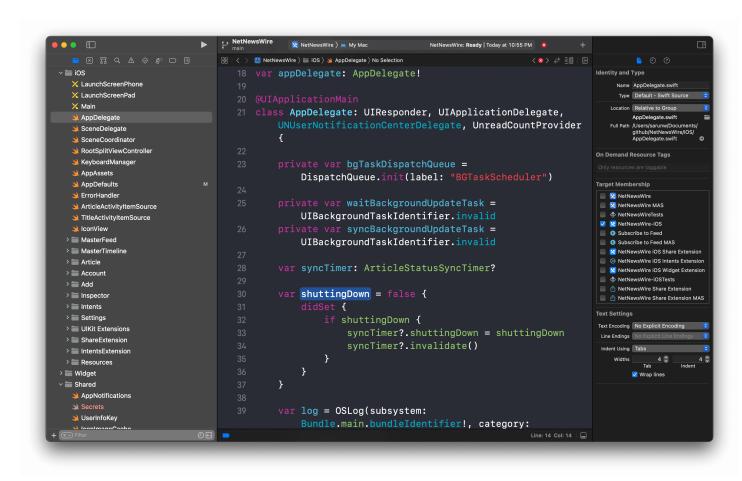
7个大型iOS项目的Xcode快捷方式

随着项目的增长,代码库也在增长。不知不觉中,您将有数百个文件要处理。

在本文中,我将分享七个**Xcode快捷方式,这些快捷方式将帮助您在一个大型iOS项目上**。您即将学习的技巧将帮助您在大大小小的项目中**找到并导航到一个文件/类/符号**。



大型iOS项目的Xcode快捷方式

- **打开快速(**介+光+**O**): 此快捷方式允许您**跳转到项目中的任何类**、枚举、结构、函数或其他实体([1])。
- **在项目导航器中显示 (**① + 出 + J): 这将**滚动到**您在**项目导航器**中工作的文件。
- 项目导航中的筛选文件: 您可以在项目导航中筛选文件, 以仅显示您感兴趣的文件。
- **返回和前进 (**図 + 出 + ← 和 図 + 出 + →): 这将**在文件之间来回导航**在Xcode导航堆栈。
- 跳转到行(+L): 此命令将跳转到当前文件中的特定行。
- 显示文档项(+6): 这将显示一个下拉菜单,其中包含当前文件中的所有符号。
- 小地图: 您可以使用小地图来发现文件中的符号并直接跳转到它。

快速打开: $^{\circ}$ + $^{\circ}$ + O(Shift+Command+O)

我把这个命令放在这里,因为**没有它我就活不下去**。

Open Quickly允许您**跳转到任何**项目中的类、枚举、结构、函数或其他实体([1:1]),使用**模糊搜索查询**。

如何使用Open Quickly

要使用快速打开,我们使用此快捷方式。

-shift+-Command+OorFile menu>Open Quickly...

使用快速打开后,您将看到一个搜索字段弹出。

- 1. 输入一个实体的名称,你想打开。它将执行一个模糊搜索([2]),所以你可以跳过一些单词。
- 2. 您将看到与搜索词匹配的符号列表。
- 3. 您可以使用向上/向下箭头键来突出显示您想要的那个。
- 4. 然后按返回键以打开包含该符号的文件。
- 按返回键在主编辑器(左侧)中打开文件。
- 按-**选项+返回**键在**辅助编辑器**中打开文件。

下面是一个使用Open Quickly打开 Scene代理 类的示例。

```
NetNewsWire
                                           (Wire ) 🚞 iOS 🕽 🔌 Ap
                                                                                            < □> ≠ == □
                                                                                                                ▶ ② ⓒ
                                     import Secrets
import WidgetKit
                                     var appDelegate: AppDelegate!
                                     {\tt class} \ {\tt AppDelegate:} \ {\tt UIResponder, \ UIApplicationDelegate,}
                                         UNUserNotificationCenterDelegate,
                                         UnreadCountProvider {
                                         private var bgTaskDispatchQueue =
    DispatchQueue.init(label: "BGTaskScheduler")
                                 24
                                          private var waitBackgroundUpdateTask =
                                              UIBackgroundTaskIdentifier.invalid
                                          private var syncBackgroundUpdateTask =
                                              UIBackgroundTaskIdentifier.invalid
                                          var syncTimer: ArticleStatusSyncTimer?
                                          var shuttingDown = false {
                                              didSet {
    if shuttingDown {
                                                                                                                     4 🖫
                                                        syncTimer?.shuttingDown = shuttingDown
                                                        syncTimer?.invalidate()
> 🛅 Reso
```

