The NeceMoon : L'Album

Le Rêve à Temps Plus Que Partiel, le Travail à Temps Carrément Complet

Volume 2

Pleine Lune

(Hardcore)

Par

Necemon Yai

Avec la Participation de

Darren Mart
Jean-Patrick Ehouman
et Holty Sow

Album.NeceMoon.com
Première édition | Evasium ®

The NeceMoon : L'Album

Le Rêve à Temps Plus Que Partiel, le Travail à Temps Carrément Complet

Par Necemon Yai

Première édition

Publié par Evasium ®

Avril 2018

Londres, UK

Album.NeceMoon.com

Le contenu de ce fichier est protégé par la **Loi Britannique de 1988 sur le Droit d'Auteur, les Conceptions et les Brevets**.

Licence

Tu es libre de distribuer et d'offrir ce fichier à autant de personnes que tu veux.
Tout ce que je te demande, c'est de ne pas le vendre, et de ne pas publier le contenu sur un site web.
Si tu utilises une ou plusieurs citations de ce document, mentionne bien la source et le lien vers le fichier original.
Si tu écris un livre et que je te cite, je t'accorderai des mentions et des liens aussi.

© Necemon Yai

necemon@gmail.com

www.necemonyai.com

Tous Droits Réservés

Version 1.0.7.186

A celles que j'ai perdues, à celles que j'ai retrouvées

Une de perdue, dix de retournées

Table des Matières

Introduction

A propos de l'Album "The NeceMoon" : ça c'est quoi ça encore ?

The NeceMoon™ est un Blog à propos de Technologie et de Stratégie. L'Album est le Best-Of, une compilation des articles les plus populaires sur The NeceMoon™.

L'objectif principal de The NeceMoon™, c'est de partager des analyses judicieuses sur divers sujets, pour permettre aux autres d'apprendre de mes erreurs et succès, et j'ose espérer, faciliter la tâche à celui ou celle qui entreprend des choses similaires.
L'objectif principal de l'Album est de favoriser un accès optimal à ce contenu. Le format Blog ne rend pas toujours justice au contenu techno-stratégique. A la base, les Blogs ont été conçus dans un esprit journalistique et sont mieux adaptés aux évènements chronologiques et aux discussions (plus ou moins banales) sur l'actualité. Même si l'utilité et l'importance d'une analyse perdurent dans le temps, l'empilement progressif des articles la rend quasiment introuvable et difficilement consultable.
C'est pourquoi les meilleurs articles ont été repris, revus et agencés dans un ordre logique correspondant mieux à celui d'un livre. L'Album est gratuit.

The NeceMoon™ est disponible sur NeceMoon.com (ou necemonyai.com/blog)
L'Album peut être téléchargé en entier dans différents formats à l'adresse Album.NeceMoon.com (ou necemonyai.com/blog/page/L-Album.aspx). Les formats PDF, EPUB, MOBI/AZW3/KF8 (Amazon Kindle) et MP3 y sont disponibles. Aussi, les différents chapitres et volumes peuvent être téléchargés indépendamment/séparément, selon tes intérêts.

A propos de l'Auteur : qui est Necemon Yai ?

Je suis un ingénieur logiciel à fond dans les technologies Microsoft .NET. Développeur à temps complet. Artiste digital, stratégiste, essayiste et entreprenerd à temps partiel. Je me suis spécialisé en informatique depuis NIIT, Christ University et Swansea University (Master en Génie Informatique). A l'heure où j'écris ces mots, j'ai travaillé pour l'une des principales entreprises de E-Commerce en Europe, pour l'un des groupes financiers les plus importants d'Angleterre, pour la multinationale General Electric et pour quelques startups technologiques dont tu n'as probablement jamais entendu parler.
Depuis une dizaine d'années , je tiens le blog "The NeceMoon", où je décris mes expériences, mes observations et mes réflexions. Je parle surtout de Technologie et de Stratégie. Je partage ici mes articles les plus populaires.

A propos des Contributeurs : qui est dans ton Conseil de Guerre ?

J'ai invité les meilleurs auteurs de mon réseau à inclure des contributions dans ce livre, notamment certaines de leurs pensées les plus pertinentes en termes de technologie et de stratégie. Il s'agit en l'occurrence de, Ahou l'Africaine, Antoine Mian, Cyriac Gbogou, Darren Mart, Edith Brou, Holty Sow, Israël Yoroba, Jean Luc Houédanou, Jean-Patrick Ehouman, Karen Kakou, Monty Oum, Nanda Seye, Nnenna Nwakanma, Olivier Madiba, Vanessa Lecosson et Yehni Djidji.
Au niveau de leurs textes respectifs, tu peux retrouver des liens vers leurs pages Web. De plus, la plupart de ces contributeurs et contributrices se révèlent et te livrent quelques tactiques dans des interviews exclusives que tu trouveras aussi dans ce livre.

A propos de Toi, cher lecteur, chère lectrice : à qui s'adresse ce livre ? Qu'est-ce que tu y trouves ?

Dans L'Album, il y a 9 chapitres organisés en 2 volumes. Chaque chapitre aborde un thème différent. Tu n'es pas obligé(e) de tout lire. Si tu es une personne qui s'intéresse de près ou de loin à un ou plusieurs de ces thèmes, tu apprécierais éventuellement le(s) chapitre(s) concerné(s) :

Volume 1 : Clair de Lune (softcore)
- Chapitre 1 : Stratégies et Tactiques
- Chapitre 2 : Marketing Digital et Visualisations Web
- Chapitre 3 : Carrières et Emergence
- Chapitre 4 : Bons Plans et Petites Victoires Faciles
- Chapitre 5 : Dégammage - Délires et Réflexions Rapides

Volume 2 : Pleine Lune (hardcore)
- Chapitre 6 : Génie Logiciel - Quelques Notions Remarquables
- Chapitre 7 : Programmation informatique avec C# .NET
- Chapitre 8 : Prototypes Epiques, Projets Classiques, Genre Historique
- Chapitre 9 : Recherches et Etudes de Cas

Si tu veux, tu peux télécharger et lire uniquement le(s) chapitres ou volume(s) qui t'intéressent.
Plusieurs formats sont disponibles sur Album.NeceMoon.com (ou necemonyai.com/blog/page/L-Album.aspx)
Tous les liens Web de ce document sont fonctionnels, n'hésite pas à cliquer dessus.

A Toutes Fins Utiles

Ce document contient le Volume 2 de l'Album, il y a 4 chapitres. Si ça t'intéresse, 5 chapitres supplémentaires sont disponibles dans le Volume 1. En fonction de tes intérêts, tu peux télécharger, (re)lire et partager chaque chapitre ou volume indépendamment/séparément. Les formats PDF, EPUB, MOBI/AZW3/KF8 (Amazon Kindle) et MP3 sont disponibles.

Il suffit de cliquer sur le(s) document(s) qui t'intéresse(nt) ci-dessous ou d'aller sur Album.NeceMoon.com (ou necemonyai.com/blog/page/L-Album.aspx).

L'Album est également disponible en Anglais à l'adresse TheAlbum.NeceMoon.com (ou necemonyai.com/blog/page/The-Album.aspx).

Volume 2

Pleine Lune

(Hardcore)

Dans ce volume:

Avec la participation de : Darren Mart, Holty Sow and Jean-Patrick Ehouman

Chapitre 6

Génie Logiciel : Quelques Notions Remarquables

Avec la participation de Darren Mart et Jean-Patrick Ehouman

8 raisons pour lesquelles ça te ferait du bien d'être programmeur(euse)

Par Necemon

Si tu es programmeur(euse), il y a probablement beaucoup de raisons qui te motivent à faire ce que tu fais. J'espère que tu trouveras ici un peu plus de motivation.

Si tu veux être programmeur(euse), tu pourrais y trouver quelques raisons de plus pour opter pour cette voie extraordinaire.

Cependant, je suppose que j'écris ceci surtout pour ceux qui ne savent pas trop quoi faire dans leur vie. Voici quelques indices sur un job qui est amusant, utile et satisfaisant à plusieurs niveaux.

Voici les 8 raisons qui me viennent en tête et qui me font penser que tu aimerais être un programmeur(euse). J'espère que ceci t'inspire.

1. La programmation réalise tes rêves. Quand tu comprends la programmation, tu peux donner vie à tes idées en les appliquant à la vie réelle. tu peux littéralement créer des choses.

2. La programmation est la forme ultime d'art interactif. Tu peux faire des logiciels, sites web et des jeux pour que d'autres puissent jouer avec. Ainsi, tu peux leur parler de manière indirecte et ils peuvent te parler en retour. Aucune autre forme d'art n'est aussi interactive. Alors que le dessin, la peinture, le cinéma et la musique vont à l'audience (dans un sens), le code va dans les deux sens.

3. C'est le genre de job que tu peux faire de partout. Depuis ton canapé, de ta maison, du bureau, en voyage, quelque soit le pays ... Les seules choses dont tu as besoin sont un ordinateur et ton cerveau.

4. C'est facile à apprendre. Il y a des tonnes de ressources disponibles en ligne, beaucoup d'entre elles sont gratuites. Aussi, il y a beaucoup de communautés en ligne qui peuvent te soutenir à travers les forums, chats, courriels, etc.

5. Tu n'as à compter sur personne pour faire de la programmation. Tu peux avoir la chance de le faire pour une entreprise ou lors d'un programme de recherche. Même si ce n'est pas dans un travail d'entreprise, tu peux travailler avec une équipe. Et même si tu ne trouves pas une équipe qui te convient, tu peux toujours travailler par toi-même. Par ailleurs, tu n'as pas besoin de beaucoup d'argent pour commencer.

6. Tu ne peux pas t'en dégoûter. Tu n'as pas le temps de t'ennuyer. Les exigences de projets et la technologie évoluent si vite que tu auras toujours de nouveaux défis à relever.

7. C'est le domaine de la méritocratie. On sait que tu ne simules pas tes compétences. Tu sais ce que tu sais. Tu fais ce que tu peux et tu reçois une reconnaissance équitable pour ça.

8. C'est passionnant !

N.

Ecrire et Programmer, Finalement la Même Chose

Par Jean-Patrick Ehouman

Il y a 5 ans si l'on me disait que je pouvais tenir un blog, je ne l'aurai pas cru. En tant qu'ingénieur de développement, je passais plus de temps à concevoir et écrire des programmes. Mais de là à écrire des articles, à priori rien ne s'y prêtait.

Avec le recul, je peux dire que ces deux activités ont plus de points en commun que de points de différenciation. Lorsque vous écrivez, vous créez, vous remplissez un espace (le blanc de la feuille), vous donnez vie. De la même manière, lorsque vous concevez un programme, vous faites naitre un système qui sera utilisé au quotidien. Vous avez donc dans ces deux cas, le besoin de vous substituer au lecteur ou à l'utilisateur de vos programmes.

Ce jeu de substitution vous emmène à faire preuve d'imagination et de créativité de la même manière qu'un peintre ou un pianiste. Ainsi, lorsque l'on me demande le background pour apprendre à faire de bons programmes informatiques ou des logiciels, je demande à l'intéressé s'il a déjà écrit une lettre, une petite histoire, un article, etc. Si la réponse est « oui » alors la personne possède déjà des prérequis pour apprendre à faire des programmes. Sinon elle peut toujours essayer d'écrire une petite histoire pour juger elle-même de ses capacités à créer ou à innover.

Savoir s'orienter : 5 considérations fondamentales quand on débute en programmation

Par Necemon

Un Freshman (étudiant de première année) m'adresse la requête suivante :
Salut Necemon, merci pour l'acceptation. Je suis nouveau dans le domaine informatique et cela me ferait plaisir que tu m'orientes. Peux tu me dire ce que les sociétés recherchent chez un informaticien, toi qui as de l'expérience professionnelle ?
J'ai entendu parler de toi par les seniors de Christ University, je suis à Bangalore et j'aime l'informatique mais je ne sais pas quoi apprendre et par quoi commencer.
Bon j'aime la programmation mais on me dit que le langage C n'est plus utile, on me parle de Ruby, de C#, de Python, etc. Je suis confus.

Il ne faut pas être confus. La technologie est juste un moyen de résoudre un problème ou d'atteindre un objectif. Quel est ton objectif ?
Créer des applis ? Quelles applis tu veux créer et pourquoi ?
C'est comme si tu viens me voir pour me demander quel véhicule tu devrais emprunter. Si je te demande c'est pour faire quoi, tu ne me dirais pas que c'est pour te déplacer, n'est ce pas ? Je sais que c'est pour te déplacer... Ma question c'est où tu vas ?

Ce que les compagnies recherchent ? Ok, je comprends parfaitement ce que tu es en train de demander. Tu veux t'assurer que la formation que tu suivras te garantira un emploi intéressant plus tard dans le développement informatique. Evidemment, je pourrais te dire qu'une Technologie T est très demandée en ce moment, mais ce n'est pas si simple. Il y a d'autres éléments à considérer :

1. La demande change avec l'endroit (le pays ou la région). Les emplois les plus populaires aux USA ne sont pas forcément les plus populaires en Inde. Donc à moins que tu saches déjà où tu vas travailler, ce n'est pas facile de cibler.

2. La demande change avec le temps. Ce qui est populaire aujourd'hui pourrait ne pas être (aussi) populaire demain. Les technologies évoluent et se remplacent. Donc ce qui est populaire actuellement pourrait être différent de ce qui sera populaire au moment où tu finis tes études.

3. Tu pourrais ne pas aimer la technologie en vogue ou les usages de cette technologie. Si je te dis qu'une technologie T est très demandée, ça permet de repérer et réparer des bugs/erreurs dans un système super ennuyeux/énorme/compliqué de transactions bancaires, il y a plein de calculs... Mais si toi tu n'aimes pas les calculs, est ce que tu vas quand même adopter cette technologie et accepter cette voie pour le reste de ta carrière ?

4. Même si on considère une seule ville à un moment donné, différentes compagnies recherchent différentes choses, tout dépend de ce que la compagnie fait. Il n'y a pas une technologie parfaite qui est meilleure que toutes les autres dans tous les domaines. Chaque technologie a ses avantages et ses inconvénients. Il y a certaines choses que C et C++ font mieux que Ruby (et vice versa), il ya certaines choses que Python fait mieux que C# (et vice versa), etc.

5. Comme je le disais plus haut, la technologie, c'est juste un moyen pour arriver quelque part. Quand tu considères Facebook par exemple, la plupart des utilisateurs s'en fichent, si ça été construit avec PHP, C, Java, Perl, C# ou Python. Ce qui est important pour les gens, c'est comment le site ou l'application peut les aider dans leur vie.

C'est par là que tu devrais commencer, je crois. Quelles sont les atouts que tu as déjà ? (ne me dis pas que tu n'en as pas), Quelle contribution tu comptes apporter à ta famille, à tes amis, à ta communauté, à ton pays, au monde ? Et qu'est ce que tu attends en retour ?

Si tu ne sais pas que faire de ta vie, j'ai écris un article il y a quelques temps qui pourrait peut être t'inspirer : Qu'est ce que tu vas faire dans la vie ? Prends le temps de réfléchir sur tes ambitions et on pourra parler des ressources dont tu auras besoin.

Si tu sais CE QUE tu veux faire, ce serait plus facile pour moi de t'expliquer COMMENT le faire.

A bientôt,

N.

Quand les judokas se mettent à la programmation

Necemon:
mais je maintiens ce que j'ai dit
la programmation c'est comme le judo
Banglet:
soit plus explicite
Necemon:
ok
donne moi 2 techniques de judo
2 techniques différentes
ok
laisse moi choisir
Banglet:
tu m'envoies loin la
Necemon:
disons o soto gari
Banglet:
je sais pas comment on écrit
Necemon:
et o goshi
tu te souviens de ces techniques ?
Banglet:
t'as les vrais souvenirs !
Necemon:
?
Banglet:
humm
pas vraiment
Necemon:
osoto gari
Banglet:
le nom me dit quelque chose en tout cas
Necemon:

voici o soto gari
o goshi c'est ce qu'on fait avec la hanche en se retournant
alright ?

Banglet:
ok
Necemon:

voici o goshi
Banglet:
o soto gari
j'aimais bien ça
Necemon:
ok
donc c’est ça qu’on va prendre comme exemple
Banglet:
ok
Necemon:
quelqu'un qui maitrise son o soto gari
il peut frapper des gens qui connaissent juste un peu de toutes les autres techniques
à force de pratiquer ça, le geste devient de plus en plus précis
il devient un expert
Banglet:
ouais
Necemon:
mais ça ne veut pas dire que o soto gari est la meilleure technique du monde
o soto gari c’est mieux si le gars recule
si il fonce sur toi, c’est clair que o goshi est mieux
donc celui qui maitrise le o soto gari et qui veut te battre avec
va se débrouiller pour toujours te faire reculer
c'est pareil pour les langages de programmation
celui qui est fort en C sera plus à l'aise dans les projets où C est le langage adéquat
par exemple C est connu pour échanger plus directement avec les périphériques et la mémoire
C est aussi très bon pour les applications graphiques
si le mec passe des années à faire ce genre d’applications avec C, quand tu vas voir de quoi il est capable
tu vas te dire, "wow c’est un champion…"
mais ça ne veut pas dire que C est le meilleur langage du monde
il y a des applications ou Java peut être plus adéquat que C
et puis quand je dis que c’est comme au judo
ce n'est pas que le langage qui compte, il y a aussi l'expérience
un grand maitre peut te clouer au sol avec un bras
de la même manière, un expert de C qui a pratiqué le langage pendant des décennies peut faire des merveilles

que toi tu ne peux pas faire même si tu utilises une librairie graphique de C++ genre Qt
tu comprends ?
Banglet:
un peu un peu
Necemon:
pour compléter avec ce que je viens de dire aujourd'hui
le programmeur parfait utilise la bonne technique au bon moment
de la même façon, le judoka parfait utilise la bonne technique au bon moment
exemple simple : si tu avances, il te fait o goshi
tu recules, il te fait o soto gari
Banglet:
et c'est IPPON !
Necemon:
lol
j'ai dit "le programmeur parfait utilise la bonne technique au bon moment"
je voulais surtout dire "le programmeur parfait utilise le bon langage au bon moment"
Banglet:
ok ok
mais comme la perfection n'est pas de ce monde...
il vaut mieux maitriser un langage
et la question c'est de savoir lequel maitriser
Necemon:
la perfection n'est pas de ce monde donc pourquoi tu cherches le langage parfait ?
Banglet:
looooooooool
bon disons le langage qui tend vers la perfection
Necemon:
la façon pour toi de tendre vers la perfection c'est de choisir au mieux ton langage en fonction de la situation
en fonction de tes besoins
et surtout en fonction de tes gouts
c'est important que tu fasses quelque chose que tu aimes
comme au judo ou tu as une technique que tu aimes bien
dans laquelle tu te sens à l'aise
en programming ce que tu dois te demander c'est
qu'est-ce que je veux faire au fait ?
et ensuite, tu te donnes les moyens de faire ce que tu veux faire de façon optimale
si tu construis un logiciel pour un client
il s'en fiche de savoir si ton langage était orienté objet
si tu as utilisé des pointeurs
des librairies graphiques
des génériques
etc.
Banglet:
ça c'est pas faux
Necemon:
bref, toutes les technologies de ton langage préféré dont tu es si fier, ton client s'en fiche
Banglet:
lool
Necemon:
la question c'est, est ce que ton logiciel correspond à ses besoins, ses objectifs ?

est ce que tu as marque' IPPON ?
c'est ça qui compte
maintenant comme je t'ai dit
tu peux t'orienter
comme je sais que tu aimes O Soto gari, ça veut dire que tu dois aimer faire reculer l'adversaire
de cette même façon, si tu aimes Java, tu dois savoir en quoi Java est bon
et qu'est ce que tu peux faire de beau avec Java
si tu veux je peux t'expliquer en quoi C# est bon
et pourquoi j'aime C#
mais je ne prétends pas que C# est le meilleur langage du monde
ce que je sais
c'est qu'il me convient
et qu'il me rend très efficace
je ne peux pas garantir qu'il te convient
parce que tout dépend de ce que toi même tu veux
et de ce qu'on attend de toi

Banglet:

ok ok

Création De Jeu Vidéo : 4 Tactiques Substantielles Qui Vont Forcément Booster Tes Compétences

Par Necemon

Un utilisateur de l'une de mes applications web m'a fait part de sa préoccupation:
Bonjour, je suis T. j'ai 15 ans, quand je serai grand j'aimerais faire partie de l'industrie du jeu vidéo et je me demandais si vous pouviez m'aider. Pouvez-vous me donner quelques conseils et partager votre expérience avec moi ?
Est-ce que le fait de publier sur votre plateforme pourrait m'aider dans cette situation? J'aimerais beaucoup lire votre réponse, ça m'aidera vraiment.

Telle fut ma réponse:

Il y a plein de chose à dire sur le sujet mais aujourd'hui j'essaierai de faire simple en te donnant quelques conseils pratiques que tu pourrais utiliser dès maintenant.

1. **Lis beaucoup sur les sujets qui t'intéressent.** Lis chaque jour si tu peux. Il y a des tonnes de ressources gratuites en ligne. Pour ce qui est de la création de jeux vidéo, certains de mes blogs et sites favoris incluent:
gdne.ws
lostgarden.com
procworld.blogspot.co.uk
whatgamesare.com
gamestudies.org
designer-notes.com
higherorderfun.com
gamecareerguide.com
webwargaming.org
raphkoster.com/gaming

gamedevelopment.tutsplus.com
nwn.blogs.com

2. **Etablis des objectifs précis.** Tu remarqueras que dans le premier point, je t'ai simplement donné une petite partie des informations disponibles, mais c'est déjà relativement étendu. Nous vivons des temps exponentiels, tu ne pourrais pas tout faire et tout apprendre à la fois. Tu dois donc être précis dans ce que tu veux. Les projets vagues provoquent des résultats vagues. Qu'est ce que tu aimerais faire dans l'industrie des jeux vidéo? Il existe plusieurs options sur lesquelles tu pourrais faire quelques recherches. Nous vivons des temps exponentiels, tu ne pourrais pas tout faire et tout apprendre à la fois. Tu dois donc être précis dans ce que tu veux. Les projets vagues provoquent des résultats vagues. Qu'est ce que tu aimerais faire dans l'industrie des jeux vidéo? Il existe plusieurs options sur lesquelles tu pourrais faire quelques recherches.

3. **Choisis tes étoiles.** Pour rebondir sur le point précédent, après avoir fait une liste des compétences que tu voudrais acquérir, fais quelques recherches sur les personnes qui les maitrisent déjà. Trouves toi des modèles de référence, vois comment ils ont commencé et comment ils ont procédé. Fais la combinaison de leurs méthodes et ajoute ta touche personnelle pour développer tes propres techniques. Cela peut prendre des mois, mais si tu t'y mets de façon constante, tu deviendras aussi très bon. Quelques personnes sur qui tu pourrais t'informer:

Développement de jeux:
- Shigeru Miyamoto
- Hideo Kojima
- Notch (Markus Persson)

Artistes digitaux:
- Akira Toriyama
- Masashi Kishimoto
- Monty Oum

Scénaristes:
- J.K. Rowling
- Stephen King
- Tom Clancy

4. **Construis un jeu.** Non, tu n'es pas trop jeune pour commencer. Pas besoin de débuter avec quelque chose d'immense dès le départ, tu peux déjà commencer à apprendre. Plus tu t'y mets tôt, plus tu as le temps de t'exercer et plus tu pourras devenir excellent rapidement. Tu peux débuter avec quelque chose de vraiment basique ou au moins lire des documents sur la procédure. Si tu veux des conseils sur comment débuter, comment trouver des tutoriels et documents, je peux t'aider. A ce propos, j'ai crée un jeu: babifraya.com Il n'est pas extraordinaire et je travaille présentement sur une version un peu plus améliorée. Mais le plus important, c'est de commencer et de continuer à pratiquer :-)

Pour résumer, approfondis tes connaissances continuellement.

Quant à savoir si le fait de poster sur degammage.com t'aidera, ma réponse honnête serait: peut-être. Tu vois, c'est un projet qui en est à ses débuts et c'est trop tôt pour évaluer à quel point il aura du succès. Toutefois, il y aura du contenu frais chaque jour, donc tu auras de nouvelles choses à apprendre, et tu pourrais aussi visiter flux.evasium.com qui est plus orienté sur le monde virtuel, l'éducation et la technologie. Ton compte Evasium fonctionne là également.

En plus, tu pourrais aussi rejoindre d'autres communautés et forums de jeux comme ceux que j'ai listé plus haut. Apprends autant que possible!

10 Ans De Programmation

Par Necemon

J'ai commencé la programmation quand j'étais au lycée. On nous apprenait, entre autres, Pascal et HTML. Entre temps, pendant les vacances d'été, je prenais des cours de programmation dans un institut TIC. C'est là-bas que j'ai appris Visual Basic et la conception de logiciels et de bases de données (A ce temps là, on utilisait MERISE (Méthode d'Etude et de Réalisation Informatique pour les Systèmes d'Entreprise)). Un jour, j'ai pu obtenir le CD d'installation de Visual Basic, donc je l'ai installé sur mon PC et j'ai commencé à pratiquer tout seul, avec des bouquins.
Mais, ce n'est qu'après le bac que j'ai commencé à apprendre et à appliquer C# presque à plein temps pendant les études supérieures, et c'est ça que je considère comme mon véritable début en programmation et ingénierie logicielle.

J'ai appris quelques leçons au cours de la dernière décennie, et je me suis dit que je prendrais un moment pour recueillir mes réflexions sur ces sujets. Il m'a fallu environ dix ans et beaucoup d'expérimentation pour assimiler certaines de ces choses.

1. Apprendre un langage de programmation, c'est la partie la plus facile: attention aux plateformes
Prenons C# par exemple. Apprendre le langage C# n'est pas difficile. Si tu as déjà une bonne compréhension des fondamentaux de langues informatiques, et si tu as un peu d'expérience dans d'autres langues orientées objet, tu peux devenir un programmeur C# compétent en quelques jours, du moins en ce qui concerne le langage en lui-même. Cependant, le vrai prix à payer de l'apprentissage n'est pas dans le langage, c'est dans la plateforme. Pour développer avec C# sur .NET, tu dois connaître:

- le Framework .NET
- une ou plusieurs technologies .NET telles que ASP.NET ou WPF
- et l'environnement de développement Visual Studio.

Le temps requis pour devenir compétent dans le développement sous .NET se mesure généralement en mois, même pour un développeur expérimenté. Apprendre une plateforme est toujours plus coûteux que d'apprendre un langage spécifique, donc choisir la plateforme est la décision la plus cruciale.
L'apprentissage a toujours un coût et ce coût est l'un des facteurs clés à prendre en considération lors du choix de la technologie que tu souhaites apprendre. Le coût réel de l'apprentissage est dans le temps, l'apprentissage prend toujours du temps. Puisque tu n'as pas le temps de tout apprendre, il est important de penser stratégiquement à ce que tu veux apprendre. Et puisque les langages sont faciles, c'est surtout aux plateformes qu'il faut faire attention : les technologies associées au langage, les outils de développement et de déploiement, les systèmes d'exploitation, et autres infrastructures.

2. Je répète, apprendre un langage de programmation est la partie la plus facile: à la rencontre des concepts fondamentaux de l'ingénierie logicielle
La syntaxe en elle-même, les mots que tu utilises lorsque tu utilises le langage sont relativement simples et tu peux les apprendre facilement. Cependant, c'est loin d'être suffisant pour produire du code de qualité, qui implique souvent des principes OOP, TDD, BDD et SOLID, des tests unitaires, des patrons de conception et d'autres concepts techniques qui dépassent le cadre de cet article. Enfin, bref.

3. En fait, écrire du code n'est qu'une (petite) partie du travail
Un ingénieur logiciel est souvent impliqué dans la recherche technologique, la configuration d'outils et de projets, les tâches d'administration et de déploiement, les procédures de débogage et d'essai, la documentation et la dette technique (correction et refactorisation du code existant). De plus, il doit réfléchir à des solutions et concevoir des systèmes: parfois, le travail le plus important se fait pendant qu'on est loin du clavier.

