IoT | Kitura on iOS | offline mode | Open Source | Web Framework

Kitura/iOS: Running a Web Serv

Vadim Eisenberg

Published on March 13, 2017 / *Updated on December 8, 2017*

Server-side Swift applications, and Kitura applications in particular, can run o
macOS and Ubuntu Linux. There are use cases, however, that require runnin
server-side application on iOS means embedding its code in a plain client-si
these use cases and conclude with a Hello-world Kitura/iOS app.

Until now, developers had to produce two different server-side implementat
applications on iOS. To enable running the server-side part on a "server OS",
web frameworks for Java, Node.js, PHP, and Ruby, among others. However, i
device. Consequently, the developers wrote the iOS server-side part using so
server-side Swift web framework provides the opportunity to prevent this du
single server-side application that could run both remotely on a "server OS"
this is "the killer argument" in favor of server-side Swift web frameworks.

I read about some of the use cases presented below while googling about iO
proposed by my colleagues. Some of the use cases just came to mind. I gues
server-side applications. I would be happy to hear your ideas.

Use Cases for Running Server-Side Apps o

I found six use cases for running server-side apps on iOS. Let me explain the

Use Case 1: Running server-side code in Xcode to development

This use case is the most obvious one. Once your server-side code can run o side code in Xcode together with your client-side unit tests. Furthermore, it's **side and the server side** of your application in the same Xcode instance. You application and then step back and forth between the client and the server p **server** for iOS and use the mockup server before the real server is developed Xcode playground.

Use Case 2: Offline mode

According to the title of this Forrester report, supporting offline mode is the mobile apps. Running a web server on iOS can enable **offline mode** in a simp of your iOS app.

Note that several databases can run on iOS, for example, SQLite or IBM Clou embedded database, you can have a full-fledged web server inside your iOS embedded web server when it is disconnected from the network. The client- server through a *localhost* URL. The networking code of the iOS app could re server will be replaced. Alternatively, you can use URLSessionDelegate.

The app can synchronize a local iOS database replica with the remote databa connected to the Internet. Alternatively, you can synchronize the local data e Furthermore, some databases facilitate synchronization between the local a

Naturally, using the network stack for offline mode could have performance
mode implementations. However, the simplicity of the implementation could
backend.

Use Case 3: On-device demo mode

The local iOS backend can be used for **offline demos**. Suppose you want to
the client's premises. You are not sure that your device will have network co
pitch anyone?). Furthermore, your app may encounter firewall issues. When
if the wireless network will be reliable and fast enough for your app to functi
can take your server on your device with you, always ready for demonstratio

Use Case 4: Peer-to-peer

An additional use case involves the **peer-to-peer** communication of iOS devi
of TCP, e.g., HTTP, WebSocket, or MQTT. The caveat here is that currently yo
the background mode. Once your screen is locked, or the user switches to so
will have to ensure that your peer-to-peer apps run in the foreground. You ca
your iOS device. See the next use case for an explanation about *Guided Acce*

Use Case 5: Ad-hoc server

Another use case is an **ad-hoc client-server topology**. Suppose you implem
has to function at some site with a local Wi-Fi network. The catch is that the
or has a prohibitively slow connection. Imagine you are at some military loca
Internet connectivity is restricted.

Alternatively, imagine you have a cellular device that provides a personal hot
the hotspot device has slow cellular connectivity. The tethered devices will h

remote server. However, connectivity **between** the tethered devices could be
happen on the Wi-Fi network of the personal hotspot.

Using server side code on iOS, you can set up one of your iOS devices to act
devices to use it as their backend.

You may want to set your server-side iOS app to run in Guided Access mode.
protected by a passcode. Disabling hardware buttons is one of the options o
your iOS app not to report its idle time by setting *UIApplication.shared.isIdle*
ensure that neither you nor the OS will accidentally deactivate your app and
with these settings neither you nor the OS can lock the screen or switch off y
continue to run forever. Well, OK, not forever, but either until the battery is d
Access mode explicitly, by providing the passcode.

