eBook Gratuit

APPRENEZ Xcode

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#xcode

Table des matières

À propos

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: xcode

It is an unofficial and free Xcode ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official Xcode.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapitre 1: Démarrer avec Xcode

Remarques

Xcode est un environnement de développement intégré pour macOS qui prend en charge le développement d'applications natives pour macOS, iOS, watchOS et tvOS. Xcode est le successeur de Project Builder et PBX de NeXT. (En fait, les fichiers de manifeste du projet Xcode sont toujours nommés avec l'extension `.pbxproj` .)

Les versions Xcode incluent des versions stables du compilateur clang C / C ++ / Obj-C, du compilateur Swift , du débogueur LLDB et des simulateurs iOS / watchOS / tvOS. Xcode inclut également Interface Builder , ainsi que des outils permettant d'afficher et de modifier des modèles et des scènes 3D, des éléments d'image, etc.

Versions

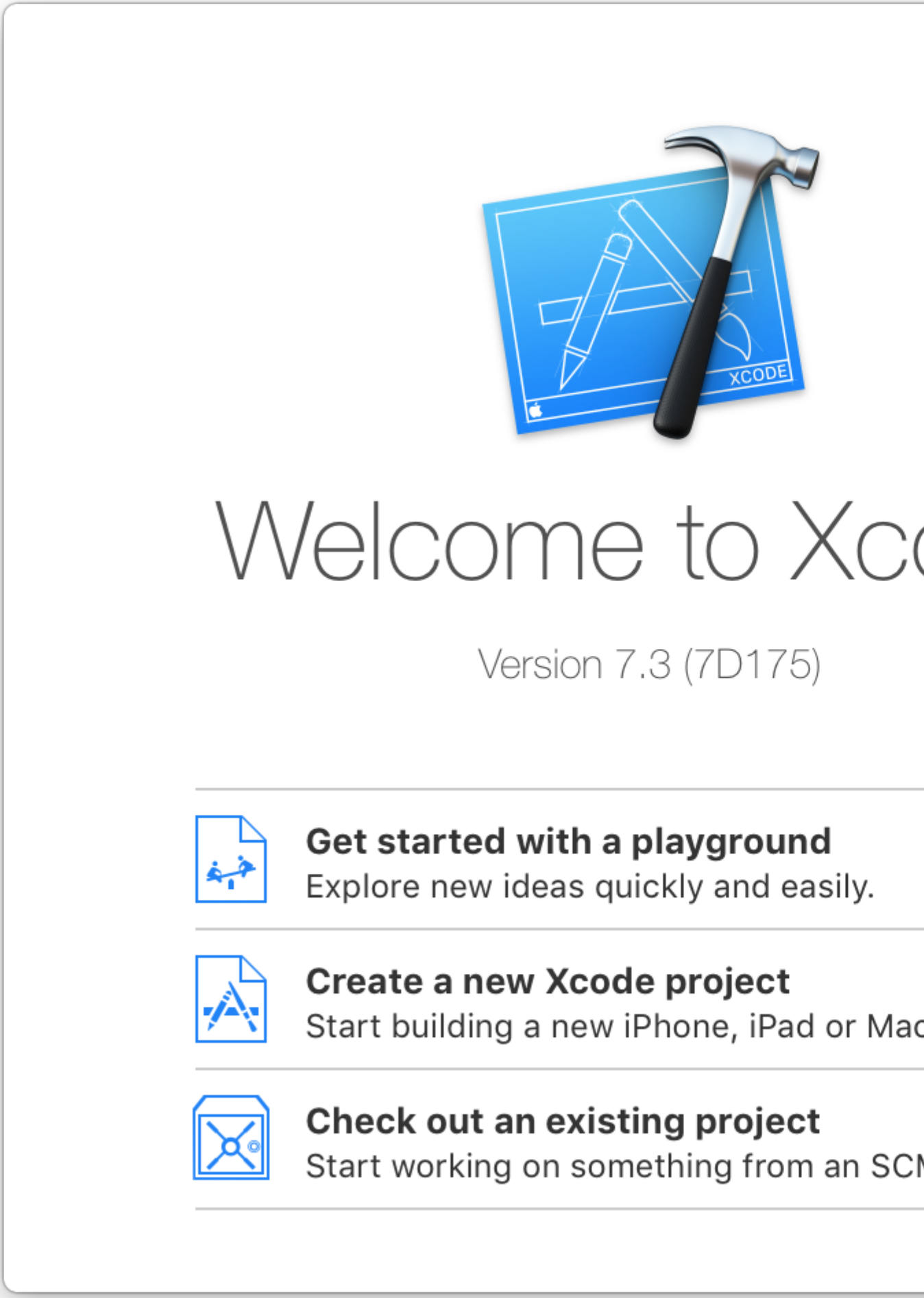
Version	Date de sortie
1.0	2003-09-28
2.0	2005-04-04
3.0	2007-10-26
4.0	2011-03-14
5.0	2013-09-18
6,0	2014-09-17
7.0	2015-09-16
7.1.1	2015-11-09
7.2	2015-12-08
7.2.1	2016-02-03
7.3	2016-03-21
7.3.1	2016-05-03
8.0	2016-09-13

Version	Date de sortie
8.1	2016-10-27
8.2	2016-12-12
8.2.1	2016-12-19
8.3	2017-03-27
8.3.1	2017-04-06
8.3.2	2017-04-18
8.3.3	2017-06-05

Examples

Commencer

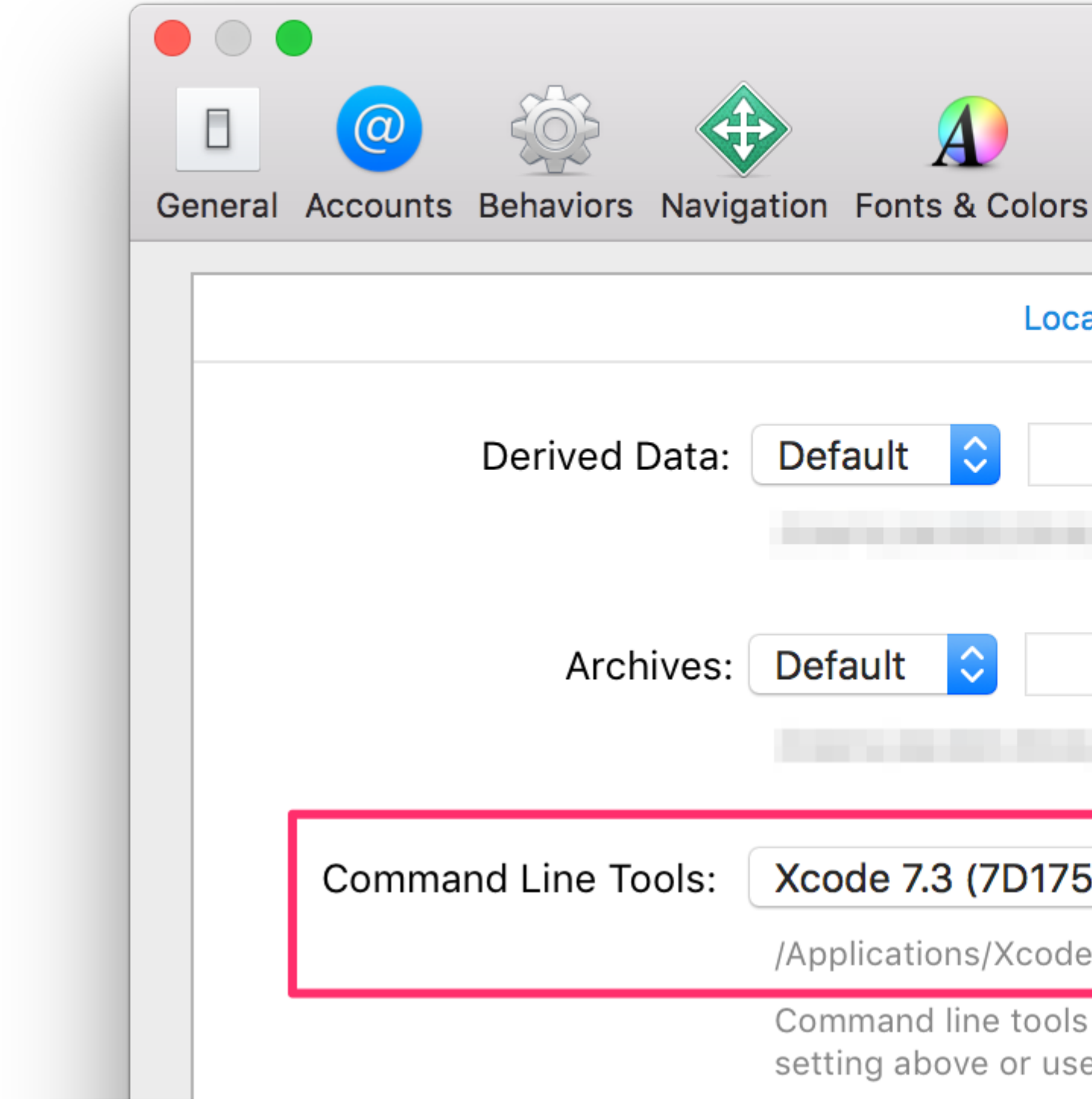
- Téléchargez Xcode depuis le Mac App Store.
- Cliquez pour créer un nouveau projet ou un terrain de jeu:


XCODE
Welcome to Xc
Version 7.3 (7D175)
Get started with a playground
Explore new ideas quickly and easily.
Create a new Xcode project
Start building a new iPhone, iPad or Mac
Check out an existing project
Start working on something from an SCM

une version existante sur votre ordinateur. Vous pouvez également installer Xcode depuis un téléchargement direct pour mieux contrôler les versions que vous avez.

Chaque copie de Xcode inclut des outils de ligne de commande (`clang` , `xcodebuild` , etc.). Vous pouvez choisir celles qui sont appelées par les commandes dans `/usr/bin` .

Dans les préférences de Xcode, sous l'onglet Emplacements, choisissez une version de Xcode:

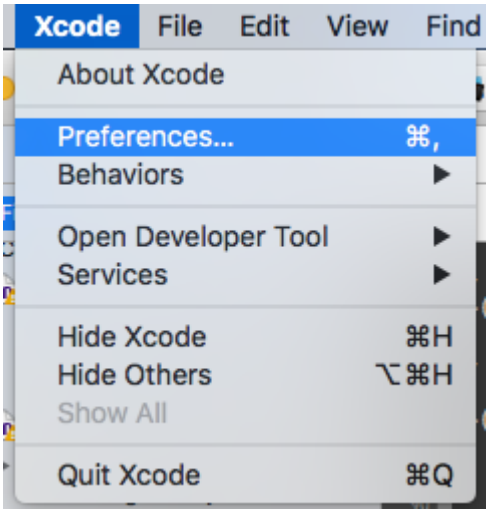


Ou vous pouvez gérer des versions à partir de la ligne de commande en utilisant `xcode-select` :

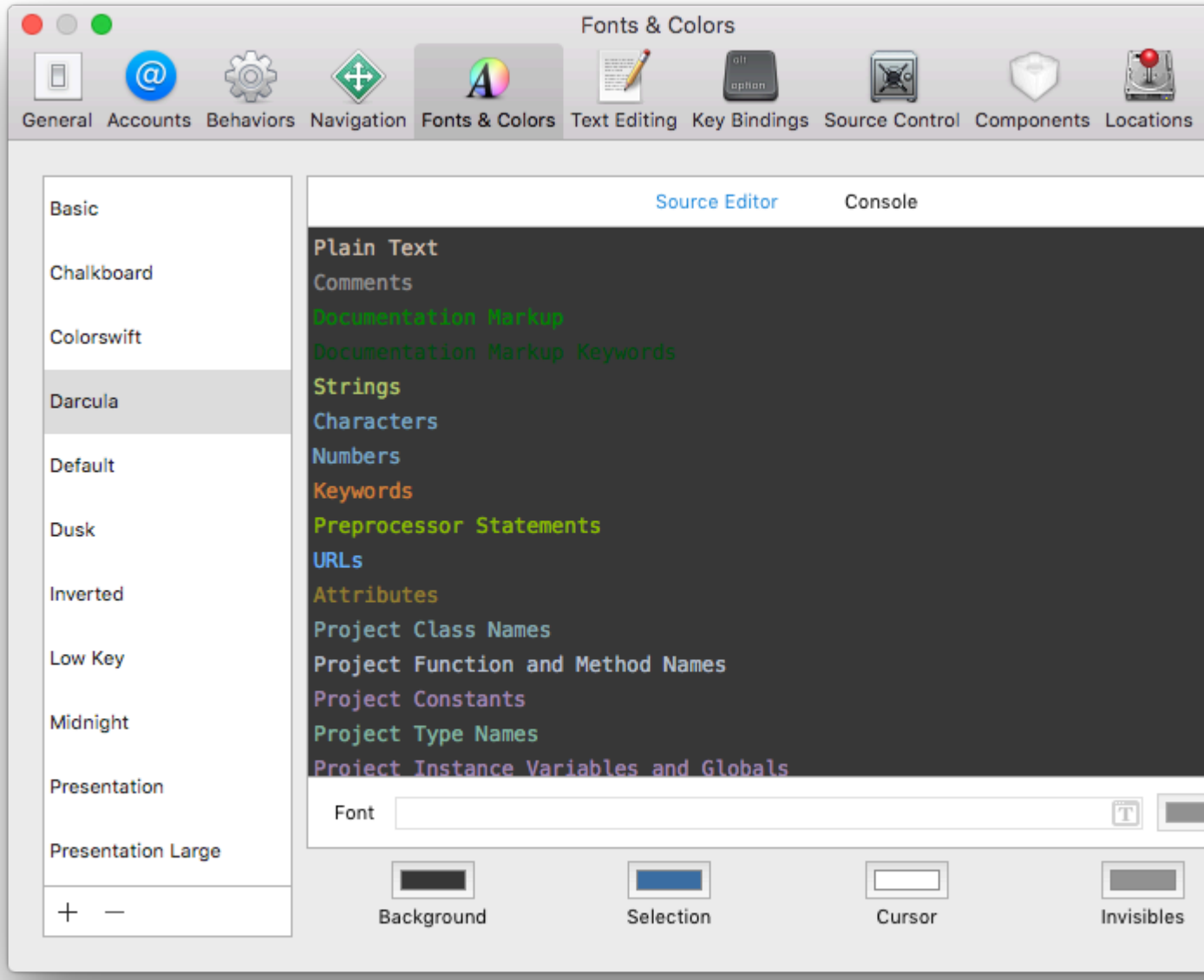
```
# Print the currently selected version
$ xcode-select --print-path
/Applications/Xcode.app/Contents/Developer

$ clang --version
Apple LLVM version 7.3.0 (clang-703.0.29)
Target: x86_64-apple-darwin15.4.0
Thread model: posix
InstalledDir:
/Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/bin

# Find all installed versions using Spotlight
$ mdfind 'kMDItemCFBundleIdentifier = "com.apple.dt.Xcode"'
/Applications/Xcode.app
/Applications/Xcode72.app

# Check their version numbers
$ mdfind 'kMDItemCFBundleIdentifier = "com.apple.dt.Xcode"' | xargs mdls -name kMDItemVersion
kMDItemVersion = "7.3"
kMDItemVersion = "7.2.1"

# Switch to a different version
$ sudo xcode-select --switch /Applications/Xcode72.app

$ clang --version
Apple LLVM version 7.0.2 (clang-700.1.81)
Target: x86_64-apple-darwin15.4.0
Thread model: posix
```

Changer le schéma de couleurs

De nombreux développeurs aiment personnaliser la police, le texte et la couleur d'arrière-plan de leurs IDE. Vous pouvez le faire dans Xcode en ouvrant le volet des préférences de l'application, soit en allant dans XCODE-> Préférences, soit en appuyant sur «».

