arduino-board

Le Bluetooth Low Energy

Ajouter les repositories pour les paquets :

```
vi /etc/opkg/base-feeds.conf

> touche : "i" (insérer)
> Puis ajoutez les lignes suivantes :
>
src/gz all http://repo.opkg.net/edison/repo/all
src/gz edison http://repo.opkg.net/edison/repo/edison
src/gz core2-32 http://repo.opkg.net/edison/repo/core2-32

> touche Echap (quitter insérer)
> touche : ":w" (écrire)
> touche : ":q" (quitter)
```

Installez Bluez après avoir mis à jour les paquets :

```
opkg update
opkg install bluez5-dev
```

On installe nano au passage pour être plus à l'aise avec l'éditeur de texte :

```
opkg install nano
```

On active le Bluetooth Low Energy

```
rfkill unblock bluetooth
// De manière temporaire (nécessairre à chaque redémarrage)
killall bluetoothd
// De manière permanente
systemctl disable bluetooth
// On active l'adaptateur
hciconfig hci0 up
```

Node.js

Déjà présent avec la mise à jour, il ne vous reste plus qu'à utiliser npm pour installer vos paquets. Aucune action spéciale n'est donc requise !

MISE EN ROUTE DU RASPBERRY PI 2

Le Raspberry Pi est une "single-board computer" dotée entre autres d'une sortie video Full HDMI. Si cet ordinateur de poche permet la mobilité, il faut cependant faire attention à la consommation générée par la gestion de l'affichage.

Pré-requis

- La dernière image de Raspbian (Jessie) : http://raspberrypi.org/downloads
- Une carte Micro-SD
- Un dongle WiFi
- Un dongle Bluetooth 4.0 (BLE)
- Le logiciel Putty http://www.putty.org/

J'ai utilisé deux dongles (WiFi & Bluetooth) du fabricant Belkin.
Formater la carte SD et y installer Raspbian Jessie
La procédure est relativement simple, surtout sous Windows.
Je vous laisse donc suivre la procédure à l'adresse suivante :
https://www.raspberrypi.org/documentation/installation/installing-images/

Configurer les interfaces réseaux et les outils de base

Une fois Raspbian installée, vous devez alors explorer la carte Micro-SD pour vous rendre dans le dossier de configuration nommé etc

Pour le réseau filiaire (RJ-45)

Ouvrez le dossier network et modifiez le fichier interfaces avec n'importe quel éditeur de texte. Remplacez iface eth0 inet manual par iface eth0 inet dhcp. Insérez la carte Micro-SD dans le Pi 2, puis démarrez-le et branchez-le à votre box.

Pour le WiFi (dongle USB)

Ouvrez le dossier network et modifiez le fichier interfaces avec n'importe quel éditeur de texte. Remplacez iface wlan inet manual par iface wlan inet dhcp. Puis vérifiez que la ligne ci-dessous (indiquant la configuration des réseaux WiFi) est bien présente :

```
wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf
```

Dans la majorité des cas nous sommes en authentification WPA. Il faut modifier le fichier "wpa_supplicant.conf" et ajouter :

```
ap_scan=1

network={
   ssid="Nom de votre réseau wifi (ssid)"
   proto=WPA RSN
   key_mgmt=WPA-PSK
   psk="Password de votre réseau wifi"
}
```

Vous pouvez ajouter autant de networks que désiré.

Se connecter en SSH ou SFTP au Pi 2

Pour retrouver facilement votre Pi 2, il suffit de vous connecter à raspberrypi.local.
Remarque : sous Windows vous devez avoir le logiciel "bonjour" de Apple installé (lien ci-dessous). https://support.apple.com/kb/dl999
Via le logiciel Putty, remplir :

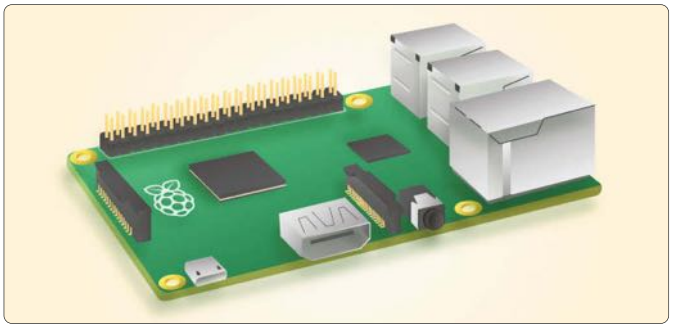
```
Hostname : raspberrypi.local
Connection type : SSH
Port : 22
```

Pour accéder au fichier du Raspberry, on peux utiliser le SFTP (SSH over FTP) avec la même configuration et un client SFTP (tel WinSCP sous Windows).

Etendre l'espace de stockage

```
sudo raspi-config
```

Puis sélectionner la première option.



Installer Node.js

Mettre à jour les paquets :

```
sudo apt-get update
```

Télécharger et installer Node.js :

```
wget http://node-arm.herokuapp.com/node_latest_armhf.deb
sudo dpkg -i node_latest_armhf.deb
```

Installer le Bluetooth Low Energy (dongle USB)

On installe le bluetooth avec Bluez :

```
sudo apt-get install bluetooth bluez libbluetooth-dev libudev-dev
```

On branche le dongle USB, puis on vérifie que tout est ok.

```
sudo hciconfig

>_ Résultat :
hci0:  Type: BR/EDR  Bus: USB
      BD Address: 5C:F3:70:6D:08:81  ACL MTU: 1021:8  SCO MTU: 64:1
      UP RUNNING
      RX bytes:652 acl:0 sco:0 events:40 errors:0
      TX bytes:954 acl:0 sco:0 commands:40 errors:0
```

Ensuite on redémarre le Bluetooth :

```
sudo service bluetooth restart
sudo hciconfig hci0 up
```

On teste le scan du Bluetooth :

```
sudo hcitool lescan

>_ Résultat :
LE Scan ...
46:E9:1B:42:B0:68 (unknown)
46:E9:1B:42:B0:68 (unknown)
6B:F2:C9:23:E5:96 (unknown)
57:49:93:2B:BE:6A (unknown)
57:49:93:2B:BE:6A (unknown)
4C:91:BE:F4:31:B3 (unknown)
...
```

Petit plus

Pour accéder à l'interface graphique du Pi 2 sans écran on peut installer x11vnc
On le configure avec un mot de passe :

```
sudo apt-get install x11vnc
x11vnc -storepasswd "VOTRE MOT DE PASSE" ~/.vnc_passwd
x11vnc -many -rfbauth ~/.vnc_passwd
```

Il suffit maintenant de lancer un client VNC sur raspberrypi.local

La suite dans Programmez! 193

MEAN.IO (4/4)

Voici notre dernier article de la série consacrée à MEAN.IO, après la présentation générale du framework lors du premier, et celles des concepts permettant de créer un nouveau module dans le second, nous nous sommes attardés sur un cas d'application concret : celui de la visualisation de données géographiques. Depuis le départ, nous souhaitons en effet créer une application permettant de visualiser un ou plusieurs itinéraires GPS (type randonnée VTT ou pédestre). Nous avons défini précédemment le modèle de données permettant de stocker l'information en base, ainsi que l'API REST permettant de le manipuler. Ensuite, nous avons créé une partie cliente incluant des interfaces homme-machine (IHM) pour la présentation et l'édition de ces données. Il ne nous restait qu'à ingérer et visualiser nos chemins sous formes de cartes à la façon de "Google Maps" (objet du précédent article) ou de vues 3D à la façon de "Google Earth" (objet du présent article).

