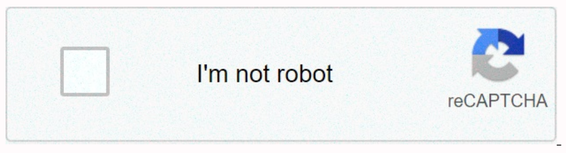
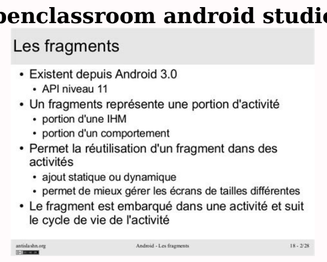


Continue

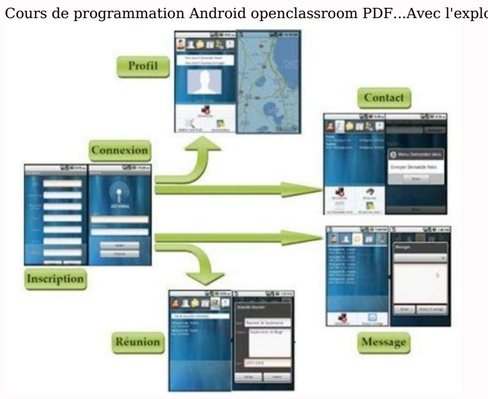
Developpement android openclassroom pdf

Openclassroom android studio.



Android openclassroom.

Cours de programmation Android openclassroom PDF...Avec l'explosion des ventes de smartphones ces dernières années, Android a pris une place importante dans la vie quotidienne. Ce système d'exploitation permet d'installer des applications de toutes sortes : jeux, bureautique, multimédia, etc.



Que diriez-vous de développer vos propres applications pour Android, en les proposant au monde entier via le Play Store, le marché d'applications de Google ? Eh bien figurez-vous que c'est justement le but de ce cours : vous apprendre à créer des applications pour Android !Cependant, pour suivre ce cours, il vous faudra quelques connaissances :Les applications Android étant presque essentiellement codées en Java, il vous faut connaître ce langage. Heureusement, le Site du Zéro propose un cours et même un livre sur le Java.Connaître un minimum de SQL pour les requêtes (ça tombe bien, le Site du Zéro propose un cours sur MySQL). Si vous ne connaissez absolument rien en SQL, vous pourrez tout de même suivre le cours dans son intégralité, mais constituer votre propre base de données sans théorie me semble risqué.Et enfin, être un minimum autonome en informatique : vous devez par exemple être capables d'installer Eclipse tout seul (vous voyez, je ne vous demande pas la lune).Rien de bien méchant, comme vous pouvez le voir. Mais le développement pour Android est déjà assez complet comme cela, ce serait bien trop long de revenir sur ces bases-là. Ce cours débutera cependant en douceur et vous présentera d'abord les bases essentielles pour le développement Android afin que vous puissiez effectuer des applications simples et compatibles avec la majorité des terminaux.