Use Case 6: Gateway for IoT

Yet another use case for a server on iOS is **Internet-of-Things**. You could hav
IoT devices. To facilitate IoT data collection and processing, the gateway dev
sensors locally. Once in a while the iOS device will send the data to the remo
perform some processing of data before sending, for example to filter or to a

Bonus Use Case: Just for the fun of it

I hope you are convinced now regarding the use cases. The last use case is j
iPhone as a web server?

Kitura iOS Hello World app

Now it's time for some action. Let's run a simple Kitura application on iOS. M

Zats and I implemented the Kitura/iOS Hello World app. The app demonstra
devices on a local network. The server returns a “Hello World” message with

The Kitura server-side application is embedded in a client-side app that facil
iOS server experience, the app displays the URL of the server and the QR coc
start and stop the server and shows the server logs.

To build the app, clone the project and run *make*. *make* generates an Xcode
the server side parts as projects. Once finished, *make* will open the workspa
actually performs in a future blog post.

Build and run the client side part on an iOS device or in the simulator. To test
displayed URL in a browser of any device on the same Wi-Fi network. The “H
displayed in the browser. See log messages running on the main screen of th
the log in a separate screen.

Note that you can run the server-side code of the project as a plain server-si
the *ServerSide* folder of the repository. Just build it with Swift Package Mana
“killer” advantage of server-side Swift web frameworks – the same code car

Conclusion

In this blog post, I describe several use cases for running Kitura applications
technical issues involved in developing Kitura/iOS applications.

Update on December 8, 2017
Additional blog post about Kitura/iOS: Kitura/iOS Part 2: Software Engineerin

TAGS IOS, IPAD, IPHONE, KITURA, SERVER, SERVER SIDE SWIFT, SERVER-SIDE

---

by Vadim Eisenberg

Twitter

I work as a senior software developer in IBM Research - Haifa. In my 16 years in IBM Research, variety of projects in different areas, from binary executable optimization to microservice platfo roles - as a developer, an architect and a project lead, and programmed in C, C++, Java, Swift, A Ruby, JavaScript, Go and Bash. My interests include Software Engineering and Architecture, Pro Software Design/Design Patterns, Application Environments/Application Frameworks/Middlewa technologies, Web and Mobile applications, Server-Side Swift, Microservices and Service Meshe Technion - Israel Institute of Technology.

---

5 comments on"Kitura/iOS: Running a Web Serve

Richard Applebaum • March 13, 2017

Or... maybe, run Xcode on the iPad Pro — and do it all on a single device!

Reply

David James • March 22, 2017

Database synchronization for offline mode is very hard to get right, for reasons this, but it still requires a significant amount of paperwork to get set up and wo approach to offline mode, the selection being dependent on available time, mo connection, 2. Offline – read only from local database, 3. Limited Offline – reac Offline – full read and write (CRUD) with synchronization and error handling.

Reply

Arno.Nyhm • March 29, 2017

additinal use case: also to make a onboarding local database before an user sv to register just to see how an app works

Reply

Daniel • November 09, 2018

Does this still work? I've tried following your steps on github but when I try run
*** [setDeploymentVersionOfSharedServerClient] Error 1
ruby Builder/Scripts/set_deployment_version.rb SharedServerClient/SharedSe
ruby: No such file or directory — Builder/Scripts/set_deployment_version.rb (L

Reply

Vadim Eisenberg • November 18, 2018

Hi Daniel, sorry it took me some time to get to the issue and to fix th
build badge on the github page of https://github.com/IBM-Swift/Kit
Swift/Kitura-Sample-iOS. You can click on the badge and see the bu

Reply

Join The Discussion

Your email address will not be published. Required fields are marked *

Enter your comments...

Name *

Email *

Website

Recent Posts

- Announcing Kitura 2.8 and more
- Introducing SwiftKafka: Accessing Event Streams in Swift
- JWT authentication using KituraKit
- Using swift-log with Kitura
- Announcing Kitura 2.7 and more
- Swift-JWT 3.3: Adding ES256, ES384, and ES512 Support
- BlueECC: Encrypt, Decrypt, Sign and Verify with Elliptic Curve