Luc CLAUSTRES
Consultant indépendant
Créateur d'Applications Numériques
http://www.digital-innovation.fr

MISE À JOUR DE L'APPLICATION

Mise à jour des routes

Aux routes côté front-end, qui sont gérées via l'AngularUI Router, nous en ajoutons une nouvelle dans le fichier **ApplicationRoutes.js** du dossier **routes** de la partie publique ; il s'agit de la déclaration permettant d'accéder à la page pour afficher la vue 3D d'un chemin :

```
// Page permettant de voir un chemin sur la carte
$stateProvider.state('track globe', {
  url: '/track/:trackId/globe',
  templateUrl: '/application/views/TrackGlobe.html',
  controller: 'TrackGlobeController',
  resolve: {
    loggedin: function(MeanUser) {
      return MeanUser.checkLoggedin();
    }
  }
});
```

Mise à jour des vues

Pour la présentation des chemins (sous forme de liste ou sous forme unitaire) nous avions utilisé précédemment des panels Bootstrap. L'en-tête (classe CSS *panel-heading*) contenait notamment les actions réalisables par l'utilisateur (e.g. éditer ou effacer) sous forme d'icônes, auxquelles nous rajoutons la possibilité d'ouvrir la vue 3D (Fig.1) dans **public/views/Track.html** :

```
<!-- Actions associés au chemin (effacer, éditer, vue 3D et vue carte) -->
<a data-ng-click="remove(track)"><i class="pull-right glyphicon glyphicon-trash"
> </i></a>
<a href="/track/{{track._id}}/edit"><i class="pull-right glyphicon glyphicon-edit"
> </i></a>
<a href="/track/{{track._id}}/globe"><i class="pull-right glyphicon glyphicon-globe"
> </i></a>
</div>
```

VUE 3D

Plusieurs librairies Open Source permettent de visualiser des données géographiques sur un globe virtuel 3D comme GlobWeb ou WebGLEarth mais j'ai choisi d'utiliser Cesium pour sa large communauté et sa simplicité (en version 1.5 lors du codage). Pour ce faire il faut rajouter "cesium": "1.5" dans le fichier bower.json à la racine de votre dossier applicatif et exécuter bower install. Ensuite il faudra vous rendre dans le dossier d'installation et exécuter ant pour générer la version minifiée des sources. Ensuite il faut modifier le **app.js** du module applicatif pour rajouter les fichiers nécessaires à l'agrégation (voir article précédent) :

```
Application.aggregateAsset('css', '../../../../../bower_components/cesium/Build/Cesium
Unminified/Widgets/widgets.css');
Application.aggregateAsset('js', '../../../../../bower_components/cesium/Build/Cesium
Unminified/Cesium.js', {weight: -1});
```

Services

Cesium étant une librairie JavaScript "standard" je l'ai tout d'abord encapsulée dans un service afin de pouvoir l'injecter dans d'autres composants AngularJS. Ceci permet de conserver une injection de dépendance à la mode AngularJS sans accéder de façon directe à un objet global :

```
// Service utilisé pour accéder à la librairie Cesium via l'injection de dépendances
angular.module('mean.application').factory('Cesium', [ function() {
  return window.Cesium; // Assume que la librairie est déjà chargée dans la page
}
]);
```

Ensuite, j'ai créé un service dédié à la création du chemin suivi en 3D. En effet, pour représenter une animation 3D, Cesium se base sur un format interne nommé CZML qu'il nous faudra donc générer. De plus, il faut pouvoir convertir le chemin depuis le système géodésique utilisé par le GPS vers un repère 3D utilisé par Cesium ; en préambule je vous propose de découvrir ces différents systèmes de coordonnées qui devront être manipulés (Fig.2).

Systèmes de coordonnées

Afin de positionner un objet sur la Terre il est nécessaire de lui attribuer des coordonnées dans un repère lié à la Terre. Un tel repère doit donc être défini, et le cas échéant complété d'une représentation de la Terre, pour

Fig.1

Ajout dans la vue listant les chemins les actions d'accès à la vue cartographique ou 3D

qu'une action de positionnement puisse être menée. Il existe aujourd'hui un grand nombre de systèmes de références de coordonnées dont je présenterai les deux principaux.

Système de coordonnées cartésiennes

Un système de référence terrestre (SRT) est un repère cartésien tridimensionnel (OXYZ) que l'on positionne par rapport à la Terre de telle sorte que :

- L'origine O est le gravité de la Terre ;
- L'axe OZ est l'axe de rotation de la Terre ;
- Le plane OXZ est le plan méridien origine ;
- Le plan OXY est le plan de l'équateur.

Un point de la croûte terrestre est considéré comme fixe dans un tel système car le repère "tourne" en même temps que la Terre. Il est repéré par ses coordonnées cartésiennes géocentriques tridimensionnelles : X, Y, Z. Un SRT est également appelé Système de Référence Géodésique ou encore "Earth-Centered, Earth-Fixed" (ECEF).

Système de coordonnées géodésiques

Comme il est relativement complexe de repérer un point sur Terre via ses coordonnées cartésiennes, est associé à un SRT un ellipsoïde de révolution qui est un modèle mathématique de la Terre débarrassée de ses reliefs. Il s'agit approximativement d'une sphère aplatie aux pôles qui est une simplification du géoïde : la surface équipotentielle de référence du champ de pesanteur terrestre. Dans un système géodésique ainsi défini, un point est localisé par ses coordonnées géographiques (ou géodésiques), exprimées en valeurs angulaires par la latitude, la longitude, et la hauteur géodésique h mesurée suivant la normale à l'ellipsoïde (h est petit à proximité de la surface terrestre). Le système géodésique le plus utilisé dans le monde est le système WGS 84, associé au système de positionnement GPS.

Génération du chemin en 3D

Dans Cesium une position en système de coordonnées géodésique est un objet de type Cesium.Cartographic et une position 3D un objet de type Cesium.Cartesian3. Nous écrivons dans notre service une première fonction permettant de transformer notre chemin stocké en base au format géographique vers un chemin en coordonnées 3D ; la transformation aura lieu "en place" puisque chaque point possède trois coordonnées quel que soit le système (X, Y et Z ou longitude, latitude et altitude) :

```
// Service utilisé pour accéder à l'API REST des chemins
angular.module('mean.application').factory('TrackGenerator', ['Cesium',
function (Cesium) {
  // Conversion depuis tableau de coordonnées géographiques vers cartésiennes
  TrackGenerator.cartographicToCartesian = function(waypoints) {
    for (var i = 0; i < waypoints.length / 3; i++) {
      var cartographicPosition = Cesium.Cartographic.fromDegrees(waypoints[3*i], waypoints
[3*i+1], waypoints[3*i+2]);
      var position = new Cesium.Cartesian3();
      Cesium.Ellipsoid.WGS84.cartographicToCartesian(cartographicPosition, position);

      waypoints[3*i] = position.x;
      waypoints[3*i+1] = position.y;
      waypoints[3*i+2] = position.z;
    }
  }
}]);
```

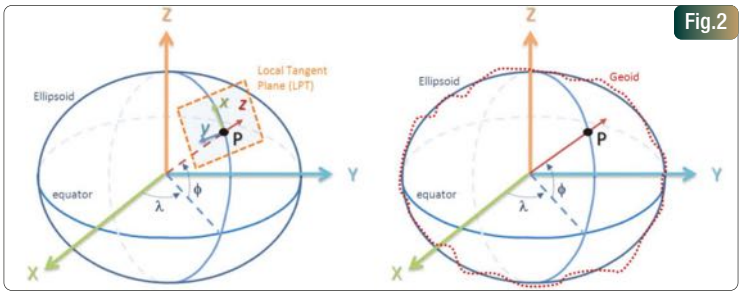


Fig.2

Système de coordonnées cartésiennes Earth-Centered Earth-Fixed (ECEF) avec son plan tangent local (à gauche), système de coordonnées géodésique (à droite) et différence entre ellipsoïde (à gauche) et géoïde (à droite)

