

Leigh Williamson, John Ponzo,  
Patrick Bohrer, Ricardo Olivieri,  
Karl Weinmeister, Samuel Kallner



**IBM**  
Press.

# Swift<sup>TM</sup> in the Cloud



**WILEY**



# Swift<sup>TM</sup> in the Cloud



# Swift<sup>TM</sup> in the Cloud

Leigh Williamson

John Ponzio

Patrick Bohrer

Ricardo Olivieri

Karl Weinmeister

Samuel Kallner

WILEY

## Swift™ in the Cloud

Published by  
John Wiley & Sons, Inc.  
10475 Crosspoint Boulevard  
Indianapolis, IN 46256  
[www.wiley.com](http://www.wiley.com)

Copyright © 2017 by IBM Corporation. All rights reserved.

The following terms are trademarks or registered trademarks of IBM Corporation in the United States, other countries, or both: IBM, the IBM Press logo, DB2, AIX, WebSphere, Rational, IBM MobileFirst, Bluemix, z/OS, z Systems, IBM LinuxONE, CICS, and IMS. Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates. Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both. Other company, product, or service names may be trademarks or service marks of others. A current list of IBM trademarks is available on the web at “copyright and trademark information” as [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

Published by John Wiley & Sons, Inc., Indianapolis, Indiana

Published simultaneously in Canada

ISBN: 978-1-119-31937-5  
ISBN: 978-1-119-36853-3 (ebk)  
ISBN: 978-1-119-36847-2 (ebk)

Manufactured in the United States of America  
10 9 8 7 6 5 4 3 2 1

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permissions>.

**Limit of Liability/Disclaimer of Warranty:** The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Web site is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or website may provide or recommendations it may make. Further, readers should be aware that Internet websites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services please contact our Customer Care Department within the United States at (877) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at <http://booksupport.wiley.com>. For more information about Wiley products, visit [www.wiley.com](http://www.wiley.com).

**Library of Congress Control Number:** 2017946220

**Trademarks:** Wiley and the Wiley logo are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. Swift is a trademark of Apple, Inc. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in this book.

*Karl would like to dedicate this book to his supportive wife, Samantha, and their crazy kiddos, who still gave him enough time to write.*

*Leigh dedicates this book to his wife, Cheryl, always his compass through life, and his daughter, Claire, the light of his life.*





# About the Authors

**Leigh Williamson** is an IBM Distinguished Engineer who has been working in the company's Austin, Texas lab since 1989, contributing to major software projects for IBM, including OS/2, DB2, AIX, Java, WebSphere, Rational, and MobileFirst products. He is currently leading a cross-disciplinary team of IBM Cloud Consultants who assist clients with cloud computing strategy and execution. Leigh is an active mentor and leader in the IBM technical community, working with international mentees; working in the Advanced Technical Eminence program; leading multiple patent brainstorming teams; publishing technical blogs and articles; conducting external broadcast webinars; and speaking at IBM and non-IBM conferences.

This is the fourth IBM Press book on which Leigh has collaborated over the past 15 years, with works covering Java standards, WebSphere Application Server, and enterprise-class mobile application development. He holds a B.S. in Computer Science from Nova University and an M.S. in Computer Engineering from The University of Texas at Austin.

You can follow Leigh on Twitter at @leighawillia. His LinkedIn address is <https://www.linkedin.com/in/leigh-williamson-9048654>.

**John Ponzo** is an IBM Fellow and Chief Technology Officer for Mobile who has shaped the future of IBM business in mobile computing and delivered innovative products and services to web browser and server-based standards. He is a pioneer in technology that promotes end-user interaction through mobility, web programming, web application middleware, and software tools. John led the development of key software technologies including HTML/JavaScript runtime libraries, XML middleware, web services runtime libraries, Eclipse Web Integration, and Web 2.0 enterprise collaboration services. He also authored the “Enterprise Mobile” and the “Mobile First” theses that are the cornerstone of mobile strategy and execution at IBM. John also serves as the IBM technology ambassador to Kenya.

John is the primary technical collaborator between Apple and IBM in the effort to define and enhance the Swift programming language for both mobile client development and cloud services development.

**Patrick Bohrer** is a Distinguished Engineer in the IBM Cloud division. His responsibilities include serving as the technical lead for the company's global efforts around Swift@IBM ([developer.ibm.com/swift](http://developer.ibm.com/swift)). He formerly served as the technical lead of the Mobile Innovation Lab in Austin, Texas. Patrick also helped lead IBM Research's Mobile Research agenda after co-leading the 2012 Global Technology Outlook topic entitled "Mobile First," which helped set the technical direction for current mobile and cloud efforts at IBM. Patrick received a B.A in Computer Science from The University of Texas at Austin.