搜索一个类名

这是另一个跳转到 KeyboardManager 文件中 _keyCommands 变量的示例。我们使用 🛛 + ↩ (选项+返回)在辅助编辑器中打开文件。

```
< •> ≠ ≣□ Œ
                          No
<u>=</u> ⊠ ≌ Q ∆
                           import Secrets
import WidgetKit
h NetNewsWire-Bridging-He
                           var appDelegate: AppDelegate!
                           24
                                  UIBackgroundTaskIdentifier.invalid
                                      /ar syncBackgroundUpdateTask =
                                  UIBackgroundTaskIdentifier.invalid
> III MasterFeed
                              var syncTimer: ArticleStatusSyncTimer?
> E Article
                              var shuttingDown = false {
> 🛅 Inspe
                                     if shuttingDown {
                                         syncTimer?.shuttingDown = shuttingDown
                                         syncTimer?.invalidate()
                              }
```

搜索一个变量名

在Project Navigator: ① + \mathfrak{H} + J (Shift+Command+J) 中显示

这是一个Open Quickly命令的同伴。

我通常将相关文件分组在同一个文件夹下。例如,我可能会将模型、视图模型和视图控制器分组在用户文件夹下。

跳转到文件后,您可能需要打开该组中的其他文件。Reveal in Project Navigator命令将**滚动到您正在 处理**的产品导航器中的文件,并**展开文件所在的文件夹**。

```
| Netherwitie |
```

在同一文件夹下分组的相关文件。

如何在Project Navigator中使用Reveal

要在Project Navigator中使用Reveal,我们使用此快捷方式。

-shift+-Command+JorNavigate menu>Reveal in Project Navigator

这是一个在Project Navigator中展示 MasterFeedViewController 的示例。

```
<^>> ≠ ≣0 Œ
                                 import UIKit
                                 import Account
import Articles
CONTRIBUTING
                                 import RSCore
                                 import RSTree
                                 import SafariServices
                                 class MasterFeedViewController: UITableViewController,
                                     {\tt UndoableCommandRunner,\ MainControllerIdentifiable\ \{}
                              17 |
                                     @IBOutlet weak var filterButton: UIBarButtonItem!
                              19
O
                                     private var refreshProgressView: RefreshProgressView?
                                     @IBOutlet weak var addNewItemButton: UIBarButtonItem! {
                                             addNewItemButton.primaryAction = nil
                                     var mainControllerIdentifer = MainControllerIdentifier.masterFeed
                                     var undoableCommands = [UndoableCommand]()
                                     private let keyboardManager = KeyboardManager(type: .sidebar)
                                     override var keyCommands: [UIKeyCommand]? {
                                              don't want to sunnly any keyhoard
```

在Project Navigator中显示

Project Navigator中的过滤器文件

这不是一个捷径,但仍然是一个很好的学习技巧。**您可以在Project Navigator中过滤文件以仅显示您感兴趣的文件。**

```
| NethbourdWire | NethbourdWir
```

过滤包含"MasterFeed"的文件。

如何在Project Navigator中过滤文件

若要筛选Project Navigator中的文件,**请在Project Navigator底部的搜索栏中键入搜索词**。 然后,您可以处理感兴趣的文件的子集。

```
<△> ≠ 10 0
                                       guard let unreadCountNode = node, let indexPath =
CONTRIBUTING
                                          coordinator.indexPathFor(unreadCountNode) else { return }
                                          let cell = tableView.cellForRow(at: indexPath) as?
                                          MasterFeedTableViewCell {
                                           cell.unreadCount = unreadCountProvider.unreadCount
                                   @objc func faviconDidBecomeAvailable(_ note: Notification) {
                                       applyToAvailableCells(configureIcon)
                                   @objc func webFeedIconDidBecomeAvailable(_ note: Notification) {
                                       guard let webFeed = note.userInfo?[UserInfoKey.webFeed] as?
                                          WebFeed else {
                                       applyToCellsForRepresentedObject(webFeed, configureIcon(_:_:))
                                   @objc func webFeedSettingDidChange(_ note: Notification) {
                                       guard let webFeed = note.object as? WebFeed, let key =
                                          note.userInfo?[WebFeed.WebFeedSettingUserInfoKey] as? String
```