Ensuite nous ajoutons une nouvelle fonction qui utilisera la première et générera l'animation 3D au format CZML à partir du chemin. La partie délicate consiste à affecter à chaque point du chemin un temps pour créer une animation qui soit réaliste. En effet, le GPS échantillonne la position à une fréquence fixe (par exemple un point toutes les 5 secondes) et il manque donc de l'information entre deux points d'échantillonnage pour avoir un mouvement continu. Pour éviter d'obtenir des "sauts" entre les positions lors de la visualisation 3D (ce qui semblerait peu réaliste) la position entre deux échantillons est "interpolée". L'interpolation numérique est une opération mathématique permettant de construire une courbe continue à partir d'un nombre fini de points. Cesium supporte plusieurs types d'interpolation allant de la plus simple qui est l'interpolation linéaire (la trajectoire entre deux points est supposée être une ligne droite) à des formes plus complexes telles que l'interpolation Lagrangienne que j'utilise et qui est basée sur un calcul polynômial permettant de représenter un mouvement de façon plus réaliste. Nous supposons une vitesse de base de 90 km/h pour rejouer l'animation dans un temps "raisonnable", le parcours réel pouvant avoir pris plusieurs heures. Dans l'animation 3D nous définissons une entité (i.e. le "véhicule") qui suivra le chemin ainsi créé et l'icône d'un véhicule pour représenter notre position, le code de production du CZML est le suivant :

```
// Génère une animation 3D au format CZML à partir d'un chemin en coordonnées cartographiques
TrackGenerator.generateCzml = function(waypoints) {
  // La scène CZML contient :
  // - l'icône qui représentera le véhicule (nommé 'VehicleIcon')
  // - le chemin suivi par le véhicule et utilisé pour interpoler sa position (nommé 'Vehicle')
  // Le chemin est donné comme un tableau de coordonnées cartographiques, donc sans
référence par rapport au sol (i.e. altitude).
  // La position réelle au niveau du sol sera calculée en temps-réel lors de l'animation.
  var builtInCzml = [ ... ]; // Voir code complet pour détails

  var cartesianRoute = waypoints.slice();
  this.cartographicToCartesian( cartesianRoute );

  // Définit la date et l'heure de départ
  var epoch = Cesium.JulianDate.now();
  var previousPosition;
  var timeStep = 0;

  // L'animation 3D est définie sous la forme d'une liste de temps/position
  for (var i = 0; i < waypoints.length / 3; i++) {
    var position = new Cesium.Cartesian3(cartesianRoute[3*i], cartesianRoute[3*i+1],
cartesianRoute[3*i+2]);

    if ( Cesium.defined( previousPosition ) )
    {
      var distance = Cesium.Cartesian3.distance(previousPosition, position);
      // Calcul du pas de temps, on considère une vitesse de 90 km/h soit environ 25 m/s
      timeStep += distance / 25;
```

```
    }
    // Pas de temps
    builtInCzml[2].position.cartesian[4*i] = timeStep;
    // Position
    builtInCzml[2].position.cartesian[4*i+1] = position.x;
    builtInCzml[2].position.cartesian[4*i+2] = position.y;
    builtInCzml[2].position.cartesian[4*i+3] = position.z;

    previousPosition = position;
  }

  // Calcul de la date et l'heure de fin
  var end = new Cesium.JulianDate();
  Cesium.JulianDate.addSeconds(epoch, timeStep, end);
  var timeInterval = new Cesium.TimeInterval({
    start : epoch,
    stop : end,
    isStartTimeIncluded : true,
    isStopTimeIncluded : true
  });
  builtInCzml[2].position.epoch = epoch.toString();
  builtInCzml[2].availability = timeInterval.toString();

  return builtInCzml;
};
```

Directive

Il n'existe pas de directive AngularJS réellement "officielle", j'ai donc créé une directive permettant d'instancier une vue 3D Cesium sur un élément HTML (on parle de *Viewer*). Elle configure par défaut une couche de données image et une couche topographique pour disposer d'un terrain en 3D, l'ensemble des options possibles est détaillé sur https://cesiumjs.org/Cesium/Build/Documentation/Viewer.html. Enfin, elle rajoute au scope une propriété viewer qui référence l'instance Cesium de la vue 3D et permet d'accéder au contenu 3D dans viewer.scene comme le point de vue courant viewer.scene.camera ou le globe virtuel 3D viewer.scene.globe :

```
directive("cesium", ['Cesium', function (Cesium) {
 return {
  restrict: "E",
  controllerAs: "TrackGlobeController",
  link: function (scope, element, attributes) {
   var options = {
    sceneMode : Cesium.SceneMode.SCENE3D,
    sceneModePicker : false,
    scene3DOnly : true,
    homeButton : false,
    geocoder : false,
    navigationHelpButton : true,
    baseLayerPicker : false
   }
   scope.viewer = new Cesium.Viewer(element[0], options);
   // Définition des fournisseurs de données image/terrain à utiliser
   scope.viewer.imageProvider = new Cesium.BingMapsImageryProvider({
     url : '//dev.virtualearth.net'
   });
   scope.viewer.terrainProvider = new Cesium.CesiumTerrainProvider({
     url : '//assets.agi.com/stk-terrain/world'
   });
  }
 }
}]);
```

Contrôleur

Le contrôleur permet la mise en musique du service back-end et des éléments front-end :

- Récupération du chemin suivi via l'API REST (de façon similaire à la carte 2D du précédent article) ;
- Génération de l'animation 3D au format CZML via le service précédent ;
- Chargement de l'animation 3D dans Cesium ;
- Récupération de l'entité et de son icône ;
- Exécution de l'animation 3D ;
- Suivi de l'entité.

L'essentiel du code se concentre sur cette dernière partie, les autres reposant sur des fonctions Cesium existantes. En effet, par défaut, le chemin suivi ne contient pas forcément l'altitude des points, ou s'il la contient elle n'est pas forcément adaptée à la résolution des données du terrain chargé dans Cesium. Ainsi, l'entité pourrait se retrouver aléatoirement positionnée au-dessus ou en-dessous du terrain 3D, suivant la précision des données terrain, ce qui peut poser des problèmes lors de la visualisation. Aussi nous allons recalculer en temps-réel la position 3D au niveau du sol. Pour ce faire nous utiliserons la technique du lancer de rayon qui permet de calculer l'intersection entre une demi-droite dans l'espace (définie par une origine et une direction) et un objet 3D (en l'occurrence le terrain). A partir de la position à altitude zéro (point rouge de la Figure 3) nous calculons l'origine du rayon de façon à ce qu'il soit au-dessus du terrain en lui affectant une altitude supérieure à toutes les montagnes connues. Ensuite nous calculons la direction du rayon comme le vecteur unitaire allant de cette origine à la position sur l'ellipsoïde, et donc pointant vers le centre de gravité terrestre. Enfin, nous demandons à Cesium de calculer l'intersection entre le rayon et la géométrie 3D du terrain (point vert de la Fig.3). Une fois cette position déterminée nous demandons à Cesium de calculer le repère local tangent à l'ellipsoïde qui permet d'être orienté de façon fixe par rapport au pôle Nord. Nous "attachons" alors le point de vue, i.e. la caméra virtuelle de la scène 3D, à ce repère pour que l'on ait l'impression que la caméra suive l'entité sur le chemin. Au final le code du contrôleur est le suivant :

```
// Contrôleur utilisé pour afficher un chemin sur un globe 3D
angular.module('mean.application').controller('TrackGlobeController', ['$scope', '$http',
'$stateParams', 'TrackService', 'TrackGenerator',
function($scope, $http, $stateParams, TrackService, TrackGenerator) {
 // Projette au sol la position donnée
 var clampToGround = function(position) {
  // Position au sol à altitude 0
  var cartographicPosition = new Cesium.Cartographic();
  Cesium.Ellipsoid.WGS84.cartesianToCartographic(position, cartographicPosition);
  // Origine du rayon à la même position mais au-dessus de tous les sommets existants
  var cartographicOrigin = Cesium.Cartographic.clone(cartographicPosition);
  cartographicOrigin.height = 20000; // Everest ~8000m
  // Calcul du point 3D origine
  var origin = new Cesium.Cartesian3();
  Cesium.Ellipsoid.WGS84.cartographicToCartesian(cartographicOrigin, origin);
  // Calcul de la direction du rayon : de l'origine vers le centre de la Terre
  var direction = new Cesium.Cartesian3();
  Cesium.Cartesian3.subtract(position, origin, direction);
  Cesium.Cartesian3.normalize(direction, direction);
  var ray = new Cesium.Ray(origin, direction);

  var groundPosition = new Cesium.Cartesian3();
```

```
    // Intersection avec le sol
    return $scope.viewer.scene.globe.pick(ray, $scope.viewer.scene, groundPosition);
  }

  // Recherche l'entité dans le contenu 3D
  var getTrackedEntity = function() {
    if (Cesium.defined($scope.lookAt)) {
      $scope.trackedEntity = $scope.viewer.dataSources.get(0).entities.getById($scope.lookAt);
      if (Cesium.defined($scope.trackedEntity)) {
        // Vitesse de défilement par défaut
        $scope.viewer.clock.multiplier = 1;
        // Récupérer l'icône représentant l'entité
        $scope.trackedEntityIcon = $scope.viewer.dataSources.get(0).entities.getById($scope.
lookAt + 'Icon');
      }
    }
  }

  // Charger l'objet source au format Czml
  // L'argument lookAt est le nom de l'entité à suivre dans les données source
  var loadSource = function(source, lookAt) {
    $scope.lookAt = lookAt;
    // Chargement de l'animation 3D
    var dataSource = new Cesium.CzmlDataSource();
    dataSource.load(source, 'Built-in CZML');
    $scope.viewer.dataSources.add(dataSource);
    // Récupération de l'entité suivant le chemin
    getTrackedEntity();
    // Positionnement du callback appelé à chaque pas d'exécution de l'animation
    $scope.viewer.clock.onTick.addEventListener(function(clock) {
      if ( Cesium.defined($scope.trackedEntity) ) {
        // Récupération de la position courante
        var cartographicPosition = new Cesium.Cartographic();
        var position = $scope.trackedEntity.position.getValue(clock.currentTime);
        Cesium.Ellipsoid.WGS84.cartesianToCartographic(position, cartographicPosition);

        // Calcul de la position projetée au sol
        var groundPosition = clampToGround(position);
        if ( Cesium.defined(groundPosition) ) {
          // Mise à jour de la position 3D en conséquence
          position = groundPosition;
          Cesium.Ellipsoid.WGS84.cartesianToCartographic(groundPosition, cartographicPosition);
          // Et positionnement de l'icône
          if ( Cesium.defined($scope.trackedEntityIcon) ) {
            $scope.trackedEntityIcon.position = new Cesium.ConstantPositionProperty(ground
Position, Cesium.ReferenceFrame.FIXED);
            var text = cartographicPosition.height.toFixed() + ' m';
            $scope.trackedEntityIcon._label._text = new Cesium.ConstantProperty(text);
          }
        }
```

