Très bien. Donc, maintenant que nous en avons fini avec l'installation et la configuration, nous sommes prêts à commencer avec

le développement de Flutter et à construire notre toute première application. Et cette application est une application appelée I Am Rich.

Maintenant, si vous n'avez pas entendu parler de l'histoire, au début de l'app store, quand c'était encore

le Far West du développement d'applications, un gars nommé Arman a décidé de créer une application appelée "Je suis riche".

Et c'était une idée plutôt intelligente.

Il a décidé qu'il allait créer une application qui ne fait absolument rien.

Il montre juste une seule icône d'un rubis rouge sur l'écran. Et il a décidé de la mettre au prix maximum possible

possible sur l'App Store, soit 999,99$, et il l'a présentée comme une application d'art et de style de vie, pour que les gens très riches puissent montrer aux autres combien ils sont riches.

Ou bien, pour les personnes un peu folles, pour montrer au monde qu'elles ont acheté une application qui ne fait rien pour 1 000 dollars.

Mais de toute façon, avant que l'application ne soit retirée par Apple, sept ou huit personnes l'ont achetée, dont cinq

étaient probablement des enfants. Des enfants qui ont probablement été battus par leurs parents pour avoir payé une application qui ne fait rien pour 1000 $.

Mais c'est une application parfaite pour nous permettre de commencer à développer des applications Flutter.

Elle nous permettra de nous habituer à l'environnement de développement et à tous les petits boutons.

Et aussi de voir comment une véritable application Flutter est construite à partir de zéro.

Donc, une fois que vous êtes prêt, lancez Android Studio et commençons.

Donc une fois que vous êtes prêt, allez-y et ouvrez Android Studio et sélectionnez l'option

où nous pouvons commencer un nouveau projet Flutter. Et nous allons sélectionner la première option pour créer une application Flutter.

et aller de l'avant et cliquer sur suivant.

Maintenant nous devons nommer notre projet, et nous allons l'appeler i am rich.

Ce nom doit être en minuscule, avec un trait de soulignement entre chaque mot du nom du projet.

Et puis nous allons laisser tout le reste tel quel.

Cette partie vous indique où votre projet va être enregistré. Par défaut, il est généralement enregistré dans un dossier

appelé Android Studio projects et vous serez en mesure de le trouver si vous recherchez Android Studio projects.

Et c'est là que votre toute nouvelle application va être créée.

Maintenant que nous avons nommé notre projet, allons-y et cliquons sur suivant et ici, il s'attend à ce que nous tapions un nom d'entreprise.

de l'entreprise.

Par exemple, notre site web est www.appbrewery.co

Donc le domaine de notre entreprise sera appbrewery.co.

Maintenant, la raison pour laquelle ils veulent cela est qu'ils peuvent créer un nom de paquet pour votre application, en combinant le domaine de votre entreprise avec le nom de votre projet.

domaine de l'entreprise avec le nom de votre projet pour créer quelque chose qui peut identifier cette application particulière

de manière unique, sur les magasins d'applications.

Ils espèrent donc que personne sur appbrewery.co ne créera une application en double et n'essaiera de la télécharger.

Maintenant, que faire si vous n'avez pas de site web ou de domaine d'entreprise ?

Eh bien, vous pouvez simplement taper votre nom complet et terminer par un .com.

Et tant que personne n'a le même nom que vous et n'a essayé de développer une application appelée "Je suis riche", alors vous...

vous devriez vous en sortir. Si vous avez un site web qui vous appartient,

alors bien sûr, personne d'autre ne possédera le site que vous possédez.

Cela garantit donc qu'il sera unique.

Donc maintenant, allons-y et cliquons sur "finish", et Android Studio va créer un tout nouveau projet Flutter pour vous.

et il va vous mettre en place avec du code de départ. Le code de départ est créé par l'équipe Flutter, et

tout ce qu'il fait c'est créer une simple application de comptage.

Maintenant, dans ce cas, c'est une application de compteur qui est créée en utilisant ce qu'on appelle un échafaudage.

Et si vous voulez juste l'exécuter rapidement juste pour voir à quoi ça ressemble, alors allez-y et ouvrez votre

émulateur Android ou votre simulateur iOS, et cliquez sur Run.

Et une fois que c'est fait, voici ce que vous devriez voir.

C'est une application très simple avec une barre supérieure avec du texte au milieu et un bouton que vous pouvez presser.

Maintenant, ce n'est pas l'application que nous voulons créer.

C'est juste l'équipe de Flutter qui vous montre ce que vous pouvez faire avec des échafaudages, et nous allons construire

la nôtre à partir de rien.

—

Nous allons donc créer un échafaudage, et nous allons l'utiliser pour construire notre application "Je suis riche". Mais

afin de comprendre ce qui se passe,

Je vais supprimer tout ce qui est actuellement dans le fichier, sauf les deux premières lignes. Donc

tout jusqu'à la ligne 12.