Project Navigator中的过滤器文件。

来回: \square + \square + \leftarrow 和 \square + \square + \rightarrow (Control+Command+ \leftarrow and Control+Command+ \rightarrow)

每次在编辑器中打开文件时,前一个文件都会在Xcode导航堆栈中被记住。**您可以使用** \boxtimes + Ξ + \leftarrow **和** \boxtimes + Ξ + \rightarrow **在这些历史记录之间来回切换**(Control+Command+ \leftarrow 和Control+Command + \rightarrow).

如何回到过去

要返回到以前打开的文件,我们使用此快捷方式。

-控制+-命令+←或导航菜单>返回

```
import Secrets
   import WidgetKit
18 var appDelegate: AppDelegate!
   class AppDelegate: UIResponder, UIApplicationDelegate,
      UNUserNotificationCenterDelegate, UnreadCountProvider {
       private var bgTaskDispatchQueue = DispatchQueue.init(label:
24
               var waitBackgroundUpdateTask =
           UIBackgroundTaskIdentifier.invalid
               var syncBackgroundUpdateTask =
           UIBackgroundTaskIdentifier.invalid
       var syncTimer: ArticleStatusSyncTimer?
       var shuttingDown = false {
                  shuttingDown {
                   syncTimer?.shuttingDown = shuttingDown
                   syncTimer?.invalidate()
```

回去吧。

如何前进

要继续您的导航堆栈,我们使用此快捷方式。

```
器 〈 〉 MetNewsWire〉 ■ iOS〉 🍑 AppDelegate〉 🖸 AppDeleg
                                                                                                                  import Secrets
import WidgetKit
X Laun
                                18 var appDelegate: AppDelegate!
                                21 class AppDelegate: UIResponder, UIApplicationDelegate,
UNUserNotificationCenterDelegate, UnreadCountProvider {
                                         private var bgTaskDispatchQueue = DispatchQueue.init(label:
                                              "BGTaskScheduler")
                                24
                                         private var waitBackgroundUpdateTask =
                                             {\tt UIBackgroundTaskIdentifier.invalid}
                                         private var syncBackgroundUpdateTask =
                                             UIBackgroundTaskIdentifier.invalid
                                         var syncTimer: ArticleStatusSyncTimer?
                                         var shuttingDown = false {
                                             didSet {
   if shuttingDown {
                                                       syncTimer?.shuttingDown = shuttingDown
                                                       syncTimer?.invalidate()
UIKit Exte
```

向前走。

跳转到对话框: +L (Command+L)

此命令将跳转到当前文件中的特定行。

当您想知道该行到底发生了什么时,这很有用。调试时,您可能需要这个。

如何使用跳转到行

要使用跳转到行,我们使用此快捷方式。

-命令+L或导航菜单>跳转到行

触发跳转到行后,您将看到一个**搜索字段弹出**。

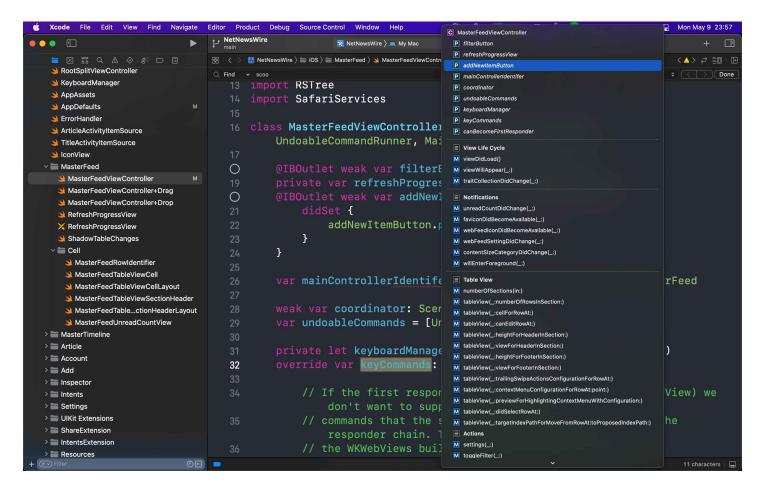
- 1. 键入要跳转到的行号。
- 2. 打回击。

```
| Netherwiffer | Neth
```