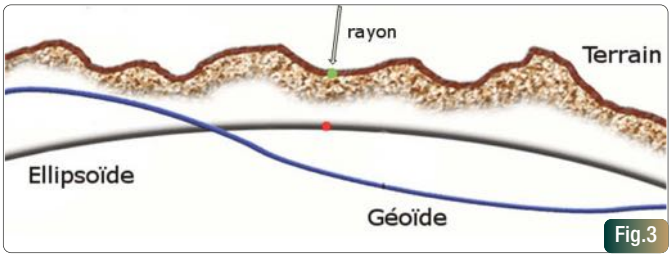


Fig.3

Lancer de rayon pour calculer la position sur le terrain 3D (point vert) à partir de la position géographique sur l'ellipsoïde (point rouge)

```
        // Calcul du repère local à la position
        var transform = Cesium.Transforms.eastNorthUpToFixedFrame(position);
        // Attachement du point de vue à ce repère
        Cesium.Matrix4.clone(transform, $scope.viewer.scene.camera.transform);
        $scope.viewer.scene.camera.constrainedAxis = Cesium.Cartesian3.UNIT_Z;
      }
    });
  }

  //Récupère un chemin via son ID
  $scope.findOne = function() {
    TrackService.get({
      trackId: $stateParams.trackId
    }, function(track) {
      $scope.track = track;

      // Position par défaut de la caméra
      $scope.viewer.scene.camera.lookAt(
        new Cesium.Cartesian3(-1500.0, -1500.0, 500.0),
        Cesium.Cartesian3.ZERO,
        Cesium.Cartesian3.UNIT_Z);
      loadSource( TrackGenerator.generateCzml($scope.track.waypoints), "Vehicle");
    });
  };
}
]);
```

Vue

La vue est la partie la plus simple car elle se contente d'instancier la directive et de requêter le chemin en faisant appel au contrôleur :

```
<div data-ng-init="findOne()">
  <!-- Ajout d'un globe 3D -->
  <cesium/>
</div>
```

Fig.4

CONCLUSION

Cet article a été l'occasion de découvrir le domaine de la cartographie 3D et d'approfondir à nouveau votre connaissance du framework MEAN.io en le mettant en pratique sur un cas concret d'utilisation via l'intégration de librairies externes. J'espère que vous serez maintenant prêts à faire vos armes en développant vos propres modules ou applications. Accès au code complet de l'article sur https://github.com/claustres/meanio-tutorial-3.

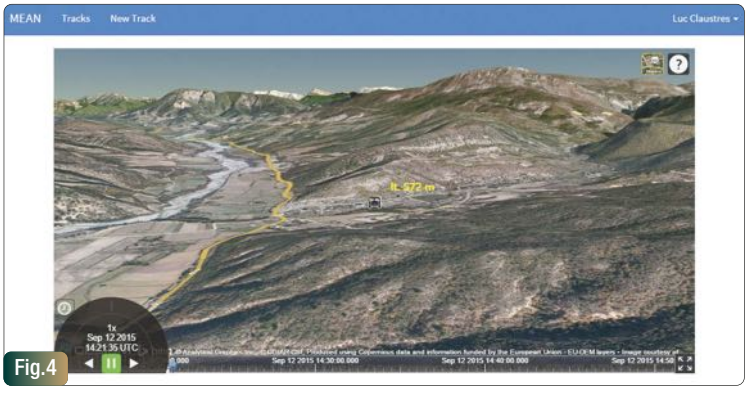


Fig.4

Vue 3D animée d'un chemin où le marqueur suit le parcours, grâce à la barre "magnétoscope" en bas il est possible d'accélérer ou de se déplacer dans le temps

L'art de l'architecture en Java 2e partie

On va mettre l'accent sur les étapes à suivre pour aboutir à une bonne architecture, solide, robuste et maintenable. Pour commencer, on va voir comment bien choisir les technologies. Ensuite on va examiner l'architecture en étudiant les différents outils à utiliser dans un projet. Puis on va voir les techniques et les stratégies à mettre en place pour bien développer. Enfin on va voir comment maintenir le code à travers les tests unitaires et les tests d'intégrité.

Bouhanef Hamdi
Développeur full-stack chez **sfeir**
https://fr.linkedin.com/in/bouhanef-hamdi-62386051

Dridi Manel
Freelance web
https://www.linkedin.com/in/manel-dridi-ep-bouhanef-908316105

MISE EN PLACE DE L'USINE DE DÉVELOPPEMENT

Après avoir choisi les technologies à utiliser dans l'application, on va voir plus en détail le choix des outils qu'on va utiliser pour gérer le bon fonctionnement des différentes plateformes et l'automatisation des tâches récurrentes comme le build des applications. Avant de commencer, un petit rappel des différents environnements s'impose.

Intégration continue

Cela dépend de la stratégie du projet et des équipes, mais la logique ici c'est que, pour les développeurs, on a un environnement buildé. Généralement cela est fait pour chaque commit. Et pour chaque build, les tests sont passés pour vérifier si le commit est conforme aux tests.

Livraison continue

Le but ici est d'avoir une version pour la production ou pour la recette, toute version de l'intégration continue peut être livrée. De même cette livraison dépend des environnements, des sprints, si en mode agile...

Déploiement continu

Le déploiement continu est le mécanisme qui permet d'avoir un environnement pour tester l'application, après l'avoir testée et déployée. En général ces environnements sont pour les recettes ou la production.

Exécution automatique des tâches

Il est très utile de configurer l'exécution des tâches en automatique, comme on vient de voir tout à l'heure, pour l'intégration, livraison ou le déploiement continu, on aura besoin de les automatiser. Il y a plusieurs outils sur le marché, mais les plus utilisés sont les suivants :

Apache Continuum

Il intègre la plupart des fonctionnalités attendues d'un tel outil comme le support des outils de construction, système de gestion de versioning, notifications...

Jenkins (Hudson)

Sans doute l'outil le plus utilisé, cet outil est très puissant, il gère la plupart des systèmes de gestion de versioning, il gère aussi les tests avec plusieurs outils et il s'intègre aussi facilement avec Sonar...

Vérification du code

Il y a plusieurs outils qui permettent de vérifier la qualité du code et les tests :

FindBugs

Cet outil permet de chercher les bugs éventuels qui peuvent être levés à l'exécution.

PMD

Cet outil vérifie la qualité du code, les codes morts, les imbrications trop compliquées...

