

[StoreKit](#) / Offering media for sale in your app

Sample Code

Offering media for sale in your app

Allow users to purchase media in the App Store from within your app.

Download

iOS 12.0+ | iPadOS 12.0+ | Xcode 16.0+

Overview

This sample code project shows how to let users purchase media from your app using the [SKStoreProductViewController](#) class. Additionally, the sample demonstrates how to measure the effectiveness of a cross-promotional campaign for your apps or third-party apps using the [SKStoreProductParameterCampaignToken](#) and [SKStoreProductParameterProviderToken](#) keys. When cross-promoting your apps, set `SKStoreProductParameterProviderToken` to your own provider token. When promoting apps for other developers, set it to their provider token.

The app in this sample project displays a list of media items that the user can tap to make purchases from the App Store. You can also track advertising and promotions from this interactive list of media items for sale.

The sample app requires an iOS or iPadOS device.

Configure the sample code project

This sample code project defines a `Product` data type to represent media information it saves in the `Products.plist` file. `Products.plist` contains a single `Product` entry. Before you can run this sample on your device, you need to update this entry with your own data.

1. Set the sample's `productIdentifier` key to your media's iTunes identifier. For more information about iTunes identifiers, see [SKStoreProductParameterITunesItemIdentifier](#).

2. Set the `title` key to your media's name.
3. Set the `isApplication` key to YES if your media is an app, and NO, otherwise.
4. Set the `campaignToken` and `providerToken` keys to the appropriate tokens if you're cross-promoting apps, and leave them empty if you're not. For more information about generating campaign and provider tokens, see [Manage campaigns](#), [Campaign Token](#), and [Provider Token](#).

To add more data, duplicate and update the `Product` entry in `Products.plist` as necessary.

Display media from the App Store

This sample launches to the `TableViewController`, which shows a list of media associated with iTunes identifiers in `Products.plist`. When the user taps any item in the list, `TableViewController` presents a page where they can purchase the media from the App Store.

The sample creates a parameters dictionary and sets the `SKStoreProductParameterITunesItemIdentifier` key to the media's iTunes identifier.

```
var parametersDictionary = [SKStoreProductParameterITunesItemIdentifier: product.pr
```

The sample creates an `SKStoreProductViewController` object and sets the view controller class as its delegate. Then it passes the dictionary to the `loadProduct(withParameters:completionBlock:)` method of the `SKStoreProductViewController` object.

```
// Create a store product view controller.
let store = SKStoreProductViewController()
store.delegate = self
```

Then the sample presents the `SKStoreProductViewController` object modally.

```
/*
    Attempt to load the selected product from the App Store. Display the
    store product view controller if successful. Print an error message,
    otherwise.
*/
store.loadProduct(withParameters: parametersDictionary, completionBlock: {[unowned s
    if result {
        self.present(store, animated: true, completion: {
            print("The store view controller was presented.")
        })
    } else {
```

```

        if let error = error {
            print("Error: \(error.localizedDescription)")
        }
    })
})

```

The delegate needs to dismiss the store view controller when the purchase completes, or if the user cancels it. `TableViewController` implements the `productViewControllerDidFinish(:)` method to dismiss the presented store product view controller. Apps can also use this method to resume any activities that they pause before presenting the store.

Track advertising and promotions

The `SKStoreProductParameterCampaignToken` and `SKStoreProductParameterProviderToken` keys track advertising and promotion for the app. When the user taps an item in the list, `TableViewController` adds these keys and their values to a dictionary, which already contains `SKStoreProductParameterItemIdentifier` with the iTunes identifier.

`TableViewController` passes the dictionary to a created `SKStoreProductView` Controller object, and then displays the store product view controller modally.

```

/*
    Update `parametersDictionary` with the `campaignToken` and
    `providerToken` values if they exist and the specified `product`
    is an app.
*/
if product.isApplication, !product.campaignToken.isEmpty, !product.providerToken.isEmpty {
    parametersDictionary[SKStoreProductParameterCampaignToken] = product.campaignToken
    parametersDictionary[SKStoreProductParameterProviderToken] = product.providerToken
}

```

See Also

Recommendations

`class SKStoreProductViewController`

A view controller that provides a page where customers can purchase media from the App Store.

`class SKOverlay`

A class that displays an overlay you can use to recommend another app or an App Clip's corresponding full app.