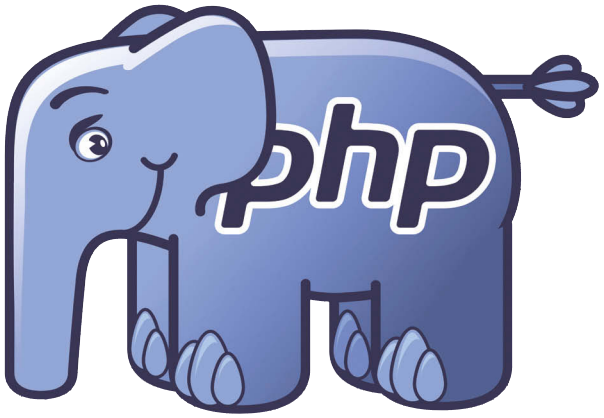
programmez!
#201 - Novembre 2016
le magazine des développeurs



PHP 7.1

et le futur de PHP

Recrutement

Quel IDE JAVA choisir ?

eclipse
NetBeans
IJ

SWIFT 3.0

Découvrir RUST

Construire son clavier

Quel écran pour son Arduino ?

M 04319 - 201 - F: 6,50 € - RD

Printed in EU - Imprimé en UE - BELGIQUE 7 € - Canada 9,80 $ CAN - SUISSE 13,10 FS - DOM Surf 7,50 € - TOM 1020 XPF - MAROC 55 DH

ikoula
HÉBERGEUR CLOUD

PRÉSENTE

CLOUD**IKOULA**ONE

Le succès est votre prochaine destination

MIAMI SINGAPOUR PARIS
AMSTERDAM FRANCFORT ___

*CLOUD**IKOULA**ONE* est une solution de Cloud public et privé qui vous permet de déployer en 1 clic et en moins de 30 secondes des machines virtuelles à travers le monde sur des infrastructures SSD haute performance.

www.ikoula.com sales@ikoula.com 01 84 01 02 50

ikoula HÉBERGEUR CLOUD

Le jour d'après

Le mois d'octobre est déprimant : Mr Robot saison 2 terminée, Halt & Catch Fire saison 3 terminée, House of Cards, Veep, Games of Throne, Twin Peaks Saison 3… pas avant 2017. Il reste The man in the High Castle saison 2 pour nous faire tenir. Mais tout n'est pas déprimant.

La preuve, depuis septembre, les nouveaux casques de réalité virtuelle / augmentée, se multiplient : Hololens arrive en France (+3 000 le kit de dév) et le PlayStation VR de Sony. Ce dernier reçoit de bonnes critiques même sans atteindre le niveau de ses concurrents mais il est vendu à 399 ! Incontestablement, le marché est en pleine ébullition et ce n'est qu'un début.

Pour ce nouveau numéro de Programmez!, plusieurs dossiers :

- PHP : l'actualité de la communauté est très riche en monuments. La v7 est disponible depuis un an et le travail avance sur les futures versions (7.1, 8.0), preuve que PHP continue à évoluer.
- L'IDE Java a toujours été une question sensible pour les développeurs. Eclipse, NetBeans ou IntelliJ ?
- Swift : vous avez déjà entendu parler du langage Swift, conçu par Apple pour remplacer Objective-C ? Désormais en version 3, le langage devient mature et les nombreux changements de cette version permettent à Swift de combler une grande partie des lacunes. A quand une distribution Swift sur Windows ?

Nous parlerons aussi de Web Scraping, de responsive sous Drupal, de Xamarin, d'Azure, de watchOS 3.0, de Python en mode Forensic, et, bien entendu, de Maker avec Constellation, créer son propre clavier, ou encore de comment choisir le bon écran pour son Arduino.

Bonne lecture

ftonic@programmez.com
Dresseur de codes.

Programmez! #202 - le 3 décembre 2016

Apprendre à coder dès 7 ans !
Pourquoi apprendre le code ? Coder à l'école, les outils, les bonnes adresses

NOUVEAU !
Rubrique dédiée à la Réalité Augmentée et à la Réalitée Virtuelle

Choisir son fournisseur de Cloud Computing
Comment choisir son cloud ? De quel cloud parle-ton ? Quelles fonctionnalités cherchez-vous ? Les critères pour choisir au mieux ? Un dossier complet !

novembre

DEVFEST TOULOUSE 2016 :

3 novembre

Le DevFest rassemble les plus grands événements organisés par les communautés autour des technologies Google : les Google Developer Groups. Pendant une journée vous pourrez assister à des talks autour des technologies affiliées à Google (Web Apps, Mobile, Tools & Methods). Les communautés toulousaines vous attendent pour cette première édition dans la ville rose ! Plus d'informations et inscription : https://devfesttoulouse.fr/ .

MAKER FAIRE LILLE :

5 & 6 novembre

La saison Maker a repris ! Lille accueillera le Maker Faire durant deux jours, les 5 et 6 novembre. Comme toujours, des dizaines de makers seront présents et des conférences et ateliers auront lieu durant l'événement. Le 4 novembre est réservé aux scolaires. Lancez-vous !
Site : www.makerfairelille.com

XEBICON'16 :

9 novembre / Paris

Xebia organise sa conférence annuelle qui parlera technologies, innovations, techniques. Cette année, +50 sessions seront proposées (sessions, ateliers, etc.). Il y en aura pour tous les goûts : Java, mobilité, Cloud, architectures, etc. Les retours d'expérience seront à l'honneur avec pas moins de 13 témoignages dont Softbank Robotics, Air France, Fnac.
Informations & inscriptions : xebicon.fr

PARIS OPEN WORLD SUMMIT 2016 :

16 & 17 novembre/Plaine Saint-Denis

La nouvelle édition de la grande conférence open source française se tiendra en novembre prochain. Le thème central sera l'innovation qui s'appuie de plus en plus sur l'open source, l'open data, l'open hardware, etc. Le Paris Open Source Summit 2016 aura pour objectif de démontrer ces postulats par l'action, en mettant l'accent sur le « faire ». Un ensemble de projets sont ainsi développés tout au long de l'année pour illustrer concrètement l'orientation choisie – Empowering Open Innovation – dans des secteurs variés, à la fois transverses (blockchain, gouvernement ouvert, smart cities, financement, IA) et métiers (mobilité, énergie, santé, droit, robotique, agriculture). Fédérant ainsi les efforts de plus d'une centaine de personnes, tous seront présentés le jour J. A cette thématique principale, trois volets seront abordés : la technologie, l'entreprise et la société. Chacun de ces secteurs a des défis propres à relever.
Pour en savoir plus : http://www.opensourcesummit.paris

DEVOPS REX:

28 novembre/Paris

Devops REX est la conférence devops 100% retours d'expérience (REX). Des speakers reconnus traiteront des applications concrètes de la méthodologie devops en entreprise, avec ses bénéfices mais aussi ses contraintes et ses limites. + 400 personnes sont attendues pour cette première édition.
Site : https://www.devopsrex.fr/

POUR 1 EURO DE PLUS

JUSQU'AU 12 DÉCEMBRE

COMMANDEZ WINDEV 22

OU WEBDEV 22 **OU** WINDEV MOBILE 22

ET RECEVEZ LE NOUVEL iPhone 7

POUR 1 EURO DE PLUS

Actuellement sur les routes:
WinDevTour 22. 11 villes. Gratuit.
Inscrivez-vous sur pcsoft.fr

iPhone 7 Plus

iPhone 7 Plus **128GB**.

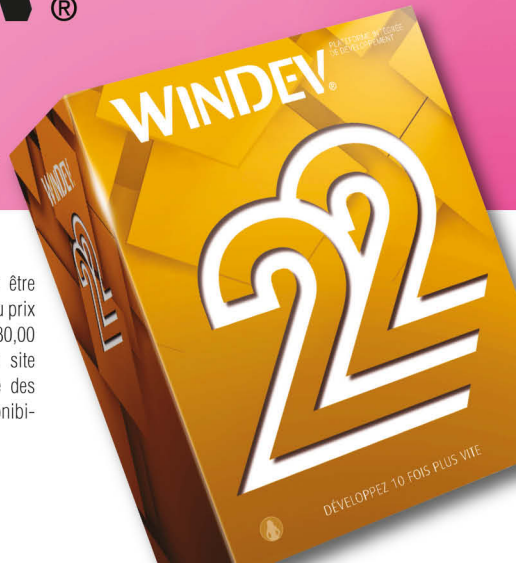
Ou choisissez :

- iPhone 7 **256GB**
- iPad Pro **32GB** 12,9"
- MacBook Air **128GB** 11,6"
- ...

(voir le détail sur www.pcsoft.fr)

WINDEV®

Atelier de Génie Logiciel Professionnel



Pour bénéficier de cette offre exceptionnelle, il suffit de commander WINDEV 22 (ou WINDEV Mobile 22, ou WEBDEV 22) chez PC SOFT au tarif catalogue avant le 12 Décembre 2016. Pour 1 Euro de plus, vous recevrez alors le ou les magnifiques matériels que vous aurez choisis. Offre réservée aux sociétés, administrations, mairies, GIE et professions libérales, en France métropolitaine. **L'offre s'applique sur le tarif catalogue uniquement**. Voir tous les détails et des vidéos sur : **www.pcsoft.fr** ou appelez-nous (04.67.032.032).

Le Logiciel et le matériel peuvent être acquis séparément. Tarif du Logiciel au prix catalogue de 1.650 Euros HT (1.980,00 TTC). Merci de vous connecter au site www.pcsoft.fr pour consulter la liste des prix des matériels et les dates de disponibilité. Tarifs modifiables sans préavis.

WWW.PCSOFT.FR

• **Linus** pas content d'un vilain bug dans le noyau 4.8. Un patch remplace BUG_ON par WARN.On…

• **Microsoft** arrête son bracelet Band. Il était sympa pourtant. Rumeur ou pas ?

• **Sonder Keyboard** travaille sur un clavier utilisant une encre numérique : avantage, pouvoir personnaliser les touches du clavier selon la langue, l'écriture et une consommation très basse. Apple est fortement intéressé…

• **Le langage P** fait son apparition. Créé dans les laboratoires de Microsoft, P est un langage asynchrone par évènements. Lien : https://github.com/p-org/P .

• **Galaxy Note 7** tourne au cauchemar pour Samsung qui arrête la production et la vente de ce modèle. Ce modèle devait concurrencer frontalement l'iPhone. Reste à connaître les causes réelles des surchauffes et des explosions. A suivre.

• **Google Noto** est une police de caractères True Type, et open source, prenant en charge 800 langues et 110 000 caractères Unicode ! Lien : https://www.google.com/get/noto/

C.H.I.P. EST EN PHASE DE LANCEMENT

C.H.I.P. est passé en production massive mi-octobre ! Cette board est attendue avec impatience depuis plusieurs mois. Les livraisons sont attendues pour le mois de novembre. Une version Pro est déjà annoncée pour décembre ; pas sympa pour les acheteurs de la v1 ! Mais cette version améliorée coûtera 16 $ et aura de belles ressources : ARMv7 à 1 Ghz, 256 à 512 Mo de mémoire, WiFi + BlueTooth et Open Source (matériel et logiciel). Le kit de développement, incluant la board I/O sera elle aussi disponible en décembre pour un prix de 49 $.
Si la C.H.I.P. tient ces promesses, la carte connaîtra le succès et la communauté apparaîtra très rapidement. Mais attention, la concurrence est déjà très vive sur les mini-boards et le prix ne fait pas tout. Il faut que l'écosystème se développe rapidement notamment sur les shields, les accessoires, ainsi que sur la partie développement (outils, SDK, librairies). La stabilité de l'OS sera un point crucial à surveiller. Enfin, espérons que la board sera facilement trouvable en France. Sur le papier, cette carte est concurrente des Arduino et Raspberry Pi.

• **Cmake** sera bientôt mieux intégré à Visual Studio, une annonce attendue car Cmake est très populaire pour générer les makefile.

• **Twitter** veut se faire racheter ? Oui mais par qui et à quel prix ?

• **Hololens** est arrivé en France. 3 299 pour le kit développeur.

INDEX TIOBE DU MOIS

10/2016	10/2015	Tendance	Langage	%	Evolution en %
1	1		Java	18.799%	-0.74%
2	2		C	9.835%	-6.35%
3	3		C++	5.797%	+0.05%
4	4		C#	4.367%	-0.46%
5	5		Python	3.775%	-0.74%
6	8	▲	JavaScript	2.751%	+0.46%
7	6	▼	PHP	2.741%	+0.18%
8	7	▼	VB .NET	2.660%	+0.20%
9	9		Perl	2.495%	+0.25%
10	14	▲▲	Objective-C	2.263%	+0.84%

Incroyable mais vrai, Objective-C reprend quelques couleurs et passe 10e mais cela ne se fait pas au détriment de Swift qui continue à grimper dans l'index, il est 11e. Pour le reste, pas de changement majeur à signaler.

• **OVH** veut se faire une place dans les services back-end pour les apps mobiles. Le service mobile Hosting permet cela avec des VM, des containeurs, du stockage, de la donnée, etc. Le but : faciliter le déploiement et accéder à de multiples services. Lien : https://www.ovh.com/fr/mobile-hosting/

VERS UNE GUERRE DES SYSTÈMES (LINUX) DÉDIÉS AUX CONTENEURS ?

Aujourd'hui, la plupart d'entre nous qui utilisons des conteneurs, notamment Docker, passons par des systèmes Windows, Linux classiques, voire, sur des solutions Cloud, que se soit en x86 ou en ARM pour les plus téméraires.
Mais une autre tendance de fond existe et commence à pointer son code : ce que l'on appelle les Linux Container Host ou OS Container Host pour être plus générique.
Côté Linux, deux projets existent : Photon OS de Vmware et RancherOS de Rancher. On peut éventuellement rajouter le projet Atomic Host de Red Hat, voire, CoreOS.

L'idée générale de ces OS Container Host est de pouvoir fournir un système minimal et optimisé pour y faire tourner les conteneurs, Docker ou non Docker, car il n'y a pas que Docker dans la vie, il y a aussi Kubernetes. Sur la partie Windows, vous pouvez utiliser Nano Server qui peut servir de Container Host (mais uniquement avec conteneur Windows).
L'idée de ces OS Host est de fournir un système minimaliste vidé de tous les packages, de toutes les librairies qui ne servent à rien dans un contexte de conteneurs. Le moteur Docker tourne directement sur le noyau.

Nous reviendrons sur cette question dans un dossier dédié dans le n°202.

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

NOUVEAU ! 1&1 MANAGED CLOUD HOSTING

Le meilleur de deux mondes

Un pack d'hébergement performant associé à des ressources serveur flexibles et modulables à tout moment : **le nouveau Managed Cloud Hosting 1&1 est arrivé !** Idéal pour les projets Web les plus exigeants en termes de disponibilité, de sécurité et de flexibilité.

- ✓ **Ressources dédiées**
- ✓ **+ de 20 combinaisons de stack**
- ✓ **Géré par les experts 1&1**
- ✓ **Flexible & évolutif**
- ✓ **Prêt en moins d'1 minute**

Cloud Vendor Benchmark 2016
Rising Star Germany
expertON GROUP an ISG business

intel XEON inside

Trusted Performance.
Intel® Xeon® processors.

À partir de **9,99** € HT/mois (11,99 € TTC)*

1 CHOIX VOTRE DURÉE D'ENGAGEMENT

1 APPEL UN EXPERT VOUS RÉPOND

1 CERTITUDE LA PROTECTION CONTRE LES DÉFAILLANCES

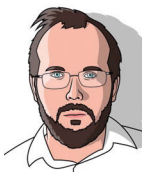
0970 808 911
(appel non surtaxé)

1&1

1and1.fr

*1&1 Managed Cloud Hosting : à partir de 9,99 € HT/mois (11,99 € TTC). Pas de durée minimale d'engagement. Pas de frais de mise en service. Conditions détaillées sur 1and1.fr. 1&1 Internet SARL, RCS Sarreguemines B 431 303 775.

Les joies des écrans avec **Arduino**



François Tonic

Quand vous faites un montage Arduino, souvent, vous allez vous confronter à l'affichage. Dans une majorité des projets, un simple écran LCD avec 2 ou 4 lignes sera suffisant. Mais un jour vous vous direz : je veux aller au-delà… Et là, les problèmes commencent.

Depuis que nous faisons régulièrement des montages pour tester de nouveaux capteurs ou cartes, nous sommes confrontés à cette question d'écran ; de plus en plus vous utilisez des OLED, et, depuis peu, des écrans 2,8 pouces minimum, avec plus ou moins de problèmes.

Un écran, des écrans et des usages

Le LCD RGB (3 couleurs donc) est l'écran standard des kits makers. Il est bon marché et il en existe de nombreuses versions : 2 ou 4 lignes, monté ou nu, etc. Pour un premier objet / montage, c'est idéal. Et surtout, il est assez robuste. Il se code très facilement, et la librairie (en général lcd.h) fonctionne très bien sur toutes les cartes.

Le LCD a plusieurs défauts : il est encombrant, lourd et les possibilités graphiques sont très limitées. Dès que vous souhaitez afficher des images, du graphisme, voire, des animations ou des textes utilisant plusieurs styles, le TFT ou l'OLED sont indispensables.

La technologie OLED est celle qui domine actuellement, par sa consommation d'énergie et la qualité de son affichage même sur des tailles d'écran réduites. Nous utilisons essentiellement des OLED 0,96'' et la qualité d'affichage n'est pas à démontrer. Si nous utilisions les modèles Grove, nous sommes rapidement passés aux OLED "génériques" que l'on trouve facilement en Chine à -3 $ / pièce.

Pour la connexion à la carte, l'OLED utilise en général SPI ou I2C ou les deux. Le I2C est un véritable protocole permettant une grande modularité et connectivité physique réduite (4 broches), mais les performances sont mauvaises. Le SPI nécessite plus de fils, mais son gros avantage est la performance et la consommation. Le SPI est point à point et n'est pas un véritable protocole.

Après vous pouvez utiliser des tailles d'écran plus grandes (minimum 1,8 ou 2,8''), que les dalles soient tactiles ou non. Là aussi, il existe de nombreuses offres. Les shields TFT possèdent souvent un lecteur de cartes de type SD ou micro-SD. Pour des questions de performances, préférez le SPI.

Les scénarios pour les TFT peuvent être nombreux : cadre photo interactif, borne, système de surveillance, etc. Les prix ont beaucoup baissé. Un TFT 1,8'' peut se trouver à 5-6 $. Les modèles tactiles sont plus chers. Si dans les OLED 0,96'', la qualité est relativement homogène, ce n'est pas le cas dans les TFT.

Pour faciliter les choses, il arrive que les constructeurs n'adoptent pas les mêmes brochages ainsi Adafruit et SainSmart diffèrent parfois :
SCK -> SCL
MOSI -> SDA
TFT_CS -> CS
D/C -> RS/DC

Comment cela se passe concrètement ?

Pour les TFT de grandes tailles, en général, il vous suffira d'enfoncer le shield sur votre UNO ou Mega. Si vous utilisez une autre carte, il faudrait établir les correspondances manuellement. Pour un modèle courant, de type 1,8TFT SPI, il faudra connecter 7 broches :

- 3,3V (VNC)
- GND
- TFT_CS (=CS) en 10
- TFT_RST (=RESET) en 9
- TFT_DC en 8
- TFT_SCLK (=SCL) en 13
- TFT_MOSI (=SDA) en 11

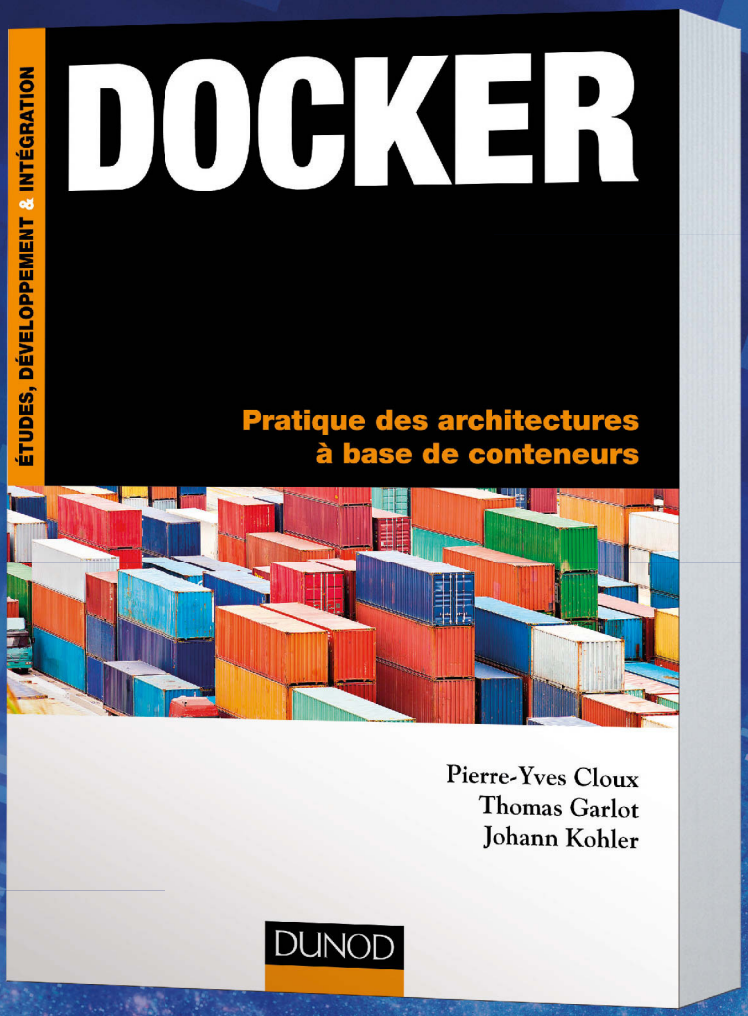
Ou pour certains modèles :

Arduino	TFT
D9	Pin 07 (A0)
D10 (SS)	Pin 10 (CS)
D11 (MOSI)	Pin 08 (SDA)
D13 (SCK)	Pin 09 (SCK)
D8	Pin 06 (RESET)
5V (VCC)	Pin 02 (VCC)
GND	Pin 01 (GND)
3.3V or 5V (VCC)	Pin 15 (LED+)
GND	Pin 16 (LED-)

Pour un OLED 0,96, vous aurez à connecter les broches SCL, SDA, VCC et GND.

Il faut ensuite passer à la partie librairie à utiliser dans votre code. Il vous en faudra plusieurs :

- La librairie SPI (si l'écran est en SPI) : SPI.h ;
- La librairie matérielle : par exemple, Adafruit_ST7735.h ;

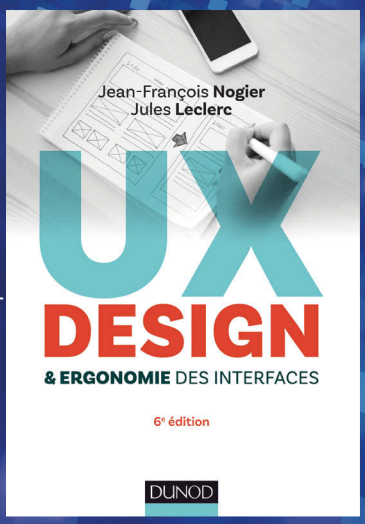


9782100747443 • 320 pages • 29,90 €

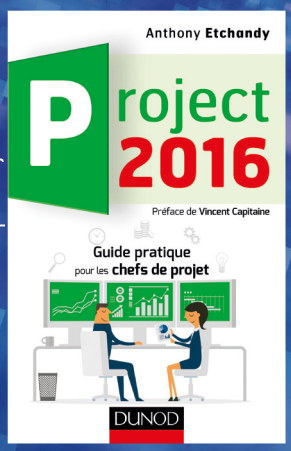
DOCKER
LA SOLUTION POUR DÉPLOYER DU CODE EN PRODUCTION

Cet ouvrage vous explique le **concept de conteneur** et vous apprend à **installer et à utiliser Docker**

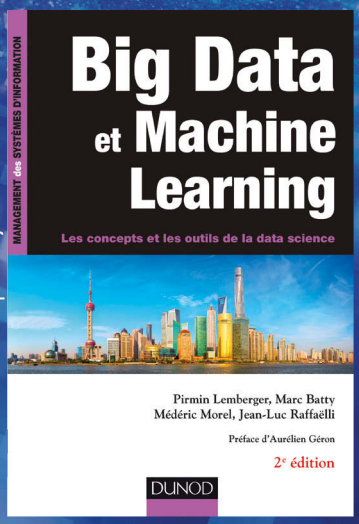
À DÉCOUVRIR ÉGALEMENT



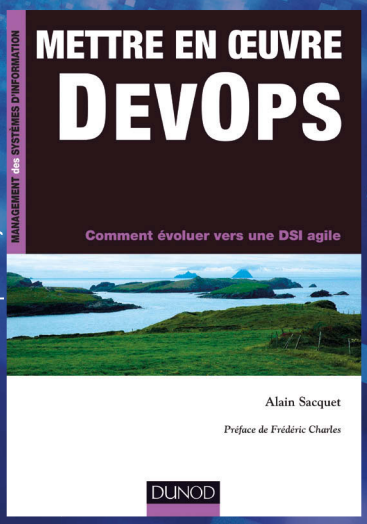
9782100754618 • 320 p. • 35 €



9782100749348 • 160 p. • 19,50 €



9782100754632 • 272 p. • 29,90 €



9782100740444 • 240 p. • 24,90 €

Tout le catalogue sur **dunod.com**

DUNOD
ÉDITEUR DE SAVOIRS

- La librairie graphique : par exemple, Adafruit_GFX.h .

Vous l'aurez compris, pour utiliser le TFT, vous devez utiliser plusieurs librairies, particulièrement pour la partie matérielle et l'affichage (librairie graphique).

Outre le montage physique de l'écran, il faut s'occuper du code. La plupart des écrans nécessite deux librairies, sauf les LCD simples :
- 1 librairie matérielle ;
- 1 librairie graphique.

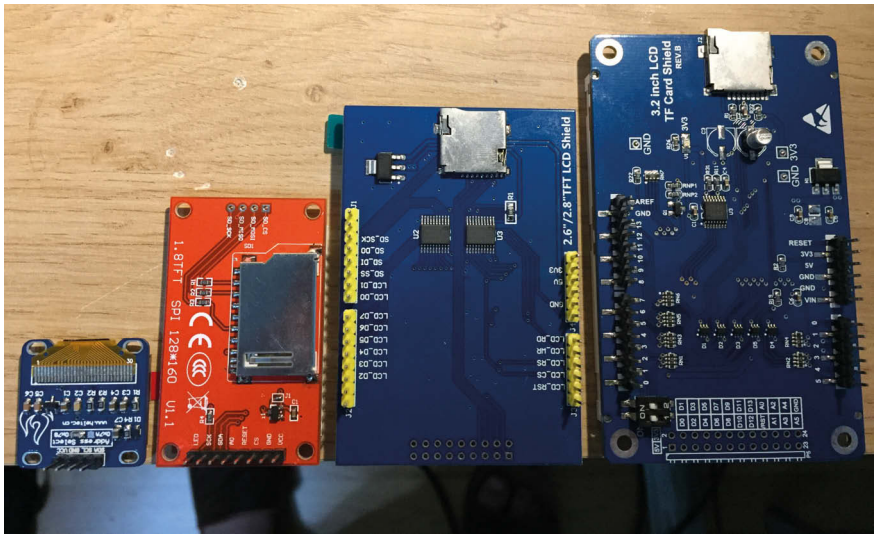
La librairie matérielle permet de supporter l'écran utilisé soit par le constructeur avec sa propre librairie, soit une librairie générique ou compatible. La librairie graphique va permettre d'utiliser les capacités graphiques de l'écran : objets graphiques, textes, animations, gestion des couleurs, etc. Les librairies graphiques ne sont pas identiques et les fonctionnalités non plus.

En OLED, en version I2C, c'est plus simple. Vous aurez avant tout besoin d'une librairie graphique compatible avec un maximum d'écrans, une des plus connues est le U8glib.h. Le brochage est déclaré, comme d'habitude, dans le code.

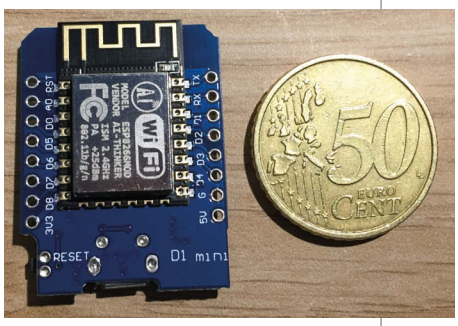
Les librairies graphiques permettent une très grande souplesse dans l'utilisation des écrans : graphismes, images, animations, textes. Certaines librairies sont bien documentées, d'autres non et c'est un des soucis. Et d'une librairie à une autre, nous n'avons pas la même chose. Et certaines librairies fonctionnent mal, ou pas du tout. N'hésitez pas à tester ces librairies sur un code basique.

Et le tactile ?

Vous rêvez de créer un objet tactile ? Malgré les capacités limitées de l'Arduino, il est possible de gérer un écran tactile, mais ne vous attendez pas à des miracles : latence, instabilité de fonctionnement, dégagement de chaleur, difficulté de calibration. Une carte plus puissante est vivement conseillée ainsi que la possibilité d'utiliser différents threads.
Pour une démo réalisée durant les Microsoft Experiences 2016, nous avons utilisé un modèle Adafruit (1), un peu cher, mais de qualité.
Nous utilisons les librairies de base :
- Aafruit_GFX.h
- SPI.h



D1 MINI PRO : L'ESP DE RÉFÉRENCE ?



Les ESP8266 sont de minuscules micro-contrôleurs WiFi aux capacités surprenantes. Sébastien Warin, contributeur régulier de Programmez!, a publié plusieurs dossiers sur ces cartes. Le constructeur Wemos propose d'autres ESP particulièrement intéressants : les modèles D1 mini. La comparaison fait mal à Arduino :

	Arduino Uno	Arduino Nano	D1 mini	D1 mini pro
Puissance	5V	5V	3,3V	3,3V
Microcontrôleur	Atmel 16 MHz	Atmel 16 MHz	ESP 80 MHz	ESP 80 MHz
Mémoire	32 Ko	32 Ko	4 Mo	16 Mo
USB	oui	oui	oui	oui
WiFi	non	non	oui	oui
GPIO (analogique / numérique)	6 / 14	8 / 14	1 / 11	1 / 11
Poids	25 g	6 g	4 g	2,5 g
Dimension	68,6x53,4 mm	45x18mm	34,2x25,6 mm	34,2x25,6 mm
Tarif (carte officielle)	22 $	20 $	5 $	5 $

Les D1 surclassent largement les Arduino sur la puissance du micro-contrôleur, le WiFi par défaut, le poids de la board, et, surtout, le stockage interne, le gros point noir des Arduino. L'absence de 5V peut être gênante pour certains capteurs, mais on gagne en consommation. La possibilité de coder avec Arduino IDE permet aux makers et développeurs de rapidement démarrer avec les boards D1. Il faut parfois un peu tâtonner pour installer le support des ESP sur l'IDE, mais rien de très difficile (pilotes et les fichiers boards). Même si les GPIO sont réduites, la D1, et les ESP8266 en général, promet beaucoup. Très prochainement, nous reviendrons sur cette board.

À cela se rajoutent plusieurs librairies dédiées :
- Adafruit_ILI9341.h -> librairie matérielle pour les écrans Adafruit ;
- Adafruit_STMPE610.h -> librairie pour gérer le tactile résistif de l'écran.

Vous l'aurez compris, vous travaillerez sur deux niveaux : la partie graphique et la partie tactile. La calibration des zones tactiles n'est pas simple. Patience tu auras ! Quoi qu'il en soit, le tactile vous ouvre des possibilités d'interactivités plus profondes avec l'utilisateur. •

(1) 2,8" TFT Touch Shield for Arduino with Resistive Touch Screen, 34,95 $

Eclipse versus NetBeans versus IntelliJ : quel IDE Java choisir ?

• David Wurteisen
Expert Java
SOAT

Nos projets Java grossissent : il faut gérer de plus en plus de classes. Nous utilisions Ant comme outil de build mais la norme est maintenant Maven, quand Gradle pointe le bout de son nez. Nos projets s'enrichissent de bibliothèques et d'autres frameworks qui ont leurs spécificités : Spring, Cucumber avec son langage Gerkins ou tout simplement JDBC où l'on se retrouve à devoir écrire des requêtes SQL au milieu de notre code Java. Pour gérer cette complexité et éviter de passer d'outil en outil, une majorité des développeurs Java utilise un IDE (Integrated Development Environment, soit un environnement de développement « intégré »).

Trois IDE majeurs se disputent le marché sur l'écosystème Java : Eclipse, IntelliJ et Netbeans. Eclipse est certainement le plus connu. Il est Open Source et développé par la fondation Eclipse. La dernière version, Eclipse Neon, est sortie lors de cet été 2016. [Fig1].

Jetbrains commercialise IntelliJ IDEA. Ce dernier, souvent appelé IntelliJ ou IDEA, est décliné en deux versions : la version Ultimate et la version Community. La version Community est gratuite et Open Source, mais se trouve amputée de fonctionnalités que l'on retrouve dans la version Ultimate. C'est cette dernière déclinaison qui sera utilisée dans ce comparatif.

Le dernier challengeur est Netbeans. Cet IDE est sponsorisé par Oracle. Même s'il se fait discret, une nouvelle version devrait très prochainement sortir. Tout comme Eclipse, Netbeans est un IDE gratuit et Open Source.

Les versions utilisées dans ce comparatif seront les dernières versions de chaque IDE disponibles, c'est à dire Eclipse Neon, IntelliJ Ultimate Edition 2016.2 et Netbeans 8.1.

ÉDITEUR JAVA

Autocomplétion

Un développeur va passer une grande partie de son temps dans l'édition de code Java. Tous les IDEs proposent les mêmes fonctionnalités : coloration syntaxique, auto-complétion, mise en avant des erreurs de syntaxe, etc. Pourtant, sur certaines de ces fonctionnalités, ces IDE peuvent traiter le problème d'une manière différente.

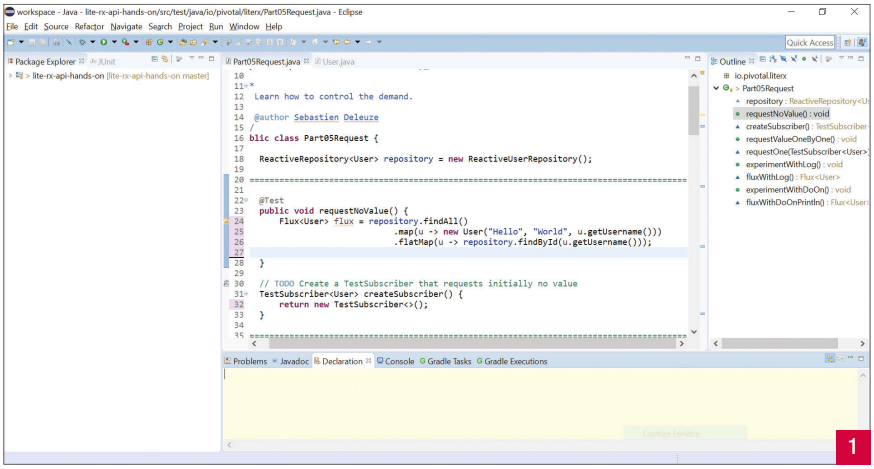
Par exemple, avec le code suivant, que se passe-t'il si on essaye de faire l'auto-complétion lors de l'utilisation de l'interface List et lors de l'auto-complétion, quand on choisit une implémentation ?

```
List<String> myList = new ArrayList<>();
```

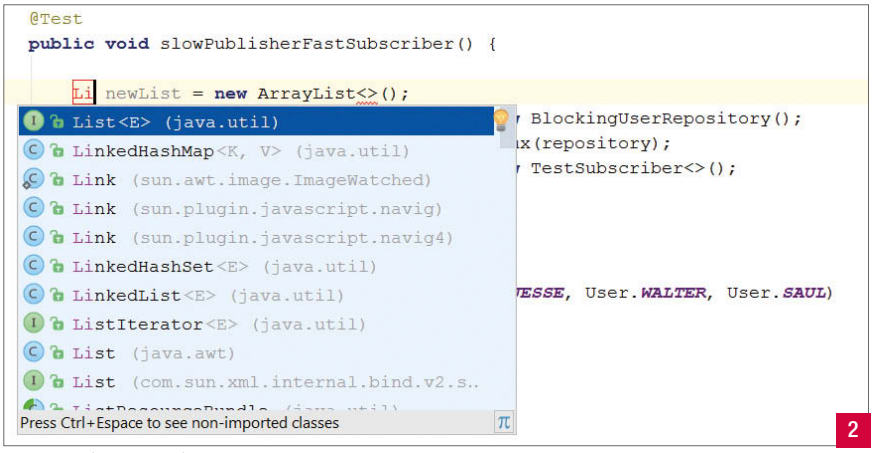
Dans Eclipse et Netbeans, il faut utiliser les raccourcis clavier Ctrl + Espace pour invoquer l'auto-complétion, alors qu'IntelliJ le suggère naturellement au cours de la frappe.

Les différentes propositions de chaque IDE correspondent bien à l'intention du moment : utiliser l'interface List. Netbeans et Eclipse proposent également la Javadoc de la classe voulue automatiquement, alors qu'il faut explicitement la demander via un raccourci clavier sur IntelliJ. [Fig2]

Lors du choix de l'implémentation de notre List, chaque IDE propose une implémentation possible. Ils n'utilisent pas uniquement les lettres déjà tapées pour nous proposer une classe qui n'a pas de lien avec notre contexte. Netbeans



Eclipse IDE



Autocomplétion sous IntelliJ

marque cette séparation visuellement. Ainsi, une barre sépare les propositions qui sont des implémentations possibles de notre List, des autres classes qui correspondent au niveau du nom, mais qui semblent techniquement incompatibles.

L'appui sur la touche "Entrée" permet de valider notre choix. Dans notre exemple, Netbeans affiche également, toujours via le système d'auto-complétion, les différents constructeurs possibles. Ces concurrents quant à eux, se contentent d'afficher ces constructeurs sous forme de bulle.

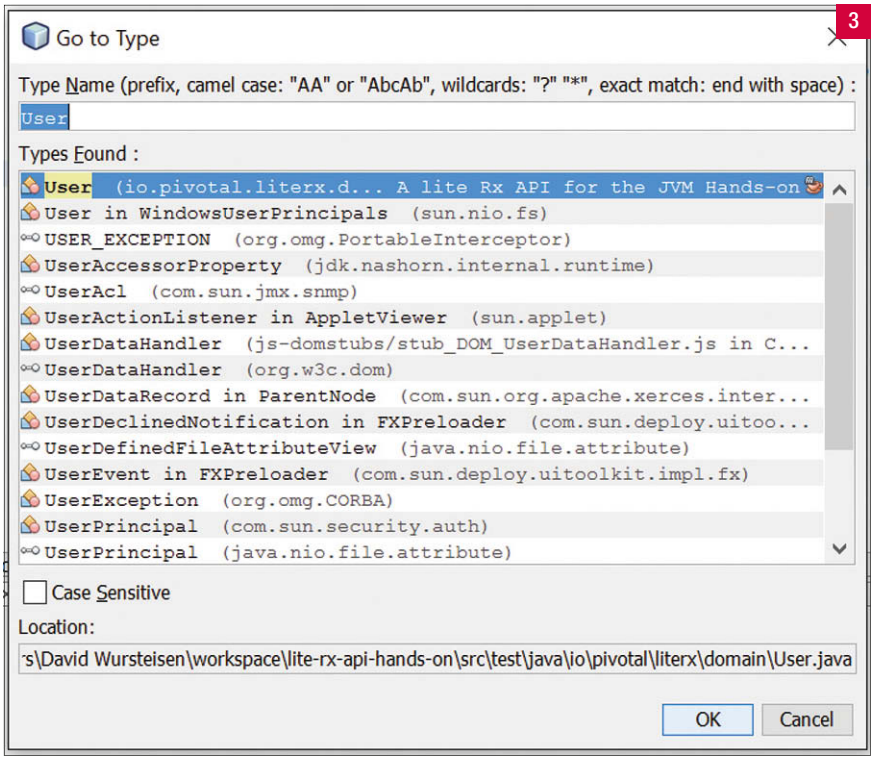
L'auto-complétion semble plus naturelle sous IntelliJ, car elle s'impose d'elle-même, mais à part ce petit détail, la qualité de l'auto-complétion est très proche entre IntelliJ, Eclipse et Netbeans. Toutefois, les différences sont plus importantes sur d'autres fonctionnalités, comme par exemple, le refactoring.

Refactoring

Le refactoring consiste à opérer des modifications structurelles, grâce à l'éditeur de code. Cela peut être une opération simple, comme renommer une variable, ou représenter une opération plus conséquente : remplacer des paramètres par un objet, et modifier automatiquement tous les appelants, par exemple. Chaque IDE propose grossièrement les mêmes capacités : introduction de variable, de constante, déplacement de méthodes dans une autre classe, introduction d'une interface… Mais l'utilisation de ces fonctions de refactoring diffère entre chaque éditeur.

Si l'on veut extraire une partie de notre code dans une méthode, sous Netbeans, il est nécessaire de sélectionner le code que nous voulons extraire, autrement un message survient indiquant un problème au niveau du code sélectionné. Eclipse, lui, affiche l'option dans le menu de refactoring uniquement si la sélection est réalisée correctement. Si on force l'opération à l'aide de raccourci clavier, Eclipse n'affiche pas de message d'erreur si nous n'avons pas de code sélectionné. Ce message d'erreur ne s'affiche que si la sélection du code est présente et invalide pour lui. IntelliJ, quant à lui, s'efforce systématiquement de réaliser l'opération. Selon la position du curseur, IntelliJ va sélectionner le bloc de code le plus logique pour introduire une méthode. Il est assez dur de le mettre en défaut, et quand cela arrive, il indique l'impossibilité de l'opération par un petit tool tip rouge. Ce petit tool tip est moins intrusif que les popups d'erreurs.

Même si techniquement, le résultat est identique pour chaque IDE, Eclipse et Netbeans ne sont pas très souples quand on utilise une fonction de refactoring, et il n'est pas toujours facile de trouver comment déclencher la méthode de refactoring sous Eclipse. A contrario, IntelliJ tente systématiquement de réaliser une opération, quitte à faire des suppositions dans notre intention. Le refactoring est donc moins fastidieux à réaliser grâce à ce dernier.



Navigation entre class sous Netbeans

```
21
22  @Test
23  public void requestNoValue() {
24      Flux<User> flux = repository.findAll()
25                              .map(u -> new User("Hello", "World", u.getUsername()))
26                              .flatMap(u -> repository.findById(u.getUsername()));
27
28  }
29
```

4

Support Java 8 sous Eclipse IDE

Navigation dans un projet

Pouvoir naviguer de fichier en fichier simplement sans avoir à parcourir l'ensemble des packages de son projet est obligatoire, surtout avec un projet ayant un nombre conséquent de class. Passer d'onglet en onglet via le clavier fait gagner énormément de temps. Que ce soit Netbeans, Eclipse ou IntelliJ, chacun propose une recherche par nom de class ou par nom de fichier. IntelliJ propose en plus une recherche par symbole : pratique pour retrouver la méthode dont vous avez le nom, mais dont le nom de la class vous échappe. Il est dommage que contrairement à ses concurrents, Eclipse ne propose pas en priorité les class de notre projet. [Fig3]

La navigation entre les fichiers ouverts se fait simplement, que ce soit sous Eclipse, IntelliJ ou Netbeans. Chaque IDE propose un raccourci clavier pour passer à un autre fichier rapidement. Eclipse et IntelliJ partagent le même modèle de navigation (et le même raccourci clavier ! Ctrl + E). Les IDEs affichent dans une liste les fichiers ouverts, nous permettant de sélectionner celui qui nous intéresse avec les flèches du clavier, ou, mieux, de filtrer sur le nom de la ressource qui nous intéresse. Netbeans est moins intéressant de ce point de vue là : il n'est pas possible de filtrer rapidement par nom.

Support Java 8

Netbeans a été le premier à supporter Java 8 et notamment les fameuses lambdas. IntelliJ lui a embrayé le pas. Eclipse a tardé à supporter les lambdas mais il a rattrapé son retard. Tous proposent des nouvelles actions de refactoring : passer d'une classe anonyme à une lambda, et

vice et versa ou encore introduire une référence de méthode. Ce n'est pas sur ce critère que vous choisirez votre IDE préféré : et c'est tant mieux pour nous ! [Fig4]

Support de la gestion de version

Chaque éditeur offre, à travers leurs gouttières de droite, une vision macros des warnings et erreurs présents dans l'ensemble du fichier actuellement ouvert. La gouttière de gauche, quant à elle, affiche généralement les numéros de lignes, mais également, quand un gestionnaire de source est utilisé, le statut de la ligne concerné : la ligne a-t-elle été ajoutée, modifiée ou un bloc de code a-t-il été supprimé ? De plus, un clic sur la gouttière affiche des options permettant de faire un retour arrière sur les modifications, en affichant les différences sur IntelliJ et Netbeans ! Car même si Eclipse propose le même mécanisme, la référence utilisée, par défaut, est la dernière version présente sur le disque dur et non la dernière version de son système de version préféré. Ainsi, dès que l'on sauvegarde, les informations présentes dans la gouttière disparaissent... [Fig5]

Git est supporté par tous les IDEs et ceci par défaut. Subversion est également supporté mais comme ce gestionnaire de source se fait remplacer par Git, son intégration ne sera pas testée. Chaque IDE propose les opérations de base : push, pull, merge, branch…IntelliJ affiche en plus la branche courante. Mais en termes de fonctionnalités avancées, IntelliJ se retrouve un peu plus limité que ces concurrents. Par exemple, rajouter un dépôt distant n'est pas possible à travers les menus. On devra alors se retourner vers la ligne de commande.

En aparté

L'ampoule est le symbole utilisé par IntelliJ, Netbeans et Eclipse pour avertir l'utilisateur d'un quick-fix possible, selon la position du curseur. Pour connaître les propositions de chaque IDE, le raccourci utilisé est Ctrl+Entrée sauf sur Eclipse, où le raccourci est Ctrl+1. Ce raccourci propre à Eclipse semble avoir été conçu pour des claviers QWERTY, en effet sur un clavier AZERTY, il faut ajouter la touche Majuscule en plus. Ce petit détail nous incite à penser que cet IDE n'a pas été développé pour nous autres utilisateurs de claviers AZERTY. Toujours est-il que nos trois IDEs font des propositions pertinentes, sauf à de rares occasions, mais qui se rapportent à des cas très spécifiques. [Fig6]

SUPPORT OUTIL DE BUILD

Maven

Maven est généralement utilisé comme outil de build. Quand ce n'est pas le cas, le projet utilise alors Gradle. Comment Eclipse, Netbeans et IntelliJ supportent-ils ces deux outils de builds majeurs dans l'écosystème Java ?

Par défaut, tous ces IDEs supportent Maven. Ouvrir un projet Maven, s'apparente à ouvrir un projet standard sous Netbeans et IntelliJ. Les dépendances et plugins du projet Maven sont détectés automatiquement par les deux IDEs, et tous les goals possibles sont correctement détectés. De plus, les deux IDEs séparent les sources dites "de production" et les sources utilisées pour les tests. Si Maven est efficace et simple d'utilisation, qu'en est-il d'Eclipse ?

Avec Eclipse, il faut tout d'abord importer son projet : les dépendances du projet et les répertoires contenant le code source sont bien détectés. Contrairement à ces concurrents, Eclipse détecte bien deux répertoires contenant des sources différentes : un pour le code source de production, un pour le code source de test, mais ne les classe pas dans des catégories différentes. Eclipse ne sachant pas différencier le code de test du code de production, on peut

```
@Test
public void slowPublisherFastSubscriber() {

    newList = new ArrayList<>();
    B                                          BlockingUserRepository();
    F                                          (repository);
    T                                          TestSubscriber<>();
    t
                                               .await()
            .assertValues(User.SKYLER, User.JESSE, User.WALTER, User.SAUL)
            .assertComplete();
}
```

Create local variable 'newList'
Create field 'newList'
Create parameter 'newList'
Rename reference
This file is indented with tabs instead of 4 sp...

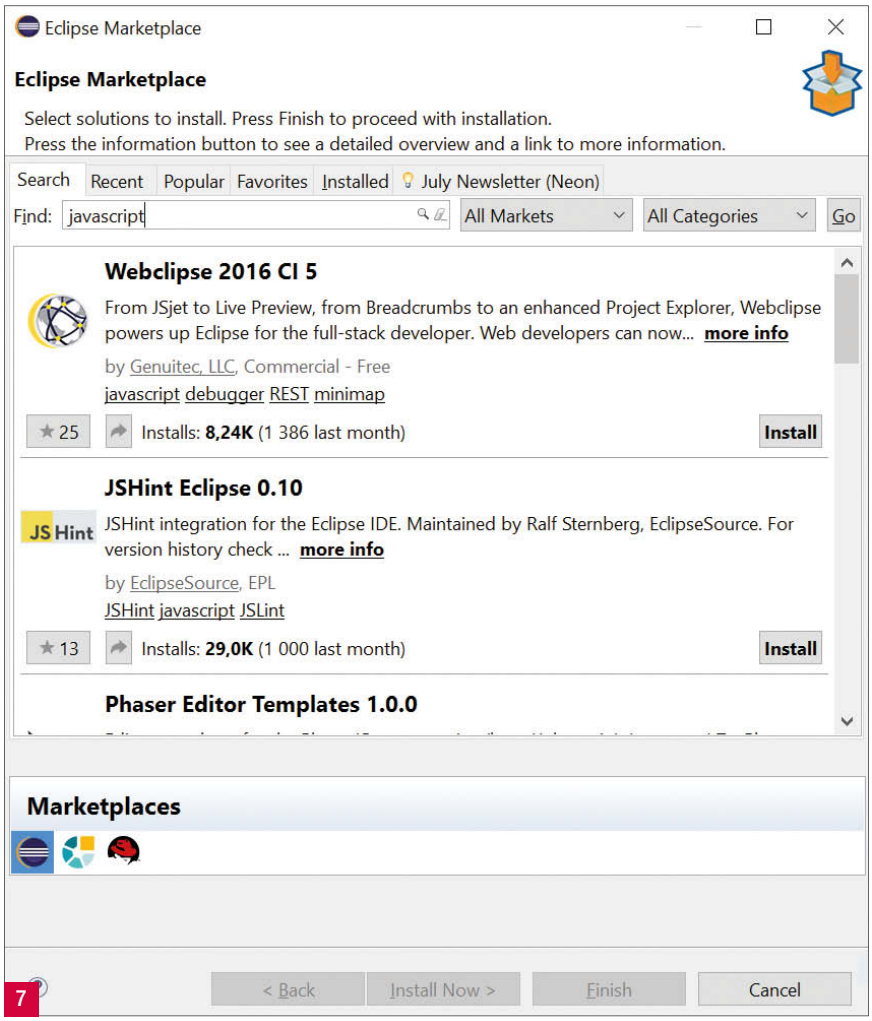
6

Quick-fix sous IntelliJ

```
54
55          // TODO Merge flux1 and flux2 values with no interleave (flux1 values, and then flux2 values)
56          Flux<User> mergeFluxWithNoInterleave(Flux<User> flux1, Flux<User> flux2) {
57              return Flux.concat(flux1, flux2);
58
59              return null;
60
61
```

5

Quick Diff sous Netbeans



se retrouver à utiliser des classes de test dans du code de production, sans aucune remarque de la part d'Eclipse, là où Netbeans et IntelliJ auraient remonté une erreur.

