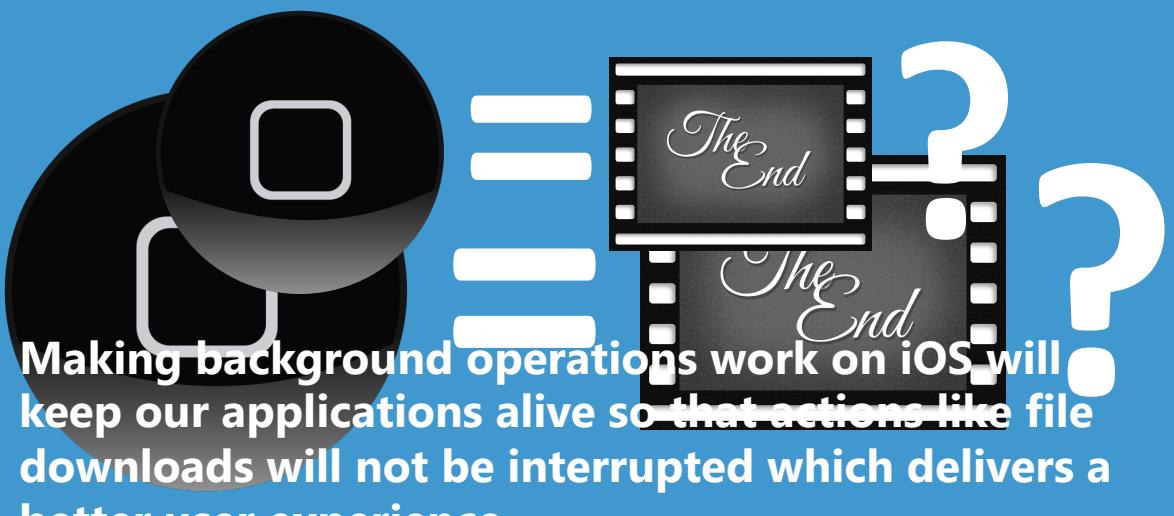
A portrait of a smiling man with short brown hair, wearing a light blue t-shirt. He is positioned on the left side of the slide, partially obscured by a diagonal blue line.

Xamarin Evolve 2014

The home button is not the end: Backgrounding and multitasking on iOS

René Ruppert
rene.ruppert@xamarin.com

 **Xamarin**
University

An illustration featuring a large black Apple Home button icon on the left. To its right is a film strip icon with the words "The End" repeated twice. Three white question marks are scattered around the film strip. The background is a solid blue.

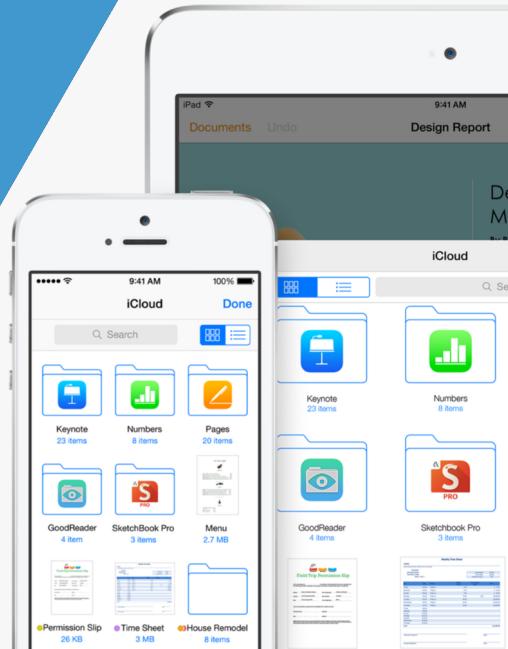
Making background operations work on iOS will keep our applications alive so that actions like file downloads will not be interrupted which delivers a better user experience.