显示文档项: +6(Control+6)

随着项目的增长,您可能会在同一个文件中拥有一百行代码和更多相关实体,例如 变量 、 函数 、 扩展名 。

如果您不知道**显示文档项**命令,滚动所有这些都是一种痛苦。显示文档项将**显示一个下拉菜单,其中 包含当前文件中的所有符号**。



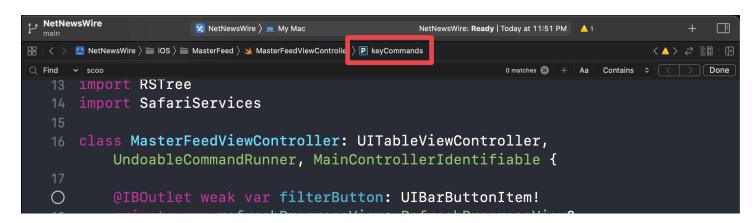
显示文档项。

如何使用显示文档项

要使用显示文档项,我们使用此快捷方式。

控制+6或查看菜单>编辑器>显示文档项

或者你可以**点击跳转栏的最后一部分**。



单击跳转栏的最后一部分以显示文档项。

使用显示文档项后,您将看到一个**下拉菜单,其中包含当前文件中的所有符号**。您可以通过这**三种方法**中的任何一种选择所需的符号。

1. 使用**向上/向下箭头**移动选定的符号,然后**按返回键进行选择**。

```
< △ > ₽ 10 Œ
   MetN
                                           Wire ) ≡ iOS ) ≡ Ma
                                             RSTree
                                     import SafariServices
 AppDefaults
                                     class MasterFeedViewController: UITableViewController,
                                          UndoableCommandRunner, MainControllerIdentifiable {
                                 0
                                          @IBOutlet weak var filterButton: UIBarButtonItem!
                                         private var refreshProgressView: RefreshProgressView?
                                          @IBOutlet weak var addNewItemButton: UIBarButtonItem! {
    didSet {
                                 0
                                                   addNewItemButton.primaryAction = nil
                                          var mainControllerIdentifer = MainControllerIdentifier.masterFeed
                                          weak var coordinator: SceneCoordinator!
                                          var undoableCommands = [UndoableCommand]()
   MasterFeedTable...ctionHeaderLay
> E MasterTir
                                         private let keyboardManager = KeyboardManager(type: .sidebar)
> 🚞 Article
                                         override var keyCommands: [UIKeyCommand]? {
> 🚞 Add
                                              don't want to supply any keyboard

// commands that the system is looking for by going up the responder chain. They will interfere with
) 🚞 Intents
> 🚞 Sett
> 🚞 ShareExtension
                                                   specifically the up and down arrow keys.
Resources
```

1. 滚动并单击。

```
⊠ ∏ Q
                                     🔼 Ne
                                      import UIKit
                                      import Account
 AppAssets
                                     import Articles
import RSCore
import RSTree
import SafariServices
                                      class MasterFeedViewController: UITableViewController,
                                           UndoableCommandRunner, MainControllerIdentifiable {
                                  0
                                           @IBOutlet weak var filterButton: UIBarButtonItem!
                                  19
O
                                           private var refreshProgressView: RefreshProgressView?
                                           @IBOutlet weak var addNewItemButton: UIBarButtonItem! {
    MasterFeedTableViewCell
                                                    addNewItemButton.primaryAction = nil
    MasterFeedTableViewSectionHead
                                  25
    MasterFeedUr
                                           var mainControllerIdentifer = MainControllerIdentifier.masterFeed
> masterTimeline
> = Account
                                           weak var coordinator: SceneCoordinator!
Add
                                           var undoableCommands = [UndoableCommand]()
> 🚞 Inspecto
lntents
                                           private let keyboardManager = KeyboardManager(type: .sidebar)
override var keyCommands: [UIKeyCommand]? {
> a UlKit Extensi
> IntentsExtension
                                                // If the first responder is the WKWebView (PreloadedWebView) we
```