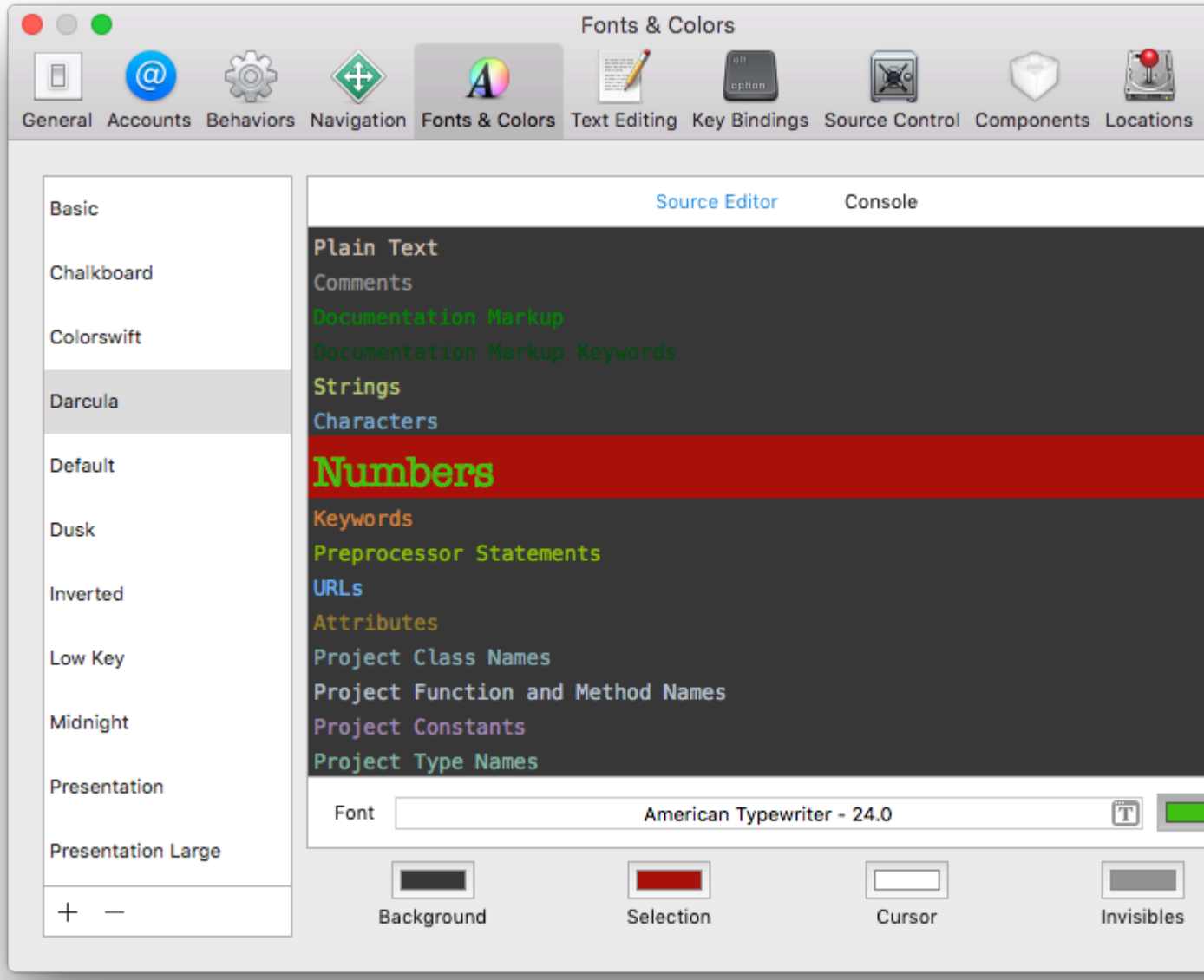


Avec le volet de préférences ouvert, vous pouvez cliquer sur l'onglet "Polices et couleurs".

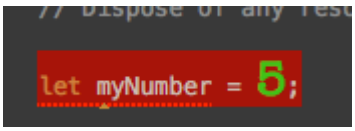


À partir de là, vous pouvez modifier les arrière-plans et les couleurs de police de la source ET de la console. Il existe de nombreux schémas de couleurs et de polices prédéfinis fournis avec Xcode. Vous les choisissez dans la liste de gauche (Basic, Chalkboard, etc.). Vous pouvez trouver et télécharger plus en ligne (comme ici par exemple).

Pour personnaliser davantage un thème, vous pouvez personnaliser tous les types répertoriés dans le volet de droite (Texte brut, Commentaires, Marquage de la documentation, etc.). Par exemple, disons que je veux vraiment que mes «numéros» apparaissent dans mon code. Donc, je change la police en "American Typewriter" à 24 px, la couleur à une couleur verdâtre et définissez la ligne en surbrillance rouge:



Maintenant, dans mon édition de texte, je peux vraiment voir mes chiffres:



Maintenant, vous pouvez personnaliser l'apparence de l'éditeur de source et de la console pour votre plus grand plaisir!

Astuce Pro

De nombreux développeurs aiment mettre en scène leur IDS sombre (texte clair, fond sombre). Dans Xcode, vous ne pouvez le faire que pour l'éditeur de sources et la console. Cependant, les sections Navigation (côté gauche), Débogage (bas) et Utilitaire (extrême droite) ne sont pas

personnalisables. Il y a deux solutions à cela. Tout d'abord (un peu délicat, il faut laisser le thème IDE clair (Fond clair, texte sombre) puis inverser les couleurs de l'écran. Cela rendra tout sombre, mais les couleurs du simulateur et du reste du système sont désormais vides. La deuxième solution consiste à masquer les zones de navigation, de débogage et d'utilitaire lorsque vous ne les utilisez pas Vous pouvez rapidement basculer ces zones à l'aide des commandes suivantes:

Navigateur: 0

Zone de débogage: Y

Utilitaire: 0

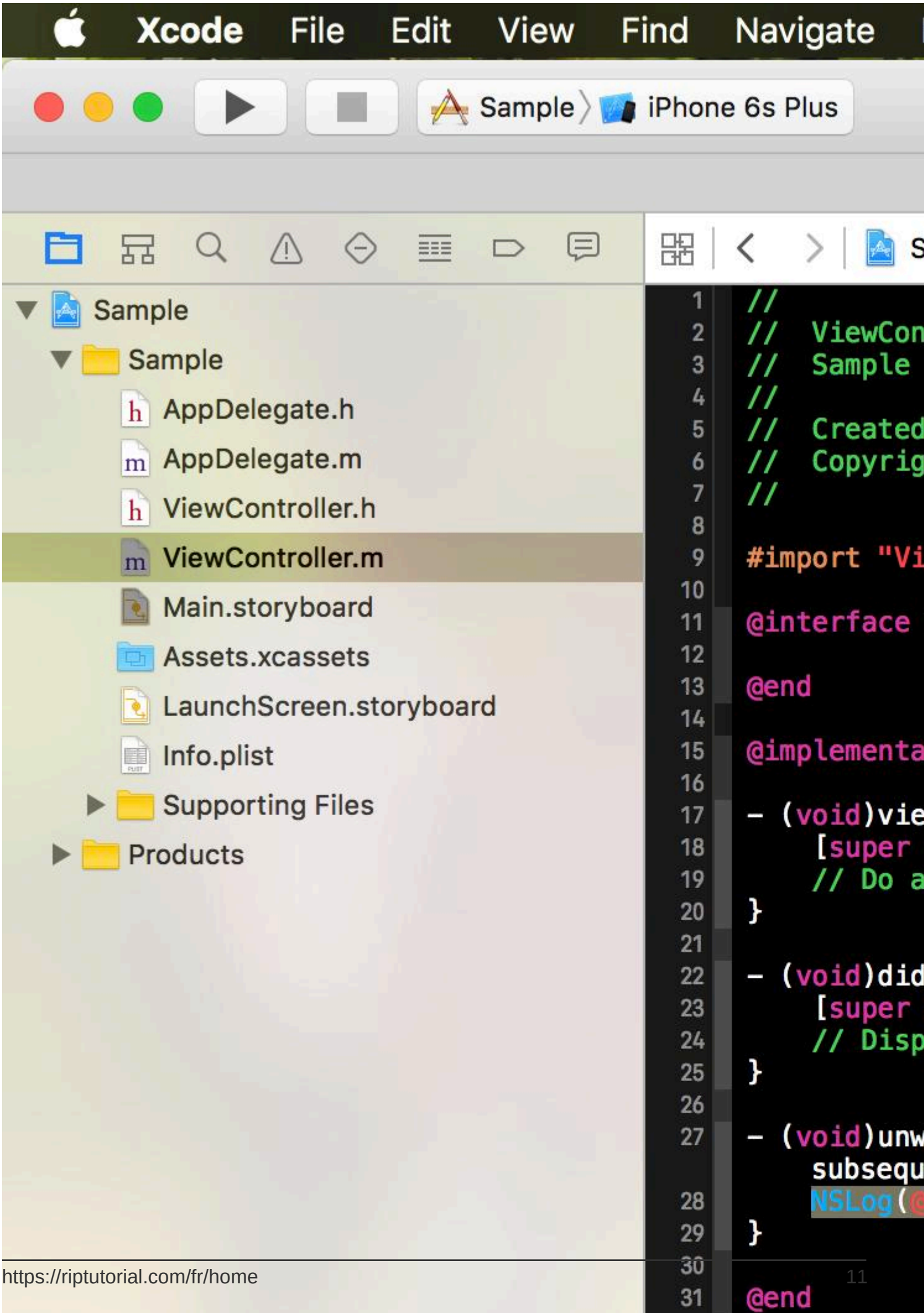
Lire Démarrer avec Xcode en ligne: https://riptutorial.com/fr/xcode/topic/294/demarrer-avec-xcode

Chapitre 2: Astuces Xcode

Examples

Réutiliser les extraits de code dans Xcode

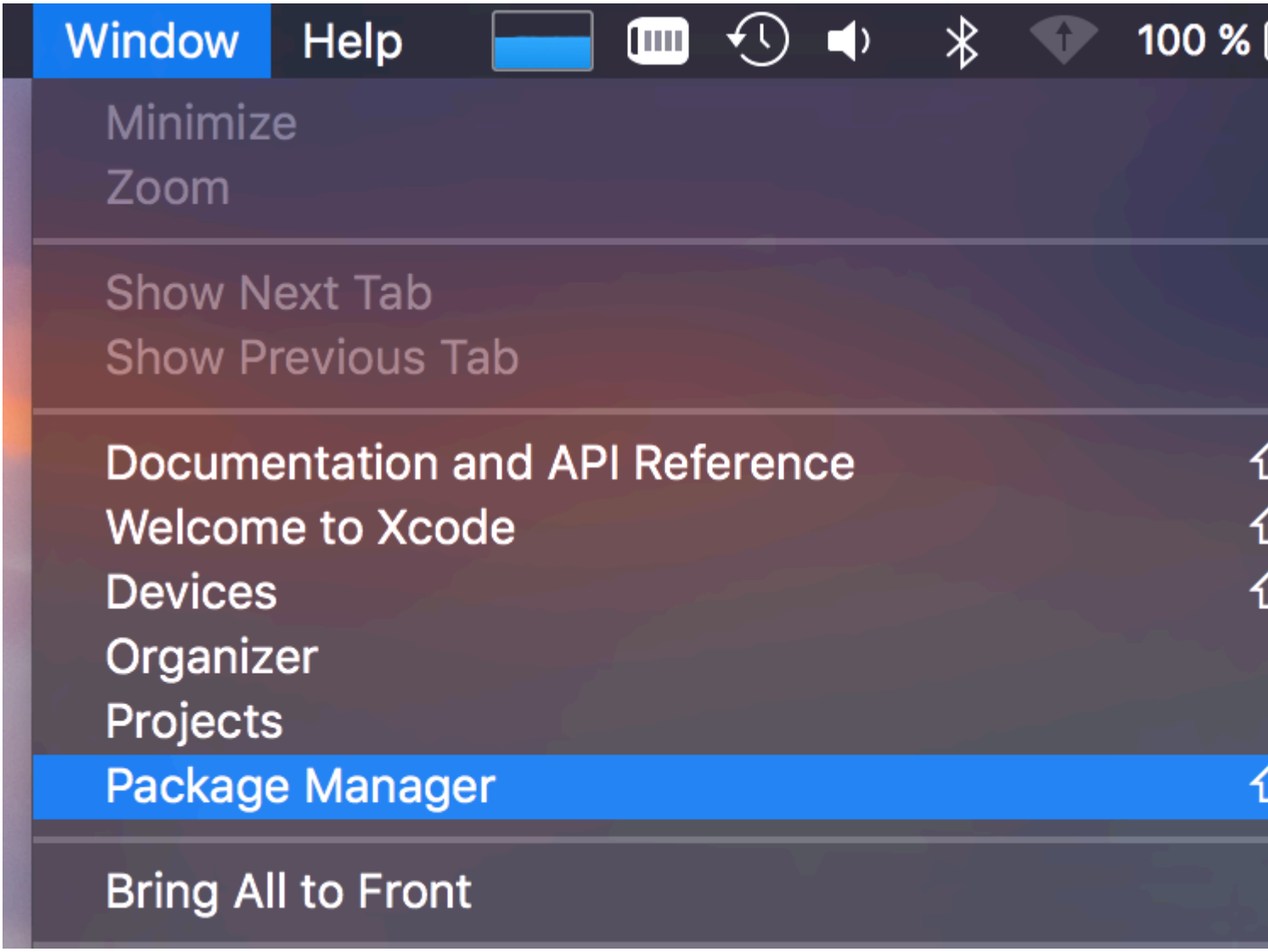
Vous pouvez enregistrer vos extraits de code pour les utiliser ultérieurement par simple glisser-déposer. Par exemple: si vous avez une déclaration NSLog utilisée pour tant d'endroits ailleurs dans le projet, vous pouvez enregistrer les instructions NSLog dans la bibliothèque d'extraits de code.