Xamarin University

Agenda

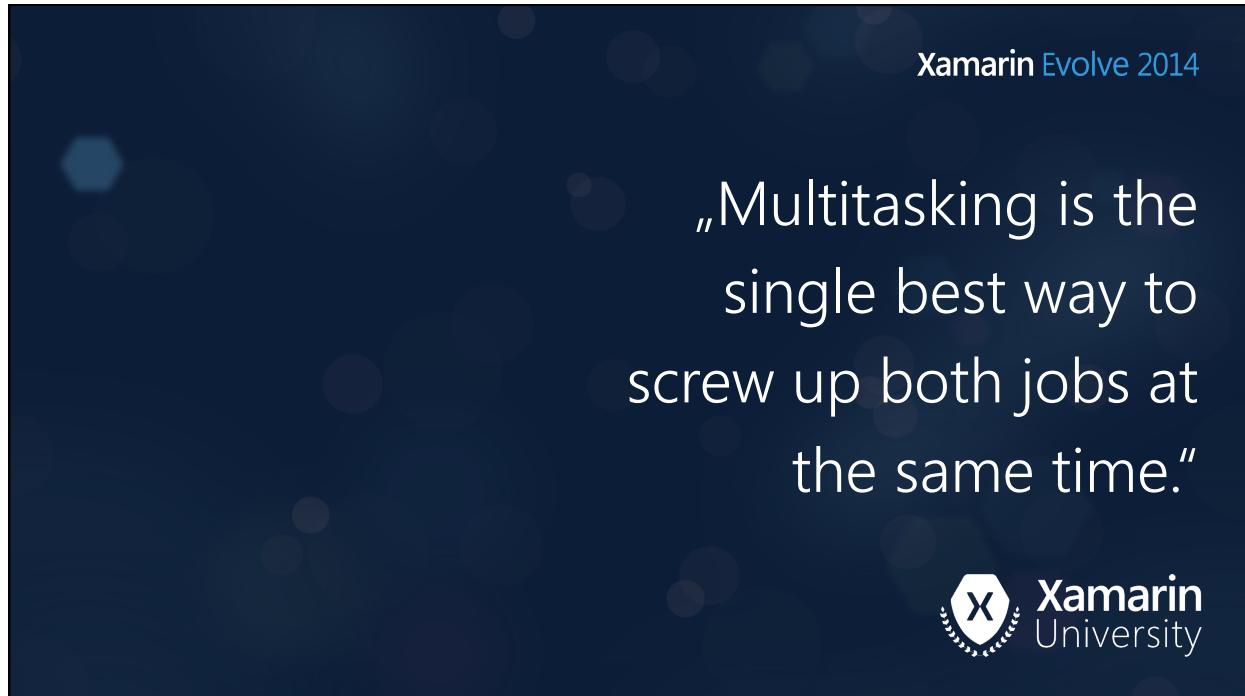
1. Why multitasking?
2. The Application Lifecycle
3. Long Running Tasks
4. Multi Tasking APIs

Xamarin University

The image shows a white iPhone on the left and a silver iPad on the right. The iPhone screen displays the 'iCloud' app with a list of documents including 'Keynote', 'Numbers', 'Pages', 'GoodReader', 'SketchBook Pro', and some files from 'Permission Slip'. The iPad screen shows a document editing interface with tabs for 'Documents' and 'Design Report', and a sidebar labeled 'iCloud'.

Why Multitasking?

Xamarin University



Your iPhone never sleeps

- Downloads / Uploads
- Notifications
- Music
- Navigation
- Bluetooth Communication

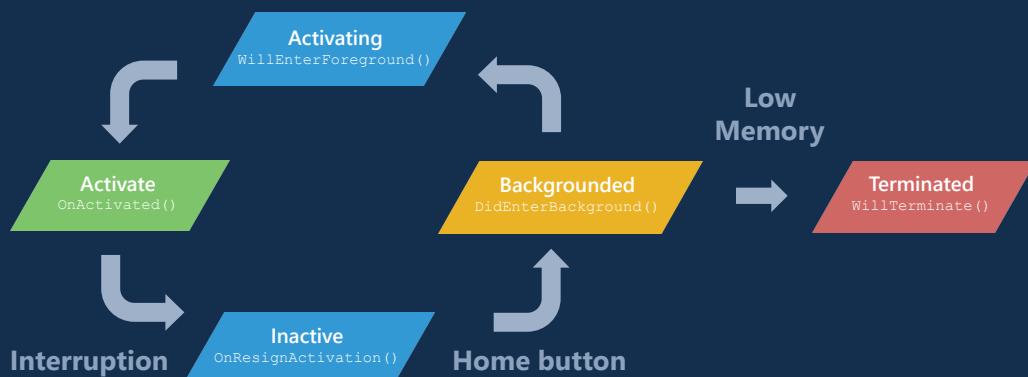
Xamarin University

Application Lifecycle

Xamarin University

Lifecycle states

AppDelegate provides callback methods



Xamarin University

Lifecycle states

Observing state changes

```
public override void ViewDidLoad()
{
    base.ViewDidLoad();

    UIApplication.Notifications.ObserveDidEnterBackground(
        (object sender,NSNotificationEventArgs e) => {
            // AppDelegate.DidEnterBackground() triggered
        } );
}
```

Xamarin University

iOS is very strict

- Very strict time limits
- Better keep your app responsive...
- ...or it will be terminated