De même, Eclipse ne propose pas les différents goals Maven disponibles au sein du projet Maven, là où Netbeans et IntelliJ proposent tous ceux décrits dans le fichier pom.xml. Le support Maven dans Eclipse s'avère bien pauvre comparé à ses concurrents

Gradle

Maven n'est pas le seul outil de build : Gradle ne peut plus être ignoré. Netbeans ne le supporte pas par défaut, mais un plugins corrige ce manque. L'ouverture du projet dans l'IDE se réalise sans problème ainsi que la construction du projet. Malheureusement, le support Gradle s'arrête là : la liste des "tasks" spécifiques à l'outil n'est pas présentée. Par défaut, c'est la commande gradle build qui est utilisée. Eclipse et IntelliJ quant à eux, gèrent les projets Gradle de la même manière et affichent les différentes "tasks", etc. Mais ils ont aussi les mêmes défauts : il est impossible d'ouvrir un projet Gradle invalide : un message d'erreur, dans chaque IDE, indique l'impossibilité de l'opération et tout s'arrête là. On doit donc corriger notre fichier build.gradle dans un autre logiciel que notre IDE.

Plugins

Les IDEs ne peuvent pas, de base, supporter tous les frameworks, tous les langages existants, etc. Pour pallier ce problème, chaque IDE peut se voir rajouter de nouvelles fonctionnalités grâce à un système de plugins. C'est ainsi que Netbeans peut supporter Gradle.

IntelliJ propose un catalogue fourni de plugins de qualité quand Netbeans lui, propose un catalogue beaucoup plus restreint. Le nombre de plugins existants sur l'IDE d'Oracle est faible, mais les plugins couvrent une bonne partie des besoins : on y trouve un plugin pour Gradle, Scala, jRebel. En revanche, vous n'en trouverez pas pour le support de Spring ou de Cucumber avec son langage Gerkins. Selon votre projet, l'absence de ce type de plugin peut être pénalisant. [Fig7]

A contrario, il est dur de trouver une technologie qui n'ait pas son plugin dans Eclipse. Le marketplace d'Eclipse est rempli de plugins. Certains ont comme particularité d'avoir une licence commerciale. Le choix des plugins venant avec chaque packaging d'Eclipse peut quant à lui poser question : Eclipse pour développeur Java vient par défaut avec le plugin Gradle, mais sans support Javascript. Alors qu'Eclipse pour développeur Java EE, lui, est installé avec le support Javascript mais sans le plugin Gradle. Il faut donc choisir la bonne distribution dès le départ, si on ne veut pas à avoir à installer une pléthore de plugins dès le premier démarrage.

La qualité des plugins est dépendante de leurs auteurs. Difficile ici de différencier un IDE sur ce type de critère. Par exemple, le plugin Gradle pour Eclipse est de très bonne facture mais un autre plugin peut très bien dégrader votre expérience utilisateur.

Finalement, quel IDE choisir ?

Aucun des IDE n'est mauvais et quel que soit votre choix, il sera viable à l'usage. Pourtant, la qualité, la pertinence des propositions, le support de différentes technologies fait qu'IntelliJ est un IDE de choix. Ensuite viennent Netbeans puis Eclipse, même si ce dernier est entaché par des défauts mineurs ou un manque de consistance.

Dans votre contexte, d'autres facteurs devront être pris en compte. Il est plus difficile d'imposer IntelliJ dans une entreprise quand on parle uniquement de coût à l'achat : Netbeans et Eclipse sont imbattables car gratuits. Sur le créneau des IDE gratuits, il est difficile de choisir entre Netbeans et Eclipse. Dans ce cas, les facteurs différenciants vont se situer au niveau de la communauté : le market place d'Eclipse déborde de plugins, alors que Netbeans semble être la cinquième roue du carrosse : le choix est conséquent mais les éditeurs semblent privilégier Eclipse (et IntelliJ), plus rarement Netbeans.

Un dernier point : Jetbrains améliore son produit. Il y ajoute des fonctionnalités sur son logiciel ou dans sa gamme de produits, qui se fait - pour l'instant - sans perte de qualité. La fondation Eclipse, s'efforce à améliorer son image, égratignée ces dernières années, en corrigeant des bugs sur son IDE et en y ajoutant des fonctionnalités innovantes. Quant à Netbeans, son développement est lié au sponsoring d'Oracle. Ce dernier ne semble pas communiquer beaucoup dessus, laissant ainsi laisser planer le doute sur le futur de Netbeans. C'est dommageable pour cet IDE de qualité, car la roadmap du produit est rassurante. •

1ER ÉVÉNEMENT EUROPÉEN
LIBRE & OPEN SOURCE

île de France

MAIRIE DE PARIS

EMPOWERING
OPEN INNOVATION
opensourcesummit.paris
#OSSPARIS16

PARIS OPEN SOURCE SUMMIT

16 & 17 NOVEMBRE 2016

DOCK PULLMAN
Plaine Saint-Denis

SPONSOR DIAMOND

Inria — INVENTEURS DU MONDE NUMÉRIQUE

SPONSORS PLATINUM

alter way · Microsoft · Smile OPEN SOURCE SOLUTIONS

SPONSORS GOLD

BlueMind Messagerie & espaces collaboratifs · henix

SPONSORS SILVER

AXEL IT ZABBIX · axelor ACCÉLÉRATEUR DE PERFORMANCE · centreon · cozy.io · ORACLE · redhat · Savoir-faire LINUX · XiVO open-minded telecom systems · YaZiba.net Powered by zimbra

POUR TOUTE INFORMATION COMPLÉMENTAIRE :
Email : contact@opensourcesummit.paris – Tel : 01 41 18 60 52

un événement
Systematic Paris Region Digital Ecosystem · Tarsus FRANCE

CTO Partners, le premier fonds d'investissement des CTO et managers IT français

Fin 2014, deux anciens de l'executive MBA Epitech, Alexis Huet et Olivier Quéméneur, forts des compétences acquises en finance et en entreprenariat, imaginent l'idée d'un nouveau fonds. Des échanges avec Gilles Enguehard (gérant du fonds Network Finance) et Cyril Pierre de Geyer (serial entrepreneur et Business Angel) les ont amenés à ce constat : les start-ups early stage présentent globalement un déficit sur la partie technologique, ce qui est un frein pour leur recherche de fonds.

• Jean-François Naud – *Freelance 10 ans, il accompagne Relais & Châteaux dans sa transformation digitale depuis 2012. Après les projets Hospitalité sur Mobile (Travel d'Or iPad 2014), il pilote la refonte du site www.relaischateaux.com.*

• Cyril PIERRE de GEYER *exerce actuellement plusieurs missions Directeur des Executive MBA du groupe Ionis, Professeur affilié à HEC au département Entrepreneuriat et Innovation, Directeur de l'organisme de formation informatique AgoraTIC et Business Angel.*

CTO Partners est l'illustration d'une mutation : ce nouveau fonds d'investissement créé par des managers IT passionnés et officiant dans les plus belles entreprises de la Frenchtech donne un nouveau souffle à l'entreprenariat et à la transformation digitale française.

« *Trois compétences sont essentielles pour monter une startup : business, marketing et technologique. La défaillance de l'une de ces compétences peut compromettre un projet* » explique Olivier Quéméneur, Président de CTO Partners.

Le principal point de blocage des start-ups du numérique : l'expertise technologique

La France manque d'experts et de managers de l'innovation capables de transformer une idée en un produit ou un service. Trouver un associé CTO (Chief Technical Officer) est le premier point de blocage rencontré par les entrepreneurs qui veulent lancer leur projet. Si les événements de mise en relation sur ce thème fleurissent (adopteuncto, meetup...), les rares experts techniques qui s'y rendent sont sollicités de toute part. C'est pourquoi des managers IT se sont regroupés pour créer CTO Partners, le premier fonds d'investissement des CTO. Issus des grands noms de la « tech française » (BlaBlaCar, Vente privée, Le Bon Coin…) et des meilleures formations techniques et managériales (EPITA, Executive MBA Epitech …), ces passionnés accompagnent les start-ups, en recherche d'expertise technologique pour :

- Transformer une idée en un projet fonctionnel,
- Aider au recrutement,
- Construire les bases de la croissance et de l'industrialisation.

« *J'adore la technique, pouvoir créer avec l'outil informatique et je suis resté hyper curieux. CTO Partners me permet de mettre mon expertise au service d'entreprises passionnantes. Ma spécialité : préparer le passage à l'échelle supérieure d'une architecture IT d'entreprise. Forcément avec* Le Bon Coin *: la haute disponibilité c'est notre dada !* » explique Jean-Louis Bergamo, CTO Partners et Responsable des Infrastructures de *Le Bon Coin*.

La méthode CTO Partners

Chaque projet retenu est suivi par un « parrain » qui l'accompagne pour favoriser l'éclosion d'une future pépite. Outre le financement (plus d'un million d'euros en valeur à investir), la startup a ainsi accès à l'expertise de tous les CTO Partners :

- **Une optimisation du « time to market »**, par l'accompagnement à la réalisation du MVP (*Minimum Viable Product*) et l'apport d'expertise et de capital ;
- **Une expertise technologique** pour crédibiliser et consolider le projet ;
- **Une aide au recrutement** grâce aux écoles partenaires et à l'incroyable réseau des associés ;
- **Un vaste réseau d'acteurs du monde des start-ups**, des fonds d'investissement partenaires et des directeurs des systèmes d'information de grandes entreprises.

L'approche de CTO Partners n'est pas de prendre la place d'un CTO ni de rentrer dans le développement pur et dur, mais plutôt de poser de bonnes bases technologiques, de monter les bonnes équipes, de financer et de préparer la croissance.

Focus sur la start-up Vidmizer

Vidmizer est une plateforme de marketing vidéo que CTO Partners accompagne depuis mai 2016. Leur outil permet aux marques de centraliser, gérer et multiposter tous leurs contenus vidéos sur les réseaux sociaux, les plateformes vidéos et tous les sites Internet qui leur sont propres. C'est Alexis Huet (Fondateur de l'Agence Serum & Co) le parrain de Vidmizer : après avoir challengé, posé les choix technologiques et préparé la roadmap de la plateforme à horizon 2019, il a guidé l'équipe dans la sélection des meilleures solutions pour la réalisation du MVP. Par l'intermédiaire du parrain, Vidmizer a pu accéder à tous les CTO Partners : Nicolas Silberman (ex CTO de Mediapart et 20Minutes, fondateur de ThankYouMotion) a mené les choix technologiques pour l'architecture technique. Côté hébergement, point critique pour les vidéos, Clément Moulin, Fondateur de l'hébergeur SimpleZero, a apporté son expertise pour identifier le meilleur ROI possible. Enfin, le CTO de Vidmizer a été recruté avec l'aide de Samir Rinaz (Directeur de l'école ETNA) et Jean-François Naud (ex-Responsable tech des projets Web et Mobile de Relais & Châteaux)

Interview du président de CTO Partners : Olivier Quéméneur

Il y avait vraiment besoin d'un fonds d'investissement de plus ?

Oui, il manquait d'un fonds apportant une vraie valeur CTO et pas que du financement.

On s'est regroupés avec des managers IT de boites assez connues pour créer CTO Partners, un fonds d'investissement *Smart Money*, orienté sur la valeur Tech.

Ça veut dire que les CTO en question donnent de l'argent et puis donnent surtout de l'accompagnement ?

Tout à fait. On part plutôt sur de petits investissements. On va valoriser davantage l'accompagnement : c'est là la valeur qu'on apporte.

L'investissement sera la cerise sur le gâteau pour montrer notre engagement à l'accompagnement. L'idée c'est du PowerPoint au MVP, au *Produit Minimum Viable* vendable. Tout ça en 6 mois à peu près.

Ça veut dire qu'on peut créer sa startup et qu'on peut se retrouver accompagné, drivé par le Head of Engineering de BlaBlaCar par exemple ?

Exactement !

www.ctopartners.fr •

Recrutement : du choix, mais pas pour tous les profils

François Tonic
Programmez!

Si vous suivez un peu l'actualité, vous savez que l'on entend chaque semaine, ou presque, que les entreprises cherchent des développeurs, que l'on manque de développeurs, que vous trouverez « facilement » un poste, etc. La demande de développeurs reste forte pour étoffer les équipes existantes, dans les startups et les nouvelles technologies. Les profils demandés sont très disparates, avec des grilles tarifaires élastiques.

Oui, le marché des développeurs est tendu depuis plusieurs années, non, cela ne concerne pas tous les profils. Certains profils sont plus faciles à trouver que d'autres. Développeur Web (front-end, back-end), intégrateur Web ne sont pas les profils les plus rares, notamment avec les jeunes diplômés et les reconversions professionnelles. Mêmes remarques pour les compétences CMS, Java, JavaScript, avec pas ou peu d'expérience. En revanche, quand on commence à demander, au minimum, 3-4 ans d'expérience, il y a déjà moins de profils disponibles : plus, on monte dans l'expérience de technologies précises, plus cette tendance se confirme.

Cependant, on constante toujours de nombreuses offres d'emplois exigeant un développeur à 5 pattes : maîtrise du front et du back, fullstack sur x technologies/langages, maîtrise de l'agilité, du mobile, avec x années d'expérience, le tout pour un salaire tiré vers le bas ou tout juste en adéquation avec le profil demandé. Le chômage ambiant et l'économie toujours moribonde, sur de nombreux secteurs, ne doivent pas brader systématiquement les compétences..

Chercher à tout prix un mouton à 5 pattes n'a aucun sens, surtout quand dans cette catégorie de compétences, le développeur aura souvent plusieurs choix. Et l'entreprise demandeuse risque de recruter des profils juniors avec nécessité d'une mise à niveau technique sur x mois. L'offre d'emploi doit être réaliste, mais le développeur doit aussi être réaliste par rapport à ses propres compétences.

Un profil technique recherché est très important pour le salaire

Les écarts salariaux existent toujours et ils ne se réduisent pas, notamment quand on regarde l'île de France et la province ; même si certaines métropoles régionales, très dynamiques, offrent des conditions salariales égales ou proches de celles de la région parisienne. Par contre, certains profils, en début de carrière, restent toujours très bas : 27-28 000 bruts comme nous pouvons le constater pour un développeur Web/PHP/intégrateur Web. Mais il faut bien débuter sa carrière.

Si vous regardez les grilles de chooseyourboss.com, un développeur PHP (je parle bien de PHP pur), avec un profil junior débutera parfois à 23-28 000 bruts. Un profil CMS pourra améliorer cette base, idem si vous maîtrisez des frameworks.

À vous de monter en compétences, de vous former, d'être toujours en veille technologique. La stagnation technique est l'ennemi du développeur, surtout si vous souhaitez évoluer dans votre carrière de développeur. À vous aussi

Développeur Web

Le développeur est une pièce maîtresse dans un projet informatique et principalement web. Son environnement évolue tous les jours, ou presque. L'envie de changer varie suivant les personnes et les cycles des projets. Nous constatons un renouvellement régulier des équipes (le turn-over peut être très élevé, NDLR). En général, le développeur change pour les mêmes raisons qu'un autre métier : les conditions de travail.

Les motivations

Les motivations qui poussent le développeur Web (et pas seulement un dév Web) à vouloir changer d'entreprise, sont très diverses. Notre secteur étant en constante évolution, pour l'entreprise, c'est souvent un casse-tête. Tout d'abord, les cabinets de recrutement sont en recherche constante de nouveaux talents pour renforcer les équipes de leurs clients, avec différentes promesses et avantages pour inciter les candidats à répondre à leurs appels.

Ensuite, la fin d'un gros projet (en moyenne 3 ans), ou le positionnement dans une catégorie ne fera pas forcément évoluer le développeur et montrera parfois un avenir morose dans son poste actuel. Bref, le développeur va vouloir chercher de nouveaux défis. Enfin, le parrainage, ou la cooptation, est une méthode de recrutement pour se faire débaucher, ou rejoindre une connaissance.

Le poste

La sélection d'un poste est très variable, car même si le salaire est un point important, le développeur pense aussi à la mission, son cadre de vie, l'environnement où il va arriver, et dans quel but. Le lieu de la mission est un critère non négligeable, principalement dans les grandes agglomérations avec le temps de transports (un point à ne pas négliger). Pour certaines personnes, la formation, la participation, la contribution à des projets utilisés dans les entreprises sont des motivations importantes. De plus en plus la demande de télétravail, au moins sur une journée, est un critère demandé, qui sera à prendre en considération et bénéfique pour l'entreprise ou pour certains postes occasionnellement.

En résumé

Pour ma part, le changement de poste ne m'a jamais fait peur, même lors de démissions, sans contrat pour la suite, car l'ensemble des motivations présentées dans l'article sont des critères importants à un instant précis, et même si je ne suis pas à l'écoute intensive du marché, je surveille toujours les propositions au cas où.

• Christophe Villeneuve

d'être proactif, votre employeur ne le sera pas toujours. Et malheureusement, des entreprises préfèrent parfois recruter des développeurs débutants que des profils expérimentés, pour une question de salaires… Les développeurs Java, C# (monde .Net en général) demeurent des valeurs sûres du marché, et la demande est toujours là, avec des salaires très proches.
Sur le développement mobile, le marché demeure tendu. Aujourd'hui, les développeurs maîtrisant à la fois Android et iOS sont recherchés et c'est aussi une demande du marché des apps car le parc mobile est divisé entre ces deux plateformes. La compétence Windows Mobile peut être un plus, mais pas en profil principal. Le marché sur cette plateforme étant actuellement très faible, les entreprises et éditeurs miseront sur les leaders du marché.

Un C.V. ne fait pas tout, mais l'entreprise doit aussi comprendre le développeur

Restez ouverts au marché, regardez ce qui s'y passe. Ne croyez pas que seul votre C.V. vous fera recruter. À vous de vous vendre, mais à l'entreprise demandeuse de se vendre aussi. Car l'entreprise n'est plus forcément en position de force, surtout pour attirer des talents du code. Comme nous l'avions écrit au printemps dernier, il faut en finir avec les annonces stéréotypées et toutes les mentions plus ou moins rébarbatives :

- Cherche développeur mobile ou développeur Web : soyez précis dans la recherche, indiquer les plateformes, les technologies ;
- Rigoureux, ponctuel, etc. : est-ce vraiment utile de l'indiquer ?
- Mettez en avant l'ambiance, les avantages, les technos utilisées (et si possible les plus récentes) ;
- Possibilité de donner des heures pour assister à des conférences, pour faire de la veille technologique, etc.

N'hésitez pas à monter votre book, à mettre en avant votre participation dans les communautés ou dans Programmez ! (eh oui, contribuer à Programmez ! pourra faire partie de votre book) ou encore les hackatons auxquels vous participez. Vous avez là une valeur ajoutée que tout développeur n'a pas forcément. Et renseignez-vous sur l'activité de l'entreprise, ses marchés, etc.

Quelques compétences à suivre, ou pas

Le tableau est donné à titre indicatif et n'est nullement exhaustif. Il est ce que nous voyons à la rédaction.

Technos/langages	Tendances	Commentaires
Java, .Net (C#, ASP.Net)	=/+	Des valeurs sûres du marché. Attention à vous former aux nouvelles évolutions des langages et des outils.
JavaScript, frameworks JS, Node.JS…	++	C'est le langage phare du Web actuellement. La demande est là. La maîtrise des frameworks est vivement recommandée.
Cloud Computing	++	De plus en plus, le développeur va développer pour le Cloud, utiliser des services Cloud même si la petite PME ne sera pas forcément concernée immédiatement, ou que l'entreprise ne l'utilise pas encore pour ses développeurs.
Python	+	Python reste méconnu en France et pourtant il s'agit d'un langage en pleine progression notamment en infrastructure. Les offres d'emploi dédiées Python sont peu fréquentes. À surveiller et à apprendre.
Mobilité	++	Bien entendu.
Xamarin, Cordova…	=/+	Les solutions de développement mobiles sont nombreuses. Si Xamarin fait beaucoup parler de lui, ce n'est pas la solution la plus connue et la plus utilisée en France.
CMS	++	Les CMS demeurent les solutions phares du Web et des projets Web. Une compétence à acquérir pour aller au-delà de PHP. Drupal reste la reference.
HTML/CSS	-/=	Oui c'est la base du Web, mais non, ne vous contentez pas d'eux.
Wearable	?	C'est tendance, c'est geek, mais pour le moment ce marché se cherche. Suivez-le, testez les solutions, montez des PoC.
IoT	++	Oui, oui et oui. Suivez attentivement ce qui s'y passe.
Réalité virtuelle/réalité augmentée	++	Tendance lourde, marché quasi inexistant pour le moment. Mais attention, il va rapidement décoller. Si vous souhaitez travailler sur un marché prometteur et innovant, c'est celui-ci ! Conseil : ne vous enfermez pas dans une seule techno.
Big Data, Machine learning, Deep learning, IA…	++	Ces domaines explosent depuis plusieurs années, les outils se multiplient.
Ruby, Scala, Go, Rust…	=/+	De nombreux langages émergent, notamment sur des usages pointus, les offres sont peu nombreuses, mais souvent très intéressantes. Ruby réapparaît après plusieurs années de silence.
Nouvelles architectures	++	Quand j'entends en soutenance informatique : Docker c'est quoi ? Je saute au plafond. Les nouvelles architectures (conteneurs, micro-services, lambda, etc.) sont déjà là et ce sont les fondations des prochaines années.
Sécurité du code, qualité du code, performances…	++	Bien entendu, mais est-ce encore utile de la préciser année après année ?
Agilité, DevOps	++	Là encore, aucune surprise. Connaissez les outils et les méthodes. Quand je vois passer des offres d'emplois « développeur DevOps » je me demande de quoi on parle…
Open Source	++	Oui et encore oui. Aujourd'hui, les outils, les librairies Open Source sont utilisées partout.
C, C++	=	Dans le monde de l'embarqué par exemple, le plus souvent ce sera un profil C++. Peu de recrutement spécifiquement C ou C++ mais ce sont les bases.

Introduction à notre tableau recrutement.

Durant plusieurs semaines, les entreprises pouvaient répondre à notre questionnaire. Voici les synthèses des réponses données :

- Le secteur d'activité : les technologies pour 37 % des entreprises puis les services, les services Internet et le consulting ;
- Pour les 2/3 ce sont des SSII (soit SSII, soit des purs players), les PME représentent 23 % ;
- Les entreprises ayant répondu sont très diverses par la taille : 25 % ont entre 101 et 250 personnes, et 1/3 dépassent les 250 personnes.

Comme vous le verrez, les profils recherchés sont très variés :

- Développeurs PHP, DRUPAL, Symfony ;
- Développeurs Java ;
- Compétences DevOps, agilité, CMS ;
- Développeurs C++ ;

Développeurs .Net.

Nous retrouvons aussi des compétences peu fréquentes comme Python, Go, embarqué.

QUAND UN OPS CHANGE DE JOB

Étienne Deneuve, évangéliste technologique chez ISI Expert, nous parle de son expérience de changement de postes.

Pourquoi as-tu eu envie de changer ?

J'ai eu envie de changer de poste, car celui que j'occupais, n'était pas en adéquation avec mes opinions en termes de management d'équipe (poste de responsable IT). De plus, chez le client final en IT, les tâches sont très répétitives et tournent toutes autour du MCO ; avec souvent des difficultés chez un ISV à faire changer les modes opératoires et/ou les technologies (lorsqu'elles sont obsolètes par exemple), ceci afin de rendre le SI plus performant ou « secure » en interne ou vis-à-vis des clients. Je suis aussi de cette génération Y, les changements de postes ne me font pas peur, si je ne trouve pas ce que je souhaite ou si j'ai l'impression de ne pas être écouté, je préfère partir. Les entreprises qui payent des personnes « cher » ne doivent pas se dire, on les paye cher, ils vont rester. Ça fonctionnait peut-être avec les anciennes générations, mais plus avec la mienne, on ne vient plus travailler pour un salaire uniquement, mais aussi pour un échange et avoir du « fun » au travail. Pour moi, si on me paye « cher », je dois être utile, avec une valeur ajoutée dans ce que je produis ; non ce n'est pas de la prétention, mais un constat, un salarié « cher » est cher parce qu'on reconnaît son expertise ou son expérience, dès lors que les choix proposés par celui-ci sont remis en question, c'est là que la relation de confiance n'est plus là.

Quand tu as changé comment s'est fait le choix, qu'est-ce que tu cherchais ?

J'ai fait plusieurs entretiens avec des entreprises de type SSII de tailles moyennes en IT (10 à 200). Je cherchais un poste de type consultant Avant-Vente, ou évangéliste. Les postes en clients finaux ne sont pas faits pour moi, ce n'est pas assez « vivant » et la partie veille technologique est difficilement réalisable. Dès lors, avant même les entretiens, j'ai préféré mettre un focus sur les petites SSII avec un fort potentiel techno et une ambiance sympa. Je cherchais également à partir sur les technos Cloud (Aws et Azure), les infrastructures hyper convergées, du docker, et de l'infrastructure as a code (IaaS).

Comment t'es-tu préparé pour les entretiens ?

Pour ma part, les entretiens d'embauche ne sont pas un souci, je suis à l'aise à l'oral et donc je n'ai pas trop préparé les entretiens, mis à part les vérifications sur l'entreprise. Il est important de savoir quel type de clients l'entreprise sert. Pour ma part, je ne souhaitais pas travailler avec des « grands comptes », et donc je devais m'assurer que ceux-ci ne sont pas dans la stratégie de l'entreprise.

As-tu été surpris par certaines demandes de compétences ou ce que l'on te proposait ?

J'ai été surpris par certaines qui proposent des « Packages » avec tout dedans, on dirait qu'ils cherchent à grossir les chiffres en incluant les primes, les choses obligatoires (mutuelle, primes vacances, Syntec). Sinon, la compétence demandée était en adéquation entre ce que je souhaite apprendre ou approfondir, ou que je connais déjà, là-dessus pas de surprise.

Finalement, comment s'est fait ton choix de poste ?

Mon choix s'est fait en comparant les options, il me restait deux entreprises à départager. Mon choix s'est fait sur plusieurs critères, liberté dans le travail, mission à effectuer, salaire, et avantages divers. De plus, je cherche à monter un projet IoT et une de ces entreprises m'a proposé de m'aider dans les démarches. Cette entreprise m'a également proposé de travailler chez moi en l'absence de rendez-vous fournisseurs ou clients. Je suis « obligé » d'aller au travail le lundi, ce qui est vraiment génial, puisque j'ai 4 jours de télétravail !

Quels conseils peux-tu donner ? Ou les trucs à ne pas faire ?

Pour moi le truc à ne pas faire serait de prendre un poste pour lequel on a 100 % des compétences dès le départ, en l'absence de challenge, où est l'intérêt ? Il est très important de vérifier les dires des entreprises qui veulent avant tout vendre leur entreprise, il peut être intéressant de consulter des membres de l'équipe afin de valider que les actes suivent les paroles. En informatique, on a encore le choix des entreprises et elles sont souvent prêtes à faire pas mal de choses pour vous attirer. Une fois entré dans les négociations, il ne faut pas se laisser « avoir » et ne pas hésiter à insister sur les points importants pour vous, selon si vous préférez du temps, de l'argent (beaucoup : -)), ou un équilibre entre les deux. Les environnements technologiques sont aussi très importants, il faut demander plus de précisions que par exemple « Serveur Linux », un serveur Linux peut être un simple Apache ou un OpenLdap un peu plus complexe. Il faut également savoir si les entreprises sont prêtes pour, le Cloud, Docker ou autre. Il faut aussi penser à l'après, si la boîte ferme par exemple, que ferez-vous d'une compétence sur un logiciel de niche utilisé en interne ?

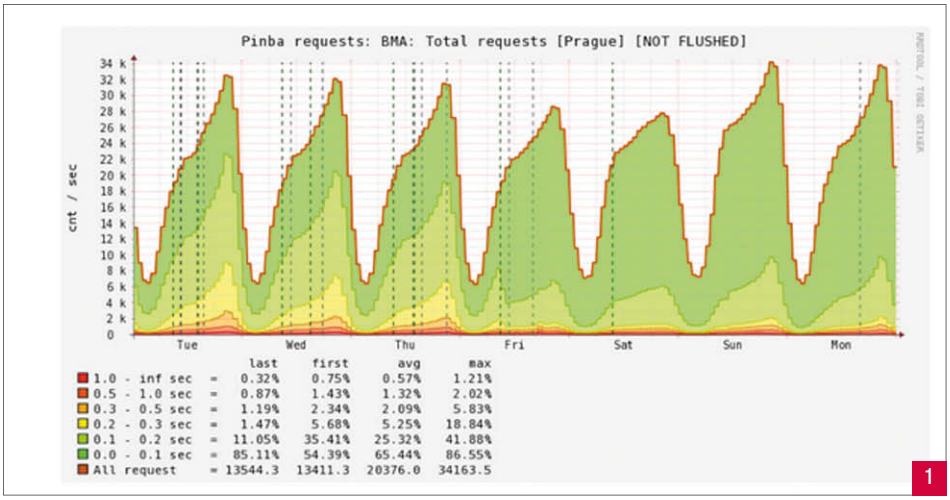
Société	Ville principale	Secteur d'activité	Type société	Effectif actuel	Responsable du recrutement	Nb recrutements prévus 2016-2017	Profils recherchés	Postes ouverts
Openska	Paris	Technologie		11-50	cyril@openska.com	10	Analystes data et développeurs front-end	1 expert Talend, 2 développeurs/formateurs Javascript, 1 expert SGBD (MySQL, MongoDB), 1 consultant BI (ElasticSearch, Tableau, Talend...), 1 expert PHP haut niveau
Webnet	Sèvres	Consulting	SSII	101-250	Cécile Mesguich (cmesguich@webnet.fr)	70	Nous recherchons des spécialistes du Web sur les plateformes actuelles (PHP, Java, .Net) : Architectes techniques, Développeurs BackEnd, Développeurs Full-Stack, Développeurs Front, Lead Developers, Intégrateurs.	Nous cherchons à pourvoir des postes dans nos 3 centres de services (Paris, Lyon, Nantes) et à pourvoir des missions chez nos grands clients (Secteurs Banque Assurance, Energie, E-Commerce, Industrie, Défense, Associations ...)
Smile	Asnières sur Seine	Technologie	SSII	501-1000	S.BAOUZE sabrina.baouze@smile.fr	150	Développeurs Web Php (Symfony, Drupal Magento), Tech Lead Php, développeurs Java EE, Tech Lead Java EE , chefs de projets et directeurs de projets Web, consultants ESB/ETL, consultants Big data, ERP, ingénieurs Système Linux, consultants DevOps, ingénieurs systèmes embarqués, ingénieurs qualité logiciel, spécialistes temps-réels.	
Kaliop	-	Fournisseur de services Internet	PME	101-150	Aurélie Armand recrutement@kaliop.com	30	Dev fullstack / devops / chef de projet / intégrateur / développeur Web / dev magento / dev symfony	http://www.indeed.fr/cmp/Kaliop/jobs
Zenika	Paris	Services	SSII	101-250	Laure Vallade laure.vallade@zenika.com	90	Experts Java EE	Développeurs Java EE et Front End, Experts techniques, DevOps, Coach Agile, Architectes techniques
Osinet	-	Consulting	SSII	-10	osi@osinet.fr	1	Lead développeur front end	
Xebia	Paris	Consulting	SSII	101-250	ljanne@xebia.fr	Recrutement sur profil		http://www.xebia.fr/career.html
Core-Techs	Paris	Technologies	PME	11-50	Mme Olfa HAOUAS-BENOIST / ohaouas@core-techs.fr	10	Développeur Symfony. Développeur Drupal, Lead Technique Symfony, Lead Technique Drupal et chefs de projet techniques	Développeurs Web , avec spécialisation Drupal
Cellenza	Paris	Technologie	SSII	51-100	martina.gagova@cellenza.com	20	Nous cherchons des passionné(e)s de technologies, ouvert(e)s, et généreux (ses) vis-à-vis de la communauté, pour partager une expertise, une vision du métier de développeur mais aussi une philosophie de travail équilibrée ! Nos domaines de compétences : - développement d'applications mobiles - développement d'applications d'entreprise - architectures Cloud - développement Web - gestion du cycle de vie des applications - solutions collaboratives - Big Data	ALM et devOps consultant H/F Consultant Cloud et Integration H/F Craftsman Web H/F Craftsman mobile H/F - Développeur Xamarin H/F Consultant data et analytics Consultant collaboration H/F
Inéat Conseil	Lomme	Services	SSII	101-250	Agathe RITOUET agathe.ritouet@ineat-conseil.fr	30		Développeurs Java, AngularJS, PHP Symfony2, Administrateur système, Ingénieur DevOps, Développeur front end, Développeurs back end
Actency	Paris	Fournisseur de services Internet	PME	11-50	cyrille.imbernon@actency.fr	5	Développeur PHP / Drupal / Symfony	Développeur PHP / Drupal / Symfony

Société	Ville principale	Secteur d'activité	Type société	Effectif actuel	Responsable du recrutement	Nb recrutements prévus 2016-2017	Profils recherchés	Postes ouverts
Actency	Strasbourg	Fournisseur de services Internet	PME	11-50	cyrille.imbernon@actency.fr	5	Développeur PHP / Drupal / Symfony	Développeur / Lead Développeur Drupal
Genymobile	Paris	Logiciels	Editeur	51-100	Béatrice Victor(RH) recrutement@genymobile.com	2-10	Devops, QA, Support, Dev web fullstack, Scrum Master, Développeur C++/Qt	Ingénieur Devops polyvalent de 5 à 10 ans d'expérience, Développeur Web fullstack au moins 4 ans d'expérience
SOAT	Paris	Services	SSII	251-500	Jessica Sberro : jessica.sberro@soat.fr	90	Développeur, Architecte, Craftsman, Consultant MOE/MOA	Développeurs .Net / Java / Web Développeurs iOS et Android Craftsman .Net / Java Développeurs .Net / Java orientés Devops Développeurs.Net / Java orientés Big Data Consultant MOE (.Net / Java / C++) double compétence Finance de Marché Consultant MOA Finance de Marché
Avisto	Vallauris	Logiciels	SSII	101-250	Marion Ruciak marion.ruciak@avisto.com	100	Nous recrutons dans les spécialités suivantes : métiers du développement logiciel (Java, C++, .Net, Web, Big Data, Cloud), de la validation, du système, de l'intégration et de la sécurité. Jeunes diplômés (BAC+5), confirmés et experts.	Développeurs, ingénieurs d'étude, data analysts, architectes, leaders techniques, chefs de projets, responsables d'affaires.
Ikoula	92	Technologies	PME	51-100	Arnaud Noulette - jobs@ikoula.com	6	Support, Commerciaux, Webmaster, ADV, Développeurs, Marketing	Développeurs Web, Techniciens support, Webmaster, etc.
SQLi	93	Technologies	SSII	1001-2500	Frédérique MAGNOL fmagnol@sqli.com	env. 220	.net; JEE, scrum, agile, BI, UX, data analytics, NetWeaver, E marketing, Sharepoint, Java, Front End, Hybris, Magento, Angular...	Architecte, business Analyst, CP, CPT, Consultants (métier, Mktg, Si, UX), DP, Experts techniques, Testeurs, Commerciaux, Développeurs, ICD
Ippon	Paris	Technologies	SSII	251-500	jmcaillaud@ippon.fr	80	Ingénieur Java EE, Solutions Architect, Expert BigData, Consultant Cloud, Expert Devops, Développeur Full-Stack, Consultant-Manager technique	
OVH	Roubaix	Technologies	Fournisseur Internet	1001-2500	jobs-contact@ovh.com	+100	De Bac + 3 à Bac +5, école d'ingénieur ou université. Actif sur les projets IT ou contributeurs, Langage type Perl, Python, Go, C++, Java (Surtout prêts à apprendre de nouveaux langages et donc à l'aise en algorithmie)	Développeurs, DevOps, Architectes logiciel, Datascientists
SoftFluent	Antony	Logiciels	SSII	51-100	cpa@softfluent.com	20	Développeurs .Net, Architecte .Net, Chef de Projet .Net	
Mega International	Paris	Logiciels	Editeur	251-500	Karima Chavalon, kchauvalon@mega.com	17	Développeurs juniors et confirmés, ingénieurs produit juniors/débutants	Développeurs Fullstack, Développeurs .Net/Azure, Développeur Salesforce, Ingénieurs produit Java, Vb/, .Net, Html, Cs, Sql Server
Groupe IT Link	Paris	Technologies	SSII	501-1000	Cécile Chopinet : cchopinet@itlink.fr	120	Diplômés bac +5 école d'ingénieur, d'informatique, Master 2 informatique, ingénieur développement langages C, C++, C embarqué, Java, Php, Python, ingénieur en intégration logicielle, ingénieurs validation logicielle	Ingénieur développement langages C, C++, C embarqué, Java, Php, Python, ingénieur en intégration logicielle, ingénieurs validation logicielle
Avanade France	92	Intégration de systèmes	SSII	501-1000	Antoine Quenet antoine.quenet@avanade.com	150	Tous les candidats ayant des compétences autour des technologies Microsoft : développement d'applications (mobiles, web ou applicatifs métier), base de données/business intelligence, ERP/CRM, infrastructure	Consultant développeur d'applications sur les technologies Microsoft, consultant Business Intelligence, consultant Infrastructure Microsoft

PHP 7 : un an après

Cyril PIERRE de GEYER
Cyril est formateur spécialisé sur PHP pour Openska, l'organisme de formation dédié au Web, à la Data et à l'Opensource.
Auteur de plusieurs livres références sur le sujet (PHP 5 avancé, PHP 7 avancé et performances PHP MySQL) il travaille et apprécie le Web depuis 1999. Il a créé, avec Pascal Martin, pour Openska une formation sur PHP 7 qui permet en deux jours de mettre à jour ses compétences.

Pour ses vingt ans PHP s'était offert une petite cure de jouvence avec une septième version majeure (ou plutôt une sixième du fait de l'abandon de PHP 6). Un an après faisons le point !



1

Request time distribution sur Badoo avant et après PHP 7

Alors bien sûr pour le développeur, PHP 7 n'est pas une révolution dans son quotidien, par rapport au passage de la version 4 à la version 5 ; les impacts quotidiens sont bien moindres. Par contre pour l'administrateur ou du moins pour la charge serveur c'est une évolution majeure ; les nouveaux paradigmes ont été pris en compte et PHP 7 rivalise de vélocité avec les approches optimisées pour les échanges légers (échanges JSON nombreux versus la génération d'une page html complète). Programmez vous invite donc, par l'intermédiaire des trois auteurs du livre « PHP 7 avancé » de faire un état des lieux un an après et d'évoquer les nouveautés de PHP 7.1 qui pointe le bout de son nez !

Pour mémoire les changements par rapport à PHP 5 sont considérables notamment du fait d'une refonte importante du moteur. Au menu des principales nouveautés on peut trouver :

- Nouvelle version du Zend Engine ;
- Gestion des erreurs ;
- Typages scalaires et return type ;
- Différentes modifications destinées à améliorer la cohérence du langage.

Nous allons faire un tour rapide des principales nouveautés de PHP 7, nous évoquerons PHP 7.1 et nous ferons ensuite un focus sur deux migrations : dailymotion et badoo.

RAPPEL SUR LES NOUVEAUTÉS DE PHP 7

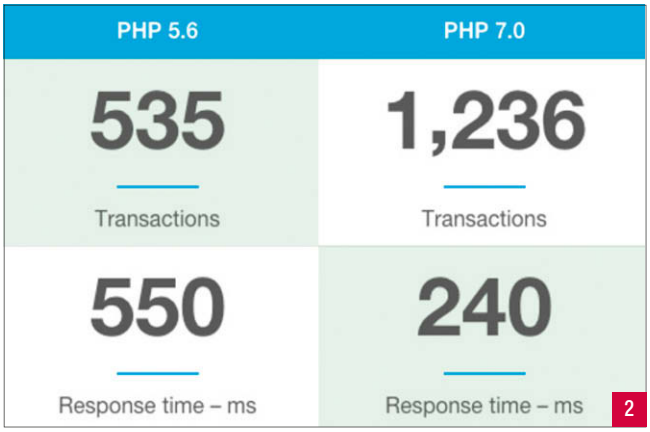
PHP date de 1995, il servait alors uniquement de système de gabarits pour pages Web. La version 3 amène en 1998 un vrai moteur de script tout à fait fonctionnel qui gagne vite une forte communauté.

En 2000 le moteur voit arriver une nouvelle version, PHP 4. Les performances sont améliorées et la modularité permet l'apparition d'extensions pour gérer tout ce qui peut l'être, de la connexion LDAP jusqu'aux interfaces GTK, en passant par la correction orthographique.

La venue de PHP 5 amène de grandes nouveautés dont une première grosse refonte de son cœur, le Zend Engine, qui permet d'implémenter une programmation orientée objet de bon niveau.

Avec PHP 5 on voit vraiment l'industrialisation de PHP, chaque nouvelle version complète une base solide :

- PHP 5.1 renforce l'accès aux bases de données avec PDO ;



2

PerfWP

- PHP 5.2 supporte JSON ;
- PHP 5.3 apporte les namespaces ;
- PHP 5.4 implémente les Traits ;
- PHP 5.5 complète les itérateurs avec les générateurs, tellement plus simples à utiliser.

L'histoire de PHP 6 a été plus compliquée avec notamment des grosses difficultés à implémenter le support complet d'unicode. Longtemps attendue, toujours retardée et enfin abandonnée cette sixième version majeure à finalement été remplacée par PHP 7 pour des raisons de cohérence (beaucoup de choses étaient annoncées sur PHP 6 depuis des années, des livres étaient même sortis sur le sujet…). Dans la pratique une partie des améliorations qui auraient dues être intégrées sur PHP 6 l'ont été sur PHP 5.

Nouvelle version du Zend Engine

Un des apports de PHP 7 que l'on remarque rapidement lorsqu'on l'on migre une application depuis une version précédente est qu'un effort

Abonnez-vous à programmez!

le magazine des développeurs

Nos classiques

1 an — 11 numéros — **49€***

2 ans — 22 numéros — **79€***

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

* Tarifs France métropolitaine

Abonnement numérique

PDF — 1 an - 11 numéros — **30€***

Souscription uniquement sur www.programmez.com

Option : accès aux archives **10€**

Vous souhaitez abonner vos équipes ? Demandez nos tarifs dédiés* : redaction@programmez.com

(* à partir de 5 personnes)

Devenez un Jedi du code avec programmez!

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 201

Offre limitée, valable jusqu'au 2 décembre 2016

☐ M. ☐ Mme Entreprise : Fonction :

Prénom : Nom :

Adresse :

Code postal : Ville :

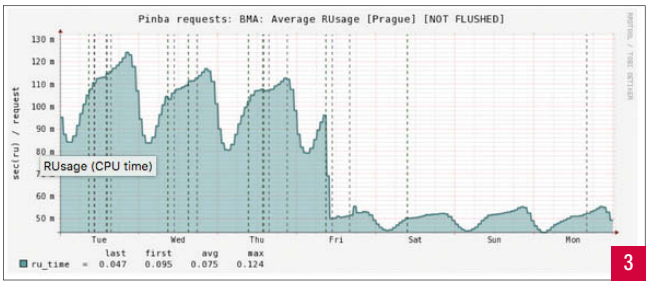
email indispensable pour l'envoi d'informations relatives à votre abonnement

E-mail : @

☐ Je joins mon règlement par chèque à l'ordre de Programmez !

☐ Je souhaite régler à réception de facture

** Tarifs France métropolitaine*



Utilisation des ressources CTP sur Badoo avant et après PHP 7

conséquent a été fait au niveau de l'optimisation et des performances. Selon les applications, on constate en effet une amélioration de celles-ci qui peut aller de 30% jusqu'à 100% ! Dans la figure suivante on voir la différence sur une plateforme WordPress : PHP 7 traite plus de deux fois plus de transactions. Nous verrons que les retours suite aux migrations de dailymotion et Badoo sont impressionnants ! [Fig.1, 2 et 3]

Ce résultat a été atteint en retravaillant en profondeur certaines parties du moteur interne de PHP, qui monte lui-même également de version, on parle désormais du Zend Engine 3.

Ces améliorations au niveau des performances s'accompagnent d'une meilleure gestion multi-threads ainsi que d'un meilleur support des plateformes 64 bits. Pour la petite histoire deux moteurs auraient pu accompagner PHP 7 : HHVM créé par FaceBook et le Zend Engine soutenu par le partenaire historique, Zend. Le PHP Group à opté par vote, de façon massive, pour le ZE3.

Gestion des erreurs

En PHP 5, une partie des problèmes étaient interceptables par le mécanisme des exceptions et une autre partie ne l'était pas. Une des principales problématiques était liée à la tendance de PHP à lever des erreurs (particulièrement dans le cas d'erreurs fatales) qui mettaient fin a l'exécution du script. Dans certains cas ou il est nécessaire que le script s'exécute longtemps (comme par exemple quand on code un démon) cela posait problème.

PHP 7 a fait évoluer sa gestion des erreurs : une grande partie des erreurs fatales de PHP ont été converties en exceptions, il est donc possible de les intercepter et de les traiter.

Les exceptions PHP 7

Côté PHP 7, les exceptions sont des objets implémentant l'interface *Throwable* et dérivant de la classe *Exception* lorsqu'elle est utilisée par le programmeur, ou de la classe *Error* lorsqu'elle est générée par PHP. Une erreur fatale de PHP, est en réalité une exception. La classe *Exception* et la classe *Error* implémentant toutes deux l'interface *Throwable,* vous pouvez tirer parti de ces avantages pour effectuer un filtrage efficace à la fois des exceptions utilisateur, mais aussi des exceptions PHP.

Evolution du typage

PHP est connu pour être un langage faiblement typé (ou dynamiquement typé). PHP 7 permet aux développeurs d'imposer ou non du typage sur certains éléments du langage.

Définir le type des fonctions et méthodes

PHP 7 élargit le spectre défini pour PHP 5 : vous avez désormais la possibilité de définir des typehints scalaires.

Il devient, ainsi, possible de définir une fonction en spécifiant dès son écriture qu'elle attend en paramètres deux nombres entiers :

```
<?php
function add(int $a, int $b) {
return $a + $b;
}
var_dump( add(10, 20) ); // int(30)
```

Si vous tentez de passer en paramètre une donnée qui ne peut être convertie dans le type demandé alors une exception de type *TypeError* sera levée :

```
try {
var_dump( add(10, 'abc') );
}
catch (TypeError $e) {
// string(148) "Argument 2 passed to add() must be of the type integer, string given,
called in .../test01.php on line 12"
var_dump( $e>getMessage() ); }
```

Définir le type de la valeur de retour des fonctions et méthodes

Il est possible de définir également le type de retour d'une fonction. Par exemple, ici, pour une fonction censée retourner un nombre flottant (l'exemple échouera donc) :

```
<?php
function test() : float {
return 'bonjour';
}
?>
```

Cette fonction retourne une chaîne de caractères alors qu'elle aurait dû renvoyer un flottant. PHP lève alors une erreur.

```
TypeError: Return value of test() must be of the type float, string returned
```

Gérer la souplesse de PHP

Par défaut PHP tentera de convertir les données qui lui ont été transmises pour respecter le typage demandé. Si vous ne le souhaitez pas, que vous voulez que PHP travaille en mode de typage plus strict, une nouvelle option est disponible pour l'instruction *declare()* :

```
<?php
declare(strict_types=1);
function add(int $a, int $b) {
return $a + $b;
}
add(10, '20');
```

Autres améliorations de PHP 7

PHP 7 a été l'occasion de travailler à l'amélioration de la cohérence générale. Plusieurs modifications attendues ont été intégrées :

- PHP 7 ne supporte plus les balises d'ouvertures alternatives telles que <% ou <%= ;
- PHP 7 ne permet plus d'avoir plusieurs cas default dans un switch ;
- Imbrication des « use » pour simplifier l'appel aux namespaces ;
- Utiliser des constructeurs de classe à la façon PHP 4, ou le constructeur est une méthode de même nom que la classe, lèvera désormais un

avertissement indiquant qu'ils sont obsolètes ;
- Le niveau d'avertissements *E_STRICT* a été supprime et les levées d'erreurs correspondantes ont été reclassifiées vers d'autres niveaux, comme *E_DEPRECATED* ou *E_WARNING*.

PHP 7 comporte encore beaucoup d'autres modifications plus légères mais qui permettent de franchir une nouvelle étape dans la professionnalisation de cette technologie. Les articles suivants complèteront cet aperçu qui est une toute petite partie de ce que nous avons voulu mettre dans « PHP 7 avancé », la nouvelle mouture du livre qui, dans sa version dédiée à PHP 5, à permis à plusieurs dizaines de milliers de développeurs de se perfectionner.

PHP 7, L'EXEMPLE DE DAILYMOTION

Cette partie reprend l'article de Thomas Colomb, ingénieur chez DailyMotion détaillant la migration de leur plateforme : https://medium.com/@colomb.thomas/php-7-deployment-at-dailymotion-a3066283d2aa#.hk3i9vsxi

En mars 2015 les équipes techniques de Dailymotion ont commencé à investiguer d'autres voies que l'optimisation du code et de l'architecture pour améliorer les temps de réponse du site. Comme tous les sites à fort trafic ils se sont posé la question : « comment scaler sans investir trop en ressources humaines et matériel ? »

Premiers essais avec HHVM

La première tentative a été réalisée avec HHVM, le moteur PHP créé par FaceBook. A quelques modifications de code près le site fonctionnait et le résultat était épatant : le temps de réponse des pages était divisé par deux, pareil pour la consommation CPU & RAM !

Malheureusement il y avait quand même des erreurs, voici les deux raisons qui ont poussé les équipes à désactiver HHVM :
- Il y a beaucoup de code incompatible entre HHVM et PHP, certaines fonctions ne renvoient pas les mêmes résultats, renvoient des Warning supplémentaires…
- Dailymotion utilise des extensions qui ne sont pas compatibles avec HHVM.

Sara Golemon, une personnalité active du PHPGroup, a écrit un guide pour développer des extensions HHVM, fort de cette base les équipes de DailyMotion ont développé leurs propres extensions. Après d'autres problèmes intermédiaires les résultats étaient au rendez-vous mais des questions subsistaient : est ce qu'il y a encore des incompatibilités cachées, sera-t-il possible de revenir vers l'interpréteur PHP si ils ne veulent plus d'HHVM, comment former les nouveaux développeurs aux spécificités de HHVM (grosse perte de temps en perspective) …

Finalement le sujet a été mis en stand by jusqu'à l'arrivée de PHP 7.

La migration vers PHP 7

En décembre 2015 PHP 7 a été déployé sur des grosses plateformes telles qu'Etsy et Badoo, et DailyMotion a également franchi le pas en installant PHP 7 sur deux serveurs. Les résultats ont été similaires à HHVM en terme de performances et surtout les extensions utilisées par les équipes (pinba, php-redis…) étaient déjà compatibles.

La société Etsy a développé un analyseur statique de code pour trouver les incompatibilités entre PHP 5 et PHP 7 : phan. Avec cette aide, le code de DailyMotion, vieux de dix ans et monolithique, a été modifié en moins d'une semaine. Et encore, la plupart des erreurs étaient des warnings sur E_STRICT qui n'avaient pas été fixés depuis un bon moment.

Les équipes de DailyMotion ont donc choisi PHP 7 parce que c'était plus facile à déployer, qu'il n'y avait pas de régressions à part quelques incompatibilités listées sur le manuel PHP et que son support serait à priori plus pérenne ; le tout pour des gains de performances identiques (temps de réponse, CPU, mémoire…).

•

Les nouveautés de PHP 7.1

Un an après l'arrivée de PHP 7.0, voici une nouvelle version mineure pour le langage leader du Web.
Je vous propose de découvrir le lot d'évolutions qui nous sont offertes par cette mise à jour.