4. Recettes éprouvées et efficaces : les vieilles techniques ennuyeuses sont parfois les meilleures.
Ce n'est pas vraiment "vieux contre nouveau", ni même "cool vs ennuyeux", mais plutôt la technique avec laquelle tu as le plus d'expérience. Comme disait l'autre, je ne compte pas sur le codeur qui a pratiqué 1000 technologies une seule fois mais sur celui qui a pratiqué la technologie adéquate 1000 fois. Si l'objectif est de construire un truc de manière aussi effective et rapide que possible, ce serait plus productif d'utiliser les technologies que tu maitrises le mieux.
Par exemple, un de mes contacts se fait 25 000 $ par mois avec un SaaS qu'il a construit avec une combinaison ennuyeuse : ASP.NET + SQL Server + Angular 1, parce que c'est ce qu'il connaissait. Il l'héberge sur Windows, parce qu'il sait comment rendre Windows rapide et sécurisé. Il a réussi parce qu'il consacre tout son temps à construire des fonctionnalités que ses clients réclament, plutôt que d'apprendre les technologies les plus en vogue.
Il est important de se rendre compte que le tapis roulant de la technologie ne s'arrête jamais, il y a toujours de nouvelles choses à apprendre. En fait, un nouveau Framework JavaScript est probablement en train d'être lancé pendant que tu es en train de lire cet article. Les technologies de pointe les plus populaires aujourd'hui n'existaient même pas lorsque je débutais (EF Code First, Xamarin, ASP.NET Core, Razor), ce qui nous conduit aux 2 points qui suivent.

5. Concentre-toi sur les technologies durables
La seule constante dans le monde, c'est le changement. La gestion du temps et des actions est une compétence importante chez les développeurs, en particulier, parce que nous sommes sur un tapis roulant technologique qui ne cesse de bouger, voire d'accélérer.
Par exemple, les technologies Web qui étaient populaires vers l'an 2000 (Flash, ASP Classic et Java Applets) deviennent quasiment obsolètes et de moins en moins professionalisantes. Aujourd'hui, on parle de ASP.NET Core, SignalR, Angular2, React et VueJS. Aucune de ses technologies n'existaient en l'an 2000, et ces nouvelles technologies seront probablement obsolètes d'ici 10 ans.
Qu'est ce qui n'a pas vraiment change' ? Les fondamentaux de langages tels que C++/C#, leurs implémentations d'algorithmes et leurs principes sont toujours aussi pertinents sur plusieurs dizaines d'années. Si tu maitrises les bases d'un système stable, tu pourrais mieux t'adapter au changement, tu pourrais l'apprécier et t'en servir pour évoluer.

6. Équilibre entre exploration et exploitation
L'exploration consiste à apprendre de nouvelles choses, à étudier de nouvelles techniques, à lire des livres, à regarder des tutoriels, à pratiquer et à améliorer ses compétences. L'exploitation, au contraire, consiste à tirer parti de ce que nous savons déjà pour régler de vrais problèmes. Il s'agit de penser créativement à des façons d'utiliser les connaissances que nous avons déjà pour créer de la valeur pour les autres.
Donc, oui, ces deux tâches sont à la fois nécessaires et importantes. Le risque, c'est d'être trop fixé sur l'une ou l'autre de ces activités.
Trop d'exploration, et tu ne pourras jamais atteindre un niveau d'expertise utile dans une technologie donnée. Il y a un coût d'opportunité énorme avec cette sorte d'apprentissage léger, puisque, bien que ça t'élargisse l'esprit, le temps que ca nécessite implique que tu perfectionnes moins les compétences que tu as déjà acquises.
En revanche, une trop grande exploitation peut t'empêcher d'évoluer dans les nouvelles technologies, et peut limiter tes opportunités d'emploi.

7. C'est facile d'être excellent... C'est difficile d'être constant.
C'est facile d'être excellent pendant 2 minutes. C'est difficile de l'être constamment, chaque jour. Quand tu es habité par une bonne idée pour un nouveau projet, tu ressens une grande envie de commencer la recherche, le design et la programmation. Tu as hâte de transformer ton idée en quelque chose de réel et tu deviens super productif. Mais le problème est que cette motivation se fane avec le temps.
Oui, c'est marrant et c'est facile d'avoir de nouvelles idées et de commencer à y travailler. Mais ensuite, il y a les efforts à fournir, les ajustements, le lancement, la maintenance, les corrections, les améliorations, etc. Sur plusieurs mois. C'est là que ça devient dur. C'est dur de rester concentré sur une même idée, sur un même projet pendant des mois et des années. Ça demande beaucoup de discipline.
C'est facile d'être excellent. C'est difficile d'être constant.

8. Diversifie tes compétences
Ne sois pas seulement un programmeur, devient un Expert Qui Programme, un expert dans un autre domaine pertinent dont tu es passionné. Tu peux être un entrepreneur, un chef de projet, un scientifique en Big Data, un chercheur, un spécialiste de la sécurité, etc.
Si tu es un Expert Qui Programme, en plus de pouvoir programmer (peut-être à temps plein), tu as également une crédibilité supplémentaire qui est liée à d'autres domaines, au dela de l'ingénierie logicielle. D'où l'importance de poursuivre des études supérieures. Si tu vas à l'Université ou en Grande École alors que tu sais déjà programmer, tu n'apprendras probablement pas grand-chose à propos de la programmation. Mais ça ne veut pas dire que tu ne devrais pas aller dans ces écoles. Tu auras besoin d'une certaine culture, et les universités sont d'excellents endroits pour l'obtenir. Tu acquiers la culture en étudiant et en comprenant le monde que les humains ont créé, sous différents angles. Ce serait difficile d'acquérir ce genre de connaissances si tu ne fais rien d'autre qu'étudier la programmation.

9. Choisis des niches pour te démarquer
Plus la niche est petite, plus tu es perçu comme un agent exceptionnel dans ton domaine. Par exemple, il est très difficile pour des développeurs de se démarquer avec un titre tel que "Développeur Web PHP". Ils sont compétents, polyvalents, utiles, mais pas remarquables. Ils se sentent facilement remplaçables parce qu'il y en a tellement, avec un ensemble de compétences comparables. Le domaine est trop large pour que tu puisses te démarquer facilement du lot. Si, par contre, tu deviens reconnu dans une niche, comme Xamarins.Forms ou Visualisations JavaScript, tu serais beaucoup plus apte à te distinguer par ceux qui recherchent spécifiquement ces compétences.

10. L'âge des Compétences
L'information est la connaissance spécifique dont tu as besoin pour résoudre des problèmes. Les compétences représentent la capacité de mettre en œuvre des solutions en utilisant tes connaissances. Dans un monde où la plupart des connaissances et des outils sont quasiment gratuits, qu'est-ce qui fait la différence ? C'est la compétence, bien évidemment. Nous ne sommes plus une société fondée sur le savoir, nous sommes une société axée sur les compétences. Il fut un temps où presque tous les diplômes universitaires garantissaient un bon travail. Maintenant, ce n'est plus le cas. On s'en fiche de ce que tu sais. On s'intéresse à ce que tu sais FAIRE. On te paye pour faire des choses, pas pour connaître les choses.

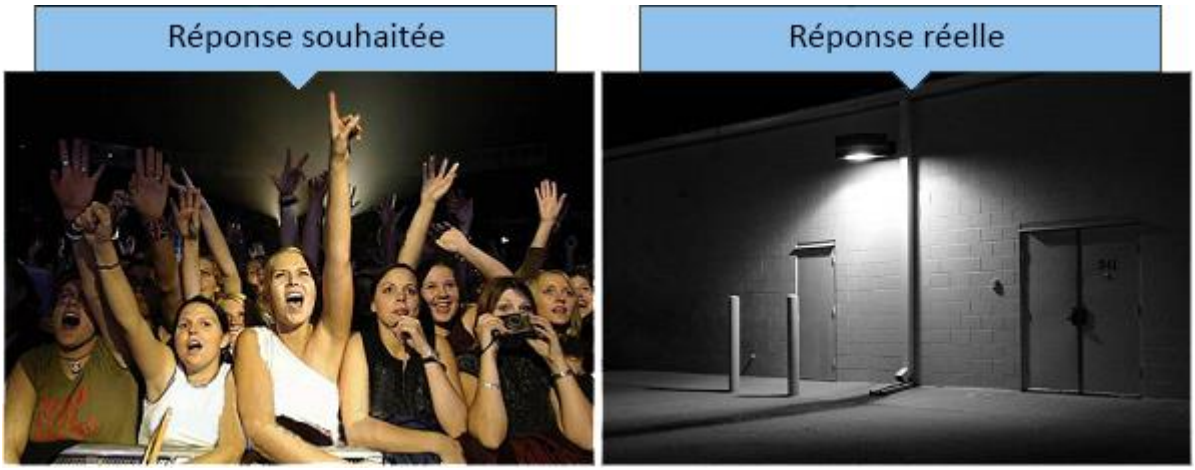
Apprendre à programmer en 10 ans.
Selon plusieurs recherches, il faut environ 10 ans (ou 10 000 heures) pour développer une expertise.
La solution, c'est la pratique réfléchie : il ne suffit pas de refaire les mêmes choses encore et encore, mais plutôt de se défier avec une tâche qui dépasse sa capacité actuelle, l'essayer, analyser ses performances pendant et après, et corriger toute erreur. Ensuite, répéter. Et recommencer. Il semble qu'il n'y ait pas vraiment de raccourcis. L'apprentissage par la lecture, c'est bien. Mettre la main à la pâte, c'est mieux. Le meilleur type d'apprentissage est l'apprentissage par la pratique.
Personnellement, les petits projets perso et les prototypes m'ont effectivement aidé à m'améliorer. Mais il y a encore beaucoup de choses intéressantes à maitriser donc continuons d'apprendre.

Faut-il construire ton site même si personne n'y vient ?

Par Darren Mart

Il n'y a pas si longtemps, j'ai reçu un e-mail d'un développeur énergique et talentueux qui s'attaquait à un projet logiciel ambitieux. Il m'a demandé si j'avais un conseil à lui donner.

Tous les souvenirs ont commencé à me revenir en masse. Les rêveries. Sortir du lit au beau milieu de la nuit parce qu'une idée ne peut pas attendre. Des mois de travail minutieux qui poussent ce bébé à ne plus vaciller et à marcher seul. Tout ça pour un public enthousiaste. Sauf que...



Le cycle après le cycle

J'avais peut-être rédigé une demi-douzaine de réponses à son e-mail. J'ai admiré son enthousiasme et compris où son esprit en était. Mais je savais aussi ce qui l'attendait probablement après le cycle de développement ardu: un autre cycle de turbulence, une série d'affrontements entre les prévisions et les tristes réalités.

Qu'est ce qui n'a pas marché?

C'est une question dangereuse que nous nous posons en tant que développeurs. "Dangereuse" parce qu'elle présuppose que nous avons fait quelque chose de travers. Il est tout à fait possible que ce que tu as conçu soit assez bon, peut-être même brillant. Le problème n'a peut-être rien à voir avec la qualité de ton travail, mais plutôt avec tes critères de réussite.

Alors, comment un développeur peut-il mesurer précisément le succès d'un projet solo? J'aurai aimé avoir une réponse concrète à te donner. Par contre, je peux te donner quelques conseils pour ce qui est de comment *ne pas* le mesurer:

Astuce 1: Méfies-toi des sirènes des plateformes sociales

Ah, mais elles nous leurrent avec la promesse d'une acceptation et d'une consommation de masse. Nous réalisons rarement leur capacité à berner nos esprits. Il est possible que tu puisses passer des mois sur un projet, que tu le partages sur Facebook avec la vigueur et l'excitation d'un chien qui tient un jouet à mordiller, et que tu observes le nombre de "J'aime" monter en flèche jusqu'à... 3. L'un des "j'aime" vient de ta mère, un autre provient d'une page que tu as conçue pour promouvoir ton projet, et l'autre d'un tapotement hasardeux sur un téléphone portable.

C'est le même effet de refroidissement que te font les photos de parcs d'attractions abandonnés. Tu essaies d'imaginer des fous rires et une excitation effrénée, et tu as du mal à accepter ce qui se trouve juste sous tes yeux.

Pendant ce temps, le nouveau et très profond statut de ton pote disant "J'aime les raviolis!" vient d'obtenir 23 likes et 16 commentaires. La popularité et la substance sont deux choses différentes, et nous devons simplement faire avec.

Astuce 2: La réaction apathique n'a rien de personnel (elle pourrait même ne pas être apathique)

Tu es déjà conscient de ça, mais il est bon de s'en rappeler. La plupart des gens n'ont aucune notion de ce qu'il faut pour réussir ce que tu as accompli. Ils ne réalisent pas que tu as construit à toi seul quelque chose qui rivaliserait avec les efforts de toute une équipe. Tu ne peux pas, et tu ne vas pas, les entraîner à s'y intéresser sincèrement.

Les applications qui agitent le monde relèvent généralement de deux catégories: celles qui aident les utilisateurs à se promouvoir eux-mêmes et celles qui nécessitent peu de réflexion. Si ton projet nécessite un investissement mental supérieur à celui de Candy Crush Saga, voici la dure réalité: tu vas te retrouver avec beaucoup de restes de nourriture lors de ta soirée de lancement.

Astuce 3: Réfléchis à deux fois avant de solliciter les commentaires de tes confrères

Dans le film "Minuit à Paris", l'écrivain en herbe Gil demande à Ernest Hemingway de lire son roman et de donner son opinion.

- Hemingway: Mon opinion est que je le déteste.
- Gil: Ah bon? Mais vous ne l'avez même pas lu.
- Hemingway: Si c'est mauvais, je le détesterai parce que je déteste quand c'est mal écrit, et si c'est bon, je serai jaloux et je le détesterai d'autant plus. Tu ne veux pas l'opinion d'un autre écrivain. Les écrivains sont compétitifs.

Ne néglige pas la profondeur de cet échange. Ceci s'applique aussi aux développeurs.

Astuce 4: Qu'on le veuille ou non, tu es un artiste

Si tu te dévoues corps et âme à la réalisation de projets solo parce que tu aimes le défi créatif, alors tu es un artiste. Si tu le fais parce que tu essaies de résoudre un problème, c'est que tu es probablement un hybride artiste-ingénieur. Si tu le fais strictement pour l'argent, tu as sûrement abandonné cet article depuis longtemps, donc peu importe comment je t'appelle.

Tout comme les peintres, les écrivains, les décorateurs et les chefs gastronomes du monde entier, nous espérons secrètement que le public sera élevé et inspiré par nos créations.

Mais il y a une différence clé. Si quelqu'un n'est pas inspiré par une peinture, il serait quand même capable d'apprécier l'effort individuel. Avec un logiciel, il n'y a pas ce luxe. Peu importe si tu es le Picasso du monde logiciel, ton appli est très nulle par rapport à Office 365 ou Google Maps ou Skyrim, peu importe la flotte de ressources requises par ce dernier.

Donc... Faut-il continuer à construire ton projet même si personne n'y vient ?

Oui. Parce qu'en tant qu'artistes, nous sommes enrichis par le processus global quel que soit le résultat final. Aussi banal que cela puisse paraître, nous avons ce sentiment d'accomplissement et la fierté de savoir à quel point il a fallu s'autodiscipliner pour poursuivre le projet jusqu'au bout. Nous avons stimulé notre intellect, nous avons appris ce qui fonctionne et ce qui ne fonctionne pas, nous avons acquis une expérience pratique. Le prochain projet, qu'il soit personnel ou professionnel, en récoltera les fruits.

Certifications Microsoft pour Développeurs

Par Necemon

Microsoft offre une large gamme de programmes de certification conçus pour t'aider à développer tes compétences et ta carrière.

Cet article se concentrera sur les certifications de développeurs. MCSD (Microsoft Certified Solutions Developer) est une certification destinée aux professionnels de l'informatique qui cherchent à démontrer leur capacité à créer des solutions innovantes dans de multiples technologies. Par exemple, la certification MCSD App Builder confirme que tu as les compétences nécessaires pour créer des applications et des services mobiles modernes.

J'ai reçu ma première certification Microsoft en 2008. En progressant, j'en ai obtenu d'autres au fil des années et je suis maintenant un Développeur de Solutions Certifié. Est ce que ça en valait la peine ? Parfois, ça n'avait pas d'importance, parfois c'était très utile.
Selon mon expérience, il y a 5 avantages à obtenir une certification:
- Obtenir une certification professionnelle Microsoft ne garantit pas l'obtention d'un emploi, d'une augmentation ou d'une promotion, mais ça augmente les chances.
- Quand il s'agit d'embaucher, certaines entreprises demandent s'il y a des certifications, en plus de l'expérience. De nombreux recruteurs vérifient les certifications parmi les candidats à l'emploi et considèrent ça comme une partie de leurs critères d'embauche. Certains considèrent les certifications informatiques comme une priorité lors de l'embauche de postes informatiques.
- Obtenir une certification peut augmenter la confiance en soi, ta confiance en tes compétences.
- Cela montre du sérieux et de la passion (tu l'as fait tout parce que tu voulais, et non parce que tu devais).
- Le processus de préparation à la certification augmente tes compétences théoriques et pratiques.

Conseils et astuces pour la préparation des examens
Ma recommandation principale est de consacrer une période spécifique à la préparation de l'examen, quelques jours avant. Trouve des livres, participe à des cours (les cours virtuels sont une préférence personnelle, mais les cours en classe sont aussi disponibles) et entraine-toi sur les technologies concernées.
- Livres: il y a souvent des livres de préparation liés à l'examen donné. Par exemple : Examen 70-740 - Installation, gestion du stockage et des traitements sur Windows Server - Préparation à la certification MCSA
- Cours: les examens les plus populaires ont des cours courts disponibles en ligne. Par exemple : Le cours 20487B: developpement de Services Web
- Au-delà du matériel de formation, c'est bien de faire des recherches sur les sujets concernés
- S'exercer sur lesdites technologies
- Répondre aux questions à choix multiples: procéder par élimination. En cas de doute sur la bonne réponse, exclure les réponses qui n'ont pas de sens (ou qui ont moins de sens)

Chemins de carrières pour les développeurs
Tu devrais peut être vérifier cette partie. Ils mettent les offres à jour assez régulierement, en fonction des technologies les plus récentes, mais au moment ou j'écris ces mots, il existe 5 options pour devenir MCSD (Microsoft Certified Solutions Developer)
- MCSD: applications Web: expertise dans la création et le déploiement d'applications et de services Web modernes.
- MCSD: applications Windows Store: expertise dans la conception et le développement d'applications Windows rapides et fluides. Il existe deux moyens pour obtenir cette certification, en utilisant soit HTML5 ou C#.
- MCSD: applications SharePoint: expertise dans la conception et le développement d'applications de collaboration avec Microsoft SharePoint.
- MCSD: Azure Solutions Architect: expertise sur toute l'étendue de l'architecture, le développement et l'administration de solutions Azure.
- MCSD: gestion du cycle de vie des applications: expertise dans la gestion du cycle de vie complet du développement d'applications.

Planificateur de certification
Le Planificateur de certification est un outil pour t'aider à connaître les conditions requises pour ta prochaine certification. Il s'agit de planifier tes prochaines étapes (tes options, les examens à suivre, dans quel ordre, etc.).

Autres liens et références:
Certification MCSD
Certification WebApps
Page Wikipedia

C'est tout pour le moment. Maintenant retourne travailler.

Chapitre 7

Programmation informatique avec C# .NET

Avec la participation de Holty Sow

Pourquoi j'aime tant C#

Par Necemon

Si tu ne t'y connais pas en programmation, je devrais commencer par te dire qu'un langage, c'est juste un système de communication et un langage de programmation, au fond, c'est un langage artificiel conçu pour communiquer des instructions à une machine, typiquement un ordinateur.

Il existe de nombreux langages de programmations. Certains sont plus populaires que d'autres. Certains sont plus récents, certains sont plus puissants dans une certaine mesure.

Dans un monde idéal, chaque langage de programmation a un but précis. Donc un ingénieur devrait être capable de s'adapter au projet en cours et d'utiliser les technologies optimales pour ce projet. Mais la vérité est que, très souvent, nous avons tendance à nous familiariser avec tel ou tel langage et nous trouvons que certains langages sont pénibles ; ça dépend des caractéristiques du langage et du temps qu'on a passé à les utiliser. C'est un peu comme les langages naturels (les langages humains). Tu peux apprendre plusieurs langues (anglais, italien, espagnol, allemand, japonais, etc.) et où que tu ailles, il y aura un langage qui sera adéquat et que tu devrais utiliser, mais si ta langue de base c'est le français, pour t'exprimer, tu serais toujours plus à l'aise en français. Si tu te retrouves dans un pays où on ne parle pas français, tu devrais bien sûr t'adapter et parler la langue locale, mais en vérité tu serais soulagé d'y rencontrer des francophones et de parler français avec eux, vu que les mots te viennent plus naturellement.

La différence est que tu ne choisis pas ton langage humain de base. C'est généralement la langue que les gens parlent là où tu nais et grandit, le langage que tes parents parlent, le langage que tes amis et tes enseignants parlent dans ton école, etc.

Pour les langages de programmation, c'est différent. Les programmeurs peuvent faire le choix d'apprendre un langage en particulier même si leurs motivations peuvent être différentes. La problématique que je veux relever ici, c'est pourquoi et comment les programmeurs choisissent leurs langages de programmation ? Quelle est la meilleure façon de s'y prendre ?

D'après ce que j'ai observé, il y a 2 raisons principales qui poussent les gens à choisir un langage spécifique:

- Suivre la tendance: je suppose que ce sont surtout les débutants qui le font. Quand tu veux apprendre à programmer au début, tu ne sais pas grand-chose sur les différents langages mais il te faut bien commencer quelque part. Donc souvent, tu commences par le langage qui est populaire dans ton école, parmi tes profs/mentors, ou juste parmi tes amis. Bref, tu choisis les langages auquel tu es le plus exposé ou tu suis juste les conseils de tes proches. L'avantage c'est que tu aurais forcément autour de toi des gens qui se spécialisent dans les mêmes technologies, donc ils seront là pour te guider, te soutenir et vous pourriez travailler sur les mêmes projets. C'est un peu comme choisir d'acheter un jeu vidéo parce que tous tes amis ont le même jeu. Si tu es bloqué à un niveau dans le jeu, il y a probablement un de tes amis qui sait ce qu'il faut faire. En plus, vous pouvez échanger des disques, discuter de l'actualité de jeux, prendre plaisir à jouer ensemble, etc. En somme, tu deviens membre de la communauté et c'est à peu près la même chose quand tu choisis un langage de programmation.

- Exigence du projet: tout au long de leur carrière, les développeurs rencontrent plusieurs projets qu'ils se doivent de terminer. Les exigences de ces projets peuvent les amener à apprendre de nouveaux langages, de nouvelles technologies. Par exemple, si tu travailles pour une compagnie et que tu es affecté sur un nouveau projet en Python, tu aurais besoin d'apprendre Python. D'autant plus que dans certaines SSII, on se retrouve consultant expert dans n'importe quel domaine en quelques heures... Même si tu développes à ton propre compte, tu peux avoir à apprendre un nouveau langage imposé par tes clients ou qui satisfait mieux les besoins de tes utilisateurs (en terme de vitesse ou d'expérience utilisateur par exemple)

Personnellement, je pense que les deux arguments sont valides. Par rapport au premier argument, j'ajouterais juste qu'il ne s'agit pas de suivre la tendance juste pour suivre la tendance. Il faut faire des recherches sur les langages pour savoir quel est celui qui t'arrange le plus. Pour ce qui est de la deuxième raison, je comprends qu'il faille parfois faire des choses qu'on n'aime pas mais je conseillerais d'utiliser les langages et les technologies où l'on se sent le plus à l'aise, tant que c'est possible. Si la programmation doit être ton travail quotidien, c'est mieux de prendre plaisir à le faire. Fais ce que tu aimes, sérieux.

Pour ce qui est de mon expérience personnelle, je suppose que j'ai opté pour C#.NET tout d'abord parce que c'était très populaire dans l'institut ou je me suis sérieusement mis à la programmation (NIIT). Mais ça, c'est juste la raison pour laquelle j'ai commencé C#. Il y a plusieurs autres raisons qui font que je continue de l'utiliser. La raison principale est que je trouve que la programmation sous C# est "confortable", mais au début ce n'était pas parce que c'était le langage que j'utilisais le plus souvent. D'ailleurs C# n'était pas mon premier langage de programmation. J'avais programmé en VB et en Pascal avant ça, donc ce n'est pas comme si je suis resté bloqué sur mon premier langage. Quand je dis "confortable", je veux dire que je prends plaisir à écrire des applications en C#. Je préfèrerais apprécier mes heures de coding plutôt que d'écrire du code dans un langage bizarre que je trouve pénible. C'est vrai qu'il faut se soucier de certains aspects de l'application tels que la performance mais je crois qu'il faut surtout aimer ce qu'on fait.

Mais qu'est ce qui est si cool à propos de C# ? Je ne vais pas comparer C# aux autres langages et affirmer que c'est techniquement mieux que les autres. Ce n'est pas le but de cet article et comme je l'ai dit précédemment, il n'y a pas de langage parfait, il y a juste des langages adéquats pour les circonstances qui se présentent. Ce que je vais faire ici, c'est d'expliquer en quoi C#.NET améliore ma productivité (et probablement la tienne aussi, quand tu voudras bien l'essayer):

Je ne peux pas m'empêcher de mentionner que Visual Studio, l'environnement de travail de C#/.NET est probablement le meilleur IDE au monde. Ces fonctionnalités d'automation telles que l'IntelliSense et le glisser-déposer des contrôles permettent d'économiser beaucoup de temps et d'effort. Ce n'est pas de la paresse, c'est vraiment à propos de productivité. Que tu travailles en solo ou en équipe, tu fais toujours face à d'importantes tâches qui ne sont pas nécessairement liées à la programmation. Les automations t'évitent de perdre du temps sur les tâches évidentes et fréquentes, ce qui te permet de te concentrer sur les choses qui sont vraiment importantes.

Le langage en lui même est facile à apprendre, il suit la même syntaxe que C, C++, Java, etc. Donc quiconque connaît un de ces langages similaires peut s'y retrouver rapidement. Aussi, on dit souvent que C# est simple et élégant.

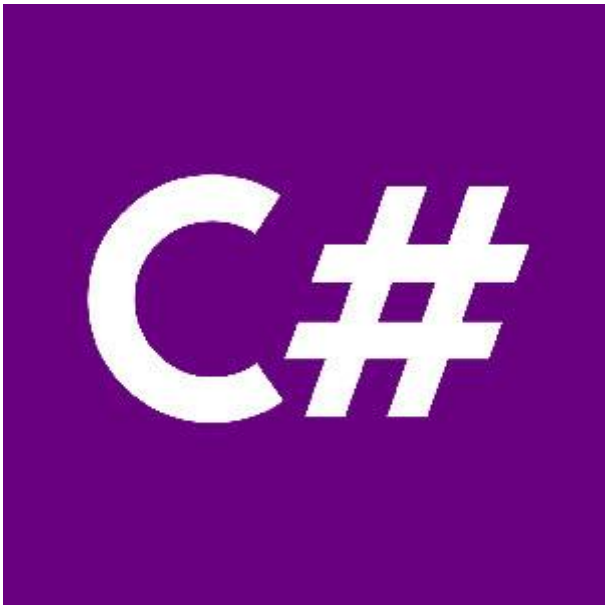
C# est puissant. Il me permet de faire tout ce que je veux, que ce soit un site Web, une application Web, un service Web, une application cliente, un jeu, une application Windows, un service Windows, un service web, une application dans le navigateur (Silverlight), etc. Tandis que la plupart des langages ne peuvent être utilisées que pour un domaine précis, soit le serveur web, soit le navigateur, soit le client, C# peut tout faire. L'avantage évident est que tu n'as pas besoin d'apprendre un nouveau langage quand tu veux commencer un nouveau genre d'application.

Interoperabilité et intégration des langages: les applications et les services C# peuvent aisément parler entre eux. En fait ils peuvent communiquer avec les autres applications .NET. Par exemple, tu peux facilement faire une application C# WPF qui reçoit des données d'un service VB.NET WCF et passer ces données à une application C# Silverlight.

Ok, maintenant tu sais pourquoi j'aime tant C#...

Condensé De Ressources Gratuites En Français Pour Apprendre Et Maitriser C#

Par Necemon



Le C# (Prononcez "cé charpe" à la française ou "ci charpe" à l'anglaise) est le langage de programmation phare de Microsoft, donc un langage très professionnalisant. Utilisé par un nombre important et grandissant de professionnels, il permet de réaliser toutes sortes d'applications.

Une question qui revient souvent est, comment débuter en C#, quels sont les meilleurs cours gratuits pour les francophones ?

Voici quelques suggestions.

Developpez.com
Ce cours est le fruit de plusieurs années d'enseignement en école d'ingénieurs à l'université d'Angers, plus particulièrement dans la formation "Génie des systèmes industriels : automatique et génie informatique". Aussi disponible en format pdf.