**Karl Weinmeister** is the Program Director for Swift@IBM Engineering, based in Austin, Texas. He is passionate about improving people's lives and experiences with technology. In his role, he has helped to enable Swift to extend from its mobile roots to become a full-stack language ecosystem.

Previously, Karl led engineering for the IBM Mobile Innovation Lab. He is a diehard Duke basketball fan and enjoys spending time with his family.

**Ricardo Olivieri** is a Senior Software Engineer at Swift@IBM Engineering. His areas of expertise include gathering and analyzing business requirements, architecting, designing, and developing software applications. Ricardo has extensive experience in the complete software development cycle and related processes, especially in Agile methodologies.

Ricardo has several years of experience in Java development as well as in Groovy, Perl, and Python development, with a strong background in back-end (server-side, business logic, SQL, and NoSQL databases) and front-end development. Several years ago, Ricardo added Business Process Manager (BPM) design and development to his skill set, which allowed him to assume the role of BPM consultant and developer using IBM Business Process Manager. While working at the Mobile Innovation Lab, Ricardo gained valuable skills and knowledge in the iOS and Android ecosystems.

Ricardo is now mainly focused on the adoption of the Swift language on the server and the IBM cloud, Bluemix. He has a B.S. in Computer Engineering from the University of Puerto Rico, Mayagüez Campus.

**Samuel (Shmuel) Kallner** is a Senior Technical Staff Member in the Smart Client Platforms group at the IBM Research Lab in Haifa, Israel. He is currently the Technical Lead of the Kitura project. Shmuel has over thirty years of experience at IBM working on a wide variety of projects including mobile apps, web-based end-user application development environments, mobile app developer tools, both sides of client-server-based applications, and more.

# About the Technical Editor

**Matthew Perrins** is a Senior Technical Staff Member working on IBM Cloud Developer Services in Austin, Texas. Matt is one of the architects for the company's production-ready public Cloud Developer Experience on Bluemix—the IBM cloud platform. He is focused on making it very easy to develop and deploy mobile, web, and digital channel applications with the IBM cloud, and to integrate them with world-class leading runtimes for Swift, Node.js, and Java. Matt has been leading the IBM cloud teams in the evolution to true DevOps continuous delivery with cloud-native architectures, and driving that integration into the IBM Bluemix user experience.

Matt has spent a significant part of his career building systems-of-record and systems-of-engagement solutions using IBM technologies with IBM clients. He understands developers, user experiences, transactions, and cognitive solutions and how to deliver them at scale on the IBM cloud.



# Credits

**Executive Editor**

Jim Minatel

**Project Editor**

Adaobi Obi Tulton

**Technical Editor**

Matthew Perrins

**Production Editor**

Dassi Zeidel

**Copy Editor**

Marylouise Wiack

**Production Manager**

Katie Wisor

**Manager of Content Development and  
Assembly**

Mary Beth Wakefield

**Marketing Manager**

Christie Hilbrich

**Professional Technology & Strategy Director**

Barry Pruett

**Business Manager**

Amy Knies

**Project Coordinator, Cover**

Brent Savage

**Proofreader**

Nancy Carrasco

**Indexer**

Johnna VanHoose Dinse

**Cover Designer**

Wiley

**Cover Image**

© fishbones/Getty Images



# Acknowledgments

**M**any thanks to everyone at Wiley Publishing for their outstanding work on this project: to Jim Minatel for encouraging us to take this book concept forward and for yet again supporting the realization of a book that engages in deeper learning; to Adaobi Obi Tulton, the project editor, for driving this project to completion in the most efficient way possible—it was a real pleasure to work with such an accomplished and adept editor; to Marylouise Wiack, the copy editor, for translating this book into readable prose; and to Dassi Zeidel, the production editor, for bringing everything together to create a final, polished product.

Sincere thanks go to Matthew Perrins, the technical editor, for the incredible amount of work and personal time he selflessly put into ensuring that the content in this book can be utilized seamlessly by readers. Also, thanks to Steven Stansel, who was instrumental in formulating the original concept and proposal for the book. Brian White Eagle provided invaluable assistance with peering over the horizon in Chapter 9. And Shereen Ghobrial contributed the bulk of the mainframe-related content.