Pascal MARTIN
Passionné de développement en général ainsi que de Web et de PHP en particulier, Pascal Martin travaille aujourd'hui à Lyon chez TEA sur une plateforme de diffusion de livres numériques.
Il publie régulièrement, notamment des articles techniques, sur son blog et il est auteur du livre «Développer une Extension PHP» et coauteur de «PHP 7 avancé».

À propos de « PHP 7 avancé »

Julien Pauli, Cyril Pierre de Geyer et moi-même avons travaillé cette année, en nous basant en partie sur l'ouvrage «PHP 5 avancé» co-écrit par Éric Daspet et Cyril, à l'écriture d'un livre présentant PHP 7. Le fruit de nos efforts, **«PHP 7 avancé»**, a été publié mi-octobre, aux éditions Eyrolles. Vous pouvez dès maintenant le trouver chez votre libraire habituel.

Le 3 décembre 2015, il y a bientôt un an et après des mois de travail intensif, l'équipe de développement de PHP publiait la version 7.0, la première version mineure de la nouvelle branche «PHP 7».

La presse, dont *Programmez!*, en a largement parlé : PHP 7 apportait de nombreuses nouveautés et améliorait considérablement les performances du moteur. Et ces nombreuses évolutions n'étaient accompagnées que d'un nombre réduit et à l'impact limité de cassures de compatibilité antérieure, en vue de faciliter la montée de version depuis PHP 5 et de ne pas reproduire le schéma infernal de la migration de PHP 4 vers PHP 5. Un an après, c'est au tour de PHP 7.1 d'apporter un nouveau lot d'évolutions.

Un rappel sur le cycle de versions de PHP

Depuis PHP 5.4, les nouvelles versions de PHP suivent un rythme de publication où une version mineure est prévue par an. Il vaut en effet mieux sortir de nouvelles versions plus souvent, mais que leur coût de migration soit réduit, plutôt qu'une mise à jour *bigbang* toutes les quelques années !

Chaque version mineure de PHP est supportée pendant deux années pendant lesquelles elle reçoit des corrections de bugs, plus une année supplémentaire pour les correctifs de sécurité. En suivant ce cycle de versions, au dernier trimestre 2016, soit un an après la sortie de PHP 7.0, il est temps de voir apparaître une nouvelle version mineure : PHP 7.1 !
Cette version mineure, pour respecter le processus défini, n'est censée casser la compatibilité antérieure qu'au strict minimum, pour ne pas complexifier inutilement la montée de version ni réduire son adoption.

Le typage : de nouvelles évolutions

J'en parlais en détails dans le numéro 195 de Programmez! : PHP 7.0 a apporté des évolutions considérables au mécanisme de typage. Nous pouvons spécifier des déclarations de types scalaires, éventuellement strictes, pour les paramètres de fonctions et leur valeur de retour.
Avec PHP 7.1, de nouvelles améliorations sont apportées à ce principe, en particulier en le complétant, en réponse aux demandes de nombreux développeurs.

Les types *nullables*

Un des reproches le plus souvent entendu à propos du mécanisme de déclarations de types de PHP 7.0 est qu'il n'était pas possible de déclarer un paramètre comme pouvant être NULL sans pour autant le rendre optionnel. Or, NULL est souvent employé pour indiquer qu'une valeur n'est pas définie. En réponse, PHP 7.1 introduit l'idée de types *nullables* : positionner un symbole ? au début d'un nom de type, comme « ?int », signifie que la valeur spéciale NULL est également acceptée :

```
function null_accepte(?int $val) {
    var_dump($val);
}

null_accepte(100); // int(100)
null_accepte(null); // NULL
```

Cette notion s'étend également à la valeur de retour d'une fonction ou méthode, où elle sera d'ailleurs probablement le plus souvent employée. On peut par exemple penser à une méthode qui chargerait une information depuis une base de données et retournerait soit un objet, soit NULL si la donnée n'est pas trouvée :

```
class User {}

class UserRepository {
    public function getById(int $id) : ?User {
        // Ici, nous pouvons retourner une instance de la classe User, ou NULL
    }
}

$user = (new UserRepository())->getById(42);
```

Avec PHP 7.0, cette écriture n'était pas possible : la méthode n'aurait pas pu retourner NULL — ou il nous aurait fallu nous passer de la déclaration du type de retour !

Type de retour void

PHP 7.1 permet également d'indiquer, explicitement, qu'une fonction ou méthode ne doit retourner aucune valeur : nous devons pour cela utiliser le pseudo-type de retour void :

```
function ne_retourne_rien() : void {
    // Cette fonction ne retourne rien
}
```

Si nous essayons d'écrire le mot-clé return suivi d'une valeur dans cette fonction, une erreur sera levée par PHP lors de la compilation de notre script : « Fatal error: A void function must not return a value ». Il est toujours possible, cela dit, d'utiliser le mot-clé return, seul, pour quitter la fonction : aucune valeur n'étant spécifiée, PHP continue d'accepter cette écriture.

Pseudo-type iterable

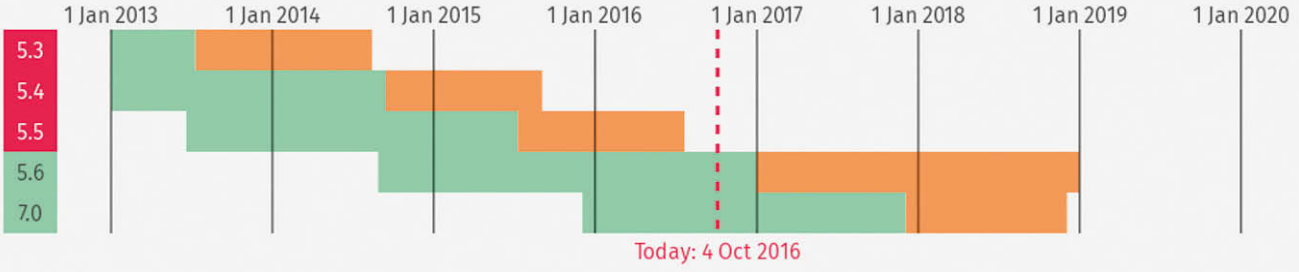
PHP 7.1 apporte une troisième évolution au mécanisme de déclarations de types : un autre nouveau pseudo-type, nommé iterable. Il regroupe tout ce sur quoi nous pouvons itérer : tableaux, données traversables, itérateurs, générateurs…

```
function accepte_iterable(iterable $data) {
    // $data est une donnée sur laquelle on peut boucler
    foreach ($data as $key => $val) {
        // ...
    }
}
```

L'intérêt de ce pseudo-type iterable est le suivant : cette fonction peut être appelée en lui passant un tableau, un objet implémentant l'interface Iterator, ou un générateur — trois données sur lesquelles PHP sait boucler :

```
// Avec un tableau
```

LES VERSIONS DE PHP ACTUELLEMENT SUPPORTÉES



La page php.net/supported-versions.php vous donnera, à tout instant, les versions de PHP actuellement supportées. La fin du support actif approche pour PHP 5.6, la dernière version de PHP 5 — et il devient réellement temps de penser à migrer vers PHP 7, même si le support pour les éventuels problèmes de sécurité a été prolongé pour cette version. Pour autant, w3techs.com rapporte que PHP 7 ne représente aujourd'hui que 1,5 % des installations actives de PHP, contre 97,4 % de PHP 5. À chacun d'entre nous de faire évoluer ces chiffres !

```
accepte_iterable([123, 456, 789]);

// Avec un itérateur
accepte_iterable(new SplFixedArray(13));

// Ou avec un générateur
function mon_generateur() {
   yield 123;
   yield 456;
   yield 789;
}
accepte_iterable(mon_generateur());
```

Comme pour les autres déclarations de types, passer à cette fonction une donnée incorrecte entraînera la levée d'une exception TypeError.

Un langage et une syntaxe plus cohérents

Une version mineure de PHP est toujours l'occasion d'améliorer la cohérence du langage et de faciliter son utilisation. Cela passe souvent par de nouvelles fonctions utiles, une orientation plus stricte du langage, ou des retouches apportées à certains points de syntaxe.

Créer une Closure à partir d'un *appelable*

Les fonctions de rappel de PHP, souvent désignées par leur nom sous forme d'une chaîne de caractères, souffrent d'un problème majeur : en cas d'erreur sur ce nom, ce n'est que lors d'une tentative d'appel qu'un problème sera détecté :

```
function ma_fonction($param)
{
   // ...
}

// Pas de problème détecté ici, alors qu'une erreur s'est glissée dans le nom de la fonction
$nom = 'ma_fonctIon';

// C'est seulement ici, lors de l'appel, qu'on identifie le problème :
// Warning: call_user_func() expects parameter 1 to be a valid callback, function
'ma_fonctIon' not found or invalid function name
call_user_func($nom, 42);
```

PHP 7.1 introduit une nouvelle méthode Closure::fromCallable(), dont le but est de permettre la détection au plus tôt de ce type de problème. Elle prend en paramètre un *appelable*, comme un nom de fonction, et retourne une fonction anonyme qui pourra être appelée directement par la suite. Si l'*appelable* n'est pas valide, une exception TypeError sera levée par fromCallable(), sans que nous n'ayons pour cela à attendre l'appel :

```
// L'erreur est immédiatement détectée
// Uncaught TypeError: Failed to create closure from callable: function 'ma_fonctIon'
not found or invalid function name
$callable = Closure::fromCallable('ma_fonctIon');
```

Cette nouvelle méthode fonctionne bien sûr avec les autres syntaxes d'*appelables*, comme [$objet, 'nomMethode'].

Spécifier les clés à extraire, avec list()

Pour ce qui est des manipulations de tableaux, nous pouvons désormais spécifier des clés lorsque nous travaillons avec l'instruction list() pour extraire des données :

```
$array = ['glop' => "Hello", 'plop' => 123456, 'who' => "World"];

list ('glop' => $a, 'who' => $b) = $array;
```

Cette écriture permet de cibler plus finement les enregistrements à extraire d'un tableau, tout en menant à du code plus explicite.

Syntaxe courte pour les affectations en dé-construisant un tableau

PHP 5.4 a introduit, il y a plusieurs années maintenant, une syntaxe « courte » pour les déclarations de tableaux. PHP 7.1 améliore la cohérence du langage, en permettant l'emploi de cette syntaxe pour les déconstructions de tableaux, là où nous devions auparavant systématiquement utiliser list() :

```
$array = [10, 20, 30];
[$a, $b, $c] = $array;
```

Cette syntaxe est bien sûr compatible avec celle vue au-dessus et nous pouvons préciser les clés des éléments spécifiques à extraire.

Généralisation du support des indices négatifs pour les chaînes de caractères

Pour cette nouveauté, même si elle est mineure, j'ai envie de dire « enfin » ! Avec PHP 7.1, nous pouvons enfin utiliser des indices négatifs lorsque nous manipulons des chaînes de caractères. En particulier, il est désormais possible de désigner un caractère donné en comptant à partir de la fin d'une chaîne, en lecture comme en écriture :

```
$str = "Pascal";
var_dump($str[-2]); // string(1) "a"

$str = "Pas.al";
$str[-3] = 'c';
var_dump($str);  // string(6) "Pascal"
```

Les fonctions de manipulation de chaînes qui n'acceptaient pas ce principe, comme strpos(), ont également été adaptées en conséquence :

```
var_dump( strpos("Pascal", "c", -5) ); // int(3)
```

Ici encore, PHP 7.1 mène à une syntaxe plus logique ; sans compter que l'absence de cette fonctionnalité surprenait souvent les développeurs venant d'autres langages.

Visibilité des constantes de classes

Je termine cette section sur les améliorations mineures de syntaxe avec une nouveauté qui répond à un souhait que j'ai exprimé moult fois. Avec PHP 7.1, nous pouvons spécifier une visibilité sur les constantes que nous définissons dans des classes :

```
class MaClasse {
   public const MA_PUBLIQUE = 42;
   private const MA_PRIVEE = 1234;

   public function test() {
      var_dump( self::MA_PRIVEE );
   }
}
```

Les mot-clés déjà employés pour les attributs et méthodes peuvent enfin être utilisés pour des constantes ! Nous n'avons donc plus besoin de pas-

ser par des *astuces* comme des variables statiques privées — et pas constantes — pour définir des données qui ne seraient pas visibles en dehors d'une classe !

Et quelques améliorations mineures…

Au-delà des améliorations majeures et des évolutions dont j'ai parlé jusqu'ici, PHP 7.1 apporte encore quelques changements, probablement moins importants, mais qui correspondent assez bien à l'orientation qu'a pris le langage ces dernières années : un objectif toujours plus professionnel et une vision plus stricte, sans pour autant casser la compatibilité avec l'existant. Ainsi, utiliser une représentation octale invalide pour un caractère dans une chaîne de caractères, comme "\400", lève désormais un avertissement. Un autre exemple serait celui du traitement des signaux avec l'extension pcntl : leur gestion est importante dans le cadre de programmes lancés en ligne de commande et destinés à fonctionner longtemps, comme des démons ou des workers. PHP 7.1 permet, par le biais d'une nouvelle fonction pcntl_async_signals(), un traitement asynchrone des signaux système, sans avoir à passer par declare(ticks=1) à l'impact négatif sur les performances. Pour un dernier exemple : les nombres flottants sérialisés en JSON le seront désormais avec une précision plus grande, réduisant ainsi le risque de perte d'information lors d'échanges entre systèmes.

La gestion d'erreurs

Ces dernières années, difficile de tester une nouvelle version de PHP sans découvrir quelques améliorations autour de la gestion d'erreurs — et c'est tant mieux !

Des extensions lèvent des exceptions de type Error

PHP 7.0 a introduit l'interface Throwable et les exceptions de type Error et a remplacé certaines erreurs fatales levées par le moteur en exceptions de ce type. PHP 7.1 poursuit le travail entamé à ce niveau, en convertissant cette fois-ci en Error quelques erreurs fatales levées par des extensions packagées avec PHP.

PHP, un moteur et des extensions

De nombreuses fonctionnalités (classes, méthodes, comportements) de PHP sont fournies par des « extensions ». Certaines de ces extensions sont développées par la même équipe que PHP lui-même et leur code source est localisé au sein du projet PHP. Par exemple : session, standard, fileinfo, pdo, sqlite3, zip… Par exemple, désérialiser une chaîne invalide vers une DateTime ne lève maintenant plus une Fatal Error :

```
// 'timezone-type' au lieu de 'timezone_type'
$serialized = '0:8:"DateTime":3:{s:4:"date";s:26:"2016-08-14 12:31:50.000000";s
:13:"timezone-type";i:3;s:8:"timezone";s:3:"UTC";}';
$dt = unserialize($serialized);
// Uncaught Error: Invalid serialization data for DateTime object
```

L'intérêt est majeur : là où une erreur fatale stoppait l'exécution de votre programme, une Error est une exception, que vous pouvez gérer avec un bloc try/catch.

Attraper plusieurs types d'exceptions avec un seul catch

La seconde amélioration apportée par PHP 7.1 autour de la gestion d'erreurs devrait vous intéresser si vous manipulez de nombreux types d'exceptions en souhaitant les gérer de la même manière, sans qu'ils n'aient pour autant une interface commune. En effet, vous pouvez désormais *attraper* plusieurs types d'exceptions dans un seul bloc catch, en séparant ces types par un | :

```
try {
    // ...
}
catch (MonException | AutreException $e) {
    // Prise en compte de deux types d'exceptions
}
catch (EncoreAutreException $e) {
    // Et ici, d'un seul
}
```

Le principe de gestion d'exceptions et le fonctionnement de try/catch et finally ne sont nullement altérés : il ne s'agit ici que d'une modification de syntaxe, qui vise uniquement à vous simplifier la vie.

Mais quelques bc-breaks ?

PHP vise toujours à faciliter l'application de mises à jour d'une version à la suivante, en veillant à réduire au strict minimum les changements de comportement qui pourraient casser nos applications.
Lors de la bascule de PHP 7.0 vers PHP 7.1, vous devrez toutefois veiller à quelques points qui pourraient peut-être vous complexifier la tâche (ou vous aider à détecter quelques bugs !).

Un manuel et des guides de migration

Le manuel de PHP contient une section « guide de migration » pour chaque version de PHP, qui reprend l'ensemble de nouveautés et des changements correspondants. Par exemple, pour la migration depuis PHP 7.0 vers PHP 7.1, vous ne manquerez pas de parcourir php.net/migration71

Utilisation de chaînes non numériques dans des calculs

PHP suit historiquement une approche de typage souple avec des conversions appliquées automatiquement lorsqu'elles sont nécessaires. Cette logique est aujourd'hui considérée par certains comme surprenante, en particulier dans des cas *étranges*.
En conséquence, avec PHP 7.1, effectuer des calculs arithmétiques sur des chaînes de caractères convertissables en entiers, mais ne contenant pas uniquement un entier, mènera à une notice. Par exemple :

```
echo 12 + "5 pommes";
// Notice: A non well formed numeric value encountered
```

La conversion et le calcul seront effectués, comme avec les versions précédentes de PHP : cette ligne entraîne ainsi toujours l'affichage de l'entier 17. En allant plus loin, dans le cas où la conversion de la chaîne de caractères ne mène pas à un entier, c'est un avertissement qui sera levé, le cas étant plus à même d'être causé par — ou de mener à — une erreur :

```
echo 12 + "pommes";
// Warning: A non-numeric value encountered
```

Ce changement devrait principalement vous aider à détecter des cas *surprenants* dans votre code et, donc, à corriger quelques bugs potentiels — mais il peut vous surprendre lorsque vous changerez pour la première fois de version de PHP sur votre environnement de développement !

Remplacement de l'avertissement « Missing argument » par une exception « Too few arguments »

Si vous appelez une fonction en lui passant moins de paramètres qu'elle n'en attend, historiquement, PHP vous prévenait avec un avertissement de niveau E_WARNING. En effet, le comportement dans ce type de situation est rarement prévu, puisque le paramètre que vous vous attendiez à voir rempli reçoit en

fait NULL. Avec PHP 7.1, appeler une fonction en lui passant moins de paramètres que prévu entraînera la levée d'une exception de type Error :

```
function ma_function($a, $b)
{
   // ...
}

ma_function(10);
// Uncaught Error: Too few arguments to function ma_function(), 1 passed
```

Si vous aviez pris l'habitude d'ignorer ce type d'avertissement, ce changement de comportement aura un impact majeur sur votre application !

Et d'autres points ?

Les fonctions accédant, en lecture ou en écriture, aux variables de la portée de leur appelant avaient un comportement indéfini lorsqu'elles étaient appelées dynamiquement. Je pense notamment à extract(), compact() ou parse_str() : appeler une de ces fonctions avec call_user_func() ou la syntaxe $fonction() pouvait mener à des résultats étranges.

Par conséquent, PHP 7.1 lève désormais un avertissement si vous effectuez un appel de ce type à une de ces fonctions : « Warning: Cannot call extract() dynamically » . Une autre incohérence a été corrigée autour de la variable spéciale $this : avec les versions précédentes de PHP, il était en effet possible de modifier sa valeur, en trichant avec des variables-variables ! Ceci n'est plus possible avec PHP 7.1 : $this a sa valeur et tenter de la modifier mènera systématiquement à une Error indiquant « Cannot re-assign $this ». Enfin, n'oublions pas que chaque nouveau mot-clé ajouté à PHP peut se traduire par du code qui cessera de fonctionner au niveau de votre application. Pensez par exemple à vérifier que vous n'employez pas de classe nommée void ou iterable, qui mènerait à une erreur fatale indiquant « Cannot use 'void' as class name as it is reserved ».

Utiliser PHP 7.1 dès maintenant

Au moment où j'écris ces lignes, PHP 7.1 RC3 a été publiée il y a quelques jours. La publication de la version stable 7.1.0 approche donc à grands pas : elle sera peut-être même disponible lorsque vous lirez ceci ! Pour autant, PHP 7.1 ne sera probablement pas disponible par défaut tout de suite pour votre distribution — mais cela ne vous empêche pas de commencer à l'utiliser.

Des dépôts fournissent PHP 7.1

Même si PHP 7.1 n'est pas nécessairement disponible sur les dépôts officiels de votre distribution, plusieurs mainteneurs de confiance ont déjà packagé cette version. Si vous êtes sous Fedora / RHEL / CentOS, vous pouvez trouver PHP 7.1 sur le dépôt de Remi Collet. Pour Ubuntu, vous vous tournerez plutôt vers le dépôt PPA d'Ond ej Surý.

PHP 7.1 et Docker

Une autre possibilité est d'utiliser une image Docker comme environnement d'exécution, sans toucher quoi que ce soit à votre installation locale. Justement, l'image officielle fournie par Docker via son Hub intègre déjà plusieurs tags correspondant aux différentes préversions qui ont été publiées pour PHP 7.1 : hub.docker.com/_/php/

Ou compiler PHP à la main

Enfin, vous pouvez toujours compiler PHP *à la main*, vous-même : vous pourrez ainsi travailler avec une version répondant au mieux à vos besoins, ou même tester une version qui n'a pas encore été publiée ou qui ne sera pas rendue disponible pour votre distribution.

Compiler PHP, une fois les sources obtenues, passent par la succession des quelques étapes requises par de nombreux logiciels open source :

```
./buildconf
./configure --prefix=$HOME/bin/php-7.1 --disable-all --enable-pcntl --enable-intl
 --enable-mbstring
Make && make install
```

En fonction des extensions que vous souhaitez construire, vous passerez d'autres options complémentaires au script de configuration ; reportez-vous à la sortie de la commande « ./configure --help » pour la liste des options qui vous sont offertes.

La route vers PHP 7.2 et PHP 8.0

PHP 8.0 n'est qu'une idée très lointaine pour l'instant et il est peu probable qu'une nouvelle version majeure soit publiée avant plusieurs années. Toutefois, plusieurs changements commencent dès maintenant à viser cette future version. En effet, les versions mineures comme PHP 7.1 peuvent marquer des fonctionnalités comme *obsolètes*, en vue de leur suppression sur une prochaine version majeure. À partir de PHP 7.1, deux fonctionnalités entrainent désormais la levée d'un avertissement de niveau E_DEPRECATED. Il est donc probable qu'elles seront retirées de PHP dans le futur.

Tout d'abord, vous ne devriez plus employer le modificateur /e des fonctions mb_ereg_replace() et mb_eregi_replace(), mais plutôt passer par mb_ereg_replace_callback(). Ces fonctions s'alignent ainsi petit à petit avec preg_replace(), qui ne supporte plus ce modificateur depuis PHP 7.0, en raison des risques qu'il faisait courir en termes de sécurité.

La seconde fonctionnalité qui est désormais marquée comme obsolète, alors que son emploi était déjà déconseillé depuis longtemps, est l'extension mcrypt et les différentes fonctions qu'elle expose. Elle est en effet basée sur une bibliothèque qui n'est plus maintenue depuis des années — ce qui est plutôt risqué pour une extension liée à la sécurité !

Avant de penser à une version majeure PHP 8.0, il est probable de voir apparaître, dans un an, une nouvelle version mineure : PHP 7.2. Plusieurs évolutions ont d'ores et déjà été discutées pour cette version (cela ne signifie pas forcément que toutes seront acceptées !), comme la levée d'un avertissement en cas de tentative d'accès par le biais d'une syntaxe de tableau à une donnée qui n'est pas un tableau, comme l'ajout d'un nouvel opérateur ??=, ou un nouvel algorithme de hachage pour password_hash(). Les prochains mois vont être riches en propositions visant PHP 7.2, je n'en doute pas !

EN CONCLUSION ?

Un an après PHP 7.0, version majeure qui a apporté des évolutions conséquentes ainsi qu'un gain allant de 30 % à 100 % sur les performances d'applications réelles, c'est au tour de PHP 7.1 d'être publiée en cette fin d'année. Cette version apporte quelques évolutions et améliore la cohérence de certains points de syntaxe, sans pour autant révolutionner le langage ou son utilisation.

La montée de version depuis PHP 5.x vers PHP 7.0 était assez simple, bien qu'il s'agisse d'un changement de version majeure, pour peu que le code de nos applications soit bien écrit.

Le passage de PHP 7.0 à 7.1 devrait presque n'être qu'une formalité, même si quelques changements de comportement sont à prendre en compte. •

Retours d'expérience après un an d'utilisation de Drupal 8

• Par l'équipe de **Core-Techs**

Voilà maintenant 1 an que Drupal 8 est sorti, et presque autant de temps que nous utilisons cette solution dans la majeure partie de nos projets de création ou de refonte.

L'enthousiasme débordant des premiers jours - où frénétiquement nous allions régulièrement sur https://drupalreleasedate.com/ parier sur une hypothétique date de sortie (ne le niez pas ! vous avez fait pareil) a eu raison de notre patience : Drupal était bien là … pour le meilleur mais aussi pour le pire ! Notre engouement - sans doute démesuré - a vite été rattrapé par la réalité du produit. Le CMS était techniquement prêt mais pas son écosystème.
L'absence de nombreux modules (y compris de certains modules phares) et le manque d'expérience général de la communauté pour jauger au mieux de la solution, ont été les premiers écueils que nous avons rencontrés, auxquels des obstacles insoupçonnés se sont vite greffés comme l'intégration des nouveaux écrans FO / BO au sein de nos différents supports, la réécriture parfois partielle, souvent complète de nos différents documents et livrables (réponses à appel d'offres, spécifications techniques et fonctionnelles, documents de formations…). Nos premiers projets n'allaient pas être de tout repos !... oh que non ! Au travers de nos différents corps de métiers regroupant des profils consultants, développeurs, Chef de projet, lead Tech, nous avons souhaité cosigner cet article sans langue de bois, résumant nos différents retours d'expériences sur la solution Drupal 8 depuis la beta finale jusqu'aux versions plus récentes.

Des débuts laborieux…

« Nous avons eu l'opportunité de démarrer très rapidement sur Drupal 8, c'est-à-dire au moment où les premières versions RC étaient disponibles (octobre 2015) sur un projet simple et très orienté éditorial. Aucune fonctionnalité complexe n'était prévue dans la V1 du projet, le risque était donc assez limité » se souvient Abdenour, Lead Technique chez Core-Techs.
« C'était une manière pour nous d'opérer tranquillement ce transfert et de bénéficier d'une première expérience concrète » renchérit Cyrielle, Chef de projet. Il était cependant nécessaire que ce projet soit porté par la direction, par l'équipe projet et surtout par la cliente. *« Nous avons présenté à la cliente l'ensemble des fonctionnalités de son cahier des charges qui étaient couvertes par Drupal 7 mais pas forcément par Drupal 8. La première question était de savoir qu'elles étaient les fonctionnalités que nous pouvions éventuellement temporiser. Dans le cas où un développement spécifique était nécessaire, il fallait jauger de sa répercussion sur le planning sachant que le site avait un impératif de mise en production assez court ».* La cliente, en toute connaissance de cause, a finalement opté pour Drupal 8. L'objectif de son site correspondait à des appels à projets sur des cycles de 3 ans qui s'étendent au-delà de 2020. C'était plus intéressant pour tout le monde de partir sur un Drupal 8 et d'éviter ainsi une migration qui aurait été obligatoire d'ici quelques années.
« Nous avons démarré les premiers développements sur la RC1, puis nous avons fait évoluer au fur et à mesure des releases candidate jusqu'à la version finale 8.0.0 qui est sortie à temps » explique Gérard, l'un des développeurs du projet. *« Nous savions que les montées de version entre les « RC » se faisaient de manière fluide et beaucoup moins chaotique que les upgrades entre les différentes versions « alpha ».*
Sans être chaotique donc, la mise en place de ce premier site a réservé quelques mauvaises surprises à nos équipes, notamment au niveau des développements : *« La V1 du projet était simple. Elle se résumait à la mise en place de quelques types de contenus, un carrousel, une médiathèque, un éditeur WYSIWYG, plusieurs formulaires de contact, une carte Google Maps et un peu de référencement. Nous nous doutions que certains modules n'étaient pas pleinement opérationnels et que nous devions avoir recours à un peu de développement custom, mais nous ne pensions pas que c'était aussi important que cela. Il manquait de nombreuses choses, qui, mises bout à bout, commençaient à peser en termes de charge ».*

Un impact pour les développeurs, les équipes formation et avant-vente !

Ce changement ne s'est pas seulement fait sentir au niveau des développements. Drupal 8 a modifié les habitudes des uns et des autres. L'indisponibilité de certains modules phares que nous avions l'habitude de proposer pour nos projets a eu une incidence sur la gestion de projet et certains livrables (spécifications fonctionnelles, document de formation en autre).
Comme l'explique Cyrielle : *« Ces changements ont induit quand même de devoir réécrire une partie de nos spécifications, notamment sur des fonctionnalités qui n'étaient pas prises en compte sur Drupal 8. Mais, à l'inverse, nous nous y sommes retrouvés car il y avait aussi des fonctionnalités non natives avant (cf. les liens internes, l'édition rapide en Front-Office …) qu'il n'était plus nécessaire d'installer ou de décrire en détail ».*
Au niveau des réponses à appels d'offre : *« Nous avons dû faire un gros travail de réécriture sur la présentation fonctionnelle des différents modules tout comme au niveau des captures d'écrans BO, y compris sur des fonctionnalités qui n'avaient pas évolué comme la gestion des utilisateurs, la gestion des droits et des rôles… »* explique Emmanuel, consultant.
L'arrivée de Drupal 8 a aussi eu un impact sur l'approche stratégique de nos réponses :*« Dans les appels d'offre publics, il est très rarement possible de discuter avec l'acheteur sur le choix de la solution CMS et moins encore sur la version à utiliser. Lorsque certains modules inexistants sur D8 sont imposés - comme Scald –, on sait tout de suite qu'on ne peut pas proposer Drupal 8. Pendant plusieurs mois nos offres proposaient de partir soit sur Drupal 7, soit sur Drupal 8 en expliquant au client les avantages et les inconvénients de chacune des versions. Pour cela, les offres comprenaient un tableau comparatif exhaustif précisant pour chaque fonctionnalité le type de solution envisagé et, éventuellement l'état du module pressenti ».*

Un choix difficile entre D7 et D8

Aujourd'hui, le choix de Drupal 8 s'impose plus facilement de lui-même. Toutefois, pendant plusieurs mois il a été difficile d'acter un choix définitif en phase d'avant-vente. Notre approche était plutôt de laisser le champ libre à la discussion et de valider la version en phase de spécifications : *« jusqu'à présent, nous avons*

eu plutôt de bonnes surprises dans ce sens. En cas d'absence de module équivalent sous Drupal 8, nous avons souvent trouvé des modules suffisamment équivalents pour couvrir la demande » explique un autre chef de projet. « *C'est vrai que développer un nouveau site sur Drupal 8 est plus motivant. Et puis, cela permet aussi de démontrer une expertise sur Drupal 8, qui peut être un élément différenciateur par rapport à certaines sociétés concurrentes* ».

La pérennité de la solution reste, pour un grand nombre de nos clients, un critère déterminant. La majeure partie d'entre eux ont finalement préféré reporter une fonctionnalité ou bénéficier d'un fonctionnement légèrement biaisé pour profiter d'une solution CMS très pérenne et permettant de s'affranchir d'une migration « coûteuse » qui sera obligatoire d'ici 3 à 5 ans ; ceci lorsque Drupal 7 disparaitra à son tour, remplacé par Drupal 9.

L'arrivée d'une nouvelle version majeure d'une solution (Drupal ou autre) peut avoir un aspect anxiogène complètement compréhensible de la part de certains clients, pour qui l'installation d'une nouvelle version majeure est souvent synonyme de « problèmes à l'horizon », tant concernant la pérennité de la solution que l'écosystème qui fait aujourd'hui sa renommée.

Une concurrence des autres solutions

La difficulté de cette période a justement été la concurrence d'autres solutions, qui exploitaient l'instabilité et les retards de Drupal 8 pour pousser une solution décrite comme plus mature ou plus pérenne. « *Honnêtement, c'est de bonne guerre. Cela nous oblige à être vigilants et à rétablir certaines vérités dès la phase d'Avant-vente pour quelques clients parfois trop crédules. Les arguments sont souvent fallacieux. Ceux qui se basent sur la « pauvreté » de l'écosystème de D8 (plus de 2000 modules quand même !) oublient de préciser qu'il a fallu plusieurs années pour que Drupal 7 puisse bénéficier de plus de 13 000 modules* » explique Emmanuel.

Rappelons aussi que la mise en place d'un site (qu'il s'agisse d'une création ou d'une refonte), nécessite un travail de 5 à 6 mois, ce qui laisse le temps de voir émerger de nouveaux modules. C'est en effet ce qui s'est passé pour Drupal 8 dont le nombre de modules explose depuis le début de l'été. « *Quant aux éventuels risques de sécurité liés à la jeunesse du produit, Drupal 8 a été testé pendant des mois de manière très minutieuse par une communauté très motivée. Sortir un produit bogué aurait été une grosse erreur. Bien au contraire, ils se sont donné le temps de peaufiner le produit, quitte à repousser régulièrement sa disponibilité. Au final, la version 8.0.0 était réellement stable* » confie Abdenour. Malheureusement le risque zéro n'existe pas et Drupal 7 et 8 ont été touchés il y a quelques mois par une faille de sécurité majeure. « *La faille a été signalée puis résorbée dans les heures qui ont suivi par les équipes Sécurité de Drupal. Aucune solution n'est à l'abri d'une faille de sécurité, l'importance c'est surtout qu'elle ait été résolue très rapidement* » justifie Henry, un des développeurs.

Avec D8, le plus beau reste à venir

Nous ne regrettons pas d'avoir consacré autant d'énergie sur nos premiers projets Drupal 8 et essuyé quelques plâtres : « *certains modules n'étaient pas assez stables et parfois, nous n'avions pas d'autre choix que de partir sur un développement custom plus ou moins important en remplacement d'un module vraiment instable* » constate Abdenour. Le choix de partir sur Drupal 8 pour certains projets s'est révélé à chaque fois positif : « *Une grande partie de nos clients ne connaissait pas la version précédente, cela a permis de ne pas souffrir de la comparaison et de limiter l'effet déceptif. Je ne m'attendais pas de toute façon à ce que la version 8 soit aussi révolutionnaire qu'annoncée : ce qui était faisable avant l'était après, parfois même avec plus de fluidité* » développe Cyrielle. « *J'ai quand même eu le sentiment que les contributeurs de contenus sont, pour le moment, les grands oubliés de Drupal 8. Le Gap entre les 2 versions majeures est minime et se résume à la mise en avant de 2 grandes fonctionnalités (Front-end édition / Back-Office en Responsive) déjà existantes sous forme de modules. Difficile dans ce cas de défendre la mise en place de D8 auprès de certains contributeurs intéressés uniquement par les aspects éditoriaux* » explique Emmanuel. C'est vrai que les créateurs de contenus / contributeurs n'étaient pas jusqu'à présent une cible privilégiée. Mais, cette « mise à l'écart » - temporelle – s'expliquait par la nécessité de poser les « bases » d'une architecture saine (en se débarrassant notamment de certains écueils liés à Drupal 7) pour envisager ensuite la mise en place de nouvelles fonctionnalités, parfois même directement au niveau du « core » Drupal 8. Depuis que Drupal dispose d'un modèle et d'une architecture technique, l'écosystème semble avoir gonflé d'un coup, passant de 1300 à plus de 2000 en quelques mois. Certains modules très sollicités sur Drupal 7 ont été portés ou réécrits. « *Tout n'est pas parfait mais sur nos derniers projets en 8.1, nous nous sommes rendus compte qu'une grande partie des lacunes avait été comblée* » commente Cyrielle.

L'UX et la contribution BO comme axe majeur d'amélioration

En avril 2016, un grand sondage est lancé auprès de la communauté pour définir les futures orientations de Drupal 8, voire de Drupal 9. Lors de la Keynote de Mai 2016 (à lire et à écouter sur http://buytaert.net/state-of-drupal-presentation-may-2016), Dries présente les résultats de ces orientations. Une attention toute particulière est enfin portée à l'expérience contributeur (EX) mais les développeurs (DX) et les utilisateurs finaux (CX) ne sont pas abandonnés pour autant. Réparties en 9 initiatives, ces orientations couvrent des besoins spécifiques à chacun des profils que nous venons de décrire et concernent : la gestion de la médiathèque et plus généralement des médias (initiative media), les workflows (initiative workflow), la migration de Drupal 6 et 7 vers Drupal 8 (initiative migrate), la gestion simplifiée des pages et des blocs (initative « Blocks and Layout »), le thème (initiative theme component library), la modélisation des données via des interfaces plus intuitives (initiative Data Modeling), l'API Rest (initiative API-first) , la contextualisation du parcours utilisateur (initiative Orchestration), la diffusion de contenus et d'informations sur différents supports innovants via texte et / ou voix (initiative Cross Channel).

Certaines d'entre-elles sont encore à l'état de POC ou de développements peu avancés mais d'autres sont déjà disponibles. C'est le cas par exemple de « l'initiative média », sortie en Juillet ou encore de l'initiative « Blocks and Layout », l'une des grandes nouveautés de Drupal 8.2 (du moins annoncée au moment où nous écrivons cet article). Ces initiatives, déjà très complètes sont promises à de nouvelles évolutions au fur et à mesure des futures versions de Drupal, ce qui démontre bien un véritable intérêt à les peaufiner encore et encore.

Bien sûr, la plus grande force de Drupal reste sa communauté, toujours très active, qui, après un petit coup de mou sans doute explicable par la nécessité d'appréhender un socle technique assez nouveau, a été d'une remarquable efficacité. Depuis la sortie de la 8.1, de nombreux modules D7 ont (enfin) été portés ou remplacés par d'autres équivalences.

L'initiative Drupal 8 est enfin en marche. Il serait bien dommage de s'en priver !

Longue vie à Drupal ! •

Les dernières nouveautés d'Unity !

• Jonathan ANTOINE et Maxime FRAPPAT
Développeurs
Infinite Square
http://blogs.infinitesquare.com/

Quelques mois après notre venue à Unity Unite à Amsterdam, il est temps de refaire un tour des dernières nouveautés autour de ce magnifique éditeur qu'est Unity.


1

Les objectifs annoncés sont clairs :

- Démocratiser et faciliter le développement de jeux et d'applications ;
- Permettre de résoudre des problèmes simples mais aussi de mettre en œuvre des solutions complexes ;
- Aider les développeurs à réussir.

Encore plus orienté design et puissance

Un des premiers critères de succès d'un jeu est son rendu graphique, sa beauté perçue par les joueurs et il est donc normal qu'Unity investisse beaucoup sur ce sujet. Pour cela, l'éditeur choisit de démontrer les possibilités du moteur autour d'un court-métrage nommé « Adam ». Celui-ci est disponible sur TonTuyau à cette adresse : https://www.youtube.com/watch?v=GXI0l3yqBrA . [Fig.1]

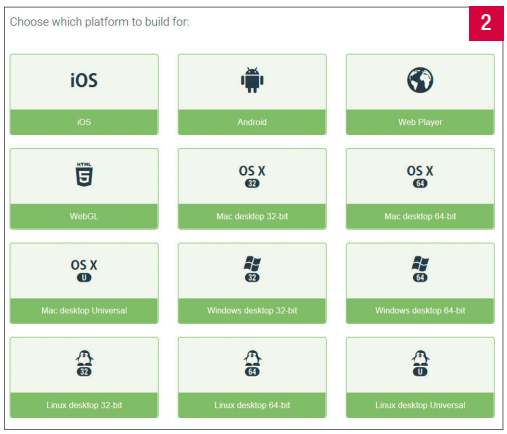
Le challenge ici est de produire un film d'une beauté époustouflante … En temps réel sans post-production comme cela est le cas à Hollywood ! Je vous laisse juger du résultat par vous-même mais il n'y a pas de doute pour moi qu'Unity est capable du meilleur.

En dehors de cette démonstration, Unity s'enrichit constamment d'outils pour en faire le préféré des designers :

- LookDev : pour tester rapidement un rendu dans différentes scènes et environnements ;
- Mise à jour des possibilités du système de particules et d'affichage de lignes ;

Qualité

Vous l'avez sans doute remarqué mais Unity évolue avec le marché et les grandes tendances : très vite. Rien qu'au cours de l'année dernière, nous avons vu l'apparition du support de la VR, de la réalité augmentée (HoloLens), l'ajout de services (Cloud Build, Analytics, Monétisation, etc.), le support de WebGL, IL2CPP, et l'ajouts de fonctionnalités à l'éditeur lui-même. Développeur que nous sommes, nous savons que le risque sous-jacent est de développer vite et … mal.



2

L'équipe dirigeante d'Unity ne veut pas tomber dans ce travers et met en œuvre les moyens nécessaires pour l'éviter.

- L'équipe de testeurs s'agrandit constamment ;
- La couverture de tests (manuels et automatiques) pour chaque version s'agrandit aussi ;
- Les players (ce qui est déployé sur les téléphones) sont aussi testés sur de plus en plus de téléphones. On connait l'importance de ce point sur Android !
- Le système de Beta publique continue de fonctionner et permet d'obtenir un retour des utilisateurs assez massifs.

De plus Unity continue de s'engager à un support de 6 mois sur ces versions majeures : que cela soit l'éditeur en lui-même ou les players spécifiques à chaque plateforme.

Faire gagner du temps aux développeurs

Toujours dans une démarche d'accompagnement des développeurs dans la création d'applications et jeux, Unity capitalise sur ses connaissances pour proposer des services connexes faisant gagner un temps précieux lors des phases de développement et de tests.

Unity Cloud Build

Ce service permettant de compiler vos projets sur les serveurs d'Unity plutôt que sur vos propres machines n'est pas nouveau, mais il continue d'évoluer. Il est maintenant possible de configurer le déclenchement de la build à chaque checkin et d'envoyer directement un lien aux testeurs. Cela permet la création de workflow de test sans opération manuelle et donc avec moins de temps perdu et d'erreur ! [Fig.2]

Unity Performance Reporting

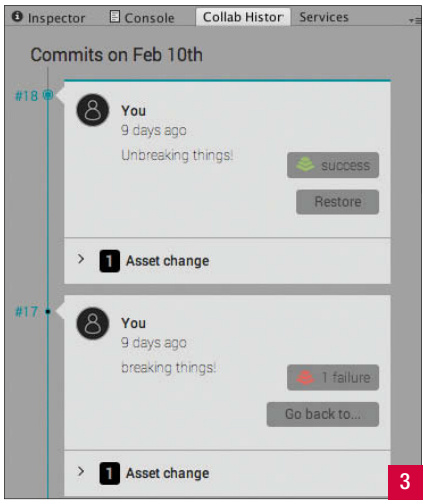
Ce tout nouveau service permet de configurer rapidement et simplement votre application pour envoyer ses logs d'erreurs sur une plateforme Unity. Cela permet de bénéficier simplement :

- D'une plateforme centralisée de gestion des erreurs (peu importe la plateforme) ;
- D'un affichage en temps réel des erreurs provenant de vos applications pour pouvoir réagir rapidement ;
- De messages et stack trace complète pour un debug rapide .

L'activation se fait un en un clic directement dans Unity : il suffit d'activer le service dans votre projet.

Unity Collaborate

Un point négatif d'Unity est la gestion du code source dans une équipe de développeurs : les conflits sont difficiles à éviter et il faut vraiment une rigueur acharnée pour ne pas perdre le travail de l'un des développeurs dans ces scenarii.



3

Unity, conscient de cette faiblesse propose tout simplement d'intégrer un gestionnaire de code source dans l'éditeur : « Unity Collaborate ».
Encore sous la forme d'une version bêta « privée », il sera possible de placer son code source sur les serveurs Unity, de gérer les membres d'une équipe, de synchroniser les différents changements de code de façon fluide. La puissance de cet outil réside dans le fait qu'Unity maîtrise bien son éditeur et peut éviter les scénarios conflictuels au maximum. [Fig.3]

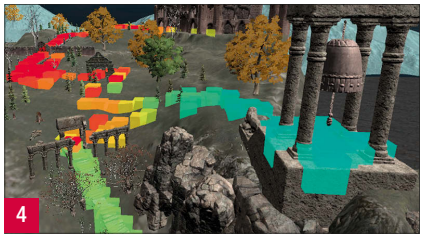
Unity Analytics

L'analyse d'usage d'un jeu est un point clef de son succès : en connaissant ce que font les utilisateurs, il devient possible de savoir comment les garder plus longtemps ou à quel moment ils quittent l'application en raison d'une difficulté trop élevée par exemple. De grands acteurs du marché proposent déjà leurs outils (Google Analytics, Microsoft AppInsights, etc.), mais Unity propose ici un outil cross-plateformes simple à intégrer : pas de SDK à installer, tout est intégré à l'éditeur. L'outil connaît un réel succès depuis sa sortie en 2015 et Unity continue donc logiquement d'investir dessus en apportant des nouveautés complétements uniques :

- Vue de l'activité des joueurs directement dans la scène Unity sous la forme d'une HeatMap : excellent pour connaître les points de blocages potentiels ;
- Vue du FPS des joueurs dans la scène Unity sous la forme d'une HeatMap : excellent pour détecter l'origine des problèmes de performance ;
- Export Excel --> permet de cibler la zone géographique, la plateforme, le device… : parfait lorsque l'on souhaite faire une analyse poussée des données.

Unity IAP

Une méthode de monétisation connue comme efficace consiste à proposer des achats à l'intérieur de vos applications/jeux : les fameux IAP. Cette méthode devient vite fastidieuse lorsque


4

vous devez déclarer chacun des produits vendus sur chacun des stores possibles et imaginables : Windows, Android, iOS, Amazon, Tizen, etc. De plus il faut développer le code spécifique (ou acheter un plugin sur l'asset store unity) pour chacune des plateformes. Cela est souvent bloquant pour les développeurs qui n'ont pas forcément les ressources (de budget et de temps !) pour le faire.
Unity propose donc maintenant un service et des APIs unifiées sous le nom « Unity IAP ». Activable en quelques clics/lignes de code il devient simple de mettre en place des achats in-app ! De plus, il est possible de combiner cette fonctionnalité avec Unity Analytics pour visualiser et analyser le processus d'achat des produits.

Nouveau modèle de souscription

Jusqu'il y a peu, Unity proposait un système d'abonnement (une année et renouvelable chaque mois ensuite) par plateforme : il fallait acheter une licence de base puis une licence spécifique pour iOS, puis une autre pour Android, etc. Il était aussi possible d'avoir une licence perpétuelle (mises à jour gratuites) ou une licence pour une version bien spécifique de l'éditeur. Finalement il fallait souscrire à chacun des services de manière séparée avec pour chacune une tarification dédiée…. Cela pouvait être assez complexe à comprendre et à gérer. Unity a donc fait le nécessaire pour simplifier cela !
Dorénavant la licence Unity correspond à :

- Toutes les plateformes : le système d'add-on disparaît !
- Les mises à jour gratuites font partie de la licence. Il est sûrement moins coûteux pour Unity de vous faire bénéficier des correctifs des nouvelles versions que de devoir maintenir plusieurs versions en même temps.
- Tous les services sont inclus de base dans toutes les souscriptions.

Il existe alors 3 niveau de souscriptions possibles :

- Unity personal – version gratuite comme auparavant mais avec toutes les fonctionnalités. Il est maintenant possible de personnaliser le splashscreen !
- Unity Pro (125$/mois) - maintenant avec toutes les plateformes incluses (plus besoin d'acheter iOS et Android!). Il reste possible d'acheter une version spécifique ;
- Unity Plus (35$/mois) : la version pro avec des options, possibilités un peu plus limitées. Uniquement possible si vous avez un revenu inférieur à 200 000$ / an.

Des packs d'Assets pour créer des jeux rapidement

Unity propose dorénavant des réductions sur des packs d'assets graphiques et sonores au bénéficiaires d'une licence PRO ou PLUS. Ceux-ci permettent de créer rapidement un prototype (ou un jeu – l'usage n'est pas contraint) tout en ayant une qualité graphique et sonore digne d'un jeu grand public.

- Les avantages sont multiples :
- Les assets sont optimisés pour un usage avec Unity ;
- Les assets sont de très bonne qualité ;

Le téléchargement se fait directement depuis l'asset store : disponible en quelques clics !
Unity propose des scènes pré-faites de mise en œuvre des assets.

Le futur du jeu vidéo

Unity pense aussi au futur des jeux vidéo en proposant le développement d'application de réalité virtuelle directement depuis son éditeur. L'idée sous-jacente est de proposer une plateforme unifiée de la même manière que pour un développement « classique ». Ainsi, Unity propose une seule et unique API, et, au fur et à mesure que des nouveaux périphériques apparaissent, Unity ajoute les players correspondants. Bien sûr la réalité augmentée n'est pas en reste car il est déjà possible de développer des applications HoloLens avec Unity. Voici une liste non-exhaustive (bien que déjà impressionnante !) des périphériques supportés : HoloLens, Oculus Rift, Samsung Gear, HTC Vive. En plus de cela, Unity travaille sur des projets portant sur l'intégration de l'éditeur directement dans le monde « virtuel ». Une démonstration bluffante a été faite pour permettre l'aménagement (ajout d'objets, redimensionnement, déplacement, etc.) dans une scène Unity … tout en étant dans un Oculus Rift !

CONCLUSION

La conclusion est simple à écrire et vous l'aurez devinée : Unity compte bien faire le maximum pour faciliter la vie des développeurs et utilise toutes ses ressources et son expérience pour réussir. Au fur et à mesure que le temps passe, de nouveaux services apparaissent pour faire gagner du temps (et de l'argent !) à tous de façon simple. •

Swift 3.0 est là

• Pascal Batty
Expert iOS chez SOAT

À l'ombre de la sortie de l'iPhone 7 et d'iOS 10, Swift continue son évolution et a atteint sa version 3.0 en septembre. Cette nouvelle mouture est la première release importante à être issue du nouveau processus Open Source. Comment s'est déroulée cette année d'évolution et quel est l'état actuel du langage ?

À l'ouverture du projet Open Source le 3 décembre 2015, les objectifs pour Swift 3.0 avaient été clairement énoncés par Chris Lattner, créateur du langage :

- **Stabilisation de l'ABI (Application Binary Interface) et résilience** : pour permettre l'interaction entre des bibliothèques compilées dans différentes versions du langage et éviter que chaque programme Swift ait besoin d'embarquer sa propre version des bibliothèques standard ;
- **Portabilité** : rendre le langage et la bibliothèque standard disponibles sur de nouvelles plateformes
- **Nettoyage et documentation du système de typage** : afin de mieux cadrer et simplifier l'ensemble du système de typage
- **Généricité complète** : afin d'étendre le support de la généricité
- **Concentration et raffinement du langage et définition de lignes de conduite de conception d'API** : pour consolider la grammaire et le vocabulaire utilisés dans le langage.

La "Core Team" visait pour la 3.0 une véritable consolidation technique du langage dans les fondations avec la stabilisation de l'ABI et du système de typage, mais aussi fonctionnelle avec la construction de conventions et l'établissement de lignes directrices claires. L'objectif annoncé était clair : effectuer le plus de *breaking changes* pour la 3.0 pour permettre une stabilité du code source à partir de cette version-là. Ce qu'il fallait casser, il fallait le casser le plus tôt possible, afin d'augmenter la confiance des développeurs dans l'adoption du langage à l'horizon fin 2016.

Processus Open Source

À l'aube de ce projet Open Source, Swift n'avait été utilisé par les développeurs que pendant une année, et très rarement dans des applications en productions. La Core Team s'est rendue très ouverte aux propositions à travers des mailing lists et un processus très cadré de *pull requests*. La communauté pouvait discuter des évolutions avant et pendant le processus de révision par la Core Team.