Open Classrooms - Apprendre à développer en C#
Ce cours a pour but de t'apprendre les rudiments du langage C# et est ouvert à tous les débutants. D'abord, apprendre comment on crée des applications informatiques et plus particulièrement celles utilisant le Framework .NET; et puis tu pourras te familiariser avec la syntaxe de base du C# pour commencer à créer des applications avec Visual Studio. À la fin de ce cours, tu maîtriseras les bases de la programmation en C# et saura créer une application capable d'interagir avec un utilisateur, de lire ses saisies au clavier et d'afficher des choses à l'écran via une console.

Open Classrooms - Programmer en orienté objet avec C#
Ce cours est la suite du cours "Apprendre à développer en C#", qui présente la syntaxe de base de C# et comment créer des applications simples utilisant le Framework .NET avec Visual Studio.
C'est là que tu peux apprendre les bases de la programmation orientée objet ainsi que la syntaxe C# à utiliser pour créer des classes et manipuler tes objets. Tu découvriras également comment utiliser les différents types du Framework .NET, comment créer des bibliothèques de code réutilisables ou encore comment créer des tests unitaires.
À la fin de ce cours tu sauras quasiment tout ce qu'il faut pour commencer à réaliser des applications d'envergure.

Open Classrooms - Créer sa première application connectée en C#
Découvre comment C# te permet de développer rapidement des applications connectées, entre autres, des applications clientes Windows ainsi que des applications serveurs robustes.

Cours langage C# sur misfu.com
Un panel de documents susceptibles de t'aider à maîtriser la programmation en C Sharp. Certaines formations s'adressent plus particulièrement aux débutants, d'autres sont plus avancées. Ces cours sont accessibles gratuitement et sont téléchargeables au format pdf.

Bonus 1 : quelques exemples de codes et d'applications
Comment ça marche (codes sources)

Bonus 2 : quelques séries de tutoriaux vidéo sur Youtube
Youtube 1
Youtube 2
Youtube 3

Comment Traquer les Bugs Mystérieux avec Visual Studio

Par Necemon

En informatique, un bug (ou bogue) est un défaut de conception (ou d'exécution) d'un programme informatique (ou d'une application, d'un logiciel, etc.), qui est à l'origine d'un dysfonctionnement.

Un débogueur (ou débugueur, de l'anglais debugger) est un logiciel qui aide un développeur à analyser les bogues d'un programme. Pour cela, il permet d'exécuter le programme pas-à-pas, d'afficher la valeur des variables à tout moment, de mettre en place des points d'arrêt sur des conditions ou sur des lignes du programme. Il s'agit de l'application à la programmation informatique du processus de troubleshooting, c'est à dire, un processus de dépannage, de recherche logique et systématique de résolution de problèmes.

Certains bugs sont faciles à traquer. Je ne vais pas m'attarder sur ceux-là. Parlons plutôt de ceux qui sont vraiment mystérieux.

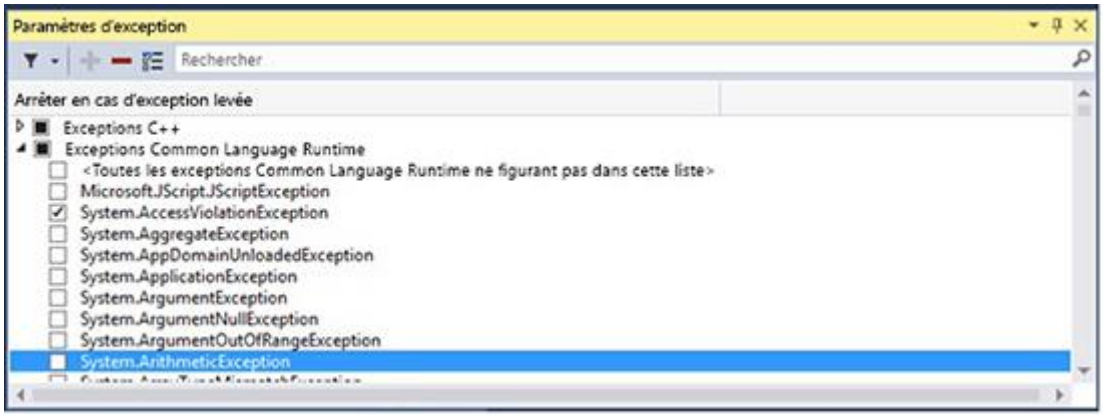
Un vieil adage disait que la Théorie, c'est quand on comprend tout, mais rien ne fonctionne ; la Pratique c'est quand on ne comprend rien, mais tout fonctionne. Mais parfois la Théorie et la Pratique se rencontrent : rien ne fonctionne et personne ne sait pourquoi. Dans ces cas là, il est bon de se rappeler de certaines techniques efficaces qui permettent bien souvent de toucher le fond du problème, et d'en afficher les détails. Au fil du temps, voici les 3 tactiques qui se sont montrées particulièrement utiles quand il s'agit de déboguer avec Visual Studio:

1. Gestion des exceptions pertinentes avec la fenêtre "Paramètres d'exception"

Une exception est une indication d'un état d'erreur qui se produit pendant qu'un programme est en cours d'exécution. Tu peux (et tu dois) fournir des gestionnaires qui répondent aux exceptions les plus importantes, mais il est important de savoir comment configurer le débogueur pour qu'il s'arrête sur les exceptions que tu veux afficher. Tu peux utiliser la fenêtre Paramètres d'exception pour spécifier les exceptions (ou ensembles d'exceptions) qui provoqueront l'arrêt du débogueur, et à quel point tu veux qu'il s'arrête. Tu peux ajouter ou supprimer des exceptions, ou spécifier les exceptions sur lesquelles effectuer un arrêt. Ouvre cette fenêtre lorsqu'une solution est ouverte en cliquant sur **Déboguer/Fenêtres/Paramètres d'exception**.

Tu peux trouver des exceptions spécifiques à l'aide de la fenêtre Rechercher de la barre d'outils Paramètres d'exception ou en utilisant la fonction de recherche pour filtrer des espaces de noms spécifiques (par exemple, System.IO). Le débogueur peut interrompre l'exécution à l'endroit où une exception est levée, ce qui vous permet d'examiner l'exception avant qu'un gestionnaire soit appelé.

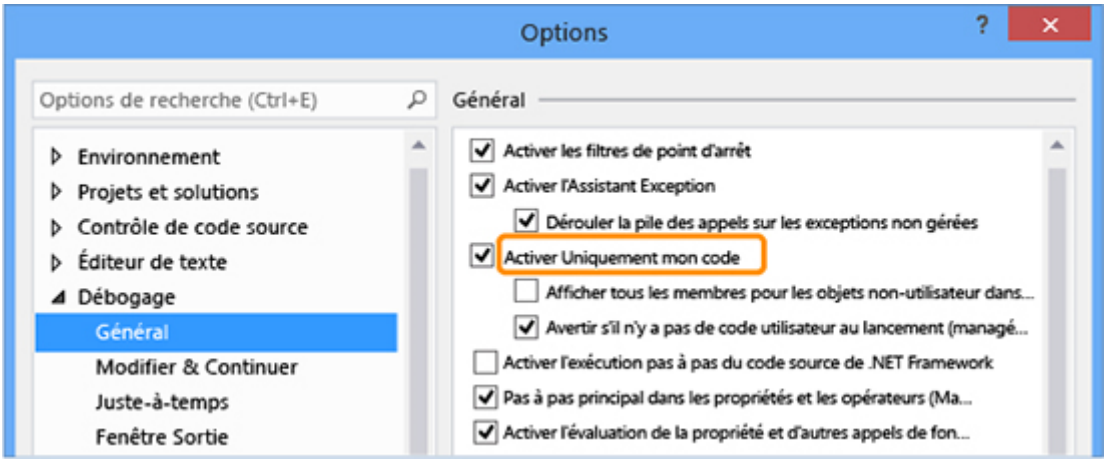
Dans la fenêtre Paramètres d'exception, développe le noeud d'une catégorie d'exceptions (par exemple, Exceptions Common Language Runtime, c'est-à-dire les exceptions .NET), puis coche la case correspondant à une exception spécifique de cette catégorie (par exemple, System.AccessViolationException). Tu peux également sélectionner une catégorie entière d'exceptions.



2. Pas "Uniquement Mon Code"

Par défaut, le débogueur de Visual Studio ne s'arrête que sur les exceptions générées par ton propre code d'utilisateur, survolant ainsi les appels externes et non-utilisateurs, par exemple les appels de systèmes et d'infrastructure. La fonctionnalité qui active ou désactive ce comportement est appelée "Uniquement Mon Code". Selon ce que tu débogues, tu pourrais penser à la désactiver, puisque la source ou la description du problème pourrait bien être située hors de "ton" code.

Pour désactiver (ou activer) Uniquement Mon Code, choisis le menu **Outils > Options** dans Visual Studio. Avec la commande **débogage > général**, annule (ou sélectionne) **Activer Uniquement Mon Code**.



3. Outils recommandés pour le traçage et la journalisation des erreurs

Parfois, il te faut enregistrer et analyser les détails complets des erreurs, des évènements et les séries d'exceptions internes : ce traçage implique un usage spécialisé de journalisation, typiquement pour des raisons de débogage. C'est l'historique, l'enregistrement séquentiel dans un fichier ou une base de données de tous les événements affectant un processus particulier.

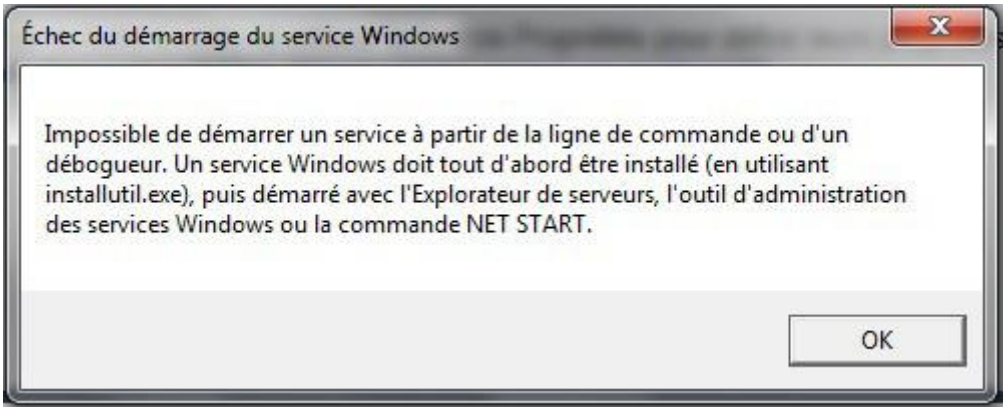
Voici mes outils de journalisation et de traçage préférés:
- Systems.Diagnostics
- Microsoft Entreprise Library
- NLog
- Elmah
- Log4net

Voilà.

Comment Déboguer un Service Windows sous Visual Studio

By Holty Sow

Lorsque vous créez un projet de type **Service Windows** sous Visual Studio, vous avez remarqué que la boîte de dialogue ci-dessous s'affiche quand on essaie d'exécuter le service :



En résumé il est tout simplement impossible d'exécuter un service Windows sous Visual Studio, on doit obligatoirement passer par la commande **NET START** pour démarrer le service après avoir préalablement installé celui-ci grâce à la commande **INSTALLUTIL**. Sauf que cette méthode nous empêche de faire facilement un débogage du service Windows. En effet il faut installer le service, le démarrer, puis dans Visual Studio attacher un processus (qui sera bien sûr le processus du service Windows) pour pouvoir déboguer notre code. Sans oublier qu'il va falloir aussi arrêter le service Windows pour pouvoir compiler suite à quelques modifications qu'on aura effectuées dans le code du service. Bref c'est un peu lourd :D.

Il y a plus simple. On utilisera les symboles de compilation pour détecter dans quel mode nous sommes : **RELEASE** ou **DEBUG**. Si nous sommes en :
- DEBUG : nous allons traiter notre service Windows comme une simple application Windows Forms en affichant une boîte de dialogue indiquant que le service est démarré
- RELEASE : le fonctionnement sera le même que quand nous avons essayé d'exécuter notre service Windows sous Visual Studio. En d'autres termes, ce mode doit être utilisé en production une fois le débogage terminé

Pour notre cas d'utilisation ce sera très simple. Il s'agira ici de rendre possible l'exécution du service Windows et de déboguer un service WCF qu'il héberge. Voici les étapes à suivre :

1. Modification du code du service Windows : nous allons ajouter deux méthodes **StartWCFService** et **StopWCFService** qui ont pour tâches respectives de démarrer et d'arrêter l'écoute des requêtes entrantes WCF. La première méthode sera appelée dans la redéfinition de la méthode **OnStart** et la seconde dans celle de la méthode **OnStop**. Ci-dessous le code :

```
private ServiceHost host;

public MyWindowsService()
{
   InitializeComponent();
}
```

```
protected override void OnStart(string[] args)
{
   StartWCFService();
}

protected override void OnStop()
{
   StopWCFService();
}

public void StartWCFService()
{
   host = new ServiceHost(typeof(IWCFService));
   host.Open();
}

public void StopWCFService()
{
   if (host != null && host.State == CommunicationState.Opened)
   host.Close();
}
```

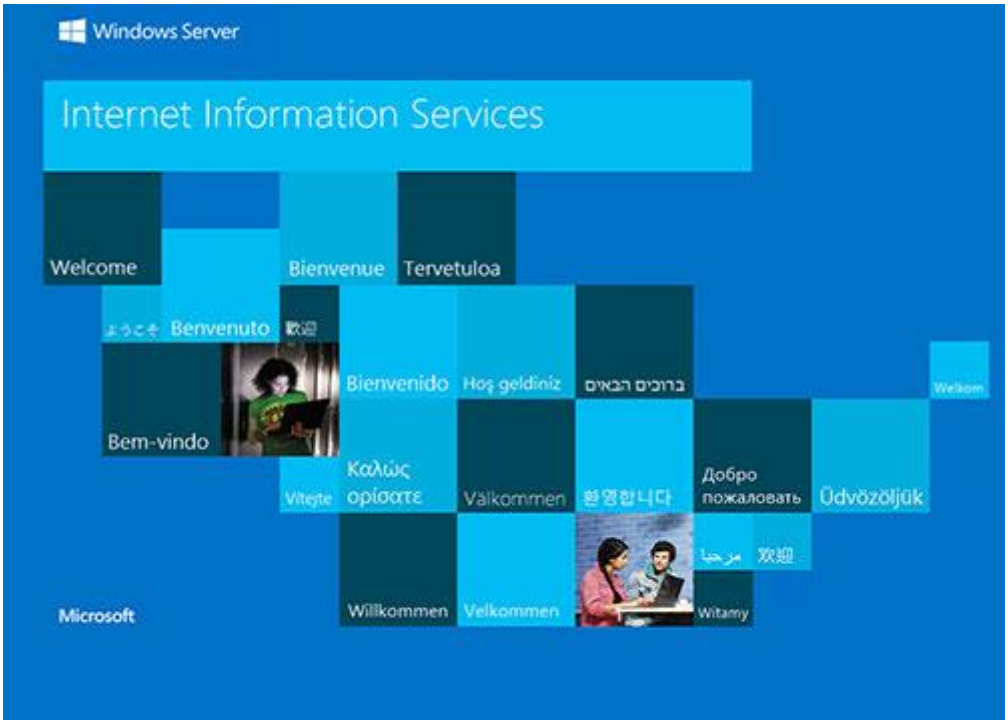
2. Modification du fichier **Program.cs** : c'est dans la méthode **Main** que nous allons faire la détection du mode de compilation dans lequel nous sommes. Si nous sommes en mode **RELEASE**, alors le service Windows s'exécute comme d'habitude et forcément on aura la fameuse boîte de dialogue si on essaie de l'exécuter sous Visual Studio avec ce mode. Si par contre, nous sommes en mode **DEBUG**, alors c'est très simple on fait appel directement à la méthode **StartWCFService** de l'instance de notre service Windows (donc la méthode **OnStart** ne sera pas appelée) puis on affiche une boîte de dialogue pour en informer l'utilisateur. si ce dernier ferme la boîte de dialogue alors la méthode **StopWCFService** est appelée pour arrêter le service WCF (donc la méthode **OnStop** du service Windows ne sera pas appelée).
Ci-dessous le code de la méthode **Main** :

```
static void Main()
{
   MyWindowsService service = new MyWindowsService();
   #if DEBUG
      service.StartWCFService();
      MessageBox.Show("Le service a démarré...");
      service.StopWCFService();
   #else
      ServiceBase[] ServicesToRun;
      ServicesToRun = new ServiceBase[]
      {
         service
      };
      ServiceBase.Run(ServicesToRun);
   #endif
}
```

J'espère que ce billet vous a été utile :-)

Déployer une application web sur IIS / Windows Server en 7 étapes ingénieuses

Par Necemon



Une routine simple, pour un déploiement robuste et efficace :-)

1. Créer/configurer le site web et son pool d'application

2. Copier les fichiers vers le serveur : installer une connexion FTP(S), ainsi que les accès clients correspondant (FileZilla, Visual Studio, etc.)

3. Attribuer les permissions d'écriture à l'application sur les dossiers appropriés (pour le journal d'erreurs, la sauvegarde des fichiers d'utilisateurs, etc.)

4. Fixer les délais de temps mort
Delai d'inactivité: Tu peux changer la valeur par défaut de 20 au nombre de minutes que tu veux. Tu peux également régler le paramètre à 0 (zéro), ce qui désactive effectivement le délai tel que le pool d'applications ne va jamais s'arrêter à cause de son inactivité. Ceci peut être configuré dans les Paramètres Avancés du pool d'applications.
Delai d'expiration de session: spécifie le temps (en secondes) que IIS attend avant d'interrompre une connexion qui est considérée comme inactive. Ceci peut être configuré dans les Paramètres avancés des Outils d'administration (system.applicationHost / weblimits).

5. Configurer le paramètre Auto-Start
Un problème commun est la nécessité d'effectuer des tâches d'initialisation et l'"échauffement" des tâches pour une application Web. Les applications Web les plus larges et les plus complexes peuvent avoir besoin d'effectuer de longues procédures de démarrage, de mise en mémoire cache, de création de contenu, etc. avant de servir la première requête HTTP. Une façon de résoudre ce problème est d'ajuster quelques propriétés dans le module d'initialisation d'applications :
- Mets la propriété StartMode du pool d'applications à AlwaysRunning
- Mets la propriété PreloadEnabled à True et précise le pool d'application

6. Configurer SQL Server / les sauvegardes automatiques
Créer une procédure de planification de sauvegarde avec l'Assistant Plan de Maintenance

7. Installation de Certifcat HTTPS / SSL
HTTPS améliore la sécurité, l'identification, le SEO, l'accès à certaines fonctionalités avancées de HTML5 et bien d'autres choses.

Lancer son application au Démarrage de Windows sans Bidouiller avec la Base de Registre

Par Holty Sow

Dans ce billet je vais vous présenter deux méthodes permettant de configurer son application .Net pour qu'elle soit lancée au démarrage de Windows. Ces deux méthodes nous évitent de toucher à la base de registre et ainsi nous n'allons pas oublier de nettoyer cette base s'il arrivait que l'utilisateur désinstalle notre application.

La première méthode :

Dans le projet d'installation (projet de déploiement avec Windows Installer) suivre les étapes suivantes :

Dans le **File System (Système de fichiers)** du projet on ajoute un dossier spécial appelé **User's Startup** Folder (Dossier de démarrage de l'utilisateur)

On crée un raccourci de la sortie de projet qui se trouve dans le dossier Application Folder (Dossier de l'application). On renomme ce raccourci pour lui donner un nom plus explicite.

On coupe (CRTL+X) le raccourci qu'on vient de créer et on le colle (CRTL+V) dans le dossier User's Startup Folder.

Deuxième méthode :

Cette solution consiste à rendre configurable le démarrage automatique à partir du code de l'application. Pour cela j'ai créé deux fonctions : une pour l'activation et l'autre pour la désactivation.

N.B. : Pour que le code fonctionne, il faudra ajouter la référence à l'assembly COM Windows Script Host Object Model dans votre projet.

La fonction d'activation est la suivante :

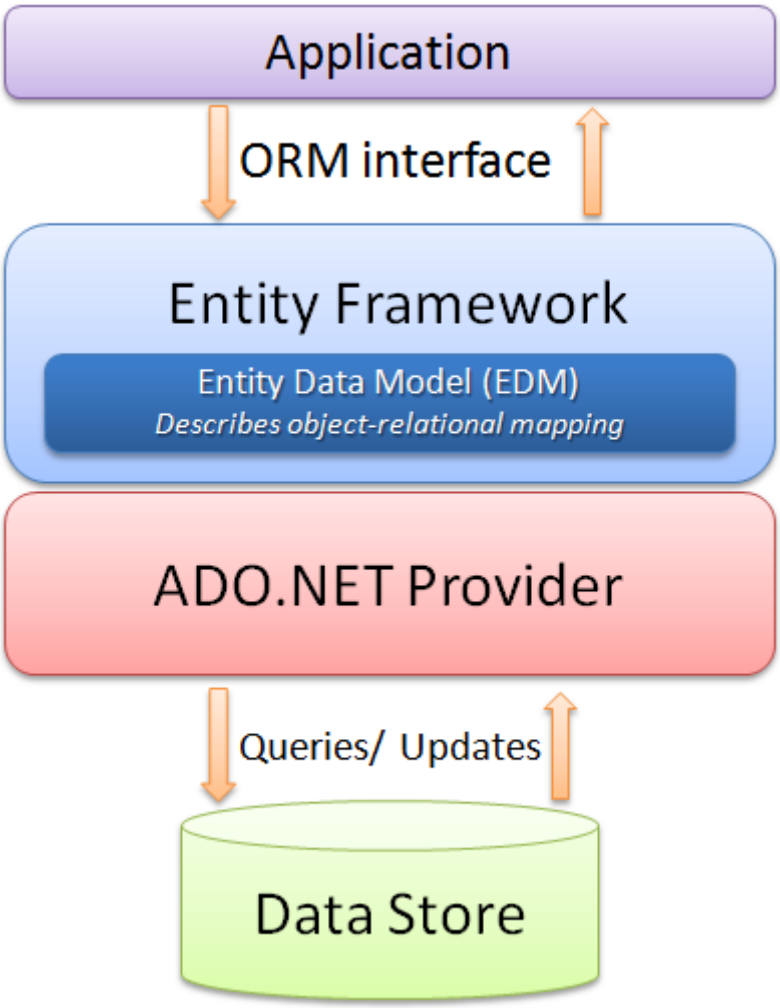
```
public void EnableApplicationStartup()
{
    string shortcutPath =
System.IO.Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.Startup),
 "MyShortcut.lnk");
    if (System.IO.File.Exists(shortcutPath)) return;
    WshShell wshShell = new WshShellClass();
    IWshShortcut shortcut = (IWshShortcut)wshShell.CreateShortcut(shortcutPath);
    shortcut.TargetPath = Assembly.GetEntryAssembly().Location;
    shortcut.Description = "Mon premier raccourci";
    shortcut.Save();
}
```

La fonction de désactivation :

```
public void DisableApplicationStartup()
{
    string shortcutPath =
System.IO.Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.Startup),
"MonRaccourci.lnk");
    if (!System.IO.File.Exists(shortcutPath)) return;
    System.IO.File.Delete(shortcutPath);
}
```

Mes 12 Astuces Préférées Avec Entity Framework

Par Necemon



Entity Framework est un mappeur objet/relationnel qui permet aux développeurs .NET d'utiliser des données relationnelles à l'aide d'objets spécifiques au domaine. Il rend inutile la plupart du code d'accès aux données que les développeurs doivent généralement écrire. Je joue avec ça depuis deux ou trois ans, et voici mes techniques préférées:

1. Prolongement de la valeur TimeOut

Lors du chargement de grandes quantités de données, il peut arriver que certaines opérations échouent chaque fois qu'elles atteignent le temps mort par défaut, du coup ça pourrait être une bonne idée d'étendre cette durée à quelques heures. C'est bien de le faire pendant l'initialisation de l'objet Contexte, donc avant tout appel de base de données.

2. Cache de deuxième niveau
C'est la mise en cache des résultats de la requête. Les résultats de commandes SQL sont stockés dans le cache, de sorte que les commandes répétées récupèrent leurs données à partir du cache au lieu d'exécuter la requête à nouveau contre la base de données. Ceci peut présenter un gain de performances pour l'application et moins d'activités au niveau de la base de données. Pour activer cette fonctionnalité, tu peux télécharger et utiliser ce projet open source.

3. Code First dynamique et migration de bases de données
EF permet de programmer contre un modèle sans avoir à toucher la base de données. Avec la technique Code-First, tu peux te concentrer sur la conception des modèles et commencer à créer des classes. Les APIs de Code-First vont créer et/ou mettre à jour la base de données à la volée en fonction de tes classes d'entités et de ta configuration.
D'ailleurs, pendant qu'on parle de configuration, tu peux faire tout ça depuis ton code C#. Voici quelques propriétés pratiques:
AutomaticMigrationEnabled : pour activer la magie de Code-First
AutomaticMigrationDataLossAllowed : Une valeur indiquant si la perte de données est acceptable lors de la migration automatique. Si la valeur est "false", une exception sera levée quand la migration automatique est susceptible de créer une perte de données.

4. Aperçu et anticipation des changements de Code-First
Ceci est un moyen facile de générer les scripts SQL que EF compte exécuter, sans confirmer ou avant de confirmer ses changements. Voici comment ça marche:

```
var configuration = new MigrationConfiguration();
var migrator = new DbMigrator(configuration);
var scriptor = new MigratorScriptingDecorator(migrator);
string script = scriptor.ScriptUpdate(null, null);
```

Ça peut être utile si tu veux faire des changements au script qui va être exécuter, ou si tu es en train de déboguer un problème ou simplement si tu es curieux de savoir ce qui se passe :-)

5. MARS
Multiple Active Result Sets (MARS) est une fonctionnalité qui permet l'exécution de plusieurs paquets sur une seule connexion. Pour le dire d'une manière simple, tu peux établir une connexion au serveur, puis soumettre plusieurs commandes au serveur en même temps, puis lire les résultats de ces demandes de la manière que tu veux. Il suffit d'inclure "MultipleActiveResultSets = true" dans ta ConnectionString.

6. Réglage de toutes les propriétés en un seul endroit
Par exemple, au lieu d'ajouter un attribut à chaque propriété String, tu peux définir une longueur maximale par défaut comme suit:

```
protected override void OnModelCreating(DbModelBuilder modelBuilder)
{
   modelBuilder.Properties<String>().Configure(p => p.HasMaxLength(360));
}
```

7. Stratégie d'héritage

EF a plusieurs façons de gérer l'héritage

-Table par Hiérarchie : une seule table avec toutes les propriétés des classes de base et des classes dérivées (Une propriété discriminatrice étant utilisée pour les différencier)

-Table par Type : les propriétés de base dans la table de base, et pour chaque classe dérivée, les propriétés dérivées dans une table séparée

-Table Par type concret : chaque classe dérivée obtient sa propre table avec toutes les propriétés (base ou dérivées).

C'est assez complet, le seul cas où je ne suis pas sûr: y a-t-il un moyen de faire en sorte que EF évite complètement l'héritage ? Je veux dire, en supposant que A dérive de B, est-il possible de faire en sorte que EF traite A et B comme des classes complètement différentes et ignorer le fait que l'une hérite de l'autre ? Pas un scénario très commun, mais juste pour éviter la duplication, ce serait alors possible d'utiliser l'héritage dans le code C# sans copier les propriétés deux fois dans des classes différentes, sans réaction de la part EF. TPC se rapproche de cela, mais les classes partagent toujours la même clé primaire.

8. Les Méthodes EntityFunctions

Lorsqu'on utilise LINQ to Entity Framework, tes prédicats dans la proposition Where sont traduits en SQL, mais SQL n'a pas d'équivalent pour certaines constructions complexes telles que DateTime.AddDays(). C'est là que les méthodes EntityFunctions entrent en jeu.

9. LINQKit

LINQKit est un ensemble gratuit d'extensions pour LINQ to SQL et Entity Framework. Je l'utilise surtout pour construire dynamiquement des prédicats et insérer des variables d'expression dans les sous-requêtes, mais il permet aussi de:

- Combiner des expressions (permettre à une expression d'en appeler une autre)
- Insérer des expressions dans les EntitySets et EntityCollections
- Créer tes propres extensions

10. Lazy Loading

L'une des fonctionnalités les plus intéressantes de Entity Framework, c'est la fonction Lazy Loading. C'est le processus par lequel une entité ou une collection d'entités sont automatiquement chargés à partir de la base de données la première fois qu'une propriété faisant référence à l'entité ou aux entités est accessible. Pour le dire plus simplement, Lazy Loading signifie: retarder le chargement des données, jusqu'à ce que tu les appelles explicitement.