The biggest thank-you must, of course, go to our own families. This book was written over six months, predominantly at night and over weekends and holidays. In addition to sharing us with extremely demanding full-time jobs, our families made further sacrifices to enable us to spend time on this project.





# Contents at a Glance

Introduction .....	xxiii
<b>1</b> Swift.org, the Open Source Project .....	1
<b>2</b> A Swift Sandbox in the Cloud .....	19
<b>3</b> A Basic Introduction to Swift .....	35
<b>4</b> The IBM Bluemix Buildpack for Swift .....	53
<b>5</b> Using Containers on Bluemix to Run Swift Code .....	91
<b>6</b> Swift Package Management .....	119
<b>7</b> Swift and Kitura for Web Applications .....	131
<b>8</b> Serverless Programming with Swift .....	175
<b>9</b> Over the Horizon: Where Do We Go from Here? .....	203
Index .....	221



# Contents

Introduction .....	xxiii
<b>1 Swift.org, the Open Source Project .....</b>	<b>1</b>
What's Included .....	1
Source Code Repositories. ....	2
How to Get Involved .....	5
Mailing Lists. ....	7
Bug Tracking .....	8
Swift Evolution and Roadmap. ....	12
Priorities for the Swift 4.0 Major Release .....	14
Binary Downloads .....	14
MacOS Binaries .....	15
Linux Binaries. ....	16
Swiftenv, Swift Version Manager .....	17
Summary .....	17
<b>2 A Swift Sandbox in the Cloud .....</b>	<b>19</b>
The IBM Cloud Platform. ....	19
Getting Started .....	26
Sign Me Up!. ....	26
Saving and Sharing Code Samples. ....	28
Selecting Swift Versions and More. ....	30
Have You Run on a Mainframe Lately? .....	30
IBM Swift Package Catalog and Sandbox .....	32
Summary .....	33

<b>3</b>	<b>A Basic Introduction to Swift</b>	<b>35</b>
	Background	35
	Let's Get Coding!	35
	Swift Standard Library	35
	Swift Foundation Library	37
	C Library Interoperability	39
	Concurrency Library	41
	Memory Management	43
	The Language Landscape	48
	Language Groupings	48
	Language Timeline	50
	Summary	51
<b>4</b>	<b>The IBM Bluemix Buildpack for Swift</b>	<b>53</b>
	Cloud Foundry Buildpacks	53
	Buildpack Phases	54
	Working with the IBM Bluemix Buildpack for Swift	55
	Where Is the Source Code Hosted?	55
	What Version of the Buildpack Is Currently Installed?	56
	File Artifacts Required for Provisioning Your Application on Bluemix	58
	Installing Additional System-Level Dependencies	61
	Downloading Closed Source Dependencies	68
	Examples of Using the IBM Bluemix Buildpack for Swift	69
	Swift HelloWorld	69
	Kitura Starter	74
	BluePic	77
	Using the Latest Code of the IBM Bluemix Buildpack for Swift	87
	Summary	88

<b>5</b>	<b>Using Containers on Bluemix to Run Swift Code</b>	<b>91</b>
	What Are Docker Containers?	91
	Docker Images for Swift	92
	Installing Docker.	93
	Using Docker as a Development Tool	94
	Exposing Your Swift Application's Port to the Host System.	96
	Using docker-compose.	96
	Why Use Containers on Bluemix?	98
	Containers for Packaging and Deployment of Swift Applications	99
	The Kubernetes Platform	99
	Running Your Docker Image in the Bluemix Cloud	100
	Install the Kubernetes Command Line	100
	Install the Bluemix Command Line.	100
	Install the IBM Container Registry Plug-In	102
	Install the IBM Container Service Plug-In	102
	Create a Runtime Image for Swift Applications.	103
	Tag a Docker Image.	106
	Push a Docker Image to Bluemix	107
	Create a Kubernetes Cluster on Bluemix	108
	High Availability in Kubernetes Clusters	112
	Binding Bluemix Services to IBM Containers.	113
	Summary	116
<b>6</b>	<b>Swift Package Management</b>	<b>119</b>
	Swift Package Manager	119
	Using Swift Package Manager	120
	Commands	121
	Package.Swift Details.	123