Puis nous verrons tout ce que vous avez besoin de savoir afin de créer de belles interfaces graphiques ; et enfin on abordera des notions plus avancées afin d'exploiter les multiples facettes que présente Android, dont les différentes bibliothèques de fonctions permettant de mettre à profit les capacités matérielles des appareils. À la fin de ce cours, vous serez capables de réaliser des jeux, des applications de géolocalisation, un navigateur Web, des applications sociales, et j'en passe. En fait, le seul frein sera votre imagination !Partie 1 : Les bases indispensables à toute applicationL'univers AndroidDans ce tout premier chapitre, je vais vous présenter ce que j'appelle l'« univers Android » ! Le système, dans sa genèse, part d'une idée de base simple, et très vite son succès fut tel qu'il a su devenir indispensable pour certains constructeurs et utilisateurs, en particulier dans la sphère de la téléphonie mobile. Nous allons rapidement revenir sur cette aventure et sur la philosophie d'Android, puis je rappellerai les bases de la programmation en Java, pour ceux qui auraient besoin d'une petite piqûre de rappel...La création d'AndroidQuand on pense à Android, on pense immédiatement à Google, et pourtant il faut savoir que cette multinationale n'est pas à l'initiative du projet. D'ailleurs, elle n'est même pas la seule à contribuer à plein temps à son évolution. À l'origine, « Android » était le nom d'une PME américaine, créée en 2003 puis rachetée par Google en 2005, qui avait la ferme intention de s'introduire sur le marché des produits mobiles. La gageure, derrière Android, était de développer un système d'exploitation mobile plus intelligent, qui ne se contenterait pas uniquement de permettre d'envoyer des SMS et transmettre des appels, mais qui devait permettre à l'utilisateur d'interagir avec son environnement (notamment avec son emplacement géographique). C'est pourquoi, contrairement à une croyance populaire, il n'est pas possible de dire qu'Android est une réponse de Google à l'iPhone d'Apple, puisque l'existence de ce dernier n'a été révélée que deux années plus tard.C'est en 2007 que la situation prit une autre tournure. À cette époque, chaque constructeur équipait son téléphone d'un système d'exploitation propriétaire. Chaque téléphone avait ainsi un système plus ou moins différent. Ce système entravait la possibilité de développer facilement des applications qui s'adapteraient à tous les téléphones, puisque la base était complètement différente. Un développeur était plutôt spécialisé dans un système particulier et il devait se contenter de langages de bas niveaux comme le C ou le C++. De plus, les constructeurs faisaient en sorte de livrer des bibliothèques de développement très réduites de manière à dissimuler leurs secrets de fabrication. En janvier 2007, Apple dévoilait l'iPhone, un téléphone tout simplement révolutionnaire pour l'époque. L'annonce est un désastre pour les autres constructeurs, qui doivent s'aligner sur cette nouvelle concurrence.