Xcode File Edit View Find Navigate
Sample iPhone 6s Plus
Sample
Sample
AppDelegate.h
AppDelegate.m
ViewController.h
ViewController.m
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Supporting Files
Products
1 //
2 // ViewCon
3 // Sample
4 //
5 // Created
6 // Copyrig
7 //
8
9 #import "Vi
10
11 @interface
12
13 @end
14
15 @implementa
16
17 - (void)vie
18 [super
19 // Do a
20 }
21
22 - (void)did
23 [super
24 // Disp
25 }
26
27 - (void)unw
subsequ
28 NSLog(@
29 }
30
31 @end

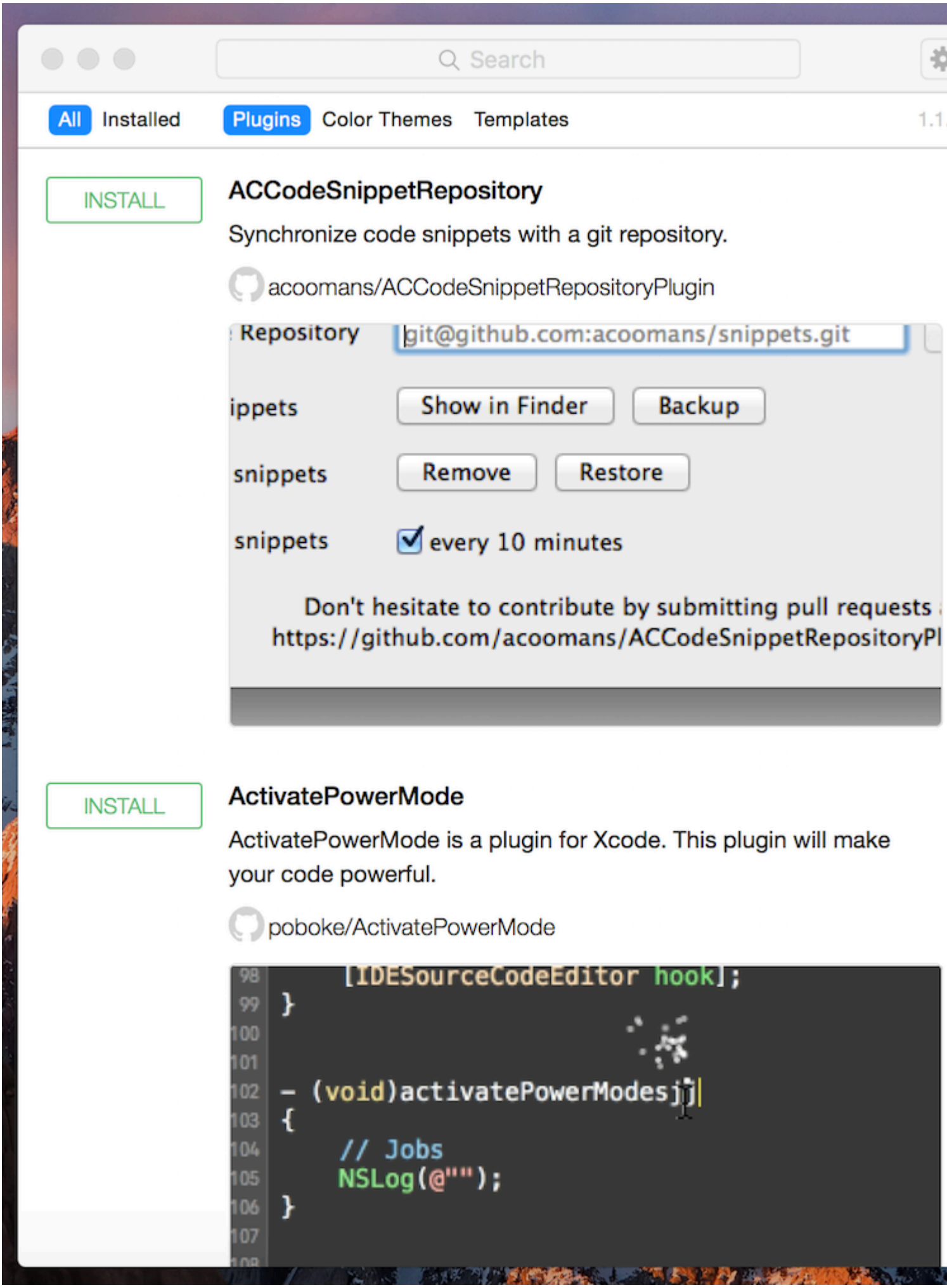
dessus de toute méthode, classe, ... déclaration pour ajouter de la documentation
- `XcodeColors` - `XcodeColors` console colorés, par exemple en utilisant CocoaLumberjack
- `FuzzyAutocomplete` - Tapez "NSog" et obtenez toujours `NSLog` autocompleted
- `BuildTimeAnalyzer` - Définit `-Xfrontend -debug-time-function-bodies` sous `Other Swift flags` dans les paramètres de construction et optimise votre temps de construction Swift

Bien sûr, il y en a beaucoup d'autres et certains sont si bons, Apple les a déjà implémentés dans Xcode 8 (FuzzyAutocomplete et VVDocumenter par exemple).

Usage

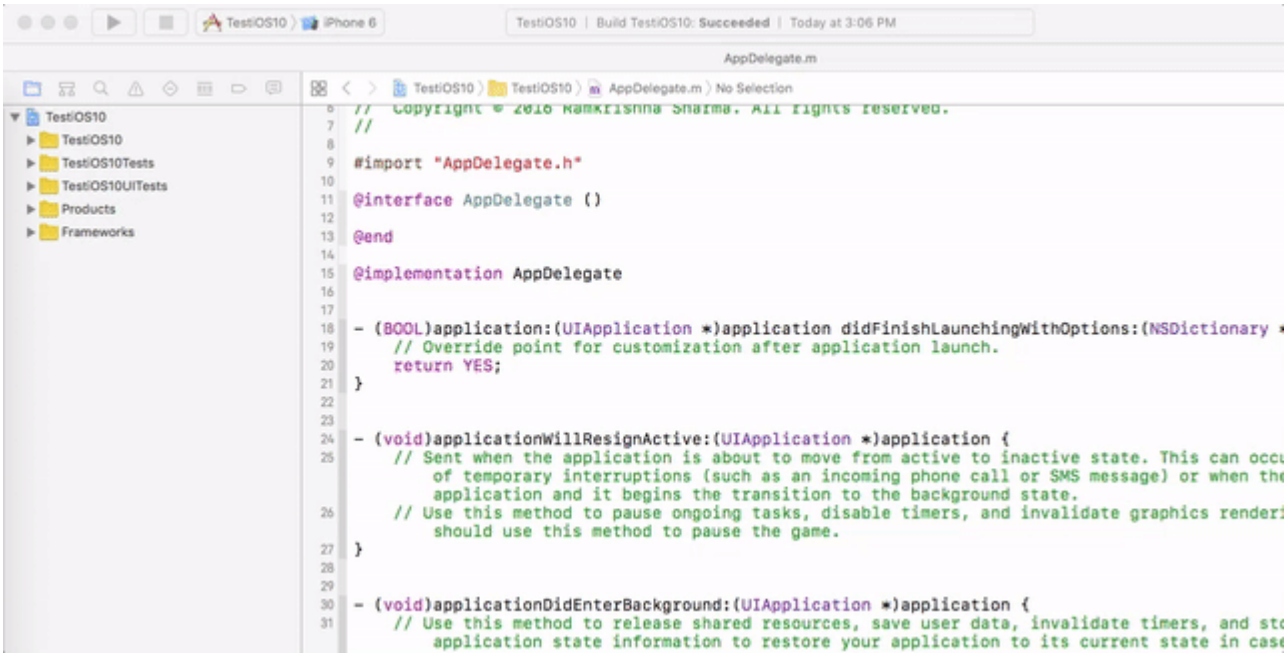


Appuyez sur ⌘ + ⌥ + 9 ou utilisez ce menu pour ouvrir le gestionnaire de paquets.



Cliquez sur installer sur tout package que vous souhaitez installer et redémarrer ensuite Xcode.

2. Sur votre environnement Variables, définissez OS_ACTIVITY_MODE = disable



Lire Astuces Xcode en ligne: https://riptutorial.com/fr/xcode/topic/3349/astuces-xcode

Chapitre 3: Certificats, profils d'approvisionnement et signature de code

Examples

Choisissez la bonne approche de signature de code

Si vous commencez un nouveau projet, il est important de réfléchir à la manière dont vous souhaitez gérer la signature du code.

Si vous n'êtes pas familier avec la signature de code, consultez la session WWDC qui décrit les principes fondamentaux de la signature de code dans Xcode.

Pour bien signer votre application, vous devez disposer des ressources suivantes sur votre ordinateur local:

- La clé privée (fichier `.p12`)
- Le certificat (fichier `.cer`) correspondant à la clé privée
- Le profil d'approvisionnement (fichier `.mobileprovision`), correspondant au certificat et à la clé privée installés localement

Sur le portail des développeurs Apple, vous devez également disposer d'un identifiant d'application valide associé à votre profil de configuration.

Utiliser la fonctionnalité de signature de code de Xcode

Parfois, le paramètre `Automatic` en tant que profil d'approvisionnement ne fonctionne pas de manière fiable car il sélectionne simplement le profil de configuration mis à jour le plus récemment, peu importe si le certificat est installé.

C'est pourquoi il est recommandé de spécifier un profil de provisionnement spécifique:

Xcode 7 et inférieur

Vous devez éviter de cliquer sur le bouton `Fix Issue` (il existe un plug - in Xcode qui désactive le bouton), car il annule parfois les certificats existants et, avec eux, les profils de configuration.

Malheureusement, vous ne pouvez pas spécifier le nom du profil d'approvisionnement dans Xcode 7. Au lieu de cela, vous pouvez spécifier l'UUID du profil, qui change à chaque fois que le profil est re-généré (par exemple lorsque vous ajoutez un nouveau périphérique).

Pour contourner ce problème, consultez XcodeProject.md sur la façon de transmettre un profil

d'approvisionnement à Xcode lors de la création de votre application.

Xcode 8 et plus

Apple a beaucoup amélioré la signature du code avec la sortie de Xcode 8, les modifications suivantes ont été apportées:

- Fini le bouton `Fix Issue` , à la place tous les processus de signature du code s'exécutent en arrière-plan et affichent le journal à droite dans Xcode.
- Vous pouvez maintenant spécifier le profil d'approvisionnement par nom, au lieu de l'UUID (consultez XcodeProject.md pour plus d'informations).
- Messages d'erreur améliorés en cas de problème. Si vous rencontrez des erreurs de signature de code, vous devriez toujours essayer de créer et de signer avec Xcode pour obtenir des informations d'erreur plus détaillées. (Consultez Dépannage.md pour plus d'informations)

Manuellement

Vous pouvez toujours créer et gérer manuellement vos certificats et vos profils d'approvisionnement à l'aide du portail de développeur Apple. Veillez à stocker la clé privée (`.p12`) de vos certificats en lieu sûr, car ils ne peuvent pas être restaurés si vous les perdez.

Vous pouvez toujours télécharger le certificat (`.cer`) et le profil d'approvisionnement (`.mobileprovision`) à partir du portail de développeur Apple.

Si vous révoquez votre certificat ou qu'il expire, tous les profils d'approvisionnement associés seront invalides.

Utiliser la correspondance rapide

Le concept de correspondance est décrit dans le guide de création de codes et constitue l'approche de signature de code recommandée si vous utilisez fastlane.

Avec match, vous stockez vos clés privées et vos certificats dans un référentiel git pour les synchroniser entre les ordinateurs. Cela facilite l'intégration de nouveaux membres et la configuration de nouvelles machines Mac. Cette approche est sécurisée et utilise la technologie que vous utilisez déjà.

Pour commencer avec la correspondance, vous devez révoquer vos certificats existants.

Lire Certificats, profils d'approvisionnement et signature de code en ligne: https://riptutorial.com/fr/xcode/topic/3711/certificats--profils-d-approvisionnement-et-signature-de-code

Chapitre 4: Cours de récréation

Examples

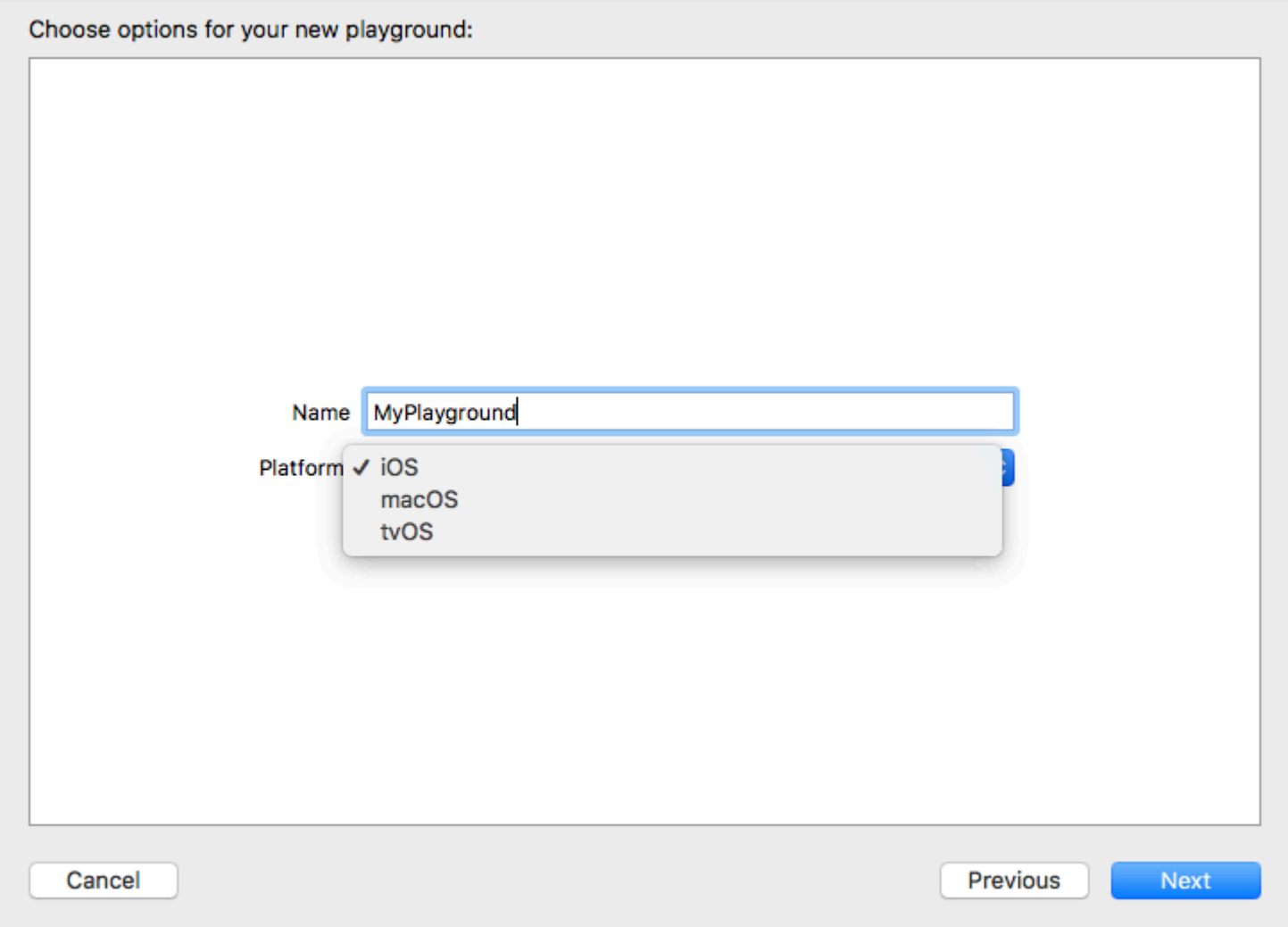
Démarrer avec Playground

1. Créez un nouveau fichier de terrain de jeu:

    - Première option: à partir de l'écran d'accueil Xcode, sélectionnez la première option (**Démarrer avec une aire de jeux**).

