**Deep Linking in iOS**

Deep linking consists of using a hyperlink that links to a specific piece of content within an app. The specific content could be a specific view, a particular section of a page, or a certain tab.

You can use deep linking for:

* Displaying a different landing page after launch.
* Inter-app communication by launching an app from another app and passing information.
* Creating a web-like URI based navigation scheme throughout your app.
* Enabling integration with other apps by letting them launch your app directly.
* Recording and analyzing user behavior to determine where your users launch your app from.

# Deep linking in iOS could be achieved by following ways

* Custom URL scheme
* Universal links (iSO 9 and above)

## Custom URL scheme: To enable deep linking, go to the Info tab in the Xcode project. In the URL Types section, click on the + button, and then add an identifier and a URL scheme. Ensure that the identifier and URL scheme you select are unique. Take note of the URL scheme you enter, as this is how iOS knows to open a link in your app. The sample app registers the following url scheme: *dlapp*

To confirm that your URL scheme has been registered, check **Info.plist** for an entry named ‘URL Types’. Expanding it will show you the new URL scheme you just registered. You can check that this is working by typing the following url into Safari Mobile: your-url-scheme:// ( For the sample app, it would be: dlapp://). This should open up your app. If not, please go through this section again before moving on.

When the link is opened, app will get the notification in the following delegate method

|  |  |
| --- | --- |
| 2 3 4 | -(BOOL)application:(UIApplication \*)application openURL:(NSURL \*)url sourceApplication:(NSString \*)sourceApplication annotation:(id)annotation |

## ****What’s New in iOS 9.2****

A change was made in Safari to make the ‘Open App’ modal a Javascript-non-blocking modal. This means that URI schemes cannot be used to open the app directly from Safari. They still work, in that they still open up the app, however if the app is not installed, the user is left with an ugly error message.

This is a significant change to the traditional way you would handle routing users who already have your app installed. Previously, you would open up the app directly from Safari when a user clicked on your link or visited your website. No more.

## ****Upgrading to Support Universal Links****

Fortunately, there is hope. Although it’s not perfect, Apple launched Universal Links in iOS 9.0, which moves the app routing into the OS so that developers don’t need to worry about doing the routing in Javascript

## ****Universal Links****

When you support universal links, iOS users can tap a link to your website and get seamlessly redirected to your installed app without going through Safari. If your app isn’t installed, tapping a link to your website opens your website in Safari.

Universal links give you several key benefits that you don’t get when you use custom URL schemes. Specifically, universal links are:

* **Unique.** Unlike custom URL schemes, universal links can’t be claimed by other apps, because they use standard HTTP or HTTPS links to your website.
* **Secure.** When users install your app, iOS checks a file that you’ve uploaded to your web server to make sure that your website allows your app to open URLs on its behalf. Only you can create and upload this file, so the association of your website with your app is secure.
* **Flexible.** Universal links work even when your app is not installed. When your app isn’t installed, tapping a link to your website opens the content in Safari, as users expect.
* **Simple.** One URL works for both your website and your app.
* **Private.** Other apps can communicate with your app without needing to know whether your app is installed.

NOTE

Universal links let users open your app when they tap links to your website within [WKWebView](https://developer.apple.com/documentation/webkit/wkwebview) and [UIWebView](https://developer.apple.com/documentation/uikit/uiwebview)views and Safari pages, in addition to links that result in a call to [openURL:](https://developer.apple.com/documentation/uikit/uiapplication/1622961-openurl), such as those that occur in Mail, Messages, and other apps.

When a user is browsing your website in Safari and they tap a universal link to a URL in the same domain as the current webpage, iOS respects the user’s most likely intent and opens the link in Safari. If the user taps a universal link to a URL in a different domain, iOS opens the link in your app.

For users who are running versions of iOS earlier than 9.0, tapping a universal link to your website opens the link in Safari.

Adding support for universal links is easy. There are three steps you need to take:

* Create an apple-app-site-association file that contains JSON data about the URLs that your app can handle.
* Upload the apple-app-site-association file to your HTTPS web server. You can place the file at the root of your server or in the .well-known subdirectory.
* Prepare your app to handle universal links.

You can test universal links on a device.