Ceci pourrait être activé en activant la propriété Configuration.LazyLoadingEnabled.

11. AsNoTracking

Entity Framework expose un certain nombre d'options d'optimisation des performances pour t'aider à peaufiner tes applications. L'une de ces options de réglage est .AsNoTracking(). Cette optimisation te permet de dire à Entity Framework de ne pas traquer les résultats d'une requête. Cela signifie que Entity Framework n'effectue aucun traitement supplémentaire ou stockage des entités qui sont retournés par la requête.

Il y a des gains de performance considérables découlant de l'utilisation de AsNoTracking().

12. Rester loin de pièges de performance et autres complications de EF

Le gestionnaire de contexte d'objets peut parfois mener à des situations où EF se comporte de manière étrange. C'est bien d'être informé des méthodes que tu peux suivre pour éviter ces pièges.

ASP.NET MVC: Eviter de Polluer le Modèle de la Vue avec des Messages d'Erreurs

By Holty Sow

J'ai récemment eu à répondre à une question posée sur le site StackOverflow. Je pose le contexte. Nous avons un modèle suivant:

```
public class ForgotPasswordMV
{
   [Display(Name = "Enter your email"), Required]
   public string Email { get; set; }
}
```

Ce modèle est utilisé dans l'action d'un contrôleur comme suit :

```
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Search(ForgotPasswordMV viewModel)
{
   if(Temp.Check(viewModel.Email))
      return RedirectToAction("VerifyToken", new { query = viewModel.Email });
   else
   {
      ViewBag.ErrorMessage = "Email not found or matched";
      return View();
   }
}
```

La question était de savoir si le fait d'utiliser la propriété dynamique **ViewBag** du contrôleur pour exposer le message d'erreur était une bonne pratique et que les recherches effectuées par le questionneur lui ont fait savoir qu'il fallait exposer une propriété au niveau du modèle.
Evidemment il est fortement recommandé de ne pas utiliser la propriété **ViewBag** étant donné qu'on ne bénéficie pas du typage fort. Si on veut communiquer avec la vue il faut toujours passer par un modèle typé. La solution proposée est donc légitime mais n'est pas une bonne pratique dans le cas de la gestion des erreurs du modèle dans le framework ASP.Net MVC.
La solution pour exposer les messages d'erreurs (comme dans l'exemple précédent qui n'utilise pas les attributs d'annotations de données) vient par l'utilisation de la méthode **AddModelError**de la classe **ModelStateDictionary**. Nous n'avons pas besoin d'instancier cette classe étant donné qu'une propriété **ModelState** contenant une instance de cette classe existe déjà au niveau du contrôleur.
Du coup la bonne solution est de faire comme suit:

```
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Search(ForgotPasswordMV viewModel)
{
   if
   {
      // ...
   }
   else
   {
      this.ModelState.AddModelError("Email", "Email not found or matched");
      return View(viewModel);
   }
}
```

Il faut noter que cette méthode reçoit en premier paramètre le nom de la propriété du modèle à laquelle le message d'erreur est associé. Ainsi pour que ce message d'erreur soit affiché dans la vue à côté du champ **Email** il faut ajouter juste à côté de ce dernier la ligne suivante :

```
@Html.ValidationMessageFor(m => m.Email)
```

Cependant il est possible d'avoir un message d'erreur général à l'ensemble du modèle c'est à dire que ce message d'erreur n'est rattaché à aucune propriété du modèle. Pour cela il faudra utiliser une chaîne vide comme premier paramètre :

```
@Html.ValidationSummary(string.Empty, ""Email not found or matched")
```

Dans la vue Razor, il faudra utiliser la ligne suivante :

```
@Html.ValidationSummary(true, "The following error has occured:")
```

Le premier paramètre booléen indique qu'on ne veut pas afficher les messages d'erreur déjà rattachés à des propriétés du modèles.

Dans ce billet nous avons vu qu'on n'a pas besoin de polluer notre modèle et d'exposer des propriétés spécifiques aux messages d'erreur. Il suffit juste d'utiliser ce que nous offre le Framework ASP.Net MVC pour nous faciliter la tâche.

J'espère que ce billet vous a été utile.

Limiter l'exécution d'une action uniquement aux requêtes AJAX

Par Holty Sow

En ASP.Net MVC nous manipulons entre autres des vues. Parmi ces vues certaines représentent des pages complètes et d'autres ne sont que des parties de page. Ces parties ou zones appartenant à une vue sont appelées des vues partielles et elles sont renvoyées elles aussi par des actions du contrôleur. Ces vues partielles ne devant être utilisées qu'à l'intérieur d'une vue alors le Framework ASP.Net MVC nous permet de protéger l'appel à ces actions partielles en les décorant avec l'attribut **ChildActionOnly**. Cet attribut permet d'être sûr que l'action :
- ne pourra pas être utilisée comme une vue entière et que les développeurs de l'application l'exécuteront toujours en passant par les méthodes **HtmHelper.Action** ou **HtmlHelper.RenderAction.**
- a une URL qui ne sera pas accessible via la barre d'adresse si un utilisateur, je ne sais par quel moyen, serait au courant de l'existence de cette URL.

Cependant comme tout site dynamique nous aurons des requêtes AJAX qui pourront faire elles aussi des demandes de contenu HTML sans qu'on ait besoin de charger la page de façon complète. Ce contenu aussi représente une partie puisqu'à la réception de la réponse du serveur Web nous devons incorporer ce bout de HTML quelque part dans la page. La requête AJAX envoyée au serveur invoquera certainement une action du contrôleur. Cette action comme celles marquées avec l'attribut **ChildActionOnly** doit posséder des restrictions, soit :
- obligation de passer par une requête faite en AJAX.
- inaccessible via la barre d'adresse du navigateur.

Cependant le Framework ASP.Net MVC n'offre aucun attribut nous permettant d'appliquer ces restrictions sur une action mais il nous fournit les outils pour en créer. Pour cela nous devons coder un filtre qui sera exécuté juste avant l'exécution de l'action concernée. Si la requête entrante respecte les conditions d'une requête faite en AJAX alors on laisse l'action continuer son chemin. Dans le cas où les conditions ne seraient pas respectées nous envoyons une page 404 (comme quoi l'URL est inexistante).

Le code du nouveau filtre que je nommerai **AjaxOnlyAttribute,** classe dérivée de la classe **ActionFilterAttribute,** est le suivant :

```
[AttributeUsage(AttributeTargets.Class | AttributeTargets.Method, AllowMultiple = false)]
public class AjaxOnlyAttribute : ActionFilterAttribute
{
   public override void OnActionExecuting(ActionExecutingContext filterContext)
   {
      if (filterContext.HttpContext.Request.IsAjaxRequest())
      {
         base.OnActionExecuting(filterContext);
      }
      else
      {
        filterContext.HttpContext.Response.StatusCode = 404;
        filterContext.Result = new HttpNotFoundResult();
      }
   }
}
```

La nouvelle classe ne fait que surcharger la méthode qui nous intéresse. C'est à dire la méthode **OnActionExecuting** qui est appelée juste avant le début de l'exécution de l'action demandée par la requête entrante. L'attribut peut être posé sur le contrôleur pour atteindre l'ensemble des actions ou unitaire sur chaque action où la requête AJAX est obligatoire.

Pour éviter des soucis de pertes de quelques petites minutes qui peuvent arriver dans le cas où l'un des développeurs ayant rejoint l'équipe en cours de route et qui ne comprendrait pas pourquoi sa requête renvoie du **404** alors je pense que ce serait un avantage d'avoir un mode **DEBUG** activé par défaut pour les codeurs. Le code de notre classe ressemblerait finalement à celui-ci :

```
[AttributeUsage(AttributeTargets.Class | AttributeTargets.Method, AllowMultiple = false)]
public class AjaxOnlyAttribute : ActionFilterAttribute
{
   public override void OnActionExecuting(ActionExecutingContext filterContext)
   {
      if (filterContext.HttpContext.Request.IsAjaxRequest())
      {
         base.OnActionExecuting(filterContext);
      }
      else
      {
         #if DEBUG
         filterContext.Result = new ViewResult { ViewName = "AjaxOnly" };
         #else
         filterContext.HttpContext.Response.StatusCode = 404;
         filterContext.Result = new HttpNotFoundResult();
         #endif
      }
   }
}
```

Etant donné que les développeurs sont amenés à compiler l'application la plupart du temps en mode **DEBUG** du coup la vue **AjaxOnly** contenant le texte qui va bien leur expliquera ce qu'il faut faire. Il faut noter que cette version de la classe fonctionne uniquement si une vue **AjaxOnly.cshtml** a été ajoutée dans le répertoire **Views\Shared** de l'application. Aussi on n'est pas obligé de renvoyer une vue partielle. On peut tout simplement remplacer le code suivant :

```
filterContext.Result = new ViewResult { ViewName = "AjaxOnly" };
```

par juste du contenu textuel :

```
filterContext.Result = new ContentResult
{
   Content = $"This action '{filterContext.HttpContext.Request.RawUrl}' was designed for AJAX requests only"
};
```

La version utilisant la vue **AjaxOnly** a l'avantage de prendre en compte automatiquement le rendu **_Layout** par défaut défini au niveau de l'application.

[Interview] Dans La Bulle de Holty Sow : "pour plus d'efficacité, je préfère travailler avec une équipe agile"

1. Présentation rapide (mais efficace) du personnage et de ses réalisations ?

Je m'appelle Holty Samba SOW. Après avoir débuté mes études supérieures en Maths-Physique puis effectué une licence professionnelle en Informatique au Sénégal, je les ai poursuivies en France pour obtenir un Master en Documents Electroniques et Flux d'informations en 2009. Ce qui m'a permis par la suite de me lancer dans le monde professionnel de la programmation informatique. Actuellement je suis consultant et développeur principalement sur la plateforme .NET chez SoftFluent, un éditeur de logiciels mais aussi une société de services.

2. Quels sont tes objectifs principaux dans la vie ?

Mon principal objectif est pour le moment de garder aussi longtemps que possible ma passion dans la programmation informatique en accumulant un maximum d'expériences dans le monde du Web et du Mobile. J'espère grâce à la richesse de cette expérience en faire bénéficier mon pays natal le Sénégal.

3. Quels outils et techniques utilises tu pour accomplir des choses efficacement ?

Pour les IDE préférés : VS Code pour tout ce qui touche au CSS, JavaScript et TypeScript. Pour tout le reste j'utilise l'IDE Visual Studio accompagné de l'extension ReSharper pour plus d'efficacité.
Mes technologies préférées sont principalement tout ce qui touche au Back-end sur la plateforme .NET donc ASP.NET MVC, ASP.NET Web API et récemment ASP.NET Core.
La méthodologie avec laquelle je préfère travailler en équipe est Scrum/Agile.

4. Des recommandations pour tes juniors ?

Un livre : "C# In a Nutshell" pour comprendre le langage de programmation C# de façon approfondie.

Twitter et Blogs : suivre, via Twitter ou à travers leurs blogs, les influenceurs (speakers, CTO, etc.) évoluant dans le monde informatique permet d'être au courant de pas mal de choses.

Sites Web :
docs.microsoft.com (je trouve que le nouveau site web de documentation de Microsoft est assez bien développé)
stackoverflow.com (je trouve qu'essayer de répondre aux questions posées par d'autres développeurs permet parfois de découvrir des trucs et astuces auxquels on n'aurait pas pensé)
pluralsight.com (pas mal de vidéos sur la programmation vraiment très intéressant. Seul hic : c'est payant)
channel9.msdn.com (idem que Pluralsight mais c'est gratuit et les speakers sont des employés de Microsoft).

5. Quel est le meilleur moyen de te contacter ?

Email: holty.sow@gmail.com
Blog : hssow.wordpress.com
LinkedIn : linkedin.com/in/holty-samba-sow-43042715
Twitter: twitter.com/CodeNotFound
Facebook : facebook.com/CodeNotFound

La Révolution Xamarin

Par Necemon

Xamarin, c'est quoi ?
Xamarin est une technologie qui te permet de concevoir des applications natives pour différentes plateformes mobiles telles que Android, iOS ou encore Windows Phone et ce, en n'utilisant qu'un seul langage de programmation, le C#.
Il ne vous sera donc pas nécessaire d'avoir les bases en Java, utilisé habituellement pour développer sous Android ou encore en Objective C pour iOS, en revanche il vous sera très utile de connaître globalement le fonctionnement de chacune des plateformes visées (cycle de vie de l'application, directives générales, etc.)

Ici, Xamarin prend le contre-pied des autres technologies multiplateforme. Le développeur commence par créer une base de code commune. Elle contient notamment la logique métier, le stockage en base de données, les appels réseaux, les éléments d'interface communs. Ce projet peut être facilement encadré par des tests unitaires car son code est indépendant de tout système spécifique. Ensuite, un projet est crée par plateforme cible. Il contient l'interface graphique, la navigation et les composants propres à chaque SDK. Ainsi, on peut tirer parti des spécificités propres à Android ou iOS sans réduire l'expérience utilisateur au plus petit commun dénominateur.

Avec l'avènement des technologies mobiles et des smartphones, Xamarin devient de plus en plus populaire, surtout depuis sa mise à disposition gratuite dans Visual Studio. Voici quelques docs que je peux recommander pour ceux qui pensent à s'y mettre :

Ce e-book gratuit et complet en Français

Xamarin sur MSDN

Channel9

Plus de ressources en anglais

8 instruments efficaces pour programmeurs Xamarin

Par Necemon

Si tu ne sais pas ce qu'est Xamarin, puis-je te suggérer de commencer par lire mon article d'introduction avant de continuer avec celui-ci ?

Pour ceux qui savent, entrons dans le vif du sujet :

Xamarin Forms

Xamarin.Forms est une librairie de code qui permet de construire des interfaces graphiques natives qui peuvent être partagées sur Android, iOS et Windows Phone, complètement en C#, à partir d'une base de code C# unique et partagée. Les interfaces graphiques sont affichées à l'aide des contrôles natifs de la plate-forme cible, ce qui permet aux applications Xamarin.Forms de conserver l'aspect et la sensation appropriés pour chaque plate-forme. Cela signifie que les applications peuvent partager une grande partie de leur code d'interface utilisateur et conservent toujours l'apparence native de la plate-forme cible. Xamarin.Forms permet un prototypage rapide des applications qui peuvent évoluer au fil du temps vers des applications complexes. Étant donné que les applications Xamarin.Forms sont des applications natives, il est possible de créer des applications qui auront certaines parties de leur interface utilisateur créées avec Xamarin.Forms tandis que d'autres parties seraient créées à l'aide d'outils UI natifs.

Les Custom Renderers

Comme on le voyait précédemment, les interfaces utilisateur Xamarin.Forms sont rendues à l'aide des contrôles natifs de la plate-forme cible, ce qui permet aux applications Xamarin.Forms de conserver l'apparence appropriée pour chaque plate-forme. Les Custom Renderers permettent aux développeurs d'annuler ce processus pour personnaliser l'apparence et le comportement des contrôles Xamarin.Forms sur chaque plate-forme. Ils peuvent être utilisés pour de petits changements de style ou pour une personnalisation sophistiquée de la mise en page et du comportement spécifique à la plate-forme.

SQLite

La plupart des applications ont besoin d'enregistrer des données sur le périphérique, localement. À moins que la quantité de données ne soit trivialement petite, il faut généralement une base de données et une couche de données dans l'application pour gérer l'accès à la base de données. SQLite est un moteur de base de données relationnelle accessible par le langage SQL.
Contrairement aux serveurs de bases de données traditionnels, comme MySQL ou SQL Server, sa particularité est de ne pas reproduire le schéma habituel client-serveur mais d'être directement intégrée aux programmes. Facile d'usage, l'intégralité de la base de données (déclarations, tables, index et données) est stockée dans un fichier indépendant sur la plateforme.
SQLite est le moteur de base de données le plus utilisé au monde, grâce à son utilisation dans de nombreux logiciels grand public comme Firefox, Skype, dans certains produits d'Apple et d'Adobe. De par son extrême légèreté (moins de 300 Ko), il est également très populaire sur la plupart des smartphones modernes.
IOS et Android ont tous deux le moteur de base de données SQLite "intégré" et l'accès aux données est simplifié par la plateforme Xamarin.

Boites de Dialogue ACR

Une bibliothèque multiplate-forme qui te permet d'invoquer les boites de dialogue habituelles à partir d'une bibliothèque partagée et portable: feuilles d'actions, alertes, confirmations, demandes de confirmation, chargement, connexion, progression, etc.

Xlabs

Un projet open source qui vise à fournir un ensemble puissant et multiplate-forme de contrôleurs et d'assistants ajustés pour travailler avec Xamarin Forms: suggestions semi-automatiques, contrôles de Calendrier, boutons-images, étiquettes d'hyperliens, etc.

UI Sleuth

Un outil de débogage et un inspecteur d'interface utilisateur. Si tu as déjà fait un site Web, c'est très similaire aux outils F12 de Microsoft Edge ou aux outils de développement de Google Chrome. Tu peux l'utiliser pour analyser efficacement les problèmes de mise en page, pour prototyper un nouveau design, ou encore contrôler un appareil à distance.

Émulateur Android GenyMotion

Si tu veux tester Android, mais sans un smartphone Android, il y a plusieurs solutions disponibles comme l'émulateur officiel d'Android qui vient avec le SDK (kit de développeur), le populaire Bluestacks App Player qui est plus axé sur l'exécution des applications Android sur PC. Cependant, l'émulateur du SDK est connu pour être très lent tandis que Bluestacks est limité en fonctionnalités si tu veux faire du développement. Si tu cherches un émulateur Android plus rapide et complet pour ordinateur, Genymotion est ce qu'il te faut. Genymotion est un émulateur pour Android, rapide et facile, pour tester le fonctionnement et les performances d'applications. C'est l'émulateur Android le plus puissant pour les testeurs et les développeurs d'applications. Conçu par la start-up française Genymobile, Genymotion permet l'émulation d'un grand nombre de marques et de modèles de tablettes et de smartphones. S'appuyant sur des technologies de virtualisation grâce à VirtualBox, il crée une machine virtuelle qui émule en temps réel Android sur ton PC Windows, Mac ou Linux. il est ainsi possible d'émuler différents terminaux comme la gamme des Samsung Galaxy Note, Google Nexus, HTC One ou encore les tablettes Sony Xperia. GenyMotion offre la possibilité de paramétrer facilement la résolution d'écran, le type de connexion Internet, le GPS, le statut de la batterie (simuler un déchargement), etc.

Simulateur iOS à distance

Pour tester et déboguer les applications iOS dans Visual Studio sous Windows. La plupart des ordinateurs Windows modernes ont des écrans tactiles et le simulateur iOS distant vous permet de toucher la fenêtre du simulateur pour tester les interactions de l'utilisateur dans votre application iOS. Cela inclut les pincements, les glissements et les gestes tactiles avec plusieurs doigts, des choses qui, auparavant, ne pouvaient être testées que sur iPhone, directement.

Chapitre 8

Prototypes Epiques, Projets Classiques, Genre Historique

necemon.com | babifraya.com | evasium.com | kpakpatoya.com

A travers Bavardica - Partie 1 : Discussion des sujets impliqués

Par Necemon, 2010

Un système de messagerie instantanée moderne est composé de différentes caractéristiques telles que les conversations publiques, les conversations privées, les images de profils des utilisateurs et leurs signatures. Quand il est temps d'innover et de faire une messagerie instantanée qui est si spéciale qu'elle est à la fois intéressante et différente des programmes déjà disponibles sur le marché, il n'est pas très évident de déduire quelles fonctionnalités supplémentaires pourraient être jointes. On a littéralement l'impression que tout a déjà été pensé et mis en œuvre dans une des applications existantes.

Toutefois, Bavardica est ma tentative de faire une plateforme de communication pas comme les autres. Bavardica (dérivé de Bavardage) se distingue par deux avantages par rapport aux messageries instantanées conventionnelles. Premièrement, il y a la présence permanente d'êtres artificiels qui interagissent avec d'autres utilisateurs. Ensuite, l'application repose sur un espace à deux dimensions qui représentent un petit monde dans lequel les utilisateurs sont des habitants représentés par des personnages animés. Ce projet est une bonne chance de planter le décor de ce que pourrait devenir un monde virtuel original.

Plusieurs formes de divertissements web sont impliquées dans le projet Bavardica. Dans sa forme initiale, Bavardica était essentiellement une application de chat, mais comme elle est en pleine évolution, Bavardica s'inscrit également dans les domaines suivants.

- Intelligence Artificielle

Comme me l'a enseigné le Dr Phil Grant, l'objectif principal de l'IA (Intelligence Artificielle) est d'essayer de rendre les ordinateurs capables d'effectuer des tâches ou les humains ont tendance à être bon. L'expression "Artificial Intelligence" a été inventée par John McCarthy dans les années 60. Il ya beaucoup de sujets qui pourraient aujourd'hui être considérés comme faisant partie du thème général de l'Intelligence Artificielle. Mais pour notre occurrence actuelle, nous ne considérons que les thèmes de la compréhension du langage naturel et la représentation des connaissances.

Maintenant, qu'est-ce que cela signifie pour une machine d'être intelligent? Alan Turing réfléchit à ce problème dans les années 50 et a élaboré le test suivant: Un humain poserait des questions à la fois à un ordinateur et à un homme situés dans d'autres salles; si l'interrogateur n'est pas capable de déduire (d'après les réponses) qui est la machine, alors on dit que l'ordinateur se comporte d'une manière intelligente. Une des premières tentatives de construire un programme qui passerait le test de Turing a été le programme ELIZA (par Joseph Weizenbaum). ELIZA mimait une conversation avec un psychologue et était assez impressionnant à première vue. Cependant, il manquait de bon sens et de compréhension réelle du langage naturel (qui sont tous deux très difficiles à acquérir d'un point de vue mécanique). Néanmoins, de nombreux programmes de ce genre ont suivi. Ils étaient basés sur des astuces comme:

-> La répétition exacte des déclarations de l'utilisateur (avec bien sûr des ajustements pronominaux). Par exemple, si l'utilisateur disait : "j'ai eu beaucoup de chance cette semaine", l'ordinateur pourrait répondre quelque chose comme : "vous avez eu beaucoup de chance cette semaine ?"

-> Précéder la déclaration répétée avec des introductions. Pour reprendre l'exemple précédent, l'ordinateur pourrait répondre "Qu'est ce qui vous fait dire que vous avez eu beaucoup de chance cette semaine ?" ou alors "Pourquoi avez vous besoin de me dire que vous avez eu beaucoup de chance cette semaine ?"

-> Demander "Pourquoi demandez-vous cela", ce qui a pour effet de changer le thème ou le niveau de la conversation.

Plus tard, Colby et Al ont écrit un programme appelé PARRY basé sur des idées de ELIZA, qui avait des propriétés plus complexes comme des éléments d'émotion et une meilleure connaissance de la grammaire. Il était destiné à imiter un patient paranoïaque qui croyait qu'il était persécuté. Dans une expérience réelle, 33 psychiatres ont été invités à évaluer le degré de paranoïa des patients qui communiquaient à travers des transcriptions dactylographiées. Trois d'entre eux étaient des patients humains et deux générés par Parry. Aucun des psychiatres n'a déclaré qu'il pensait qu'il y avait quelque chose d'inhabituel. Plus tard, les psychiatres ont été informés que certaines des transcriptions avaient été générées par la machine, mais ils étaient, dans l'ensemble, incapables de reconnaître ceux qui étaient de Parry. Bien sûr, toute cette situation est un peu bizarre de toute façon, vu que les paranoïaques humains se comportent généralement de façon étrange. Ce qu'il faut retenir ici, c'est qu'il peut être difficile de différencier un discours humain de celui d'une machine (Phil Grant).

Par ailleurs, avec le temps, les systèmes d'IA sont de plus en plus avancées, en fait, plus intelligents. Actuellement, l'un des plus importants programmes de ce type est ALICE (Artificial Linguistique Internet Computer Entity) de Richard Wallace. Selon Wikipédia, c'est un programme inspiré de ELIZA qui s'engage dans une conversation avec un être humain en appliquant certaines règles de correspondances heuristiques. Le programme utilise un schéma XML appelé AIML (Artificial Intelligence Markup Language). J'utilise un modèle semblable pour la mise en œuvre de la fonction AI dans ce projet.

- Mondes Virtuels

Un monde virtuel peut être défini comme une réalité synthétique, un environnement qui est totalement généré par un système informatique et qui est complètement transportif, dans le sens ou le participant contrôle un avatar immergé dans un monde artificiel (Armitage, Claypool et Branch, 2006). Les mondes virtuels sont destinés à des utilisateurs qui peuvent l'habiter et y interagir. Parfois, l'utilisateur peut également manipuler des éléments dans le monde en question et donc faire l'expérience d'une sorte de téléprésence. Comme le taux de personnes utilisant des mondes virtuels est en augmentation de 15% chaque mois et comme il y a près de 600 millions de personnes dans le monde entier enregistrés dans le monde virtuel d'aujourd'hui (selon K Zéro, un service de consultance en monde virtuel), je me suis dit que ce serait une bonne idée d'en apprendre davantage sur cette tendance et de m'y essayer. Dans sa forme actuelle, Bavardica est un monde virtuel dans le sens ou chaque bavard (avatar) peut voir et échanger avec les autres bavards sur la scène.

- Multimedia

Dans un navigateur Web, des opérations comme le calendrier, le streaming, la manipulation de pistes de musique, etc. ne sont pas du tout trivial en utilisant simplement du HTML et du JavaScript. Un grand avantage d'un plugin comme Silverlight, c'est qu'il donne beaucoup plus de facilité pour les opérations sur les fichiers multimédia. Le livre "Accelerated Silverlight 3" a été vraiment utile pour comprendre la classe System.Windows.Controls.MediaElement, qui fournit des capacités de lecture des médias. Ce contrôle peut gérer à la fois audio et vidéo dans une variété de formats tels que MP3, WMV, ASX, etc. Il y a même un support pour les codecs tiers et les nouvelles formes de médias tels que H.264, Advanced Audio Coding et mp4. Par ailleurs, le MediaElement de Silverlight contient beaucoup de propriétés, méthodes et événements qui permettent un large contrôle sur les médias. Certaines de ces propriétés sont AutoPlay, CurrentState, CanSeek, IsMuted, Position, NaturalDuration,NaturalVideoHeight et Volume. Les méthodes du contrôle MediaElement comprennent Pause, Lecture, SetSource et Stop. Certains de ces événements sontCurrentStateChanged, MediaEnded, MediaOpened, MediaFailed, ce dernier permettant une meilleure gestion des erreurs dans le cas où les médias ne peuvent pas être trouvés ou quand le format est incorrect.

Ce contrôle nous permet d'apporter une âme symphonique à ce projet. Ainsi, un autre aspect du projet est le streaming audio, qui est maintenant dans sa forme la plus basique avec des fichiers MP3 en musique de fond pendant la conversation. Il y a aussi des effets sonores qui servent d'alertes informant qu'un nouvel utilisateur vient de se connecter ou que quelqu'un vient de laisser un nouveau message.

Nous allons maintenant procéder à une explication de la manière dont l'application a été construite.