CheckStyle

Il permet de vérifier le style du code écrit s'il est conforme ou pas. Par exemple il vérifie les commentaires, Convention de nommage...

Sonar

Cet outil contient tous les outils cités plus haut (FindBugs, PMD et Checkstyle). Il offre une interface conviviale pour visualiser la qualité du code, et un reporting par projet, par package...

Système de gestion de versioning

Il est utile de partager le code entre les développeurs, d'indexer le code dans des endroits bien définis (pour arrêter une version...). La solution pour le faire est d'utiliser un système de gestion de versioning. Il y a deux types de systèmes de gestion de version :

Gestion de versions centralisée

Ici il n'y a qu'un seul dépôt de version, c'est-à-dire qu'il n'y a qu'un seul serveur qui gère le versioning. Exemple SVN, CVS...

Gestion de versions décentralisée

Dans ce type de système, tous les développeurs peuvent être le serveur de gestion de version. Exemple GIT ou Mercurial.

Les outils de build et de gestion de dépendances

Il y a plusieurs outils qui permettent d'automatiser le build des applications et gérer automatiquement les dépendances :

Apache Ant (Java)

Il a été créé par la fondation Apache; c'est un outil qui permet d'automatiser les opérations répétitives et de décrire les tâches de construction.

Apache Maven (Java)

C'est un outil qui permet d'automatiser les tâches et de gérer les dépendances d'un projet. À la différence avec Ant, c'est une convention plutôt que configuration.

Gradle (Java)

Gradle a réuni les avantages de Maven et de Ant sauf que ses fichiers de configuration ne sont pas en XML.

Grunt (JavaScript)

Grunt permet d'automatiser l'exécution des tâches pour le code JavaScript. Par exemple, zipper le code et compiler les fichiers Less.

Bower (JavaScript)

C'est un gestionnaire de paquets JavaScript.

Les branches et versions

Après avoir mis en place un système pour gérer le code, on va utiliser ce mécanisme pour créer des versions et des branches.

Les versions

Les versions sont généralement créées pour taguer le code dans un état bien défini. Par exemple quand on est en mode scrum, pour chaque sprint on peut créer une version. Et pour chaque sprint aussi on peut taguer une version...

Les branches

Les branches sont, comme leur nom l'indique, des dérivées de la branche principale. Par exemple supposons qu'on ait une application standard de gestion de parc automobile. On va développer une version spécifique pour un client, donc on crée une nouvelle branche pour y développer l'application pour le client sans pour autant toucher la version standard. En d'autres termes, comme si on avait créé un nouveau dépôt, sauf que l'origine de ce dépôt est une version précise de la branche principale.

Les types d'environnements

Il y a plusieurs environnements, suivant la stratégie à employer, qui peuvent être mis en place, nous allons citer les plus intéressants et les plus utilisés : Fig.1.

Environnement de développement

Au moment de développement de nouvelles tâches, l'environnement de développement permet d'avoir un système contenant la résolution des nouvelles tâches en continu (la mise à jour de cet environnement peut être journalière). Cela permet aux intervenants du projet de détecter plus rapidement si les développements sont conformes aux besoins.

Environnement de recette

Cet environnement est en général une version bien définie du dépôt; on crée un environnement pour tester un lot de fonctionnalités ou un module. Les développements de nouvelles fonctionnalités continuent, mais cet environnement n'est pas impacté par ces nouveautés. Cela permet de valider ces fonctionnalités et de pouvoir les livrer au final.

Environnement de production

Cet environnement est comme l'environnement de recette, c'est une version bien précise du dépôt, sauf que celui-ci est destiné aux utilisateurs finaux de l'application. Par exemple, supposons qu'on soit en train de développer un site de gestion de parc informatique, l'environnement de développement est mis à jour quotidiennement, l'environnement de recette est mis à jour mensuellement pour que les intervenants sur le projet puissent valider les développements de maintenance et de nouvelles fonctionnalités; et l'environnement de production c'est le site Web qui est visible par tout le monde.

DÉVELOPPEMENT

Dans cette partie on va voir plus en détail les outils et les bonnes pratiques de développement. Au début on va explorer les différents IDE et la différence entre elles. Ensuite on va analyser les bonnes pratiques à mettre en œuvre pour aboutir à une bonne architecture logicielle.

Les IDE

Pour développer en Java, on peut le faire avec un éditeur de texte simple comme Vim, Gedit ou Blocnote, mais on va se rendre compte très rapidement que cela est très très difficile. L'utilisation d'un IDE est primordiale pour un développement plus efficace et plus rapide. Un IDE est un éditeur de texte avec des outils en plus :

Analyseur de code

L'analyseur de code permet de détecter le ou les langages de programmation pour chaque fichier et pour chaque bout de code; il permet de l'analyser et de détecter les éventuelles erreurs.

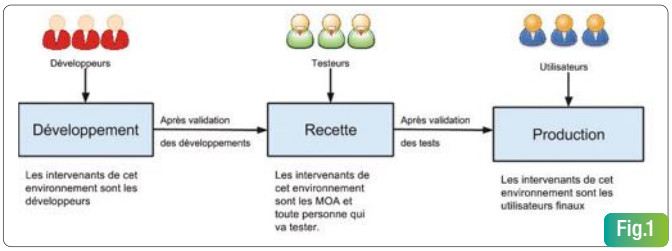


Fig.1

Indexation et moteur de recherche

L'indexeur permet d'indexer les composants du code d'une façon sémantique, par exemple il indexe les classes Java, les méthodes JavaScript... Quant au moteur de recherche, il permet de chercher un élément (classe, méthode, fichier...).

Coloration de code

L'IDE permet de colorer le code suivant les mots clés du langage pour qu'il soit plus visible.

Autres fonctionnalités

Additionnellement, un éditeur peut avoir d'autres fonctionnalités, suivant l'éditeur de l'IDE. On peut trouver des serveurs intégrés, des gestionnaires de versions, des outils pour passer les tests et voir la couverture du code...

Les différents IDE du marché

Un éditeur peut avoir une ou plusieurs des fonctionnalités citées précédemment, mais les fonctionnalités principales qu'un IDE doit avoir est l'analyse de code et l'indexation. Il y a plusieurs IDEs sur le marché qui permettent de coder en Java, voici les plus connus :

- **Eclipse :** Eclipse est un outil très puissant, sous licence EPL (Eclipse Public Licence). La puissance d'Eclipse est la communauté et le nombre de plug-ins existants. C'est un outil modulaire, c'est-à-dire qu'il a un noyau qui fait les fonctionnalités basiques (charger les plug-ins, les gérer.), et toutes les autres fonctionnalités sont sous forme de plug-ins.
- **NetBeans :** C'est l'outil promu par Oracle, sous licence GNU Public License. Il est écrit en Java. Composé d'un module central contenant les fonctionnalités majeures, il est possible d'y ajouter des plug-ins.
- **Intellij :** Il y a deux versions d'Intellij, la Community Edition qui est l'édition gratuite mais qui ne contient que les fonctionnalités primaires et la Ultimate Edition qui est payante. Surtout pour sa version Ultimate, l'outil est très puissant dans l'indexation et dans la reconnaissance des langages.

Les bonnes pratiques

Un projet peut vite devenir un calvaire si on ne suit pas les bonnes pratiques de développement et d'architecture. On va voir ici quelques techniques qui permettent d'avoir une architecture modulaire et maintenable. On va prendre comme exemple une application de gestion hôtelière n-tiers :

La partie front

Pour cet exemple on va prendre comme architecture HTML + AJAX, mais ce qu'on va voir ici est applicable pour toutes les architectures front de type HTML + Ajax :

- **Minimiser les appels serveur :** Pour chaque appel serveur, on passe par plusieurs couches, la communication passe par le protocole HTTP qui va prendre un peu de temps pour envelopper le message, puis on va passer par certaines couches réseau (Exemple les couches du modèle OSI), ensuite, passage dans le réseau et récupération du message de l'autre côté. On ne le remarque pas pour des communications non récurrentes, mais si on fait plusieurs communications avec le serveur et qu'on va à chaque fois yvéhiculer des données qui seront divisées en trames... Cela sera très vite coûteux. Donc, on doit essayer de tout ramener dans une seule requête. Par exemple si on veut afficher un affichage général d'un hôtel, et que l'on sait que l'utilisateur va utiliser la vue détaillée, on peut la ramener une seule fois. Mais il ne faut pas tomber dans l'erreur de l'excès d'informations.
- **Moduler le front :** Créer dans le front plusieurs modules, pour la réutilisation et la maintenabilité. Par exemple créer un module service (REST), qui, lui, sera la seule interface pour communiquer avec le réseau. On peut créer un module d'utilitaire, où on réunit toutes les fonctionnalités utilitaires comme manipulation des dates, les RegEx...