    - Deuxième option: Dans le menu, sélectionnez **Fichier** → **Nouveau** → **Terrain de jeu** (N).

2. Nommez votre terrain de jeux et sélectionnez la plate-forme (iOS / macOS / tvOS), puis cliquez sur **Suivant** .



3. Sur l'écran suivant, choisissez où vous voulez enregistrer votre terrain de jeu, puis cliquez sur **Créer** .

Valeur la plus récente, historique des valeurs et graphique

En utilisant Playground, il est facile de voir ce qui se passe à l'intérieur des boucles ou des objets pendant le changement.

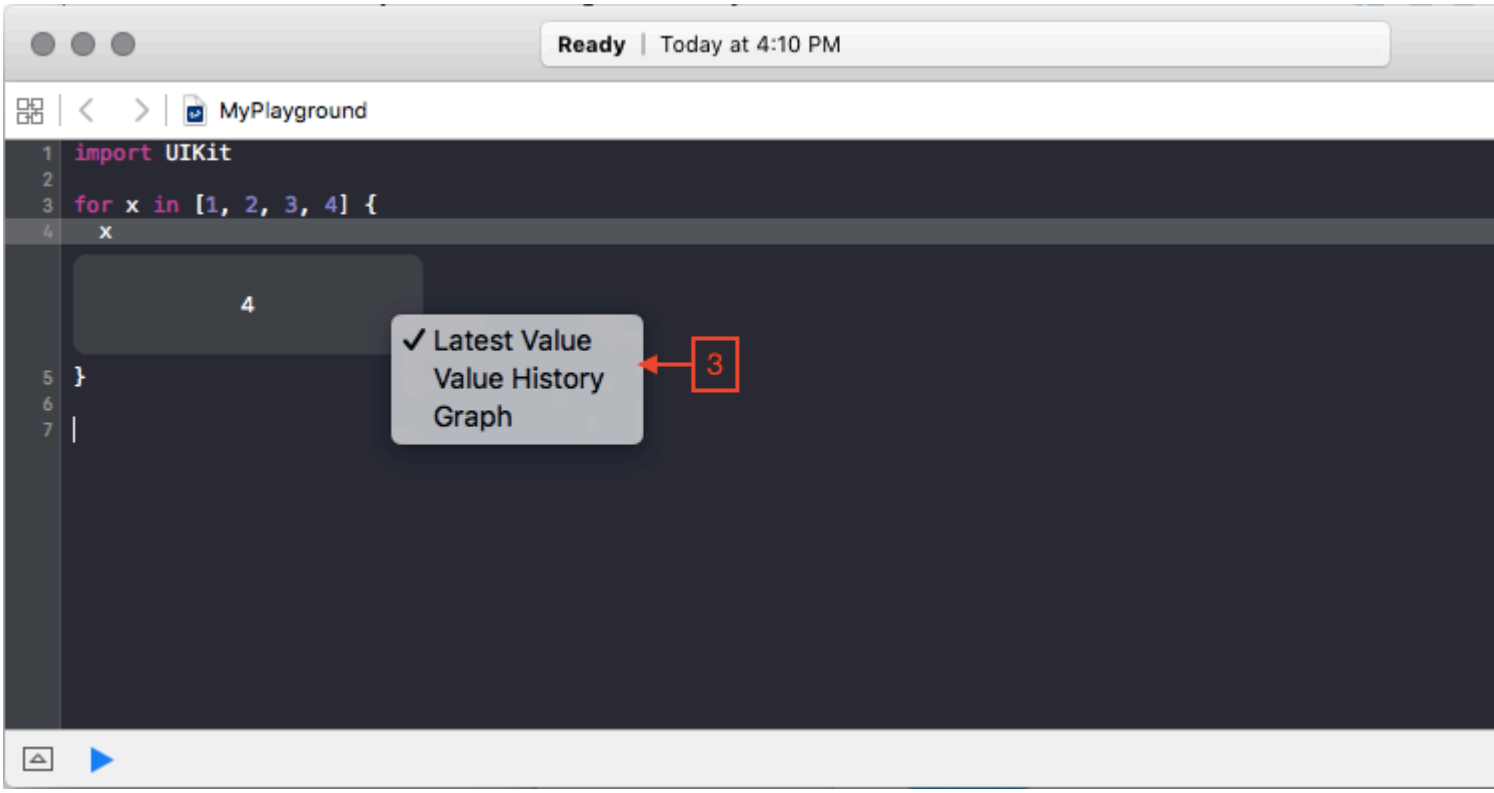
Par exemple, dans le code ci-dessous, la valeur de `x` passera de 1 à 4.

```
import UIKit
for x in [1, 2, 3, 4] {
  x
}
```

(1) En cliquant sur le symbole de l'œil à droite, vous aurez un aperçu rapide.

(2) Cliquez sur le cercle situé à côté pour afficher la *dernière valeur* sous la ligne.

(3) Un clic droit sur la vue ajoutée affichera un menu déroulant avec la **dernière valeur, l'historique des valeurs et le graphique**



Ajout d'images, de données statiques, de sons, etc. à un terrain de jeu

Les images, les données statiques, les sons, etc. sont des ressources dans un terrain de jeu.

1. Si le navigateur de projet est masqué, choisissez Affichage> Navigateurs> Afficher le navigateur de projet (1)
2. Il y a plusieurs façons d'ajouter des fichiers
   - Faites glisser vos ressources vers le dossier `Resources` ou
   - Sélectionnez le dossier Ressources et choisissez Fichier> Ajouter des fichiers à "Ressources" ou
   - Cliquez sur le dossier Ressources en maintenant la touche Contrôle enfoncée et choisissez Ajouter des fichiers à «Ressources»
3. Utilisez votre ressource. Par exemple, `let i = UIImage(named: "tacos.jpg")`

Lire Cours de récréation en ligne: https://riptutorial.com/fr/xcode/topic/1236/cours-de-recreation

Chapitre 5: Création de contrôles personnalisés dans Interface Builder avec @IBDesignable

Remarques

Il est devenu beaucoup plus facile de créer des contrôles personnalisés dans Interface Builder avec l'introduction des directives `@IBDesignable` et `@IBInspectable` dans Swift. Les développeurs peuvent désormais créer des contrôles riches, complexes et entièrement animés en utilisant seulement quelques lignes de code supplémentaires. Je suis surpris par le nombre de développeurs qui n'ont pas encore adopté cette fonctionnalité, et je trouve fréquemment que l'ajout de quelques lignes de code aux classes existantes peut les rendre beaucoup plus faciles à utiliser.

Notez que ces fonctionnalités sont également disponibles dans Objective-C et constituent un excellent moyen de donner vie aux anciennes classes. Les équivalents syntaxiques d'Objective-C sont IB_DESIGNABLE et IBInspectable, mais pour l'instant je vais me concentrer sur des exemples de Swift.

Examples

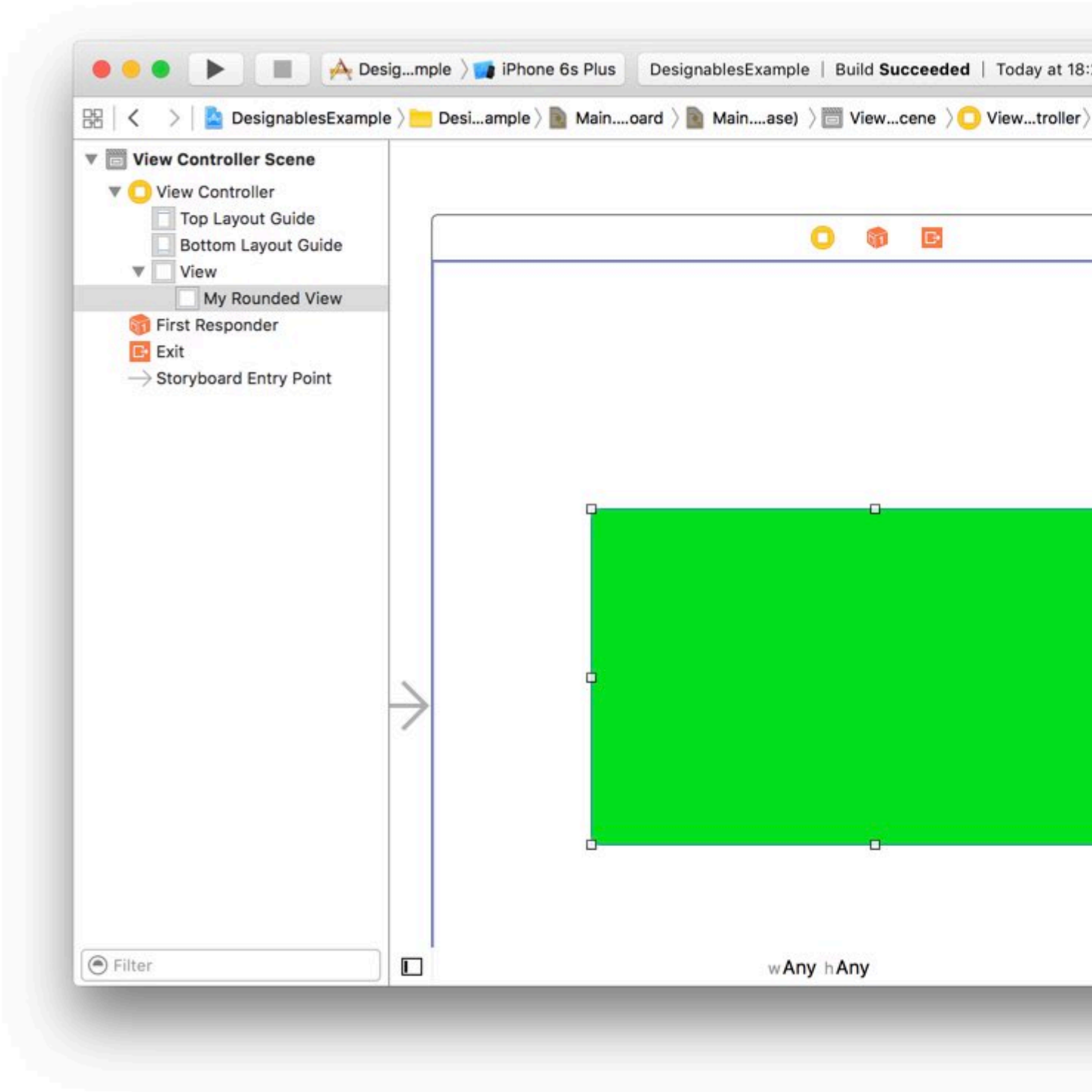
Une vue arrondie à rendu direct

C'est une exigence si commune dans le développement iOS, et c'était toujours quelque chose qui devait être fait purement en code (ou en utilisant des images - beurk!). Maintenant, il est incroyablement facile de prévisualiser ce genre de chose dans Interface Builder, il n'y a absolument aucune excuse pour ne pas l'utiliser.

Voici le code: -

```
import UIKit

@IBDesignable
class MyRoundedView: UIView {

    @IBInspectable var radius: CGFloat = 8 {
        didSet {
            self.layer.cornerRadius = radius
        }
    }

    override func awakeFromNib() {
        self.layer.cornerRadius = self.radius
        self.layer.masksToBounds = true
    }
}
```

Pour utiliser cette classe, ajoutez-la à votre projet, puis ouvrez le storyboard dans IB et créez un UIView normal d'une taille décente. Donnez-lui une couleur d'arrière-plan pour que vous puissiez le voir, puis accédez à Identity Inspector dans le panneau Utilitaires de droite et modifiez le type de classe dans la liste déroulante à `MyRoundedView` .

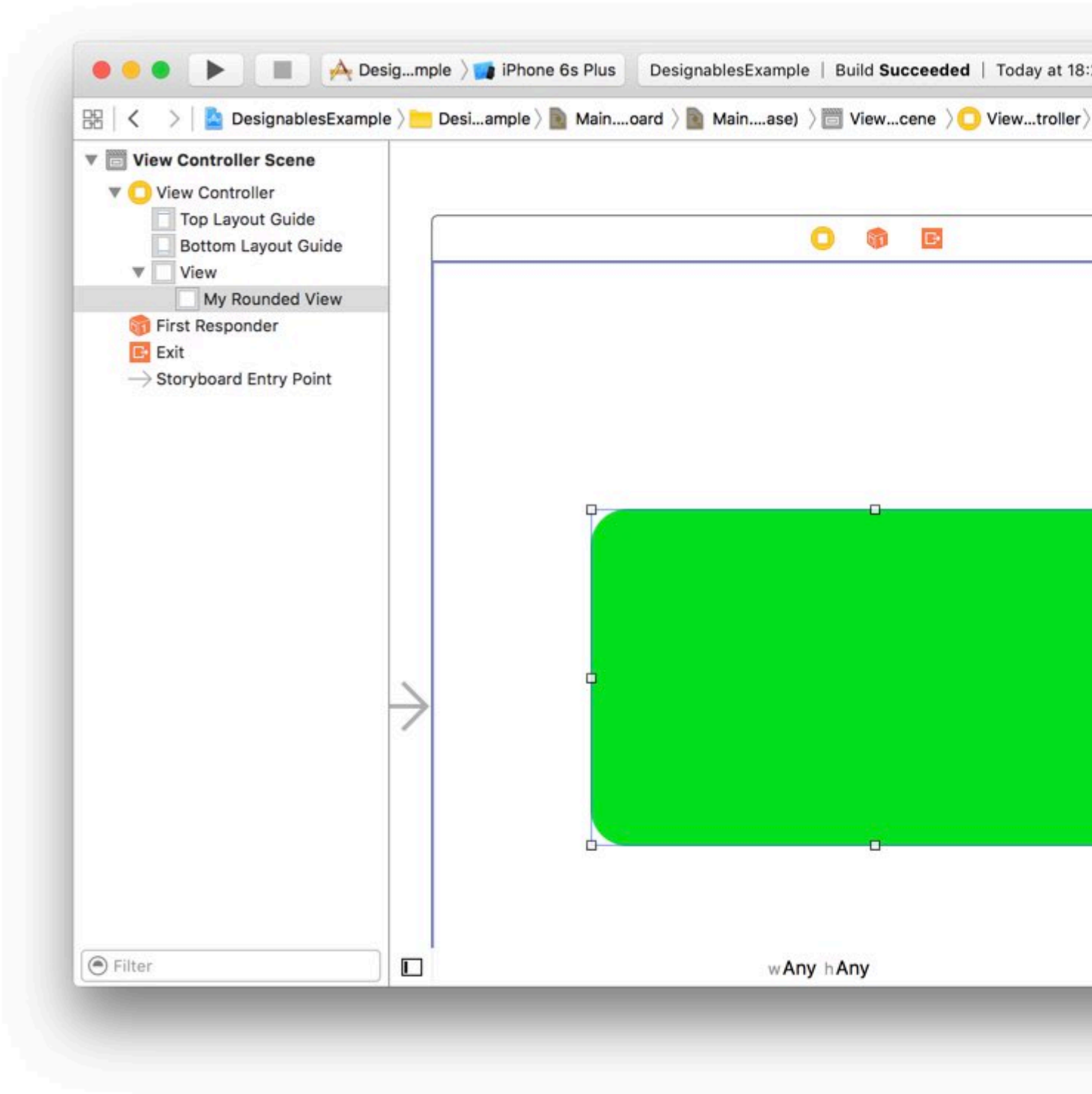


Lorsque vous faites cela, vous devriez voir une troisième étiquette apparaissant sous "Class" et "Module" qui indique "Designables", et devrait indiquer "Mise à jour" pendant un moment avant de passer à "Mise à jour". Cela signifie que Xcode a `MyRoundedView` recompilé votre code pour `MyRoundedView` .