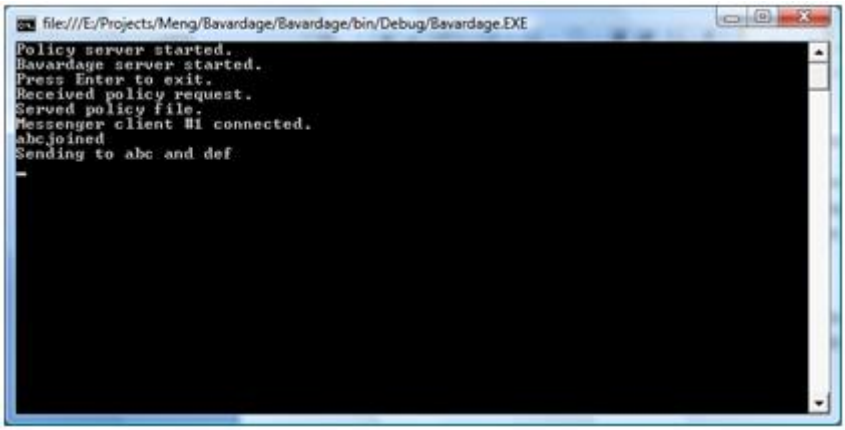
A travers Bavardica - Partie 2 : Evolution Du Projet

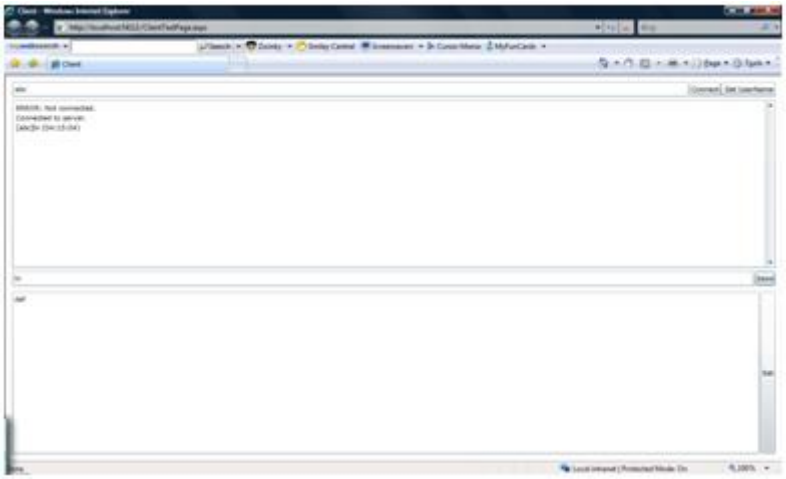
Par Necemon, 2010

Maintenant que nous avons discuté les domaines relatifs à Bavardica, nous pouvons entrer dans le vif du sujet, avec les étapes de la réalisation du projet.

1. Envoi d'un message public sur le réseau TCP à partir d'un client Silverlight

Le modèle choisi pour ce projet est le modèle de prototypage. En commençant d'abord avec un prototype jetable, je pouvais d'abord envoyer de simples paquets TCP en utilisant Silverlight pour le client et une application console en tant que serveur. C'était un chat très basique qui permet à un utilisateur, disons A d'envoyer un message à un utilisateur B de telle sorte que personne d'autre ne puisse lire le message en dehors de A et B. Le serveur est une application console .NET et le client est une application Silverlight hébergé dans une page ASP.NET





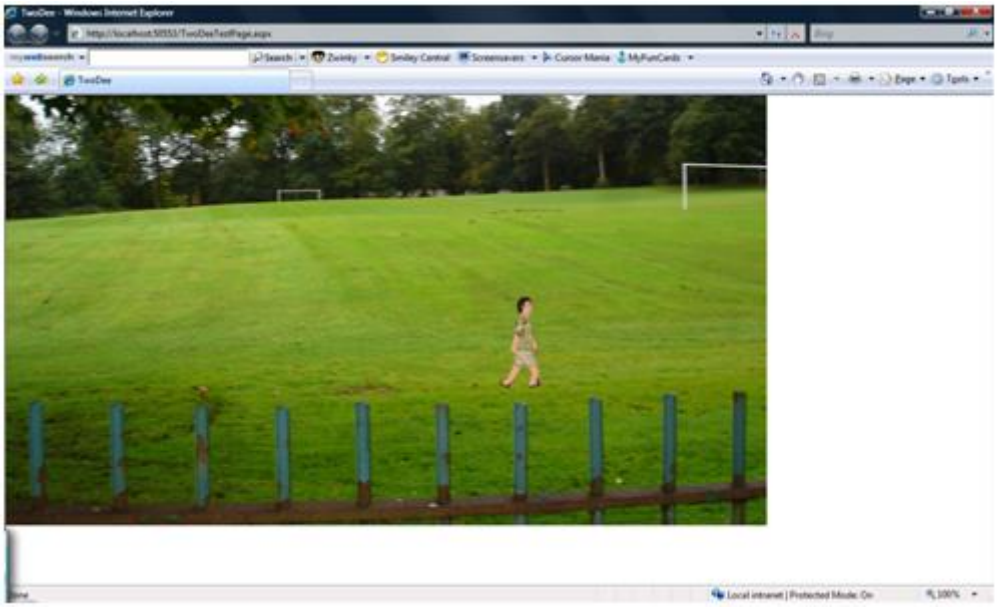
2. Premiers contacts avec les animations Silverlight

Le second programme que j'ai écrit était une simple application Silverlight avec un personnage animé en 2D marchant autour d'un parc qui était en fait le fond de la page Web. Cette expérience avait pour seul objectif l'étude du mouvement d'un personnage sur une scène 2D.

Le premier point clé de ce prototype est la façon dont l'avatar est animé quand une touche de direction est pressée:

Le deuxième point important est que le modèle utilise des "sprite sheet", c'est à dire, une animation image par image. Il consiste à glisser horizontalement une séquence d'images adjacentes avec un fond transparent, affichant ainsi une image à la fois quand une touche directionnelle est pressée.

Cette approche utilisait des KeyAnimationUsingKeyFrames mais sera abandonnée plus tard dans le projet car elle n'impliquait pas d'interpolation et qu'elle aurait exigé beaucoup de temps à dessiner chaque caractère dans chaque position unique.



3. Placer le serveur silencieusement en arrière-plan (en tant que service web)

Il était prévu d'intégrer les deux prototypes de sorte que les utilisateurs puissent à la fois chatter et se déplacer sur la scène 2D. Il était également prévu que l'application serveur reste silencieusement dans un service Web de sorte que l'application console n'aies pas à rester ouverte pendant toute l'opération. Les deux objectifs ont été atteints dans le prototype suivant.

En fait, j'ai pensé à la construction d'une application serveur plus robuste sur lequel je pourrais construire un prototype de l'évolution qui allait progressivement grandir avec le projet. Ce serveur avait deux exigences principales:

- Il fallait se débarrasser de l'application console. Il faut juste être un processus silencieux dans le fond. Pour ça, j'ai juste utilisé les services Web.

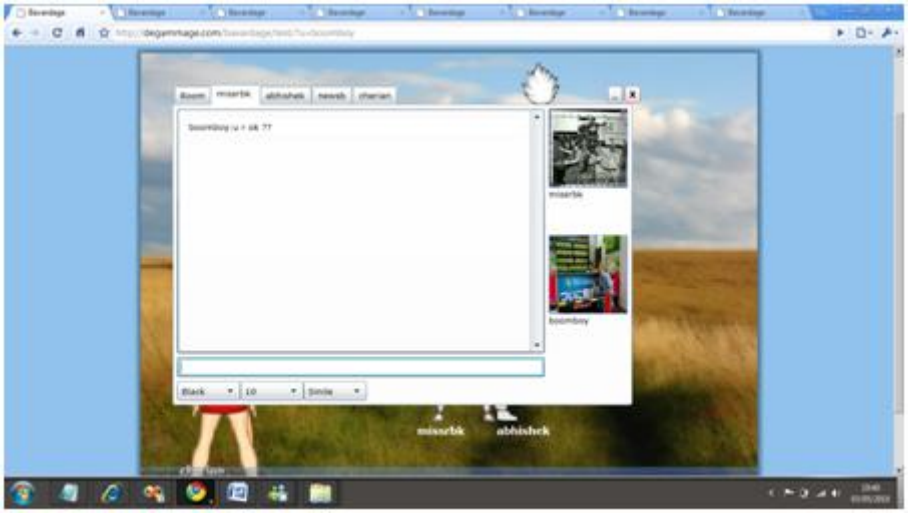
- Le serveur devait "pousser" des données aux clients. C'est l'envoi régulier des données, même quand les clients ne le demandent explicitement chaque fois. Dans le modèle classique HTTP, le client fait d'abord une demande au serveur et ensuite seulement le serveur peut répondre. Une solution à ce problème était d'utiliser les services web Duplex WCF (Windows Communication Foundation), et c'est ce que j'ai fait. C'est un modèle de communication asynchrone à deux directions.

C'est intéressant d'observer comment les services Web duplex WCF fonctionnent.

4. Gros plan sur le chat privé

Alors que le chat public se produit dans une fenêtre commune, le chat en "face à face" implique une fenêtre pour chaque conversation. L'organisation de plusieurs conversations dans Bavardica revient à utiliser des onglets, c'est à dire le contrôle Silverlight TabControl.

L'onglet RoomTab est celui utilisé pour les conversations publiques. Pour démarrer une conversation privée, l'utilisateur doit juste cliquer sur le nom de la personne à laquelle il veut parler (Il y a une liste de noms d'utilisateurs disponible dans la salle publique)



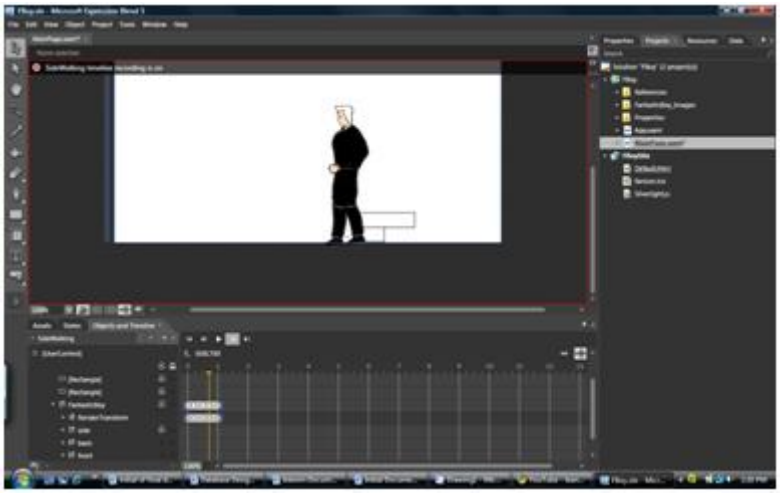
5. Dessiner et animer les personnages

L'étape suivante fut la conception et l'animation d'avatars 2D. Il y avait déjà un avatar dans les prototypes précédents, mais ce n'est pas amusant si tout le monde a la même apparence dans un monde virtuel. De plus, cet avatar là avait été emprunté à un autre développeur Silverlight (Darren Mart). Et le modèle précédent a présenté beaucoup d'inconvénients. Outre le fait que d'une animation Sprite Sheet ne fournit

pas d'interpolation entre les images, ce modèle implique aussi beaucoup de dessins. Chaque position individuelle devait être conçue manuellement. Mais le plus gros problème dans notre contexte est l'inexistence de personnalisation de ce modèle: l'un des premiers objectifs non-négociable de ce projet était la possibilité de mélanger et d'assortir des vêtements et des tenues d'autres ainsi que leurs couleurs. Cette tâche est assez simple dans un monde 3D, car il est facile de changer de mailles et de textures; mais dans un monde en 2D, c'est un peu délicat de changer une partie de l'équipement, sauf si nous choisissons de redessiner le personnage pour chaque combinaison imaginable de vêtements. La solution mise en oeuvre dans Bavardica a été plutôt de créer un ensemble d'images de référence qui, collectivement, forment un caractère. Donc j'ai fini par dessiner mes propres personnages.

Alors, même si ceci est principalement un projet de programmation, et non un projet graphique, il est utile de mentionner que beaucoup de temps et d'efforts ont été investis dans l'élaboration et l'animation des personnages originaux. La personnalisation est également une caractéristique importante d'un monde virtuel. Les utilisateurs devraient être en mesure de choisir à quoi ils ressemblent et comment ils sont habillés. Pour ce projet, j'ai choisi de tirer quatre personnages de base (2 garçons et 2 filles) et au moins deux costumes et deux coiffures pour chaque personnage. En raison de la contrainte de temps encore, je ne pouvais pas dessiner et animer plus que cela. L'opération implique de dessiner et colorier les corps, puis dessiner des vêtements par dessus ces corps. Trois couleurs de peau de base sont disponibles et les utilisateurs peuvent modifier les couleurs de leurs vêtements et les cheveux. Voici un aperçu de l'un des personnages:

Les personnages sont animés essentiellement de manière à pouvoir marcher, s'asseoir et se tenir debout. Le problème avec les animations, c'est qu'il fallait le faire à chaque fois sous les trois angles (avant, arrière, latéral). Il y a aussi des animations pour leur donner l'air vivant, même quand ils sont debout, au repos. Ces modèles ont utilisé le modèle de DoubleAnimationUsingKeyFrames par opposition à la précédente qui a utilisé des KeyAnimationUsingKeyFrames. Microsoft Expression Blend a été utilisé pour traiter les animations.



Il y a deux principaux avantages pour le système d'animation qui a été utilisé:

- l'interpolation Silverlight rend les choses soyeuses

- Les personnages sont personnalisables. Ils peuvent simplement changer de bottes ou de style de cheveux (ou autre) à la guise des utilisateurs.

Malheureusement (ou heureusement), la manipulation des personnages dans un monde virtuel n'est pas seulement sur le design. Elle implique beaucoup de codage. Commençons par la création d'un personnage.

Quelques "Arrays" sont maintenus, en offrant une variété de tenues à l'utilisateur, ainsi que les formes et les genres:

Selon le choix de l'utilisateur, les éléments requis sont exposés et les pièces qui ne sont pas requises sont masquées. Cependant, les changements de couleur suivent une autre logique.

Ce qui se passe ici est que le fichier source pour chaque partie du corps est récupéré à partir d'un répertoire différent qui change en fonction du choix de l'utilisateur. Dans l'application, lorsque l'utilisateur peut changer les couleurs de toutes les parties de leur costume, la même logique est appliquée.

C'est à peu près tout pour ce qui est de la création du personnage. Une fois que le personnage est créé, des détails sur son apparence sont stockés dans la base de données. A partir de là, chaque fois que l'utilisateur est en ligne, le serveur peut envoyer tous ses détails à l'ensemble des autres clients. Dès qu'un client réalise la présence d'un autre utilisateur en ligne, il peut représenter ledit personnage dans sa scène en fonction des données qui avaient été soumis:

Le réglage les détails des autres personnages fonctionne comme dans la création du personnage. C'est surtout une question de cacher et montrer des images au bon moment.

Maintenant que nous avons vu comment le code relatif à l'apparence des personnages fonctionne, nous allons maintenant attaquer le code utilisé dans l'animation et les mouvements. Tout part d'une touche pressée par un utilisateur, un évènement géré par Silverlight. Dans notre exemple, nous allons considérer la touche droite:

6. Le Cybavard

Après la conception et l'animation des personnages, l'accent a été mis sur le Cybavard (Cyber Bavard), la partie "intelligence artificielle" du projet. La logique derrière le Cybavard est inspirée du bot ALICE présenté plus tôt dans cette présentation. Ce qui est intéressant à propos de ALICE est qu'il était basé sur un dialecte XML appelé AIML (Artificial Intelligence Markup Language). En raison de la nature universelle de XML, il peut être utilisé avec une grande variété de langues. Le point est que, comme il s'agit d'un projet .NET, le AIML pourrait également s'adapter aux Cybavards. Il s'agissait juste de trouver un interpréteur AIML C#. Pour ce projet, j'ai utilisé une librairie .NET, AIMLBot.dll (par Toll-Nicolas), publié sous la licence publique générale GNU. Ce programme offre beaucoup d'avantages:

- Très petite taille: environ 52 kilo-octets

- Très rapide: peut traiter 30 000 catégories en moins d'une seconde

- Moyens de sauver le cerveau du bot dans un fichier binaire

- Une API simple et logique

Le fonctionnement interne de ce programme est également assez simple. Il est basé sur du filtrage: pour chaque phrase entrée, il vérifie dans sa base de connaissances quelle réponse cadrerait mieux:

Donc ce que je fais est d'ajouter une référence au fichier binaire de mon service web, j'ai aussi télécharger la base de connaissances sur le serveur. La base de connaissances est principalement un ensemble de fichiers AIML que je peux modifier selon mes besoins. Je pourrais aussi ajouter mes propres fichiers AIML. De la classe Cybavard, ce que j'avais à faire était de créer un objet bot et le chargement des fichiers AIML.

Dans la partie suivante, nous allons revisiter la conception graphique des personnages.

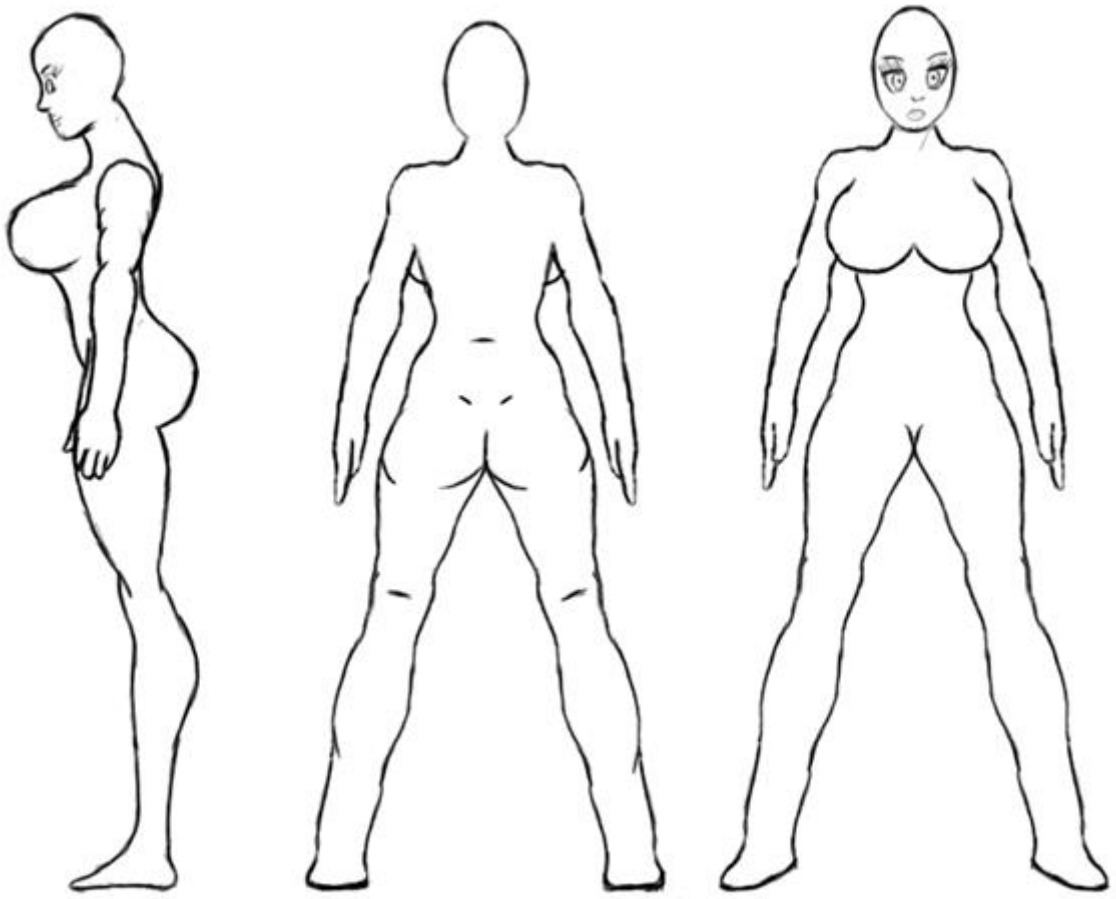
A travers Bavardica - Partie 3 : Conception Graphique Des Personnages 2D

Par Necemon, 2010

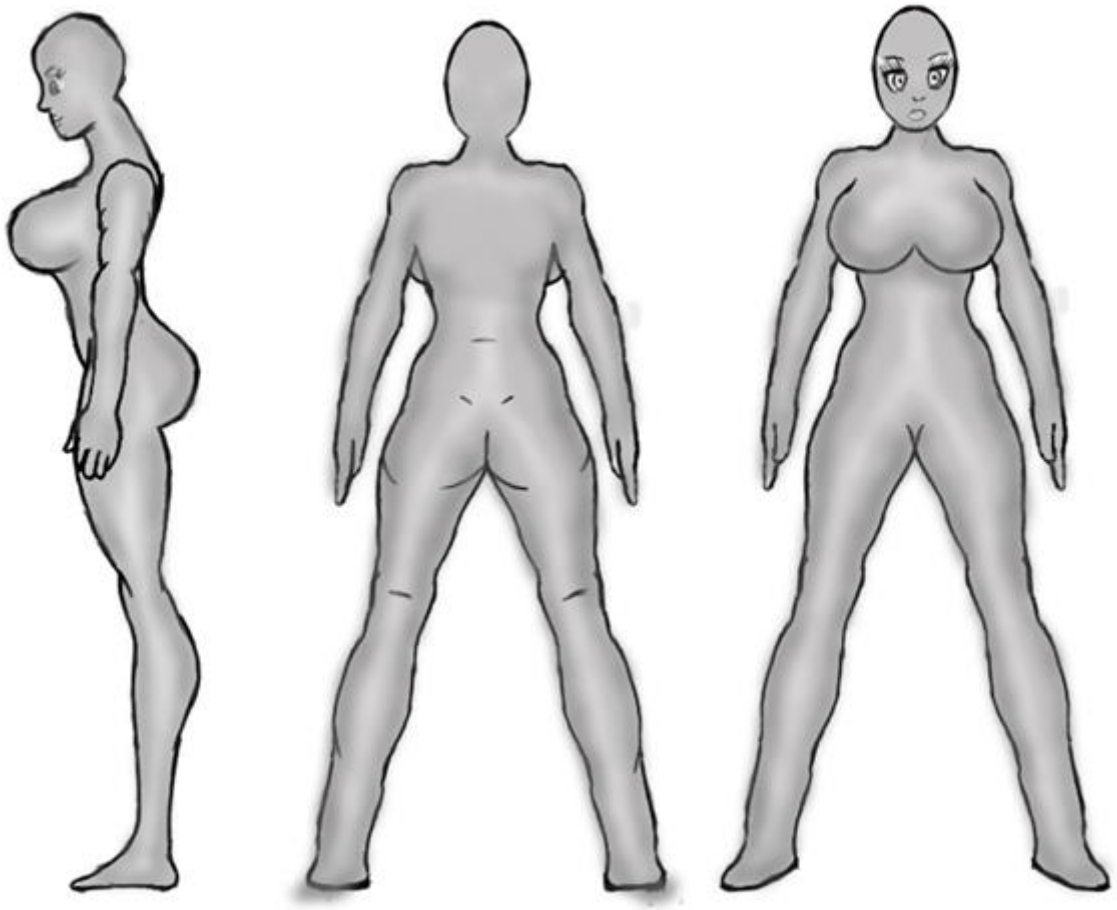
Comment j'ai dessiné les personnages 2D...

En raison de la nature artistique du dessin, qui n'est pas forcément liée à l'informatique, j'ai pensé qu'il valait mieux décrire ce processus dans une autre partie juste après la description de l'évolution technique. Ce travail a été une partie intégrale du projet et même si elle n'implique pas directement de programmation (en tout cas, pas avant d'entrer dans l'animation), il est aussi un travail sur ordinateur impliquant une tablette à stylet et un logiciel de traitement d'images (Adobe Photoshop). Les images suivantes décrivent les étapes dans la conception graphique d'un personnage (un personnage sur les quatre qui ont été créés)

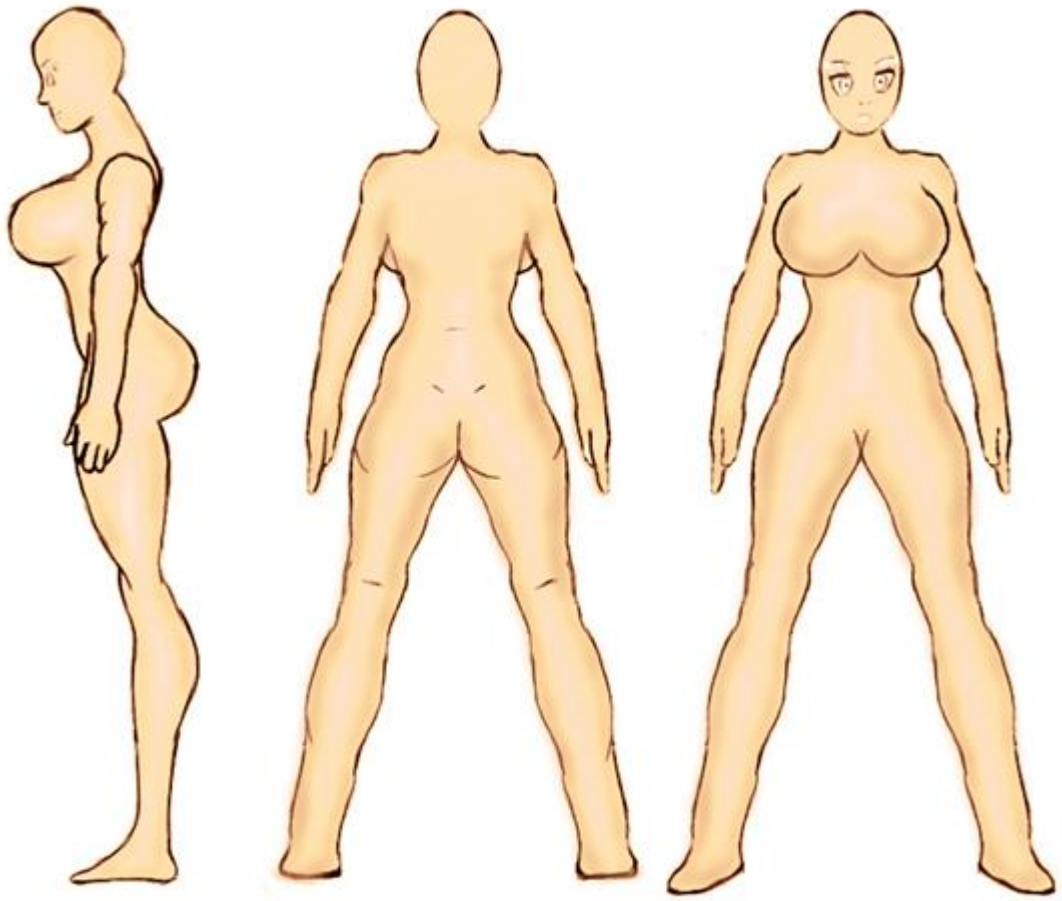
A. Croquis



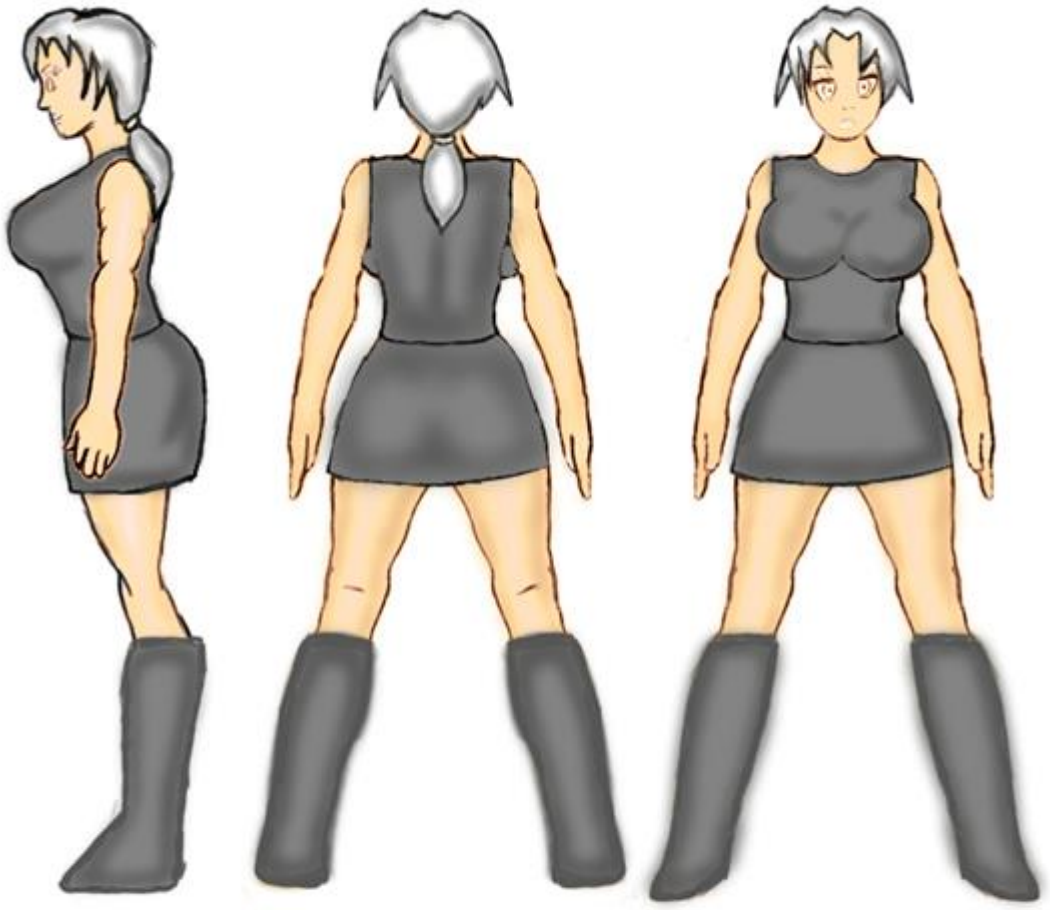
B. Remplissage



C. Coloriage



D. Habillage



Pour plus de détails sur les techniques d'animation, tu peux voir les tutoriels que Darren Mart a écrit. Pour découvrir ce que seront les prochaines étapes avec Bavardica, passe à la prochaine partie.