- **Utiliser avec précaution les frameworks JavaScript :** Minimiser le nombre des frameworks JavaScript, et les utiliser avec précaution, surtout les frameworks qui font des manipulations de DOM (comme AngularJs et Jquery). Ils peuvent changer le DOM au même moment et créer un mauvais rendu.
- **Utiliser le CPU de l'utilisateur :** Si c'est possible, faire les calculs dans la partie client, cela permet de ne pas faire d'appel serveur. Supposons qu'il y ait mille personnes connectées sur l'application de gestion hôtelière, l'application va afficher la liste des hôtels et va afficher le prix minimum; si cinquante clients cherchent le prix minimum dans le serveur au même temps, le serveur aura cinquante requêtes qu'il doit gérer. Par contre si on fait le calcul dans le front (puisqu'on a la liste des hôtels), le serveur aura zéro requête à traiter, et chaque client aura un calcul à faire (ce qui n'est pas coûteux pour le client).

La partie back-end

- **Moduler le back-end :** On crée un contrôleur qui va avoir pour fonction de recevoir et de dispatcher les requêtes pour une entité, un service qui va traiter la logique métier, et, enfin, un DAO qui va gérer les données. On voit aussi dans la figure que si on est dans le même module, la communication se fait directement entre les services (exemple UserService et HotelService), mais toutes les communications externes (application Web, mobile ou tout simplement une autre application), la communication se fait avec le contrôleur (en général via HTTP et des services Web).
- **Gérer les performances :** Un des problèmes majeurs dans les applications est la performance de la manipulation des données. On doit donc minimiser le nombre de requêtes entre l'application et la base de données puisque cette communication va passer généralement - comme on a vu pour la communication entre le client et le serveur - par plusieurs couches.

Suivre les patrons de conception

Il y a un grand nombre de patrons de conception, ce sont des solutions reconnues pour être une bonne pratique pour un problème donné. Ils sont divisés en plusieurs groupes, GoF (Gang of four), GRASP (General Responsibility Assignment Software Patterns), l'inversion de contrôle...

Bien modéliser les objets (Cohésion et couplage)

Le couplage est le lien entre deux unités du projet. Deux classes qui dépendent l'une de l'autre sont fortement couplées; ce lien doit être le plus faible possible pour que, si on doit changer le fonctionnement d'une classe, on n'ait pas à changer la deuxième. La cohésion fait référence au nombre et à la diversité des tâches dont une unité est responsable, cette cohésion devrait être la plus forte possible, pour permettre la réutilisabilité de la classe et avoir une homogénéité dans le fonctionnement de la classe.

BIEN TESTER

L'une des étapes principales d'une architecture, c'est de mettre en place un bon mécanisme de test. Ce dernier nous permet d'avoir une cohérence entre les demandes et le programme. Il y a plusieurs types de tests, les plus importants sont :

Les tests unitaires

Ce sont les tests de plus bas niveau, ces tests permettent de tester une méthode, une classe ou une fonctionnalité unitaire. Ce qui nous permettra de valider la performance et la qualité du composant testé.

Les tests d'intégration

Ces tests valident l'intégrité des modules entre eux, et dans l'environnement final. Cela permettra de détecter les erreurs d'interface entre les composants.

Les tests fonctionnels

Ce sont les tests qui permettent de vérifier si la spécification et le code sont conformes.

Les types de Mock

Dummy

C'est le plus simple de tous, c'est un objet passé au test mais jamais utilisé, en général on a besoin de passer un paramètre à la méthode à tester mais on sait bien qu'il ne va pas être utilisé.

Fake

C'est une implémentation d'une classe ou objet, minimale mais suffisante pour le fonctionnement du test.

Stub

Ressemblant au fake, sauf qu'ici on n'implémente que les méthodes qui seront utilisées.

Mock

C'est le fait de donner une implémentation proxy dans laquelle on peut y mettre des tentacules pour nous prévenir des appels de ses méthodes et pour surcharger des méthodes... Cela nous permettra de changer des valeurs de retour, de lever une exception ou de vérifier les états (passage par une méthode, lever une exception...).

Spy

C'est une dérivée du Mock, sauf que dans ce type de Mock, on ne change rien (Ni la valeur de retour, ni les exceptions), on vérifie seulement les états.

Les outils les plus connus

JUnit (Java)

C'est le plus ancien et le plus utilisé, il offre plusieurs fonctionnalités et est compatible avec la plupart des outils de build (Maven, Gradle...) et les IDEs.

TestNG (Java)

Il est inspiré par Junit, c'est un framework de test très puissant.

Jasmin (JavaScript)

Framework de test JavaScript. C'est le plus connu et le plus utilisé dans le monde JavaScript.

PhantomJS (JavaScript)

Cet outil est très puissant, il peut être utilisé pour ouvrir des pages Web, prendre des screenshots, exécuter des actions de l'utilisateur...

Selenium (End to End)

C'est un ensemble d'outils (Extention firefox, API...) destiné à automatiser les tests de l'application en exécutant un script de test sur le navigateur, appuyant sur des boutons, changeant les valeurs des champs de texte...

Couverture des tests

Un des concepts à mettre en place est le système de couverture du code, ce système nous permettra d'avoir des métriques sur la couverture des tests dans le code. Sonar est un des outils qui permettent d'avoir ces métriques bien synthétisées et détaillées (suivant le besoin). Il y a plusieurs outils permettant de gérer cette couverture :

Cobertura

C'est un outil Java permettant de couvrir le code, il s'intègre facilement avec les IDEs du marché (Eclipse, Intellij...). Il couvre le code en utilisant le bit-code.

Emma

Utilisé dans Sonar, cet outil permet d'avoir des métriques bien synthétisées pour la couverture du code.

CONCLUSION

Il reste bien évidemment beaucoup a dire concernant l'architecture, comme la sécurité (Oauth2, session...) et les plateformes (Saas, Paas...). Les architectures sont en continuelle évolution; ce qu'il faut retenir à mon avis c'est d'être toujours à la page, faire de la veille technologique, structurer et formaliser ce qu'on apprend. ⊞

Développer avec le SDK Kinect 2

2e partie

La Kinect 2 est le capteur de mouvements de deuxième génération produit par Microsoft. La Kinect intègre plusieurs capteurs dont une caméra couleur full HD, un émetteur et un récepteur infrarouge ainsi que quatre micros. De par la diversité et la nature de ses composants, exploités par le SDK de développement Kinect 2 pour Windows, la Kinect offre au développeur toutes sortes de possibilités d'interaction avec l'humain très facilement exploitables par le SDK.

Thomas Ouvré
Consultant Infinite Square
Blog : http://blogs.infinitesquare.com/b/touvre

Body Index

Les trames de profondeur donnent accès à des informations se prêtant extrêmement bien à la mise en place d'algorithmes de reconnaissance de formes. Les APIs du SDK permettent de déduire du flux de profondeur une nouvelle source de données : le Body Index. Celui-ci permet de distinguer les pixels d'une trame de profondeur appartenant au corps d'un utilisateur des pixels de l'environnement.