Vous pouvez maintenant ouvrir l'inspecteur d'attributs et vous devriez voir (peut-être après une

courte pause) une nouvelle section en haut du volet avec l'en-tête "Ma vue arrondie" et un nouvel attribut intitulé "Rayon" avec la valeur 8 (parce que est la valeur initiale que nous définissons dans le code). Cela est apparu dans l'inspecteur d'attributs car nous l'avons marqué comme `@IBInspectable` .

Vous pouvez maintenant changer cela pour un autre numéro et vous devriez voir la mise à jour du rayon de coin de la vue arrondie en temps réel!



Lire Création de contrôles personnalisés dans Interface Builder avec @IBDesignable en ligne: https://riptutorial.com/fr/xcode/topic/6193/creation-de-controles-personnalises-dans-interface-builder-avec--ibdesignable

Chapitre 6: Développement multiplateforme

Examples

CibleConditionals

L'en-tête du système `TargetConditionals.h` définit plusieurs macros que vous pouvez utiliser depuis C et Objective-C pour déterminer la plate-forme que vous utilisez.

```
#import <TargetConditionals.h>  // imported automatically with Foundation

- (void)doSomethingPlatformSpecific {
#if TARGET_OS_IOS
    // code that is compiled for iPhone / iPhone Simulator
#elif TARGET_OS_MAC && !TARGET_OS_IPHONE
    // code that is compiled for OS X only
#else
    // code that is compiled for other platforms
#endif
}
```

Les valeurs des macros sont:

7.0

Lorsque vous utilisez les SDK iOS 9.1, tvOS 9.0, watchOS 2.0, OS X 10.11 ou plus récent:

Macro	Mac	iOS	simulateur iOS	Regarder	Simulateur de montre	la télé	Simulateur de télévision
`TARGET_OS_MAC`	**1**	**1**	**1**	**1**	**1**	**1**	**1**
`TARGET_OS_IPHONE`	0	**1**	**1**	**1**	**1**	**1**	**1**
`TARGET_OS_IOS`	0	**1**	**1**	0	0	0	0
`TARGET_OS_WATCH`	0	0	0	**1**	**1**	0	0
`TARGET_OS_TV`	0	0	0	0	0	**1**	**1**
`TARGET_OS_SIMULATOR`	0	0	**1**	0	**1**	0	**1**
`TARGET_OS_EMBEDDED`	0	**1**	0	**1**	0	**1**	0
`TARGET_IPHONE_SIMULATOR`	0	0	**1**	0	**1**	0	**1**

7.0

Lorsque vous utilisez les kits de développement iOS 8.4, OS X 10.10 ou plus anciens:

Macro	Mac	iOS	simulateur iOS
`TARGET_OS_MAC`	**1**	**1**	**1**
`TARGET_OS_IPHONE`	0	**1**	**1**
`TARGET_OS_EMBEDDED`	0	**1**	0
`TARGET_IPHONE_SIMULATOR`	0	0	**1**

Lire Développement multiplateforme en ligne:
https://riptutorial.com/fr/xcode/topic/358/developpement-multiplateforme

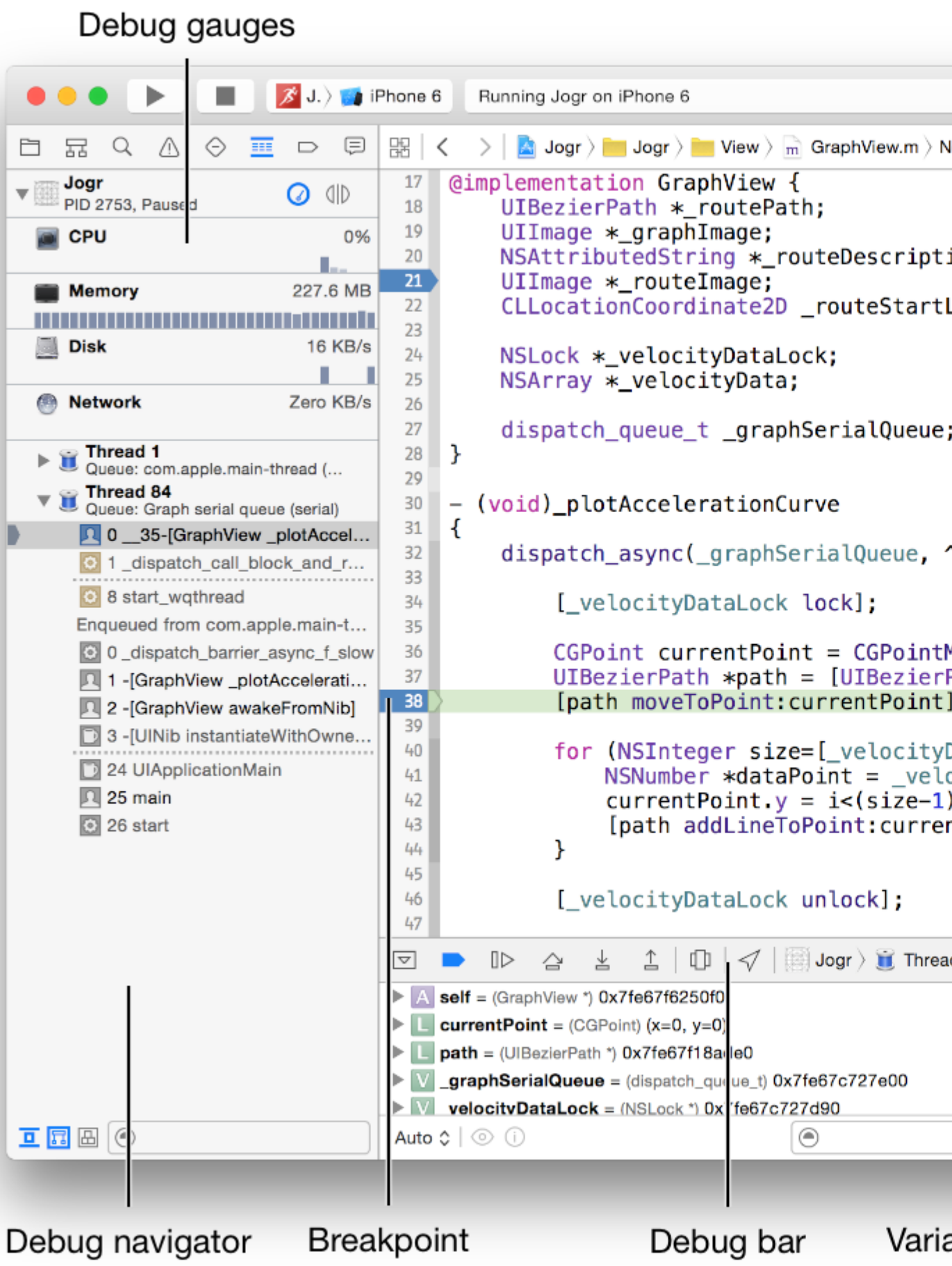
Chapitre 7: Le débogage

Examples

Points d'arrêt

Dans xcode, les développeurs peuvent suspendre / interrompre l'exécution de l'application en cours d'exécution et examiner l'état du programme.

Voici comment mettre en pause des programmes en cours d'exécution:
Ouvrez simplement n'importe quel fichier dans lequel vous voulez placer un point d'arrêt et cliquez sur la ligne de gouttière à gauche où vous voulez suspendre l'exécution.



Nous avons donc placé des points d'arrêt sur les lignes 21 et 38; lorsque l'exécution atteint la ligne

38, Xcode a interrompu l'exécution et indiqué une ligne verte sur cette ligne.

Les jauges de débogage nous donnent un aperçu de l'utilisation du processeur, de l'utilisation de la mémoire et, au fond, de la pile d'exécution avec les noms de threads et de fonctions. Nous pouvons savoir quelle pile ou séquence de fonctions mène à cette ligne de rupture.

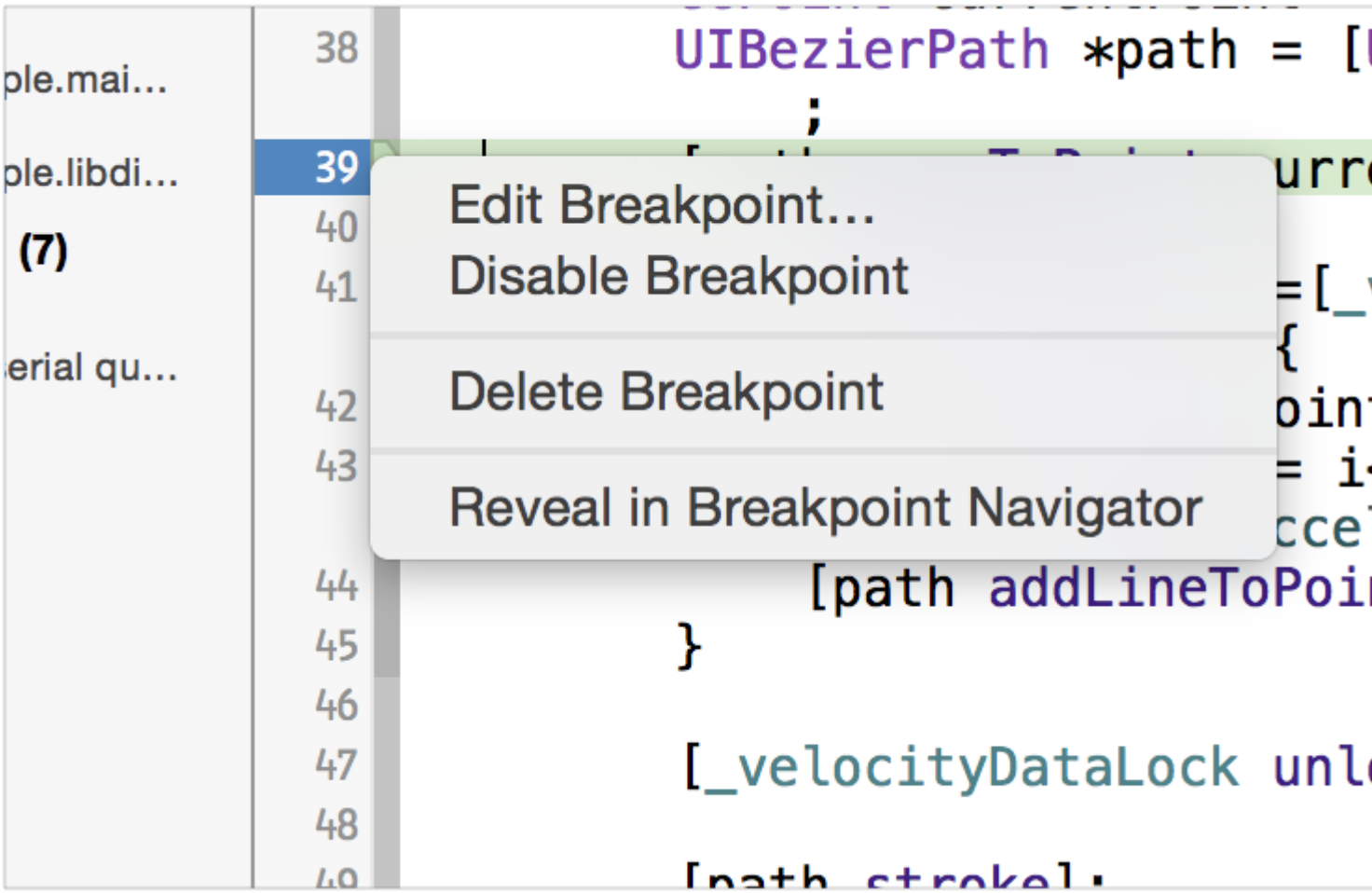
La vue Variables donne tous les détails des états et des valeurs de toutes les variables du périmètre de la ligne panée. Nous pouvons voir leurs valeurs, adresses mémoire, propriétés des instances et leurs détails.

La console peut être utilisée pour imprimer la valeur de toute variable qui est dans la portée. En utilisant la commande `po` nous pouvons y parvenir.