C'est bel et bien à force de l'utiliser que les développeurs dans le monde réel, loin des ingénieurs qui l'avaient conçu, ont remarqué les incohérences et les imperfections de Swift et de ses API. Le feedback de la communauté a réellement aidé le langage a ne pas juste évoluer comme un projet d'ingénierie "pour la beauté du langage" mais réellement heurté à des problématiques du développement d'application. En conséquence, le processus d'évolution de cette année ne s'est pas tout-à-fait déroulé comme prévu, et certains objectifs ont dû être laissés de côté.

L'heure du bilan

Le 16 mai, Chris Lattner avait communiqué avec la communauté des objectifs réévalués pour Swift 3.0. Les objectifs les plus techniques, comme la stabilisation de l'ABI et l'extension de la généricité, ont été abandonnés pour laisser la priorité à l'avalanche de changements proposés et vus avec la communauté. Ces évolutions concernent dans l'ensemble l'utilisation du langage plus que ses aspects internes. L'équipe a traité 140 propositions, venant de la communauté ou de membres de la Core Team, dont 98 ont été implémentées dans le langage que l'on peut utiliser aujourd'hui. Cette release 3.0 a réellement changé le langage sur de nombreux aspects. Les développeurs de la communauté et la Core Team se sont notamment mis d'accord sur des conventions de conception claire et dans l'ensemble le langage et la bibliothèque standard semblent beaucoup plus cohérentes. Elle agrandit également le périmètre de la bibliothèque standard avec l'ajout de Foundation (incluant notamment les dates et les objets réseau) et Dispatch (les outils d'asynchronisme et de parallélisme) qui sont à présent accessibles dans la version Linux du langage.

Processus de migration

Apple avait lors de la WWDC en juin livré Swift 2.3 qui était en grande partie source-compatible avec 2.2 et accessible dans les nouveaux outils de développement. Aujourd'hui néanmoins, la communauté doit avancer vers 3.0 dans un seul et même mouvement.

Pour les développeurs ayant du code Swift 2.x en production, le processus de migration est assez complexe même s'il est en grande partie assisté par Xcode. L'instabilité de l'ABI nécessite que chaque dépendance d'une application soit également migrée vers Swift 3.0 pour fonctionner, et tous les mainteneurs de projets Open Source (frameworks, bibliothèques, modules, …) n'ont pas forcément suivi la cadence.

Tout ceci est bien entendu causé par le nombre important de *breaking changes* présents dans Swift 3.0, mais ils sont le prix accompagnant la promesse d'un code source très stable pour les versions à venir.

CHANGEMENTS NOTABLES

Une grande partie des changements suivent l'établissement des nouvelles directives de conception des API (API Design Guidelines). De nombreux types et fonctions ont en effet changé de nom, et dans la majorité des cas l'assistant de migration est capable d'accompagner le changement sans trop de heurts. Les modificateurs d'accès ont beaucoup évolué également et forcent les développeurs à se poser de nouvelles questions. Enfin, les éléments du langages marqués comme dépréciés dans Swift 2.x disparaissent dans la version 3.0 : le mot d'ordre était bel et bien de casser un maximum pour 3.0 afin d'assurer la stabilité du langage par la suite. Pour les changements présentés dans les chapitres suivants, des références seront faites aux propositions de la communauté sous la forme *SE00XX : Nom de la proposition*. Ces propositions peuvent être consultées dans le répertoire Proposals du projet Github Swift-Evolution : https://github.com/apple/swift-evolution

Nouvelles directives de conception

Que ce soit pour rendre hommage à son aîné ou pour des raisons de compatibilité, Swift portait en lui un important héritage d'Objective-C

dans le nom des fonctions et des paramètres. Dans cette version 3.0, le langage prend son indépendance et trouve son identité.
Un des chantiers-clés de Swift 3.0 était l'établissement de règles de conception d'API. Avec l'aide de la communauté de développeurs, les nouvelles conventions sur la façon de nommer les fonctions, paramètres, protocoles et structures de données ont été décidées et clairement décrites (https://swift.org/documentation/api-design-guidelines/)
Le langage qui pouvait paraître encore assez verbeux à l'image d'Objective-C trouve une nouvelle voie plus concise sans sacrifier l'expressivité qui le qualifie.

Labellisation des paramètres

Un des aspects surprenants de Swift pour les développeurs venant d'autres langages est le nommage des paramètres au *call site*. Cet aspect très fort en Objective-C a été conservé en Swift pour des raisons de clarté et d'expressivité. Prenons par exemple une fonction insert() permettant d'ajouter un élément dans un tableau à un index donné.

```
// Ceci n'est pas du Swift
var numbers: [Int] = [4, 8, 16, 23]
numbers.insert(15, 3)
```

Est-ce que l'on insère l'élément 15 à l'index 3 ou bien l'élément 3 à l'index 15 ? Impossible à dire en lisant cette ligne.
On peut toujours s'appuyer sur la documentation ou l'auto-complétion lors de l'écriture, mais à la lecture les choses ne sont pas claires. Forcer le nommage des paramètres au *call site* donne d'ordinaire ce genre de résultat :

```
// Ceci n'est toujours pas du Swift
var numbers: [Int] = [4, 8, 16, 23]
numbers.insert(newElement: 15, index: 3)
```

La clarté est là mais le code est assez verbeux : la taille de la ligne a presque doublé. Swift propose d'ajouter aux paramètres un libellé dont l'usage est réservé au *call site*, qui de surcroît est optionnel.

```
// Ceci est bien du Swift
var numbers: [Int] = [4, 8, 16, 23]
numbers.insert(15, at: 3)
```

Ici, le seul ajout de at: sur le second paramètre clarifie l'intention sans ambiguïté : 3 est bien l'index où l'on insère l'élément 15. "Insert 15 at 3", la ligne de code se lit comme une phrase en anglais.
La signature de cette méthode de la bibliothèque standard est :

```
mutating func insert(_ newElement: Element, at i: Int)
```

- mutating indique que la méthode modifie l'état de la structure Array
- newElement est le nom interne à la fonction du premier paramètre, son libellé externe est marqué _, ce qui permet de l'ignorer (on ne nomme pas ce premier paramètre)
- i est le nom interne du second paramètre, at est son libellé externe utilisé au *call site*.

Si l'on n'indique ni libellé ni _, c'est alors le nom du paramètre qui doit être utilisé lors de l'appel de la fonction.
La différence notable entre Swift 2 et 3 à ce niveau concerne le premier paramètre. En Swift 2, le libellé du premier paramètre d'une fonction était ignoré par défaut (sauf pour les constructeurs).
En Swift 3, la règle est la même pour tous les paramètres, que ce soit pour les fonctions ou les constructeurs : par défaut il faut utiliser le nom du paramètre lors de l'appel, on peut indiquer un libellé ou un _ dans la signature de la fonction pour changer ce comportement.

Changements dans la bibliothèque standard

En accord avec les nouvelles directives de conception, de nombreuses API de la bibliothèque standard ont connu des changements de nom ou de fonctionnement. Un des changements marquants est le nom des fonctions de manipulation de séquences comme map(), filter(), reduce(). En Swift 2, ces fonctions produisaient une nouvelle séquence et pouvaient donc être appelées sur des valeurs immutables, comme on peut en avoir l'habitude dans d'autres langages.
Les nouvelles directives indiquent que lorsque la méthode est un verbe à l'impératif (comme map()), c'est alors que la méthode change l'état de la valeur cible. Pour l'équivalent sans mutation on rajoute soit le suffixe -ed (ex: mapped() pour map()) ou -ing (ex: appending() pour append()).
Les fonctions map(), filter() et reduce() deviennent donc mapped(), filtered() et reduced() dans l'usage qu'on leur connaissait jusqu'à présent.
De même, si la fonction est un nom (comme union()), elle désigne la version sans mutation. Pour l'équivalent avec mutation de la valeur, on ajoute le préfixe form- (ex: formUnion() pour union()) et on crée ainsi un verbe à l'impératif. Si ces changements peuvent paraître déroutants pour les utilisateurs d'autres langages, ils permettent de lever l'ambiguïté sur l'effet de chaque fonction. De nombreux autres changements de noms ont eu lieu, ils peuvent être consultés dans la proposition *SE-0006 - Apply API Guidelines to the Standard Library*.

Interface avec les API Objective-C et C

Dans le cadre des applications iOS et macOS, le code Swift a encore besoin de communiquer avec du code Objective-C. Les deux langages sont cependant construits sur des bases et des conventions très différentes. Concernant l'adaptation des noms, un processus élaboré de traitement des API existantes a été mis en place pour rendre l'appel des fonctions Objective-C tout-à-fait naturel dans du code Swift. Ces nombreux changements peuvent être vus dans la proposition *SE-0005 : Better Translation of Objective-C APIs Into Swift*.
Certaines fonctions ont été renommées pour correspondre aux nouvelles directives et sont désormais plus courtes. Si une fonction est composée d'un verbe à l'impératif suivi de plusieurs mots, on aura tendance à ne trouver que le verbe avant la parenthèse, puis une particule comme libellé du paramètre suivant. Le type est souvent ignoré car Swift étant un langage fortement typé, l'ambiguïté n'existe pas.
Par exemple, la méthode saveToURL() de la classe UIDocument devient save(to:). Ce changement est automatisé et marche pour les API existantes d'Apple mais également pour tout code Objective-C d'une application mélangeant les deux langages. Il est néanmoins possible de passer outre le renommage automatique et d'indiquer un nom spécifique à une méthode à l'aide de l'expression NS_SWIFT_NAME().
Objective-C utilise également des variables globales au format chaîne de caractères pour paramétrer certaines méthodes. Ces chaînes sont la plupart du temps signalées par un typedef spécifique. Par exemple, le constructeur de NSCalendar prend en paramètre une chaîne pour l'identifiant du calendrier par exemple la valeur NSCalendarIdentifierGregorian.

```
let cal = NSCalendar(calendarIdentifier: NSCalendarIdentifierGregorian)
```

Ces chaînes ne sont pas toujours faciles à découvrir, et rien n'empêche de passer une chaîne totalement inattendue à cette méthode. Il est possible de donner à ces constantes un type nommé par typedef afin de les

identifier, et ajouter l'expression NS_EXTENSIBLE_STRING_ENUM indique au traducteur Objective-C vers Swift d'utiliser une structure avec des propriétés statiques automatiquement nommées (*SE-0033: Import Objective-C Constants as Swift Types)*.

```
typedef NSString * NSCalendarIdentifier NS_EXTENSIBLE_STRING_ENUM;
```

Et dans l'appel du constructeur en Swift, le type peut même être omis :

```
let cal = Calendar(identifier: .gregorian)
```

L'appel est plus sûr et plus court. En Objective-C, tout appel à une méthode ou une propriété est véhiculé par un runtime. Par conséquent, il est possible de spécifier une méthode en donnant son nom sous forme de chaîne de caractères (on appelle cela un selector), ou de même pour une propriété, voire une chaîne de propriétés (que l'on appelle un keypath).

```
// Un selector en Objective-C pour réagir à une notification (messaging)

[[NSNotificationCenter defaultCenter] addObserver:myObject
      selector:@selector(reactToNotification:)
      name:@"MyCustomNotificationName"
      object:nil];

// valueForKeyPath permet de retourner la valeur d'une sous-propriété
[myObject valueForKeyPath:@"foo.bar"]
```

Ces notions de code "Stringly Typed" sont très utilisées dans les API de Cocoa et Cocoa Touch mais s'éloignent de l'aspect très sécurisé et fortement typé de Swift. De plus, maintenant que les API Objective-C ont deux noms : un nom original et un nom pour Swift, quelle chaîne de caractère donner pour créer un selector ou un keypath ?
En Swift 2.2, l'expression #selector() avait été rajoutée pour permettre de référencer une méthode de façon sécurisée par le compilateur (*SE-0022 : Referencing the Objective-C selector of a method*). Swift 3 ajoute l'expression #keyPath() qui ajoute le même fonctionnement pour le Keypath (*SE0062 : Referencing Objective-C key-paths*). À ces deux expressions on donne une référence vers l'API en Swift sans se poser de question.
Dans l'autre sens, lorsque l'on déclare une API en Swift qui est compatible avec Objective-C, il est possible de déclarer des noms spécifiques à ce langage avec l'expression @objc().
De nombreuses API en C font encore partie de Cocoa et Cocoa Touch. Pour ces API, une transformation est également présente en Swift 3. La convention pour manipuler un objet comme un contexte graphique est de le passer systématiquement comme premier paramètre de fonctions C. Par exemple :

```
let context: CGContext = UIGraphicsGetCurrentContext()!
CGContextSetLineWidth(context, 4)
CGContextSetGrayStrokeColor(context, 0.5, 1.0)
CGContextDrawPath(context, .Stroke)
```

La nouvelle expression NS_SWIFT_NAME permet d'importer des fonctions globales comme des membres d'un type (*SE-0044 : Import as member.*) Une fois renommée, cette API est utilisée de cette manière :

```
let context: CGContext = UIGraphicsGetCurrentContext()!
 context.lineWidth = 4
context.strokeColor = CGColor(gray: 0.5, alpha: 1.0)
context.drawPath(mode: .Stroke)
```

On remarque que certaines fonctions sont traduites comme des méthodes tandis que d'autres deviennent des propriétés. Ces traductions des noms d'Objective-C et de C permettent de s'adresser aux très nombreuses API existantes dans une syntaxe familière mais surtout de façon plus sécurisée en profitant des garde-fous du compilateur.

Contrôle d'accès

Dans Swift 2, les mots-clés de contrôle d'accès étaient :
- public qui autorise l'accès et l'héritage/surdéfinition (override) à tous ;
- internal qui verrouille l'accès en lecture et héritage/redéfinition au contexte du module ;
- private qui verrouille l'accès en lecture et héritage/redéfinition au contexte du fichier.

Swift 3 ajoute 2 nouveaux mots-clés pour le contrôle d'accès : open et fileprivate, et le sens de private et public change.
Ainsi, marquer une classe ou un membre public permet toujours l'usage de n'importe où mais n'autorise maintenant plus l'héritage ou la redéfinition. Le mot-clé open reprend l'ancien sens de public et permet à la fois l'usage et l'héritage/surdéfinition partout (*SE-0117 : allow distinguishing between public access and public overridability*).
Le sens du mot-clé private change pour obtenir le sens qu'on lui connaît d'ordinaire : visibilité limitée au périmètre de définition. Le mot-clé fileprivate prend donc le rôle de définir une visibilité sur le périmètre d'un fichier (*SE-0025 Scoped Access Level*).
On trouve ainsi les mots-clés suivants :
- open : visibilité et héritage/surdéfinition globale ;
- public : visibilité globale mais fermeture à l'héritage/surdéfinition ;
- internal : visibilité et héritage/surdéfinition dans le contexte du module ;
- fileprivate : visibilité et héritage/surdéfinition dans le contexte du fichier ;
- private : visibilité et héritage/surdéfinition dans le contexte de définition.

Disparitions

Les développeurs ont eu un an pour se préparer à la disparition de quelques fonctionnalités du langage, qui avaient été marquées comme dépréciées dans Swift 2. La syntaxe de *currying* qui permettait de séparer les paramètres dans des jeux de parenthèses différents n'est plus disponible (*SE-0002 : Removing currying func declaration syntax*). Par exemple, cette fonction add()() n'est plus acceptée en Swift 3 :

```
func add(_ a: Int)(_ b: Int) -> Int {
   return a + b
}

let five = add(1)(4)
let addTwo = add(2)
let twelve = addTwo(10)
```

Une méthode ne peut plus avoir qu'un seul jeu de paramètres, cette même méthode s'écrit donc de cette manière en Swift 3 :

```
func add(_ a: Int) -> (Int) -> Int {
   return { b in a + b }
}
```

Il faut expliciter le fait que cette fonction renvoie bien une fonction qui prend un Int en paramètre et renvoie un Int.
Aussi, les opérateurs ++ et -- en suffixe comme en préfixe disparaissent définitivement et le compilateur invitera à utiliser += 1 à la place (*SE-0004 : remove the ++ and -- operators*). Le grand compagnon de ces opéra-

teurs, la boucle for "façon C", avec index, condition et itération, disparaît aussi de Swift 3.
Il ne reste donc plus que l'opérateur for ... in qui permet d'itérer à travers une séquence.

```
for element in myArray {
    ...
}
```

Afin de parcourir une gamme (de 1 à 10 par exemple), Swift propose depuis longtemps les opérateurs ... et ..< qui permettent d'écrire des boucles plus expressives :

```
for i in 1..<myArray.count {
    print ("Element at index \(i) is \(myArray[i])")
}
```

Si cela est très efficace pour les cas les plus communs, cette méthode ne couvre pas toutes les situations.

Création de séquences à la volée

Afin de permettre plus de souplesse sur la création de séquences, Swift 3 propose deux nouvelles fonctions globales très polyvalentes (*SE-0094 : add sequence(first:next:) and sequence(state:next:) to the stdlib*).
sequence<T>(first:T, next: (T) -> T?) prend en paramètre un premier élément first et une closure (lambda) prenant en paramètre une valeur du même type et renvoie un optionnel de ce type. Cette closure est appliquée à chaque valeur de la séquence pour obtenir la prochaine, si elle renvoie nil c'est alors que la séquence est terminée.
Cette séquence contient par exemple toutes les puissances de 2 en commençant par 1 :

```
for value in sequence(first: 1, next: { $0 * 2 }) {

}
```

sequence<T, State>(state: State, next: (inout State) -> T?) est un peu plus compliquée, elle prend en compte un état, d'un type différent, qui peut être modifié dans la closure permettant d'obtenir la valeur suivante.

Typealias génériques

Bien que l'extension de la généricité ait été reportée à Swift 4, la version 3 apporte une petite nouveauté qui peut faire la différence dans certains scénarios. Le type system de Swift rend l'utilisation de Typealias très intéressante car elle permet de donner des noms spécifiques à des types pour les inscrire dans une logique de domaine et clarifier l'intention.
Par exemple :

```
typealias UserIdentifier = Int

func getUser(_ id: UserIdentifier) -> User { ... }
```

En Swift 3, il est à présent possible d'indiquer un Typealias sur un type générique (*SE-0048 : Generic Type Aliases*). On peut par exemple définir une équipe comme un tableau d'utilisateurs :

```
typealias Team = Array<User>
```

Cela permet plus de sûreté et de clarté dans le code.
Avant Swift 2.2, le mot-clé typealias permettait également de déclarer un type associé au sein d'un protocole (Interface) pour y ajouter de la généricité. En Swift 2.2, il a été remplacé par le terme plus explicite associatedtype. Swift 3 permet d'utiliser également le mot-clé typealias au sein d'un protocole dans son sens premier : nommer un autre type (*SE-0092 : Typealiases in protocols and protocol extensions*).

Foundation et Dispatch

Un des grands objectifs pour la portabilité et la cohérence de Swift était le portage de deux frameworks essentiels au développement :

- **Foundation**, qui contient de nombreux types plus complexes que ceux de la bibliothèque standard ;
- **Dispatch**, qui permet de faciliter le code asynchrone et le parallélisme.

Foundation est une bibliothèque originalement en Objective-C et son portage a été accompagné de deux propositions : l'abandon du préfixe historique "NS" de tous les types (*SE-0086 : Drop NS Prefix in Swift Foundation*) et la question de la mutabilité pour ces types (*SE-0069 : Mutability and Foundation Value Types*).
Certains de ces types ont changé pour s'accorder à la forte importance des types valeurs et de l'immutabilité en Swift. Ainsi, NSDate qui était une classe en Objective-C devient Date, une structure Swift avec la possibilité de la rendre immutable à l'aide du mot-clé let. Elle est également plus "Swifty" avec des opérateurs + et - redéfinis pour lui ajouter ou retrancher une TimeInterval pour la modifier, alors qu'il fallait appeler des méthodes au nom alambiqué en Objective-C.
On trouvera le même genre de modifications, comme l'ajout du subscript pour accéder facilement à un octet selon son index, sur Data (NSData en Objective-C) qui devient également une structure. Néanmoins, une instance de Data peut être très volumineuse et l'utilisation de types valeurs plutôt que de classes implique une grande quantité de copies.
C'est pourquoi on a implémenté un comportement de *Copy-on-write* sur ce type ainsi que sur d'autres qui ont des chances d'être volumineux : deux copies de la même structure Data pointent en réalité vers une même classe qui, elle, contient les données, et si l'une des deux est mutée c'est à ce moment-là qu'une seconde copie de la donnée est réellement créée.
Ces nombreux changements permettent d'exploiter les fonctionnalités de Swift sans sacrifier la performance et l'empreinte mémoire.
Le framework Grand Central Dispatch est accessible en Objective-C depuis 2009. Il s'agit d'un ensemble d'abstractions par-dessus les threads pour simplifier l'asynchronisme et le parallélisme. Swift 3 voit l'arrivée de Dispatch, une version beaucoup plus profilée pour le nouveau langage (*SE-0088 : modernize libdispatch for Swift 3 naming conventions*).
Dispatch permet de s'adresser à plusieurs instances de DispatchQueue, sur lesquelles on peut empiler des closures ou des DispatchWorkItem de manière synchrone ou asynchrone. Dispatch permet également l'utilisation de groupes de tâches à effectuer ou à attendre, de sémaphores et de deadlines. L'utilisation élémentaire de Dispatch est relativement aisée mais ses nombreux composants permettent un contrôle très précis des problématiques de parallélisme et d'asynchronisme que l'on peut rencontrer dans une application pointue.
Foundation et Dispatch faisant à présent partie du projet Swift dans son ensemble, ces briques sont accessibles dans la version Linux du langage.

SWIFT SUR LINUX

Avec la nouvelle stabilité des sources et l'ajout de Foundation et Dispatch au projet cœur, Swift sur Ubuntu est plus puissant et viable que jamais. La marche à suivre décrite dans le numéro 197 de Programmez pour installer et utiliser le langage sur Linux est toujours valable, et la page Getting Started du site officiel (https://swift.org/getting-started/) explique très bien les choses : un téléchargement, un apt-get pour obtenir Clang et

vous voilà lancé ! Le développement d'applications iOS et macOS n'est pour l'instant pas d'actualité sans Mac car les API Cocoa et Cocoa Touch sont loin de faire partie du projet Open Source. Il est néanmoins possible d'utiliser le REPL, d'écrire des scripts et surtout des exécutables en utilisant Swift.
La portabilité vers de nouvelles plateformes avance également avec des portages pour Raspberry Pi et Android notamment.
Le développement sur Linux bénéficie également de l'usage de Swift Package Manager, pour l'instant impraticable pour les développeurs utilisant Xcode (ironiquement). Si Apple ne propose pas encore de repository centralisé, IBM en a mis un en place : le Swift Package Catalog (https://developer.ibm.com/swift/the-ibm-swift-package-catalog/). On y trouve de nombreuses bibliothèques essentielles et omniprésentes comme Alamofire (pour embellir les appels réseau), SwiftyJSON (pour le parsing JSON), RxSwift (le portage de ReactiveX !), PromiseKit (l'implémentation des Promise en Swift)… On y trouve également des frameworks de développement Web, qui pour l'ensemble ont atteint une version stable au moment où Swift passait en 3.0.

Swift côté Serveur

Si Swift sur Linux présente un intérêt c'est en premier lieu pour le développement côté Serveur. Il est alors possible d'être "Full Stack Swift Developer" et de réutiliser des couches communes entre une application iOS et le Web Service qui l'alimente.
L'arrivée de Dispatch et Foundation dans le projet cœur démultiplie le potentiel des frameworks serveur. On en trouve actuellement 3 qui se démarquent : **Kitura** proposé par IBM (https://developer.ibm.com/swift/kitura/), **Vapor** (http://vapor.codes) et **Perfect** (https://www.perfect.org).
Tous les trois sont très légers et s'inspirent de l'état de l'art des frameworks actuels. Kitura a pour lui l'attrait d'être porté par un mastodonte qui semble avoir beaucoup investi dans Swift. Il est d'ailleurs possible, à l'aide de plug-ins, de développer et déployer une application Kitura sur Bluemix (la plateforme Cloud d'IBM) à l'aide de Xcode, ce qui permet un certain confort. Vapor, quant à lui, annonce s'exécuter sur Heroku, AWS et Docker. Swift et Foundation étant conçus pour propulser des applications côté client, on n'y trouve pas encore certaines briques essentielles au développement côté serveur comme la production de réponses, un moteur de templating ou encore des modules d'accès aux bases de données autres que SQLite. Aussi, chacun de ces frameworks doit produire ses propres solutions à ces problèmes.
Espérons que certaines de ces initiatives seront prochainement incluses dans le projet cœur afin de consolider les conventions pour le développement serveur en Swift !

ESSAYER SWIFT

Sur Mac, la solution la plus sympathique est d'installer Xcode gratuitement depuis l'App Store et d'ouvrir un nouveau Playground. Un Playground est le croisement entre un fichier et un REPL : il s'agit d'un fichier de source qui s'exécute constamment, chaque ligne de code est évaluée en temps réel, il est possible de revenir en arrière et de modifier des valeurs et de voir les changements à travers l'évaluation. Un usage plus avancé permet aussi d'y concevoir des vues interactives !
Sur iPad (à partir de l'iPad Air ou du mini 2 sous iOS 10.0), l'application Swift Playgrounds gratuite permet également de s'essayer aux joies du code interactif. L'application propose également des leçons pour apprendre à coder, ça peut révéler des vocations !
Sur Linux, une installation standard de Swift permet d'utiliser le REPL, ce qui est un excellent moyen de s'essayer à un langage. S'il n'existe pas encore de solution supportée sur Windows, il est toujours possible d'utiliser la Sandbox gratuite d'IBM (https://swiftlang.ng.bluemix .net/#/repl) pour écrire et jouer du code Swift sur n'importe quel navigateur Web. Elle s'est alignée sur la version 3.0 du langage et propose, en plus de quelques exemple de codes, d'enregistrer ses bouts de code et de les partager.
Il est également possible faire tourner ses méninges sur des exercices de code car Swift est supporté sur la plupart des plateformes populaires comme Codingame, Hackerrank, et Exercism.

SWIFT 4.0 ?

Cette fois-ci, le développement aura lieu en plusieurs stades, et le premier consiste principalement à opérer une stabilisation. Les tâches de stabilisation des sources qui ne seraient pas passées dans Swift 3.0 en font partie, mais aussi et surtout la stabilisation de l'ABI et la résilience, qui étaient des objectifs importants pour 3.0.
Le stade 1 comporte également en second plan certaines nouveautés juteuses comme l'amélioration de la généricité pour se conformer au maximum au Manifeste Générique (https://github.com/apple/swift/blob/master/docs/GenericsManifesto.md), des évolutions autour du traitement des chaînes de caractères (avec le but ambitieux de battre Perl sur ce plan !), ainsi qu'un modèle de propriété de la mémoire (à la manière de Rust) afin de rendre l'exécution plus prévisible.
Dans le stade 2 se trouvent les évolutions qui auront lieu "si on a le temps" : une meilleure réflexion, une amélioration de la concurrence (async/await ?), encore plus d'améliorations dans la généricité, les expressions régulières et des attributs de propriétés.

Participez !

Swift est donc encore un projet très dynamique ! Il est toujours possible de participer à son développement et à son évolution, soit en participant aux nombreuses discussions qui ont lieu sur la mailing list swift-evolution pour produire, soit débattre sur les nombreuses propositions qui construisent l'avenir du langage. Il est aussi possible de résoudre des bugs sur le Jira de Swift (https://bugs.swift.org) et de produire des Pull Requests. Il est possible de trouver dans la newsletter hebdo Swift Weekly Brief (https://swiftweekly.github.io) des tâches faciles pour commencer.
L'espoir de voir Swift devenir un langage omniprésent continue de se concrétiser avec cette version 3.0 et grandit davantage avec la roadmap ambitieuse pour la version 4.0.
La philosophie de départ de créer un langage capable de tout semble aujourd'hui plus solide que jamais. •

LES +

- Le code source enfin stabilisé
- Une syntaxe concise et élégante
- Foundation et Dispatch disponible pour tout le monde
- Encore plus safe que son prédécesseur

LES -

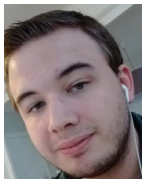
- Tous les noms ont changé
- Pas compatible avec les libs en Swift 2.x
- Toujours pas de compatibilité Windows (1)

COMPATIBILITÉ

- Xcode (sur Mac)
- VIM
- Ubuntu
- Android
- Raspberry Pi
- Heroku
- Bluemix
- Docker

(1) Apple n'a pas engagé le portage Windows, contrairement à la version Linux. Il faudra attendre que la communauté le fasse ce qui pourrait être le cas dès 2017

watchOS 3 : les nouveautés



Kévin Sibué
ksibue@gmail.com
http://www.kevinsibue.com

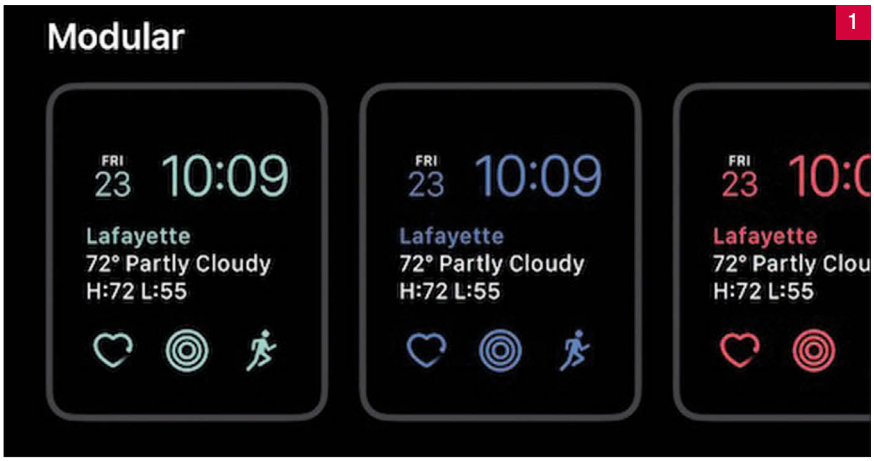
Comme vous le savez très certainement (sauf si vous vivez dans une grotte depuis quelques années), les géants du monde mobile se battent dans une guerre impitoyable pour conquérir le marché du mobile. Cette guerre a été menée sur tous les fronts : systèmes d'exploitation, brevets, capacités des machines, et, depuis quelques années sur le créneau des montres connectées. Google fut le tout premier à sortir sa montre connectée pour Android mais quelques mois après son concurrent direct, Apple, finit par sortir sa première montre connectée: l'Apple Watch.

Cette montre était très attendue par tous les technophiles et Apple addicts de la planète. Ici encore la plupart pensait que la firme de Cupertino allait changer la donne et offrir un produit qui surpasserait les autres. Revenons donc sur les différentes versions de cette montre connectée.

Retour en arrière

Dans sa version 1, la montre d'Apple servait principalement de centre de notification. Lorsque votre téléphone iOS recevait une notification, celle-ci était directement envoyée sur votre montre. Le gros point positif de cette montre était le capteur d'activité qui, suite aux tests, semblait être meilleur que son cousin chez Google. Par ailleurs elle ne pouvait être appairée qu'avec un seul téléphone iOS (les iPad n'étant pas compris). Qui plus est, aucune des applications sur la montre était "native". Ce que j'entends par là c'est que l'ensemble des applications de la montre étaient directement sauvegardées et exécutées sur le téléphone associé. Ce qui sous-entend une constante discussion via le Bluetooth entre watchOS et iOS. Et vous allez voir que ce détail aura son importance. Les utilisateurs se plaignaient aussi du manque de customisation des cadrans de la montre. Google, pour sa part, proposait déjà de très nombreux cadrans à la disposition des utilisateurs et les développeurs pouvaient même en créer. Ce qui n'était pas le cas sur l'Apple Watch. Mais le gros défaut de cette version 1 fut la longévité de la batterie: moins de 24h. Certains parlent même de seulement 4 à 6 heures en utilisation intensive.

Pas génial quand l'utilisateur doit recharger sa montre entre midi et deux pour espérer avoir l'heure dans l'après-midi. Les utilisateurs d'iPhone connaissaient déjà ce problème et malheureusement la marque à la pomme a récidivé avec sa montre. Et c'est ici que la discussion constante entre la montre et le téléphone est une mauvaise idée. Tous ces défauts ont fait que le nouveau gadget d'Apple fut moyennement bien accueilli par les technophiles et les amateurs de sport. Mais Apple a très rapidement réagi pour sortir moins d'un an après une mise à jour complète de l'OS et de l'organisation interne de notre watch.



1

Et effectivement Apple n'a pas déçu ses utilisateurs en proposant une nouvelle montre bien mieux finie. Premier changement : des applications natives ! Ce qui sous-entendait une refonte complète pour nous les développeurs et nos applications, et donc un gros travail comme vous allez pouvoir le constater.

Comme tout est exécuté sur la montre, le dialogue avec le téléphone n'est limité qu'aux transferts de données nécessaires. Pour les développeurs cela s'est traduit par l'incapacité de partager, par exemple, une base de données avec la montre, comme on pouvait alors le faire avec la première version de celle-ci. Mais pour l'utilisateur final ce fut un grand pas qui a permis de reposer un peu la batterie de la watch. Il faut aussi savoir qu'un gros travail sur la taille des composants internes avait été fait pour justement y placer une plus grande batterie. watch 2 devient également plus customisable avec de tous nouveaux cadrans disponibles et personnalisables directement par l'utilisateur. Deux nouvelles montres sont par ailleurs ajoutées à la gamme et plus de dix nouveaux bracelets !

Quoi de neuf chez la nouvelle ?

Elle est maintenant dotée de son propre GPS ! Et oui comme Apple souhaite que sa montre ne soit plus « fusionnée » au téléphone mais devienne un produit à part entière, de ce fait il fallait qu'elle puisse obtenir certaines données sans passer par l'iPhone.

Mais l'une des grandes nouveautés pour les développeurs est la possibilité de concevoir ses propres "complications" pour les rendre accessibles sur le cadran de la montre. Mais vous allez me dire : *Qu'est-ce qu'une "complication" ?* Cette fonctionnalité est tout simplement une partie du cadran réservée à une application qui pourra alors mettre à jour en temps réel une ou plusieurs informations (par exemple le taux d'humidité dans l'air, la température qu'il fait, etc.) [Fig.1]. Pour les développeurs Swift vous aurez d'ailleurs la possibilité de convertir votre code en Swift 2.3 ou même directement en

version 3 (recommandé par Apple). Cela impose quelques modifications, alors faites bien attention à bien faire une sauvegarde avant de vous lancer. Mais parlons plutôt des changements sur watchOS. Dorénavant les applications les plus utilisées sont automatiquement préchargées afin d'avoir une montre très réactive. Qui plus est les applications (qu'elles soient d'Apple ou de développeurs tiers) pourront tourner en tâche de fond. Bien entendu cette fonctionnalité risque de baisser la durée de vie de la batterie mais sans les tests, impossible d'en être sûr. Espérons simplement qu'il n'y aura pas de régression sur ce sujet et que l'on ne risque pas de se retrouver avec les mêmes problèmes qu'avec la watch 1.

De nombreux changements ont aussi été apportés à la logique de navigation. Je vous rappelle que pour Apple, la watch est le matériel le plus proche de l'utilisateur et de ce fait elle se doit d'être à la fois à l'image de celui ou celle qui la porte mais aussi doit pouvoir fournir le plus rapidement possible toutes les informations souhaitées en un minimum de temps. C'est donc dans cette optique que la navigation a été modifiée. Prenons le cas d'une liste de noms et prénoms représentant les intervenants à une conférence. Nous pouvons donc parcourir cette liste et cliquer sur les personnes qui nous intéressent, ce qui a pour conséquence d'ouvrir une nouvelle page avec en son sein un lot d'informations complémentaires. Mais si l'on souhaite ouvrir une autre fiche, nous devons faire un retour arrière, sélectionner le prochain intervenant pour de nouveau ouvrir la fiche. Avec les nouveautés watchOS3 il nous est possible directement d'aller à la prochaine fiche sans revenir à la liste globale avec un simple scroll de bas en haut avec le doigt ou via la molette présente sur le côté de la montre. Et les développeurs vont pouvoir se réjouir, nous avons accès à cette fonctionnalité, qui est très simple à mettre en œuvre. Une seule condition est à remplir : votre page détail (c'est comme cela que l'on appellera le contrôleur ouvert lors du clic sur une cellule) ne doit pas excéder la taille de l'écran. Vous comprenez donc que dans certains cas, cette fonctionnalité n'a pas de sens. Elle se prête uniquement à certains cas d'usage.

Passons à l'exemple

Pour notre exemple nous allons continuer avec notre liste de nom et prénom. Dans la page détails nous afficherons la photo de la personne, son nom et prénom et enfin sa profession. Notre design devra donc prendre en compte le fait que nous sommes restreints à une seule page. Maintenant comment passer d'une page à l'autre ? Je suis sûr que vous le savez déjà mais dans le doute voici un rapide rappel :

La méthode la plus simple est de réagir au délégué « didRowSelectAtIndex » de la WKInterfaceTable et d'exécuter l'instruction suivante : [Fig.2]

Cela nous permet notamment de passer au contexte du nouveau contrôleur un objet qui va nous permettre de customiser la page de détails. Dans notre cas l'objet représentant la personne sélectionnée. Mais avec cette méthode notre « vertical paging » ne fonctionnera pas. Et oui comment le système peut-il déterminer l'objet à donner à la prochaine page ? Eh bien il ne peut pas. Heureusement il existe une seconde méthode nous permettant de le faire. Pour cela il va nous falloir utiliser les « segue ». Donc dans un premier temps faite un « ctrl + click » sur la cellule concernée et glisser jusqu'à notre contrôleur représentant la page de détail. Vous aurez alors deux possibilités :

- Push
- Modal

Choisissez « Push » dans notre cas. Maintenant cliquez sur le segue. Vous verrez que le volet de droite s'est adapté à votre sélection. Et dans le quatrième onglet, à savoir le « Storyboard Segue » vous avez la possibilité de donner un nom à votre segue. Je vous propose de mettre « pushDetail ». De cette façon nous connaissons la méthode d'affichage « push » et la destination « Detail ». Bien entendu vous pouvez nommer comme bon vous semble. A partir de là vous avez un segue créé entre vos deux contrôleurs.

```
override func table(table: WKInterfaceTable, didSelectRowAtIndex rowIndex: Int) {
    if let row = datas[rowIndex] as? RowObject, let url = row.assocObj as? NSURL {
        self.pushControllerWithName("PhotoDetail", context: url)
    }
}
```

⚠ Conditional cast

2

```
override func contextForSegueWithIdentifier(segueIdentifier: String, inTable table: WKInterfaceTable, rowIndex:
    Int) -> AnyObject? {
    if segueIdentifier == "pushDetail" {
        if let row = datas[rowIndex] as? RowObject, let url = row.assocObj as? NSURL {
            return url
        }
    }
    return nil
}
```

3

Vous pouvez d'ailleurs tester et vous verrez que sans rien faire vous passerez d'une page à l'autre juste en cliquant sur la cellule (n'oubliez pas de supprimer l'autre méthode si vous l'avez mise en place)

Mais ce qui nous intéresse c'est de pouvoir passer des données entre les deux pages, dans notre cas la personne concernée. Bonne nouvelle ça reste très simple. Implémentez la méthode « contextForSegueWithIdentifier ». C'est elle qui sera appelée lors de l'exécution du segue. C'est donc ici que l'on va lui passer nos paramètres. Voici la méthode complète : [Fig.3]

Comme vous pouvez le voir nous testons le nom du segue. Dans le cas présent nous n'avons qu'une page ouvrable, mais nous pouvons imaginer que notre page principale gère un grand nombre de segue et donc potentiellement un grand nombre d'informations diverses et variées suivant la page appelée. Il faut donc le prévoir. Si le test est valide, nous récupérons l'objet qui nous intéresse, ici un « RowObject » contenant l'ensemble de nos informations. Maintenant dans notre page détail, il faut surcharger la méthode « awakeWithContext ».

C'est elle qui sera appelée lors de la création du contrôleur. Et c'est elle, via son contexte, qui va recevoir l'objet envoyé précédemment. A vous maintenant de décomposer l'objet et d'utiliser les informations qu'il contient pour customiser votre page détail.

D'autres nouveautés

Nous venons de voir l'une des possibilités de pagination que propose watchOS 3. Mais il en existe d'autres. Il y a eu notamment des changements dans leur disposition. Les informations sont désormais disposées de haut en bas et il faut scroller dans la page pour y avoir accès, ce qui permet de ne charger que les informations voulues et non pas comme avant ou l'on avait les pages les unes à côté des autres. Plus simplement, et toujours pour coller à la même utilisation qu'avec l'iPhone, un swipe de bas en haut permet d'ouvrir le "Control Center" afin d'avoir accès aux fonctionnalités de base sans devoir quitter notre application.

L'une des grandes nouveautés de l'Apple Watch est la possibilité de conserver vos dix applications préférées directement dans le dock. Mais qu'est-ce que le dock ? C'est une des nouveautés de watchOS 3, qui permet grâce au bouton latéral (avant permettant d'ouvrir vos contacts), d'afficher un menu permettant de naviguer entre toutes les applications

ouvertes. On retrouve encore une fois ici la même utilisation que sur un iPhone avec le bouton central. Par conséquent, comme celles-ci seront toujours stockées en mémoire elles pourront alors être mises à jour continuellement. Qui plus est cela permet un démarrage rapide de l'application. Kevin Lynch dit même que le lancement des apps sera « sept fois plus rapide, mais paraîtra un million de fois plus rapide », une promesse qui nous l'espérons sera tenue.

Du côté développeur il faut savoir que lorsque l'utilisateur se déplace dans le dock, il réactive les unes après les autres les applications mises en avant. Ce qui vous permettra d'updater vos informations même si l'utilisateur ne l'ouvre pas. Un bon moyen d'avoir constamment notre application à jour.

Mais l'Apple Watch est aussi bien plus ouverte aux développeurs. Nous avons dorénavant accès à bien plus d'informations et de fonctions comme notamment : gyroscope, accéléromètre, rythme cardiaque, la lecture de vidéos directement sur la montre, et des actions en lien avec la couronne digitale.

Vous vous rappelez de l'époque où il était impossible de pouvoir répondre à un message directement depuis sa montre ? Eh bien c'est fini. Tout comme sous Android Wear 2, l'Apple Watch permet de tracer les lettres du message. Bien entendu l'envoi d'emojis, les dessins et les pulsations cardiaques sont toujours disponibles. Mais il ne faut pas oublier que la fonction première de l'Apple Watch, mise à part celle de donner l'heure, est de tracker l'activité sportive de l'utilisateur.

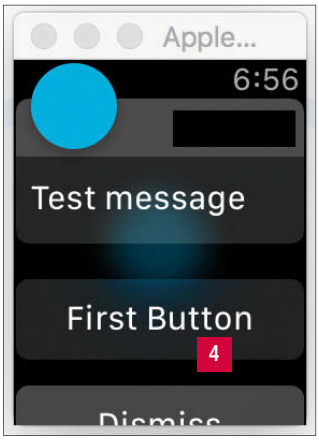
Et comme de nos jours le partage est omniprésent, Apple a ajouté la possibilité de partager les résultats de nos efforts sportifs avec nos amis. Il nous est alors possible de lancer une compétition entre amis ou de s'entraider pour progresser. De nouveaux exercices sont d'ailleurs disponibles pour les sportifs. Mais pas que. Une application nommée « Breathe » vous apprendra à respirer pour vous détendre. Elle est basée sur la respiration profonde et la méditation. Même si certaines personnes peuvent penser qu'il s'agit de pratiques idiotes, sur d'autres ça fonctionne vraiment. Des problèmes créés par le stress comme l'anxiété et la dépression peuvent être soulagés par cette pratique.

L'ayant personnellement testé via le simulateur de la watchOS 3 je vous le dis, ça fonctionne. Il ne vous reste plus qu'à tenter l'expérience. Il faut aussi voir cette application comme un moyen de souffler un peu dans notre quotidien qui en a bien besoin.

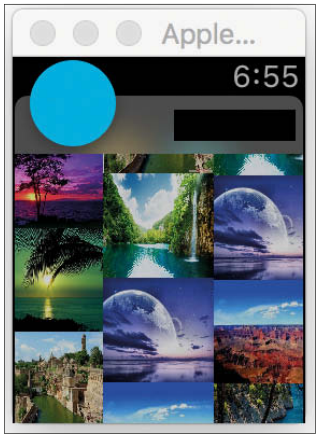
Pour la plupart des personnes, lorsque l'on pense au sport, on pense immédiatement à courir. Mais malheureusement pour certains, notamment les personnes en fauteuils roulant, c'est quelque chose d'impossible. Par conséquent Apple a introduit dans la watch la possibilité de détecter les mouvements des bras, par exemple pour faire avancer le fauteuil roulant. Un grand pas en avant, que les concurrents d'Apple n'avaient pour l'instant pas prévu. Mais avec l'arrivée de son « HomeKit », Apple souhaite aussi y faire participer sa montre. C'est pourquoi avec la dernière version de la watchOS il vous sera possible de contrôler les objets connectés de votre maison et bénéficier de leurs fonctionnalités.

Par exemple lors de sa présentation, Apple montrait le cas où quelqu'un sonne à la porte. Votre Apple Watch vous permettrait de voir directement qui est là via la caméra installée dans votre entrée et reliée à HomeKit.

De cette façon on aurait alors toujours à notre poignet la possibilité de contrôler notre maison. Plutôt pas mal !



AVANT



APRES

SpriteKit et SceneKit

Et ce n'est pas tout. Une autre nouveauté, qui devrait ravir les concepteurs de jeux vidéo et d'animations, est de pouvoir se faire plaisir puisque dorénavant « SpriteKit » et « SceneKit » sont disponibles pour Apple Watch. Bien entendu on reste sur une montre, il faut donc faire très attention à ne pas en abuser si on souhaite préserver la batterie. Mais cela ouvre la possibilité à la création d'animations de haute qualité avec facilités pour améliorer l'interface utilisateur de nos applications. Alors voyons ce que l'on peut faire avec tout ça. Je vous propose de concevoir une notification un peu plus classe que la notification de base sous watchOS. [Fig.4]

Avouez que c'est bien plus joli maintenant ! Comme vous vous en doutez, dans la version « Après », il s'agit d'une SKScene provenant de SpriteKit. Vous ne pouvez pas encore le voir mais l'ensemble des images sont animées. Imaginez, la colonne de droite et de gauche vont vers le bas, tandis que la colonne du milieu remonte. Nous allons donc voir comment faire cela. Pour cet exemple nous prendrons une application de gestion de photo distante. L'ensemble des URL étant stocké dans un fichier JSON. Dans un premier temps, il nous faut un « WKUserNotificationInterfaceController » qui se chargera d'embedder notre scène. Normalement, tout du moins si votre projet est tout neuf, vous devriez avoir dans votre storyboard un contrôleur représentant votre notification statique. Mais nous allons lui ajouter une version dynamique. Pour cela sélectionnez ce contrôleur et cochez « Has Dynamic Interface » et là, magie, un tout nouveau contrôleur est apparu. Dans celui-ci, placez une « SpriteKit Scene ». Liez-la via un « IBOutlet » à votre code behind. Mais jusque-là, nous n'avions aucun moyen de dire à notre watch, d'afficher la notification avec l'animation plutôt que la version classique. Pour ce faire nous allons devoir implémenter 2 nouvelles méthodes dans notre contrôleur chargé de gérer les notifications : [Fig.5]

Ne vous alarmez pas si votre compilateur ne reconnaît pas la méthode « didReceive Notification ». C'est tout à fait normal en vérité, puisqu'il s'agit d'une fonction que nous allons créer. Comme vous l'aurez compris, peu impor-

```
override func didReceiveLocalNotification(localNotification: UILocalNotification, withCompletion completionHandler: ((WKUserNotificationInterfaceType) -
    > Void)) {
    didReceiveNotification(completionHandler)
}

override func didReceiveRemoteNotification(remoteNotification: [NSObject : AnyObject], withCompletion completionHandler:
    ((WKUserNotificationInterfaceType) -> Void)) {
    didReceiveNotification(completionHandler)
}
```
5

te d'où provient la notification (externe ou interne) nous utiliserons notre méthode. Et la voici : [Fig.6]

Mais que faisons-nous ? Tout d'abord, je vais lire dans mon fichier JSON. S'il y a du contenu j'appelle le bloc de completion avec en paramètre le « .Custom » qui dira au système d'utiliser notre contrôleur de notification avec animation et dans le cas contraire, je renvoie « .Default ».

Passons maintenant au paramétrage de notre scène. Pour cela rendez-vous dans la méthode « init » de notre contrôleur de notifications. [Fig.7]

Ici nous créons notre scène. Je lui affecte ensuite une taille correspondant à la taille de l'écran qui m'intéresse. Et surtout je la passe en mode « AspectFill ».

Il ne vous reste plus ensuite qu'à appeler cette scène dans la « WKInterfaceScene » que vous avez reliée plutôt à votre contrôleur.

Vous pouvez également modifier le nombre de frames par seconde pour votre scène.

Si vous testez, vous verrez que maintenant vous voyez la scène et non pas le texte d'origine. Alors certes notre scène ne fait pas grand-chose pour le moment. Mais nous allons remédier à cela tout de suite.

Dans un premier temps, nous allons définir un lot de variables qui vont nous permettre de concevoir notre animation. [Fig.8]

Je pense que le nom des variables est assez explicite, vous aurez compris l'utilité de chacune d'entre elles.

Maintenant il faut que l'on récupère l'ensemble des URL stockées dans notre fichier JSON. Pour cela, nous allons dans la méthode « sceneDidLoad » ouvrir notre fichier JSON et récupérer nos données. Pour cela nous allons utiliser la même méthode maison (je vous laisserai créer la vôtre) : [Fig.9]

Pour chaque URL trouvée et valide, on remplit notre tableau. Ce qui nous permettra de télécharger nos images et de les afficher.