Nous allons effectuer un rendu de la caméra en affichant les pixels du corps de l'utilisateur avec une couleur donnée. Cela débute par l'allocation des buffers intermédiaires :

```
var source = _sensor.BodyIndexFrameSource;
var description = source.FrameDescription;
var width = description.Width;
var height = description.Height;
_bitmap = new WriteableBitmap(width, height);
_bitmapIntermediateBuffer = new byte[width * height * BGRA_BYTES_PER_PIXEL];
_bodyIndexBuffer = new byte[description.LengthInPixels];
```

On associe notre bitmap au contrôle image et on s'abonne à l'évènement FrameArrived :

```
myImage.Source = _bitmap;
_bodyIndexReader = source.OpenReader();
_bodyIndexReader.FrameArrived += Reader_FrameArrived;
```

Une fois notre tableau de données récupéré nous allons pour chaque pixel vérifier s'il appartient au décor ou à l'utilisateur puis le cas échant lui associer la couleur bleu.

```
private void Reader_FrameArrived(BodyIndexFrameReader sender, BodyIndexFrameArrived
EventArgs args)
{
  using (var frame = args.FrameReference.AcquireFrame())
  {
    if (frame == null) return;
    frame.CopyFrameDataToArray(_bodyIndexBuffer);
    for (var i = 0; i < _bodyIndexBuffer.Length; ++i)
    {
      var j = i * BGRA_BYTES_PER_PIXEL;
      var index = _bodyIndexBuffer[i];
      if (index == 0)
      {
        var color = _bodyIndexColors[index];
        _bitmapIntermediateBuffer[j + 0] = (byte)(255);    // blue
        _bitmapIntermediateBuffer[j + 1] = (byte)(0);   // green
        _bitmapIntermediateBuffer[j + 2] = (byte)(0);   // red
        _bitmapIntermediateBuffer[j + 3] = (byte)(0);   // alpha
      }
      else
      {
        _bitmapIntermediateBuffer[j + 0] = 0; // blue
        _bitmapIntermediateBuffer[j + 1] = 0; // green
        _bitmapIntermediateBuffer[j + 2] = 0; // red
        _bitmapIntermediateBuffer[j + 3] = 0; // alpha
      }
    }
    _bitmapIntermediateBuffer.CopyTo(_bitmap.PixelBuffer);
    _bitmap.Invalidate();
  }
}
```

Sans oublier de mettre à jour la méthode StopCurrentDemo pour libérer les ressources, voici un rendu pouvant être généré par l'exemple actuel : Fig.1.

Le Body

Cette source permet au développeur d'obtenir précisément les informations du corps ou squelette des utilisateurs traqués par la Kinect (Jusqu'à un maximum de 6).

Les données de chaque utilisateur traqué sont agrégées dans une instance de la classe Body. De ces instances sont accessibles les coordonnées de 25 jointures, soit 25 points précis sur le corps. Les 25 points du squelette sont représentés dans le schéma suivant : Fig.2.

Positions des jointures

La première étape consiste à créer une classe JointModel, réceptacle des données nécessaire à l'affichage de chaque jointure et permettant de notifier l'UI d'une mise à jour de celles-ci :

```
public class JointModel : INotifyPropertyChanged
{
  private double _x;
  private double _y;
  private JointType _type;
  private bool _isTracked;
  private Color _color = Colors.Red;

  public double X
  {
    get { return _x; }
    set { Set(ref _x, value); }
  }

  public double Y
  {
```

```
    get { return _y; }
    set { Set(ref _y, value); }
}

public double Width
{
    get { return 40; }
}

public double Height
{
    get { return Width; }
}

public JointType Type
{
    get { return _type; }
    set
    {
        if (Set(ref _type, value))
        {
            if (PropertyChanged != null)
                PropertyChanged(this, new PropertyChangedEventArgs("Label"));
        }
    }
}

public string Label
{
    get { return Type.ToString(); }
}

public bool IsTracked
{
    get { return _isTracked; }
    set
    {
        if (Set(ref _isTracked, value))
        {
            if (PropertyChanged != null)
                PropertyChanged(this, new PropertyChangedEventArgs("Visibility"));
        }
    }
}
public Visibility Visibility
{
    get { return IsTracked ? Visibility.Visible : Visibility.Collapsed; }
}

public Color Color
{
    get { return _color; }
    set { Set(ref _color, value); }
}

public event PropertyChangedEventHandler PropertyChanged;

private bool Set<T>(ref T prop, T value, [CallerMemberName] string propertyName = null)
{
    if (Equals(prop, value)) return false;
    prop = value;
    if (PropertyChanged != null)
        PropertyChanged(this, new PropertyChangedEventArgs(propertyName));
    return true;
  }
}
```

L'étape suivante consiste en l'ouverture de la connexion avec la source de données et en l'abonnement à l'évènement FrameArrived. C'est aussi l'étape idéale pour initialiser quelques variables comme la liste des instances de Body, les Joints des 6 Body et la taille d'une trame couleur.

```
var source = _sensor.BodyFrameSource;
_bodies = new Body[source.BodyCount]; // BodyCount == 6
var totalJointCount = source.BodyCount * JOINTS_PER_BODY;
_joints = new List<JointModel>(totalJointCount);
for (var i = 0; i < totalJointCount; i++)
    _joints.Insert(i, new JointModel { IsTracked = false , Type = (JointType)(i % JOINTS_PER_BODY) });
SceneWidth = _sensor.ColorFrameSource.FrameDescription.Width;
SceneHeight = _sensor.ColorFrameSource.FrameDescription.Height;
if (PropertyChanged != null)
{
    PropertyChanged(this, new PropertyChangedEventArgs(nameof(SceneWidth)));
    PropertyChanged(this, new PropertyChangedEventArgs(nameof(SceneHeight)));
}

_bodyReader = source.OpenReader();
_bodyReader.FrameArrived += Reader_FrameArrived;
```

Pour fonctionner, ce code nécessite d'implémenter l'interface INotifyPropertyChanged et de déclarer des champs et propriétés :

```
public event PropertyChangedEventHandler PropertyChanged;
private const int JOINTS_PER_BODY = 25;
public double SceneWidth { get; set; }
public double SceneHeight { get; set; }
private List<JointModel> _joints;
public List<JointModels> Joints
{
    get { return _joints; }
}
```

Le handler Reader_FrameArrived peut maintenant être implémenté de sorte à itérer sur tous les body mis à jour à l'aide des méthodes de l'API. En effet, quel que soit le nombre d'utilisateurs présents devant le capteur, l'API travaille sur une collection de 6 Body et les met tous à jour. Une propriété IsTracked permet ensuite de déterminer s'il s'agit d'un Body associé à un utilisateur réellement détecté. Voici une façon de les traiter :

```
using (var frame = args.FrameReference.AcquireFrame())
{
    if (frame == null) return;
    frame.GetAndRefreshBodyData(_bodies);
    var bodyCount = frame.BodyCount;
    for (var i = 0; i < bodyCount; i++)
    {
        var body = _bodies[i];
        //...
```

```
    }
}
```

Dans la boucle for, il faut mettre à jour les items de la collection _joints selon l'état du Body correspondant. Si le Body n'est pas dans l'état « traqued » (c'est-à-dire si elles n'appartiennent pas à un utilisateur), il est possible de notifier l'UI que les jointures correspondantes ne doivent pas être affichées.

```
if (!body.IsTracked)
{
    for (var j = 0; j < JOINTS_PER_BODY; j++)
        _joints[i * JOINTS_PER_BODY + j].IsTracked = false;
    continue;
}
```

Dans le cas contraire, chaque Body permet d'obtenir une collection de jointures via la propriété Joints. Celle-ci étant un Dictionary indexé sur l'énumération JointType, on utilise cette dernière pour obtenir les coordonnées et informations de chaque jointure. Ces dernières étant exprimées dans un espace en 3 dimensions et selon le point d'origine correspondant à la Kinect, il est nécessaire de faire appel au CoordinateMapper : un objet réalisant les calculs nécessaires pour transposer des coordonnées dans le référentiel des trames couleurs, entre autres.

```
var mapper = frame.BodyFrameSource.KinectSensor.CoordinateMapper;
for (var j = 0; j < JOINTS_PER_BODY; j++)
{
    var bodyJoint = body.Joints[(JointType)j];
    var uiJoint = _joints[i * JOINTS_PER_BODY + j];
    uiJoint.IsTracked = bodyJoint.TrackingState != TrackingState.NotTracked;
    if (uiJoint.IsTracked)
    {
        var pos = mapper.MapCameraPointToColorSpace(bodyJoint.Position);
        uiJoint.X = pos.X;
        uiJoint.Y = pos.Y;
    }
}
```