A travers Bavardica - Partie 4 : Possibilités d'améliorations

Par Necemon, 2010

Ceci est la dernière partie de la série sur Bavardica. Outre une interface utilisateur conviviale et colorée, les autres caractéristiques comprennent la modification de personnages en temps réel (pendant qu'ils sont sur la scène). Là encore, le style de cheveux, la peau, les vêtements et les chaussures peuvent être modifiés. Les couleurs de tous ces vêtements peuvent également être modifiés. La façon dont cela fonctionne est assez similaire à la façon dont un personnage est créé.

Entre temps, quelques autres fonctions ont été ajoutées :

- Détection de collision : Ceci est pour s'assurer que les bavards ne sortent pas de la scène. Aussi, ils ne se marchent pas les uns sur les autres quand ils se déplacent verticalement. Cette fonctionnalité est basée sur de simples conditions "if" qui vérifient si le déplacement ne conduit pas hors de la scène ou à une position occupée par un autre personnage.

- Bubbling: cette fonction est utilisée pour traiter le texte entré dans la bulle (mode graphique). Chaque fois qu'un utilisateur envoie un message, il apparaît sur la scène comme une bulle volante qui s'élève et disparaît progressivement. Une animation a été conçue à cet effet. Pour ce faire, des animations de mouvement et d'opacité sont appliquées à l'objet de bulle. Chaque touche pressée est traitée individuellement comme la bulle ne repose pas sur une zone de texte. Le problème principal dans cette tâche a été que les positions des caractères spéciaux changent selon le type de clavier. Pour l'instant, le texte des bulles a été optimisé pour des conversations sur claviers américains et britanniques.

Voici les améliorations qui sont prévues:

- Plus de salles

L'application actuelle contient une seule scène où toutes les conversations publiques se passent. Une bonne extension pourrait être d'avoir de nombreuses scènes et une carte pour naviguer parmi les lieux.

- Plus de musique

La musique de fond est toujours la même piste MP3 jouée en boucle. La prochaine version pourrait éventuellement avoir une liste de chansons comme fond musical. Vu qu'il y aurait éventuellement plusieurs scènes, chaque scène pourrait avoir un thème différent et la musique qui va avec.

- Plus de Cybavards

En raison de la grande quantité de temps et les données nécessaires pour simuler la conversation d'un Cybavard unique, il n'a pas été possible de fournir plus d'un de ces Cybavards dans cette première version. Le processus de leur création est assez simple. C'est surtout une question de temps. Par exemple, le cybavard dans cette démo a une base de connaissances de près de 4 mégaoctets (de texte).

- Plus d’animations pour une meilleure interaction

Les seules animations disponible pour le moment sont la marche dans les 4 directions et la respiration quand le personnage est inactif. Il est prévu d'ajouter plus d'animations comme la danse, courir, sauter, etc. Il y aura quelques mouvements pour exprimer l'humeur du personnage. À un stade ultérieur, il pourrait y avoir des animations qui engagent plus d'un personnage, comme étreindre ou serrer la main.

Une version de l'application est disponible actuellement à http://bavardica.com

10 choses que j'ai apprises en construisant Bavardica

Par Necemon, 2010

Ceci juste un résumé des leçons les plus importantes que j'ai apprises de la construction et l'édition de Bavardica.J'espère sincèrement que tu en tires quelques informations utiles (ou des rappels si tu sais déjà tout ceci).
(PS:une version de Bavardica est disponible à http://bavardica.com)

1. Personne ne va utiliser une application sauf si elle fournit une certaine valeur ajoutée, quelque chose d'intéressant qu'ils ne trouvent pas déjà dans ce qu'ils utilisent habituellement. Cela est vrai même pour des applications gratuites. Ils pourraient l'essayer mais ils l'abandonnent assez rapidement, s'il n'y a aucune incitation réelle.

2. Le fait même qu'ils sont priés de s'inscrire va décourager beaucoup de gens de tenter une application web. Les causes les plus évidentes pourraient être qu'ils ne veulent pas perdre de temps à le faire, ou ils ne veulent pas donner leurs coordonnées pour des raisons de confidentialité. Donc, au moins une version d'essai devrait être offerte aux visiteurs sans inscription afin qu'ils puissent voir de quoi il s'agit avant de prendre une décision.

3. Les différentes commandes et les fonctionnalités sont évidentes (seulement) pour le développeur qui a construit l'application. Cependant, elles peuvent paraître étranges et compliquées à utiliser pour certains utilisateurs. L'application doit donc être aussi simple que possible et il devrait y avoir quelques explications pour chaque commande (si possible à l'intérieur de l'application, pas forcément dans un fichier externe).

4. Google est ton ami. Ou plutôt, les moteurs de recherche en général. J'ai passé beaucoup de temps à lire des livres sur Silverlight au début du projet, mais chaque fois que j'ai eu un doute, je comptais plus sur les recherches Web que sur les livres. J'ai pu trouver la plupart des réponses que je cherchais dans les forums techniques. Je pouvais aussi trouver des travaux liés et réalisés par des développeurs seniors (surtout Anoop Madhusudanan, Tomasz Janczuk et Darren Mart dans ce cas précis).

5. L'architecture client-serveur et les Services Web silencieux. Ce point est purement technique. J'ai appris à faire une application résidant silencieusement dans le fond comme un service Web ou un service Windows; Et j'ai appris à pousser les données vers le client via HTTP en utilisant un service WCF Polling Duplex.

6. Plus on a de choix, mieux c'est. Les utilisateurs aiment avoir beaucoup de contenu à choisir. Ils tiennent à exprimer leur personnalité, à personnaliser leur avatar. Les travaux artistiques consomment souvent plus de temps que le codage réel, mais n'est-ce pas nécessaire ? Dans les versions à venir, je vais certainement ajouter plus de pièces, plus de vêtements et plus d'animations. Mais...

7. Pas tout à la fois. Il est important d'aller une étape à la fois, de ne pas se laisser submerger. C'est une question de gestion des priorités.

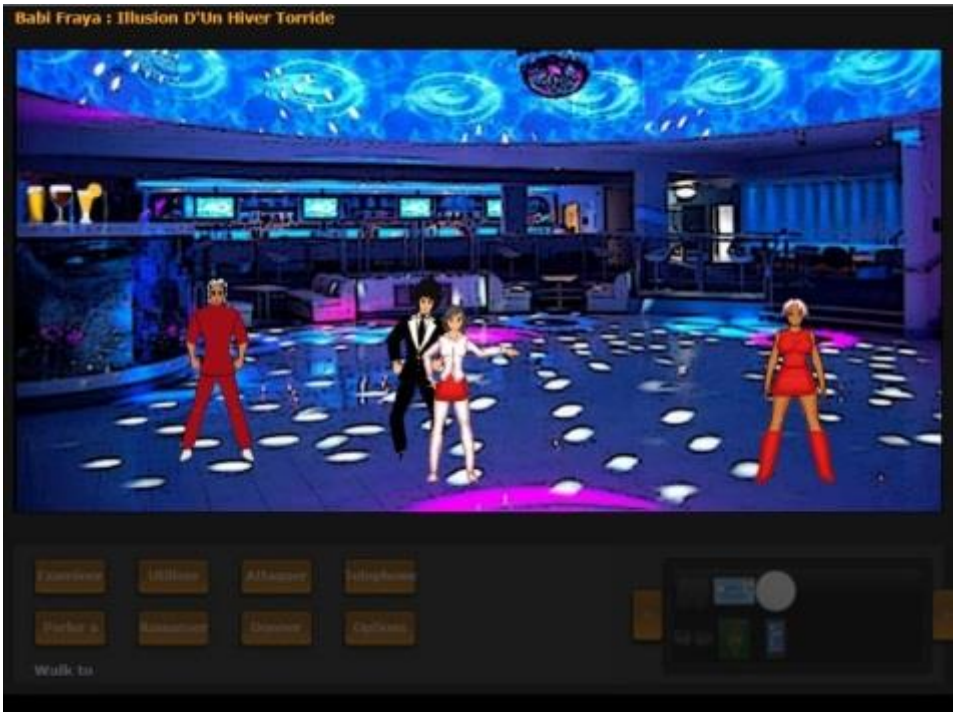
8. Le Travail paye. Ici il est surtout question de la satisfaction (du sentiment du devoir accompli) qu'on obtient du travail. On se sent bien d'accomplir quelque chose, même si c'est juste une petite chose pour l'instant. Ça motive pour la suite.

9. Le feedback est inestimable. Je ne pouvais pas apprendre toutes ces leçons que je décris dans ces messages, si je n'avais pas eu quelques garçons et filles pour me montrer mes erreurs, de me dire ce qu'ils n'aimaient pas dans mon travail et comment je pourrais l'améliorer. Je voudrais saisir cette occasion pour remercier tous ceux qui ont testé Bavardica. Merci pour vos commentaires.

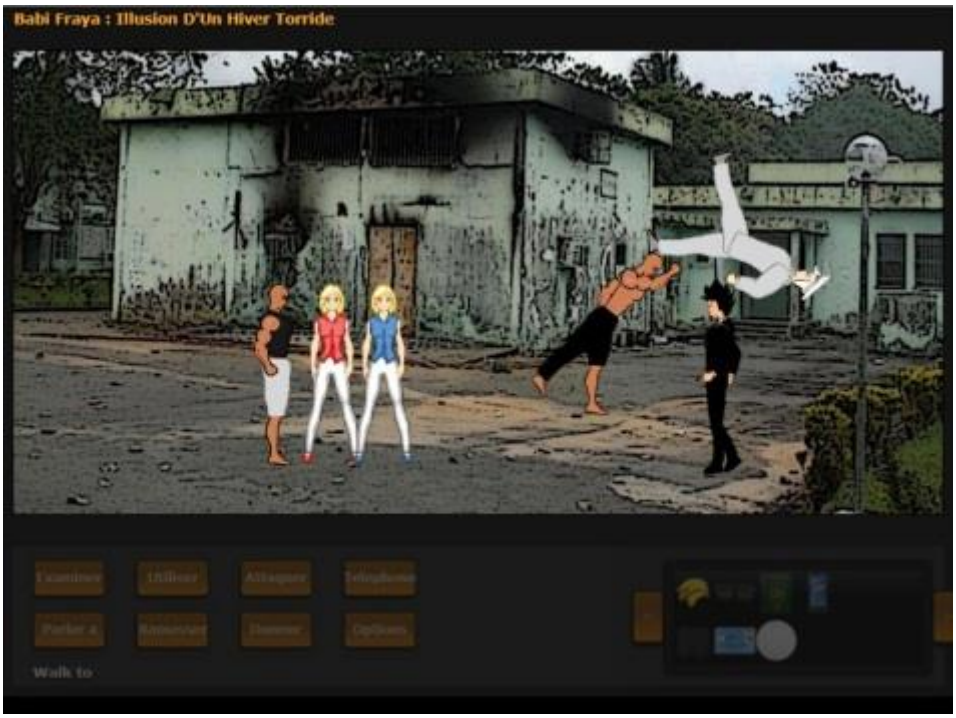
10. Il y a un long chemin à parcourir pour ce qui est de construire un monde virtuel convenable.

Babi Fraya : Illusion d'un Hiver Torride

Par Necemon, 2013



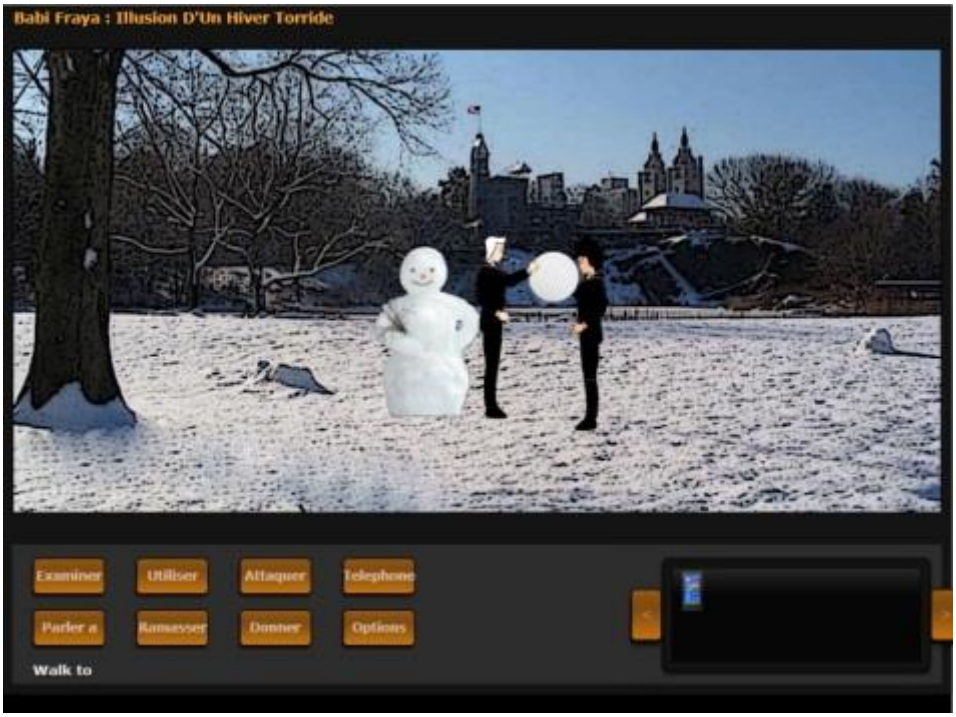
Je viens de publier une version beta de mon premier jeu complet. Si tu as un moment pour l'essayer, j'espère que tu vas apprécier: http://babifraya.com/



Babi Fraya™ est un jeu d'aventure du style "Point & Click", donc l'intérêt prédominant se focalise sur l'exploration, les dialogues et la résolution d'énigmes. C'est aussi une fiction dont l'objectif est de raconter une histoire, le joueur pouvant agir sur l'histoire, vu qu'il y a plusieurs embranchements scénaristiques.

Scenario

Nous revenons donc en Décembre 2010 avec Lewis, étudiant à New York. Il rentre chez lui à Babi pour les fêtes de fin d'année. Au début c'est cool, les shows et les festins, sur la Rue Princesse ou les plages ensoleillées, avec les vieux amis et les nouvelles rencontres... Jusqu'à un soudain changement d'ambiance. Qu'est ce qui n'a pas marché ? Personne ne sait. Tout est allé très vite... Dans l'espace de quelques minutes, ce fut un passage de l'harmonie au drame: cris, bombardements, pillages, pénuries, fermeture des frontières. Quand Babi La Joie devient Babi Frayeur, ce qui était supposé être des vacances de rêve ressemble de plus en plus à une mission commando. Que faire maintenant ? Combattre ou fraya ? Se battre pour fraya ? À toi de choisir.



En résumé, j'ai juste 3 petits commentaires:

- Ceci est un jeu d'aventure du style point-and-click, construit avec Microsoft Silverlight.

- L'histoire propose une perspective décalée de la Bataille de Babi #CIV2010, qui transforme des vacances de rêves en une mission commando.

- Le jeu est plutôt facile et il ya quelques parties qui pourraient potentiellement être améliorées, mais ça marche plutôt bien...

Babifraya.com. Toute réaction est bienvenue,

N.

Redynamisation : Kpakpatoya 2.0 en tant que plateforme interactive

Par Necemon

#Kpakpatoya

1. Introduction

Kpakpatoya™ est un catalogue de ce qui est nouveau, populaire et intéressant en ligne avec un accent particulier sur l'Afrique. Les utilisateurs peuvent publier du contenu, commenter et décider à travers des votes de ce qui est bon (kiffs) et ce qui est nul (zaps). Les liens qui reçoivent l'approbation de la majorité progressent vers le sommet, ainsi la page d'accueil serait constamment en mouvement, avec plein de liens frais et intéressants. Kpakpatoya™ fait partie de Evasium Network.

"Kpakpatoya" est un terme tiré de l'argot ivoirien qui désigne l'action de rapporter, relayer des histoires, des nouvelles, des potins, voire de favoriser la circulation de commérages et de rumeurs. Dans notre contexte, il serait donc plus approprié de comprendre "Kpakpatoya" dans le sens de scoops, de révélations insolites, et même d'actualités captivantes.

2. Contexte Historique

Retour en Avril 2011. La Côte d'Ivoire est alors en train de sortir d'une crise majeure, crise au cours de laquelle les technologies de l'information et de la communication ont joué un rôle important quand il s'agissait d'apporter une assistance pratique aux victimes collatérales de la crise ivoirienne. Au-delà de simples mots, les hashtags tels que #CIV2010 et #CIVSocial permettaient de faire circuler des informations critiques et d'apporter des solutions concrètes auxdites victimes. À travers des dons, des call-centers et surtout Twitter, qui se présente alors comme l'outil idéal pour la circulation d'informations en temps réel, voir urgentes.

La communauté ivoirienne sur Twitter est alors en plein essor, avec de plus en plus d'inscriptions, de tweets et de contributions. Mais cette communauté était toujours relativement faible en termes de nombre. Pour un si petit groupe sur Twitter, comment se retrouver, et échanger autour de thèmes communs ? Encore les #Hashtags, bien sûr… Sur les réseaux sociaux, le hashtag est un mot ou ensemble de mots précédé du symbole '#'. Le hashtag sert à centraliser les messages autour d'un terme bien précis. Il fait office de mot-clé pour que les utilisateurs puissent commenter / suivre une conversation ou se retrouver autour d'un sujet, par exemple les mesures de ravitaillement en temps de crise ou encore les urgences médicales.

Mais dans cette période d'après-guerre, place à des tags moins sinistres. #Kpakpatoya tombait à pic pour permettre à cette petite communauté de se retrouver dans une ambiance qu'on pourrait qualifier de « cool ». C'était le tag que ces « influenceurs » et « pionniers » ivoiriens sur Twitter utilisaient pour se passer des nouvelles locales, des « Inside Joke », des tubes, ou d'autres sortes d'infos ou de contenus intéressants relatifs entre autres, à des évènements sportifs, des (non) conférences technologiques, ou encore le quotidien de chacun.

Percevant une occasion d'améliorer certaines conditions socio-technologiques, j'ouvre dans la même période le site kpakpatoya.com. Au début c'était juste un flux des tweets du fil #kpakpatoya + quelques extras, notamment un condensé de l'actualité locale (ivoirienne) et internationale en temps réel. Puis j'ai essayé avec plus ou moins de succès de proposer quelques applications telles que des forums de discussions, le blog et la possibilité de tweeter directement depuis le site. J'avais aussi ouvert le compte Twitter @kpakpatoya et initié la conception de l'application mobile, qui continuent de présenter l'actualité tout comme la page Facebook et le site web.

C'est dans cette forme d'écosystème que le hashtag #Kpakpatoya montait en popularité, à travers une combinaison de compétences techniques et sociales. En suivant le fil #kpakpatoya, même pour un nouveau venu, il était facile de retrouver les sympathisants de la culture ivoirienne, leur actualité, ainsi que les thèmes locaux les plus populaires.

3 ans plus tard, la scène Web ivoirienne se présente sous une forme complètement différente…

D'une part, cette communauté virtuelle s'est beaucoup élargie, ce qui est une très bonne chose, puisqu'il y a beaucoup plus de contenu, de projets web, d'entrepreneurs web, de blogueurs, d'utilisateurs, etc. Sauf qu'un simple hashtag ne suffit plus pour une exploration efficace du contenu. Ce ne serait pas pratique de tous communiquer autour d'un seul hashtag. Bien évidemment, il y a toujours l'option de « décomposer », d'utiliser les hashtags alternatifs qui ont émergé entre-temps (même si #Kpakpatoya demeure l'un des tags les plus populaires). C'est une bonne solution, à condition d'être constamment informé des nouveaux utilisateurs et tags à suivre pour ne pas se perdre dans tout ce volume de contenu et ne pas manquer l'essentiel de l'information récente.

D'autre part, il y a eu très peu d'évolution par rapport à #Kpakpatoya, peu d'efforts que ce soit sur les plans de la technologie, de l'infographie, de la communication, des infrastructures, etc. La page Facebook a moins de 2000 fans, la page Twitter a autour de 8000 followers, l'application mobile est obsolète, et jusque récemment, les composants du site web qui avaient pas mal de succès dans les premiers mois ont été quasiment délaissés (je t'épargne les chiffres relatifs à la baisse du trafic web). Il n'y a presque pas de contributeurs à l'avancée du projet, plus que des utilisateurs. Ce qui aurait pu être un projet communautaire décent est passé à un état latent.

C'est vrai que la formation de la communauté s'est faite de manière assez organique, il n'y avait aucune stratégie concrète, tout s'est passé progressivement et naturellement, ce qui explique peut-être la décontraction après que l'effet de buzz soit passé, chacun étant préoccupé par ses projets personnels. J'avoue que les applications que j'avais écrites dans le cadre de ce projet étaient quasiment passées en mode maintenance et que j'étais moi-même passé à d'autres choses. Bref, personne n'intervenait.

Un homme sage a dit un jour : s'il y a quelque chose de potentiellement intéressant et que tu ne comprends pas pourquoi personne ne le fait, c'est parce que toi-même tu ne l'as pas fait. That's DIY : Do It Yourself.

Je reprends donc la direction du projet afin d'essayer de produire quelque chose d'utile. Ces dernières semaines, avec mon équipe, nous avons procédé à quelques rénovations, essayant de produire une plateforme plus interactive (si possible avec valeur ajoutée) pour faciliter la gestion du contenu, les rencontres et les échanges au sein de la communauté.

3. Description

Kpakpatoya™ est propulsé par Evasium ®, une petite équipe de geeks. Nous construisons des logiciels, notamment des applications Web orientées vers le divertissement et la formation intellectuelle.

Sur Kpakpatoya™, on discute un peu de tout, mais les thèmes les plus sollicités sont les suivants : Affairages, Clash, Eco-Politique, Culture, Insolite, Babi Actus, Afrique, Sport, Diaspora.

Au delà de la possibilité de lire les actualités, il s'agit maintenant d'ajouter une touche d'interactivité, ou chacun peut proposer des liens, discuter/commenter et liker les notes existantes. Un Kiff est un vote positif (comme un "j'aime", un "+1") pour une note. Un Zap, c'est l'inverse, c'est un vote négatif pour une note. Tout utilisateur de Kpakpatoya™ peut kiffer ou zapper une note, et puisque le score de la note dépend de la différence entre ses kiffs et ses zaps, chaque vote aide à établir la position de la note dans le classement de Kpakpatoya™. Les notes pourraient donc être listées par ordre de popularité, pour permettre aux prochains visiteurs de voir l'essentiel.

Rien de bien complexe, juste un système de partage comme on en trouve un peu partout sur le Web. Par exemple, les entreprenerds ont Hacker News, les Américains ont Reddit, les Français ont tapemoi.com. Le concept existe déjà depuis un moment, c'est juste qu'il n'y a pas de version Ivoirienne/Africaine.

Kpakpatoya™ est disponible dans 2 langues.

En Français :

https://kpakpatoya.com/?lang=fr
https://www.facebook.com/Kpakpatoya
https://twitter.com/kpakpatoya

En Anglais:

https://kpakpatoya.com/?lang=en
https://www.facebook.com/KpakpatoyaNews
https://twitter.com/kpakpatoyanews

Kpakpatoya™ pourrait devenir une référence en matière d'actualité et d' «affairages» pour les internautes ivoiriens (ou pas). Mais ce n'est pas ça l'objectif principal. L'objectif actuel est de proposer un outil utile pour étendre les options de partage de contenu local et de suspendre la pause de Kpakpatoya qui n'a que trop duré.

Sur une note plus personnelle, je pense qu'il faut plutôt voir ce projet comme une expérience qui pourrait contribuer à optimiser la visibilité des contenus africains en général et ivoiriens en particulier.
Pour avoir géré le site web, le compte Facebook et le compte Twitter de Kpakpatoya sur ces 3 dernières années, il me semble que beaucoup de followers réclament une meilleure interaction et recherchent une meilleure visibilité pour leur contenu, à savoir leurs projets, liens postés, ou simples avis et commentaires. Il est donc possible que cette version du site réponde à leurs attentes.

Il y a encore beaucoup de choses qu'on pourrait ajouter ou améliorer, on pourra le faire au fur et à mesure en tenant compte des feedbacks, mais le prototype de cette nouvelle version est relativement stable, et ce qu'il faut maintenant c'est l'essayer. C'est-à-dire y créer ton compte, y tester les différentes formes d'interaction, apporter ton feedback et partager le lien avec tes contacts pour qu'ils puissent en faire de même.


kpakpatoya.com
On pleure devant les infos
Mais on en redemande
comme des nymphos
on s'affaire avec
#Kpakpatoya

Projet Flux : Plateformes Communautaires du style "Social News"

Par Necemon

Juste une petite note pour récapituler la situation de mes projets en cours de dévéloppement.

Ces derniers temps, j'ai construit une série de plateformes communautaires, des applications Web du style "Social News" où les utilisateurs peuvent publier du contenu, commenter et décider à travers des votes de ce qui est bon (kiffs) et ce qui est nul (zaps). Les liens qui reçoivent l'approbation de la majorité progressent vers le sommet, ainsi les premières pages seraient constamment en mouvement, avec plein de liens frais et intéressants.

Rien d'impressionnant, mais ceci me donne une bonne base pour monter une plate-forme Web robuste, c'est a dire avec les caractéristiques universelles:
-Sécurité (cryptographie, inscription, connexion, gestion des membres, rappel de mot de passe, etc.)
-Couche Données / mise en cache (ou comment gérer un certain volume de "big" data en constante croissance)
-Services Back-end (envoi de courrier électronique, pings, requêtes http, journalisation)
-Une application ASP.NET MVC stable
-gestion et actions des utilisateurs : votes, commentaires, publications, modération
-Gestion des ressources audiovisuelles et de la logique cliente (images, Javascript, css, compression/centralisation)
-Bonus: J'apprends beaucoup par rapport au déploiement sur les serveurs (paramètres IIS, répertoires virtuels, ports, pools d'applications, etc.)

Bref, le genre de choses qui pourraient être utile dans une application Web moderne. En fait, c'est mon application Web la plus complexe à ce jour.

En mode Extreme Programming
Je reste intéressé par la construction de mondes virtuels et de jeux en ligne, mais je pense à publier des prototypes régulièrement. Ainsi, je peux m'améliorer au fur et à mesure, en apprenant et en déployant à chaque étape, au lieu de travailler sur un même projet pendant plusieurs années au risque de perdre beaucoup de temps sur des solutions qui ne sont pas vraiment pratiques/utiles.
Donc, c'est ça l'idée: sortir de nouveaux trucs dès que j'ai quelque chose de stable qui couvre les bases, puis étendre/peaufiner. Si mon expérience avec les projets Web m'a appris une chose, c'est que les internautes qui utilisent des applications savent suggérer ou demander des choses que les développeurs eux-mêmes n'auraient pas pensé à faire, et certaines de ces suggestions peuvent s'avérer de très bonnes idées, même si on ne capte pas toujours tout le potentiel au départ. Aussi, ça aide à établir un meilleur sens des priorités.

Du coup, c'est ce que j'ai fait récemment, chaque site ayant ses propres thèmes et ils sont tous disponibles en ligne (en Anglais et en Français):
flux.evasium.com (éducation, sciences et mondes virtuels)
notiflux.com (une exploration du contenu en ligne sur le développement personnel et le succès dans la vie)
kpakpatoya.com (catalogue de ce qui est nouveau, populaire et intéressant en ligne avec un accent particulier sur l'Afrique)
degammage.com (divertissement essentiellement geeky)
appuyage.com (exploration en psychologie, sexologie et relations interpersonnelles au niveau émotionnel)

Donc j'ai fait un jeu l'année précédente, je déploie ces portails web cette année et les mois à venir semblent prometteurs en termes d'améliorations techniques et de nouvelles fonctionnalités. On verra bien.