Le problème étant que pour atteindre le niveau d'iOS (iPhone OS), il aurait fallu des années de recherche et développement à chaque constructeur...C'est pourquoi est créée en novembre de l'année 2007 l'Open Handset Alliance (que j'appellerai désormais par son sigle OHA), et qui comptait à sa création 35 entreprises évoluant dans l'univers du mobile, dont Google. Cette alliance a pour but de développer un système open source (c'est-à-dire dont les sources sont disponibles librement sur internet) pour l'exploitation sur mobile et ainsi concurrencer les systèmes propriétaires, par exemple Windows Mobile et iOS. Cette alliance a pour logiciel vedette Android, mais il ne s'agit pas de sa seule activité. L'OHA compte à l'heure actuelle 80 membres.…Depuis sa création, la popularité d'Android a toujours été croissante. C'est au quatrième trimestre 2010 qu'Android devient le système d'exploitation mobile le plus utilisé au monde, devançant Symbian (le système d'exploitation de Nokia avant qu'ils optent pour Windows Phone). Désormais, on le retrouve non seulement dans les tablettes et smartphones, mais aussi dans les téléviseurs, les consoles de jeux, les appareils photos, etc.La philosophie et les avantages d'AndroidOpen sourceLe contrat de licence pour Android respecte les principes de l'open source, c'est-à-dire que vous pouvez à tout moment télécharger les sources et les modifier selon vos goûts ! Bon, je ne vous le recommande vraiment pas, à moins que vous sachiez ce que vous faites… Notez au passage qu'Android utilise des bibliothèques open source puissantes, comme par exemple SQLite pour les bases de données et OpenGL pour la gestion d'images 2D et 3D.Gratuit (ou presque)Android est gratuit, autant pour vous que pour les constructeurs. S'il vous prenait l'envie de produire votre propre téléphone sous Android, alors vous n'auriez même pas à ouvrir votre porte-monnaie (mais bon courage pour tout le travail à fournir !). En revanche, pour poster vos applications sur le Play Store, il vous en coûtera la modique somme de 25$. Ces 25$ permettent de publier autant d'applications que vous le souhaitez, à vie !Facile à développerToutes les API mises à disposition facilitent et accélèrent grandement le travail. Ces APIs sont très complètes et très faciles d'accès. De manière un peu caricaturale, on peut dire que vous pouvez envoyer un SMS en seulement deux lignes de code (concrètement, il y a un peu d'enrobage autour de ce code, mais pas tellement).Une API, ou « interface de programmation » en français, est un ensemble de règles à suivre pour pouvoir dialoguer avec d'autres applications. Dans le cas de Google API, il permet en particulier de communiquer avec Google Maps.Facile à vendreLe Play Store (anciennement Android Market) est une plateforme immense et très visitée ; c'est donc une mine d'opportunités pour quiconque possède une idée originale ou utile.FlexibleLe système est extrêmement portable, il s'adapte à beaucoup de structures différentes. Les smartphones, les tablettes, la présence ou l'absence de clavier ou de trackball, différents processeurs… On trouve même des fours à micro-ondes qui fonctionnent à l'aide d'Android ! Non seulement c'est une immense chance d'avoir autant d'opportunités, mais en plus Android est construit de manière à faciliter le développement et la distribution en fonction des composants en présence dans le terminal (si votre application nécessite d'utiliser le Bluetooth, seuls les terminaux équipés de Bluetooth pourront la voir sur le Play Store).IngénieuxL'architecture d'Android est inspirée par les applications composites, et encourage par ailleurs leur développement. Ces applications se trouvent essentiellement sur internet et leur principe est que vous pouvez combiner plusieurs composants totalement différents pour obtenir un résultat surpuissant. Par exemple, si on combine l'appareil photo avec le GPS, on peut poster les coordonnées GPS des photos prisesLes difficultés du développement pour des systèmes embarquésIl existe certaines contraintes pour le développement Android, qui ne s'appliquent pas au développement habituel ! Prenons un cas concret : la mémoire RAM est un composant matériel indispensable. Quand vous lancez un logiciel, votre système d'exploitation lui réserve de la mémoire pour qu'il puisse créer des variables, telles que des tableaux, des listes, etc. Ainsi, sur mon ordinateur, j'ai 4 Go de RAM, alors que je n'ai que 512 Mo sur mon téléphone, ce qui signifie que j'en ai huit fois moins.
Je peux donc lancer moins de logiciels à la fois et ces logiciels doivent faire en sorte de réserver moins de mémoire. C'est pourquoi votre téléphone est dit limité, il doit supporter des contraintes qui font doucement sourire votre ordinateur.Voici les principales contraintes à prendre en compte quand on développe pour un environnement mobile :Il faut pouvoir interagir avec un système complet sans l'interrompre. Android fait des choses pendant que votre application est utilisée, il reçoit des SMS et des appels, entre autres. Il faut respecter une certaine priorité dans l'exécution des tâches. Sincèrement, vous allez bloquer les appels de l'utilisateur pour qu'il puisse terminer sa partie de votre jeu de sudoku ?Comme je l'ai déjà dit, le système n'est pas aussi puissant qu'un ordinateur classique, il faudra exploiter tous les outils fournis afin de débusquer les portions de code qui nécessitent des optimisations.La taille de l'écran est réduite, et il existe par ailleurs plusieurs tailles et résolutions différentes. Votre interface graphique doit s'adapter à toutes les tailles et toutes les résolutions, ou vous risquez de laisser de côté un bon nombre d'utilisateurs.Autre chose qui est directement lié, les interfaces tactiles sont peu pratiques en cas d'utilisation avec un stylet et/ou peu précises en cas d'utilisation avec les doigts, d'où des contraintes liées à la programmation événementielle plus rigides. En effet, il est possible que l'utilisateur se trompe souvent de bouton. Très souvent s'il a de gros doigts.Enfin, en plus d'avoir une variété au niveau de la taille de l'écran, on a aussi une variété au niveau de la langue, des composants matériels présents et des versions d'Android.