En l'occurrence, il s'agit de la méthode MapCameraPointToColorSpace du CoordinateMapper qui est utilisée, ce qui permet d'obtenir des coordonnées en adéquation avec la trame couleur.
En effet, les composantes X et Y ainsi obtenues correspondent directement aux coordonnées d'un pixel de cette trame. Cela permet entre autres de superposer le rendu de ces deux sources et d'obtenir une image parfaitement alignée. Une opération similaire peut être réalisée avec les sources infrarouges, de profondeur, et BodyIndex avec la méthode MapCameraPointToDepthSpace.
Pour finir, le XAML de la vue doit être modifié pour tenir compte de ces changements. La liste des jointures avec leurs coordonnées étant une propriété de la page dans l'exemple actuel, il est possible de les afficher très simplement en tirant parti du binding en commençant par nommer la page :

```
<Page x:Class="DemoKinect.WinRT.MainPage"
    x:Name="Self"
```

Un ItemsControl avec dans une ViewBox permettra d'afficher chaque jointure dans un Canvas, sans se soucier des coordonnées des jointures pouvant être hors de l'écran : la ViewBox se chargera de tout redimensionner.

```
<Viewbox Stretch="Uniform">
    <ItemsControl ItemsSource="{Binding ElementName=Self, Path=Joints}">
        <ItemsControl.ItemsPanel>
            <ItemsPanelTemplate>
                <Canvas Width="{Binding ElementName=Self, Path=SceneWidth}"
                        Height="{Binding ElementName=Self, Path=SceneHeight}" />
            </ItemsPanelTemplate>
        </ItemsControl.ItemsPanel>
        <ItemsControl.ItemTemplate>
            <DataTemplate>
                <Grid Visibility="{Binding Visibility}">
                    <Ellipse Width="{Binding Width}"
                             Height="{Binding Height}"
                             VerticalAlignment="Top"
                             HorizontalAlignment="Left" />
                        <Ellipse.Fill>
                            <SolidColorBrush Color="{Binding Color}" />
                        </Ellipse.Fill>
                    </Ellipse>
                    <TextBlock Text="{Binding Label}"
                               Margin="30"/>
                    <Grid.RenderTransform>
                        <TranslateTransform X="{Binding X}"
                                            Y="{Binding Y}" />
                    </Grid.RenderTransform>
                </Grid>
            </DataTemplate>
        </ItemsControl.ItemTemplate>
    </ItemsControl>
</Viewbox>
```

Etats des mains et HandState

Une autre information très intéressante présente sur le Body concerne l'état des mains des utilisateurs détectés, au détail prêt qu'elle est limitée à deux utilisateurs simultanés maximum. Ces informations permettent de détecter trois postures particulières adoptées par les mains de l'utilisateur, exprimées chacune par une valeur de l'énumération HandState :

- Open : la paume de la main est ouverte (tout en faisant face au capteur) ;
- Closed : le poing est fermé, pousse et autres doigts rentrés ;
- Lasso : le poing est fermé, mais un ou deux doigts sont tendus (généralement l'index et le majeur, un seul doigt peut ne pas suffire selon les proportions de la main) ;

Ces postures vont permettre la mise en place d'interactions entre l'utilisateur et l'applicatif : le mode Lasso servant généralement à effectuer une action précise comme le détourage d'une forme ou le mode Closed représentant la volonté « d'accrocher » un élément pour le déplacer. L'état de chaque main est exposé à travers les propriétés HandLeftState et HandRightState.
Afin de faire ressortir ces informations à l'écran il est possible d'adapter le code de sorte à augmenter la taille des jointures des mains et à modifier leur couleur selon l'état détecté. Pour cela, la propriété Width de la classe JointModel peut retourner une valeur dépendante de son type :

```
public double Width
{
    get
    {
        return Type == JointType.HandLeft || Type == JointType.HandRight
            ? 40
```

```
        : 20;
    }
}
```

Ensuite, au moment de déterminer les coordonnées de chaque jointure (seconde boucle for du handler Reader_FrameArrived), un test sur la propriété Type aidant, il est possible de modifier la couleur de la jointure selon le HandState :

```
if (uiJoint.IsTracked)
{
    if (uiJoint.Type == JointType.HandLeft)
        uiJoint.Color = HandStateToColor(body.HandLeftState);

    if (uiJoint.Type == JointType.HandRight)
        uiJoint.Color = HandStateToColor(body.HandRightState);

    var pos = mapper.MapCameraPointToColorSpace(bodyJoint.Position);
    uiJoint.X = pos.X - uiJoint.Width / 2;
    uiJoint.Y = pos.Y - uiJoint.Height / 2;
}
```

La méthode HandStateToColor, pour finir, détermine la couleur associée à chaque valeur de l'énumération HandState :

```
return state == HandState.Closed
    ? Colors.Yellow
    : state == HandState.Open
        ? Colors.Green
        : state == HandState.Lasso
            ? Colors.Blue
            : state == HandState.Unknown
                ? Colors.Gray
                : Colors.Transparent;
```

Exemple de rendu effectué par-dessus le rendu de la caméra couleur : Fig.3. C'est bien évidemment de la source Body dont le développeur se servira la plupart du temps pour effectuer le code métier propre à son application et autoriser les interactions entre utilisateurs et applicatif. Néanmoins, ce dernier doit garder en tête que ce ne sont pas les seules possibilités du SDK. En effet, la source Body permet à priori de « tout » faire : reconnaître des gestures, des postures, obtenir la position de l'utilisateur ou une représentation de celui-ci dans l'espace. Bref, un peu de code et quelques algorithmes bien implémentés peuvent sembler être la solution à tous les cas de figure, mais, certains ne s'y tromperont pas : cela demande parfois de solides compétences en mathématiques.

C'est pourquoi d'autres articles à venir traiteront de cas plus précis comme la mise en place du framework Touch-Less intégré au SDK, permettant d'utiliser la Kinect comme un dispositif de pointage de manière quasi transparente pour le développeur et intuitive pour l'utilisateur. Un autre outil, Visual Gesture Builder, permet quant à lui de mettre en place des scénarios de reconnaissance de gestures même complexes, pour bien moins d'efforts de la part du développeur que si la détection ne devait se baser que sur du code maison.


Fig.1

Fig.2

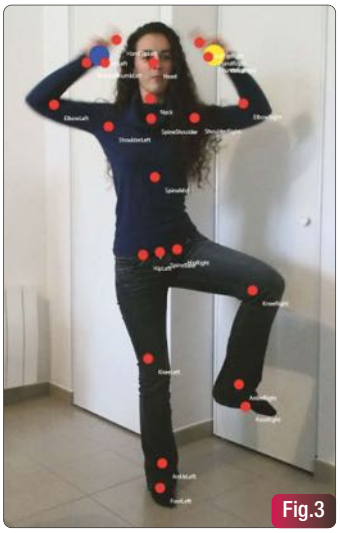


Fig.3

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 : S. Saurel

Experts : T. Ouvré, F. Chevalier, G. Deruette, N. Lamti, L. Claustres, J-F Garreau, B. Hamdi, D. Manel, M. Bouilloux, G. Delattre, J. Salomon, V. Michalak, C. Rodriguez, A. Zanchetta, B. Bideaux, S. Sibué, Jeffr3y, F. Sagez, S. Blanchard, Chengbo, Morgane, E. Cunibil, M. Turcotte, A. Byron, C. Villeuneve, E. Quadrille.

Photos/illustrations : © Drupal - D.R., F. Chevalier - © ISTOCK - DOLGACHOV

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
www.linformaticien.com


DÉVELOPPEZ 10 FOIS PLUS VITE
WINDEV® 21
un grand
MERCI !
Cette fin d'année encore, vous avez été des milliers de professionnels à venir découvrir les nouveautés des AGL WINDEV, WEBDEV et WINDEV Mobile.
11 villes, 11 salles combles grâce à votre fidélité : MERCI X11 !
www.pcsoft.fr
Fournisseur Officiel de la Préparation Olympique
VERSION 21
DISPONIBLE
WINDEV Mobile 21
WINDEV 21
WEBDEV 21
921 NOUVEAUTÉS
MONTPELLIER • MARSEILLE • LYON • TOULOUSE • BORDEAUX • NANTES • PARIS • LILLE • BRUXELLES • STRASBOURG • GENÈVE • (MONTRÉAL)
AFFLUENCE RECORD DANS LES 11 VILLES DU TDF DES VERSIONS 21
Merci de votre fidélité