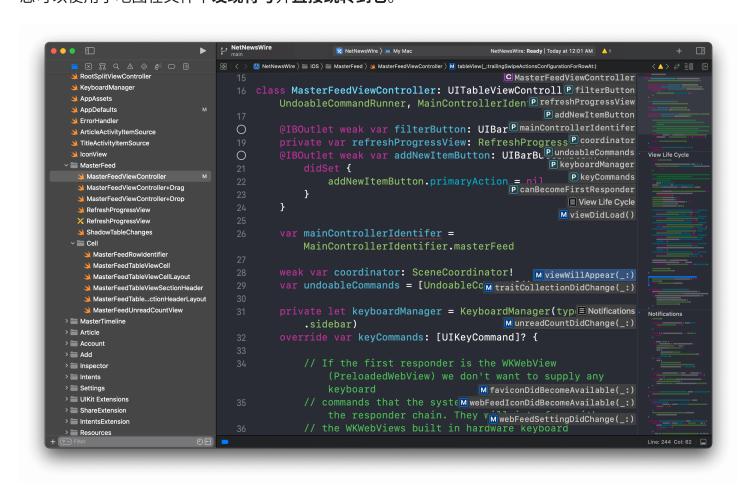
1. 您可以通过开始键入搜索词来过滤列表。

```
<△> ≠ ≣0 Œ
   import RSCore
   import RSTree
   import SafariServices
   class MasterFeedViewController: UITableViewController,
       UndoableCommandRunner, MainControllerIdentifiable {
0
       @IBOutlet weak var filterButton: UIBarButtonItem!
19
O
       private var refreshProgressView: RefreshProgressView?
       @IBOutlet weak var addNewItemButton: UIBarButtonItem! {
           didSet {
               addNewItemButton.primaryAction = nil
25
       var mainControllerIdentifer = MainControllerIdentifier.masterFeed
       weak var coordinator: SceneCoordinator!
       var undoableCommands = [UndoableCommand]()
       private let keyboardManager = KeyboardManager(type: .sidebar)
       override var keyCommands: [UIKeyCommand]? {
              don't want to supply any keyboard commands that the system is looking for by going up the
               responder chain. They will interfere with
```

//Mark 是一个工具,用于在代码编辑器和文档项弹出窗口中视觉上单独的文件。你可以阅读更多关于它在//MARK: -它是什么?

小地图

小地图最初是在Xcode 11中引入的。**小地图显示了您可以滚动浏览的代码的大纲概述**。 您可以使用小地图在文件中**发现符号**并**直接跳转到它**。



小地图。

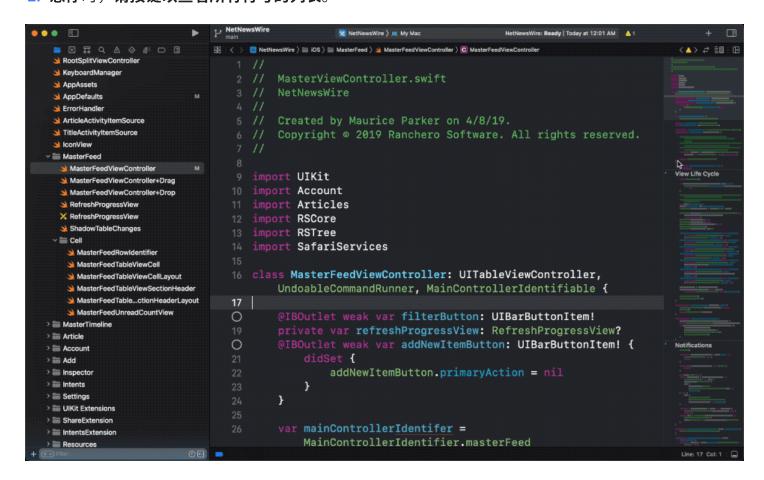
如何切换小地图开/关

要使用显示/隐藏小地图,我们使用此快捷方式。

-control+-shift+▽-命令+M或编辑器菜单>小地图

1. 当悬停在小地图上时,它将在鼠标光标位置显示符号的名称。

2. 悬停时,请按键以查看所有符号的列表。



在鼠标悬停的同时,按住以查看所有符号的列表。