Xamarin University

Long running Tasks

Xamarin University

Fire and don't forget

```
var app = UIApplication.SharedApplication;  
  
int id = app.BeginBackgroundTask();  
DoSomeBackgroundWork();  
app.EndBackgroundTask(id);
```

Xamarin University

Fire and don't forget

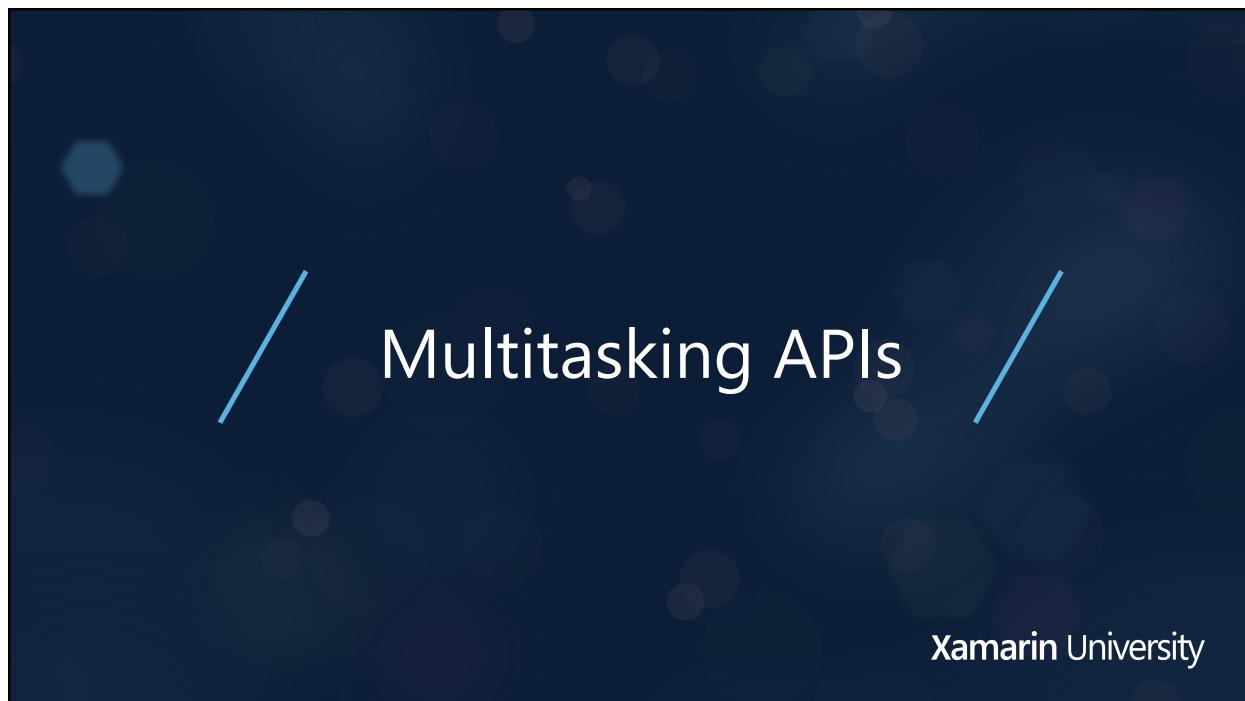
- Long running tasks can be kicked off from the lifecycle callbacks...
- ...but also everywhere else
- Can be nested
- Have to be ended
- Do not perform anything in the background on their own