Il y a une variabilité entre chaque téléphone et même parfois entre certains téléphones identiques. C'est un travail en plus à prendre en compte.Les conséquences de telles négligences peuvent être terribles pour l'utilisateur. Saturez le processeur et il ne pourra plus rien faire excepté redémarrer ! Faire crasher une application ne fera en général pas complètement crasher le système, cependant il pourrait bien s'interrompre quelques temps et irriter profondément l'utilisateur. Il faut bien comprendre que dans le paradigme de la programmation classique vous êtes dans votre propre monde et vous n'avez vraiment pas grand-chose à faire du reste de l'univers dans lequel vous évoluez, alors que là vous faites partie d'un système fragile qui évolue sans anicroche tant que vous n'intervenez pas. Votre but est de fournir des fonctionnalités de plus à ce système et faire en sorte de ne pas le perturber. Bon, cela paraît très alarmiste dit comme ça, Android a déjà anticipé la plupart des âneries que vous commettrez et a pris des dispositions pour éviter des catastrophes qui conduiront au blocage total du téléphone. Si vous êtes un tantinet curieux, je vous invite à lire l'annexe sur l'architecture d'Android pour comprendre un peu pourquoi il faut être un barbare pour vraiment réussir à saturer le système.Le langage JavaCette petite section permettra à ceux fâchés avec le Java de se remettre un peu dans le bain et surtout de réviser le vocabulaire de base. Notez qu'il ne s'agit que d'un rappel, il est conseillé de connaître la programmation en Java auparavant ; je ne fais ici que rappeler quelques notions de base pour vous rafraîchir la mémoire ! Il ne s'agit absolument pas d'une introduction à la programmation.Les variablesLa seule chose qu'un programme sait faire, c'est des calculs. Il arrive qu'on puisse lui faire afficher des formes et des couleurs, mais pas toujours. Pour faire des calculs, on a besoin de variables. Ces variables permettent de conserver des informations avec lesquelles on va pouvoir faire des opérations. Ainsi, on peut avoir une variable radis qui vaudra 4 pour indiquer qu'on a quatre radis.
Si on a une variable carotte qui vaut 2, on peut faire le calcul radis + carotte de manière à pouvoir déduire qu'on a six légumes.Les primitivesEn Java, il existe deuxtypes de variable. Le premier type s'appelle les primitives. Ces primitives permettent de retenir des informations simples telles que des nombres sans virgule (auquel cas la variable est un entier, int), des chiffres à virgule (des réels, float) ou des booléens (variable qui ne peut valoir que vrai (true) ou faux (false), avec les boolean).Les objetsLe second type, ce sont les objets. En effet, à l'opposé des primitives (variables simples), les objets sont des variables compliquées. En fait, une primitive ne peut contenir qu'une information, par exemple la valeur d'un nombre ; tandis qu'un objet est constitué d'une ou plusieurs autres variables, et par conséquent d'une ou plusieurs valeurs.Ainsi, un objet peut lui-même contenir un objet ! Un objet peut représenter absolument ce qu'on veut : une chaise, une voiture, un concept philosophique, une formule mathématique, etc. Par exemple, pour représenter une voiture, je créerai un objet qui contient une variable roue qui vaudra 4, une variable vitesse qui variera en fonction de la vitesse et une variable carrosserie pour la couleur de la carrosserie et qui pourra valoir « rouge », « bleu », que sais-je ! D'ailleurs, une variable qui représente une couleur ? Ça ne peut pas être une primitive, ce n'est pas une variable facile ça, une couleur ! Donc cette variable sera aussi un objet, ce quisignifie qu'un objet peut contenir des primitives ou d'autres objets.Mais dans le code, comment représenter un objet ? Pour cela, il va falloir déclarer ce qu'on appelle une classe. Cette classe aura un nom, pour notre voiture on peut simplement l'appeler Voiture, comme ceci :Code : Java// On déclare une classe Voiture avec cette syntaxe class Voiture {// Et dedans on ajoute les attributs qu'on utilisera, par exemple le nombre de rouesint roue = 4;// On ne connaît pas la vitesse, alors on ne la déclare pasfloat vitesse;// Et enfin la couleur, qui est représentée par une classe de nom CouleurCouleur carrosserie;}Les variables ainsi insérées au sein d'une classe sont appelées des attributs.Il est possible de donner des instructions à cette voiture, comme d'accélérer ou de s'arrêter. Ces instructions s'appellent des méthodes, par exemple pour freiner :Code : Java//Je déclare une méthode qui s'appelle "arreter"void arreter() {//Pour s'arrêter, je passe la vitesse à 0vitesse = 0;}