Donc maintenant nous avons une petite erreur ici, et vous la voyez dans Dart Analysis aussi, où il est dit "MyApp isn't

defined." Et bien c'est parce que MyApp était en fait l'application de l'équipe Flutter, c'était l'application du compteur. Donc ça

n'existe plus vraiment parce que je viens de la supprimer.

Donc à la place, l'application que nous allons lancer est une MaterialApp vide, et cette application sera quelque chose qui

qui se conforme au modèle de conception matérielle. Et le material design est simplement un style de design ou un concept

qui a été créé par Google et beaucoup de startups et beaucoup d'entreprises l'ont adopté pour leurs

sites web et leurs applications mobiles. Et en utilisant Flutter, nous pouvons exploiter un grand nombre de ces composants matériels,

à travers l'utilisation de widgets Flutter. Et nous allons voir la puissance de ces widgets et comment nous pouvons

incorporer ces conceptions dans, presque automatiquement, très très bientôt dans ce module. Revenons à notre

Material App, et comme vous pouvez le voir, nous pouvons commencer à construire notre arbre de widgets.

Dans la Material App, la chose la plus importante à définir est la maison.

Et c'est là que tout commence.

Si nous allons de l'avant et disons que dans notre Material App, dans la maison de l'application, ce que nous voulons voir est simplement

un widget texte et ce widget texte dit simplement bonjour le monde. Ensuite,

maintenant, si nous allons de l'avant et lançons cette application, alors vous pouvez voir notre texte s'afficher dans notre Material App.

Mais pour le moment, ça n'a pas l'air très bien.

C'est juste un écran noir avec du texte rouge. Mais il dit ce que nous voulions dire, c'est à dire bonjour le monde.

Nous avons vraiment commencé à construire notre arbre de widgets, et c'est un arbre très simple.

Il n'y a que deux widgets. Le premier, ou le widget parent, est le Material App, et presque toutes vos applications

applications Flutter commenceront avec ce widget comme premier widget.

La prochaine chose que nous disons est que dans cette Material App, la seule chose que nous voulons montrer, est

un widget texte. Et par défaut, les widgets de texte sont alignés dans le coin supérieur gauche.

Et c'est pourquoi nous le voyons juste ici parce qu'il essaie d'être carrément dans le coin supérieur gauche.

Maintenant, que faire si nous voulons que le texte soit au centre.

Eh bien, comme je l'ai dit, les applications Flutter sont presque entièrement construites en utilisant des widgets. Et il y a bien sûr un widget

pour centrer les éléments.

Donc si au lieu d'avoir le texte comme enfant de l'application matérielle, il était à l'intérieur d'un widget de centrage. Alors il

sera automatiquement mis au centre de l'écran.

Notre arbre est devenu un peu plus complexe.

Nous avons trois widgets maintenant. Hourra.

Et le premier widget ou le parent est Material App. A l'intérieur de Material App,

nous avons un widget central, qui centre tout ce qu'il contient.

Et nous lui avons donné un widget de texte à contenir.

Le texte est donc automatiquement centré. Allons-y et supprimons notre widget de texte.

Je vais donc utiliser COMMAND + X ou CONTROL + X pour le couper, et nous allons juste le laisser ici

pour le moment.

Et à l'intérieur de l'application matérielle, au lieu d'avoir le widget de texte, nous allons avoir un widget central.

le widget central peut aussi avoir un enfant. Donc, il y a deux niveaux de profondeur maintenant.

Et son enfant sera un widget de texte.

Donc maintenant, quand je lance l'application en utilisant le bouton play, vous verrez que mon hello world est automatiquement déplacé

au centre de l'écran.

Et c'est parce qu'il est l'enfant d'un widget central.

Ce que vous remarquerez en créant des applications Flutter, c'est qu'il y a des widgets qui remplissent

presque tous les rôles. Tout comme dans une entreprise,

vous pouvez avoir des comptables, des vendeurs et des techniciens.

Vous avez également des widgets qui remplissent une variété de rôles.

Le widget texte est chargé d'afficher le texte et de le styliser.

Le widget central est responsable de la disposition de l'écran et aide à placer quelque chose au centre.

Et l'application matérielle est en quelque sorte le grand-père de tous les widgets, parce qu'elle se trouve généralement en haut de l'écran et que tout ce que nous construisons est assez simple.

au sommet et tout ce que nous construisons est à peu près une Material App.

Maintenant, pour le moment, notre code est sur la même ligne et il est vraiment difficile de voir quel widget est à l'intérieur de

quel autre.

Alors, comment pouvons-nous rendre nos vies plus faciles ?

Eh bien, en installant les paquets Flutter et Dart, nous avons automatiquement accès à un reformateur appelé dartfmt.

Et c'est vraiment très utile.

Mais vous devez l'aider à vous aider. L'équipe de Flutter conseille que chaque fois que vous créez un widget

ce qui implique généralement ces parenthèses ou ces crochets ronds, après chacun d'entre eux, il suffit d'ajouter une

virgule.

dans votre propre application.