Et quand tout est fini, on appelle la méthode chargée de la construction de notre tableau d'images. C'est donc cette méthode qui va calculer la taille de chaque image, leur position et leur mouvement. On va donc pour cela créer une boucle qui va se charger de le faire. Mais avant ça, il nous faut calculer toutes les variables nécessaires : [Fig.10]

On connaît donc maintenant la taille de chacune de nos images (hauteur et largeur). Cependant le nombre d'images en mémoire n'est pas nécessairement suffisant pour créer toutes nos cellules. Nous allons faire une méthode qui sera chargée de boucler sur les URL du moment que nous en avons besoin, ainsi, que nous ayons 2 ou 200 images, notre animation pourra toujours se réaliser. [Fig.11]

Maintenant créons notre tableau : [Fig.12]

Donc jusqu'ici rien de spécial. Mais c'est ici que nous allons voir comment créer toutes nos cellules ou plutôt nos « SKSpriteNode ». Un Node ou nœud en français est un élément graphique ou non, qui pourra interagir au sein d'une

```
func didReceiveNotification(completionHandler: ((WKUserNotificationInterfaceType) -> Void)) {
    FileManager.readJSONFile(defaultFilename) { (content) in
        if let images = content["images"] as? [NSDictionary] where images.count > 0 {
            completionHandler(.Custom)
        }
        else {
            completionHandler(.Default)
        }
    }
}
```
6

```
override init() {

    super.init()

    let currentDevice = WKInterfaceDevice.currentDevice()
    let bounds = currentDevice.screenBounds

    let scene = NotifScene(size: CGSize(width: bounds.width, height: bounds.height))

    scene.size = CGSize(width: bounds.width, height: bounds.height * (80 / 100))
    scene.scaleMode = .AspectFill

    self.skInterface?.presentScene(scene)
    self.skInterface?.preferredFramesPerSecond = 60

}
```
7

```
let NUMBER_OF_COLUMN: Int = 3
let NUMBER_OF_ROW_PER_COLUMN: Int = 4

let ITEM_HORIZONTAL_MARGIN: CGFloat = 0
let ITEM_VERTCIAL_MARGIN: CGFloat = 0
let ITEM_MOVE: CGFloat = 0.05

var heightForItem: CGFloat = 0

var indexOfImage: Int = 0
var urlOfImages: [NSURL] = []
```
8

```
func nextImage() -> NSURL {
    if (urlOfImages.count - 1) < indexOfImage {
        indexOfImage = 0
    }
    let result = urlOfImages[indexOfImage]
    indexOfImage += 1
    return result
}
```
11

```
override func sceneDidLoad() {

    super.sceneDidLoad()
    self.removeAllChildren()

    FileManager.readJSONFile(defaultFilename) { (content) in

        if let images = content["images"] as? [NSDictionary] where images.count > 0 {
            images.forEach({ (imgDict) in
                if let urlString = imgDict["Url"] as? String, let url = NSURL(string: urlString) {
                    self.urlOfImages.append(url)
                }
            })
        }

        self.constructNodes()

    }

}
```
9

```
func constructNodes() {

    let floatColumn = CGFloat(integerLiteral: NUMBER_OF_COLUMN)
    let floatRow = CGFloat(integerLiteral: NUMBER_OF_ROW_PER_COLUMN)

    let widthForItem = (self.size.width - (floatColumn * ITEM_HORIZONTAL_MARGIN)) / floatColumn
    heightForItem = (self.size.height - (floatRow * ITEM_VERTCIAL_MARGIN)) / floatRow

    var lastX: CGFloat = self.anchorPoint.x
    var lastY: CGFloat = self.anchorPoint.y
```
10

```
for column in 1...NUMBER_OF_COLUMN {

    for row in 1...NUMBER_OF_ROW_PER_COLUMN + 2 {

        let size = CGSize(width: widthForItem, height: heightForItem)

        if let newRect = KSSpriteNode.fromUrl(nextImage()) {

            newRect.name = "rect_\(row)x\(column)"
            newRect.aspectFillToSize(size)
            newRect.size = size

            if column == 2 {
                newRect.isMiddleColumn = true
            }

            let newXPosition = lastX + ITEM_HORIZONTAL_MARGIN + (widthForItem / 2)
            let newYPosition = lastY + ITEM_VERTCIAL_MARGIN + (newRect.height / 2)

            newRect.position = CGPoint(x: newXPosition, y: newYPosition)

            lastY += ITEM_VERTCIAL_MARGIN + newRect.height

            if row == NUMBER_OF_ROW_PER_COLUMN + 2 {
                lastX += widthForItem
                lastY = self.anchorPoint.y - (newRect.height / 2)
            }

            self.addChild(newRect)

        }

    }

}
```
12

```
class KSSpriteNode : SKSpriteNode {

    var isMiddleColumn: Bool = false

    class func fromUrl(urlOfImage: NSURL) -> KSSpriteNode? {
        if let imgData = NSData(contentsOfURL: urlOfImage), let img = UIImage(data: imgData) {
            return KSSpriteNode(texture: SKTexture(image: img))
        }
        return nil
    }

    func aspectFillToSize(fillSize: CGSize) {
        if texture != nil {
            self.size = texture!.size()
            let verticalRatio = fillSize.height / self.texture!.size().height
            let horizontalRatio = fillSize.width /  self.texture!.size().width
            let scaleRatio = horizontalRatio > verticalRatio ? horizontalRatio : verticalRatio
            self.setScale(scaleRatio)
        }
    }

}
```
13

scène (ex : se déplacer, émettre des particules, avoir une image, …).
Comme dans notre cas nous avons besoin de posséder des informations et méthodes annexes pour le bon fonctionnement de notre app, j'ai créé une classe qui hérite de « SKSpriteNode ». [Fig.13]
De cette façon je peux savoir si le nœud en question fait partie de la colonne centrale et il possède toutes les méthodes nécessaires à son instanciation via une URL ou l'ajustement de sa propre texture.
Je vous laisse tester. Et oui notre tableau d'image est bel est bien là ! Mais oups ! Il n'y a aucune animation. Tout est statique. On va arranger ça. Dans chaque scène il y a une méthode « update » qui est une sorte de boucle infinie nous permettant d'updater la scène à intervalle régulier : [Fig.14]
C'est donc dans celle-ci que l'on va prendre chacun des nœuds et les déplacer en conséquence. C'est ici que la variable « isMiddleColumn » nous servira pour connaître le mouvement de l'image. [Fig.15]
Si vous le faites en même temps que moi, vous allez très vite remarquer que certaines propriétés n'existent pas. C'est encore de ma faute. Voici une série d'extensions qui devraient régler ce problème : [Fig.16]
Et voilà ! Votre animation est prête et vous pouvez désormais l'utiliser comme une notification enrichie sur votre Apple Watch ! Mais aussi sur votre iPhone. Je vous rappelle que SpriteKit et SceneKit sont également disponible sur iOS.

Sécurité et Apple Pay

Tout ça c'est très bien, mais de nos jours ce qui préoccupe beaucoup de personnes c'est la sécurité. Tout doit être sécurisé : nos comptes, nos maisons, nos voitures et bien entendu nous-même. C'est donc pour cela qu'Apple à rajouté une fonction d'appel d'urgence qui, en fonction du pays dans lequel vous vous trouvez, appellera les urgences. Il est aussi possible d'y ajouter un message automatique destiné à un proche. Et pour déclencher cette fonctionnalité rien de plus simple : laissez appuyé pendant plusieurs secondes le bouton latéral de votre montre et le tour est joué ! En parlant de sécurité, Apple ajoute aussi à sa montre la possibilité de payer avec « Apple Pay ». Un bon moyen de ne plus sortir son téléphone.

Un dernier tour de magie

Et pour finir un petit tour de magie qui devrait vous plaire. Toute personne disposant d'une Apple Watch 2 avec macSierra installé sur son Macintosh pourra automatiquement le déverrouiller lorsque sa montre (et donc vous-même en théorie) sera suffisamment proche ! Génial n'est-ce pas ?
Si cet article vous a plus et que vous souhaitez en apprendre encore plus, je vous invite à découvrir les dernières technologies d'Apple sur le site de la WWDC: https://developer.apple.com/wwdc/
Vous y trouverez toutes les vidéos de présentation de nouveautés d'Apple, qu'elles soient matérielles ou logicielles. •

```
override func update(currentTime: NSTimeInterval) {
    super.update(currentTime)
    for node in self.children {
        if let ksNode = node as? KSSpriteNode {
            infiniteScrollY(ksNode)
        }
    }
}
```
14

```
extension SKNode {

    var x: CGFloat {
        return self.position.x
    }

    var y: CGFloat {
        return self.position.y
    }

    var height: CGFloat {
        return self.frame.size.height
    }

    var width:  CGFloat {
        return self.frame.size.width
    }
}
```
16

```
func infiniteScrollY(node: KSSpriteNode) {
    if node.isMiddleColumn {
        node.position = CGPoint(x: node.x, y: node.y + ITEM_MOVE)
        if node.y > self.anchorPoint.y + self.size.height + heightForItem {
            node.position = CGPoint(x: node.x, y: self.anchorPoint.y - heightForItem)
        }
    }
    else {
        node.position = CGPoint(x: node.x, y: node.y - ITEM_MOVE)
        if node.y < -heightForItem {
            node.position = CGPoint(x: node.x, y: self.anchorPoint.y + self.size.height + heightForItem)
        }
    }
}
```
15

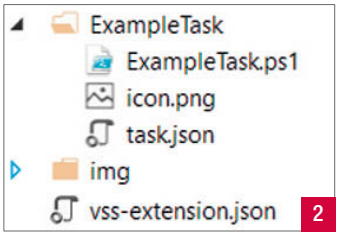
Etendre VSTS avec les extensions 2ePartie

Mikaël Krief
Microsoft MVP
Visual Studio ALM Rangers
Consultant ALM chez **Cellenza**
@MikaelKrief

Visual Studio Team Services (VSTS) et Team Foundation Server sont ouverts à l'extensibilité de leurs fonctionnalités en rajoutant des nouvelles pages qui vont comporter vos besoins métiers. Il est aussi possible de créer une tâche de build qui va permettre d'exécuter des traitements personnalisés lors du packaging ou déploiement de vos applications.
Les extensions offrent aussi la possibilité de créer un widget personnalisé afin de présenter rapidement des informations sur le Dashboard.

Créer une tâche de build (ou de Release) sous forme d'extension

Lorsque l'on crée une définition de build dans VSTS, on ajoute des tâches qui vont exécuter les traitements désirés lors de l'exécution de celle-ci.
VSTS fournit par défaut un large panel de tâches de build qui vont permettre des traitements de différentes natures. [Fig.1]
Grâce aux extensions VSTS, il est aussi possible de créer une tâche de build qui va comporter votre traitement personnalisé lors de l'exécution de celle-ci. En réalité, créer une tâche de build peut se faire sans passer par une extension, en créant la tâche et en l'uploadant dans VSTS directement, un des avantages de passer par une extension est de pouvoir la partager via le Visual Studio Marketplace.
Les éléments essentiels au bon fonctionnement de la tâche sont : le scripts PowerShell ou JavaScript à exécuter, une icône pour le catalogue des tâches et un fichier Json pour sa configuration.

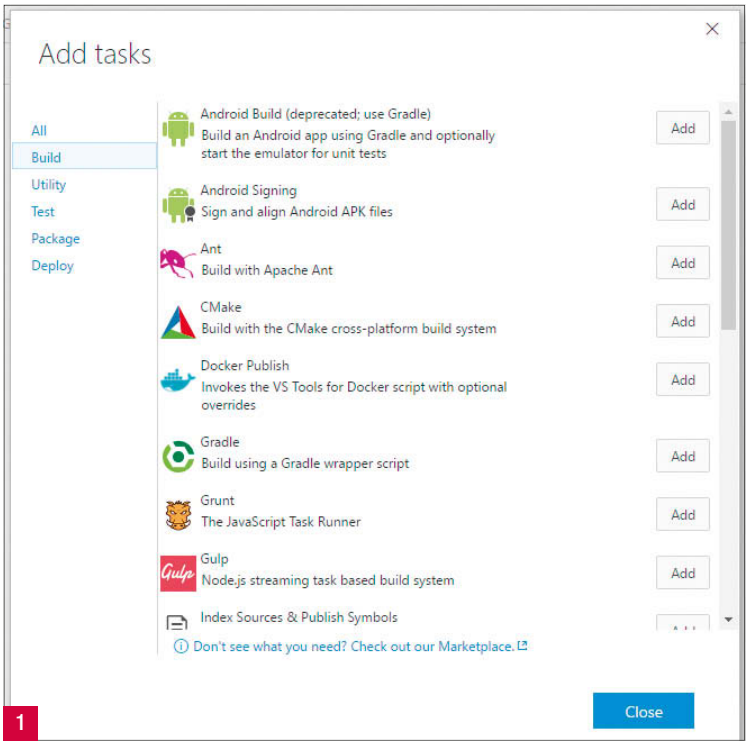


2

Pour plus de détails sur la création d'une tache de build, vous pouvez vous référez à cet article de blog http://bit.ly/1lmfech. [Fig.2]
Pour que cette tâche puisse être intégrée dans une extension, on rajoute à la racine du projet le fichier de configuration vss-extension.json. Dans celui-ci la section qui change par rapport à une extension de page c'est la section contribution.

```
"contributions": [
  {
    "id": "example-task",
    "targets": [
      "ms.vss-distributed-task.tasks"
    ],
    "type": "ms.vss-distributed-task.task",
    "properties": {
      "name": "Example Task"
    }
  }
]
```

Il est bien sûr possible de mettre plusieurs tâches de Build dans la même extension. Dans le Marketplace, il existe de nombreuses extensions portant sur des tâches de build, certaines sont fournies par Microsoft,



1

d'autres pas des éditeurs de logiciels tiers comme RedGate et d'autres par des contributeurs libres. La documentation sur la création d'une tâche de build en extension est disponible sur le site officiel http://bit.ly/29cgxJW .

Créer votre widget pour le Dashboard

Avec les extensions il est aussi possible de créer et de partager un widget personnalisé pour le Dashboard.
Un widget se compose :

- D'une page html de présentation ;
- D'un script JavaScript ou Typescript pour son traitement ;
- Et optionnellement un panel de configuration qui lui aussi se compose d'une page Html et d'un script JavaScript (ou Typescript).

Comme pour toutes les extensions, c'est dans le fichier de configuration que l'on indique son type, ses tailles disponibles, ainsi qu'une section pour sa page de configuration si ce widget en possède une.

```
"contributions": [
  {
```

```
    "id": "MonWidget",
    "type": "ms.vss-dashboards-web.widget",
    "targets": [
      "ms.vss-dashboards-web.widget-catalog",
      .MonWidget-Configuration"
    ],
    "properties": {
      "name": "Mon Widget",
      "description": "Mon Widget",
      "uri": "index.html",
      "previewImageUrl": "images/preview.png",
      "supportedSizes": [
        {
          "rowSpan": 1,
          "columnSpan": 3
        },
        {
          "rowSpan": 2,
          "columnSpan": 3
        }
      ],
      "supportedScopes": [ "project_team" ]
    }
  },
  {
    "id": "MonWidget-Configuration",
    "type": "ms.vss-dashboards-web.widget-configuration",
    "targets": [ "ms.vss-dashboards-web.widget-configuration" ],
    "properties": {
      "name": "Widget Configuration",
      "description": "Configure Widget",
      "uri": "widgetconfiguration.html"
    }
  }
]
```

Cet exemple de manifeste décrit la page du widget mais aussi sa page de configuration dans deux contributions distinctes.
Le lien entre le widget et sa page de configuration se situe dans la section *Target* du widget en précisant l'*Id* de la contribution pour sa configuration (dans notre exemple il s'agit de l'id : *MonWidget-Configuration*).
La section *SupportedSizes* permet de préciser les tailles disponibles pour le widget. Dans son panel de configuration une liste déroulante apparaîtra avec cette liste des tailles.
Attention toutefois, cette liste déroulante ne sera présente que si le widget a un panel de configuration qui a été paramétré dans le fichier de manifeste.

La page html

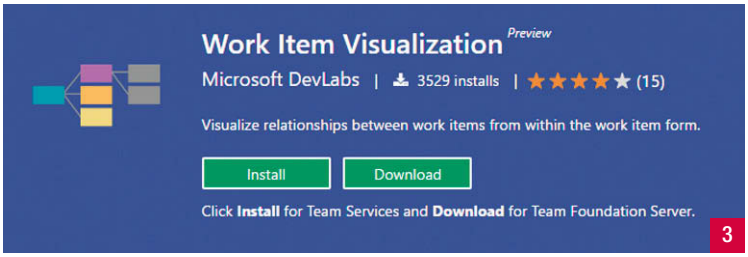
Le contenu fichier HTML est identique à celui d'une extension basic, il contient le code HTML de présentation, l'inclusion et l'appel au SDK et l'inclusion au script JavaScript qui va effectuer les traitements.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <link rel="stylesheet" href="css/app.css" type="text/css" />
</head>
<body>
<div class="widget">
<h2 class="title">Widget title</h2>
<div class="description">Description</div>
<p>Contenu du widget</p>
</div>
  <script src="scripts/VSS.SDK.js"></script>
  <script type="text/javascript">
    // Initialize the VSS sdk
    VSS.init({
      explicitNotifyLoaded: true,
      setupModuleLoader: true,
      usePlatformScripts: true,
      usePlatformStyles: true,
      moduleLoaderConfig: {
        paths: {
          "Scripts": "scripts"
        }
      }
    });

    // Wait for the SDK to be initialized
    VSS.ready(function () {
      require(["Scripts/main"],
      function (app) { });
    });
  </script>
</body>
```

Le scripts Javascripts (écrit en TypeScript) : Main.ts, est le script principal

```
VSS.require("TFS/Dashboards/WidgetHelpers", function (WidgetHelpers) {
        WidgetHelpers.IncludeWidgetStyles();
        VSS.register("HelloWorldWidget", function () {
          return {
            load: function (widgetSettings) {

              //Votre code
              var $title = $('h2.title');
              $title.text('Hello World');

              return WidgetHelpers.WidgetStatusHelper.Success();
            },
            reload: function (widgetSettings) {
              //Votre code pour le rechargement
            }
          }
        });
        VSS.notifyLoadSucceeded();
});
```

L'instruction *WidgetHelpers.IncludeWidgetStyles();* permet d'appliquer les styles Css standards VSTS au widget.
Les instructions pour le style des widgets sont détaillées ici http://bit.ly/29rJqr2 .



Toute la procédure pour développer un widget se trouve dans les liens situés en fin d'article.

Et pour TFS 2015 on-premise ?

Comme prévu initialement, Team Foundation Server (on-premise) possède désormais la possibilité d'avoir des extensions. Cette fonctionnalité est disponible depuis TFS 2015 Update 2, sortie fin mars.

Beaucoup d'extensions disponibles pour VSTS via le Visual Studio Marketplace sont aussi disponibles pour TFS 2015.

Les extensions compatibles pour TFS 2015 sont identifiables grâce au bouton *Download* qui se trouve sur la fiche de celle-ci.

La procédure d'installation d'une extension et un peu différente entre VSTS et TFS : pour une installation d'une extension sur un compte VSTS il suffit de cliquer sur *Install*, alors que pour TFS 2015, il faut télécharger l'extension puis uploader ce package vsix dans le gestionnaire d'extensions de TFS 2015.

Pour la procédure d'installation d'extension sur TFS 2015 vous pouvez lire cet article http://bit.ly/1SA3mCF .

Développer une extension pour TFS 2015 on-premise

Le développement d'une extension pour TFS 2015 on-premise est identique à celui d'une extension pour VSTS, il faut tout de même faire attention à la documentation des APIs Rest qui est plus à jour par rapport aux nouvelles fonctionnalités de VSTS. Pour configurer la cible d'une extension, il faut modifier le paramétrage du fichier *vss-extension.json*, avec le paramètre *Targets* qui peut prendre les valeurs :

- Microsoft.VisualStudio.Services, pour VSTS et TFS 2015 ;
- Microsoft.VisualStudio.Services.Cloud, pour uniquement VSTS ;

CONCLUSION

Les extensions de VSTS et TFS permettent d'intégrer vos besoins métiers que ce soit avec des nouvelles rubriques intégrées, vos tâches de build pour exécuter vos traitements ainsi que vos widgets pour le Dashboard dans l'usine logicielle de Microsoft ; là aussi, que ce soit dans sa version on-premise (TFS) ou dans sa version dans le Cloud (VSTS). •

Les liens

Documentation officielle :
https://www.visualstudio.com/docs/integrate/extensions/overview
Guide complet sur la création des widgets :
Documentation officielle sur les Widgets : http://bit.ly/29lKlaK
Démarrer avec les Widgets : http://bit.ly/29k6Sna
Développer un Widget : http://bit.ly/29rlzXm
Publier un Widget : http://bit.ly/29fgEVP

Découverte du **Data Binding** sous Android

Sylvain SAUREL
Ingénieur d'Etudes Java / JEE
sylvain.saurel@gmail.com

Peu de développeurs le savent mais le SDK Android propose un support du Data Binding depuis maintenant près de 2 ans. Ce support permet d'écrire des layouts de manière déclarative en minimisant le code Java nécessaire pour connecter les éléments de l'interface utilisateur à la partie logique d'une application. Dans cet article, nous vous proposons de découvrir le Data Binding made in Android.

Bien connu des développeurs de la plateforme DotNet de Microsoft, le Data Binding permet de simplifier le travail des développeurs lors de la réalisation d'applications en réduisant le code Java à écrire pour connecter les éléments de l'interface utilisateur à la partie logique d'une application. La figure 1 montre clairement le fonctionnement du Data Binding.

Configuration

Proposé au sein de la bibliothèque de support App Compat d'Android, le Data Binding est rétro compatible jusqu'à Android 2.1 soit l'API Level 7. Pour l'utiliser au sein d'une application, il sera également nécessaire d'utiliser le plugin Android pour Gradle en version 1.5.0 au minimum. Au niveau du fichier build.gradle d'une application, il est nécessaire d'activer le Data Binding comme suit :

```
android {
   ....
   dataBinding {
      enabled = true
   }
}
```

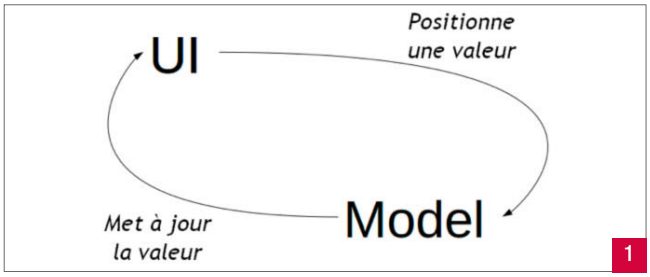
Si le module principal de votre application dépend d'une bibliothèque utilisant le Data Binding, il sera nécessaire également d'activer cette option, et ce, même si votre module n'utilise pas le Data Binding lui-même. Au niveau IDE, pas de problème particulier puisque Android Studio propose un support du Data Binding depuis la version 1.3 alors que nous sommes déjà en version 2.1.2 actuellement.

Modèle Java

Avant de pouvoir utiliser le Data Binding au sein d'un fichier layout, il est nécessaire d'avoir défini son modèle au sein du code Java. Ici, nous allons considérer un objet User assez simple contenant un prénom et un nom :

```
public class User {

   public final String firstName;
   public final String lastName;

   public User(String firstName, String lastName) {
      this.firstName = firstName;
      this.lastName = lastName;
   }
}
```

Au sein de ce POJO classique dans le monde Java, vous aurez noté que les membres de la classe ont été mis en public afin de ne pas avoir à définir de getters / setters. Si vous préférez définir les membres de la classe en accès private, il sera alors nécessaire de définir des getters / setters comme de coutume et la classe User aura l'allure suivante :



1

Fonctionnement du Data Binding

```
public class User {

   private final String firstName;
   private final String lastName;

   public User(String firstName, String lastName) {
      this.firstName = firstName;
      this.lastName = lastName;
   }

   public String getFirstName() {
      return this.firstName;
   }

   public String getLastName() {
      return this.lastName;
   }
}
```

Binding côté layout

Le modèle défini, nous pouvons l'utiliser au sein d'un fichier layout. Attention, un fichier layout recourant au Data Binding est quelque peu différent d'un fichier layout habituel puisqu'il est nécessaire de définir un élément layout comme racine suivi d'un élément data et enfin d'un élément de type View, qui constituera la racine de l'interface utilisateur, défini au sein du fichier layout. Au sein de l'élément data, il faudra rajouter les objets pouvant être utilisés dans le cadre du Data Binding via l'utilisation d'éléments de type variable. En reprenant notre objet User, notre layout pourra avoir l'allure suivante :

```
<?xml version="1.0" encoding="utf-8"?>
<layout xmlns:android="http://schemas.android.com/apk/res/android">
   <data>
```

```
    <variable name="user" type="com.ssaurel.User"/>
  </data>
  <LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@{user.firstName}"/>
    <TextView android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@{user.lastName}"/>
  </LinearLayout>
</layout>
```

Au sein du layout, les expressions de Data Binding sont écrites via l'emploi de @{}. Cela s'applique à nos deux éléments TextView avec par exemple pour l'affichage du prénom la syntaxe suivante :

```
@{user.firstName}
```

Binding côté Java

Au niveau Java, le plugin Android pour Gradle se charge de générer une classe de Binding. Pour son nommage, il reprend le nom du fichier layout qu'il convertit suivant la notation Pascal puis le suffixe avec le mot "Binding". Le layout présenté précédemment se nommant activity_main.xml, une classe MainActivityBinding sera générée. Cette dernière permet de faire le lien entre les propriétés du layout, telles la variable user, et les composants visuels définis tout en se chargeant d'effectuer les affectations de valeurs pour les expressions de binding définies. Le binding côté Java est réalisé habituellement au sein de la méthode onCreate de l'activité comme suit :

```
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  MainActivityBinding binding = DataBindingUtil.setContentView(this, R.layout.main_activity);
  User user = new User("Test", "User");
  binding.setUser(user);
}
```

Appel de méthodes

Les évènements sur l'interface utilisateur peuvent être liés directement depuis le layout en utilisant la propriété android:onClick des éléments visuels. Alors qu'Android propose l'attribut onClick sur les éléments de type View depuis la création de l'OS, cette nouvelle méthode présente un avantage majeur puisque l'expression définie en son sein est traitée au moment de la compilation. En cas d'erreur au niveau de l'expression, la compilation se terminera en erreur. Avec l'attribut onClick, une telle erreur n'apparaît qu'à l'exécution, ce qui est bien plus gênant.
Pour associer un évènement utilisateur à un handler spécifique, il faut tout d'abord définir un handler comme suit :

```
public class MyHandler {
  public void onClick(View v) { ... }
}
```

Au sein du fichier layout, il reste ensuite à définir l'expression de binding utilisant ce Handler après l'avoir défini en tant que variable :

```
<?xml version="1.0" encoding="utf-8"?>
<layout xmlns:android="http://schemas.android.com/apk/res/android">
  <data>
    <variable name="handler" type="com.ssaurel.MyHandler"/>
    <variable name="user" type="com.ssaurel.User"/>
  </data>
  <LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@{user.firstName}"
      android:onClick="@{handler::onClick}"/>
  </LinearLayout>
</layout>
```

Binding d'évènements

Apparu avec la version 2.0 du plugin Android pour Gradle, le binding d'évènements est une solution alternative aux Appels de méthodes comportant une différence majeure. En effet, le handler utilisé comme listener lors de l'appel d'une méthode est créé au moment du binding des données alors que dans le binding d'évènements, l'expression utilisant le listener n'est évaluée que lors du déclenchement de l'évènement. Autre différence notable dans le cas de l'appel à une méthode, les paramètres de la méthode doivent correspondre aux paramètres du listener d'évènement. Avec le binding d'évènements, seule la valeur retournée doit correspondre à la valeur de retour attendue du listener. Il est ainsi possible de définir une classe Presenter avec une méthode onClick comme suit :

```
public class Presenter {
  public void onClick(Task task){ ... }
}
Puis de l'utiliser comme suit au sein du fichier layout :
<?xml version="1.0" encoding="utf-8"?>
 <layout xmlns:android="http://schemas.android.com/apk/res/android">
   <data>
     <variable name="task" type="com.ssaurel.Task" />
     <variable name="presenter" type="com.ssaurel.Presenter" />
   </data>
   <LinearLayout android:layout_width="match_parent" android:layout_height="match_parent">
     <Button android:layout_width="wrap_content" android:layout_height="wrap_content"
     android:onClick="@{() -> presenter.onClick(task)}" />
   </LinearLayout>
 </layout>
```

Comme vous l'aurez sûrement remarqué, les listeners sont définis via des Lambdas expressions acceptées uniquement comme élément racine d'une expression. La bibliothèque de Data Binding se chargeant d'effectuer les différents liens nécessaires à la bonne exécution côte Java lors du déclenchement d'un évènement.

CONCLUSION

Encore méconnu d'une partie des développeurs Android, le Data Binding proposé nativement par la plateforme est un mécanisme puissant qui offre un gain de productivité notable aux développeurs. En outre, il permet de réaliser des applications de meilleure qualité en favorisant une meilleure séparation des différentes couches applicatives. Cette découverte du Data Binding aura permis de mettre en avant le potentiel de cette fonctionnalité qui, à n'en pas douter, fera partie intégrante de votre quotidien de développeur Android dès que vous l'aurez essayée. •

Créer des sites Internet responsive avec **Drupal** et **Zurb Foundation**

1ère partie

• Thomas Lepérou

La création de sites Internet, pour certains de nos lecteurs, c'est une affaire de plusieurs décennies. Mais aujourd'hui les évolutions engendrées par le responsive - et plus globalement le cross-plateforme - ont radicalement changé la manière de les créer : material design, frontend-development, headless CMS, Sass, Grunt, Bower…

Au-delà de ces technologies c'est toute la conception qui évolue. Le mobile first, c'est porter une réflexion sur l'usage qui sera fait de l'interface pour concevoir votre site Internet, depuis le mobile - et maintenant les smartwatch - vers le desktop. Du plus petit écran, au plus grand. C'est aussi penser à la disposition des blocs selon les supports, les off-canvas, les breakpoints et à d'autres aspects qui sont abordés par la suite.

Pour commencer, rappel des dix commandements du site builder :

- De site Internet sans framework backend ni CMS tu ne créeras ;
- La ligne de commande tu favoriseras ;
- D'interface graphique sans framework frontend tu ne concevras ;
- Du UX tu te préoccuperas ;
- Du material design tu t'inspireras ;
- Des gestionnaires de librairies, accro tu deviendras ;
- Dans Stackoverflow, de l'aide tu obtiendras ;
- À Github tu contribueras ;
- De l'OS et au Web server tu t'intéresseras ;
- Et à l'anglais tu te mettras.

L'article s'articule principalement autour de deux technologies : Drupal 7 et Zurb Foundation 5. Un ensemble technologique efficace qui assurera à vos projets Web une qualité professionnelle. Elles couvrent de nombreux aspects que nous allons voir ensemble.

Drupal, 7

Drupal est un puissant CMS écrit en PHP. Il permet de récolter, traiter et stocker des données en ligne. Il propose une API robuste qui lui confère une flexibilité et une modularité appréciées des professionnels. Voyages-sncf, McDonalds, le réseau FranceTV en sont des exemples probants. Si ce n'est pas par son interface d'administration qu'il tire son succès, c'est en revanche pour sa flexibilité. Ceci notamment par la conception des modèles de données, des formats d'entrées et de sorties, des d'accès et également pour le développement de module custom afin de répondre aux besoins les plus spécifiques. http://www.drupal.org.

Zurb Foundation, 5

Zurb Foundation est un framework front-end. Il met à disposition tout un ensemble de classes CSS, de composants HTML, d'utilitaires Javascript et Sass. Il apporte à la fois un cadre de développement et la liberté de personnaliser librement votre interface : navigation, structure de contenus, boutons … http://foundation.zurb.com/sites/docs/v/5.5.3/

Ceux pour qui ces choses seraient nouvelles, retenez bien la différence entre elles deux : Drupal sert à manipuler des données, et Zurb Foundation, à les afficher aux internautes. Le premier est exécuté par un serveur sur Internet, le second est interprété par le client via le navigateur. Server / client. Backend / frontend.

SASS

Zurb Foundation propose de rédiger votre CSS grâce au SASS. Le SASS est un préprocessor CSS qui facilite considérablement la rédaction de votre CSS. Si vous ne connaissez pas, consultez dès maintenant http://sass-lang.com/guide !

Grunt et Bower

Ces deux éléments sont nécessaires pour travailler avec Zurb Foundation. Grunt est un gestionnaire de tâches qui - dans notre cas - nous permettra d'exécuter des opérations telles que la concaténation de feuilles SCSS, leur minification et la génération des feuilles SCSS. Bower est un gestionnaire de dépendances qui permet de gérer très simplement les librairies externes nécessaires au fonctionnement de Zurb Foundation.

CARACTÉRISTIQUES DU RESPONSIVE

En réalité le responsive ne se limite pas qu'à ce qu'il se passe sur votre navigateur. Il est d'ailleurs souvent associé au terme d'adaptive qui prend en considération l'optimisation des ressources transmises selon le support d'où elles sont exploitées. En effet, la bande passante des réseaux mobiles de votre smartphone n'a pas la même capacité que l'ADSL à votre bureau. Le responsive est concerné par ce qui se passe sur le serveur, et ce qui s'affiche sur le navigateur.

Le responsive a apporté deux notions essentielles dans la conception de sites Internet : les composants et le touch ready. Dorénavant, les interfaces sont utilisées avec le touch des écrans tactiles et sont consultées sur des dimensions d'affichage allant de celles des smartwatchs jusqu'aux écrans de télévision. Les composants sont des éléments standardisés qui constituent les sites Internet d'aujourd'hui, vous trouverez la documentation de de Google Material Design sur Internet.

ENVIRONNEMENT DE TRAVAIL

Réaliser des sites Internet modernes exige aussi de travailler dans un environnement adéquat :

- Un serveur Web sur lequel vous avez la main (hébergement mutualisé à bannir) ;
- Un OS sur lequel vous pouvez installer, notamment SASS, Grunt et Bower ;
- Drush 8 pour la manipulation de Drupal en command shell.

L'article a été rédigé depuis un environnement sous Ubuntu Gnome 14 LTS, Node 4.0.0, SASS 3.4.21, Grunt-cli 0.1.13, Bower 1.3.12, Drupal 7.43, Drush 8.0.0-rc3.

Installation de votre environnement

Drush http://docs.drush.org/en/master/install/

```
# Télécharge Drush
php -r "readfile('http://files.drush.org/drush.phar');" > drush
```

```
# Teste l'install
php drush core-status
# Rend Drush exécutable
chmod +x drush
sudo mv drush /usr/local/bin
# Ajoute l'autocompletion et les alias
drush init
```

SASS http://sass-lang.com/install

```
# Installe RubyGems dont dépend SASS
sudo apt-get install rubygems
# Installe SASS et ses dépendances
sudo gem install sass
# Teste la version de SASS
sass -v
```

NodeJS https://nodejs.org/en/download/package-manager/
(nécessaire pour GruntJS)

```
# Installe NodeJS
curl -sL https://deb.nodesource.com/setup_4.x | sudo -E bash -
sudo apt-get install -y nodejs
# Installe GruntJS
npm install -g grunt-cli
```

Bower http://bower.io/

```
npm install -g bower
```

DRUPAL

Pour commencer, nous installons Drupal, puis le Drupal theme Zurb Foundation.

Installation de Drupal, avec Drush

L'installation de Drupal en utilisant Drush est très facile :

```
# Télécharge Drupal 7
drush dl drupal-7
# Par convention, les fichiers accessibles depuis l'Internet seront dans public_html
# Pensez à paramétrer votre serveur Web pour pointer sur le dossier public_html
mv drupal-7.43 public_html
cd public_html
# Installe Drupal, avec le profile "standard"
drush si standard --db-url=mysql://root:mysql@localhost/programmez
```

Done ! Nous avons téléchargé et installé Drupal 7. Pour l'installation de Drupal, vous pouvez ajouter ces paramètres :
--account-name=*username*
--account-pass=*mot_de_passe*
Où *username* et *mot_de_pass* seront vos identifiants du user 1 (user équivalent au root).
Plus de détails sur : http://drushcommands.com/drush-8x/core/site-install/

Installation du Drupal theme Zurb foundation

Installons de la même manière le Drupal theme Zurb Foundation :

```
# Télécharge le projet Drupal Zurb_Foundation
drush dl zurb_foundation-7.x-5.0-rc6
# Installe le thème
drush en -y zurb_foundation
# Vide tous les caches de Drupal
drush cc all
# Créé un subtheme Zurb Foundation, qui sera le thème que nous personnaliserons
drush fst Programmez! programmez 'Zurb Foundation subtheme for Programmez! magazine'
```

[theme_name] est le nom machine de votre thème. Il sera régulièrement utilisé pour la suite. C'est le deuxième argument de drush fst.
Qu'est-ce qu'un subtheme ? Lorsque nous utilisons un thème Drupal, nous sollicitons de nombreuses mécaniques - celles proposées par le Drupal theme Zurb Foundation dans notre cas - que nous ne souhaitons pas altérer directement. Cela rendrait sa mise à jour impossible. Nous créons donc un thème, que nous personnaliserons librement, qui hérite des mécaniques du thème d'origine. La maintenabilité de votre projet est conservée.
À noter : Zurb Foundation et Drupal theme Zurb Foundation. Le premier concerne uniquement le framework, le second, l'intégration du framework dans l'environnement Drupal.
La version **7**.x-**5**.0-rc6 signifie qu'elle est faite pour Drupal **7**, avec la version **5** de Zurb Foundation. Si à la lecture de l'article les versions ont évolué, consultez la page officielle :
https://www.drupal.org/project/zurb_foundation
jQuery 1.7 : Profitons-en pour installer jquery_update, car Zurb Foundation a besoin d'au moins de jQuery 1.7.

```
# Installe jquery_update (télécharge si le module est localement introuvable)
drush en -y jquery_update
```

Installer les dépendances du thème

Installons les composants nécessaires à l'exploitation de Zurb Foundation et du thème :

```
# Se positionner dans le dossier du thème
cd sites/all/themes/programmez
# Installe les dépendances
npm install
# Ajoutez --allow-root si vous êtes en root
bower install
```

Nous avons maintenant dans notre thème les éléments suivants :

```
ls -l
drwxr-xr-x  bower_components
-rw-r--r--  bower.json
drwxr-xr-x  css
drwxr-xr-x  fonts
-rw-r--r--  Gruntfile.js
drwxr-xr-x  js
-rw-r--r--  logo.png
drwxr-xr-x  node_modules
-rw-r--r--  package.json
-rw-r--r--  programmez.info
-rw-r--r--  README.txt
```

```
drwxr-xr-x  scss
-rw-r--r--  template.php
drwxr-xr-x  templates
-rw-r--r--  theme-settings.php
```

Vous trouverez dans le dossier bower_components les librairies Javascript nécessaires au fonctionnement de Zurb Foundation, tels que jQuery, fastclick et modernizr.

```
ls -l bower_components
drwxr-xr-x 3 fastclick
drwxr-xr-x 5 foundation
drwxr-xr-x 5 jquery
drwxr-xr-x 2 jquery.cookie
drwxr-xr-x 2 jquery-placeholder
drwxr-xr-x 5 modernizr
```

Dans le dossier node_modules, les librairies nécessaires au fonctionnement de Grunt et de SASS.

```
ls -l node_modules
drwxr-xr-x grunt
drwxr-xr-x grunt-contrib-jshint
drwxr-xr-x grunt-contrib-uglify
drwxr-xr-x grunt-contrib-watch
drwxr-xr-x grunt-drush
drwxr-xr-x grunt-sass
drwxr-xr-x load-grunt-tasks
drwxr-xr-x node-bourbon
drwxr-xr-x node-sass
```

Utiliser et exécuter notre thème

Nous avons Drupal et notre thème prêts à l'emploi. Pour les voir à l'œuvre ensemble, il ne nous reste plus qu'à mettre notre thème comme étant par défaut et de lancer la génération des fichiers CSS et JS qui seront interprétés par le navigateur pour afficher convenablement notre site Internet.

```
# Définit par défaut le thème programmez
drush vset theme_default programmez
# Vide les caches
drush cc all
# Exécute la commande grunt default
grunt
```

La commande grunt exécute la tâche par défaut enregistrée dans le fichier Gruntfile.js, qui permet de générer les CSS et JS de votre thème.

Rapide coup d'œil dans le fichier Gruntfile.js :

```
vim Gruntfile.js
#[... en bas de page]
grunt.registerTask('build', ['jshint','uglify','sass']);
grunt.registerTask('default', ['build', 'watch']);
```

grunt default lance les tâches build et watch. Watch permet d'écouter (CTRL + C pour interrompre le processus) les modifications faites dans les fichiers contenus dans les dossiers scss et js du thème afin de générer au fil des travaux les feuilles CSS et scripts JS utilisés pour le bon fonctionnement de votre site Internet. Build, quant à lui, lance jshint, uglify et sass, des tâches qui, en somme, génèrent les feuilles de styles CSS et scripts JS à partir des différents fichiers du thème. Lorsque vous ferez une mise en production de votre projet, lancez grunt build directement depuis votre ligne commande.
Attaquons-nous maintenant à la personnalisation du thème !

PERSONNALISATION DU THÈME

Le thème Zurb Foundation de Drupal laisse de nombreuses possibilités pour personnaliser rapidement votre thème. Pour ce faire, copions le fichier _settings.scss du *base_theme* zurb_foundation se trouvant dans le dossier scss/foundation pour le coller dans scss/base de votre subtheme. En d'autres termes :

```
cp ../zurb_foundation/scss/foundation/_settings.scss scss/base
```

Pour terminer, ajouter la ligne suivante dans /scss/programmez.scss (correspondant à *[theme_name]*.scss) après @import "base/init"; :

```
@import "base/settings";
```

Rappelez-vous que Grunt s'occupe de générer les CSS et les JS. Pour que cela s'exécute automatiquement, Watch doit tourner en tâche de fond (depuis la command grunt). À défaut, utilisez grunt build. Apparaît dans la console le résultat du Grunt, notamment les erreurs syntaxiques s'il devait y en avoir.

```
Running "jshint:all" (jshint) task
>> 2 files lint free.

Running "uglify:dist" (uglify) task

Running "sass:dist" (sass) task

Running "watch" task
Waiting...
```

Attention, à la création de nouveaux fichiers scss, pensez à bien l'importer dans *[theme_name]*.scss. En effet, en regardant dans Gruntfile.js, initConfig de sass.files indique bien que *[theme_name]*.scss sera le fichier qui sera compilé par SASS pour générer votre CSS. Importez donc systématiquement votre SCSS dans *[theme_name]*.scss ou personnalisez Gruntfile.js.
Concentrons-nous maintenant sur _settings.scss et commençons la personnalisation visuelle.

Les variables à personnaliser

Vous trouverez dans _settings.scss de nombreuses variables, que vous pouvez modifier à votre guise ! Les caractéristiques du grid, les couleurs, les fonts, les media-query ranges, et les propriétés des components que propose Zurb Foundation tels que les buttons, forms et side-nav. Ne soyez pas impressionné par le nombre de propriétés, concentrez-vous à partir de la table des matières sur les éléments qui vous intéresse. Pour l'exemple, je vais personnaliser les couleurs :

```
// c. Global
```

```
// We use these as default colors throughout
$primary-color: #E02D2D;
$secondary-color: #2C3E50;
# vous pouvez laisser en commentaires les lignes que vous ne modifiez pas.
// $alert-color: #f04124;
// $success-color: #43AC6A;
// $warning-color: #f08a24;
$info-color: #F9F9F9;
```

Ainsi que la largeur maximale des rows :

```
// b. Grid
$row-width: rem-calc(1200);
// $total-columns: 12;
// $column-gutter: rem-calc(30);
```

Structure des fichiers SCSS

Les contributeurs du projet Zurb Foundation de Drupal propose la structure suivante :

```
drwxr-xr-x 2 base
drwxr-xr-x 2 layout
drwxr-xr-x 2 modules
-rw-r--r-- 1 [theme_name].scss
drwxr-xr-x 2 states
drwxr-xr-x 2 theme
```

Libre à vous de suivre leur convention. Par expérience, j'utilise la structure suivante :

```
drwxr-xr-x 2 base
drwxr-xr-x 2 layout
drwxr-xr-x 2 components
-rw-r--r-- 1 [theme_name].scss
```

base contient les propriétés de base : variables et mixins en tout genre. layout contient les règles de styles liées aux pages telles que node-*[type]*, node-*[type]*-edit, node-*[type]*-teaser et page-user. Enfin, components contient la définition des composants communément utilisés dans l'ensemble de mes sites Internet, notamment les cards, les collections, les buttons et bien d'autres (cf: Material Design). Cela donne une vision claire de votre intégration et facile à maintenir.

Ajouter des librairies JS

Il existe un fichier *_[theme_name]*.js dans le dossier js. Celui-ci est traité par la tâche grunt default qui génère un fichier minimifié. Il est d'usage d'encapsuler le code Javascript dans une fonction qui est exécutée que lorsque le DOM est construit et rendu. Pour effectuer des manipulations sur des éléments du DOM, avec par exemple jQuery.

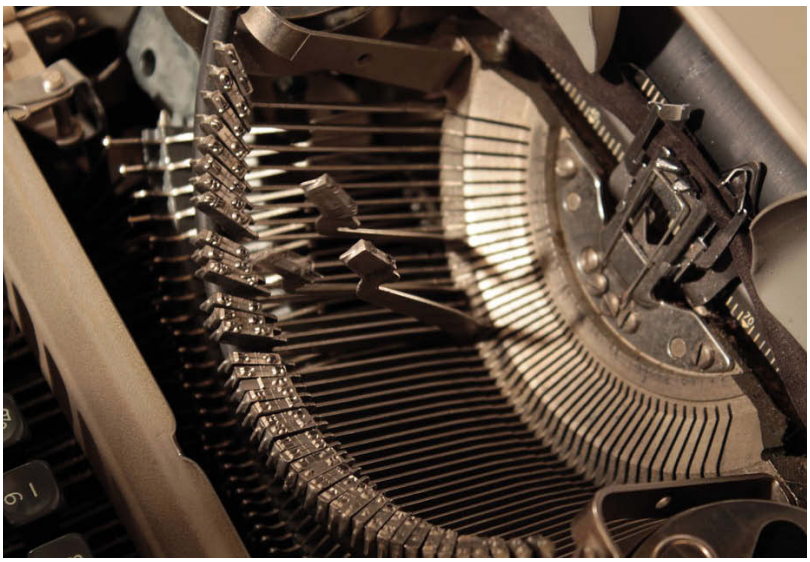
```
(function ($, Drupal, window, document, undefined) {
 $(function () {
  /* Ici, mon code est exécuté lorsque le DOM est prêt ! */
  console.log('DOM is ready. Go ahead!');
 });
})(jQuery, Drupal, this, this.document);
```

Pour ajouter des librairies JS externes, sans passer par *[theme_name]*.info, utilisez Gruntfile.js. Ajouter une ligne à l'array jsLibs en précisant correctement le path de votre ressource et le tour est joué. •

Suite dans Programmez! 202

Faire son propre clavier

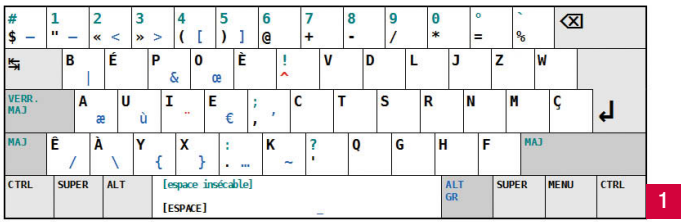
Guillaume COLLIC



Le lourd passé de nos claviers

Les touches principales de nos claviers ont à peine changé depuis les premières machines à écrire de la fin du 19e siècle. Les contraintes de l'époque étaient que les leviers de chaque touche ne pouvaient pas être en face d'un autre, il était donc obligatoire de disposer les touches en quinconce. Cette contrainte a été levée depuis longtemps, mais la disposition a été conservée.

Depuis plus d'un siècle, des personnes proposent des solutions pour rendre nos claviers plus ergonomiques. Même l'inventeur de la machine à écrire a voulu apporter des améliorations, refusées par le distributeur pour ne pas perturber les premiers utilisateurs. Les solutions alternatives portent soit sur la disposition physique (où sont les touches?) soit logique (que fait chaque touche?), 2 aspects indépendants.



1

Dispositions physiques alternatives

Les ergonomes conseillent de garder les mains dans le prolongement des épaules, plus écartées que sur un clavier classique où nos pouces se touchent presque. Ils conseillent aussi de limiter l'usage des frêles auriculaires, au bénéfice d'un usage plus important des pouces. Enfin, ils privilégient de suivre la forme de la main et les mouvements des doigts, et donc non plus organisé en quinconce (lignes droites et colonnes penchées), mais plutôt orthogonalement (colonnes droites positionnées en fonction de la longueur des doigts).

De nombreuses marques vendent des claviers qui suivent à différents degrés ces principes (Kinesis, TrulyErgonomicKeyboard, TypeMatrix, Maltron…). Ils sont parfois prescrits dans le cadre du syndrome du canal carpien.

Dispositions logiques alternatives

La disposition physique ne traite qu'une moitié du problème. La position des lettres en qwerty a été choisie pour limiter les blocages des machines à écrire lorsqu'on tape successivement sur 2 touches adjacentes. Il n'y a plus aucune raison à cela, et depuis 1932 des modèles scientifiques (en particulier par le Dr Dvorak) proposent des placements ergonomiques, mettant notamment les lettres les plus fréquentes sous les positions de repos des doigts. Cette approche se base sur un corpus de langue (des ouvrages de référence représentatifs des lettres les plus utilisées). Appliqué à un corpus anglais, cela donne la disposition Dvorak. Appliqué au français littéraire, cela donne la disposition bépo, aux multiples avantages (ergonomie, cédille et accents disponibles sur les majuscules, nombreux caractères spéciaux faciles d'accès et à mémoriser …). [Fig.1]

Les considérations physiques et logiques sont bien distinctes. On pourra profiter des bienfaits des dispositions logiques Dvorak ou du bépo sans changer de clavier physique (à l'aide de stickers si besoin). Il suffit d'installer ces dispositions sur son ordinateur. On peut aussi avoir un clavier physiquement ergonomique, mais conserver une disposition logique qwerty ou azerty.

Fonctionnement d'un clavier

Quand on dit qu'un clavier est qwerty-britannique, azerty ou bépo, cela n'est que cosmétique. L'ordinateur reçoit exactement les mêmes signaux : un "scan-code" correspondant à l'emplacement appuyé. C'est ensuite la disposition logique configurée dans le système d'exploitation qui traduira ce "scan-code" physique en une lettre sous forme d'un "virtual key-code". Cela veut aussi dire qu'un clavier physique fait maison ne pourra pas écrire n'importe quel symbole. Si vous voulez avoir des caractères autres que ceux supportés par votre disposition logique actuelle, il vous faudra développer une autre disposition logique.

Un clavier consiste tout simplement à transformer l'appui sur un interrupteur en un code d'emplacement de touche (scan-code, ou son équivalent sur le protocole USB HID) et à le transmettre à l'ordinateur, généralement par USB.

Construction du clavier

Des kits existent, pour faire des claviers standard (comme l'infinity), ergonomiques (comme l'ergodox), ou ultra-compacts (comme le planck). Ils contiennent en général un circuit intégré et tout ce dont vous avez besoin, prêt à assembler en une après-midi. Sans kit ou sans circuit intégré, il est facile de faire le câblage à la main.

Choix des interrupteurs et des capuchons

Les claviers premier-prix du commerce sont dits à membranes. Ils sont composés d'une carte électronique surmontée de dômes en plastique

contenant un conducteur à leur sommet, faisant contact entre 2 pistes lors de l'appui. Ce mode de fonctionnement n'est pas adapté pour être fait artisanalement. Les claviers faits maisons sont alors basés sur les interrupteurs mécaniques que l'on trouve sur les claviers haut de gamme, avec le budget qui va avec (premier prix aux alentours de 100 à moins d'opter pour de la récupération ou pour un mini-clavier). Ces interrupteurs existent en plusieurs familles, déterminant quels types de capuchons peuvent se clipser dessus. La famille la plus courante est celle des compatibles Cherry MX, produite par plusieurs marques (je conseille Gateron ou Cherry MX), et déclinée suivant la force nécessaire à l'activation (généralement de 35 à 80g) et la sensation de clic (de prononcé à entièrement absent). On peut trouver des sets de capuchons pour Cherry MX à tarif raisonnable (en particulier qwerty ou vierge), ou plus évolués (impression quadricolor à la demande, ou séries limitées) mais les prix s'envolent.

Choix du microcontrôleur

Il nous faut un microcontrôleur pouvant communiquer avec l'ordinateur par USB. Le plus fréquemment utilisé pour cela est le ATMEGA32U4 car le principal firmware de clavier Open Source le supporte bien. On l'achète souvent via la carte Teensy 2.0 qui a pour avantage une grande simplicité d'utilisation, mais un Arduino Leonardo Pro Micro fera aussi l'affaire à moindre coût.

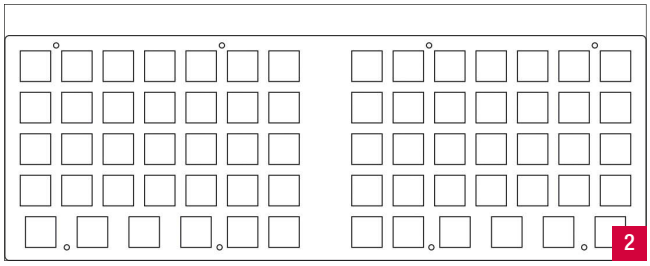
Choix du boitier

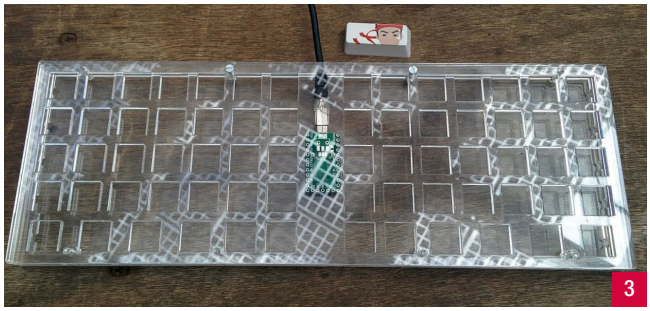
Il faut "simplement" concevoir un boîtier avec des trous adaptés à vos interrupteurs. Les dimensions des trous sont disponibles dans les spécifications de vos interrupteurs (attention aux touches plus grandes comme shift et espace qui nécessitent des stabiliseurs).

Une option fréquente pour faire le boîtier est la découpe laser. Ce service est souvent proposé localement (éditeurs de plaques et tampons), et http://damengo.com offre de bons services avec une tarification claire et avantageuse. Le format svg est souvent supporté pour la découpe. Ce format de fichier basé sur xml est manipulable dans Inkscape pour la forme globale, et on peut facilement générer par code la partie répétitive et précise des trous des interrupteurs via une balise rect pour chaque touche à la bonne position (x, y) :

```
<rect x="19" y="38" width="14" height="14" />
```

[Fig.2 et 3]

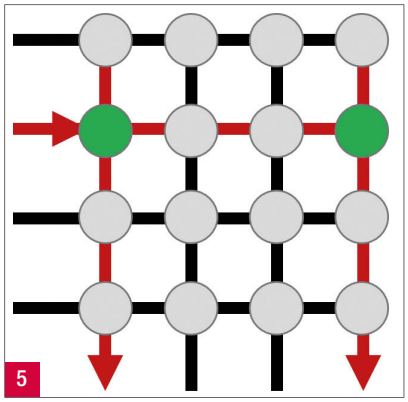

2


3


4

La découpe laser se prête bien à un clavier "plat", mais pour épouser la forme des mains il faut s'orienter vers l'impression 3D. Le clavier "Dactyl" de Matt Adereth en est un bon exemple, généré en clojure https://github.com/adereth/dactyl-keyboard. [Fig.4]

Connecter les interrupteurs au microcontrôleur


5

On pourrait brancher chaque interrupteur à une entrée d'un microcontrôleur, mais cela nécessiterait beaucoup d'entrées. Il est donc d'usage de faire une matrice : mettre des fils en lignes et en colonnes, et les relier par les interrupteurs à chaque intersection. Les lignes sont alors toutes éteintes électriquement, puis le microcontrôleur les allume une par une, et il observe les colonnes. Si la colonne reçoit du courant, alors l'interrupteur à l'intersection est enclenché. [Fig.5]

Ce concept a un problème : si trop d'interrupteurs sont enclenchés, le courant peut passer d'une ligne à l'autre et faire de faux positifs ("ghosting"). Une solution fréquente à ce problème est de rajouter des diodes (référence 1N4148) à chaque interrupteur ("n-key rollover").

Si on prend une matrice pour un clavier classique, on peut faire correspondre les lignes et colonnes de la matrice électronique avec ceux de la disposition physique, cela donnerait par exemple 6 lignes et 20 colonnes, soit 26 connexions nécessaires sur le microcontrôleur au lieu des 105 au départ. Mais on peut aller plus loin et faire notre matrice la plus carrée possible. Une matrice de 11 par 10 suffit pour brancher nos 105 touches et ne nécessite plus que 21 connexions, le branchement est par contre un peu plus complexe puisque les lignes et colonnes de la matrice ne correspondent plus directement aux positions des interrupteurs. [Fig.6, 7 et 8]

Programmation du clavier

Pour programmer le microcontrôleur, je conseille d'utiliser le firmware Open Source TMK https://github.com/tmk/tmk_keyboard. Il est riche en fonctionnalités, éprouvé, et avec de nombreux exemples.

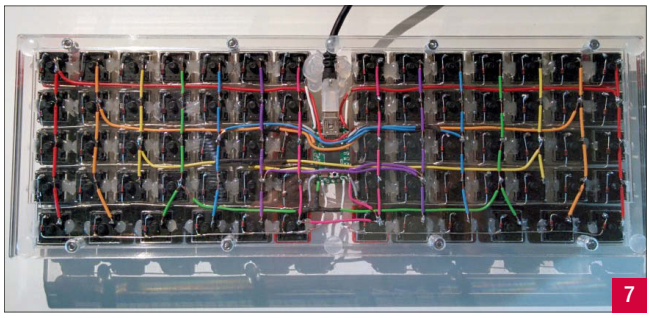
Le code source sur lequel est basé l'exemple est disponible sur https://github.com/gcollic/Grimoire60_keyboard.

Nous allons le modifier pour faire un clavier à 5 touches, avec les 4 flèches et une touche de fonction : [Fig.9]

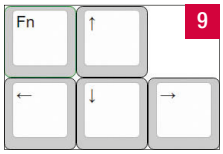
Lorsque la touche de fonction aura été appuyée, le clavier deviendra : [Fig.10]

Nous partons du principe que le microcontrôleur aura été connecté ainsi : [Fig.11]

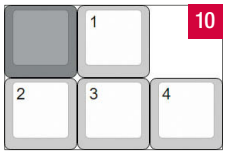

6


7

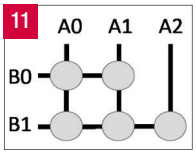

8



9



10



11

Environnement de compilation

TMK est écrit en c, et nécessite un environnement de compilation avr. C'est possible d'en mettre un en place de différentes façons suivant le système d'exploitation sur lequel on est, mais le plus simple est d'utiliser Docker (sur Linux, Windows ou Mac).

Voici les étapes faites par le Dockerfile, que vous pouvez suivre à la main si vous préférez. Nous partons d'une debian. TMK a besoin de quelques packages que nous installons avec apt-get : make, gcc-avr, et avr-libc. Nous récupérons ensuite la librairie tmk depuis https://github.com/tmk/tmk_core et nous la mettons dans le répertoire /tmk_core. Nos sources iront dans le répertoire /src, où un Makefile nous permettra de compiler le projet avec la commande make, ce qui produira un fichier keyboard_firmware.hex. Si vous utilisez Docker, le fichier sera dans le conteneur, pour le récupérer sur votre machine vous pouvez utilisez la commande "docker cp", comme dans le fichier build.sh :

```
docker build -t gcollic/grimoire60 .
docker run --name tempcontainer gcollic/grimoire60
docker cp "tempcontainer:/src/keyboard_firmware.hex" .
docker rm tempcontainer
```

Maintenant que l'environnement de compilation est en place, passons à nos sources : Makefile, config.h, matrix.c et keymap.c.

Makefile

Le fichier Makefile configure la compilation. Nous n'avons pas besoin de le modifier. Il y est indiqué le nom du binaire final, où se situe la librairie TMK, quel microcontrôleur on utilise (ATMEGA32U4), etc.

config.h

Le fichier config.h configure les informations générales du périphérique USB. Il faut y renseigner le nom du périphérique, constructeur, etc. La partie qui nous intéresse le plus y est la configuration de la dimension de la matrice : son nombre de lignes et de colonnes.