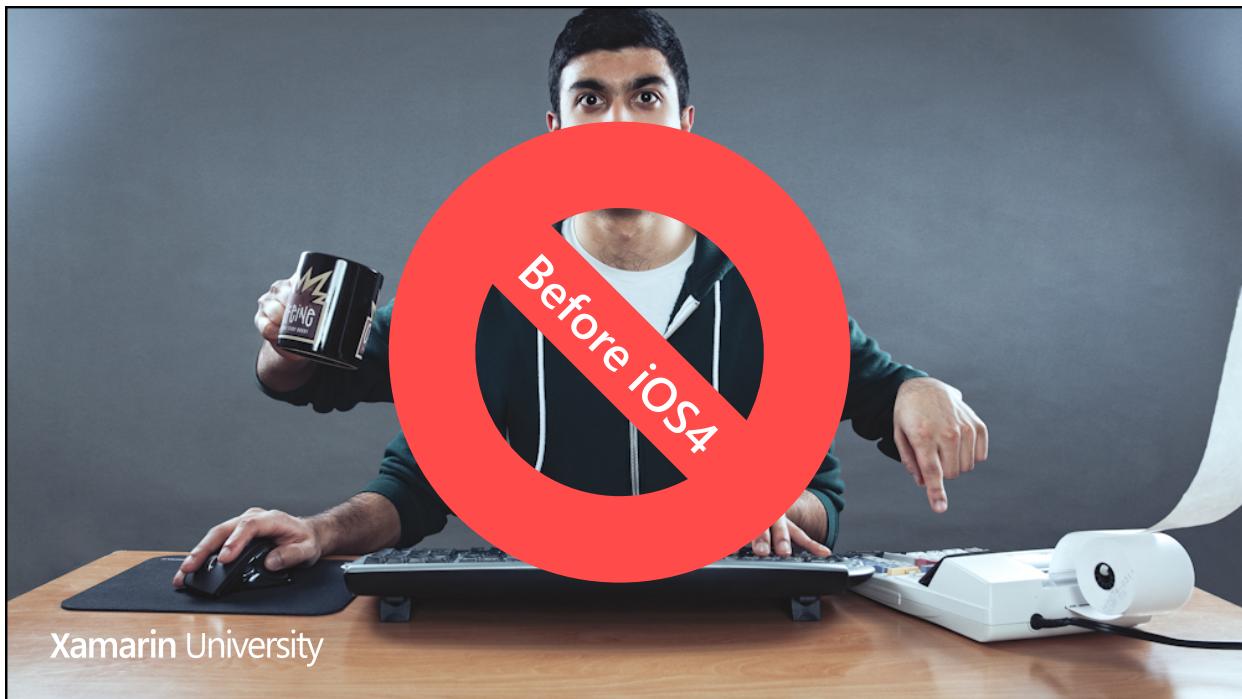
Xamarin University



Demo

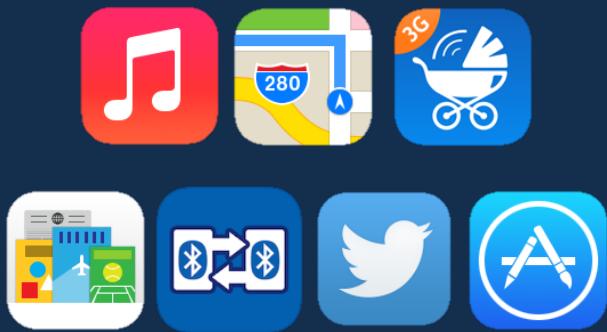
Xamarin University





Specific features only

- Audio & Video / AirPlay
- Location Updates
- Voice over IP
- Newsstand downloads
- External accessories
- Bluetooth LE accessories
- Act as Bluetooth LE accessory
- Local and Remote notifications
- Background updates



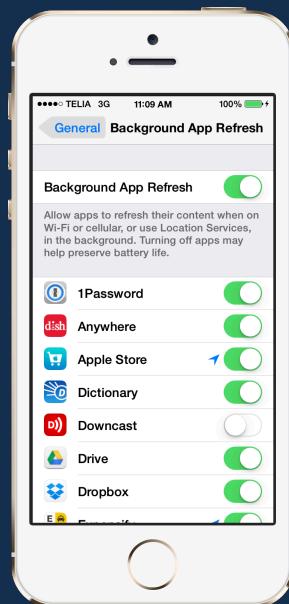
Xamarin Evolve 2014

Demo

Xamarin University

iOS7

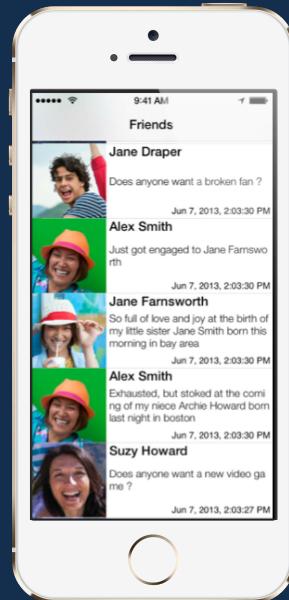
- Background Updates
- Background Fetch
- Remote Notifications
- Background Transfer Service



Xamarin University

Specific iOS7 APIs

- Background Fetch
- Use for apps that require frequent, uncritical updates
- Have data ready when user comes back to your app...
- ...instead of showing „please wait”
- iOS observes app usage
- Will update „just in time”



Xamarin University

Specific iOS7 APIs

- Remote Notifications
- App is launched before notification is shown
- Supports silent notifications
- Use for updates with higher importance

Xamarin University

Specific iOS7 APIs

- Background Transfer Service
- Transfers managed by iOS
- No time limits
- Continue when app exits

Xamarin University



What's new in iOS8

- APIs changed
- Location usage notifications
- Recent contacts
- Rumor: split screen support



Summary

- Background operations are necessary
- Watch out for device capabilities and limitations
- Use appropriate APIs: background task vs. background necessary app

Xamarin University