Reste informé sur tes sujets préférés via les plateformes sociales

Par Necemon

Ayant écrit Notiflux™ et AUDE (Automatic Updates Distribution Engine) il y a quelques mois, je me suis dit que je jouerais un peu avec les APIs de Facebook et Twitter, de façon à partager des articles relativement utiles, au fur et à mesure qu'ils sont publiés.

Donc, selon le genre de choses que tu aimes, tu voudrais peut être rejoindre ou suivre l'une de ces pages (ou plusieurs, pourquoi pas):

Degammage™ : actus de jeux vidéo, jeux web, jeux pour Windows.

->Facebook ->Twitter

Evasium Park France : L'actualité des mondes virtuels.

->Facebook ->Twitter

Notiflux France : Ton flux de notes et de notifications. L'essentiel des sources francophones pour les sujets qui concernent ton succès : organisation, stratégie, tactiques, philosophie, marketing, richesse, etc.

->Facebook ->Twitter

Evasium France ® : Articles et réflexions sur l'éducation à travers les technologies, apprendre tout en s'amusant.

->Facebook ->Twitter

Appuyage™ : l'essentiel séduction, rencontres, relations amoureuses, sexualité (pas forcément dans cet ordre)

->Facebook ->Twitter

Aussi quelques pages avec un accent particulier sur l'Afrique de l'Ouest:

Kpakpatoya™ : L'actualité locale et internationale

->Facebook ->Twitter

La Cour Commune™ : Nous construisons une plateforme virtuelle de rencontres et d'échanges. Mais pour l'instant, ce sera actus, showbiz, technologie, joie !

->Facebook ->Twitter

Babi Fraya™ : Partage d'articles touchant à l'art digital africain : jeux vidéo, illustration, musique, nouvelles, cinéma, photographie.

->Facebook ->Twitter

Fais ton choix !

N.

Furtivue (Ou Comment Envoyer Des Messages Furtifs Comme Des Ninjas)

Par Necemon

Furtivue : une vue furtive sur vos messages. (www.furtivue.com)

Furtivue est un service web qui vous permet d'envoyer des messages temporaires, autodestructibles. C'est a dire que vous pouvez contrôler combien de temps vos messages restent chez le destinataire.

Quand le destinataire ouvre votre message Furtivue, il apparaît pour un nombre spécifié de secondes, puis s'auto-détruit. La durée dépendra de la longueur du message (Ou vous pouvez choisir combien de temps vous voulez que le message reste affiché).

Comment cela fonctionne:

L'utilisateur A écrit un message à l'utilisateur B, un lien est envoyé à B. Lorsque B clique sur le lien, ce lien est désactivé (ne peut donc pas être réutilisé), le message est affiché à partir d'un tableau Silverlight et disparait après que le temps prédéfini soit terminé.

But:

Cela pourrait être utilisé pour l'envoi d'informations confidentielles / secrètes (telles que les mots de passe), ou toute autre donnée que vous ne voulez pas que le récepteur garde dans sa boîte e-mail : pour une raison quelconque, vous voudriez peut-être vous exprimer sans laisser une trace permanente.

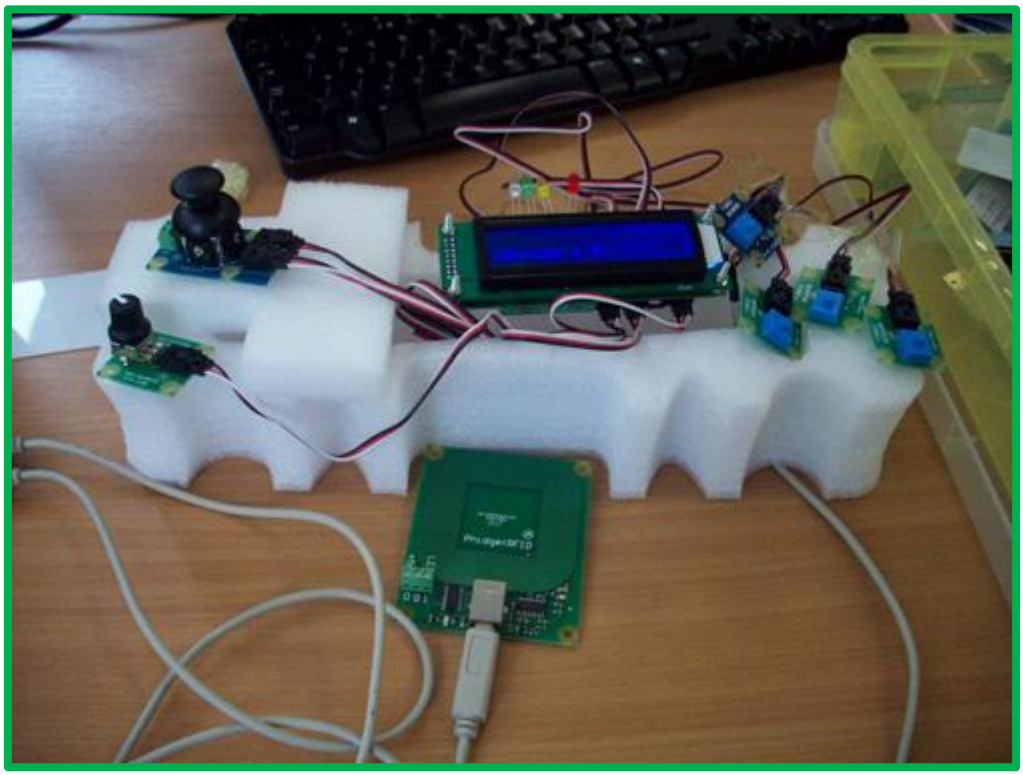
Et maintenant...

Vous pouvez l'essayer ici ou alors télécharger/explorer le code source à partir d'ici.

Chapitre 9

Recherches et Etudes de Cas

L'informatica è un'esperienza che ti arricchisce molto
Almeno stando a quello che dice il mio estratto conto !



Phidgets : Premiers Pas En Robotique ?

Par Necemon

Les phidgets sont des blocs de construction, des composants électroniques pas cher que vous pouvez contrôler à partir de votre ordinateur personnel via un port USB. Ils vous donnent quelques méthodes supplémentaires d'entrée/sortie de données au-delà de la combinaison classique souris + clavier + écran.

Comme Harold Thimbleby l'a mentionné dans son livre Press On, les phidgets sont un moyen très pratique de débuter dans la programmation hardware, au cas ou on veut construire des systèmes réels, et non pas des simulations Web ou sur écran: les phidgets sont appelés ainsi parce qu'ils sont l'équivalent physique des widgets de l'écran (gadgets Windows).
Phidgets = Physique + de Widgets
(Widgets = Windows + de Gadgets)

Où les trouver?

il y a différents distributeurs officiels phidgets répartis à travers le monde. Par exemple, pour ceux qui sont en France, vous pouvez consulter le site de RoboShop Europe.

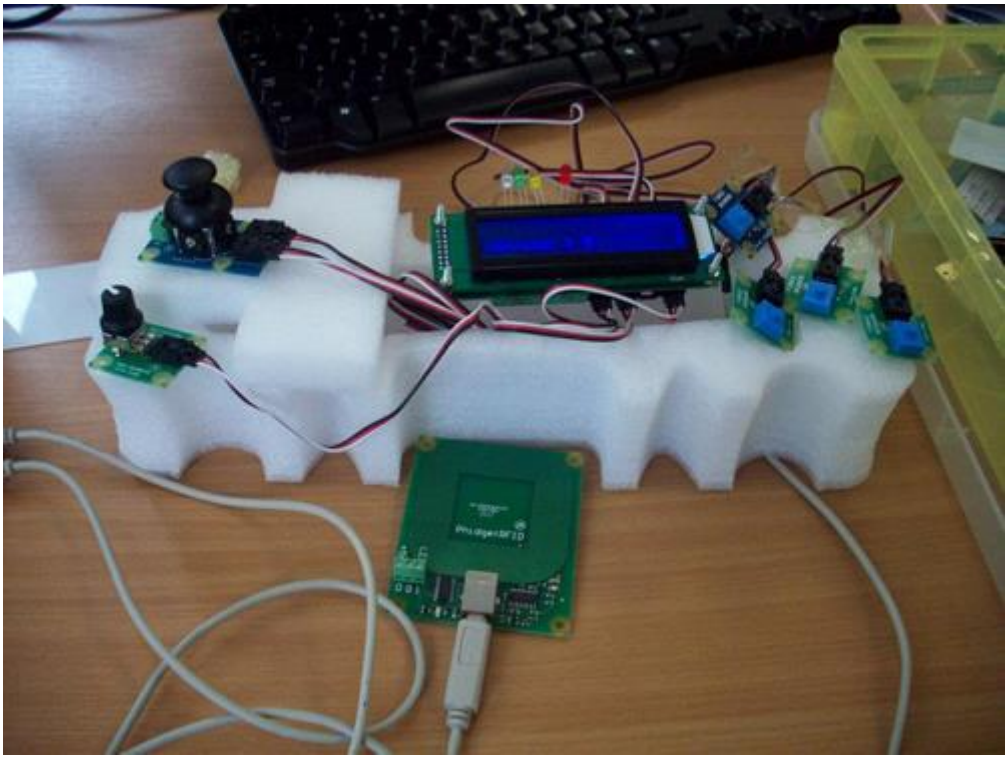
Commencer la manipulation/programmation de phidgets

Toute la complexité USB est gérée derrière l'API (Application Programming Interface). Les applications peuvent être développées rapidement par les programmeurs en utilisant leur langage préféré: C / C + +, C#, cacao, Delphi, Flash AS3, Flex AS3, Java, LabVIEW, MATLAB, Max / MSP, MRS, Python, REALBasic, Visual Basic.NET, Visual Basic 6.0, Visual Basic pour Applications, script Visual Basic, Visual C / C + + / Borland.NET, etc. (Si vous ne savez pas programmer, vous pouvez cependant utiliser un logiciel tel que Microsoft Robotics Studio ou même Microsoft Excel). Tout ce dont vous avez besoin, ce sont les pilotes que vous pouvez obtenir en téléchargeant l'installateur adéquat. De même, vous pouvez accéder aux différents manuels, aux échantillons et à l'API à partir d'ici.

Par exemple, en tant que développeur C#, il suffit de télécharger les ressources suivantes et vous êtes prêt à démarrer:
1. Guide de démarrage
2. L'installeur Windows
3. Les exemples de code

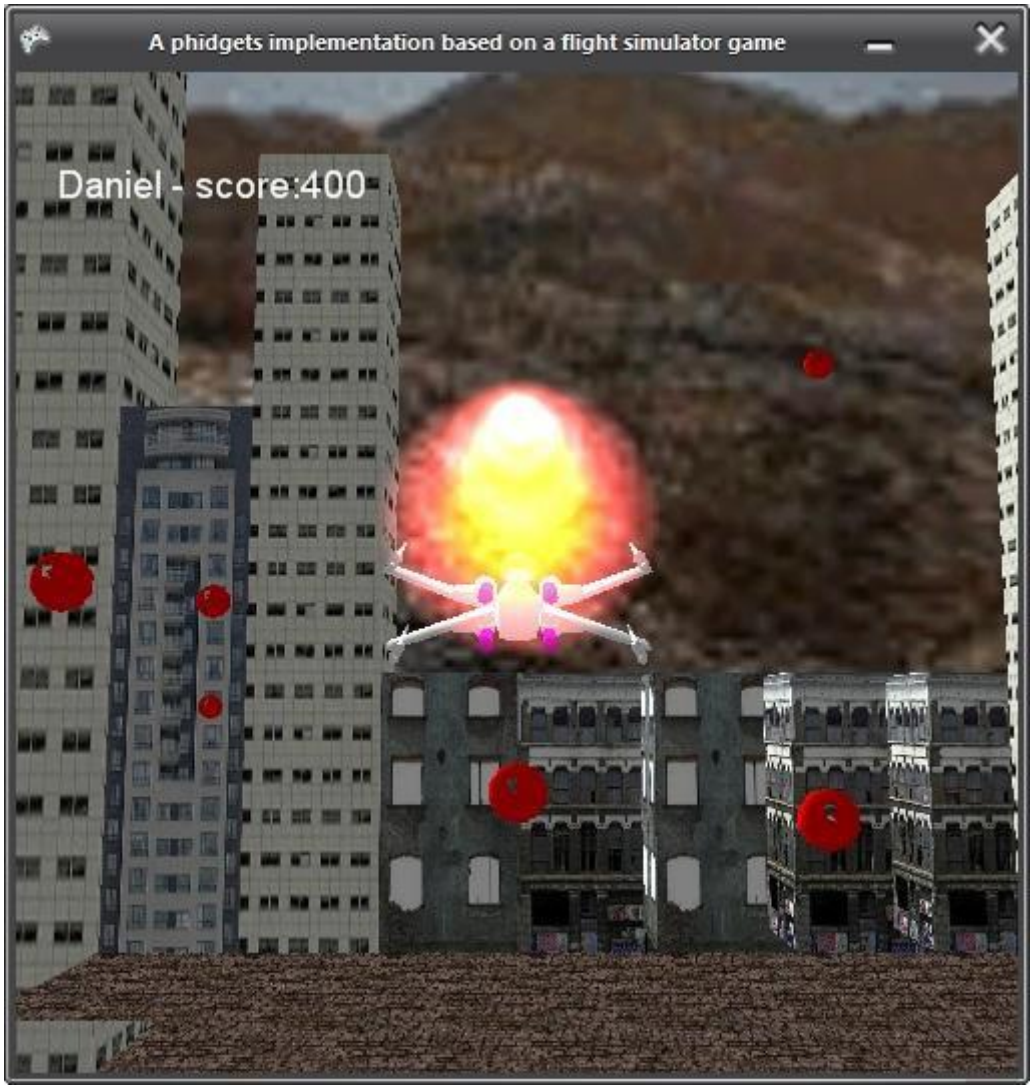
Un exemple pratique

Permettez-moi de vous présenter le Quick-&-Dirty-Pad (le temps que je pense à un meilleur nom), un "contrôleur de jeu" que j'ai construit avec mon partenaire Daniel Williams, alors que nous étions à l'Université, classe de technologies de l'interaction menée par le Docteur Parisa Eslambolchilar.



Oui je sais, ce n'est pas terrible, mais il faut bien commencer quelque part...

Nous l'avons utilisé principalement pour aller de pair avec un jeu de simulation de vol que nous avions programmé. Nous avons utilisé des capteurs de force pour le tir, un joystick pour l'orientation, des étiquettes RFID pour l'authentification d'utilisateur et un capteur de rotation pour réguler la vitesse du jeu. Les événements du jeu transparaissent à travers un écran LCD et quelques indicateurs LED.



(Merci d'avoir lu jusqu'ici. Le reste est pour les nerds seulement. C'est la partie où nous discutons brièvement les principaux détails techniques.)

Le jeu lui-même a été écrit en C#. Il est inspiré d'un tutoriel XNA de Riemers. Les éléments clés du code incluent le système de navigation:

```
                if (keys.IsKeyDown(Keys.Right))
    leftRightRot += turningSpeed;
if (keys.IsKeyDown(Keys.Left))
    leftRightRot -= turningSpeed;
float upDownRot = 0;
if (keys.IsKeyDown(Keys.Down))
    upDownRot += turningSpeed;
```

```
        if (keys.IsKeyDown(Keys.Up))

            upDownRot -= turningSpeed;
```

L'avion se déplace continuellement vers l'avant et le code ci-dessus est utilisé pour contrôler les mouvements de rotation. Un de nos défis a été d'intégrer le code Phidget pour le joystick dans cette partie du projet de telle sorte que le joystick se comporte comme les touches directionnelles.

Tout d'abord, nous avons créé un nouvel input wrapper, la classe qui contient le gestionnaire de capteurs pour détecter tout changement dans le mouvement de la manette. La classe input wrapper implémente l'objet interface kit qui détecte les entrées de données

```
InputWrapper iw = new InputWrapper();

ifKit.SensorChange += new SensorChangeEventHandler(ifKit_SensorChange);
```

La première section du code suivant contrôle l'axe X de la manette de commande. Les valeurs de sortie de la manette sont comprises entre 0 et 999, donc pour toute valeur supérieure à 500 le plan serait tourné vers la droite et toute valeur inférieure à 500 le plan serait tourné vers la gauche. La deuxième partie du code est utilisé pour contrôler l'axe Y sur la manette de commande. Cette fois-ci une valeur supérieure à 500 tournerait le plan de sorte qu'il s'oriente vers le haut tandis que toute valeur inférieure à 500 ferait tourner le plan de sorte qu'il s'oriente vers le bas.

```
                if(iw.XJoystick>550)

            leftRightRot = (iw.XJoystick-500)/10000;

        if (iw.XJoystick < 450)

            leftRightRot = (iw.XJoystick-500)/ 10000;

                if (iw.YJoystick > 550)

            upDownRot = (iw.YJoystick - 500)/10000;

        if (iw.YJoystick < 450)

            upDownRot = (iw.YJoystick - 500) / 10000;
```

L'écran LCD a été utilisé pour afficher le nom du joueur et son score. L'utilisateur doit scanner sa carte RFID pour démarrer le jeu. Nous avons écrit des wrappers LCD et RFID qui contiennent les différents gestionnaires d'événements. La classe RFID détecte si un tag est présent, sinon un message s'affiche sur l'écran LCD demandant à l'utilisateur de placer son tag

```
            TextLCDWrapper tlw = new TextLCDWrapper();

    RFIDWrapper rw = new RFIDWrapper();

            while (rw.tag == null || rw.tag.Length < 2)
```

```
        {
            tlw.JustTyped("Tag, Please!");
        }
```

Lorsque l'utilisateur scanne son étiquette, le capteur RFID permet de détecter si l'utilisateur est enregistré, et afficher le message correspondant sur l'écran LCD

```
private void DrawText()
    {
      if(rw.tag != null && rw.tag.EndsWith("cb"))
        {
            tlw.JustTyped("Necemon : " + score);
            spriteBatch.DrawString(font,"Necemon - score:" + score, newVector2(20, 45), Color.White);
        }
        else if (rw.tag != null && rw.tag.EndsWith("8a"))
        {
            tlw.JustTyped("Daniel : " + score);
            spriteBatch.DrawString(font,"Daniel - score:" + score, newVector2(20, 45), Color.White);
        }
        else if (rw.tag != null)
        {
            spriteBatch.DrawString(font, rw.tag + " - score:" + score, newVector2(20, 45), Color.White);
            tlw.JustTyped("Unknown : " + score);
        }
    }
```

Le détecteur de rotation est utilisé pour contrôler la vitesse de l'avion. Les valeurs du capteur de rotation sont obtenues par la classe input wrapper, comme mentionné précédemment dans la description de la manette. Plus vous tournez le capteur, plus la vitesse augmente

```
gameSpeed = iw.RotationValue/100;
```

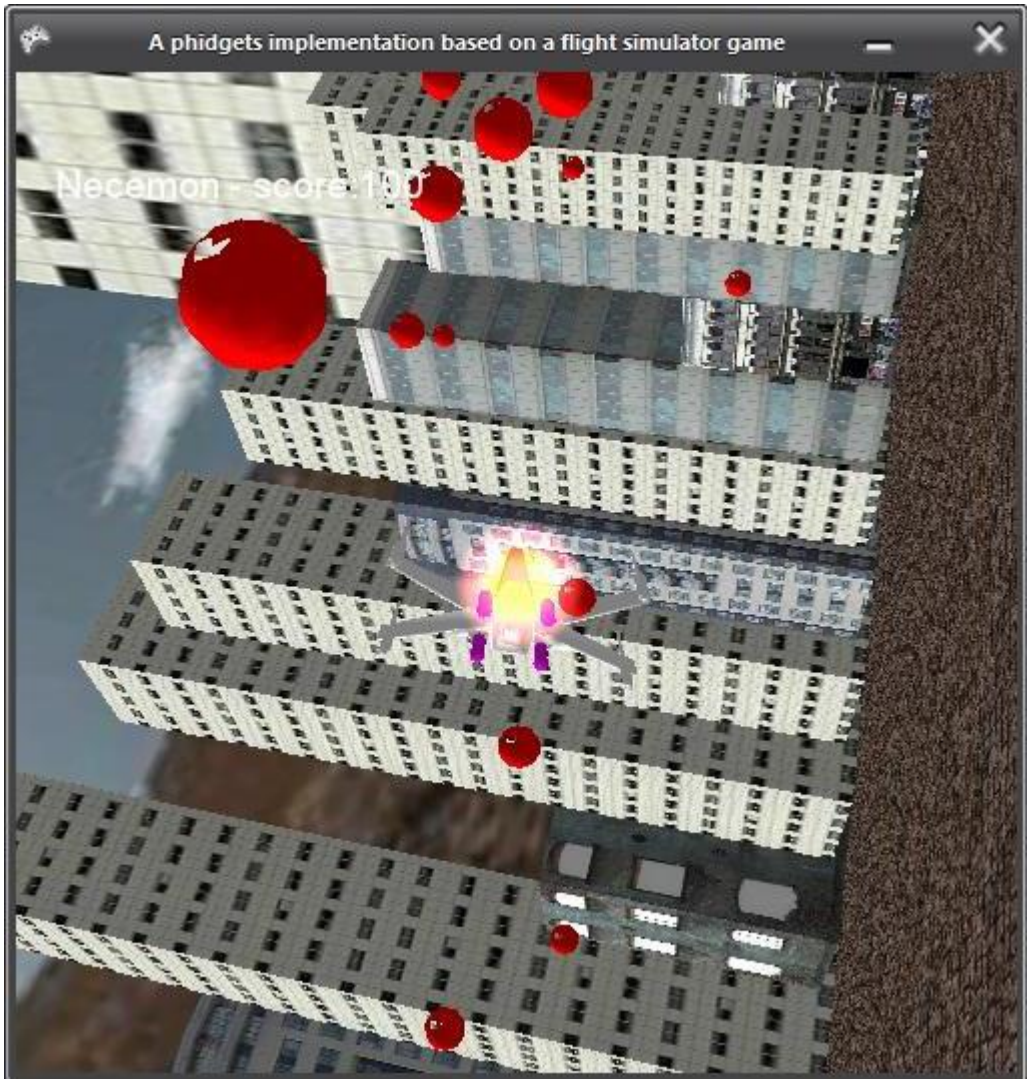
Le code suivant crée un nouveau projectile à chaque fois que la barre d'espace est enfoncée à condition que la balle précédente ait été tirée au moins 100 millisecondes plus tôt.

```
    if (keys.IsKeyDown(Keys.Space))
{
double currentTime = gameTime.TotalGameTime.TotalMilliseconds;
    if (currentTime - lastBulletTime > 100)
    {
        Bullet newBullet = new Bullet();
        newBullet.position = xwingPosition;
        newBullet.rotation = xwingRotation;
        bulletList.Add(newBullet);
        lastBulletTime = currentTime;
    }
}
```

Notre tâche consistait à mettre en œuvre quatre capteurs pour contrôler les projectiles étant tirés par l'avion. Les valeurs du capteur de force sont également obtenues via la classe input wrapper. Le code suivant crée maintenant une nouvelle balle de type 3 à chaque fois que les capteurs de force sont pressés

```
if (keys.IsKeyDown(Keys.C) || iw.Force3 > 25)
{
    //make the lights blink
    iw.ifKit.outputs[0] = (!iw.ifKit.outputs[0]);
    iw.ifKit.outputs[1] = (!iw.ifKit.outputs[1]);
    iw.ifKit.outputs[2] = (!iw.ifKit.outputs[2]);
    iw.ifKit.outputs[3] = (!iw.ifKit.outputs[3]);
    double currentTime = gameTime.TotalGameTime.TotalMilliseconds;
    //within a certain frequency
```

```
    if (currentTime - lastBulletTime > 300)
    {
        //Create the bullet and add it to the bulletlist
        Bullet newBullet = new Bullet();
        newBullet.position = xwingPosition;
        newBullet.rotation = xwingRotation;
        bulletList3.Add(newBullet);
        //play a sound
        soundEffect1.Play();
        lastBulletTime = currentTime;
    }
}
```


A phidgets implementation based on a flight simulator game
Necemon - score 150

Prototype Papier

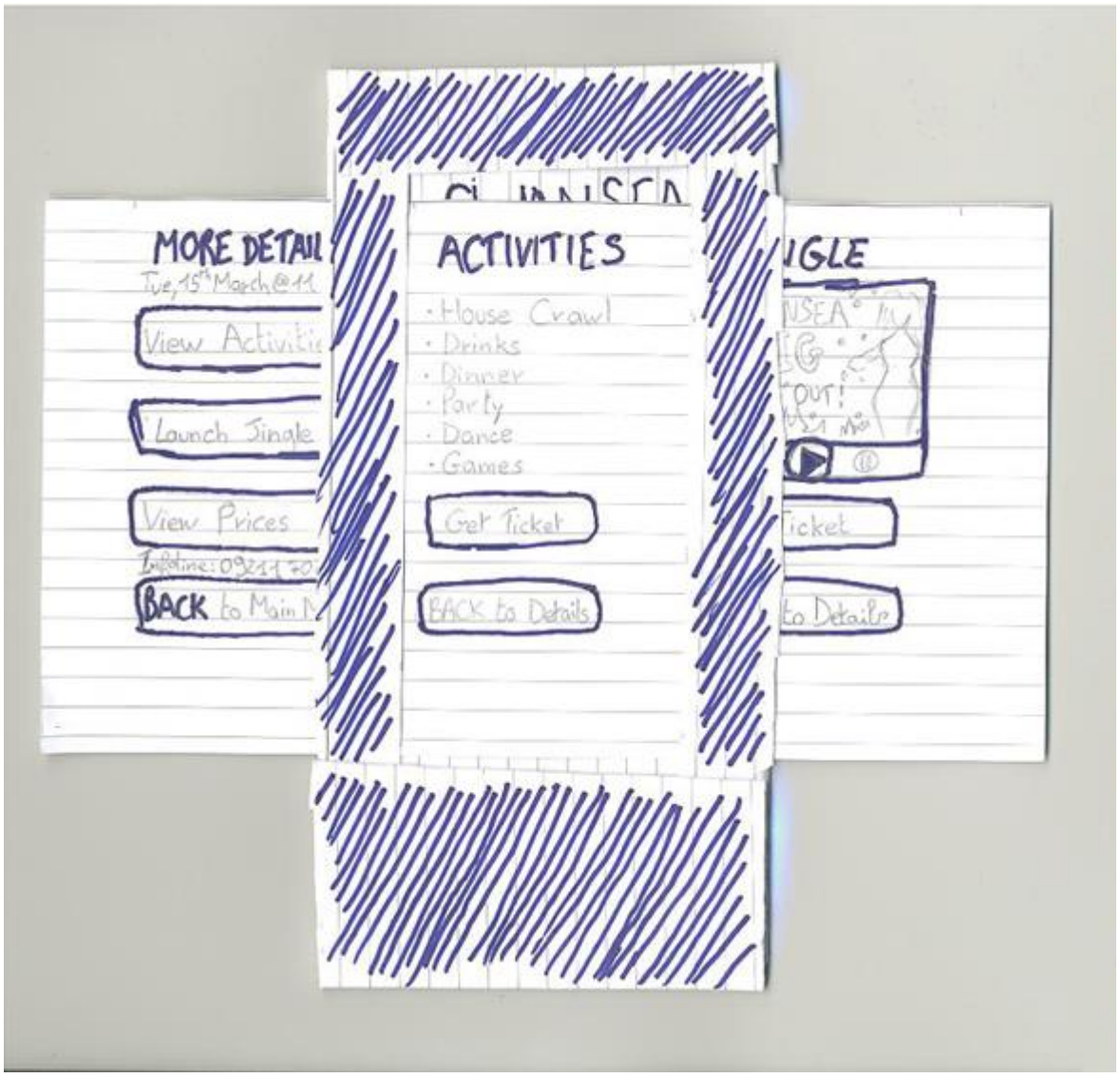
Par Necemon

La semaine dernière, j'ai eu à réaliser un prototype en papier pour une application mobile sur laquelle je travaillais. J'ai fini par construire quelque chose de pas trop mal (photo ci-dessus), mais au début je n'étais pas sûr de savoir comment m'y prendre. Donc j'ai pensé que je pourrais écrire une petite note à ce sujet.

La première chose était de faire un peu de recherche, juste pour voir comment les gens le font normalement. Le site paperprototyping.com avait été assez instructif par rapport à ça (ainsi que cette video) .