```
#define MATRIX_ROWS 2
#define MATRIX_COLS 3
```

matrix.c

Le fichier matrix.c est le code qui configure quelles pins du microcontrôleur correspondent à quelles lignes et colonnes de la matrice, sous forme d'opérations binaires assez rébarbatives. Les pins sont nommées par une lettre (A à F) suivie d'un chiffre (0 à 7). Pour modifier le comportement de ces pins, il y a 3 registres : DDR, PORT, et PIN. Chaque registre est accessible sous forme d'un octet en ajoutant à la fin la lettre correspondante. Au sein de cet octet, chaque bit du registre correspond aux chiffres de 0 à 7. Par exemple, le registre DDR de la pin B3 est accessible depuis le quatrième bit de l'octet DDRB.

Dans init_cols, pour chaque pin de colonne (A0 à A2) il faut configurer le registre DDR à 0 et le registre PORT à 1.

```
static void init_cols(void)
{
  DDRA &= ~(1<<0 | 1<<1 | 1<<2);
  PORTA |= (1<<0 | 1<<1 | 1<<2);
}
```

Dans read_cols, il faut lire la valeur de chaque colonne et si elle est allumée alors on met à 1 dans le résultat le bit positionné à son index de colonne dans la matrice :

```
static matrix_row_t read_cols(void)
{
  return (PINA&(1<<0) ? 0 : (1<<0)) |
      (PINA&(1<<1) ? 0 : (1<<1)) |
      (PINA&(1<<2) ? 0 : (1<<2));
}
```

Dans unselect_rows, il faut mettre les registres DDR et PORT à 0 pour les pins des lignes (B0 et B1).

```
static void unselect_rows(void)
{
  DDRB &= ~(1<<0 | 1<<1);
  PORTB &= ~(1<<0 | 1<<1);
}
```

Dans select_row, il faut allumer la ligne demandée en paramètre. Pour cela, il faut mettre les registres DDR à 1 et PORT à 0 pour la pin correspondant au numéro de la ligne demandée.

```
static void select_row(uint8_t row)
{
  // Output low(DDR:1, PORT:0) to select
  switch (row) {
    case 0:
      DDRB |= (1<<0);
      PORTB &= ~(1<<0);
      break;
    case 1:
      DDRB |= (1<<1);
      PORTB &= ~(1<<1);
      break;
  }
}
```

keymap.c

Le fichier keymap.c est le cœur de notre clavier, c'est là que l'on configure ce que font chaque touche de la matrice.

TMK propose une couche principale active par défaut, et des couches activables par des touches de fonctions (comme sur un ordinateur portable). Une couche est configurée sous forme d'un tableau de constantes, de même dimension que notre matrice. Les constantes commencent par KC_ et décrivent ce qui doit être fait (envoi d'une touche au PC, changement de couche, etc.).

La première chose à coder est une macro pour nous simplifier la vie. Au lieu d'écrire directement les tableaux de constantes en suivant la matrice, on va plutôt décrire le clavier comme on le voit, et la macro remettra dans le bon ordre pour correspondre à la matrice si elle diffère. La macro nous évitera aussi de préfixer chaque constante par KC_. La matrice contient souvent des cases qui sont vides (la barre espace par exemple prend beaucoup de place, mais n'est branchée que sur une seule colonne), on les remplira par la constante KC_NO, sans effet.

```
#define KEYMAP( \
  K00, K01, \
  K10, K11, K12 \
) { \
  { KC_##K00, KC_##K01, KC_NO  }, \
  { KC_##K10, KC_##K11, KC_##K12 } \
}
```

Il nous suffit ensuite de remplir le tableau keymaps avec les codes de chaque touche, à l'aide de la macro. Comme indiqué précédemment, nous envoyons des scan-codes qui sont indépendants de la disposition logique, mais pour autant les noms des constantes de TMK reprennent le contenu des touches équivalentes en Qwerty, c'est donc en Qwerty qu'il faudra réfléchir à cette étape. La documentation explique bien ce que chaque constante fait https://github.com/tmk/tmk_keyboard/blob/master/tmk_core/ doc/keymap.md.

Certaines ont des effets particuliers. Par exemple KC_NO n'aura aucun effet, et KC_TRNS sera "transparent", c'est-à-dire que cette couche n'aura pas d'effet pour cette lettre et que la couche précédente active sera appliquée.

```
const uint8_t PROGMEM keymaps[][MATRIX_ROWS][MATRIX_COLS] = {
  // Keymap 0: Default Layer
  KEYMAP(
    FN0,  UP,  \
    LEFT, DOWN, RGHT),
  // Keymap 1: Main function layer
  KEYMAP(
    FN0,  1,  \
     2,  3,  4),
};
```

Une autre catégorie de constante va nous intéresser : les FNx (FN0, FN1, ...). Ces touches déclencheront des actions plus complexes, décrites dans le tableau fn_actions, à l'index équivalent. On pourra en particulier y affecter l'activation des couches de fonctions, soit en restant appuyé (ACTION_LAYER_MOMENTARY) soit avec un appui court (ACTION_LAYER_TOGGLE).

```
const uint16_t PROGMEM fn_actions[] = {
  [0] = ACTION_LAYER_TOGGLE(1)
};
```

Compiler et uploader

Il n'y a plus qu'à compiler avec les commandes vues précédemment, et uploader le firmware keyboard_firmware.hex ainsi généré dans le microcontrôleur. Dans le cas du Teensy il suffit d'appuyer sur le bouton et de charger le fichier depuis le programme Teensy loader. Votre clavier est prêt !

Faire sa propre disposition logique

On peut faire sa propre disposition logique aussi, mais c'est très différent d'un OS à un autre. Le code source du bépo est un bon exemple pour approfondir cette voie-là.

Plus d'information

Pour plus de détail sur le passé des claviers et les dispositions Dvorak/bépo, la très bonne BD "Dvorak zine" est disponible gratuitement sur http://bit.ly/dvorakzinefr.

- Communautés

http://forum.bepo.fr
https://deskthority.net/en-francais-f21
https://geekhack.org
https://www.reddit.com/r/MechanicalKeyboards

- Disposition ergonomique

http://bepo.fr

- Fournisseurs

http://damengo.com
https://www.pjrc.com/store/teensy.html
https://www.massdrop.com/mechanical-keyboards
http://falbatech.pl
http://olkb.com
http://www.keyboardco.com
http://www.4keyboard.com
http://www.wasdkeyboards.com
http://matias.ca/switches/quiet

Introduction à Rust

Partie 1

Damien Lecan,
Directeur Technique,
SQLI Nantes

Rust est un jeune langage qui a pour ambition de se substituer au C/C++ en proposant de nouveaux paradigmes de programmation, une librairie standard de haut niveau et un écosystème riche soutenu par une communauté très active.

Multiplateforme (systèmes d'exploitation ou architectures de processeurs) et pourvu de tous les concepts de programmation attendus pour un langage moderne (programmation orientée "objet", programmation fonctionnelle, facilités à développer des programmes d'exécution concurrente), Rust a beaucoup d'atouts pour séduire. Avant de plonger dans le code, je vous propose de revenir sur les origines du langage. Elles remontent à 2010 : Graydon Hoare, ingénieur chez Mozilla, révèle ses travaux sur Rust. À l'époque, le "marché" des langages informatiques est un peu similaire à aujourd'hui, à savoir une domination de C/C++ et de Java, chacun cantonné à ses domaines de prédilection. En schématisant, Java règne sur l'informatique de "gestion" (systèmes d'information bancaires, assurance …) ainsi que sur Android, tandis que C/C++ est le choix par défaut de la programmation que je qualifierais de "système" ou "bas niveau" : système d'exploitation, pilotes, informatique embarquée, machines virtuelles (Java lui- même, ou NodeJS par exemple) …

Quelle caractéristique clivante peut bien isoler à ce point ces deux mondes de l'informatique ? Je pense qu'il s'agit principalement de la gestion de la mémoire, qui diverge totalement. Avec Java, elle est en grande partie masquée au développeur, c'est-à-dire que lorsqu'un développeur crée des objets en mémoire, il n'a pas vraiment à se soucier de son nettoyage. En effet, un processus qui s'exécute en arrière-plan, le ramasse-miettes (garbage collector), se charge de détecter les objets qui ne seraient plus utilisés et de les supprimer de la mémoire. Le travail du développeur est donc grandement simplifié, et celui-ci peut se concentrer sur le code métier le plus utile. C'est ce qui explique le succès de tous les langages à machine virtuelle et ramasse-miettes, comme JavaScript ou .Net. Si le fonctionnement du ramasse-miettes n'a pas ou peu de conséquences négatives -largement compensées par d'autres avantages- dans les applications de haut niveau comme celles dédiées au Web, il est problématique, voire rédhibitoire pour d'autres. La latence, certes réduite, induite par l'initialisation ou le fonctionnement du ramasse-miettes, ne permet pas de construire des systèmes de plus bas niveau, comme les systèmes d'exploitation, les pilotes, les machines virtuelles elles-mêmes ou encore des programmes en ligne de commandes efficaces … Dans ces cas, la mémoire doit donc être gérée "à la main".

En C/C++, la responsabilité du cycle de vie des structures ou objets en mémoire incombe au développeur et gare à la qualité du logiciel si cette tâche est mal effectuée : corruption de données, plantages, failles de sécurité … sont les conséquences principales d'une mauvaise gestion de la mémoire, phénomène encore aggravé dans un contexte de programme à exécution concurrente. Ne jetons pas la pierre aux développeurs : c'est une tâche ingrate, pénible et peu outillée. Alors dans ces conditions, comment produire des logiciels bas niveau de qualité ?

Depuis les premières ébauches, c'est précisément l'un des objectifs du langage Rust. Il ne repose ni sur un ramasse-miettes (ex: Java), ni sur une gestion mémoire manuelle du développeur (ex: C/C++). Rust propose une nouvelle façon de gérer la mémoire, qui se veut sûre et garantie à la compilation, ce qui permet d'en limiter l'impact sur les performances à l'exécution. Autant vous prévenir tout de suite : la courbe d'apprentissage de Rust est donc plus lente que dans d'autres langages mais largement compensée en qualité et fiabilité des programmes produits. Concrètement, vous allez transpirer au début mais serez fiers de la qualité de vos productions.

La syntaxe du langage, ainsi que la librairie standard, ont été débattues, testées, amendées par la -déjà très active- communauté, pendant des années ; la première version stable 1.0 de Rust ayant été publiée en avril 2015. Ce délai a permis aussi de mettre en place une gouvernance ouverte et transparente qui régit l'évolution du langage, à laquelle chacun peut participer (*board* central, RFC ouvertes …). Trois *channels* de Rust sont édités en parallèle : nightly, beta et stable. De nouvelles versions de Rust *stable* et *beta* sont publiées toutes les 6 semaines, apportant à chaque fois leurs lots d'évolutions au langage.

Avant de démarrer, soyez attentifs lors de vos recherches sur Internet à la version de Rust concernée par les articles de blogs ou les solutions apportées sur Stack Overflow. On y trouve en effet beaucoup de contenus obsolètes, car applicables à des versions de Rust antérieures à la version 1.0. Enfin, un mot sur la communauté de développeurs Rust. Elle est accueillante et bienveillante, vous trouverez de nombreux développeurs prêts à vous aider, à vous faire progresser ainsi qu'à critiquer votre code de manière constructive. Une bonne surprise et une des forces de ce langage.

Bonjour lecteurs de Programmez!

Commençons à écrire quelques lignes de code, sans installation préalable de Rust sur votre poste. Rendez-vous sur https://play.rust-lang.org/ pour ouvrir l'interpréteur Web Rust, qui fonctionne très bien sur smartphone ou tablette. Idéal pour tester simplement Rust sans vous prendre la tête ! Ne modifiez pas le paramétrage par défaut de la page (boutons du haut) et concentrez-vous sur la zone de texte centrale : c'est là qu'il faut taper le code Rust. Lors de votre première connexion, le formulaire est déjà préinitialisé avec un morceau de code que je vous propose de substituer par un contenu plus adapté à notre contexte :

```
fn main() {

    println!("{}", "Bonjour Programmez!");
}
```

Si vous avez des difficultés à saisir ce code, vous pouvez ouvrir directement cette URL : https://is.gd/JzBaCy. Par la suite, je présenterai systématiquement un lien vers le code présenté, vous permettant éventuellement de le copier/coller. Cliquez ensuite sur le gros bouton rouge "Run" en haut pour exécuter ce programme. Vous devez alors obtenir sous la zone de texte de code l'affichage de "Bonjour Programmez!". Bravo, vous venez d'écrire votre premier programme Rust !

Analysons ensemble ces quelques lignes :

La syntaxe s'inspire fortement du langage C (que l'on retrouve en Java, JavaScript, .Net, …) et le formatage du code est standardisé par le langage : indentation, espaces, positions des accolades … Notez le bouton "Format" dans l'interpréteur Web qui formate le code de la zone de texte comme on peut s'y attendre :)

- fn permet de déclarer une fonction, nommée ici *main* et sans argument. Par convention, c'est le point d'entrée unique d'un programme écrit en Rust ;
- println! permet d'écrire dans la sortie standard, du texte ou des structures plus complexes. Notez-le ! qui signifie que println! est une macro. C'est une routine de génération de code **à la compilation** et un pattern de développement très utilisé par les développeurs Rust pour masquer une complexité ;
  Je vous recommande l'utilisation de println! à deux arguments et plus, syntaxe familière à ceux qui connaissent les fonctions printf/fprintf du C ou str.format de Python : le 1er paramètre représente la mise en forme, les suivants les valeurs à substituer aux {} présents dans le contenu du 1er argument (plus de détails sur https://doc.rust-lang.org/std/fmt/) ;
- ; obligatoire en fin de ligne la plupart du temps en Rust ;

Modifions maintenant notre programme en déclarant une variable (https://is.gd/SlDHl3) :

```
fn main() {
    let une_chaine = "Bonjour Programmez!";
    println!("{}", une_chaine);
}
```

Les variables qui se déclarent avec le mot-clé let, sont immuables (que l'on ne peut pas réaffecter) et sont fortement typées. Pourtant vous remarquerez que nous n'avons pas déclaré le type de cette variable. En effet, Rust met en œuvre de l'inférence de type, à savoir qu'il tente de détecter le type des variables mais pas des paramètres. Un allègement significatif de la syntaxe qui facilite l'écriture et la relecture. Il faudra cependant parfois l'aider à deviner correctement le type. Remarquez enfin la syntaxe de type *snake case*, avec des _ pour séparer les concepts dans les noms des variables (par opposition à la syntaxe *camel case*: uneVariable). Pratique, car le compilateur râlera si vous ne respectez pas cette convention.

Lancez avec "Run" pour compiler et exécuter ce code.

Poursuivons la découverte des concepts de Rust en introduisant une nouvelle fonction qui permet d'effectuer la division d'un nombre par un autre :

```
fn calculer_division(x: i32, y: i32) -> i32 {
    println!("Numérateur: {}", x);
    println!("Dénominateur: {}", y);
    x / y
}
```

L'appel de la fonction que nous venons de déclarer s'effectue exactement comme l'on s'y attend (programme complet : https://is.gd/0U61pa) :

```
...

fn main() {
    let resultat = calculer_division(-4, 2);
    println!("Résultat : {}", resultat);
}
```

Par choix, il n'y a pas d'inférence de type en Rust lors de la déclaration des fonctions : les types des paramètres d'entrée et du retour doivent être explicitement spécifiés. Le type des arguments est indiqué après chaque nom de variable suivi de : et le type de retour est spécifié après la flèche ->. Rust propose une liste de types primitifs très complète, puisque vous avez par exemple la possibilité de choisir des entiers, des flottants *signés* ou *non signés*, des entiers ou des flottants de "longueur" variable.

Exemples :

-u8 : est un nombre de longueur 8 octets uniquement positif, donc un nombre compris entre 0 et 255 (2 -1) :

-i16 : est un nombre de longueur 16 octets, positif ou négatif, donc un nombre compris entre -32 768 (-2 ^ 16/2) et 32 767 (2 ^ 16/2-1).

Je vous conseille d'explorer la liste complète des types primitifs pour vous faire une idée des autres types disponibles (https://doc.rust-lang.org/book /primitive-types.html).

Les plus attentifs d'entre vous auront remarqué que cette fonction ne "retourne" explicitement rien comparé à d'autres langages et que la dernière ligne ne se termine pas par un point-virgule, alors que l'on a précédemment vu qu'il était obligatoire. En Rust, il y a un return implicite sur la dernière expression exécutée d'une fonction. Attention, ce n'est pas nécessairement la dernière ligne de code de la fonction. Dans l'exemple ci-dessous, il y a 2 façons de sortir de la fonction et aucune d'entre elles ne correspond à la dernière ligne de code de la fonction.

Exemple complet : https://is.gd/9LKj2d.

À propos du point-virgule, dans notre cas ici qui semble manquant, vous ne devez pas en mettre en fin de ligne. Essayez, vous aurez une erreur de compilation ! Pourquoi ? Rust est un langage basé sur les expressions et non sur des déclarations. Cela signifie que *tout* renvoie *quelque chose* : assigner une variable renvoie quelque chose, if renvoie quelque chose … L'expression x / y renvoie le résultat de la division, compatible avec le type i32, là où l'expression x / y; a un résultat dont le type est (), incompatible avec i32. Enfin, modifions une dernière fois notre programme pour induire une syntaxe plus habituelle pour les *Rustacéens* (traduction de *Rustaceans", le nom officiel des développeurs Rust) : le matching.

```
fn calculer_division(x: i32, y: i32) -> i32 {
    match y {
        0 => panic!("Division par 0"),
        1 => x,
        _ => x / y
    }
}

fn main() {
...
}
```

Exemple complet : https://is.gd/X7889d.

La syntaxe parle d'elle-même, c'est simple à comprendre. Le match permet de gérer plus de cas que ne peut le faire un simple if tout en rendant obligatoire le traitement du cas par défaut. Qui n'a jamais oublié un else ou un case default en Java, par exemple ? En Rust, c'est impossible, car le compilateur s'assure que tous les cas possibles de matching sont bien déclarés et gérés par le développeur. _ signifie au compilateur "tous les autres cas de matching" (dans notre cas, "tout sauf 0 et 1"). Sachez aussi que le matching offre aussi beaucoup plus de possibilités que ne montre ce simple exemple. •

Introduction au Web Scraping

1ère partie

Michael Bacci
michael.bacci.software@gmail.com
https://www.linkedin.com/in/michaelbacci
Senior Software Engineer R&D
Expert en HPC, C/C++, J2EE, 3D
Michael travaille dans des projets R&D pour l'industrie et instituts de recherche : universités Paris 13, Paris 7 et INRIA

Pour avoir les dernières news en temps réel, il suffit de regarder Twitter ou bien une application d'actualités sur son propre smartphone.
Certaines applications peuvent être configurées pour chercher certains contenus plutôt que d'autres. Mais comment aller au-delà des limites de ces outils (applications et sites Web) ? Comment est-il possible de chercher et extraire n'importe quelle information depuis le net ?
Le Web Scraping est la réponse.

Cet article explorera les bases de cette technique qui représente le fondement même des moteurs de recherche comme Google. Parmi les nombreux frameworks et librairies existants, on explorera le framework Python Scrapy. La ligne directrice qui pousse l'auteur de l'article, qui pratique le scraping aussi dans le milieu professionnel, est de faire en sorte que le lecteur puisse se mettre dans la peau, ou bien dans la tête, d'un Web scraper professionnel, et de donner les clefs pour comprendre comment pense un *hacker des informations*.

Les domaines d'applications

Les exemples d'usage du Web Scraping sont nombreux:

- Alerter d'un changement de prix d'un produit sur un site Web ;
- Comparer les prix de produits concurrents ;
- Consulter automatiquement des réputations sur des réseaux sociaux comme L*inkedin* et *Viadeo ;*
- Gérer votre popularité, ou celle d'une compagnie, sur le Web ;
- Chercher des copies de votre article ou d'un produit sur le net ;
- S'enrichir de nouveaux numéros de téléphone et emails pour envoyer des campagnes publicitaires ;
- Acheter automatiquement des produits : tickets de concert, vols last minute, ventes aux enchères...
- S'enregistrer en premier pour louer un terrain de tennis ;
- Connaître la meilleure cote d'un pari sportif en comparant plusieurs sites du secteur ;
- Etre averti pour un nouveau contenu, news ou changement de n'importe quel site.

Quand éviter le scraping

Il y a des domaines où le Web Scraping est déconseillé.
Prenons l'exemple d'un *bot* (programme qui sonde le Web en recherche d'information) qui extrait les dernières cotes de votre titre boursier préféré depuis l'une des nombreuses plateformes Web existantes. Ces informations ne seront pas en temps réel, chose fondamentale dans ce domaine! Avez vous envie de jouer votre argent avec des informations ni précises ni en temps réel ?
Evitez le Web Scraping quand vos besoins sont d'avoir :

- 100% de certitude dans les informations à extraire ;
- Une très grand performance d'acquisition des données.

Il est donc préférable dans certains cas d'utiliser des systèmes comme des API, librairies, requêtes HTTP/RESTful, etc., déjà existantes. Dans le cas de la finance, il existe plusieurs solutions.

Les fondements théoriques

Le Web Scraping se base sur l'extraction de données contenues dans des pages HTML. Le HTML a une structure hiérarchique, où ses éléments sont situés à l'intérieur d'autres éléments. Pour identifier les différents éléments dans une page HTML, on utilise des attributs, des noms et des identifiants uniques. C'est en cherchant les identifiants uniques et leur hiérarchie dans le code source des pages HTML que l'on extrait de façon automatique les informations d'un site Web. La première phase dans le métier du Web scraper, c'est donc de comprendre la structure dans laquelle se trouvent les informations qui l'intéressent.

Web Scraping et Web Crawling

Si le Web Scraping est orienté vers l'extrapolation des données de pages Web spécifiques, le Web Crawling est consacré à l'extraction automatique de données de n'importe quel site Internet.
Tous les jours, des *bots* créés par Google, Yahoo, Microsoft, etc., sondent les sites Web du monde entier pour :

- L'indexation de sites Web ;
- La mise à jour des références des moteurs de recherche ;
- Le calcul de la popularité d'un site Web - ce qui a fait la fortune de Google grâce à son algorithme nommé PageRank.

Ce sont ainsi des *bots* qui tournent 24h sur 24 sur des milliers de serveurs dans le monde qui nous permettent de trouver toutes les informations que nous cherchons tous les jours.
Le Web Scraping est donc un cas particulier du Web Crawling.

Comment se protéger ?

Les industriels sont parfois à la source de requêtes d'extrapolation automatique, mais en sont aussi parfois les victimes. Prenons l'exemple d'une boutique de vente en ligne : comment cacher les prix au concurrent Robot ? Comment empêcher que des *bots* volent les informations sur les produits et les utilisent pour offrir un meilleur prix ?
Voici des méthodes couramment utilisées :

- Enregistrer les adresses IP

  S'il y a un grand nombre de requêtes depuis un seul ordinateur/client, c'est fort probable qu'il s'agisse bien d'un programme de soustraction automatique des données. Dans ce cas, la stratégie est de bloquer l'accès de votre plateforme (ou site Web) à l'adresse IP qui vous surcharge de requêtes. Mais attention, vous risquez de bloquer aussi d'autres ordinateurs (clients potentiels) si l'adresse IP de la source des requêtes provient d'un sous-réseau au proxy, utilisé par des milliers d'autres utilisateurs réels.

- Utiliser CAPTCHA
  Le système CAPTCHA [Fig. 1] est un rétroacronyme qu'on pourrait traduire en français par "Test public de Turing complètement automatique ayant pour but de différencier les humains des ordinateurs". Le lecteur aura peut-être déjà rencontré cela en surfant sur le Web. Il existe des systèmes CAPTCHA plus ou moins fiables, mais en règle générale, les voleurs d'informations moyens s'éclipsent dès qu'ils tombent sur ce système sur un site Web. L'inconvénient de ce système c'est tout simplement que les gens ne l'aiment pas ; ils préfèrent donc utiliser des sites similaires où il n'y a pas ce *fastidieux machin*. Pensez donc le type d'utilisateur de votre site.

1

- Forcer un login
  Le développeur du système de scraping sera obligé d'implémenter une nouvelle fonctionnalité d'authentification, et, au moins, même si vous n'êtes pas capable de le bloquer tout de suite, vous pourrez l'identifier, et ensuite, grâce au système de log, reconnaître l'utilisateur suspect.
- Changer souvent l'HTML
  En utilisant des simples scripts, il est possible de changer les noms des attributs dans les fichiers HTML, css et javascript. Ainsi le *bot*, construit pour extraire les informations sur un site avec certain *tag* et *id* uniques, ne pourra plus fonctionner. Dans ce cas, le développeur du *bot* devra chercher les éléments qui ont changé pour mettre à jour son son code.
- Multimédia pour représenter les contenus
  Cette technique consiste à changer le contenu textuel des informations, avec des images et documents multimédia.. Voici des exemples :
  - Transformer le prix d'un produit en une image et afficher donc l'image plutôt que le texte.
  - Mettre les détails techniques dans une page pdf plutôt que dans un tableau HTML.

  Dans le premier cas l'inconvénient est un ralentissement du chargement des pages Web et dans le deuxième, l'ennui du lecteur qui n'aime pas forcément télécharger un pdf pour consulter les informations qu'il cherche.
- Solution extrême : Adobe Flash à la place d'HTML/CSS/Javascript
  Le scraping des données depuis une source Flash est réellement compliqué. Ce processus s'appelle *décompilation*, et il existe dans le commerce des produits pour le faire. Mais il existe également des paramètres avancés dans Flash pour l'obfuscation du code source. En outre, en utilisant Flash et des scripts automatiques, il est possible de changer très souvent le code de votre site Web en laissant intact le layout de votre site. L'ingénieur qui cherche à vous piquer vos informations en pratiquant du *reverse engineering*, devra recommencer son travail à chaque nouvelle version obfusquée qui sera mise en ligne.

Le Web Scraping et la loi

Le Web Scraping et le Web Crawling collectent des informations publiques et/ou à accès restreint par un login et mot de passe. Les données cherchées par ces programmes sont les mêmes données accessibles depuis n'importe quel navigateur Web. Ce sont des techniques informatiques qui, en général, n'enfreignent pas la loi. Des exceptions peuvent toujours exister. Par exemple, en l'an 2000, la compagnie eBay a cité en justice une entreprise concurrente pour avoir utilisé un *bot*. En effet, dans les termes d'accès à la plateforme de vente en ligne il est bien mentionné le fait qu'aucun *robot, spider ou système équivalent* ne peuvent être utilisés pour accéder à la plateforme. Renseignez-vous donc sur la loi des pays concernés et sur les clauses d'accès du site cible avant de construire vos programmes. Le Web Scraping et Web Crawling ne sont pas non plus des techniques de hacking, car les informations recherchées ne sont pas "volées'", mais juste acquises de façon automatique, sans l'utilisation directe d'un navigateur Web. Dans le cas du Web Scraping avec accès à un site par login et password, on présume que les accès login et password, on les a eus de façon légale !
Une des techniques de hacking les plus répandues c'est le DOS (Denial Of Service) qui a comme but de surcharger un serveur Web avec un nombre très élevé de requêtes. Sur le Net on trouve des critiques du Web Scraping pour ses liens avec des "incidents" de DOS ; des *bots* mal programmés qui ont surchargé des requêtes sur un serveur Web causant un DOS. Mais le Web Scraping n'a aucun intérêt à causer un DOS, car, si le site Web cible est hors service, les données cherchées par le *bot* sont momentanément indisponibles.

Web Scraping et Python

En faisant une recherche du mot "Web Scraping" dans le site github.com [Fig. 2], on s'aperçoit en un clin d'oeil que le langage Python est le plus utilisé dans ce domaine. L'explication en est très simple.
Le Web Scraping est par nature une activité qui demande beaucoup de mises à jour dans le code, car en général un site Web est mis à jour très fréquemment. Le contenu des sites change, le layout se transforme, de nouveaux contenus apparaissent, des changements de liens se font, etc. Pour faire face à une telle problématique, il n'y a rien de mieux qu'un langage dynamique, et Python s'est imposé dans ce domaine sans aucune difficulté. Mon avis personnel, c'est que Python s'est largement imposé par rapport à d'autres langages dynamiques comme *php* et *ruby* sur ce domaine, car Python a été développé suivant une idée très simple : l'interaction dynamique avec l'utilisateur en ligne de commande.

2

Le travail principal dans le Web Scraping est la construction des requêtes Web et la recherche des identifiants uniques dans la hiérarchie HTML des informations à extraire. Python a des librairies très puissantes et faciles d'usage pour ces deux tâches, ce qui le rend le candidat idéal.

Le principe de Darwin

Dans le Web Scraping, la priorité c'est l'adaptation au changement !
Les questions de haute performance, comme la rapidité de téléchargement des données (HTML, css, js, images…) et l'extraction du contenu sont secondaires. On cherche avant tout la facilité de la réécriture du code en raison du changement quasi inévitable du site Web ciblé.

Haute performance et latence des requêtes

Un programme de Web Scraping écrit en langage C, et fortement optimisé, n'apportera pas un avantage significatif qui justifierait l'énorme effort de développement par rapport au même programme écrit en langage Python.
Voici les paramètres qui permettent de faire la comparaison :

- Nombre de lignes de code : Python fortement gagnant sur C.
- Création du make/cmake, linking et nombre de dépendances externes : Python gagne encore.
- Latence des requêtes: aucun langage de programmation ne peut

influencer la latence des requêtes Web.

- Extraction des données : le C n'est même pas deux fois plus rapide que son équivalent en Python, car python peut utiliser des wrappings sur les librairies écrites en C.
- Essai réel et adaptation du code : c'est ici que Python montre ses muscles et permet de tester à la volée si le code écrit fonctionne dans son vrai contexte Web : authentification, gestion des sessions et cookies, pipeline des requêtes, interprétation Javascript, extraction des données.

Dans cette comparaison, on arrive aussi à démontrer facilement que le temps de développement est largement inférieur avec Python, et jusqu'ici rien de surprenant : les avantages du langage dynamique par rapport à des langages statiques sont bien connus. Le but est de mettre l'accent sur la partie la plus importante du développement d'une application de Web Scraping : l'adaptation et ses essais en continu sur le site Web ciblé.

La latence est aussi un élément important, car ni en Python ni en C, on ne peut forcer le téléchargement d'un page Web plus rapidement. Les autres éléments à comparer sont donc la construction de la requête et l'extraction des données, qui ne sont pas très coûteuse en termes de calcul.

Web Scraping tools pour non-développeur

Il existe des plateformes Web et des logiciels, capables d'extraire de façon intelligente des données Web, de la même façon qu'un programme de Web Scraping. Ces outils sont programmés de façon générale pour extraire n'importe quel contenu, ou presque, d'un site Web. A la place du code, ces outils acceptent comme paramètres, des informations comme :

- L'URL du site/page Web ;
- Login/password si nécessaire ;
- Les différentes étapes et input pour rejoindre une certaine page Web.

Ces tools acceptent aussi de façon intelligente les éléments à extraire, simplement en sélectionnant des éléments sur les pages Web à l'aide de la souris, rien de plus simple. Ils peuvent aussi extraire, et donc transformer, les contenus des pages Web en fichier excel/csv/txt.

La limite de ces tools c'est qu'ils ne sont pas capables d'extraire la totalité des contenus de façon automatique de tous les sites Web existants. Il y aura toujours des contenus de certain sites Web que ces tools ne seront pas capables d'extraire.

Seul un développeur serait capable de créer un programme qui aille au-delà de la limite de ces tools. Il y a des cas où même un développeur n'est pas en mesure d'extraire des données de façon automatique, lire la section *"Comment se protéger ?"*. Le choix entre payer un développeur qui construise une solution *ad hoc* et acheter un tools de Web Scraping peut être résolu en suivant la check list suivante :

PRO Developpeur :

- Je suis sûr d'avoir mes données extraites ;
- Je ne doit pas perdre du temps à apprendre ces tools ;
- Je ne risque pas que le tool ne soit pas capable d'extraire les données dont j'ai besoin ;
- Il y a urgence à extraire les données.

PRO Tools :

- Un coût économique limité ;
- Pour des sites simples (c'est à dire, avec une structure HTML "pas trop" hiérarchique) les tools fonctionnent très bien ;
- Je peux demander assistance.

CONTRE Développeur :

- Si le site ciblé effectue des changements réguliers de sa structure HTMLHTML/CSS, je dois faire de nouveau appel au développeur afin qu'il modifie vite le programme, ce qui implique :
  - Un coût pour chaque mise à jour ;
  - Un délai de mise à jour si le développeur n'est pas disponible tout de suite.

CONTRE Tools :

- Le temps d'apprentissage de l'outils ;
- L'échec dû à ses limites pour extraire les données de ma cible.

Voici une liste des tools conseillés, parmi les multiples solutions existantes sur le net :

- Framework Web :
  - scrapinghub.com : il utilise le framework scrapy que l'on utilisera dans cet article ;
  - import.io : il existe aussi la version desktop téléchargeable ;
  - kimonolabs.com : avec une interface très simple et intelligente.
- Logiciel desktop :
  - uipath.com : très complet, pour des vrais professionnels et grandes entreprises ;
  - ubotstudio.com : il résout les captcha, permet une écriture automatique des *form* HTML et beaucoup plus encore.

Les outils du métier

La boîte à outils du métier de web scraper est constituée, dans ses éléments fondamentaux, de ces composants :

Le langage d'interrogation XPath

- Sert à extraire ou identifier, une portion du contenu depuis une page en format Xml ; en appliquant la syntaxe de ce langage d'interrogation, des librairies ont été créées pour permettre d'extraire du contenu aussi depuis une page en format HTML.

Un langage d'expression régulière parmi Posix, ECMA, Python (extension de Posix), etc.

- Couteau suisse du traitement de texte : il localise et extrait des sous séquence de caractères ou mots dans un texte. Tout bon programmeur a besoin de connaître comment écrire une expression régulière.

Les langages HTML/Dom, Css et Javascript

- HTML, c'est la base pour comprendre les données que l'on veut extraire, et le DOM est la représentation interne du code HTML dans un Browser qui permet de repérer dans un code HTML chaque élément
- Le CSS nous donne les clefs de l'aspect d'une page Web, très utile pour le web scraper quand il s'agit d'extraire un texte avec un code css particulier.
- Le langage Javascript qui dans le contexte du scraping est utilisé pour :
  - Faire du *reverse engineering* du code et comprendre la nature des applications de type RESTful ou l'API de communication entre le client (browser) et le server
  - Charger dynamiquement les données de la page Web (Ajax) ;
  - Reproduire exactement le comportement de la page Web comme si elle était chargée à partir d'un navigateur Web ;

Protocols Http et Https

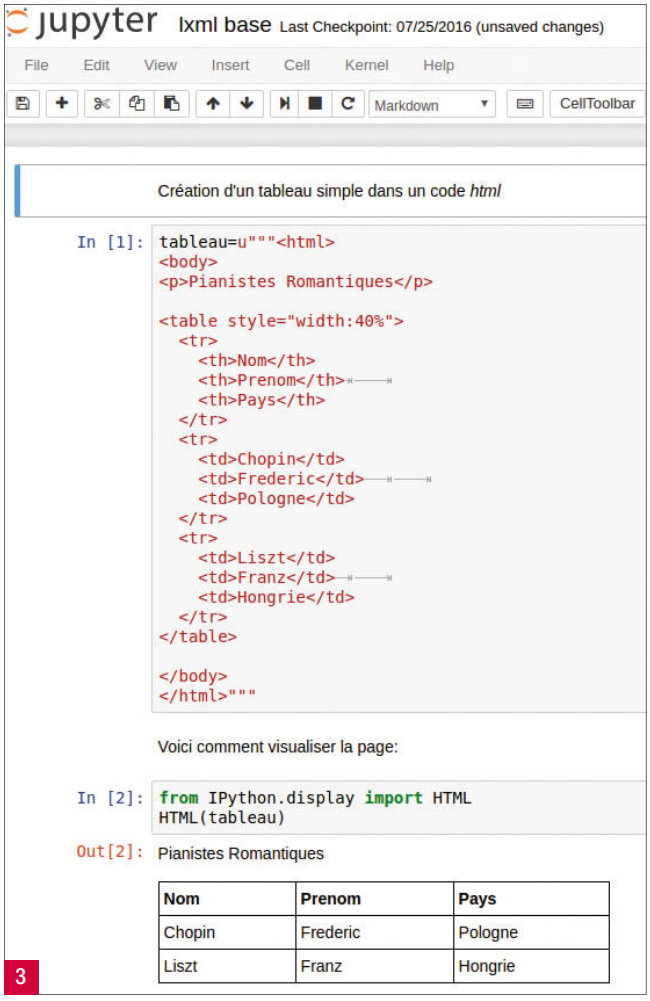
- Rien n'est plus essentiel que de construire sa propre requête Web, comprendre les *headers* des paquets Http et savoir lire la RFC (Request For Comment) 2616 qui décrit en détail le protocole Http 1/1.

Sessions et Cookies

- Parfois tout au long d'une procédure d'extraction, les concepts de session et cookie accompagnent le web scraper dans son travail : simuler l'identité de quelqu'un pour pouvoir ensuite interroger le serveur Web et extraire les données recherchées.

Moteur Javascript

- Une fois franchie l'étape de la récupération des données depuis le serveur Web ciblé, ces données (HTML + Css + Javascript + donnée multimedia) ont besoin d'être mélangées ensemble comme dans un vrai



jupyter lxml base Last Checkpoint: 07/25/2016 (unsaved changes)

File Edit View Insert Cell Kernel Help

Markdown CellToolbar

Création d'un tableau simple dans un code *html*

In [1]:
```
tableau=u"""<html>
<body>
<p>Pianistes Romantiques</p>

<table style="width:40%">
  <tr>
    <th>Nom</th>
    <th>Prenom</th>
    <th>Pays</th>
  </tr>
  <tr>
    <td>Chopin</td>
    <td>Frederic</td>
    <td>Pologne</td>
  </tr>
  <tr>
    <td>Liszt</td>
    <td>Franz</td>
    <td>Hongrie</td>
  </tr>
</table>

</body>
</html>"""
```

Voici comment visualiser la page:

In [2]:
```
from IPython.display import HTML
HTML(tableau)
```

Out[2]: Pianistes Romantiques

Nom	Prenom	Pays
Chopin	Frederic	Pologne
Liszt	Franz	Hongrie

3

La librairie **lxml** permet de faire le *parsing* du code html

In [3]:
```
from io import StringIO
from lxml import etree
#creation du document lxml
doc = etree.parse(StringIO(tableau))
```

Comme premier exemple, voici une extraction des noms des colonnes du tableau:

In [4]:
```
doc.xpath('//table/tr/th/text()')
```

Out[4]: `['Nom', 'Prenom', 'Pays']`

La syntaxe d'interrogation:

1. **'//table'** indique la recherche d'un élément avec un nom 'table' à partir de la racine du document
2. une fois dans un élément ou branche 'table', on passe à la recherche du code html qui identifie les noms des colonnes, c'est à dire la couple **(tr,th)**
3. la fonction **text()** permet enfin de retourner le text correspondant pour chaque colonne trouvée

La nature des résultats:

Le type retourné est un **list** contenant des **objets** de la classe **lxml**.
Ces objets ont une représentation de string car ils sont été convertis en utilisant la fonction **text()** dans la query d'interrogation.

Pour comprendre la nature des objects retourné depuis une query,
voici un exemple qui extrait les objets lxml qui représentent chaque ligne du tableau

In [5]:
```
doc.xpath('//table/tr')
```

Out[5]:
```
[<Element tr at 0x7fc07c0b5b48>,
 <Element tr at 0x7fc07c0b5b90>,
 <Element tr at 0x7fc07c0b5c68>]
```

In [6]:
```
type(doc.xpath('//table/tr')[0])
```

Out[6]: `lxml.etree._Element`

In [7]:
```
#Extraction des lignes
for ligne in doc.xpath('//table/tr[position()>1]'):
    print(ligne.xpath('td/text()'))
```

```
['Chopin', 'Frederic', 'Pologne']
['Liszt', 'Franz', 'Hongrie']
```

In [8]:
```
#Extraction des seuls noms
doc.xpath('//table//tr//td[1]//text()')
```

Out[8]: `['Chopin', 'Liszt']`

4

browser Web. Ce qui fait le mélange de tous ces composants c'est normalement le moteur Javascript contenu dans chaque browser. Dans le cas du scraping, le moteur Javascript est un logiciel sans interface graphique dédié à cette tâche.

Architecture Ajax

- Les requêtes asynchrones exécutées grâce au moteur Javascript sont déclenchées lors d'une série d'évènements sur une page Web. Par exemple, lorsque sur un site de voyage on fait le choix du pays de départ, la liste des destinations possibles est automatiquement chargée. Connaître le mécanisme asynchrone de communication permet donc de pouvoir extraire les informations aussi sur des sites qui exploitent cette technologie.

Exemple avec XPath

Pour mieux montrer comment fonctionne le langage XPath, on se sert de la fonctionnalité notebook d'iPython : [Fig. 3 et 4]. •

Suite dans Programmez! 202

Une **application Electron** pour suivre un repository GitHub

2e partie

• Philippe Charrière
Solution Engineer chez **GitHub** (@k33g)
Twitter: @k33g_org

Dans ce deuxième article sur Electron, je souhaite aller un peu plus loin en utilisant :

- Un framework "typé composants" pour tout ce qui est affichage. J'ai choisi **Riot.js** (http://riotjs.com/) car il est léger et simple ;
- Un framework CSS pour obtenir une IHM plus sympatique. J'ai choisi **Semantic-UI** (http://semantic-ui.com/) parce que les goûts et les couleurs, cela ne s'explique pas ... ;
- Les API GitHub, pour, dans un 1er temps, être notifié lorsqu'il se passe quelque chose sur un repository, puis dans un 2ème temps, changer les statuts de "merge" d'une pull request sur mon repository.

PRÉAMBULE(S)

Prérequis

Pour vous faire gagner du temps, je vous ai fait un petit "starter kit" que vous trouverez ici : https://github.com/k33g/electron-walking-skeleton.
Pour l'installer, c'est très simple :

```
git clone https://github.com/k33g/electron-walking-skeleton.git your-application-name
cd your-application-name
npm install
bower install
```

Si vous avez lu mon précédent article, cela peut vous aider aussi.

Découverte (très rapide) de Riot

Riot est extrêment simple à appréhender, il est très intuitif, donc vous n'aurez aucun problème à lire mon code. Riot est un framework qui permet de définir des composants d'affichage pour votre page ou application HTML. Voyez Riot comme un Polymer allégé. Un composant Riot (on parle de tag en fait) est dans un fichier .tag (mais vous pouvez donner l'extension que vous voulez, en ce qui me concerne, j'utilise .html) et ressemble à ceci :

```
<my-title>
  <h1>{titleContent}</h1>
  <script>
    this.titleContent = "Hello World!";
  </script>
</my-title>
```

Pour l'utiliser dans votre page html, c'est tout simple :
Vous déclarez le composant dans la page :

```
<script src="my-title.html" type="riot/tag"></script>
```

Vous utilisez votre nouveau tag (<my-title>) comme ceci :

```
<body>
  <my-title></my-title>
</body
```

Et vous demandez à Riot de monter le tag dans la page, comme cela :

```
riot.mount('my-title')
```

Passer des informations à un tag

Si vous voulez échanger des informations avec les tags, vous pouvez leur passer des données comme ceci :

```
let tool = {
  hello: () => {
    return "hello";
  }
}
riot.mount('my-title', {tool: tool, text: "message"});
```

Et l'utiliser comme ceci :

```
<my-title>
  <h1>{titleContent}</h1>
  <script>
    this.titleContent = opts.tool.hello() + ":" + opts.text;
    // on obtiendra <h1>hello:message</h1>
  </script>
</my-title>
```

Remarque : opts est un objet spécifique à Riot. Chaque tag possède une propriété opts.

Electron avec Riot

Voyez ceci comme un 1er exercice (vous pouvez utiliser mon starter-kit pour essayer). Voici donc comment j'ai procédé pour faire fonctionner Electron et Riot ensemble (j'ai installé Riot avec Bower, mais vous pouvez très bien le faire manuellement).
Je référence Riot dans la page index.html de mon application :

```
<script src="bower_components/riot/riot+compiler.min.js"></script>
```

Je monte mes tags Riot dans renderer.js comme ceci :

```
riot.mount('display-message', {ipcRenderer: ipcRenderer} )
riot.mount('button-send-message', {ipcRenderer: ipcRenderer} )
```

Vous remarquerez que je passe ipcRenderer à mes tags, ils pourront ainsi communiquer avec l'application Electron (ou même entre eux) (vous pouvez voir ipcRenderer comme un "broker" de message).

L'IHM envoie un message

Par exemple, mon tag button-send-message peut envoyer des messages à l'application grâce à ipcRenderer:

```
<button-send-message>
  <button onclick={send}>Send message to Electron</button>

  <script>
    send (event) {
      opts.ipcRenderer.send('message-hello',{message:"yo"});
    }
```

```
  </script>
</button-send-message>
```

Et l'application s'abonne à message-hello (dans main.js) comme ceci :

```
ipcMain.on('message-hello', (event, arg) => {
  console.log("hello", arg);
});
```

L'IHM reçoit un message

Dans ce cas-là, c'est mon tag qui s'abonne à un topic (par exemple 'message-random'):

```
<display-message>
  <hr>
  <h2>{messageContent}</h2>
  <hr>
  <script>
    opts.ipcRenderer.on('message-random', (event, arg) => {
      this.messageContent = arg;
      this.update(); // mettre à jour le composant
    });
  </script>
</display-message>
```

Et l'application notifiera le composant (dans main.js) comme ceci :

```
app.on('ready', () => {
  mainWindow = new BrowserWindow();
  mainWindow.webContents.loadURL(`file://${__dirname}/index.html`);

  mainWindow.webContents.on('did-finish-load', () => {

    setInterval(() => {
      // notifier toutes les 2 secondes
      mainWindow.webContents.send("message-random", Math.random());
    }, 2000)

  })
});
```

Maintenant que vous avez compris comment utiliser Riot avec Electron (si si!), passons à l'utilisation de l'API GitHub avec JavaScript.

Utilisation de l'API GitHub

Création d'un "client" GitHub

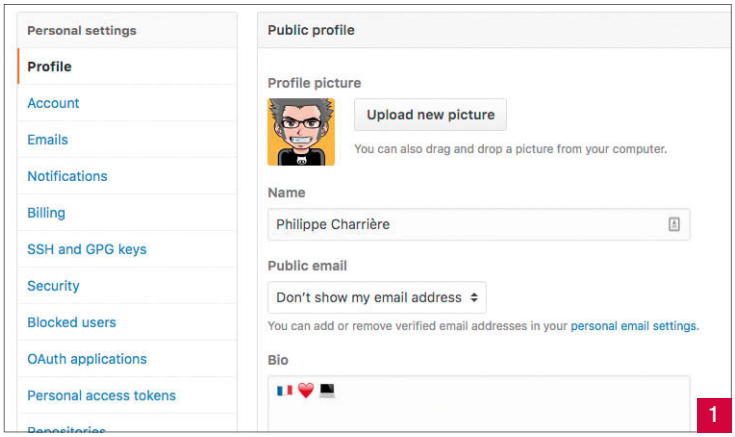
Nous aurons besoin "d'interroger" les API REST GitHub, pour cela je vais utiliser le polyfill **node-fetch** (https://www.npmjs.com/package/node-fetch). Pour l'installer, c'est tout simple :

```
npm install node-fetch --save-dev
```

Ensuite, j'aurai besoin que ma future application puisse s'authentifier sur GitHub (pour faire des requêtes de type POST par exemple), et pour cela j'utiliserai les **Personal access tokens**.

Création d'un token

Pour créer un "Personal access token", vous devez aller sur votre profil GitHub :
Token : *persotoken00-2 [Fig.1]*
Puis choisir "**Personal access tokens**" dans le menu latéral de gauche. Ensuite :



- Cliquer sur "**Generate new token" ;**
- Donner une description de votre token ;
- Sélectionner la case "**repo";**
- Cliquer en bas de page sur "**Generate token" ;**
- Copier le token généré pour pouvoir l'utiliser.

Utilisation du token

Vous pouvez utiliser le token de différentes manières. En ce qui me concerne (et pour cet article), j'utilise les variables d'environnement :

```
# === GitHub ===
export TOKEN_GITHUB_DOT_COM=<MON_TOKEN>
export TOKEN_GITHUB_ENTERPRISE=<MON_AUTRE_TOKEN>
```

Et ensuite côté JavaScript, j'utiliserai ceci pour récupérer mon token (donc pour lire ma variable d'environnement) :

```
process.env.TOKEN_GITHUB_DOT_COM
```

Code côté client

Nous pouvons donc maintenant écrire notre client (GitHubClient.js) qui nous servira à interroger l'API :
Remarque: j'ai fait au plus simple, mais n'hésitez pas à me contacter en cas de problème (ou pour proposer des améliorations).