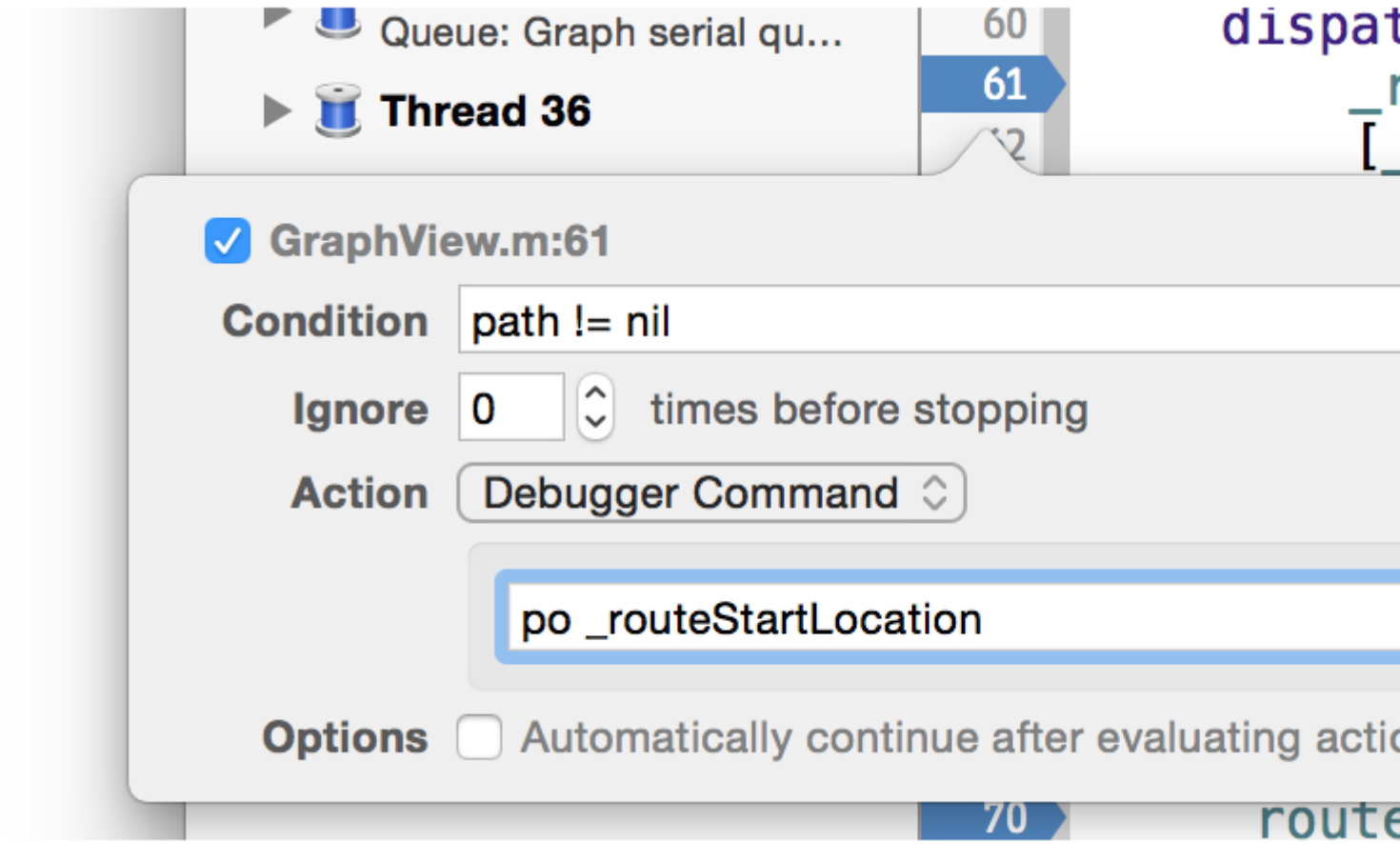
La barre de débogage a des contrôles pour les points d'arrêt.

- Le premier bouton permet d'activer / désactiver le point d'arrêt suspendu.
- Deuxième bouton utilisé pour mettre en pause / reprendre l'exécution des programmes
- Le troisième est le bouton Step-Over utilisé pour exécuter la ligne suivante
- Quatrième bouton dans Step-In utilisé pour entrer dans la fonction en cours d'exécution
- Cinquième est le bouton Step-Out pour sortir de la fonction actuelle

Configurez le point d'arrêt:
Nous pouvons même avoir plus de contrôle sur les points d'arrêt.



Supprimer sont Désactiver les fonctions simples.
Reveal in Navigator nous amène au navigateur Breakpoint où tous les points d'arrêt du projet sont répertoriés en tant que Navigateur de fichiers.
Edit Breakpoint est quelque chose que nous devrions utiliser plus souvent pour un débogage détaillé. Nous pouvons configurer des points d'arrêt en utilisant cette fonction. Nous pouvons définir des conditions et des actions pour les points d'arrêt en tant que:



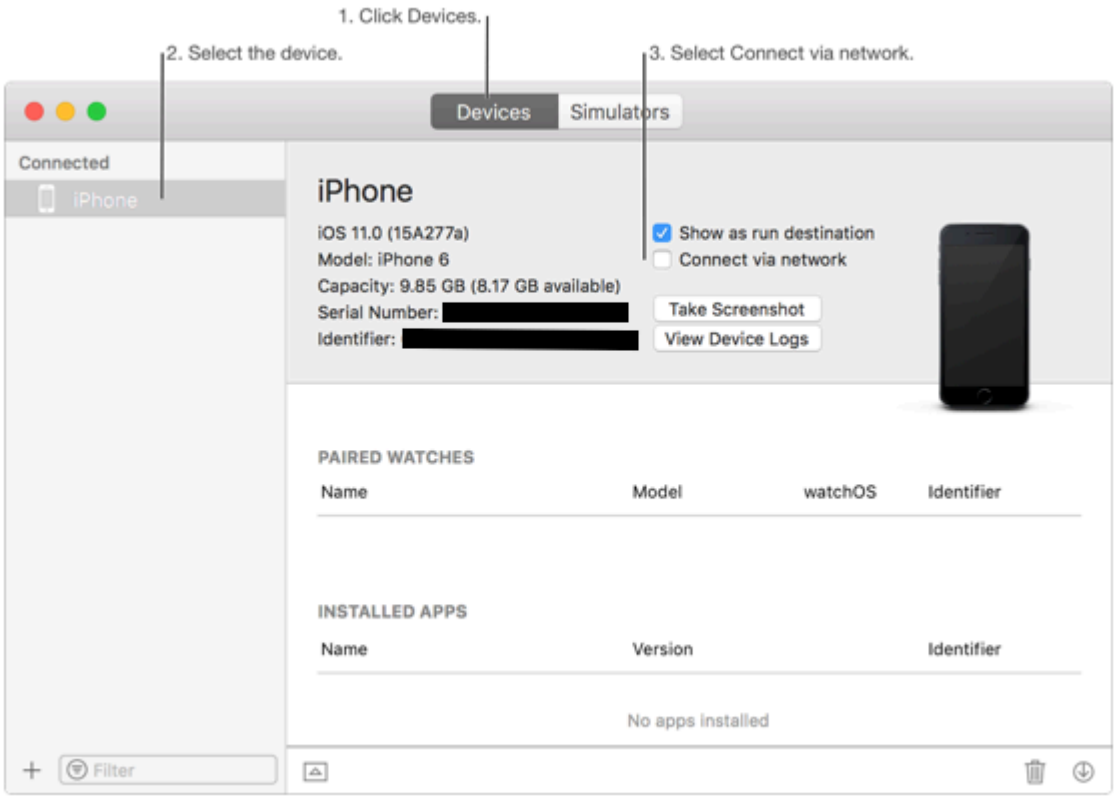
Comme indiqué dans l'image, ce point d'arrêt ne sera mis en pause que si `path != nil` . Si cette condition est vraie, l'action `po _routeStartLocation` est exécutée et mentionné précédemment `po` affichera la valeur de `_routeStartLocation` sur la console.

Formulaire explication détaillée, suivez ce lien détaillé .

Débogage sans fil dans Xcode-9

Comme récemment Apple a sorti iOS11 et Xcode-9, nous pouvons maintenant déboguer des applications sur des appareils sans connecter de périphériques à Xcode via USB.
Nous pouvons tirer parti de la fonctionnalité de débogage sans fil ajoutée à ce Xcode-9.

Pour activer le débogage sans fil, nous devons configurer certaines étapes dans Xcode.
1 Commencez par connecter le périphérique exécutant iOS11 au Xcode-9.
2 Allez dans Fenêtre> Périphériques et simulateurs dans les menus Xcode et sous Périphérique connecté, le périphérique connecté sera répertorié.
3 Cochez ensuite la case à cocher Connect vis réseau comme sur cette image:



(Courtoisie d' image: poste de SO de Surjeets)

4 Déconnectez ensuite votre appareil du câble USB, assurez-vous que les appareils iPhone / iPad / iPod et Mac exécutant Xcode sont sur le même réseau sans fil.
5 Dans Xcode, vous verrez ces appareils listés et vous pourrez directement exécuter votre application sur cet appareil.

Nous pouvons effectuer toutes les opérations avec Xcode sur cet appareil, comme s'il était connecté via USB; sauf que nous **ne pouvons pas voir les journaux si l'** application est exécutée à l'aide de Xcode, la placer en arrière-plan et suspendue en arrière-plan et nous la relançons. Ceci est possible avec le débogage USB.

REMARQUES:
1 Nous devons utiliser Xcode-9, iOS 11 sur le périphérique
2 L'appareil et le Mac doivent se trouver sur le même réseau sans fil

Lire Le débogage en ligne: https://riptutorial.com/fr/xcode/topic/10459/le-debogage

Chapitre 8: Outils de ligne de commande

Examples

Tests en cours

Pour exécuter vos tests unitaires dans le simulateur à l'aide de l'utilisation de `xcodebuild`

Si vous avez un espace de travail (par exemple lors de l'utilisation de CocoaPods)

```
xcodebuild \
  -workspace MyApp.xcworkspace \
  -scheme "MyScheme" \
  -sdk iphonesimulator \
  -destination 'platform=iOS Simulator,name=iPhone 6,OS=9.1' \
  test
```

Si vous avez un fichier de projet

```
xcodebuild \
  -project MyApp.xcproj \
  -scheme "MyScheme" \
  -sdk iphonesimulator \
  -destination 'platform=iOS Simulator,name=iPhone 6,OS=9.1' \
  test
```

Les valeurs de `destination` alternatives sont

```
  -destination 'platform=iOS,id=REAL_DEVICE_UDID'
  -destination 'platform=iOS,name=IPHONE NAME'
```

Liste des cibles disponibles, des schémas et des configurations de construction

Pour répertorier tous les schémas disponibles pour le projet dans votre répertoire actuel

```
xcodebuild -list
```

Vous pouvez éventuellement passer un chemin vers un projet ou un fichier d'espace de travail

```
xcodebuild -list -workspace ./MyApp.xcworkspace
xcodebuild -list -project ./MyApp.xcodeproj
```

Exemple de sortie

```
Information about project "Themoji":
    Targets:
        Themoji
```

```
        ThemojiUITests
        Unit

    Build Configurations:
        Debug
        Release

    If no build configuration is specified and -scheme is not passed then "Release" is used.

    Schemes:
        Themoji
        ThemojiUITests
        Units
```

Compiler et signer le schéma

Nettoyage et compilation du code pour iPhone, sur le projet MyProject for schema Qa:

```
xcrun xcodebuild clean \
    -workspace "MyProject.xcworkspace" \
    -scheme "YourScheme" \
    -sdk iphoneos \
    -configuration Debug \
    archive \
    -archivePath builds/MyProject.xcarchive
```

La configuration peut être `Debug` ou `Release` .

Signer le code précédemment compilé:

```
xcrun xcodebuild -exportArchive \
    -archivePath builds/MyProject-Qa.xcarchive \
    -exportOptionsPlist config.plist \
    -exportPath builds
```

`config.plist` contient les informations sur la manière de créer et de signer l'application, car les versions de développement utilisent:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-
1.0.dtd">
<plist version="1.0">
<dict>
        <key>method</key>
        <string>development</string>
        <key>uploadSymbols</key>
        <true/>
</dict>
</plist>
```

Un plist de publication App Store devrait contenir quelque chose comme:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-
```

```
1.0.dtd">
<plist version="1.0">
<dict>
        <key>teamID</key>
        <string>xxxxxxxxxxx</string>
        <key>method</key>
        <string>app-store</string>
        <key>uploadSymbols</key>
        <true/>
</dict>
</plist>
```

Où l'identifiant de l'équipe peut être obtenu à partir de votre trousseau.

Tous les paramètres disponibles

- `compileBitcode`
- `embedOnDemandResourcesAssetPacksInBundle`
- `iCloudContainerEnvironment`
- `manifest`
- `method`
- `onDemandResourcesAssetPacksBaseURL`
- `teamID`
- `thinning`
- `uploadBitcode`
- `uploadSymbols`

Pour obtenir plus d'informations sur chacun des paramètres, lancez `xcodebuild --help`

Accédez à n'importe quel outil de ligne de commande dans le regroupement d'applications Xcode (xcrun)

`xcrun` utilise la version Xcode par défaut du système (définie via `xcode-select`) pour localiser et exécuter les outils de ligne de commande à partir du lot d'applications Xcode, par exemple, llvm-cov.

```
# Generate code coverage reports via llvm-cov
# /Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/bin
xcrun llvm-cov [parameters]

# Execute xcodebuild
# /Applications/Xcode.app/Contents/Developer/usr/bin
xcrun xcodebuild [parameters]

# Use Xcode's version of git, e.g., if you have installed a newer version
# /Applications/Xcode.app/Contents/Developer/usr/bin
xcrun git [parameters]
```

Changer les outils de ligne de commande avec xcode-select

Imprimer le chemin d'accès au répertoire développeur actif (Xcode sélectionné)

```
xcode-select -p
```

Sélectionnez une autre version de Xcode, par exemple Beta

```
sudo xcode-select -s /Applications/Xcode-beta.app
```

Réinitialiser à la version par défaut de Xcode

```
sudo xcode-select -r
```

Cela équivaut à exécuter `sudo xcode-select -s /Applications/Xcode.app`

Pour plus de détails: `man xcode-select`

Lire Outils de ligne de commande en ligne: https://riptutorial.com/fr/xcode/topic/2158/outils-de-ligne-de-commande

Chapitre 9: Personnalisation de l'IDE Xcode

Introduction

Ceci est une collection de différents trucs et astuces, pour personnaliser et améliorer votre IDE Xcode

Examples

Terminal ouvert dans le dossier de projet Xcode actuel

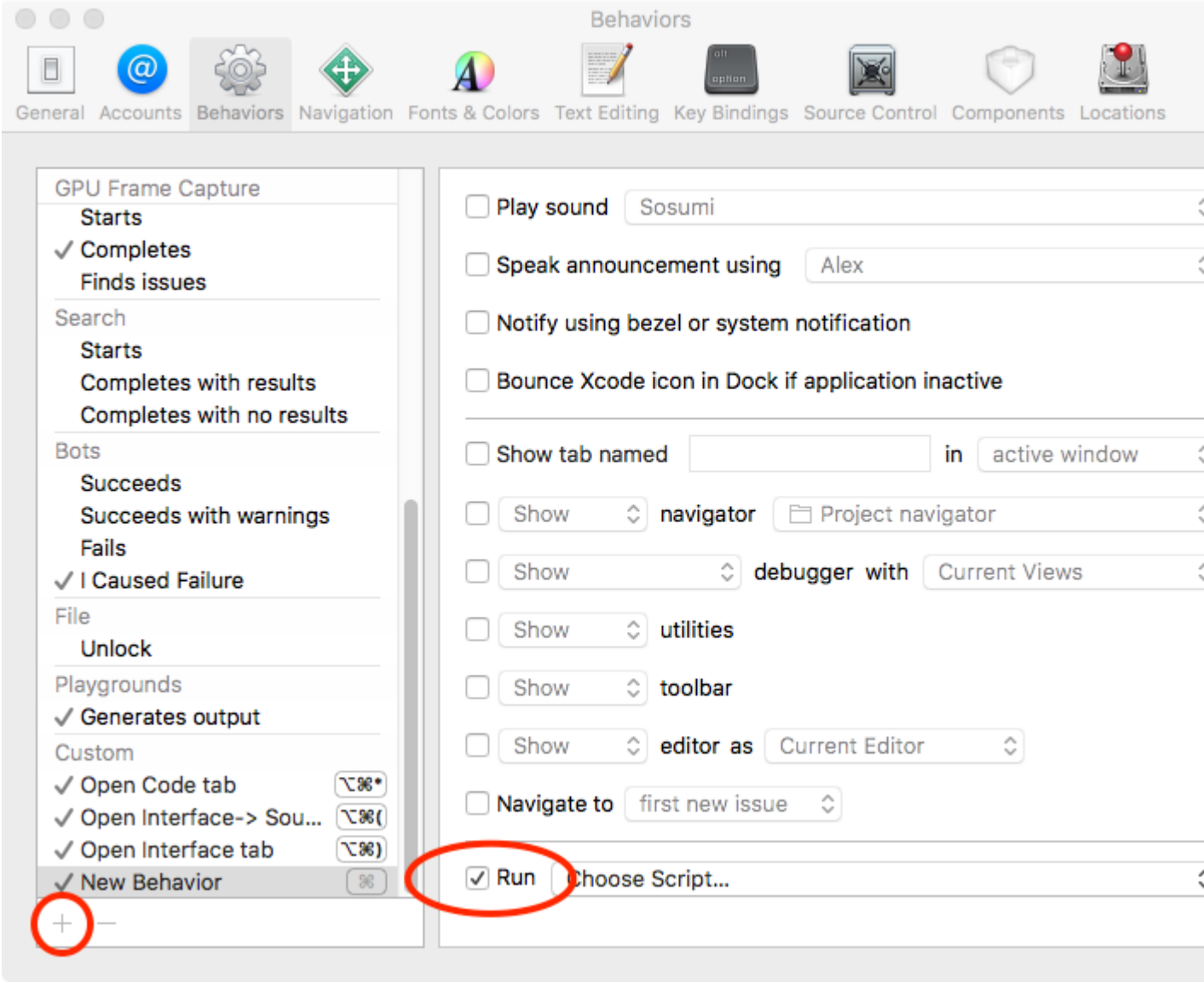
Xcode peut exécuter n'importe quel script avec un raccourci clavier.