—

Donc je vais aller de l'avant et ajouter une virgule à la fin de tous mes crochets. Et cela signifie que lorsque

j'appuie sur sauver, ou quand je fais un clic droit et que je clique sur reformater le code avec dartfmt, alors ça va automatiquement transformer

mon code d'une ligne et très difficile à reconnaître, en une structure indentée.

Donc maintenant, il est beaucoup plus facile de voir que ce widget de texte est à l'intérieur de ce widget central qui est à l'intérieur de cette application matérielle.

Material App et que tout se trouve à l'intérieur de cette chose appelée main.

Maintenant cette chose appelée main est vraiment importante, parce que quand notre application est lancée, le fichier de départ est ce

appelé main.dart dans lequel nous sommes en train d'écrire du code.

Donc il lance cette application et il commence par le haut. Il cherche quelque chose appelé main. Et ensuite il cherche à voir ce qu'il doit faire.

C'est le point de départ de toutes nos applications.

Maintenant, afin de rendre notre structure indentée encore plus organisée, nous pouvons remplacer cette soi-disant

grosse flèche ou ce qui est un = et > avec un ensemble d'accolades.

Nous pouvons donc la transformer en une accolade au début, et une accolade à la fin.

Donc, cela fait exactement la même chose que ce qu'il faisait avant. Mais maintenant, notre code est reformaté de sorte que nous avons

cette indentation vraiment agréable et soignée. Et en un coup d'oeil, je peux déjà dire ce qui est à l'intérieur de quoi, ce qui devient

vraiment important lorsque vous construisez votre application avec de plus en plus de widgets les uns dans les autres.

La dernière chose que je veux souligner sur cette page est en haut, ici, vous voyez ceci

Si vous cliquez dessus, le contenu sera développé. Et vous pouvez toujours réduire et étendre les choses juste en

en cliquant dans la marge ici.

Donc si je veux réduire toute mon application matérielle, je peux le faire et faire en sorte qu'elle apparaisse comme si elle n'occupait

une ligne.

Maintenant cette partie est un message de l'équipe de Flutter, et il vous dit ce que le code d'exemple original était

essayait de démontrer et de vous montrer. La particularité de ce message est qu'il commence par deux lignes

// et dans Dart, comme dans beaucoup d'autres langages, cela transforme tout texte qui vient après, en

un commentaire.

Les commentaires sont très utiles, car ils vous permettent de laisser des notes à vous-même ou à d'autres personnes de votre équipe, lorsque vous travaillez sur un projet.

de votre équipe, lorsque vous travaillez sur la même base de code.

Cela signifie que je pourrais dire à mon futur moi que, lorsque j'ai écrit ce code pour la première fois et que j'ai compris comment il

fonctionnait parfaitement, ce que chaque partie est censée faire. Et pendant que vous apprenez, c'est une très bonne idée de laisser des notes pour vous-même

à travers l'utilisation de ces commentaires pour que vous puissiez vous rappeler plus tard, quand vous réviserez votre code, ce que chaque partie est censée faire.

Donc, par exemple, laissez-moi commencer.

Nous pouvons écrire, "La fonction principale est le point de départ de toutes nos applications Flutter."

Ce qui veut dire que la prochaine fois que vous reviendrez à ce code et que vous regarderez ce truc appelé main, vous vous direz .

"Oh, qu'est-ce que ça fait déjà ?"

Eh bien, il y a un commentaire très utile de l'ancien vous, pour essayer de vous aider à comprendre ce qu'il fait.

Et vous vous souvenez, "ah je sais que c'est le point de départ de toutes nos applications".

La première chose qui se passe quand on lance notre application, c'est qu'elle regarde dans cette fonction principale pour voir ce qu'elle doit faire.

ce qu'elle doit faire. Et elle lui dit qu'elle doit lancer une application.

L'application qu'elle doit exécuter est une Material App, et la Material App aura un widget central qui vous aide à

de centrer ce widget texte sur l'écran. Et c'est comme ça qu'on se retrouve avec quelque chose qui ressemble à ça.

Une fois que vous êtes satisfaits, je vous recommande de le supprimer car nous n'en aurons pas besoin.

La partie du code à laquelle il se réfère, nous l'avons déjà supprimée depuis longtemps, et je recommande de commencer à

à laisser des notes pour vous-même à la manière de ces commentaires afin que vous compreniez, plus tard, lorsque vous revenez

quand vous reviendrez à votre code, ce que tout fait et comment ça marche.

Dans cette leçon, nous avons déjà commencé à construire un arbre de widgets et grâce à l'utilisation de ces widgets, nous avons

créé une application très simple qui affiche simplement le mot "hello world" à l'écran.

Maintenant, dans les prochaines leçons, nous allons développer cette application et la rendre plus sophistiquée, plus ressemblante

à l'application "Je suis riche" que nous allons construire.

Mais c'est le point de départ pour construire n'importe quelle application Flutter, et quand vous construisez vos propres applications.

savez maintenant comment passer par l'assistant et enlever le code de départ pour que vous puissiez le personnaliser pour qu'il devienne

macos/deepLFree.translatedWithDeepL.text