```
const fetch = require('node-fetch');

class GitHubClient {
  constructor({baseUri, token}) { //eg token = process.env.TOKEN_GITHUB_DOT_COM
    this.baseUri = baseUri
    this.credentials = token !== null && token.length > 0 ? "token" + ' ' + token : null
    this.headers = {
      "Content-Type": "application/json",
      "Accept": "application/vnd.github.v3.full+json",
      "Authorization": this.credentials
    }
  }

  getData({path}) {
    let _response = {}
    return fetch(this.baseUri + path, {
      method: 'GET',
      headers: this.headers
    })
    .then(response => {
```

```
    // save reference of response / then we can access to headers
    _response = response
    // if response is ok transform response.text to json object
    return response.ok ? response.json() : null;
   })
   .then(jsonData => {
    // add json data to _response
    _response.data = jsonData;
    return _response;
   })
 }

 postData({path, data}) {
  let _response = {}
  return fetch(this.baseUri + path, {
   method: 'POST',
   headers: this.headers,
   body: JSON.stringify(data)
  })
  .then(response => {
   _response = response
   return response.ok ? response.json() : null;
  })
  .then(jsonData => {
   _response.data = jsonData;
   return _response;
  })
 }
}

module.exports = GitHubClient
```

Utilisation de notre client

Imaginons que je veuille avoir des informations sur moi-même (mon handle GitHub est @k33g), j'utiliserai mon client de la façon suivante :

```
const GitHubClient = require('./js/GitHubClient')

let githubCli = new GitHubClient({
 baseUri:"https://api.github.com",
 token:process.env.TOKEN_GITHUB_DOT_COM
});

// get user information
githubCli.getData({path:"/users/k33g"})
.then(response => {
 console.log(response.data);
})
.catch(error => {
 console.log("error", error)
})
```

Et j'obtiendrai quelque chose comme ceci :

```
{
 login: 'k33g',
 id: 491848,
 avatar_url: 'https://avatars.githubusercontent.com/u/491848?v=3',
 gravatar_id: '',
 url: 'https://api.github.com/users/k33g',
 html_url: 'https://github.com/k33g',
 followers_url: 'https://api.github.com/users/k33g/followers',
 following_url: 'https://api.github.com/users/k33g/following{/other_user}',
 gists_url: 'https://api.github.com/users/k33g/gists{/gist_id}',
 starred_url: 'https://api.github.com/users/k33g/starred{/owner}{/repo}',
 subscriptions_url: 'https://api.github.com/users/k33g/subscriptions',
 organizations_url: 'https://api.github.com/users/k33g/orgs',
 repos_url: 'https://api.github.com/users/k33g/repos',
 events_url: 'https://api.github.com/users/k33g/events{/privacy}',
 received_events_url: 'https://api.github.com/users/k33g/received_events',
 type: 'User',
 site_admin: true,
 name: 'Philippe Charrière',
 company: 'GitHub',
 blog: 'http://www.k33g.org',
 location: 'Lyon (France)',
 email: null,
 hireable: null,
 public_repos: 85,
 public_gists: 232,
 followers: 155,
 following: 73,
 created_at: '2010-11-22T13:35:09Z',
 updated_at: '2016-08-12T09:51:08Z'
}
```

Un code plus intéressant avec les GitHub API events

GitHub fournit une API pour "détecter" les événements (push, pullrequest, issue,...) optimisée pour le polling: https://developer.github.com/v3/activity/events/. Et elle est simple à utiliser.

Il faut utiliser la propriété ETag dans le header qui permet lors du polling de l'API de recevoir 304 Not Modified dans la réponse s'il n'y a rien de nouveau (s'il y a du neuf vous recevrez un code 200). L'avantage de cette méthode, c'est que la limite permise du nombre de requêtes n'est pas impactée (tant qu'il n'y a pas de changement). A la 1ère requête vous obtiendrez aussi dans le header la propriété X-Poll-Interval qui vous donne en secondes l'intervalle de temps permis (Pour vos tests vous pouvez diminuer l'intervalle qui est généralement de 60 secondes. Pour quelque chose de plus instantané, il y a le système des webhooks, mais cela sera pour une autre fois).

Polling

Je souhaite être notifié de ce qui se passe sur un repository en particulier. Voyons donc comment procéder. Je vais donc utiliser cette API https://developer.github.com/v3/activity/events/#list-repository-events qui consiste à faire une requête de ce type :

```
GET /repos/:owner/:repo/events
```

Donc pour "surveiller" mon repository sandbox-api, j'interrogerai /repos/k33g/sandbox-api/events. Au niveau code, nous aurons ceci :

```
const GitHubClient = require('./js/GitHubClient')

// création du client GitHub
let githubCli = new GitHubClient({
 baseUri:"https://api.github.com",
 token:process.env.TOKEN_GITHUB_DOT_COM
```

```
});

// polling de l'API toutes les 10 secondes (10000ms)
setInterval(() => {
 githubCli.getData({path:"/repos/k33g/sandbox-api/events"}).then(response => {

  githubCli.headers["If-None-Match"] = response.headers.get('etag')
  console.log(response.status, response.ok)
  /*
    si changement, affichera: 200 true
    sinon, affichera: 304 false
   */


  if(response.status == 200) {// Il y a eu un changement

   // Récupérer le dernier event
   let lastEvent = response.data[0];

   // Tester le type d'event
   if(lastEvent.type == "CreateEvent") {
    console.log("CreateEvent", response.data)
   }

   if(lastEvent.type == "PullRequestEvent") {
    console.log("PullRequestEvent", response.data)
   }

   if(lastEvent.type == "PushEvent") {
    console.log("PushEvent", response.data)
   }

  }

 })
 .catch(error => {
  console.log(">>> error:", error)
 })
}, 10000)

/*
Normalement le délai est de 60 secondes
Il est possible de le récupérer de cette manière:

  githubCli.getData({path:"/repos/k33g/sandbox-api/events"}).then(response => {
    let delay = response.headers.get('X-Poll-Interval')
    console.log("delay:", delay * 1000)
  })
*/
```

Remarque: l'utilisation de `githubCli.headers["If-None-Match"] = response.headers.get('etag')` permet de mettre en place la détection des changements (vous trouverez plus d'information ici https://en.wikipedia.org/wiki/HTTP_ETag). Pour résumer :

- La 1ère fois, vous faites une requête à l'API ;
- Vous allez recevoir un code 200 avec un ETag d'une certaine valeur ;
- La fois suivante, vous requêtez l'API mais avec dans les headers la propriété "If-None-Match" ;
- Si rien ne s'est passé, le serveur va retourner un code 304 ;
- Et le client ira chercher ses informations dans son cache.

Vous pouvez déjà vous créer un repository sur GitHub et tester ce bout de code avec **NodeJS** en ligne de commande (bien sûr, n'oubliez pas de modifier "/repos/k33g/sandbox-api/events" avec vos propres informations). Dès que vous procéderez à une modification sur votre repository, vous serez notifiés.

Récupérer plus d'informations

Ce qui m'intéresse, c'est essentiellement d'être notifié au moment d'un "PushEvent" (dès qu'il y a une synchronisation sur mon repository). Par exemple, je voudrais savoir sur quelle branche a eu lieu la modification et avoir la liste des commits associés. Je vais donc modifier mon code de la façon suivante :

```
setInterval(() => {
 githubCli.getData({path:"/repos/k33g/sandbox-api/events"}).then(response => {

  githubCli.headers["If-None-Match"] = response.headers.get('etag')

  if(response.status == 200) {
   let lastEvent = response.data[0];

   if(lastEvent.type == "PushEvent") {
    let ref = lastEvent.payload.ref.split("refs/heads/")[1];

    console.log(">>> ref:", ref);
    console.log(">>> commits:", lastEvent.payload.commits);
   }
  }
 })
 .catch(error => {
  console.log(">>> error:", error)
 })
}, 10000)
```

Si vous lancez le code, vous obtiendrez quelque chose comme cela :

```
>>> ref: master
>>> commits: [ { sha: 'a59cab1346f9629553681a2e920978af6ebdd9f6',
  author: { email: 'k33g@github.com', name: 'Philippe Charrière' },
  message: 'Update hello.md',
  distinct: true,
  url: 'https://api.github.com/repos/k33g/sandbox-api/commits/a59cab1346f96
29553681a2e920978af6ebdd9f6' } ]
```

NOTRE APPLICATION... ENFIN!

Je voudrais donc que mon application **Electron** m'affiche les commits lorsqu'il y a une synchronisation sur mon repository.

Préparation

Important : tout d'abord, vous devez vous créer un repository sur GitHub, et appelez-le sandbox-api, c'est votre repository de test.
Ensuite, si vous avez lu depuis le début, vous avez dû installer mon petit starter-kit pour gagner du temps. Sinon, c'est le moment :

```
git clone https://github.com/k33g/electron-walking-skeleton.git votre-application
cd votre-application
```

```
npm install
bower install
```

Vous devriez obtenir une arborescence comme celle-ci :

```
votre-application
  node_modules
  app
    bower_components/
    js/
    index.html
    main.js
    renderer.js
  tags
  package.json
```

Tout d'abord, copiez GitHubClient.js dans app/js, puis modifiez index.html.

Modification d'index.html

- Déclarer le tag que nous allons créer pour afficher les commits: <script src="tags/grid-events.html" type="riot/tag"></script>
- Utiliser le tag <grid-events></grid-events> dans le <body>

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width,initial-scale=1">
 <title>Application</title>
 <script src="bower_components/riot/riot+compiler.min.js"></script>
 <!-- include the tags -->
 <script src="tags/grid-events.html" type="riot/tag"></script>

 <!-- Semantic-UI -->
 <link rel="stylesheet" type="text/css" href="bower_components/semantic/dist/semantic.min.css">
 <script src="bower_components/jquery/dist/jquery.js"></script>
 <script src="bower_components/semantic/dist/semantic.min.js"></script>

</head>
<body class="ui container">
 <h1>Application</h1>

 <grid-events></grid-events>

 <script>
  require('./renderer');
 </script>

</body>
</html>
```

Modification de main.js

Nous allons modifier le code de main.js de la façon suivante : *(explications dans le code)*

```
const {app, ipcMain, BrowserWindow} = require('electron');
const GitHubClient = require('./js/GitHubClient')

/*
  Création du client GitHub:
*/

let githubCli = new GitHubClient({
 baseUri:"https://api.github.com",
 token:process.env.TOKEN_GITHUB_DOT_COM
});

/*
  Création d'une méthode pour interroger les events du repository:
  la variable `sender` correspond à l'objet `mainWindow.webContents` de
l'application Electron, qui permettra d'envoyer des messages à l'objet
`ipcRenderer` de `renderer.js`
*/

let getEvents = ({owner, repo, sender}) => {
 githubCli.getData({path: `/repos/${owner}/${repo}/events`}).then(response => {
  githubCli.headers["If-None-Match"] = response.headers.get('etag')

  if(response.status == 200) {
   let lastEvent = response.data[0];

   if(lastEvent.type == "PushEvent") {
    let ref = lastEvent.payload.ref.split("refs/heads/")[1];

    /*
     A chaque "PushEvent" j'envoie des informations sur le topic "PushEvent"
     */

    sender.send("PushEvent", {ref: ref, commits: lastEvent.payload.commits})
   }
  }
 })
 .catch(error => {
  console.log(">>> error:", error)
 })
}

app.on('ready', () => {
 mainWindow = new BrowserWindow();
 mainWindow.webContents.loadURL(`file://${__dirname}/index.html`);

 mainWindow.webContents.on('did-finish-load', () => {

  /*
   Une fois mon applicaion "chargée", je commence le polling de l'API
   */

  setInterval(() => {
   getEvents({owner: "k33g", repo: "sandbox-api", sender: mainWindow.webContents})
  }, 10000) // réduire le délais pour les tests
 })
});

ipcMain.on('close-main-window', (event, arg) => {
 app.quit();
});
```

Remarque importante : j'ai utilisé mon propre handle GitHub, n'oubliez pas d'utiliser le vôtre à la place (getEvents({owner: "k33g", repo: "sandbox-api" ...).

Ce qu'il faut retenir : c'est avec l'objet mainWindow.webContents de l'application Electron que nous pourrons communiquer les informations de commits avec la commande sender.send("PushEvent", {ref: ref, commits: lastEvent.payload.commits}).

Modification de renderer.js

renderer.js va essentiellement nous servir à "monter" notre tag riot, tout en lui passant ipcRenderer qui permettra d'écouter les messages en provenance de mainWindow.webContents.

```
// modules pour l'envoi de messages et la construction du menu
const {ipcRenderer, remote} = require('electron');

riot.mount('grid-events', {ipcRenderer: ipcRenderer} )
```

Et enfin notre tag d'affichage des commits

Dans le répertoire tags de l'application, créez un fichier grid-events.html avec le contenu suivant :

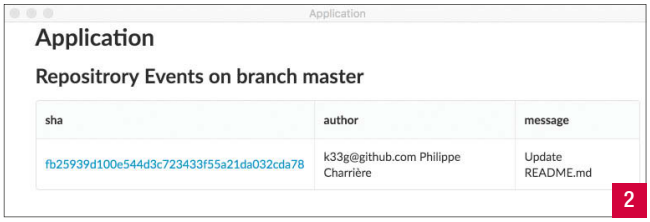
```
<grid-events>
 <h2>Repositrory Events on branch {ref}</h2>

 <table class="ui celled table">
  <thead>
   <tr>
    <th>sha</th>
    <th>author</th>
    <th>message</th>
   </tr>
  </thead>
  <tbody>
   <tr each={commits}>
    <td><a href={url} target="_blank">{sha}</a></td>
    <td>{author.email} {author.name}</td>
    <td>{message}</td>
   </tr>
  </tbody>
 </table>

 <script>
  opts.ipcRenderer.on('PushEvent', (event, arg) => {
   this.ref = arg.ref
   this.commits = arg.commits
   this.update();
  });
 </script>
</grid-events>
```

Le code est très simple : ipcRenderer que l'on récupère via la propriété opts du tag, écoute sur le topic 'PushEvent' et met à jour l'affichage dès qu'il est notifié.

Il ne nous reste plus qu'à lancer notre application avec la commande npm start. Faites des ajouts, modifications dans votre repository et vous verrez des changements dans votre application :

1er commit: [Fig.2]

Suite dans Programmez 202



MODÉLISATION DES DONNÉES
1ere partie

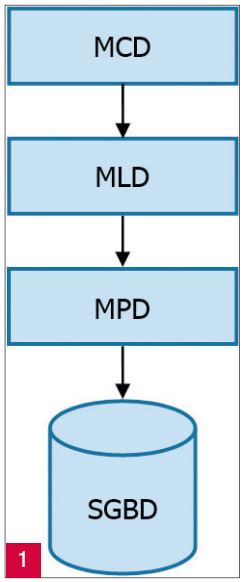
SMART Model et génération de code pour une base de données relationnelle

Steve Berberat
Collaborateur scientifique
Haute école de gestion Arc, HES-SO
steve.berberat@he-arc.ch

Pierre-André Sunier
Professeur HES
Haute école de gestion Arc, HES-SO
pierre-andre.sunier@he-arc.ch

L'informaticien de gestion, un jour ou l'autre durant sa carrière, est forcément mené à travailler sur une base de données relationnelle. Les données structurées, qu'elles soient à l'intérieur de l'entreprise ou dans le Cloud, restent nécessaires au fonctionnement des applications de gestion développées sur mesure.
Pour créer, mettre à jour ou compléter une base de données, deux solutions s'offrent à lui : écrire le code SQL à la main ou utiliser un outil qui va lui permettre de modéliser sa base de données pour ensuite automatiquement générer le code SQL.

Dans cet article, nous allons vous présenter un concept qui va plus loin que cette seconde variante. Le modèle dessiné dans l'outil deviendra SMART, beaucoup plus riche. Il décrira toute une série de règles métiers propres aux données. Le code généré automatiquement par l'outil comprendra, en plus du code SQL, du code de programmation que nous, développeurs, n'aurons plus à écrire nous-mêmes. Ce code assurera alors automatiquement le respect des règles métier modélisées, sans interventions supplémentaires de sa part. Dans cet article, nous allons vous présenter comment un outil de ce genre nous permet de gagner en productivité, tout en assurant un meilleur contrôle des données qui constitueront la base de données. Nous illustrerons ce concept en utilisant un automate développé au sein de la Haute école de gestion Arc dans le cadre d'un projet dénommé « MVC-CD ». Nous vous présenterons des exemples de codes générés pour une base de données Oracle.



Modèle conceptuel (MCD), logique (MLD) et physique (MPD) de données

Rappel sur les modèles de données

Avant d'appréhender le concept des modèles de données SMART et de l'automate, il est nécessaire de comprendre les 3 modèles que l'on retrouve dans tout bon logiciel de modélisation des données et de génération de code. La Figure 1 montre l'enchaînement de 3 modèles de données, chacun d'eux ayant son utilité et se trouvant à un niveau d'abstraction différent.

Modèle conceptuel de données

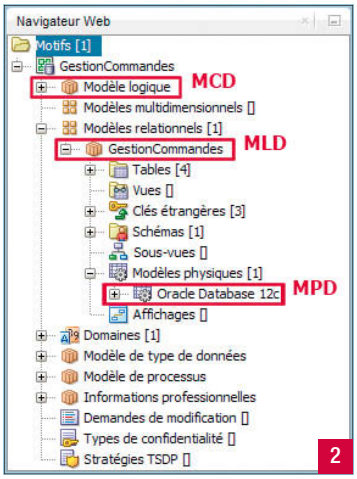
Que faisons-nous en tant que modélisateur d'une base de données pour générer le code SQL automatiquement avec un outil ? Chronologiquement, nous commençons par concevoir le modèle conceptuel de données (MCD). Ce modèle a un haut niveau d'abstraction ; entendez par là qu'il représente la structure des données sans se soucier d'une technologie ou d'un fournisseur de base de données particulier. À ce niveau, nous retrouvons notamment les entités, leurs attributs et les associations entre entités.

Modèle logique de données

Quand le MCD est réalisé, nous utilisons notre outil pour créer automatiquement le modèle logique de données (MLD). Ce dernier est propre à une technologie, mais n'est pas dépendant d'un fournisseur. Dans la plupart des cas, la technologie visée est la base de données relationnelle. Dans ce cas nous parlons de modèle logique de données relationnel (MLD-R). Le MLD-R, généré automatiquement à partir du MCD, comprend les tables et leurs colonnes, les différentes contraintes de clé primaire et étrangère, les contraintes d'unicité, les contraintes « NOT NULL »... Potentiellement, il comprend tout élément standardisé dans la norme ANSI SQL.

Modèle physique de données

En troisième temps, nous choisissons une base de données cible (par exemple Oracle) et générons le modèle physique de données (MPD) à l'aide de notre outil. Dans ce modèle, les éléments du MLD sont repris et adaptés de façon à ce qu'ils respectent le fournisseur choisi. Par exemple, les types SQL que sont « Varchar » et « Numeric » sur le MLD deviennent « Varchar2 » et « Number » dans un MPD pour Oracle.
Le MPD comprend ainsi l'ensemble des éléments pouvant être déployés vers la base de données. Nous utilisons alors l'outil pour générer et exécuter le code. Ce principe de transformation de modèles est connu scientifiquement sous le nom de « Model Driven Engineering » (MDE). Trois avantages ressortent de cette démarche : les modèles réalisés ser-



Les 3 modèles dans SQL Developer Data Modeler

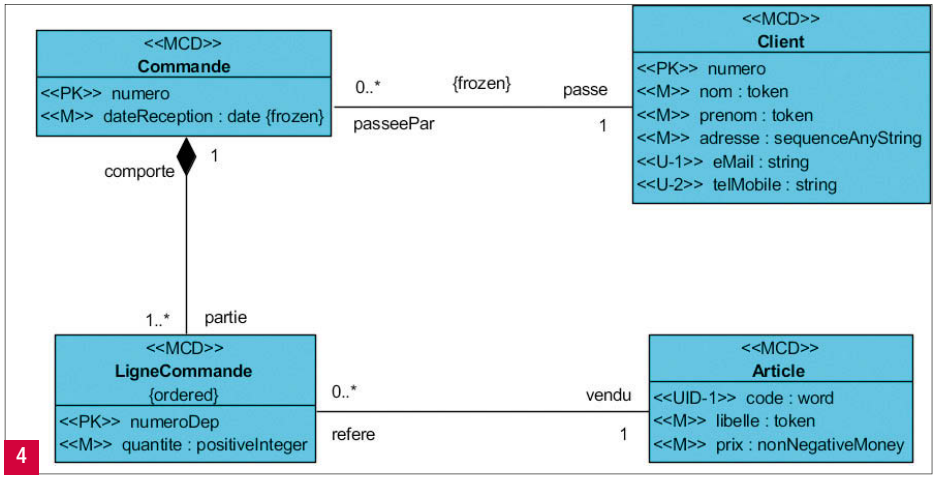


Illustration d'un SMART Model

vent de documentation, nous gagnons en productivité et, enfin, la qualité du code généré est constante.

Exemple dans Oracle Data Modeler

Le nommage de ces 3 modèles peut différer d'un outil à l'autre. Par exemple, dans Oracle SQL Data Modeler, le MCD se nomme « Modèle logique », le MLD « Modèle relationnel » et le MPD « Modèle physique ». La Figure 2 montre les 3 modèles dans le référentiel d'un projet SQL Data Modeler.

Automatisation de règles métier ?

La génération du code SQL marche généralement très bien et nous évite de devoir l'écrire à la main. Seulement, il n'y a pas que le code SQL qui peut être généré automatiquement. Un ensemble de règles métiers, si elles sont spécifiées sur un modèle de données, peuvent également être transformées automatiquement en code. Il ne s'agira plus uniquement de code SQL, mais de code de programmation de base de données.

Éléments générés pour Oracle

Pour une base de données cible Oracle, les règles métier peuvent être transformées en code PL-SQL sous forme de Triggers et de Procédures au sein de paquetages. Une fois déployés, ces Triggers et Procédures vont alors assurer le respect des règles modélisées en empêchant toute modification des données qui ne les respecteraient pas.
La Figure 3 illustre quelques exemples de règles métier qui peuvent être automatisées à l'aide de la génération de code PL-SQL pour Oracle.

Règles métier	Éléments générés pour Oracle
Le code d'un article doit être un mot sans espace	Un Trigger intercepte les ajouts et modifications d'articles, et lève une erreur si le code de l'article soumis comporte un espace, une tabulation ou un retour à la ligne.
La commande d'un client ne peut pas être affectée à un autre client	Un Trigger intercepte les modifications de commandes et lève une erreur si le numéro du client associé est modifié.
Les lignes d'une commande doivent être ordonnées	Un Trigger intercepte l'ajout d'une ligne de commande et affecte un numéro d'ordonnancement en faisant appel à une séquence. Des procédures permettent de modifier l'ordre d'une ligne parmi les autres lignes, en assurant l'unicité entre les numéros d'ordre.

Figure 3 : Exemples de règles métiers automatisables

Quels outils ?

L'idée doit certainement vous sembler bonne, mais, comment pouvons-nous, en tant que développeur, réellement automatiser ce genre de règles métier et ainsi gagner en productivité ? Existe-t-il des outils capables de faire cela ?
À ce jour, nous ne connaissons pas de logiciels de modélisation capables de cela. Cependant, si des règles métier reviennent régulièrement chez vous, alors profitez-en pour écrire votre propre automate ! Les outils et les environnements permettent aujourd'hui de faire cela sans grandes difficultés.
À la Haute école de gestion Arc, nous avons développé un automate qui prend en charge un ensemble de types de règles ; il génère le nécessaire pour une base de données Oracle. Le chapitre « L'automate de transformation MVC-CD » vous apportera plus de détails.

Modèle conceptuel de données SMART

Pour générer automatiquement du code à partir de règles, nous devons commencer par décrire ces règles sur notre modèle de données à l'aide de notre outil de modélisation.
Les règles métier ne dépendent d'aucune technologie. C'est donc sur le modèle conceptuel de données (MCD) que nous devons les décrire. Nous appelons SMART Model ou, en français, Modèle intelligent, un MCD enrichi de ces règles.

Exemple de SMART Model

Nous avons fait le choix d'utiliser le diagramme de classes UML pour dessiner le SMART Model. En effet, UML offre suffisamment de richesse pour nous permettre d'apporter la sémantique nécessaire à la description des règles. Pour illustrer celles que nous rendons possibles, nous nous baserons sur le modèle de gestion de commandes clients visible en Figure 4.

Éléments UML visibles sur l'exemple

Nous avons regroupé les éléments UML visibles sur le diagramme et leur avons donné une brève description. Cette synthèse est présentée par le tableau de la Figure 5.

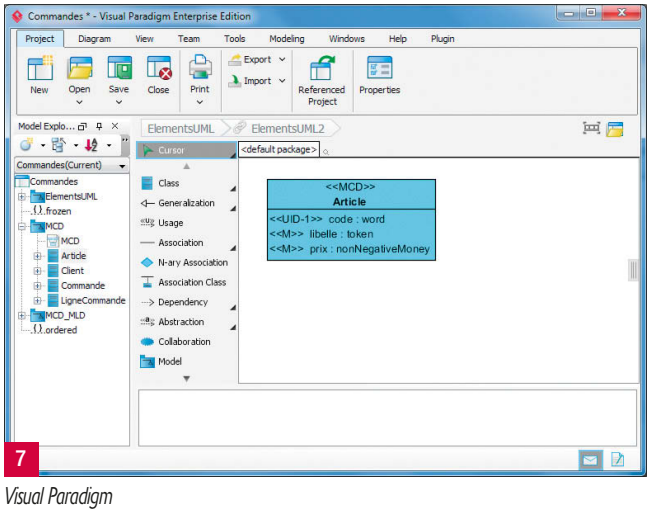
Éléments UML	Signification
0..* / 1	Association qui relie deux entités. L'objet de gauche réfère obligatoirement 1 objet de droite. L'objet de droite réfère aucun, un ou plusieurs objets de gauche.
	Association identifiante qui relie deux entités. L'entité à gauche est l'entité parente. L'entité à droite est l'entité enfant qui s'identifie avec son parent.
<<MCD>>	Stéréotype qui indique que la classe doit être considérée comme une entité du MCD.
<<PK>>	Stéréotype qui spécifie que l'attribut est un identifiant technique (Primary Key).
<<UID-x>>	Stéréotype qui spécifie que l'attribut est un identifiant naturel (Unique Identifier).
<<M>>	Stéréotype qui spécifie que l'attribut est obligatoire (Mandatory).
<<U-x>> {frozen}	Stéréotype qui spécifie que l'attribut est unique. Contrainte qui spécifie que l'élément, une fois affecté, ne peut plus changer.
{ordered} sequenceAnyString, string, token, word, date, positiveInteger, nonNegativeMoney	Contrainte qui spécifie que l'élément doit être ordonné. Types de données dont les significations sont reprises du W3C (voir les références de l'article pour plus d'informations). Par exemple, les données relatives à un attribut de type « word » devront contenir uniquement des mots, sans espace ni retour à la ligne.

Figure 5 : Les éléments UML utilisés et leur signification

Règles métier de l'exemple

Sur l'exemple de la Figure 4, nous retrouvons un certain nombre de règles métier que nous ne pouvons pas spécifier dans les outils de modélisation des données qui génèrent du code.

Nous avons tout d'abord celles qui définissent les différents types de données. Par exemple, le code de l'article est de type « word », ce qui signifie qu'il doit être un mot. À l'inverse, l'adresse du client, de type « sequenceAnyString », autorise les retours à la ligne et autres caractères de formatage. La date de réception de la commande est obligatoire et comporte une contrainte {frozen}. Cela signifie qu'elle ne doit pas pouvoir être changée. Cette contrainte se retrouve également entre les entités « Client » et « Commande ». Une commande, une fois créée pour un client, ne peut plus être affectée à un autre client. Soulignons encore la contrainte {ordered} sur l'entité « LigneCommande » qui spécifie que les lignes de commandes doivent être ordonnées dans la commande.



Visual Paradigm

L'automate de transformation MVC-CD

Ce qui rend le modèle de données conceptuel SMART, c'est l'automate de transformation qui est capable de le lire et de générer le code capable d'assurer le respect des règles qu'il comporte. Nous allons par la suite vous présenter le code que l'automate « MVC-CD » est capable de générer. Mais avant cela, une petite présentation de cet automate s'impose.

Principe de fonctionnement

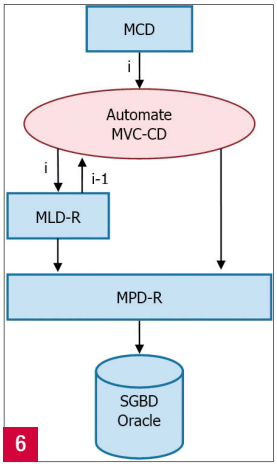
Cet automate a été développé dans le cadre d'un projet intitulé « MVC-CD ». Le nom du projet a notamment été repris pour l'automate. Le principe des transformations de modèles MCD, MLD et MPD a été conservé, à la seule différence que l'automate génère le modèle logique et physique en même temps à partir du MCD (voir Figure 6). L'automate prend en charge le développement itératif et incrémental. En effet, il a été architecturé de façon à générer, au temps « i », uniquement les différences entre le nouveau modèle logique à obtenir et celui qui existe déjà au temps « i-1 ».

Visual Paradigm comme outil de modélisation

Le développement de l'automate consistait surtout à écrire les règles de transformation. Nous n'avons ni développé la gestion d'un référentiel ni la gestion de l'interface graphique. À la place, nous nous sommes appuyés sur un outil de modélisation existant, Visual Paradigm, auquel nous y avons adjoint un plugin. Grâce à l'API de cet outil, nous avons pu profiter entièrement de son interface graphique et de ses fonctionnalités. Cette stratégie nous a permis de nous concentrer sur l'essentiel et d'être ainsi plus rapides dans le développement de l'automate. En Figure 7 vous trouverez une capture-écran de l'interface de dessin de Visual Paradigm.

Contrôle de conformité

Le plugin dispose d'une fonctionnalité de contrôle de conformité du MCD. Il assure la cohérence du modèle vis-à-vis des différents éléments de modélisation utilisés. Par exemple, si nous plaçons un stéréotype <<PK>> en plus de <<UID-1>> sur l'attribut « code » d'une entité « Article », le message visible en Figure 8 s'affichera dans la console de messages.



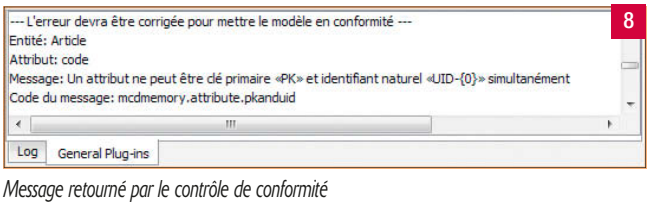
Fonctionnement de l'automate MVC-CD

Téléchargement du plugin

Le plugin pour Visual Paradigm que nous avons développé est téléchargeable librement. Vous trouverez toutes les informations nécessaires dans les références de cet article.

Transformation des règles métier

Nous allons voir quels résultats nous obtenons lorsque nous exécutons l'automate « MVC-CD » sur le MCD de la Figure 4, en sélectionnant une base de données Oracle comme cible. Nous



Message retourné par le contrôle de conformité

partons du postulat que le contrôle de conformité a déjà été exécuté sur le MCD et que ce dernier est conforme.

Génération des modèles logiques et physiques

Lorsque l'automate est exécuté, le référentiel se voit complété par les modèles logiques et physiques. La Figure 9 montre la table « Articles » obtenue, associée à 3 éléments physiques, respectivement de gauche à droite : une séquence, un ensemble de Triggers et enfin un paquetage. Derrière ces éléments se trouve le code PL-SQL nécessaire à assurer le respect des différentes règles posées sur notre SMART Model !

Triggers générés

Si nous visualisons, par exemple, le code contenu dans le Trigger « Art_TAPIs_BIR », nous remarquons que l'automate a généré ceci :

```
CREATE OR REPLACE TRIGGER Art_TAPIs_BIR
BEFORE INSERT ON Articles FOR EACH ROW
DECLARE
 vl_newrec Articles%ROWTYPE;
BEGIN
 vl_newrec.num := :NEW.num;
 vl_newrec.code := :NEW.code;
 vl_newrec.libelle := :NEW.libelle;
 vl_newrec.prix := :NEW.prix;
 Art_TAPIs.autogen_column_ins(vl_newrec);
 Art_TAPIs.autogen_column(vl_newrec);
 Art_TAPIs.checktype_column(vl_newrec);
 Art_TAPIs.uppercase_column(vl_newrec);
 Art_TAPIs.column_PEA(vl_newrec);
 :NEW.num := vl_newrec.num;
 :NEW.code := vl_newrec.code;
 :NEW.libelle := vl_newrec.libelle;
 :NEW.prix := vl_newrec.prix;
END;
```

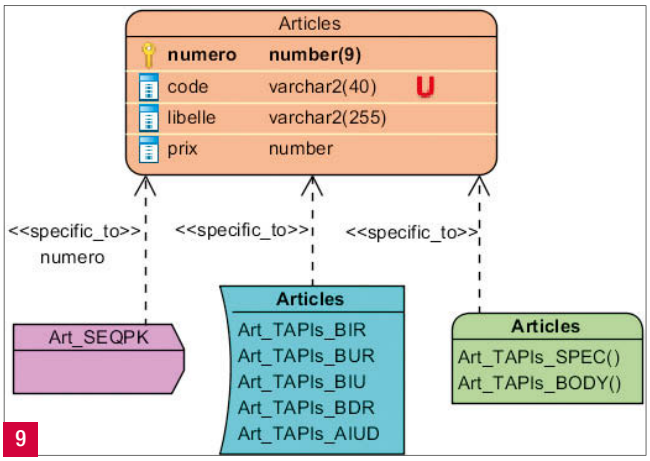
Le Trigger fait en sorte que, lors de l'insertion d'un article, un enregistrement représentant les données en insertion soit créé. Cet enregistrement porte le nom générique « vl_newrec ». Le Trigger appelle ensuite plusieurs procédures avec cet enregistrement. Chacune d'elle va effectuer un traitement bien spécifique. Par exemple, la procédure « checktype_column() » vérifiera le respect des types de données. Ce mécanisme d'interception des requêtes SQL de manipulation de données puis d'appel de procédures est connu sous le nom d'« APIs de tables ».

Procédures générées

Les procédures appelées par le Trigger, stockées dans le paquetage « Art_TAPIs_BODY » visible en Figure 9, assurent le respect des différentes règles spécifiées sur le MCD. Les procédures générées pour la table « Articles » se chargent de :

- Alimenter la colonne « numero » à l'aide de la séquence ;
- Lever une erreur si le code de l'article ne respecte pas le type « word » spécifié sur le MCD ;
- Lever une erreur si le libellé ne respecte pas le type « token » ;
- Lever une erreur si le prix ne respecte pas le type « nonNegativeMoney ».

Le code présent dans la procédure « checktype_column() » vérifie, en effet, que la donnée insérée dans le code de l'article, par exemple, soit de type « word ». Une erreur est levée si une tabulation, un retour à la ligne ou un espace est trouvé :



MLD et MPD obtenu après transformation

```
CREATE OR REPLACE PACKAGE BODY Art_TAPIs IS
 [...]
 PROCEDURE checktype_column(pio_crtrec IN OUT Articles%ROWTYPE) IS
 BEGIN
   IF(INSTR(pio_crtrec.code, CHR(9)) > 0)
     OR(INSTR(pio_crtrec.code, CHR(10)) > 0)
     OR(INSTR( pio_crtrec.code, CHR(13)) > 0)
   THEN
    raise_application_error(-20001, 'Table Articles, Colonne code: Les caractères
de contrôle ne sont pas autorisés pour une donnée de type string');
   END IF;
   IF INSTR(pio_crtrec.code, ' ') > 0 THEN
    raise_application_error(-20001, 'Table Articles, Colonne code: Les espaces
ne sont pas autorisés pour une données de type word');
   END IF;
   [...]
 END;
 [...]
END;
```

En nous aidant de l'automate, nous avons ainsi obtenu ce code de contrôle sans écrire une ligne à la main ! Ce code étant généré, il est assurément compilable. Il ne nous restera plus qu'à déployer le code SQL ainsi que les Séquences, Triggers et Paquetages vers la base de données cible, ce que Visual Paradigm est capable de faire pour nous sans difficulté.

CONCLUSION

Au travers de cet article, vous avez pu voir comment il est possible d'automatiser bien plus que la simple génération de code SQL à partir d'un modèle de données. En utilisant un automate tel que « MVC-CD », vous pouvez générer le code de programmation de bases de données qui va vous assurer que toutes les règles métiers spécifiées sur votre modèle conceptuel soient respectées lors de manipulations de données. Ce MCD riche, nous l'appelons SMART Model. Au vu des gains de productivité et de l'assurance qualité que cela apporte sur vos données, nous ne pouvons que vous encourager à enrichir votre outil de modélisation d'un petit automate qui se chargera d'automatiser les règles qui reviennent de manière récurrente dans votre contexte.

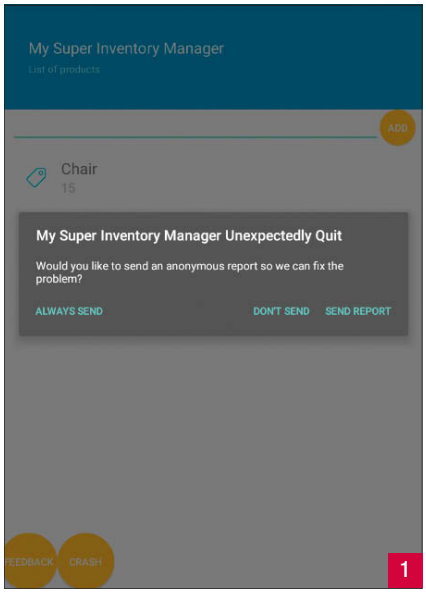
Références

Projet MVC-CD et téléchargement de l'automate de transformation : http://lgl.isnetne.ch/Sagex35793/index.htm
Types de données des Schémas XML du W3C : https://www.w3.org/TR/xmlschema-2

Comment assurer une **belle vie** à votre application mobile ! (Partie 4)

Mathilde Roussel
R&D Software Engineer chez
MEGA International
MVP Visual Studio and Development Technologies
Son Blog : http://www.mathroussel.fr

Jason De Oliveira
CTO | MVP chez
MEGA International
Son Blog :
http://www.jasondeoliveira.com



Dans le domaine du mobile, différents outils gravitent autour d'une application dans le but de faciliter sa gestion pendant sa durée de vie, que l'on espère longue. Nous avons pu voir les outils d'intégration continue ainsi que ceux de tests dans les trois précédents articles (N°198, 199, 200). Mais si l'intégration continue, les tests unitaires et d'interface impactent en premier lieu les développeurs, comment agir à l'étape suivante, quand les utilisateurs vont devoir être confrontés à l'application ?

Bien sûr, avant de penser à soumettre une application mobile sur les différents stores, il est clairement préférable de faire appel à une phase de bêta. Finis les tests automatiques réalisés par une machine, les utilisateurs humains entrent en jeu. Cette partie souvent redoutée par les développeurs l'est pour de bonnes raisons, car il est fréquent que les bêta-testeurs trouvent des comportements inhabituels, pratiquent les monkey-test pour tenter de faire crasher l'application… Et généralement lors de la première livraison en bêta, les premiers bugs se font vite sentir. Pour faciliter la vie à la fois des développeurs et des bêta-testeurs, qui s'investissent tous à la réalisation d'une application de qualité, il existe des outils, tels qu'HockeyApp.

HockeyApp au service du mobile

Comme l'ensemble de ses camarades présentés jusqu'à présent, HockeyApp n'a pas échappé au puissant Microsoft qui ne cesse de s'entourer d'alliés du domaine mobile. La plateforme de HockeyApp s'arme en effet de fonctionnalités qui intéressent fortement les acteurs d'un projet mobile. Cet outil vise à remplacer l'utilisation de Microsoft App Insights dans le cadre du mobile, d'autant plus que HockeyApp travaille désormais main dans la main avec Xamarin Insights pour fournir un service le plus complet possible. Ce dernier ne sera donc pas traité dans cet article, même si son utilisation est toujours possible et offre quelques fonctionnalités bientôt implémentées par HockeyApp.

Mettre HockeyApp en selle côté mobile

Afin de préparer un projet mobile à utiliser HockeyApp, il est nécessaire de suivre plusieurs étapes. Avant toute chose, l'application doit être créée sur la plateforme Web de HockeyApp, afin de récupérer l'appId de celle-ci, qui est à garder précieusement. Ensuite, différents blocs de code devront être ajoutés en fonction des besoins du côté des applications. Il est recommandé pour une lisibilité optimale de les placer dans la méthode qui est appelée au lancement de l'application. Sous iOS ce sera donc dans l'AppDelegate.cs, et Android dans la méthode OnCreate du fichier Application.cs, créé au préalable (celui-ci n'est pas obligatoire, mais permet un bon découpage de la logique).
Si l'application doit pouvoir détecter des crashs et envoyer les données à HockeyApp, la ligne à ajouter est la suivante :

```
CrashManager.Register(this, "idDeMonApplicationHockeyApp");
```

Le premier paramètre représente le contexte de l'activité courante, le deuxième est l'id de l'application HockeyApp récupéré au préalable depuis l'interface Web.
Ensuite, un simple lancement d'exception permet de tester facilement si l'enregistrement des crashs fonctionne. Evidemment, en cas de crash l'application sera quittée. C'est lors de la réouverture de celle-ci qu'une popup va proposer d'envoyer ou non un rapport de crash vers HockeyApp, qui sera visible depuis le portail Web de HockeyApp (Fig. 1). Un crash remonté recense le type du périphérique, les informations telles que la version de l'OS, ainsi que le détail et la trace de l'exception.
HockeyApp propose également un module de feedback. Celui-ci fournit entre autres une page préfabriquée sous forme de formulaire, qui sera intégrée à l'application. Tout comme la gestion des crashs, la configuration du module de feedback est très simple et prend seulement une ligne :

```
FeedbackManager.Register(this, "idDeMonApplicationHockeyApp");
```

Il faut ensuite intégrer un bouton ou autre contrôle pour lancer la page de feedback. Dans le cas de MySuperInventoryManager, un bouton a été placé sur la page d'accueil (MainActivity). Suite au tap sur celui-ci, le reste est délégué au SDK de HockeyApp grâce à l'appel au FeedbackManager :

```
var feedbackButton = FindViewById<Button>(Resource.Id.feedbackButton);

feedbackButton.Click += delegate
```

```
{
FeedbackManager.ShowFeedbackActivity(ApplicationContext);
};
```

En plus de la gestion des crashs et des feedbacks, HockeyApp met à disposition un gestionnaire de métriques. Encore une fois, la configuration de celui-ci sur Android est semblable à celles présentées précédemment :

```
MetricsManager.Register(this, this, "idDeMonApplicationHockeyApp");
```

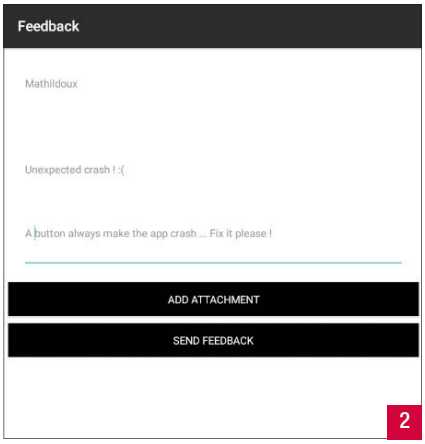
Le premier paramètre est toujours le contexte de l'activité (dans le cadre Android), le deuxième est la référence de l'application. Comme ses deux compagnons, cette ligne est située dans la classe Application, qui emporte avec elle le contexte global de l'application, ce qui explique le deuxième « this ».

Après vérification que la page de feedback fonctionne correctement (Fig. 2), et que les crashs proposent effectivement un envoi de données à HockeyApp, l'application est prête pour les bêta-tests.

Analyse des données de bataille sur HockeyApp

A partir du moment où l'application est configurée et envoyée aux bêta-testeurs, des informations vont remonter au fur et à mesure dans HockeyApp. Afin d'étudier l'état de l'application et ce qu'il s'y passe, il faut passer du côté de la plateforme Web de HockeyApp. Après sélection de l'application souhaitée, le dashboard principal apparaît. L'interface fournit différentes informations, comme la liste des dernières versions du projet envoyées sur HockeyApp, avec pour chacune quelques détails succincts. Des métriques supplémentaires sont disponibles sous forme de graphiques, comme les crashs, téléchargements, utilisateurs et sessions sur les sept derniers jours. Des graphiques plus étendus représentant ces informations sur les trente derniers jours sont disponibles en choisissant l'option de métrique voulue.

Un menu sur le portail offre l'accès à un ensemble de données détaillées, via les onglets « Overview », « Versions », « Crashes », « Events », « Feedback » et « Users ». Le menu « Overview » n'est autre que le dashboard principal présenté ci-dessus. Dans l'onglet « Versions » se retrouvent tous les packages disponibles via HockeyApp. Chaque version regroupe différentes informations telles que le statut, le type de périphérique (Android, iOS ou Windows Phone), ou encore l'OS minimum.



Dans le détail d'une version, il existe également un sous menu qui détaille les fichiers, les crashs et feedbacks associés à cette version, ainsi que ses statistiques. Grâce à l'intégration de HockeyApp au sein des outils Microsoft, il est possible de profiter d'informations de Visual Studio Team Services depuis la plateforme Web d'HockeyApp. Pour cela, une petite phase de configuration s'impose, dont voici les étapes :

- Accéder aux paramètres de l'application depuis le bouton « Manage App » sur le dashboard principal.
- Dans la liste des sous menus, se trouve un onglet Visual Studio Team Services. Il faut ensuite configurer l'url du VSTS voulu, le repository et le projet (MySuperInventory Manager dans ce cas).
- Pour pousser plus loin l'interaction, il est possible de gérer la création de tickets lors de crashs et/ou de feedbacks, à partir de combien de crashs un ticket est créé, etc.

A la suite de ces étapes, la connexion VSTS vers HockeyApp est effective. De retour sur la page de détail d'une version se trouvent deux onglets qui deviennent intéressants : « Commits » et « Work Items » (Fig. 3). L'onglet Commits affiche les commits qui correspondent à cette version, avec leur message et leur auteur. L'onglet Work Items liste les tâches VSTS associées aux commits. Attention, ces deux parties sont encore en bêta, il est donc possible de rencontrer des bugs.