Swift Package Catalog . . . . .	123
Browsing . . . . .	123
Searching. . . . .	124
Package Details . . . . .	126
Dependency Visualization . . . . .	127
Trying Out a Package in the Sandbox . . . . .	128
Summary . . . . .	130
<b>7 Swift and Kitura for Web Applications . . . . .</b>	<b>131</b>
Kitura . . . . .	133
Sending Simple Responses to Requests . . . . .	136
A Real-World Library Example . . . . .	137
Accessing Information Sent in Requests . . . . .	138
Starting the Library Application . . . . .	140
Working with Various HTTP Features Using Kitura . . . . .	149
Other Ways of Serving Content Using Kitura . . . . .	155
Other Useful Kitura Middleware. . . . .	157
Authentication Using the Kitura-Credentials Framework . . . . .	159
The Library Sample with Authentication . . . . .	160
Kitura and Data Access . . . . .	163
Swift-Kuery . . . . .	163
Kitura-redis . . . . .	170
Summary . . . . .	173
<b>8 Serverless Programming with Swift . . . . .</b>	<b>175</b>
Microservices and Serverless Computing . . . . .	175
Serverless Computing Concepts . . . . .	177
OpenWhisk . . . . .	179
Swift and OpenWhisk. . . . .	182
Using the Web-Based OpenWhisk Tools. . . . .	183
Command Line OpenWhisk . . . . .	189
A More Involved Example . . . . .	195
Summary . . . . .	201

<b>9</b>	<b>Over the Horizon: Where Do We Go from Here? . . . . .</b>	<b>203</b>
	Bringing Swift to the Server . . . . .	203
	IBM Cloud Tools for Swift . . . . .	204
	Server-Side Frameworks . . . . .	210
	Expanding the Range of Swift . . . . .	215
	Swift Support for Linux . . . . .	215
	The Internet of Swift Things. . . . .	215
	Big Iron Swift. . . . .	216
	Swift DevOps . . . . .	218
	Summary . . . . .	219
	Index . . . . .	221

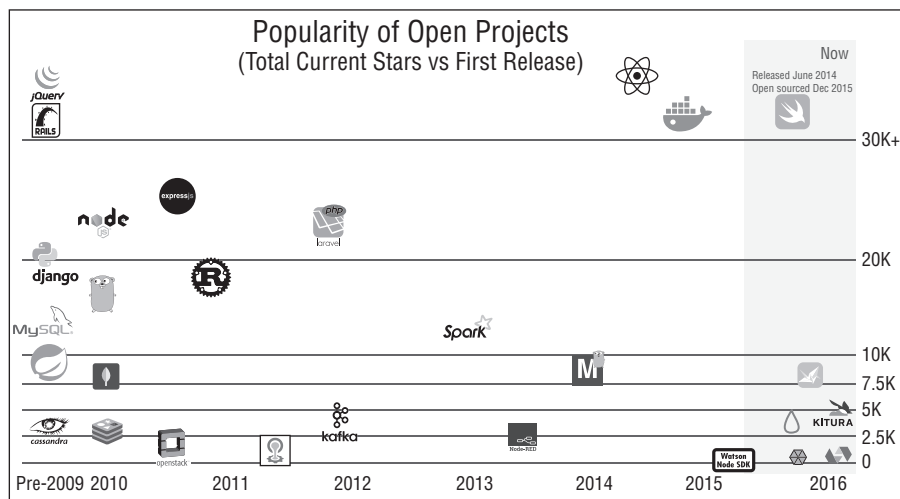




# Introduction

Since Apple introduced the Swift programming language in 2014, it has become one of the most rapidly adopted computer programming languages in history. Programmers love the modern syntax used by Swift and the way it's fun to develop code, similar to how they felt about Java a generation ago. Programming skills and experience in Swift are in high demand in the industry, with the promise of high salaries for those who invest the time to learn and practice the language.

Apple originally introduced Swift as an alternative language to Objective-C for developing iPhone, iPad, and macOS applications. The company has now expanded Swift into the realm of solutions for the Internet of Things, with support for tvOS (Apple TV) and watchOS (Apple Watch wearable devices). As illustrated in Figure 1, Swift is now one of the most popular open source projects and Swift frameworks such as Kitura are gaining ground quickly.



**Figure 1:** Popularity of the Swift programming language

At about the same time that Swift was first introduced, Apple and IBM formed a strategic partnership to produce innovative, industry-specific mobile applications for the Apple iOS ecosystem. IBM proceeded to embrace the Swift programming language and implemented over 100 mobile apps as part of this partnership. IBM engineers saw the value of Swift firsthand in these applications. The open source release of the Swift language in late 2014 began another chapter in the partnership, as IBM chose to invest in and support the cause of bringing the Swift language to the cloud as a result of the focus on server-side Swift environments.