Voici un exemple d'affectation des touches de raccourci ⌘+⌥+^+⇧+T pour ouvrir l'application Terminal dans le dossier du projet en cours.

1. Créer un script bash et l'enregistrer dans un dossier

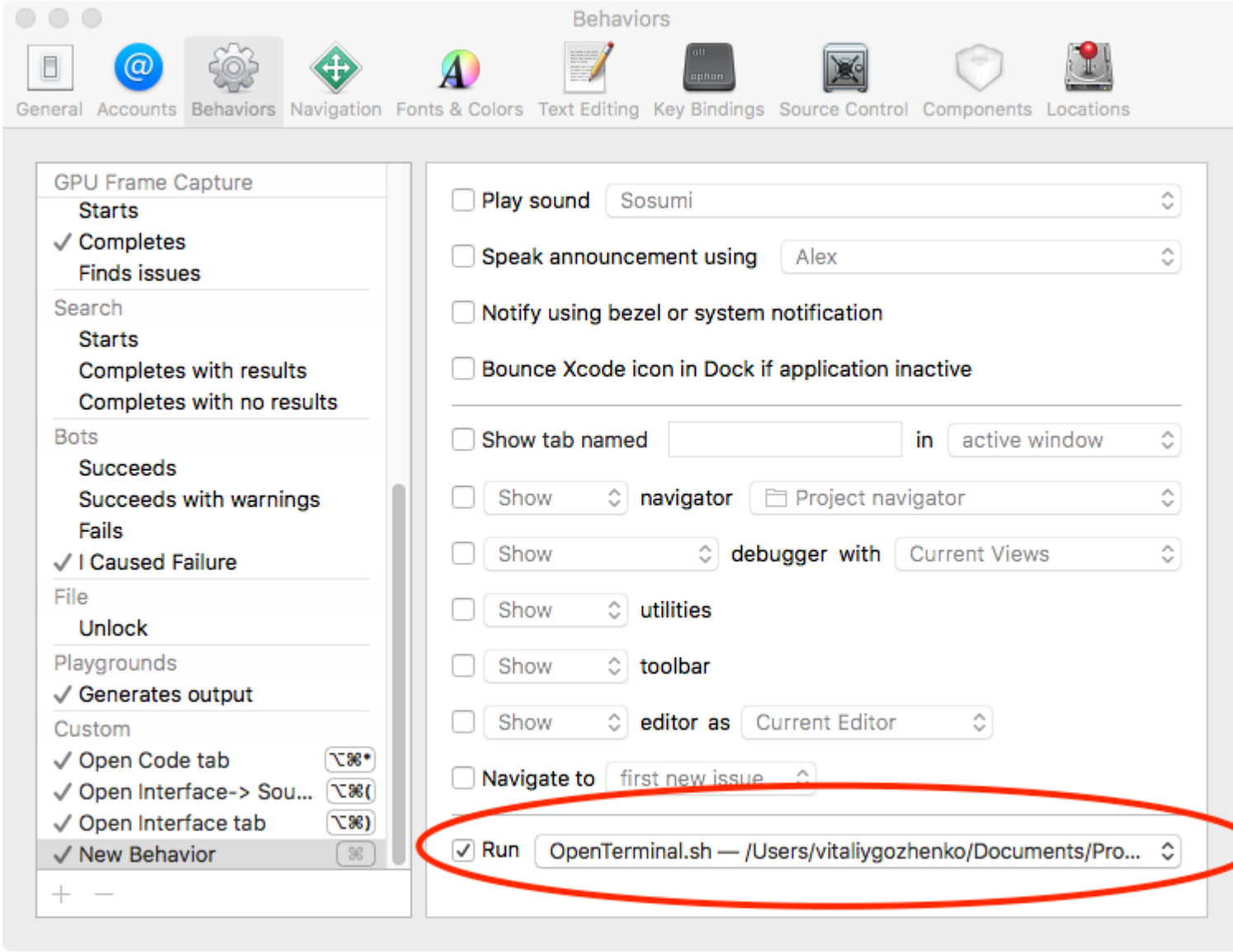
```
#!/bin/bash

# Project Name:  $XcodeProject
# Project Dir:   $XcodeProjectPath
# Workspace Dir: $XcodeWorkspacePath

open -a Terminal "$(dirname $XcodeProjectPath)"
```

2. Rendre le script exécutable: ouvrir Terminal at script folder et exécuter `chmod +x your_script_name.sh`
3. Ouvrir les préférences Xcode dans l'onglet Comportements
4. Ajouter un nouveau comportement personnalisé en appuyant sur `+` dans le coin inférieur gauche
5. Vérifiez `Run` l' action à droite

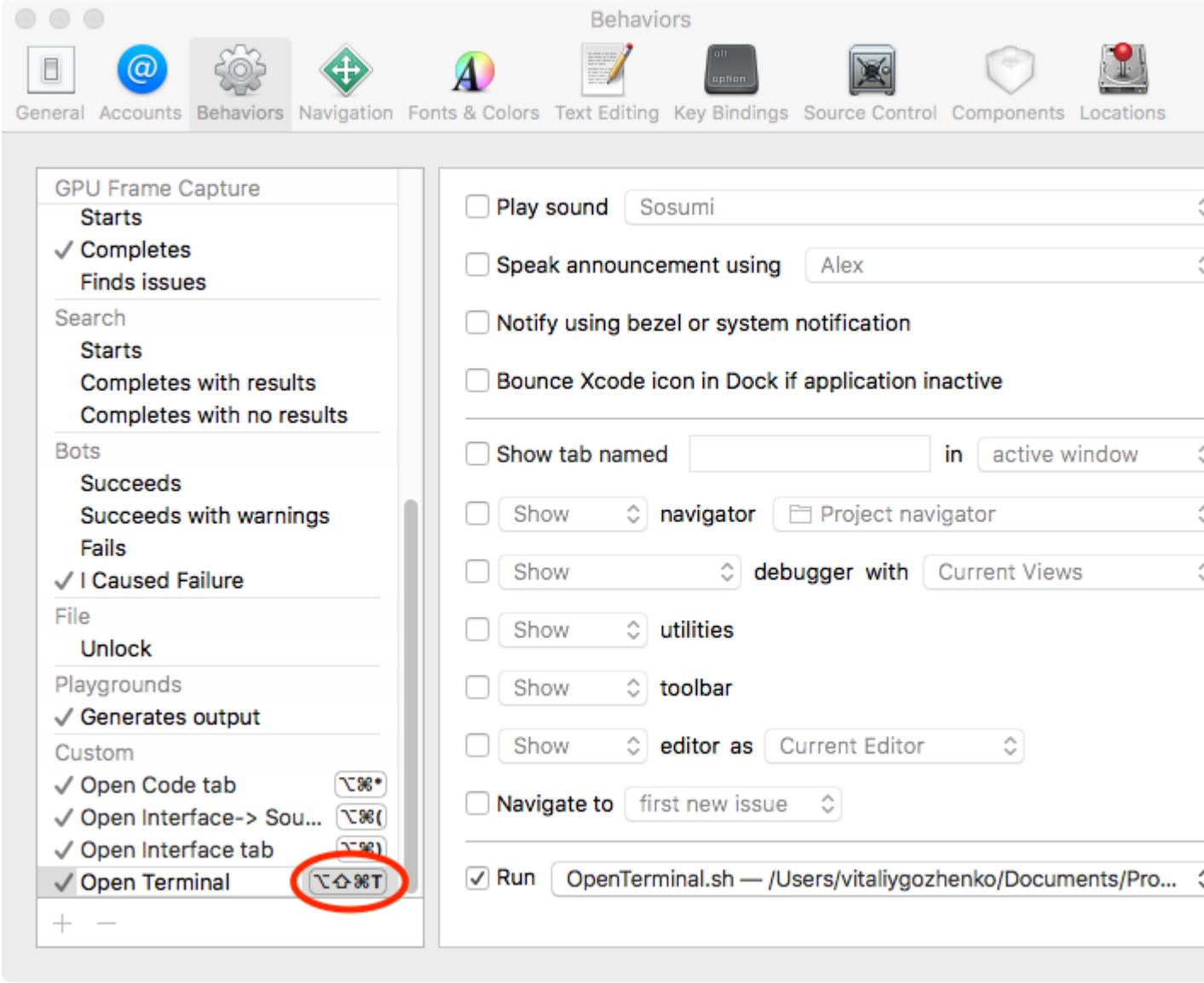


6. Choisissez le script que vous créez précédemment en cliquant deux fois sur le `Choose Script...`.

Si votre script est grisé, assurez-vous que vous exécutez `chmod +x` sur votre fichier de script.



7. Attribuer un raccourci clavier (par exemple ⌘+⌥+⌃+⇧+T) à votre comportement et le renommer



Maintenant, vous pouvez ouvrir le terminal dans le dossier du projet avec un raccourci clavier.

Ceci n'est qu'un exemple d'utilisation des comportements Xcode, mais vous pouvez créer n'importe quel script et lancer n'importe quelle application avec lui.

Auteur du script Bash: http://mattorb.com/xcode-behaviors-for-fun-and-profit/

Effacer les données dérivées avec raccourci clavier

De la même manière que dans `Open Terminal in current Xcode project folder` exemple de `Open Terminal in current Xcode project folder` , vous pouvez ajouter un répertoire de données dérivées avec un raccourci clavier.

Créez un comportement personnalisé (suivez les étapes de Open Terminal dans le dossier du projet Xcode en cours). Mais utilisez un autre script.

Texte du script:

```
#!/bin/bash
```

```
rm -rf $HOME"/Library/Developer/Xcode/DerivedData/"
```

Lire Personnalisation de l'IDE Xcode en ligne:
https://riptutorial.com/fr/xcode/topic/8260/personnalisation-de-l-ide-xcode

Chapitre 10: Projets et espaces de travail

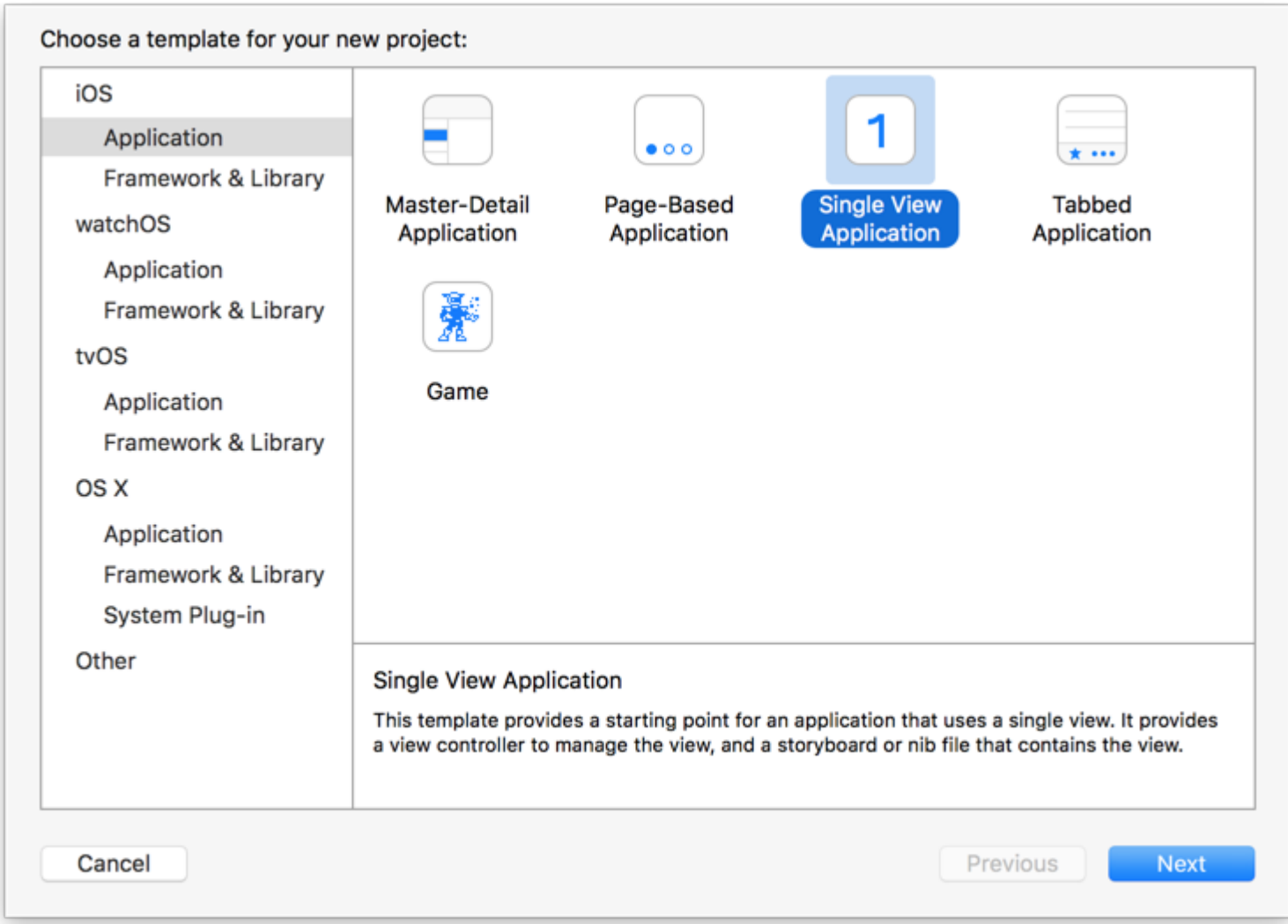
Examples

Aperçu des projets

Les **projets** Xcode sont utilisés pour organiser les fichiers sources, les dépendances de bibliothèque et d'autres ressources, ainsi que les paramètres et les étapes nécessaires à la création des produits du projet. **Les espaces de travail** sont des groupes de projets et d'autres fichiers.