Ensuite, direction la papeterie pour collecter les trucs nécessaires. Pour l'ensemble du processus, j'ai juste utilisé un crayon, un feutre, un bâton de colle, quelques fiches et une paire de ciseaux. C'est tout.

Les prochaines étapes étaient de couper certaines des fiches à l'échelle d'un (gros) téléphone mobile, envelopper l'une d'elles autour des autres et faire une ouverture de chaque côté afin que les «écrans» puissent passer :



Tout cela peut sembler plutôt évident, mais je suppose que c'est le genre d'article que j'aurais aimé pouvoir lire quand je commençais.

Merci.

Dual Shock 3 - Partie 1: Historique des contrôleurs de jeu

Par Necemon, 2011

Cette étude porte sur un appareil qui est probablement la meilleure manette de tous les temps: le Dual Shock. Il a de nombreuses variantes et nous pouvons nous attendre à ce qu'il continue d'évoluer dans les prochaines années, cependant, nous allons nous concentrer ici sur la version actuelle, la Dual Shock 3.

Dans cet article, nous allons revoir l'histoire de la manette de jeu en général, et ensuite se concentrer sur l'histoire du Dual Shock lui-même, afin que nous comprenions comment il a été construit. Dans les prochaines parties, nous allons continuer avec le contexte dans lequel l'appareil a été construit et ensuite analyser la manette en étudiant ses spécifications.

Les bases

Il y a de nombreuses catégories de contrôleurs de jeu, parmi lesquelles nous retiendrons le plus populaire, le gamepad, vu que c'est exactement ce que la Dual Shock est. Le Gamepad, souvent appelé la manette de jeu, peut avoir des boutons d'action combinés avec un ou plusieurs joysticks omnidirectionnels ou d'autres boutons. Ils sont tenus avec les deux mains, avec les doigts (généralement les pouces) utilisés pour envoyer des commandes au jeu; comme mentionné sur Wikipédia, les contrôleurs de jeu modernes sont une variation du gamepad standard. Des ajouts communs incluent des boutons d'épaule placés le long des bords de l'appareil, des joysticks, des boutons placés au centre et de moteurs internes qui fournissent une rétroaction haptique (généralement, cela s'exprime par des vibrations).

Un peu d'histoire

Dans son article intitulé: "Histoire de la manette de jeu", Catalin Ivan a déclaré: « Depuis le début du jeu vidéo, la manette a été le meilleur (et souvent le seul) moyen d'interaction homme-machine ». C'est effectivement un fait intéressant, et il serait bon de suivre l'évolution depuis ce début pour voir comment nous avons atteint les technologies actuelles de gamepad. Beaucoup de gamepads ont été commercialisés à travers l'histoire, nous n'allons pas tous les étudier, nous allons nous contenter d'examiner ceux qui apparaissent comme des étapes importantes que nous avons observées à partir d'une étude par John Honnibal et que nous résumons dans le tableau suivant:

Nom du contrôleur	Description	Image
Le TV Tennis Game Paddle	- Un contrôle analogique basé sur un monostable - les touches de démarrage étaient situées sur la console, et non pas sur les manettes - Une variante était d'utiliser un curseur - Certains jeux n'avaient même pas de manettes avec des câbles , mais les touches étaient placées directement sur le jeu lui-même - Généralement, pour les premiers modèles, il n'y avait pas de boutons sur les pads	  

Le joystick Atari 2600	- Un joystick numérique à quatre directions avec une seule touche d'action - Encore une fois, la touche de démarrage du jeu était sur la console elle-même	
Le joystick Atari 7800	- Un joystick numérique à quatre directions avec deux boutons de tir - Il s'agissait juste d'ajouter un bouton supplémentaire au joystick 2600	
Le joystick Home Computer 8-bit	- Un joystick analogique avec un bouton de tir simple - Ici, le jeu pourrait être lancé en utilisant le clavier	
Le joystick Vectrex	- Un joystick analogique avec quatre boutons - Les jeux pouvaient être lancés par le contrôleur en utilisant les touches de tir normales - Les touches de tir sont étiquetées de '1 'à '4'	
Le joystick Nintendo Entertainment System	- Un joypad numérique avec deux boutons de démarrage et deux boutons de tir	

	- Introduit l'idée de placer des composants actifs et des circuits à l'intérieur du joypad. - La forme de la manette de jeu, cependant, est une boîte rectangulaire très basique. - Les deux boutons de tir sont tous les deux de couleur rouge et sont marqués 'A' et 'B'.	
Le joystick Super Nintendo Entertainment System	- Une manette de jeu numérique avec deux boutons de démarrage et six boutons de tir. - Introduit l'idée de boutons d'épaule - Forme beaucoup plus naturelle pour s'adapter aux mains du joueur.	
Le Joypad Sony PlayStation I	- Une manette de jeu numérique avec deux boutons de démarrage et huit boutons d'action. - Ajoute deux boutons d'épaule de plus que le pad SNES - Une forme encore plus ergonomique	
Le Joypad Sony PlayStation II	- Une mise à jour du contrôleur PlayStation I, qui ajoute des contrôles analogiques et une paire de moteurs de vibration. - Le convertisseur analogique - numérique est dans le contrôleur et l'interface est numérique. - Les deux moteurs sont équipés avec des poids excentriques de tailles différentes, et peut être contrôlée indépendamment par le jeu.	

De ce bref résumé, nous pouvons voir les différentes étapes de l'histoire du contrôleur de jeu avant l'avènement du Dual Shock. Les quelques déductions que nous pouvons faire sont tout d'abord que nous sommes lentement passés de contrôleurs avec zéro bouton vers les contrôleurs avec un, deux boutons et ainsi de suite. Cette évolution est devenue de plus en plus significative avec des couleurs et des étiquettes. Nous observons également que les commandes sont progressivement passées de la console à la manette. En outre, nous notons que les contrôleurs devenaient de plus en plus ergonomiques avec le temps.

Maintenant, nous allons revisiter l'histoire exceptionnelle de la DualShock.

Dual Shock 3 - Partie 2: L'histoire exceptionnelle de la Dual Shock

Par Necemon, 2011

Nous venons d'examiner l'évolution de la manette de jeu en général. Maintenant, nous allons plonger dans l'histoire plutôt percutante de la création de la manette de PlayStation. Les faits rapportés ici proviennent principalement de Kevin Gifford qui a écrit l'article "PlayStation 1 Design" et de Reiji Asakura qui a écrit le livre "Revolutionaries at Sony - The Making of the Sony PlayStation and the Visionaries Who Conquered the World of Video Games."

Pour bien entrer dans le contexte, permettez-moi de présenter quelques-unes des personnalités publiques japonaises qui sont clés de cette histoire. Tout d'abord, Ken Kutaragi, qui était président de Sony Computer Entertaiment. Puis nous avons aussi feu Norio Ohga (1930-2011) qui fut président de Sony Corporation au moment de l'histoire qui est arrivée. Et il y avait Teiyu Gotoh, le concepteur principal du projet PlayStation, qui est le héros de cette histoire. Il est également intéressant de mentionner que la PlayStation de Sony a été un énorme succès quand elle a été mise en vente. C'était en fait le modèle le plus vendu sur le marché des équipements électroniques (avec 40 millions d'unités dans le monde entier, sans changement de modèle en trois ans). «L'un des principaux facteurs qui ont conduit au succès de la Playstation», a déclaré Ohga, «a été la conception de l'unité principale et le contrôleur (la manette)».C'est ce que confirme Gotoh en déclarant «Je suis convaincu que le design a contribué à l'énorme succès de la PlayStation, nous avons dû partir de zéro, la Playstation n'ayant aucun antécédent. Un problème majeur était de savoir si les éditeurs de logiciels publieraient des titres pour elle. J'ai pensé que le premier obstacle était de s'assurer que les créateurs de jeu apprécieraient la conception physique. Après tout, les créateurs de jeu sont les premiers utilisateurs de la machine. Nous avons vu juste, s'ils se disent qu'ils veulent une plate-forme qui ressemble à la nôtre.»

Pour Gotoh, le design était l'expression même du contenu. Il visait un design simple mais originale et sans tendance. A l'origine, Gotoh était un concepteur de téléviseurs. Il était bien connu pour son style de design sophistiqué et d'excellentes performances sur des projets comme le Sony Profil PRO et le miracle de Sony, le PC VIAO. Cependant, le projet PlayStation était un problème complètement différent, comme Gotoh dit lui-même : « les téléviseurs ont une forme fixe et utilisent tous les mêmes composants. A travers ces contraintes, je me suis débrouillé pour trouver des moyens pour créer des designs innovants qui sont différents des autres produits, une façon de souligner la différence. Mais la console de jeu est complètement différente de produits conventionnels, parce que la PlayStation est un produit unique. Bien qu'il rivalise avec d'autres plateformes sur le marché du jeu, il n'y a pas d'autre Playstations. Les autres catégories de produit de Sony ont leur propre histoire et un processus de conception prédéfini, mais il n'y avait pas d'exemples à suivre avec la PlayStation, seulement l'idée de faire quelque chose de différent des autres produits du même genre ».

Dans notre étude, nous sommes plus concernés par la manette de PlayStation que la console elle-même. Mais la partie intéressante est que Gotoh conçut non seulement les parties principales de la PlayStation, mais aussi de ses périphériques tels que la carte mémoire et la manette. Cependant, au début, la conception de la manette n'a pas connu une approbation unanime chez Sony. Tandis que Ohga l'a approuvée, Kutaragi, le technologue, l'a immédiatement rejetée: "Qu'est-ce que c'est que ça ? La forme est originale, mais il ne semble pas très facile à utiliser". Elle était en effet très différente des manettes classiques qui étaient plates, alors que la manette de Gotoh était tridimensionnelle. Il se souvient: "J'ai travaillé chez Sony pendant vingt et un ans maintenant, mais il n'y a jamais eu de produit aussi difficile. La manette a été considérablement plus difficile à concevoir que la console. " Il lui a fallu plus d'un an pour le concevoir. Il commença par sculpter des morceaux de mousse acrylique à plusieurs reprises pour sentir l'effet dans les mains. La forme de la manette a été affinée grâce à d'innombrables répétitions de ce processus. Un inconvénient des manettes plates était qu'elles devaient être fermement maintenue lors de l'utilisation vu que la paume n'était pas en contact avec l'appareil. Ceci tendait à être inconfortable et stressant. En outre, ces manettes ne pouvaient pas s'adapter correctement aux différents types et tailles de mains. Voilà comment Gotoh a ajouter des poignées de telle sorte que le contrôleur puisse être tenu aussi naturellement que possible.

À un certain point, Gotoh a mené une expérience, en rassemblant quelques enfants pour leur faire tester sa manette. Ils ont trouvé ça un peu bizarre au début car ils étaient habitués aux manettes plates. Cependant, une fois qu'ils ont appris comment l'utiliser, ils semblaient vraiment apprécier. Cela était encourageant, car les enfants sont honnêtes dans leurs commentaires. S'ils aiment une chose, ou s'ils n'aiment pas, ils disent juste comment ils se sentent vraiment par rapport à la chose. De plus, il y avait un autre point très intéressant dans cette expérience. Comme disait Harold Thimbleby, les enfants sont de meilleurs testeurs pour les appareils que les adultes, car ils ont tendance à se comporter comme des lutins. Ils n'hésitent pas à essayer de nouvelles choses sans craindre d'abimer l'appareil. Aussi, le fait qu'ils soient aptes à se débarrasser de toute idée préconçue sur la technologie leur permet d'éviter toute hypothèse de conception. Cela fait d'eux les testeurs parfaits pour un appareil révolutionnaire, comme la manette de PlayStation, le fer de lance d'une nouvelle génération de manettes de jeu.

Dual Shock 3 - Part 3 : Analyse et Spécification

Par Necemon, 2011

Maintenant que nous avons compris le contexte et les motivations qui ont conduit à la création de la Dual Shock, nous allons apprendre à connaître davantage l'appareil lui-même.

Les dimensions de la manette sont d'environ 6 x 3,5 x 2 pouces.

La Dual Shock a sur le côté gauche, le même pad directionnel trouvé sur la plupart des contrôleurs. Toutefois, sur la droite, les touches d'action sont étiquetées avec les symboles originaux, ce qui est également une innovation par Gotoh, tandis que les contrôleurs classiques utilisaient normalement des chiffres et des lettres pour marquer les boutons, comme il l'affirme lui-même: "A l'époque, les autres compagnies de jeu vidéo attribuaient des lettres de l'alphabet ou des couleurs aux boutons. nous voulions quelque chose de simple à retenir, c'est pourquoi nous avons choisi des icônes ou des symboles, et immédiatement après j'ai proposé la combinaison Triangle-Cercle-X-Carré". Dans une interview rapportée par Radar Jeux, Gotoh a expliqué comment il en est arrivé à ces symboles "J'ai donné à chaque symbole une signification et une couleur, le triangle fait référence à la vue... Je voulais qu'il représente une tête ou une direction et je l'ai fait vert. Le carré est une référence à un morceau de papier, je voulais qu'il représente les menus ou les documents et je l'ai fait rose. Le cercle et le X représentent les prises de décision, "oui" ou "non" et je les ai faits rouge et bleu, respectivement".

Additionnellement, il y a aussi quatre boutons sur le dessus, pour les doigts du milieu et deux joysticks qui permettent d'améliorer l'expérience de certaines catégories de jeu comme la course ou la simulation de vol.

La vibration est un effet intéressant vu que le périphérique "gronde" quand certains événements dramatiques se produisent pendant le jeu. Ceci améliore l'expérience de jeu.

Toutefois, les joysticks et les effets haptiques ne figuraient pas dans la version originale de la manette PlayStation. Pour clarifier, il convient de mentionner que le Dual Shock est en fait une version avancée de cette version originale.

Maintenant parlant à la dernière version, celle qui nous intéresse le plus ici, la Dual Shock 3, qui elle-même est une version avancée de la Sixaxis,le contrôleur de jeu original pour la Playstation 3 de Sony. Les deux appareils ont la même apparence, mais ils sont différents dans la mesure où le Dual Shock 3 est plus lourd, plus opaque et il vibre.

Même si le Dual Shock 3 peut être branché à la console avec un câble USB amovible, c'est aussi un appareil sans fil qui peut fonctionner sur une connexion Bluetooth. Une autonomie de 30 heures est assurée par la batterie interne. Il semble que la connexion sans-fil pourrait fonctionner sur une distance de plus de 20 mètres.

Ergonomiquement, toutes les manettes Dual Shock sont à peu près les mêmes. Et honnêtement, ce sont les contrôleurs de jeu les plus confortables que j'ai personnellement eu la chance de manier (j'avais essayé une large gamme de contrôleurs au cours des années). Cependant, un inconvénient du Dual Shock 3 pourrait être qu'il est plus léger que ses prédécesseurs. Mais là encore, le joueur s'habitue après quelques temps.

Mettons la Dual Shock 3 en perspective avec les concurrents

Ceci est le contrôleur pour la Xbox 360; c'est celui que nous avons choisi pour notre comparaison parce qu'ils appartiennent à la même génération de contrôleurs, la PlayStation 3 de Sony et la Xbox 360 de Microsoft sont parmi les consoles de jeux qui sont d'actualité à travers le monde et ils ont tous deux à peu près les mêmes performances.

Physiquement, la manette Xbox 360 semble être plus grande et est bien connue généralement pour être un appareil pour les grandes mains (adultes), mais ils ont tous deux presque les mêmes dimensions. La couleur par défaut de la manette Xbox 360 est le blanc tandis que la couleur par défaut pour la Dual Shock 3 est le noir. Toutefois, les deux contrôleurs, ont été mis sur le marché avec une gamme de couleurs différentes afin que chacun puisse choisir selon ses goûts.

Techniquement, la Xbox 360 propose une option intéressante quand il s'agit de batteries. Ils utilisent des piles AA, qui peuvent donc être facilement changées. A l'inverse, la batterie interne du Dual Shock ne peut pas être changée (pas par le joueur, en tout cas). Mais quand il s'agit de câbles, la Dual Shock 3 utilise des câbles standard USB alors que le contrôleur de la 360 nécessite un câble spécifique qui est unique à ce contrôleur. Tous ces facteurs influent sur le coût global de l'appareil du point de vue du joueur. Cependant, la Dual Shock 3 est plus cher sur le marché que le contrôleur 360 (£39 contre £29 au moment où j'ai acheté le mien).

Mais quand il s'agit de la jouabilité, chaque contrôleur a ses propres forces et faiblesses. Mike Ferro a déjà comparé la jouabilité de contrôleurs basés sur les combats, les courses et le tir catégories. Si l'on considère la course, par exemple, la différence clé est dans les boutons d'épaule que les coureurs utilisent normalement pour décélérer. Ils sont beaucoup plus confortables sur le contrôleur 360 vu qu'ils montrent moins de résistance.

Pour les jeux de tir, les contrôles critiques sont les joysticks. Le stick analogique du Dual Shock 3 a un point mort légèrement plus petit, ce qui offre une plus grande précision et un meilleur temps de réponse pour le joueur. Pour en venir aux jeux de combat, l'utilisation du pavé directionnel est essentielle pour permettre aux personnages d'exécuter des mouvements précis. Le Dual Shock 3 est nettement supérieur pour cette affaire en raison de la façon dont les touches sont partitionnés. Au contraire, les mouvements sont un peu plus difficiles à exécuter sur le contrôleur 360 parce que l'utilisateur sent littéralement qu'il essaie de déplacer un seul morceau de plastique, et non des touches indépendantes haut, bas, droit et gauche.

GOOSE (moteur de ravitaillement Google)

Par Necemon

Google GOOSE est un produit que j'ai proposé lors d'un exercice de marketing. On nous avait demandé de penser à un nouveau produit d'une marque bien établie qui pourrait combler un besoin des clients. Merci de noter que le produit GOOSE est purement fictif. Je ne travaille pas à Google et la description du produit qui suit n'est pas approuvée par Google ou un de ses employés.

Le problème

Le shopping est devenu un processus coûteux en temps. Le rythme de nos vies, emplois et autres activités va tellement vite que nous ne trouvons guère le temps de faire des courses correctement. Les technologies actuelles nous fournissent des solutions pour accélérer le processus par des achats en ligne. Mais même en ligne, le shopping reste une tâche qui implique quelques efforts, de réflexion et de temps pour chaque achat. Parfois, vous n'avez simplement pas envie de le faire, ou vous avez beaucoup d'autres choses à faire. De plus, nous sommes parfois tellement occupés que nous pouvons simplement oublier de le faire.

Une autre chose est que cela pourrait être pénible. Bien sûr, il est souvent intéressant de sortir s'acheter des chaussures et des vêtements neufs. Mais ce n'est pas si amusant d'aller chercher des légumes et de la viande chaque semaine.

La solution

Et si vous pouviez simplement décider une fois ce qui devrait être là à tout moment dans votre réfrigérateur et ne plus vous en soucier? Vous pourriez juste avoir ce que vous voulez à tout moment sans avoir à appliquer régulièrement des efforts, du temps et de la réflexion (ce que les achats d'aliments nécessitent). Et c'est ce que le nouveau Google GOOSE propose.

Le GOOSE (GOOgle Supply Search Engine, Moteur de Ravitaillement Google) est une plateforme logicielle pour une nouvelle technologie de réfrigérateurs de maisons qui systématiquement:

- Sait quels sont les aliments dont vous avez besoin

- Sait quand ils sont absents de votre réfrigérateur

- Vérifie les meilleures offres pour ces articles sur internet

- Commande les aliments manquants en ligne

- Active la livraison de ces articles à votre réfrigérateur (en fait à votre adresse)

Comment ça fonctionne

-> Du côté serveur:

GOOSE se compose d'un service Web où chaque client peut enregistrer et fournir une liste des éléments dont il a besoin à tout moment. Le client peut également définir des éléments spéciaux qui seront livrées que dans des circonstances particulières (par exemple, un gâteau d'anniversaire à livrer une seule fois à une date donnée). Toutes ces informations sont stockées dans la base de données GOOSE.

Le client peut également définir la fréquence sur laquelle il veut que son réfrigérateur soit comblé, disons tous les vendredis après-midi. Alors, quand vient le moment, le service web GOOSE commande les éléments manquants qui avaient été communiquées à la boutique, que le client a indiqué. Mais une chose intéressante est que le client peut donner la liberté au service de choisir à partir de quel magasin il faut acheter. Evidemment cela ne devrait pas être trop loin du domicile du client, mais l'intérêt est que le service GOOSE permet de comparer les offres des différents magasins et choisir la meilleure offre. Les marchandises sont ensuite livrées à la maison du client.

-> Du côté client:

Le logiciel GOOSE fonctionne sur un nouveau modèle de système de réfrigérateur de maison qui contient certains composants spéciaux:

- Un écran tactile pour que l'utilisateur puisse mettre ses options en place. L'écran permet également a l'utilisateur de voir et de choisir les offres promotionnelles que GOOSE lui envoie (Au fait, cela apporte des revenus supplémentaires, car les annonceurs peuvent payer pour montrer des publicités aux utilisateurs).

- En plus, toutes les commandes peuvent également être entrées à partir d'un ordinateur normal qui est connecté sur le compte dudit client Goose. Toutes les modifications apportées seront mises à jour pour la base de données Goose et communiquées au réfrigérateur et vice-versa.

- Le réfrigérateur est équipé avec certains périphériques qui traque le contenu du frigidaire et les entrées et sorties des produits alimentaires. Ces dispositifs sont principalement les caméras, les infrarouges et de capteurs. Les codes-barres peuvent également être utilisés pour identifier les éléments complexes.

Plus de détails

Le logiciel aura à interagir avec l'Internet pour les achats en ligne et la reconnaissance de contenu dans une certaine mesure. Cela signifie que le réfrigérateur devrait avoir accès à une connexion Wi-Fi. L'intérieur du réfrigérateur devrait être faite dans une matière spécifique pour la reconnaissance de contenu. Par exemple, le plateau à oeufs, les compartiments pour le pain, les légumes et les fruits ont besoin de s'adapter à un petit détecteur qui pourrait reconnaître si l'élément est présent ou non. Par exemple, un faisceau de lumière peut être coupé quand un produit particulier est présent. Toutefois, ce ne serait pas résoudre complètement le problème vu que le logiciel aurait à reconnaître quel type d'élément est présent. Par exemple si on prend le cas des fruits et légumes, le logiciel devra savoir de quel fruit ou légume il s'agit, cela pourrait être fait en ayant un appareil photo dans le réfrigérateur, qui est aussi capable de compter le nombre de fruits ou de légumes. Un lecteur de code-barres pourrait être utilisé pour les articles qui utilisent cette technologie. Les utilisations de lecteurs de codes-barres et de caméras peuvent être combinées: Une fois que la camera reconnaît le code-barres, il est ensuite envoyé en ligne et vérifié dans des bases de données en ligne. La technologie marche de la même manière pour l'application de lecteur de code-barres sur Smartphone.

Maintenant que nous avons vu la manière dont le contenu est reconnu, que faisons-nous avec cette information? En fonction de ces données une liste de courses est générée et stockée temporairement jusqu'au prochain achat. La liste des achats pourrait également être classée en fonction de différentes sections telles que

-> Les articles les plus demandés, comme les boissons gazeuses ou tout élément que le client serait désireux d'avoir à tout moment.

-> Les éléments moins importants, les éléments que le client ne voudrait pas particulièrement avoir en tout temps.

-> Les articles à durée de vie courte (comme le lait). Le logiciel afficherait la date à laquelle le dernier achat a été fait et vous donnerait la possibilité d'ajouter le nombre de jours avant expiration du produit.

Le logiciel serait également capable de vous rappeler avant le jour de l'expiration que l'élément doit être consommée.

Lorsque le logiciel doit être utilisé pour la première fois une liste de courses générique est créée indépendamment des catégories. GOOSE, en gardant chaque information concernant les achats passés, pourrait apprendre progressivement quel type de client vous êtes et pourrait vous proposer une liste d'achats fondée sur la catégorisation automatique que vous pourriez accepter ou refuser.

Base de clients potentiels

Goose est un produit logiciel qui recherche et commande les produits alimentaires en fonction des précédents achats qui avaient été rangés dans le réfrigérateur. Il opère lors de sessions planifiées a des dates choisies, afin que les gens qui travaillent puissent l'utiliser à leur convenance. En outre, GOOSE cible principalement :

- Les jeunes, qui sont les plus familiarisés avec la technologie Internet et les achats en ligne. Il serait parfaitement adapté aux étudiants qui se déplacent loin de leurs parents et ne sont pas habitués aux achats de nourriture.

- Les parents, qui sont responsables des approvisionnements alimentaires de la famille et ont parfois du mal à gérer cela (en particulier les parents seuls).

- Les pays développés: les lieux avec une pénétration d'Internet haute et la livraison à domicile répandu.

- Aussi, GOOSE pourrait mettre tout le monde d'accord, les hommes autant que les femmes. les hommes sont généralement les moins à l'aise quand il s'agit d'achats de nourriture et de cuisine. Inversement, les femmes qui ont souvent tout le fardeau de ces tâches seraient soulagées dans une large mesure.

Notre produit GOOSE ne commande pas le produit dans le seul magasin, il compare le produit dans les nombreux magasins disponibles sur le marché et commande automatiquement le produit le moins cher ou le meilleur. Toutefois, certains magasins peuvent être promus en payant certains frais de publicité. Ce serait la deuxième forme de profit pour GOOSE après les frais de licenciement du logiciel.

Défis

Le premier défi rencontré dans la mise en œuvre serait pour Google de trouver un partenaire qui fabrique les réfrigérateurs et qui mettrait en place le système GOOSE, vu que Google n'est pas vraiment dans l'industrie du hardware. Il devrait être assez possible de collaborer avec un des fabricants majeurs de réfrigérateurs (comme LG, Samsung, etc.) grâce à la notoriété de Google en tant que marque.

Un autre défi serait la pénétration du marché. Un grand nombre d'études de marché ont besoin d'être fait pour savoir comment construire un produit satisfaisant. Et une fois que le produit est prêt, il aura besoin d'une promotion adéquate, ce qui n'est pas vraiment un problème, Google est le roi de la publicité en ligne après tout.

Rappel Final

Avant de conclure, je voulais juste te rappeler que ce document contient le Volume 2 de L'Album et que le Volume 1 est aussi disponible. Tu peux télécharger, (re)lire et partager chaque chapitre ou volume indépendamment/séparément, en fonction de tes intérêts. Les formats PDF, EPUB, MOBI/AZW3/KF8 (Amazon Kindle) et MP3 sont disponibles.

Il suffit de cliquer sur le(s) document(s) qui t'intéresse(nt) ci-dessous ou d'aller sur Album.NeceMoon.com (ou necemonyai.com/blog/page/L-Album.aspx).

The NeceMoon, L'Album Complet

Volume 1 : Clair de Lune (softcore)

Volume 2 : Pleine Lune (hardcore)

L'Album est également disponible en Anglais à l'adresse TheAlbum.NeceMoon.com (ou necemonyai.com/blog/page/The-Album.aspx).

Conclusion

C'est le moment de se quitter, mais on peut rester en contact. N'hésite pas à m'ajouter sur LinkedIn, Twitter ou Facebook. Si toi aussi, tu as des outils ou des techniques pour accomplir les choses efficacement, je voudrais bien que tu m'en parles. Mon adresse email, c'est necemon@gmail.com. N'hésite pas à m'écrire pour signaler d'éventuelles erreurs dans ce document, pour suggérer des améliorations ou pour me faire part de tes passages préférés. Par contre, si tu n'aimes pas L'Album, ce n'est pas la peine de m'écrire.

Dans tous les cas, bon courage pour la suite, et merci d'avoir lu. Et merci à Wikipédia, MSDN, IconFinder et FreeDigitalPhotos pour les clarifications et les ressources infographiques. Remerciements spéciaux à la Team Evasium. Merci à tous ceux et celles qui ont contribué, merci Ahou l'Africaine, Antoine Mian, Cyriac Gbogou, Darren Mart, Edith Brou, Holty Sow, Israël Yoroba, Jean Luc Houédanou, Jean-Patrick Ehouman, Karen Kakou, Nanda Seye, Nnenna Nwakanma, Olivier Madiba, Vanessa Lecosson et Yehni Djidji.
Merci Monty Oum, repose en paix.

Partage L'Album avec Tes Potes

Clique simplement sur l'icône qui te convient ci-dessous. Ça ne prend que quelques secondes et c'est gratuit pour tout le monde. Alternativement, tu peux partager ce lien sur toute Plateforme Web/Sociale ou l'envoyer par e-mail à tes contacts qui pourraient en bénéficier : http://Album.NeceMoon.com