Development of these applications highlights the critical, expanding role of server-side logic in powering these new experiences that users now take for granted. From syncing data across devices; to connecting people with friends and co-workers; to monitoring news and alerting us about new events based on our interests and activities; to providing cognitive insights into our applications; server-side logic is critical to creating truly brilliant apps. Now the ability to develop, debug, and deploy this logic in the same language used to create mobile experiences is a game changer for the development community.

This book, written by members of the development team at IBM who helped bring Swift to the cloud, covers everything you need to know about how to develop Swift programs that run in cloud environments. It combines technical information with the concepts that originally led to the development of the technology. This book provides plenty of examples of Swift language code, as well as a living website where a community consisting of the authors and other passionate Swift experts continues to discuss Swift and its future directions.

## IBM, Apple, and Swift

---

On July 15, 2014, IBM and Apple announced what was at the time a very surprising partnership, with the goal of transforming business applications by building mobile enterprise and industry-specific solutions for the Apple platform. This partnership was unexpected by most industry watchers and radically altered the enterprise mobile computing landscape.

The IBM offering that resulted from the Apple partnership is called MobileFirst for iOS. It focuses on enterprise and industry transformation by providing users with the latest features of the Apple platform and user design, coupled with the back-end data center integration required to reinvent the next generation of enterprise applications.

Key features of the IBM offering include large-enterprise–class robustness and scalability; back-end enterprise data center integration; big data and analytics integration; and a highly polished mobile front-end user interface experience.

The user interface experience was codesigned by Apple and IBM to significantly upgrade the standards for usability, elegance, and user satisfaction beyond typical enterprise software. Since 2012, IBM has been making massive investments in IBM Design Thinking philosophy and techniques, building up several large design studios in an effort to apply good design to business software. This software design emphasis by IBM was one of the natural collaboration areas of the partnership, with the strong Apple design culture being applied to all of their own products.

As IBM MobileFirst for iOS was being developed, a choice was made to use the latest iOS platform APIs for iPhone and iPad business applications. The scope was later extended to include Apple Watch. IBM leveraged many of the extended development kits provided by Apple, such as HomeKit, CloudKit, and the connected car capabilities in various new and innovative business solutions. As of December 2015, IBM had created over 100 industry-specific mobile apps for the IBM MobileFirst for iOS collection.

While Apple has pioneered the transformation of the consumer mobile app experience, IBM and Apple consider business mobile app transformation to be an underserved market and a really great design opportunity that they can uniquely address together.

## Introduction of Swift

Apple introduced the Swift language at the Apple World Wide Developers Conference (WWDC) in 2014, and IBM decided to begin using the language to build the first wave of IBM MobileFirst for iOS mobile applications. IBM assembled a team of developers with expertise in building mobile solutions. Their previous programming language skills included Java, JavaScript, and Objective-C. This team of IBM programmers quickly learned Swift and began using it to implement the mobile apps in the IBM MobileFirst for iOS collection.

Working with Apple, we at IBM learned a lot about what it takes to build amazing mobile business applications. IBM also discovered the value of Swift.

Swift was designed by Apple to be a safe, interactive, and high-performance systems language. It also blends the ease of scripting language syntax with the performance of a systems language. The IBM team found that in comparison to mobile apps developed in Java for the Android mobile platform and Objective-C for iOS, Swift apps required less code.

The IBM team appreciated everything about Swift, from its type safety—which empowered them to be agile and evolve the applications quickly while knowing that the compiler would catch any errors—to its performance and memory advantages, which are critical for application responsiveness. The concise syntax of the language also led to great developer productivity.

The IBM teams also learned to understand what is necessary to develop application-specific web services to power these business mobile applications. It significantly enhanced the overall productivity of the team to not have to switch languages away from what was used for the mobile front end (Swift) to work on the back-end services.

The developers found the Swift code easier to read, share, and evolve. What the Swift language did for legibility of the application code represented a large increase in productivity for the development team. Other languages used for mobile apps were generally more verbose. It also significantly increased code quality with Swift-based type checking. The importance of a strongly typed language in the productivity of the development team can hardly be overstated. Most of the more than 100 applications developed by IBM for various business solutions could be produced by a handful of programmers working in small teams.

Figure 2 shows a comparison of Swift with other programming languages and illustrates how Swift enables inherent application performance and developer productivity benefits through its attributes such as:

- Modern programming language constructs
- Error detection at compile time, not runtime
- Code reengineering