Créer un projet

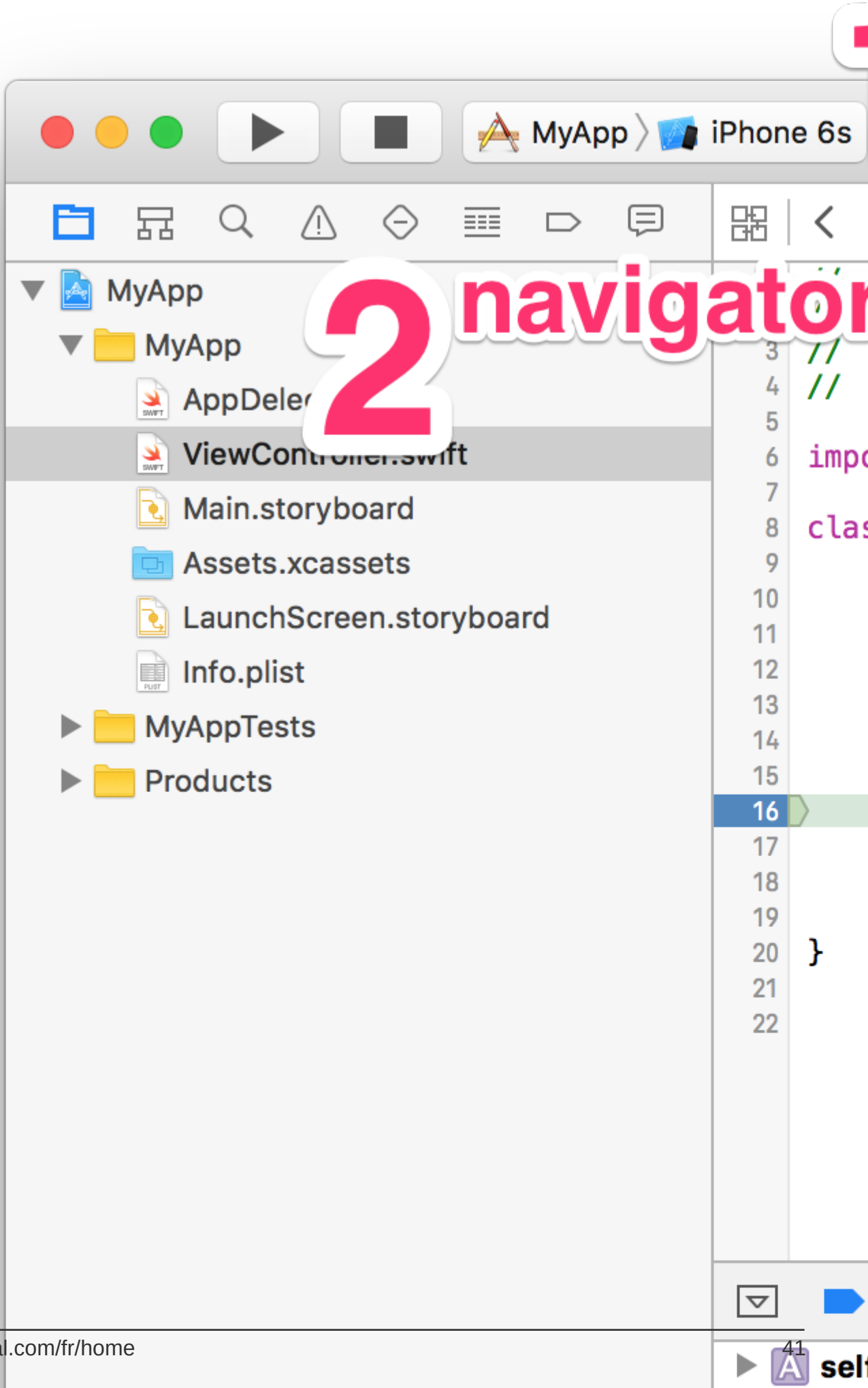
Vous pouvez créer un **nouveau projet** (N) à partir de plusieurs modèles intégrés:



Travailler avec des projets

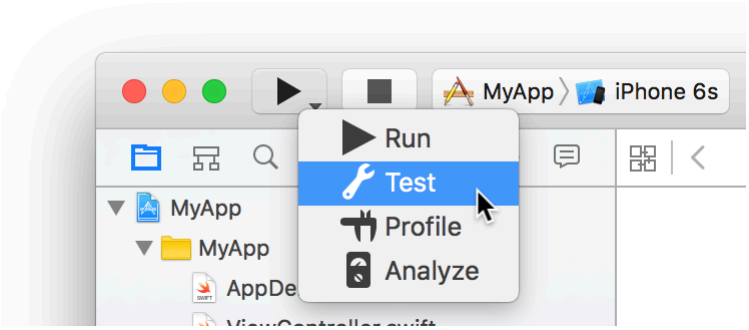
Une fenêtre de projet Xcode comprend:

1. **Barre d'outils** (en haut)
2. **Navigateur** (à gauche)
3. **Éditeur** (centre)
4. **Inspecteur** (à droite)
5. **Affichage des variables** (milieu inférieur gauche)
6. **Sortie de la console** (milieu inférieur droit)
7. **Bibliothèque** (en bas à droite)

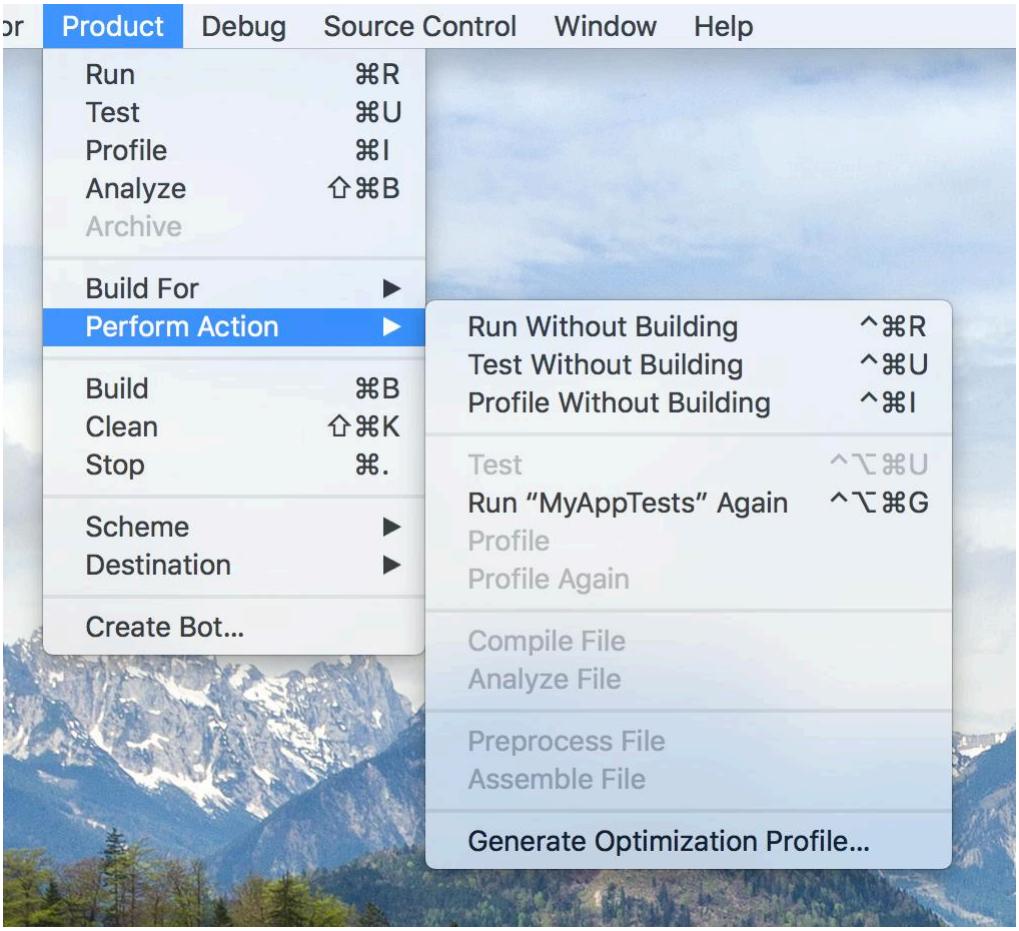

MyApp
iPhone 6s
MyApp
MyApp
AppDele
ViewCont
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
MyAppTests
Products
2
navigator
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
//
//
impo
clas
}

R) pour créer et exécuter votre projet. Cliquez sur Arrêter (ou appuyez sur .) Pour arrêter l'exécution.

Cliquez et maintenez enfoncé pour voir les autres actions, Test (U), Profile (I) et Analyser (B). Maintenez les touches de modification ⌥ option , ⇧ shift et ^ pour plus de variantes.



Les mêmes actions sont disponibles dans le menu Produit:



Ajustez l'espace de travail à vos besoins et naviguez librement

L'une des choses qui peuvent vraiment améliorer votre productivité lors de l'écriture du code est la navigation efficace dans l'espace de travail. Cela signifie également le rendre confortable pour le

moment. Il est possible d'y parvenir en ajustant les zones des espaces de travail que vous voyez.

Les boutons situés en haut des zones de navigation et d'assistant ne sont pas si volumineux et il est difficile de cliquer avec le pointeur de la souris. C'est pourquoi il existe des raccourcis utiles et faciles à retenir qui vous permettent de basculer entre différents onglets ou de masquer la zone.

Vous pouvez basculer entre les différents panneaux de la zone de navigation en maintenant le bouton ⌘ (Commande) enfoncé et en appuyant sur différentes touches numériques de 1 à 8 ou 0. La touche `0` active la présence du navigateur. Voici une liste de raccourcis pour votre commodité:

1. ⌘ + `1` - Navigateur de fichiers;
2. ⌘ + `2` - Navigateur de symboles;
3. ⌘ + `2` - Recherche (également accessible par ⌘ + ⌥ + `F);`
4. ⌘ + `4` - avertissements, les erreurs et les messages d'analyse statique;
5. ⌘ + `5` - Essais disponibles dans le projet;
6. ⌘ + `6` - panneau de la session de débogage;
7. ⌘ + `7` - Les points d' arrêt;
8. ⌘ + `8` - Rapport / Action navigateur histoire;
9. ⌘ + `0` - Affiche / masque le panneau de navigation;

Vous pouvez basculer entre les différents panneaux de la zone d'inspection en maintenant les touches ⌥ et ⌥ `enfoncées` et en appuyant sur les différentes touches numériques de 1 à 6 (selon l'éditeur actif: code, storyboard, xib ou autre type de ressource). Appuyez sur `0` tout en maintenant ces deux boutons enfoncés pour masquer la zone de l'inspecteur.

Donc, si vous éditez un storyboard et que vous avez besoin d'un espace plus visible, appuyez simplement sur ⌘ + `0` et ⌘ + ⌥ + `0` pour obtenir des pixels supplémentaires pour le canevas.

Alors que la commutation des panneaux dépend de la combinaison de ⌘ ou ⌥ et ⌥ , les champs de recherche inférieurs sont activés en maintenant ⌘ et ⌥ et en appuyant sur `j` pour la barre de recherche de la zone de navigation ou `k` pour la barre de recherche.

L'activation de la zone de recherche de navigation peut vous aider à affiner la liste des éléments affichés dans la zone de navigation en fonction du navigateur actif (fichiers de filtrage dans le navigateur de fichiers, symboles du navigateur de symboles, tests dans le navigateur de test, etc.).

L'activation du panneau d'inspection vous aidera à filtrer la liste des modèles de fichiers, des extraits de code, des objets ou des ressources multimédia. Essayez d'utiliser ce champ de recherche lorsque le storyboard est ouvert et que vous devez rapidement trouver des composants `UINavigationItem` ou `UITableViewCell` !

En parlant de bibliothèque, vous pouvez basculer entre les panneaux de bibliothèque (modèles de fichier, fragments de code, bibliothèques d'objets et de médias) ^ + ⌘ + ⌥ et les touches numériques respectives: 1 à 4.

Lire Projets et espaces de travail en ligne: https://riptutorial.com/fr/xcode/topic/335/projets-et-espaces-de-travail

Chapitre 11: Xcode 8 fonctionnalités

Remarques

Cela ne fonctionne qu'avec des projets utilisant Swift 3+

Examples

Littéraux de couleur et d'image

Xcode 8 reconnaît automatiquement toutes les images que vous avez dans un catalogue d'actifs et les propose comme suggestion à l'intérieur d'un initialiseur UIImage.

Ainsi, vous pouvez essentiellement déclarer une nouvelle variable, puis ajouter un nom d'actif que vous avez ajouté à votre catalogue d'actifs. Par exemple, `let img = dog.img` contient maintenant l'image de `dog` dans le catalogue d'actifs.

Sous le capot, il crée un code qui ressemble à ceci: #imageLiteral (resourceName: "dog.png"). Mais en ligne dans l'éditeur de source, vous ne verrez que le nom de fichier de l'image.

Donc, vous pouvez le faire maintenant `imageView.image = img`.

Notez que vous devez cliquer sur la suggestion instellisense pour voir une vignette de l'image dans le code, puis le nom de l'image.

Lire Xcode 8 fonctionnalités en ligne: https://riptutorial.com/fr/xcode/topic/7155/xcode-8-fonctionnalites

Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec Xcode	Anh Pham, Community, Jbryson, jtbandes, Md. Ibrahim Hassan, Undo
2	Astuces Xcode	Anand Prem, Finn Gaida, jtbandes, user459460
3	Certificats, profils d'approvisionnement et signature de code	KrauseFx
4	Cours de récréation	Anh Pham, Idan, Rob Wright
5	Création de contrôles personnalisés dans Interface Builder avec @IBDesignable	Echelon
6	Développement multiplateforme	J F, jtbandes
7	Le débogage	D4ttatraya
8	Outils de ligne de commande	Ali Beadle, David Snabel-Caunt, Fabio, Idan, Jens Meder, KrauseFx
9	Personnalisation de l'IDE Xcode	Vitaliy Gozhenko
10	Projets et espaces de travail	Eimantas, jtbandes
11	Xcode 8 fonctionnalités	Rashwan L, Sharukh Mastan