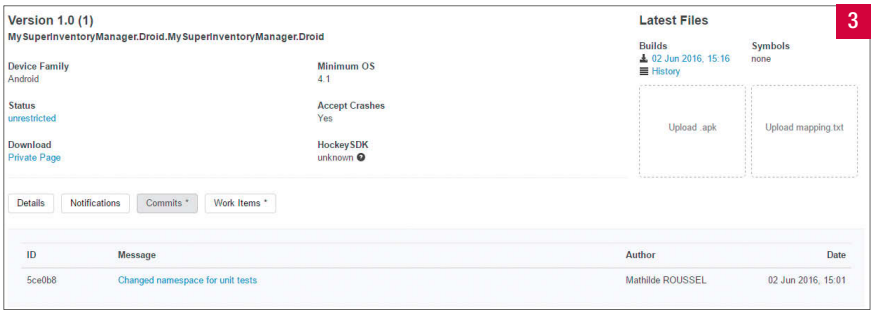
Les crashs

A la suite du menu « Versions » se trouve la page « Crashes », qui affiche les crashs survenus sur l'application. Ils sont regroupés selon le type d'exception, précisent la ou les versions concernées par le crash, ainsi que le nombre d'occurrences. Cela permet de visualiser rapidement lorsqu'un crash revient sur plusieurs versions, ou de voir s'il est fréquent ou non.

Les crashs sont regroupés sous trois statuts différents : le statut ouvert (affiché en rouge), suivi (en orange), résolu (en vert), ou ignoré (en noir). Les deux plus simples, les résolus ou ignorés, signifient respectivement que le bug lié au crash a été résolu, ou que, pour une raison quelconque (faux positif par exemple), le crash généré n'est pas significatif et donc ignoré. Un bug ouvert est un bug qui n'a pas encore été traité ou suivi, sachant qu'un bug suivi (« tracked ») est considéré comme tel par modification manuelle du statut ou après ajout d'un ticket dans Visual Studio Team Services. La création d'un ticket ou l'assignation à un ticket existant se fait depuis le sous menu « Bug Tracker » dans le détail du crash, ou de façon automatique, selon la configuration choisie.

Justement, après avoir fait le tour de la page listant les crashs, il est temps d'étudier le détail de l'un d'eux, afin de mieux comprendre comment celui-ci s'est produit. Et cela tombe bien car HockeyApp récupère un grand nombre d'informations. Après avoir choisi et cliqué sur un crash de la liste, l'utilisateur est redirigé vers la page de détails où un résumé l'attend. Parmi les données captées se retrouvent la date de la première occurrence ainsi que de la dernière, un graphique des jours où le crash s'est produit, ainsi que trois onglets qui affichent la trace de l'exception, le ou les téléphones concernés et la ou les versions des OS. Si la trace de l'exception ne suffit pas, un menu permet d'accéder à la page « Crash logs », qui affiche un peu plus de détails. Le dernier menu, le « Bug Tracker » permet comme expliqué précédemment d'associer ce crash à un ticket dans VSTS afin de faciliter le suivi. •

Suite le mois prochain



XAMARIN : mon premier jeu Cross Platform avec CocosSharp

2e partie

François Botte
Microsoft Technical Lead,
SQLI Toulouse

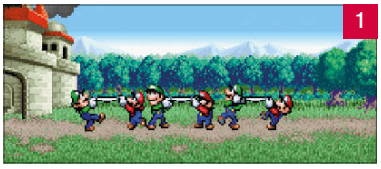
CocosSharp est un Framework de développement de jeux 2D multiplateformes (Android, iOS, et Windows). Il s'agit du portage XAMARIN du projet open source Cocos2D intégrant un moteur physique et une gestion poussée des animations, des sprites, des transitions, … Et quoi de mieux qu'un petit jeu rétro pour découvrir un peu plus cet univers ?

Etape 6 : Création des acteurs

Dans cette partie, nous allons créer les personnages qui tiennent le trampoline : Mario et Luigi. Dans ce jeu, ils se trouvent sur le chemin et tiennent une sorte de planche qui permettra à Yoshi de rebondir jusqu'au carrosse et ainsi être sauvé. C'est trop mignon n'est-ce pas ? C'est l'utilisateur qui contrôlera les deux personnages en les faisant naviguer soit vers la gauche soit vers la droite. Trois positions sont possibles pour afficher les deux personnages : [Fig.1]. Dans le fichier « MainScene.cs », rajoutez trois nouvelles méthodes : [Fig.2].

Notez que nous cachons les personnages au centre et à droite pour ne laisser que ceux de gauche par défaut au démarrage du jeu. N'oubliez pas d'appeler ces trois méthodes dans le constructeur de votre classe et de rajouter les ressources dans les deux projets : [Fig.3]. Exécutez le projet (Android ou iOS) : [Fig.4]. Rajoutez le carrosse situé complètement à droite. C'est la porte de sortie pour notre personnage Yoshi : [Fig.5].

Note : *N'oubliez pas de rajouter les ressources goal.plist et goal.png aux projet Android et iOS.*



1

```
public MainScene(CCGameView gameView) : base(gameView)
{
    layer = new CCLayer();
    this.AddChild(layer);

    Création de notre décor anime

    CreateMarioLuigiLeft();
    CreateMarioLuigiCenter();
    CreateMarioLuigiRight();
}
```

3

```
private CCSprite _spriteMLLeft;
private void CreateMarioLuigiLeft()
{
    _spriteMLLeft = CreateAnimationFromSpriteSheet("paddlegauche.plist", "paddlegauche", 0.4f);
    _spriteMLLeft.AnchorPoint = new CCPoint(0, 0);
    _spriteMLLeft.PositionX = 43; _spriteMLLeft.PositionY = 35;
    _spriteMLLeft.IsAntialiased = false;
    layer.AddChild(_spriteMLLeft);
}

private CCSprite _spriteMLCenter;
private void CreateMarioLuigiCenter()
{
    _spriteMLCenter = CreateAnimationFromSpriteSheet("paddlecentre.plist", "paddlecentre", 0.4f);
    _spriteMLCenter.AnchorPoint = new CCPoint(0, 0);
    _spriteMLCenter.PositionX = 90; _spriteMLCenter.PositionY = 35;
    _spriteMLCenter.IsAntialiased = false;
    layer.AddChild(_spriteMLCenter);
    _spriteMLCenter.RunAction(new CCHide());
}

private CCSprite _spriteMLRight;
private void CreateMarioLuigiRight()
{
    _spriteMLRight = CreateAnimationFromSpriteSheet("paddledroite.plist", "paddledroite", 0.4f);
    _spriteMLRight.AnchorPoint = new CCPoint(0, 0);
    _spriteMLRight.PositionX = 137; _spriteMLRight.PositionY = 35;
    _spriteMLRight.IsAntialiased = false;
    layer.AddChild(_spriteMLRight);
    _spriteMLRight.RunAction(new CCHide());
}
```

2

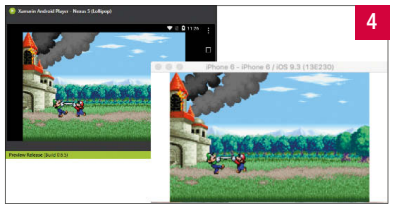
```
private CCSprite _spriteGoal;
private void CreateGoal()
{
    _spriteGoal = CreateAnimationFromSpriteSheet("goal.plist", "goal", 0.4f, false);
    _spriteGoal.AnchorPoint = new CCPoint(0, 0);
    _spriteGoal.PositionX = 190; _spriteGoal.PositionY = 41;
    _spriteGoal.IsAntialiased = false;
    layer.AddChild(_spriteGoal);
}
```

5

Etape 7 : Animation des acteurs

Pour animer les deux personnages, ajoutez une nouvelle méthode « CreateTouchListener » : [Fig.6].

Le listener va « attraper » tous les « taps » sur l'écran (« CCEventListenerTouchAllAtOnce ») et exécuter la méthode « Handle TouchesBegan ». Celle-ci analyse si un « tap » ou un « click » utilisateur a été déclenché et regarde où l'utilisateur a appuyé. Nous avions, lors de la création de la classe « GameView » défini la taille de l'écran ainsi :[Fig.7]. Donc si la coordonnée X de l'appui de l'utilisateur est <120, alors c'est un clic gauche, sinon clic droit. En fonction de cela, il ne nous reste plus qu'à rendre visibles les bons personnages suivant la position actuelle de celui déjà affiché. N'oubliez pas d'ajouter dans le constructeur de la scène l'appel de la méthode « CreateTouchListener ». Exécutez le projet (Android ou iOS) et testez les clics ou taps à droite et à gauche : [Fig.8].



4

```
private void CreateTouchListener()
{
    var touchListener = new CCEventListenerTouchAllAtOnce();
    touchListener.OnTouchesBegan = HandleTouchesBegan;
    layer.AddEventListener(touchListener);
}

private void HandleTouchesBegan(List<CCTouch> touches, CCEvent touchEvent)
{
    if (touches.Count > 0)
    {
        //Tap a droite ou a gauche de l'écran
        if (touches[0].Location.X > 120)    //Droite
        {
            if (_spriteMLLeft.Visible)
            {
                _spriteMLLeft.RunAction(new CCHide());
                _spriteMLCenter.RunAction(new CCShow());
                _spriteMLRight.RunAction(new CCHide());
            }
            else if (_spriteMLCenter.Visible)
            {
                _spriteMLLeft.RunAction(new CCHide());
                _spriteMLCenter.RunAction(new CCHide());
                _spriteMLRight.RunAction(new CCShow());
            }
        }
        else                                //Gauche
        {
            if (_spriteMLCenter.Visible)
            {
                _spriteMLLeft.RunAction(new CCShow());
                _spriteMLCenter.RunAction(new CCHide());
                _spriteMLRight.RunAction(new CCHide());
            }
            else if (_spriteMLRight.Visible)
            {
                _spriteMLLeft.RunAction(new CCHide());
                _spriteMLCenter.RunAction(new CCShow());
                _spriteMLRight.RunAction(new CCHide());
            }
        }
    }
}
```

6

Etape 8 : Le lancer de Yoshi (un peu de math…)

Rajoutez la méthode « CreateYoshi ». N'oubliez pas de l'appeler dans le constructeur de la scène et avant la méthode « CreateGoal » car Yoshi doit passer derrière le carrosse pour plus de réalisme : [Fig.9].

Attention, on attaque la partie un peu pénible pour ceux qui sont allergiques aux mathématiques. Vous pouvez passer cette étape et copier-coller le code tel quel si vous le voulez. Pour les autres, un peu de théorie. Voici le mouvement que va suivre notre Yoshi. Il va se jeter du château en suivant une parabole et son mouvement va accélérer au fur et à mesure de sa descente jusqu'à ce qu'il rencontre soit le trampoline soit le sol. S'il rencontre le trampoline, alors on inverse la courbe mais il ira moins haut. S'il rencontre le sol, la partie est finie et notre pauvre Yoshi aussi…

```
//Ici nous utiliserons une faible résolution pour donner une apparence
//pixel art
int width = 240;
int height = 161;
GameView.DesignResolution = new CCSizeI(width, height);
```
7

```
//Constante de vitesse de jeu
float totalFrame = 2;

void RunGameLogic(float frameTimeInSeconds)
{
    //Définition de la vitesse de jeu
    totalFrame += (frameTimeInSeconds * 20);

    //Yoshi doit sauter
    if (!_spriteYoshi.Visible) JumpYoshi();

    //OOn trace un rectangle invisible autour de Mario et Luigi
    //Cela nous permettra de gérer la collision avec Yoshi pour le faire rebondir ou non
    CCRect smRect;
    if (_spriteMLCenter.Visible)
        smRect = new CCRect(_spriteMLCenter.PositionX, _spriteMLCenter.PositionY,
                            _spriteMLCenter.ContentSize.Width, _spriteMLCenter.ContentSize.Height - 13);
    else if (_spriteMLRight.Visible)
        smRect = new CCRect(_spriteMLRight.PositionX, _spriteMLRight.PositionY,
                            _spriteMLRight.ContentSize.Width, _spriteMLRight.ContentSize.Height - 13);
    else
        smRect = new CCRect(_spriteMLLeft.PositionX, _spriteMLLeft.PositionY,
                            _spriteMLLeft.ContentSize.Width, _spriteMLLeft.ContentSize.Height - 13);

    //PREMIERE BOUCLE DU SAUT
    if (Math.Floor(totalFrame) > _spriteYoshi.PositionX && _spriteYoshi.PositionX < 59)
    {
        _spriteYoshi.PositionX += 1;
        _spriteYoshi.PositionY = (float)((Math.Abs(Math.Cos((_spriteYoshi.PositionX - 20)
            * ((Math.PI / 2) / (58 - 20)))))
            * (120 - 52) + 52);
    }

    //DEUXIEME BOUCLE DU SAUT
    if (Math.Floor(totalFrame) > _spriteYoshi.PositionX && (_spriteYoshi.PositionX >= 59 && _spriteYoshi.PositionX < 105))
    {
        _spriteYoshi.PositionX += 1;
        _spriteYoshi.PositionY = (float)((Math.Abs(Math.Cos(((_spriteYoshi.PositionX - 59)
            * Math.PI / (105 - 59)) + (Math.PI / 2))))
            * (105 - 52) + 52);
    }

    //TROISIEME BOUCLE DU SAUT
    if (Math.Floor(totalFrame) > _spriteYoshi.PositionX && (_spriteYoshi.PositionX >= 105 && _spriteYoshi.PositionX < 152))
    {
        _spriteYoshi.PositionX += 1;
        _spriteYoshi.PositionY = (float)((Math.Abs(Math.Cos(((_spriteYoshi.PositionX - 105)
            * Math.PI / (152 - 105)) + (Math.PI / 2))))
            * (85 - 52) + 52);
    }

    //DERNIERE BOUCLE POUR ALLER DANS LE CARROSSE
    if (Math.Floor(totalFrame) > _spriteYoshi.PositionX && (_spriteYoshi.PositionX >= 152 && _spriteYoshi.PositionX < 202))
    {
        _spriteYoshi.PositionX += 1;
        _spriteYoshi.PositionY = (float)((Math.Abs(Math.Cos(((_spriteYoshi.PositionX - 152)
            * Math.PI / (202 - 152)) + (Math.PI / 2))))
            * (75 - 52) + 52);
    }

    //YOSHI EST DANS LE CARROSSE :-)
    if (_spriteYoshi.PositionX >= 202)
    {
        _spriteYoshi.RunAction(new CCHide());
    }
    else if (_spriteYoshi.PositionY <= 53)
    {
        //YOSHI EST PROCHE DU SOL, TRAMPOLINE PRESENT ?
        if (!_spriteYoshi.BoundingBoxTransformedToWorld.IntersectsRect(smRect))
            GameOver(_spriteYoshi.PositionX);
    }
}

private void GameOver(float positionX)
{
    UnscheduleAll();
    _spriteYoshi.RunAction(new CCHide());
    CreateYoshiBoom(positionX);
}
```
10

```
private CCSprite _spriteYoshi;
private void CreateYoshi()...

private void JumpYoshi()
{
    totalFrame = 2;
    _spriteYoshi.PositionX = 2; _spriteYoshi.PositionY = 113;
    _spriteYoshi.RunAction(new CCShow());
}

private CCSprite _spriteYoshiBoum;
private void CreateYoshiBoom(float positionX)
{
    _spriteYoshiBoum = CreateAnimationFromSpriteSheet("yoshiboum.plist", "yoshiboum", 0.3f);
    _spriteYoshiBoum.AnchorPoint = new CCPoint(0, 0);
    _spriteYoshiBoum.PositionX = positionX; _spriteYoshiBoum.PositionY = 30;
    _spriteYoshiBoum.IsAntialiased = false;
    layer.AddChild(_spriteYoshiBoum);
}

#endregion
```
12

Cela se traduit donc par trois boucles sur lesquelles nous allons itérer pour calculer les coordonnées de i et y en fonction de chaque point x :

Boucle	i	y
1	(x-X1).((pi/2)/(X2-X1))	(abs(cos(i))*(Y1-Y0)) + Y0
2	((x-X2).(pi)/(X3-X2))+(pi/2)	(abs(cos(i))*(Y2-Y0)) + Y0
3	((x-X3).(pi)/(X4-X3))+(pi/2)	(abs(cos(i))*(Y3-Y0)) + Y0

Bon, le but n'est pas de faire des mathématiques alors mettons en place cela. Nous allons rajouter la méthode CocosSharp « Schedule » qui permet d'effectuer tout ce que l'on a besoin de faire durant un temps exprimé en frame (rafraîchissement de l'écran). En résumé, à chaque rafraîchissement de l'écran, cette méthode est appelée : [Fig.10]

Dans le constructeur de la scène on rajoute la méthode suivante : [Fig.11]. La méthode « RunGameLogic » appelle en début de procédure une autre méthode « JumpYoshi » qui permet de réinitialiser notre héros une fois qu'il est arrivé au carrosse. Par contre, s'il tombe, alors c'est la méthode « CreateYoshiBoom » qui affichera Yoshi à l'endroit de la chute avec une animation particulière : [Fig.12].

Note : *N'oubliez pas de rajouter les ressources yoshiboum.plist et yoshiboum.png aux projet Android et iOS.*

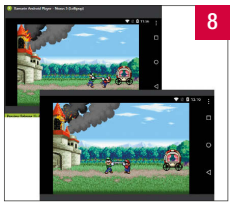
Exécutez le projet (Android ou iOS) : [Fig.13].

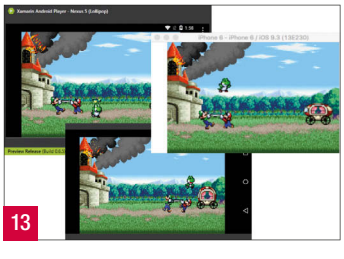
Conclusion

Voilà un rapide aperçu de la bibliothèque CocosSharp pour Xamarin. Evidemment, notre jeu est loin d'être fini et Il faudrait intégrer la notion de score, pouvoir paramétrer la difficulté, lancer plusieurs Yoshi en même temps, etc. Les idées ne manquent pas mais vous avez tous les éléments pour continuer ce petit jeu style « rétro gaming ». CocosSharp remplit véritablement son contrat avec Xamarin. Une seule classe partagée permet le déploiement sur toutes les plateformes mobiles. Avec ses milliers de possibilités et son moteur de physique intégré, c'est la solution idéale pour commencer la programmation de jeu qui n'auront qu'une seule limite : vous ! Annexe : Vous trouverez toutes les ressources et les sources de ce projet sur le site suivant :

http://francois.botte.free.fr/CocosSharp/Xamarin.GameWatch.Fire.zip
http://francois.botte.free.fr/CocosSharp/ressources.zip

```
    #region Création des acteurs
    CreateMarioLuigiLeft();
    CreateMarioLuigiCenter();
    CreateMarioLuigiRight();
    CreateYoshi();
    CreateGoal();
    #endregion

    //Création du listener
    CreateTouchListener();
}

Décor animé

#region Acteurs
private CCSprite _spriteMLLeft;
private void CreateMarioLuigiLeft()...

private CCSprite _spriteMLCenter;
private void CreateMarioLuigiCenter()...

private CCSprite _spriteMLRight;
private void CreateMarioLuigiRight()...

private CCSprite _spriteGoal;
private void CreateGoal()...

private CCSprite _spriteYoshi;
private void CreateYoshi()
{
    _spriteYoshi = CreateAnimationFromSpriteSheet("yoshi.plist", "yoshi", 0.2f);
    _spriteYoshi.AnchorPoint = new CCPoint(0, 0);
    _spriteYoshi.PositionX = 2; _spriteYoshi.PositionY = 113;
    _spriteYoshi.IsAntialiased = false;
    layer.AddChild(_spriteYoshi);
    _spriteYoshi.RunAction(new CCHide());
}
```
9


8


13

```
public MainScene(CCGameView gameView) : base(gameView)
{
    layer = new CCLayer();
    this.AddChild(layer);

    Création de notre décor animé

    Création des acteurs

    //Création du listener
    CreateTouchListener();

    //Logique du jeu
    Schedule(RunGameLogic);
}
```
11

Contrôler votre Constellation depuis votre montre Gear S2

Partie 3

Sébastien Warin
Owner / CEO
@ myConstellation.io
http://www.myconstellation.io
http://sebastien.warin.fr

Heureux détenteur d'une montre Samsung Gear S2, j'admets l'avoir toujours autour du poignet. Très pratique pour donner l'heure, elle permet aussi de recevoir les notifications de son téléphone, prendre les appels, répondre aux SMS, mesurer sa fréquence cardiaque, ses pas, etc. Et forcément en bon développeur, j'ai très rapidement cherché à connecter cette montre à la plateforme Constellation afin de pouvoir facilement piloter alarme, lumières, volets, thermostat, media-center, porte de garage ou même la voiture !

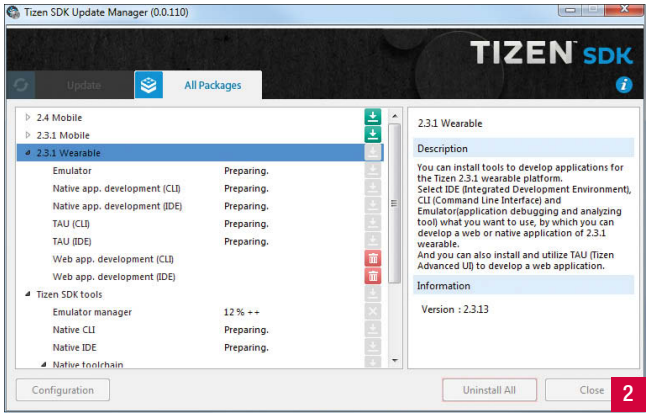
Découverte de Tizen

Concrètement la Samsung Gear S2 tourne sous Tizen. Cet OS a été initié par Samsung sous le nom LiMo (Linux Mobile) avant d'être renommé en 2011 en Tizen lorsque qu'Intel a rejoint le projet en abandonnant MeeGo. Depuis 2012, Tizen est développé par la "Tizen Association" qui regroupe entre autres Samsung et Intel mais aussi Huawei, Fujitsu, NEC, Panasonic, Orange, NTT DoCoMo ou encore Vodafone. Dans leurs premières versions toutes les applications Tizen étaient des applications HTML/JS permettant de garantir une portabilité entre les différents équipements. Depuis la version 2.0 sortie en 2013, il est également possible de développer des applications natives. C'est également à cette date que Samsung a annoncé l'abandon de son précédent système Bada pour se consacrer exclusivement à Tizen. On retrouve Tizen dans plusieurs équipements de la marque : depuis les smartphones (série Samsung Z), tablettes ou PC, jusqu'aux wearables (série Gear), en passant par les TV comme la dernière UN65H8000AF ou les appareils photos (comme le NX1). Cet OS a pour objectif d'équiper une multitude d'équipements dont des produits électroménagers (micro-onde, lave-linge, réfrigérateur), des voitures, des imprimantes, etc.

Hello World Tizen

Pour commencer, rendez-vous sur le site développeur Samsung pour télécharger le SDK Tizen.

Une fois la base installée, il faudra lancer "l' Update Manager" afin de sélectionner les composants dont vous avez besoin. Dans notre cas, nous allons télécharger le Framework "Wearable" 2.3.1 pour développer des applications pour la Gear S2 ainsi que les outils du SDK Tizen (dont l'Emulateur Manager), mais surtout, dans la catégorie "Extras", les extensions "Tizen Wearable" et le Certification Manager.



Tizen SDK Update Manager

Pour développer une application Tizen Wearable, on peut soit faire une application "Web" en utilisant HTML5/CSS/JavaScript, soit une application native C/C++, ou soit une application hybride (Web + native).

Ces applications Wearables peuvent être rangées dans deux catégories :

- "Standalone" : fonctionne sur votre montre de manière autonome ;
- "Companion" : est liée à une application "host" qui tourne sur un mobile (Tizen ou Android).

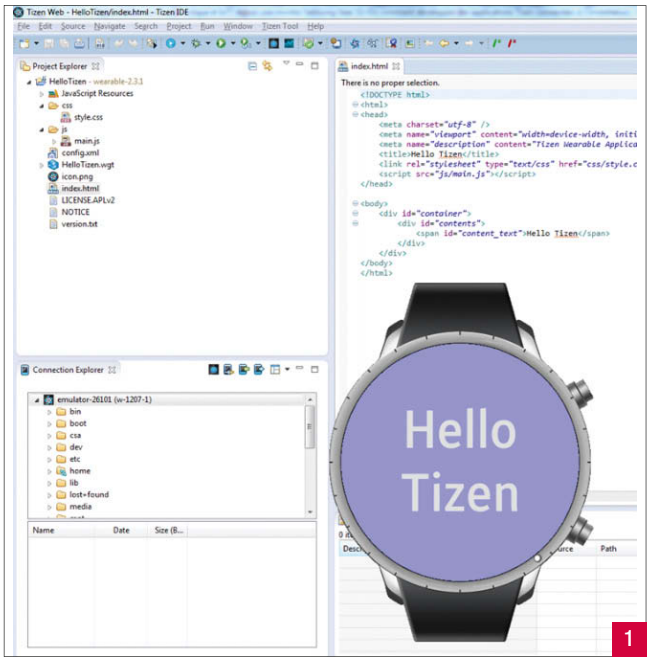
Lançons donc l'IDE Tizen, un IDE basé sur Eclipse. Nous pouvons alors créer un nouveau projet à partir d'un template ou d'exemple prêt à l'emploi.

Ici la page HTML est un simple Hello World avec quelques lignes de JS pour changer le contenu d'une div au clic et gérer le bouton « back ».

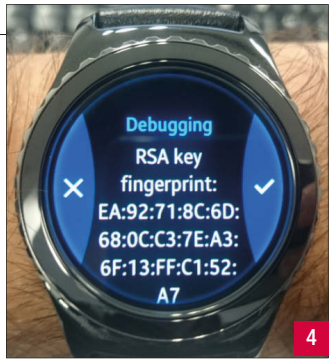
Pour tester votre application, vous disposez du « Emulator Manager » vous permettant de créer des VM pour émuler votre Gear S2 et tester vos applications en local : [Fig.1]

Comme pour le développement d'une application Cordova/Ionic sur Android, vous disposez de l'inspecteur Chrome intégré : [Fig.2]

Bien entendu, vous pouvez également déployer et debugger directement sur votre montre. La première chose à faire est d'activer le débogage dans les paramètres de votre Gear S2 et de la connecter sur votre réseau Wifi. Depuis le "Connection Explorer" de l'IDE Tizen, vous



Emulateur Tizen



Debugging sur un Gear S2

pourrez alors ajouter un nouveau "Remote Device" en spécifiant l'IP de votre Gear : [Fig.3]. Vous devrez alors approuver la connexion sur votre montre : [Fig.4]. Après quoi elle sera connectée dans l'IDE Tizen de la même manière que l'émulateur. Nous pouvons alors relancer l'application en mode "Run" ou "Debug" : Avec toujours la possibilité de faire du "Remote Debug" avec l'inspecteur Chrome directement sur la montre !

Créer une application Tizen connectée à Constellation pour piloter sa maison ou sa voiture

Comme il s'agit d'une application Web/Javascript, nous pouvons simplement utiliser la librairie JavaScript de Constellation. Et pour simplifier d'avantage le code, utilisons AngularJS et son module Constellation. Créons donc un nouveau projet Tizen Wearable. Puis ajoutons les librairies Constellation dans le dossier « js » . Après avoir référencé ces libraires dans la page « index.html », éditons le fichier « app.js » à la racine pour créer un contrôleur AngularJS en y injectant le module Constellation. Nous initialiserons le client Constellation avec l'URI de votre Constellation et une clé d'accès :

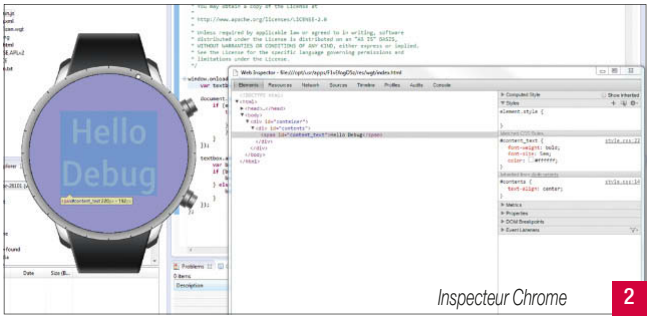
```
var swatch = angular.module('swatch', ['ngConstellation'])
  .controller('MainController', ['$scope', 'constellationConsumer',
    function ($scope, constellation) {

    constellation.intializeClient("https://monServerConstellation.com/constellation", "Mon
AccessKey", "SWatch");


    constellation.connect();
}]);
```

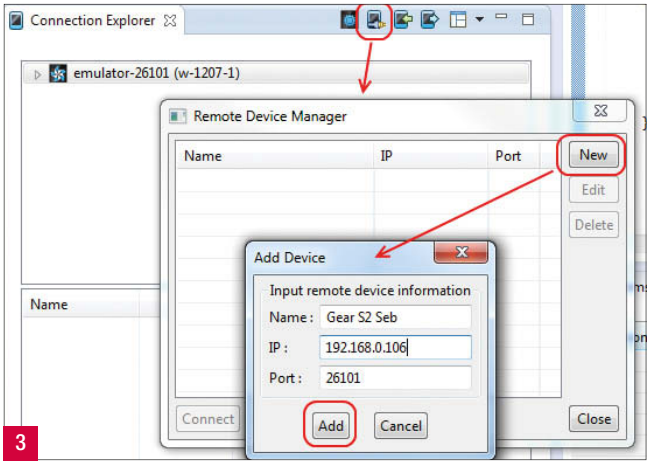
Pour chaque StateObject reçu de la Constellation, stockons-le dans une variable de notre scope Angular ce qui nous permettra de les exploiter dans le template HTML :

```
constellation.onUpdateStateObject(function (message) {
  $scope.$apply(function () {
    if ($scope[message.PackageName] == undefined) {
      $scope[message.PackageName] = {};
    }
    $scope[message.PackageName][message.Name] = message;
    });
});
```

Enfin, lorsque que nous sommes connecté à Constellation, abonnons-nous aux StateObjects des différents packages de notre Constellation que nous souhaitons intégrer dans notre application Tizen.
Pour ma part je demande les StateObjects :

- Du thermostat Nest ;
- Des caméras de sécurité connectées à ZoneMinderde mon système home-made de monitoring des ressources énergétiques S-Energy (basé sur un Raspberry) et S-Opener (gestion de la porte de garage) ;
- Des capteurs NetAtmo et Oregon via un RFXCOM ;
- De la domotique Z-Wave via une Vera ;
- Des media-centers Xbmc/Kodi ;





Ajout d'un remote device

- De mon ampli Pioneer ;
- Des onduleurs ;
- De l'alarme (Paradox) ;
- Des capteurs de luminosité home-made (Raspberry et ESP8266) ;
- Des capteurs hardware via WMI et ceux du routeur DD-WRT ;
- Egalement les StateObjects du package Tesla présentés cet été.

```
constellation.onConnectionStateChanged(function (change) {
  if (change.newState === $.signalR.connectionState.connected) {
    constellation.requestSubscribeStateObjects("*", "tesla", "*", "*");
    constellation.requestSubscribeStateObjects("*", "ZoneMinder", "*", "ZoneMinder.Monitor");
    constellation.requestSubscribeStateObjects("*", "SEnergy", "*", "*");
    constellation.requestSubscribeStateObjects("*", "SOpener", "*", "*");
    constellation.requestSubscribeStateObjects("*", "NetAtmo", "*", "*");
    constellation.requestSubscribeStateObjects("*", "rfxcom", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Vera", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Nest", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Xbmc", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Pioneer", "*", "*");
    constellation.requestSubscribeStateObjects("*", "BatteryChecker", "*", "*");
    constellation.requestSubscribeStateObjects("*", "Paradox", "*", "*");
    constellation.requestSubscribeStateObjects("*", "LightSensor", "Lux", "*");
    constellation.requestSubscribeStateObjects("CEREBRUM", "HWMonitor", "*", "*");
    constellation.requestSubscribeStateObjects("*", "ddwrt", "RouterAJS.WAN", "*");
    constellation.requestSubscribeStateObjects("*", "ddwrt", "RouterAJS.WAN.Stats", "*");
  }
});
```

Pas besoin de plus de code, on peut dès maintenant éditer la page HTML pour créer notre interface. Par exemple pour afficher la température et l'humidité captées par les sondes NetAtmo, il me suffira dans le template HTML d'ajouter le code suivant :

```
<p>Salon : {{NetAtmo['Salon.Temperature'].Value}}°C | {{NetAtmo['Salon.Humidity'].Value}}%</p>
<p>Jardin : {{NetAtmo['Jardin.Temperature'].Value}}°C | {{NetAtmo['Jardin.Humidity'].Value}}%</p>
<p>Chambre1 : {{NetAtmo['Chambre1.Temperature'].Value}}°C | {{NetAtmo['Chambre1.Humidity'].Value}}%</p>
<p>Chambre2 : {{NetAtmo['Chambre2.Temperature'].Value}}°C | {{NetAtmo['Chambre2.Humidity'].Value}}%</p>
```

Tizen propose des classes CSS prêtes à l'emploi que nous pouvons directement appliquer sur nos balises HTML pour rendre le visuel plus agréable : **[Fig.5].** De plus grace à la notion de StateObjects de Constellation, nous pouvons sans trop d'efforts afficher tout type de donnée peu importe la source. Par exemple, ici nous affichons la température de 3 pièces différentes captées par trois technologies différentes (NetAtmo, RFXCOM et Z-WAve)

```
<p>Chambre (Netatmo) : {{NetAtmo['Chambre1.Temperature'].Value}}°C</p>
<p>Bureau (RFXCOM/Oregon) : {{rfxcom['Capteur Bureau'].Value.Temperature}}°C</p>
<p>Cave (Vera/Z-WAve) : {{Vera['Temperature Cave'].Value.Temperature}}°C</p>
```

La librairie Tizen fournit également quelques contrôles Javascript. Par exemple utilisons le « tau.widget.CircleProgressBar » que nous attachons sur un élément « progressbar » lié au StateObject « Nest » :

```
<div class="ui-content">
  <div>Consigne: <span id="nestTarget">{{ Nest['Living Room'].Value.target_temperature_c }}</span>°C</div>
  <div>Actuelle: {{Nest['Living Room'].Value.ambient_temperature_c}}°C</div>
</div>
<progress class="ui-circle-progress" id="nestProgress" value="{{(Nest['Living Room'].Value.target_temperature_c}}" min="9" max="32"></progress>
```

Ainsi il devient alors possible de contrôler son thermostat en tournant le cadran rotatif de la montre : **[Fig.6].** Pour envoyer la température de consigne, il me suffit d'invoquer le MessageCallback « SetTargetTemperature » au package Nest par la ligne :

```
constellation.sendMessage({ Scope: 'Package', Args: [ 'Nest' ] }, 'SetTargetTemperature', [ deviceid, temperature ])
```

Pour reprendre la sonnette connectée présentée dans l'article précédent, on peut facilement ajouter le contrôle du mode « muet » sur notre montre. Il suffit d'ajouter une checkbox dont l'état est coché si notre StateObject « IsMute » est vrai :

```
<div class="ui-toggleswitch ui-toggleswitch-large">
  <input type="checkbox" class="ui-switch-input" ng-checked="DoorBell.IsMute" ng-click="SetMute(!DoorBell.IsMute)">
  >
  <div class="ui-switch-button"></div>
</div>
```

Et lorsque l'on clique sur la checkbox, on invoque le MessageCallback « SetMute » en passant l'état inverse de notre StateObject :

```
$scope.SetMute = function(isMute) {
  constellation.sendMessage({ Scope: 'Package', Args: [ 'DoorBell' ] }, 'SetMute', isMute);
};
```

De quoi contrôler notre sonnette à n'importe quel moment depuis son poignet : **[Fig.7].** On peut aussi très facilement contrôler toute notre domotique Z-Wave. Pour ce faire, créons une listview (<ul>) pour tous nos StateObjects du package « Vera » afin d'afficher une checkbox liée à l'état de notre module Z-Wave en utilisant un « ng-repeat ».
Et comme pour la sonnette, lorsque l'on clique dessus, on envoie un message au package Vera pour définir le nouvel état :

```
<ul class="ui-listview">
  <li class="li-has-multiline li-has-toggle" ng-repeat="device in Vera">
   <label>{{device.Name}}
     <span class="li-text-sub ui-li-sub-text">{{device.Metadatas.Room}}</span>
     <div class="ui-toggleswitch">
       <input type="checkbox" class="ui-switch-input" ng-click="SwitchVeraDevice(device)" ng-checked="device.Value.Status">
       <div class="ui-switch-button"></div>
     </div>
   </label>
  </li>
</ul>
```

Et là encore, juste avec ces quelques lignes de HTML on obtient une interface de contrôle de tous nos équipements Z-Wave (lumières, appliances, volets) sur notre poignet : **[Fig.8]**
De ce fait, chaque objet, service ou programme connecté dans Constellation peut être facilement intégré dans vos applications Javascript, que ce soit une page Web, une application mobile Cordova/Ionic ou bien comme ici une application Tizen pour votre montre ! Avec les StateObjects produits par le package Tesla présenté dans l'article précédent, et sans rajouter de code, on retrouve toutes les informations sur sa voiture à son poignet (position, état, batterie, clim) avec la possibilité régler le chauffage, le toit ouvrant, faire des appels de phares, klaxonner, etc. Au final avec à peine 150 lignes de Javascript et 350 de HTML, cette application me permet au quotidien de garder un œil sur mes serveurs, ma consommation énergétique et les différents capteurs météo de la maison, voir en temps réel mes caméras de sécurité, piloter mon système d'alarme, les lampes et volets de la maison, le thermostat, l'ampli ou encore porte ma porte de garage. **[Fig.9]** •

StateObjects NetAtmo

Pilotage du thermostat Nest

Pilotage de la sonnette ESP

Pilotage domotique des devices Z-Wave

Menu principal de l'application

Forensic en Python

Franck Ebel
Expert R&D et Formations Serval-concept / serval-formation
Responsable Licence CDAISI, UVHC
Commandant de Gendarmerie réserviste, cellule Cyberdéfense

On désigne par informatique légale ou investigation numérique légale l'application de techniques et de protocoles d'investigations numériques respectant les procédures légales et destinée à apporter des preuves numériques à la demande d'une institution de type judiciaire, par réquisition, ordonnance ou jugement. On peut donc également la définir comme l'ensemble des connaissances et méthodes qui permettent de collecter, conserver et analyser des preuves issues de supports numériques en vue de les produire dans le cadre d'une action en justice.

Nous allons donc tenter de créer des scripts Python pour permettre de trouver des preuves informatiques, de retrouver des éléments dispersés dans le disque dur ou sur d'autres médias.

Modules os.walk()

Le module os.walk() est très utile si nous souhaitons parcourir un répertoire simplement, ou en récursif, ou parcourir tout le disque dur.
os.walk renvoie trois éléments, le path en train d'être listé, les répertoires et les fichiers. Nous pouvons donc récupérer ces trois informations pour les utiliser ensuite. Il travaille en récursif.
Voici un premier exemple d'utilisation simple :

```
import os

for chemin, rep, fichier in os.walk(".", topdown=False):
    print "====================================\n"
    print "résultat pour le chemin :"
    print "====================================\n"
    print chemin
    print "====================================\n"
    print "résultat pour les répertoires :"
    print "====================================\n"
    print rep
    print "====================================\n"
    print "résultats pour les fichiers :"
    print "====================================\n"
    print fichier

franck:articles_programmez franckebel$ python parcours_simple.py
==================================
résultat pour le chemin :
./rep1/rep2/rep3
==================================

résultat pour les répertoires :
[]
==================================

résultats pour les fichiers :
['fichier31', 'fichier32', 'fichier33']
==================================

résultat pour le chemin :
./rep1/rep2
==================================

résultat pour les répertoires :
['rep3']
==================================

résultats pour les fichiers :
['fichier21', 'fichier22', 'fichier23']
==================================

résultat pour le chemin :
./rep1
==================================

résultat pour les répertoires :
['rep2']
==================================

résultats pour les fichiers :
['fichier1', 'fichier2']
==================================

résultat pour le chemin :
.
==================================

résultat pour les répertoires :
['rep1']
```

```
==================================

résultats pour les fichiers :
['079_081_195.pdf', 'article1.pages', 'article1.pdf', 'article1.txt', 'article2_scapy1.odt',
'article2_scapy1.rtf', 'article2_scapy2.rtf', 'encadre_article 1.rtf', 'parcours_
simple.py', 'Prog195-bat.pdf']

franck:articles_programmez franckebel$ nano parcours_simple.py
```

Nous voyons donc que nous pouvons parcourir simplement le disque dur. Nous récupérons trois éléments, le chemin où nous sommes en train de lister, les répertoires contenus à cet endroit ainsi que les fichiers.
Ensuite, s'il existe des répertoires, nous entrons automatiquement à l'intérieur et recommençons le listing.
Nous pouvons ensuite facilement rendre l'affichage plus agréable, nous modifierons donc le script comme ceci :

```
import os

rootDir = '.'
for chemin, repertoire, fichiers in os.walk(rootDir):
    print('répertoires trouvés: %s' % chemin)
    for fnom in fichiers:
        print('\t%s' % fnom)

franck:articles_programmez franckebel$ python parcours_simple2.py
répertoires trouvés: .
    079_081_195.pdf
    article1.pages
    article1.pdf
    article1.txt
    article2_scapy1.odt
    article2_scapy1.rtf
    article2_scapy2.rtf
    encadré_article 1.rtf
    parcours_simple.py
    parcours_simple2.py
    Prog195-bat.pdf
répertoires trouvés: ./rep1
    fichier1
    fichier2
répertoires trouvés: ./rep1/rep2
    fichier21
    fichier22
    fichier23
répertoires trouvés: ./rep1/rep2/rep3
    fichier31
    fichier32
    fichier33
```

Application de os.wlak() pour la crypto

Nous allons utiliser le ROT13 pour faire une application "utile". Qu'est-ce que le ROT13 ?
Le **ROT13** (*rotate by 13 places*) est un cas particulier du chiffre de César, un algorithme simpliste de chiffrement de texte.
Comme son nom l'indique, il s'agit d'un décalage de 13 caractères de chaque lettre du texte à chiffrer. Son principal aspect pratique est que le codage et le décodage se font exactement de la même manière.

Nous pouvons simplement faire du ROT13 avec Python.

```
ROT13 = string.maketrans('abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQ
RSTUVWXYZ', 'nopqrstuvwxyzabcdefghijklmNOPQRSTUVWXYZABCDEFGHIJKLM')
```

Nous allons donc utiliser cette méthode pour essayer de trouver dans notre arborescence des contenus de fichiers qui seraient chiffrés en ROT13.

```
import sys, os, string

ROT13 = string.maketrans('abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQ
RSTUVWXYZ', 'nopqrstuvwxyzabcdefghijklmNOPQRSTUVWXYZABCDEFGHIJKLM')

dictionaryFile = open("dictionary")
dictionary = dictionaryFile.readlines()

for root, dir, files in os.walk(str(sys.argv[1])):
  for file in files:
    if ".txt" in str(file):
      foundWord = 0
      notFound = 0
      lines = open(file).readlines()
      for line in lines:
        translatedLine=line.translate(ROT13)
        translatedWords=translatedLine.split()
        for eachWord in translatedWords:
          if (eachWord+'\n') in dictionary:
            foundWord=foundWord+1
          else:
            notFound = notFound+1
      if (foundWord > notFound):
        print file+" contient peut-être un encodage ROT-13."
```

Recherche et crack de mot de passe de vpn Cisco

Des Librairies spécialisées ont été développées afin de nous aider dans certaines tâches, nous pouvons par exemple associer os.walk() avec les méthode decrypt() de la librairie cisco_decrypt.
L'utilisation en est assez simple, en voici un exemple d'utilisation.

```
Import os, sys
From_cisco_decrypt import CiscoPassword
for root, dir, files in os.walk(str(sys.argv[1])):
  for file in files:
    if ".pcf" in file:
      lines = open(file).readlines()
        for line in lines:
          if "password" in line:
            crack = CiscoPassword()
            password = crack.decrypt(line)
          print plainTextPass
```

La librairie cisco_decrypt se trouve sur ce site :
https://pypi.python.org/pypi/cisco_decrypt/0.8.3

Nous donnerons le chemin du répertoire de recherche en argument de notre script. Nous récupérons cet argument grâce à sys.argv[1].

Python et base64

Nous pouvons transposer l'exemple avec le ROT13 à la base64. base64 est un codage de l'information utilisant 64 caractères, choisi pour être disponibles sur la majorité des systèmes. Défini en tant qu'encodage MIME dans le RFC 2045, il est principalement utilisé pour la transmission de messages (courrier électronique et forums Usenet) sur l'Internet. Il est par ailleurs défini en propre dans le RFC 4648.

Pour coder en base64 en python, nous utiliserons une bibliothèque spécialisée :

```
import base64
msg = raw_input("Entrez un message en base64:")
print "Decoded Val:"+base64.decodestring(msg)
```

Imaginons que nous ayons récupéré des clés asymétriques encodées avec un algorithme type DES ainsi qu'un message que nous supposons encodé avec une de ces clés. Les clés sont dans un fichier texte nommé keys.txt.
Voyons comment associer os.walk() et la bibliothèque spécialisée base64 et crypto.cipher (https://pypi.python.org/pypi/pycrypto).

```
import base64, string
from Crypto.Cipher import DES
THRESH = 0.9

keyFile = open("keys.txt")
keys = keyFile.readlines() ciph=base64.decodestring("cG0okyHpOAADuNLv8bR
pxpyZeU8kMA2kWV2zoV+YUos=")

for key in keys:
  obj=DES.new(key[0:8], DES.MODE_ECB)
  decodedStr=obj.decrypt(ciph)
  foundLetters = 0
  for eachChar in decodedStr:
  if eachChar.isalpha() or eachChar.isdigit()\ or eachChar.isspace():
     foundLetters = foundLetters+1
  if (float(foundLetters)/float(len(decodedStr)) > THRESH):
     print "DES(ciphertext,"+key[0:8]+")="+decodedStr
```

Test des fichiers sur hash.cymru.com

Le site hash.cymru.com permet de soumettre des fichiers et à une base de donnée des bash des malwares connus. Nous pouvons donc en utilisant os.walk(), hashlib() et socket() (pour la connexion réseau) écrire un script pour tenter de découvrir si un de nos fichiers est infecté par un de ces malwares.

```
import os, hashlib, sys, socket, string
for root, dir, files in os.walk(str(sys.argv[1])):
    for fp in files:
        try:
            fn = root+fp
            infile = open(fn, "rb")
            content = infile.read()
            infile.close()
            m = hashlib.md5()
            m.update(content)
            hash = m.hexdigest()
            mhr = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
            try:
                mhr.connect(("hash.cymru.com", 43))
            except:
                print "erreur de connexion au serveur"
            try:
                mhr.send(str(hash + "\r\n"))
            except:
                print "erreur d'envoi"
            response = ''
            while True:
                d = mhr.recv(4096)
                print d
                response +=d
                if d == '':
                    break
            if "NO_DATA" not in response:
                print "<INFECTED>:" +str(fn)
        except:
            pass
```

Le script est assez simple pour les connaisseurs de Python, les autres devront lire ou relire le premier chapitre du livre "hacking et forensic, Développez vos propres outils en Python" aux éditions ENI.
Le résultat devrait être le suivant :

```
franck:articles_programmez franckebel$ python trouve_malware.py /
fde0790600ced3a153e3ccbbbfbb92ea 1475835416 NO_DATA

a35142bb7ae8f572868f746ec86c694c 1475835417 NO_DATA
```

CONCLUSION

Dans ce premier article nous avons vu principalement la méthode walk() de la bibliothèque os. Nous avons aussi abordé le ROT13 et la base64. Nous reviendrons dans les prochains articles sur le forensic en Python, les bibliothèques de cryptographie et nous verrons comment lire les méta données de différents formats comme le pdf, les fichiers office, les photos.
Bon Python ! •

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 contribué à ce numéro : S. Saurel
Nos contributeurs techniques : C. Villeneuve, E. Deneuve, J. Antoine, M. Frappat, P. Batty, T. Lepérou, S. Berberat, P-A. Sunier, M. Roussel, J. De Oliveira, G. Collic, P. Charrière, Core-Techs, K. Sibué, D. Lecan, P. Martin, David Wurteisen, J-F Naud, C. Pierre de Geyer, S. Warin, F. Ebel, F. Botte, M. Bacci, M. Krief
Couverture : © : StockFinland - **Maquette :** Pierre Sandré.
Publicité : PC Presse, Tél.: 01 74 70 16 30, Fax : 01 40 90 70 81 - 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 : 1220K78366 - ISSN : 1627-0908 - © NEFER-IT / Programmez, octobre 2016

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

Sur abonnement ou en kiosque

Le magazine des pros de l'IT

Mais aussi sur le web

Ou encore sur votre tablette

POUR 1 EURO DE PLUS

JUSQU'AU 12 DÉCEMBRE

COMMANDEZ WINDEV 22

OU WEBDEV 22 **OU** WINDEV MOBILE 22

ET RECEVEZ UN SUPERBE MATÉRIEL

POUR 1 EURO DE PLUS

Actuellement sur les routes:
WinDevTour 22. 11 villes. Gratuit.
Inscrivez-vous sur pcsoft.fr

Smart TV **Samsung** Full HD **152cm**
ou Smart TV Samsung 4K 138 cm

Ou choisissez :
- Samsung Galaxy S7 Edge 32+128 Go
- 2x Samsung Galaxy Tab S2 9,7"

Voir le détail sur www.pcsoft.fr

ASUS
IN SEARCH OF INCREDIBLE

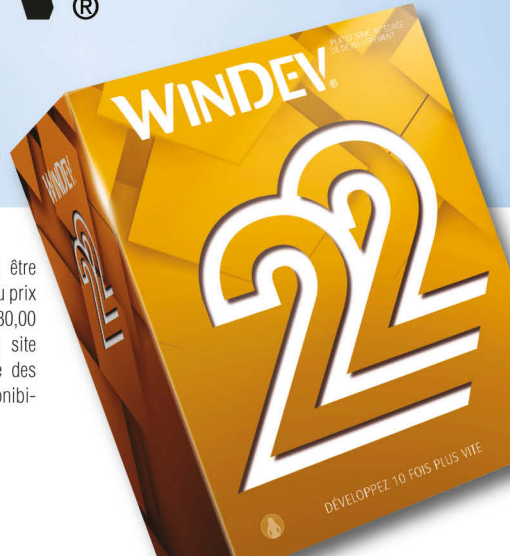
Ultraportable **Asus Zenbook** Aluminium
Windows 10, 13,3", Core i5, 512Go SSD-M2

Ou choisissez :
- Asus Portable 17,3" Core i7 1To
- Asus Mini Tour Core i7 1 To + Ecran 24"

Voir le détail sur www.pcsoft.fr

WINDEV®

Atelier de Génie
Logiciel Professionnel



Pour bénéficier de cette offre exceptionnelle, il suffit de commander WINDEV 22 (ou WINDEV Mobile 22, ou WEBDEV 22) chez PC SOFT au tarif catalogue avant le 12 Décembre 2016. Pour 1 Euro de plus, vous recevrez alors le ou les magnifiques matériels que vous aurez choisis. Offre réservée aux sociétés, administrations, mairies, GIE et professions libérales, en France métropolitaine. **L'offre s'applique sur le tarif catalogue uniquement**. Voir tous les détails et des vidéos sur : www.pcsoft.fr ou appelez-nous (04.67.032.032).

Le Logiciel et le matériel peuvent être acquis séparément. Tarif du Logiciel au prix catalogue de 1.650 Euros HT (1.980,00 TTC). Merci de vous connecter au site www.pcsoft.fr pour consulter la liste des prix des matériels et les dates de disponibilité. Tarifs modifiables sans préavis.

WWW.PCSOFT.FR

Logiciel professionnel. Version 22 annoncée.