Curriculum Vitae

Regan Russell BSc

January 2, 2026

Phone: +61 41 428 7577
Email: regan @ pymblesoftware.com
regan.russell @ gmail.com
https://www.linkedin.com/in/pymblesoftware/

GitHub:
https://github.com/pymblesoftware

Latest CV

Previous Apps
https://humm-group.com
https://play.google.com/store/apps/details?id=com.sunsuper.prod
https://play.google.com/store/apps/details?id=au.com.heritage.app .
https://play.google.com/store/apps/details?id=au.gov.nsw.onegov.parknpay.release
https://play.google.com/store/apps/details?id=com.Rydotechnologies.Rydo&hl=en
https://play.google.com/store/apps/details?id=com.foxtel.epg&hl=en_AU&gl=US
https://play.google.com/store/apps/details?id=com.dopel.dopel

Personal Apps:

https://apps.apple.com/sa/developer/pymble-software-pty-ltd/id553990081

Latest CV: https://www.pymblesoftware.com/cv.pdf

Previous Experience

Contract roles through PymbleSoftware Pty Ltd since 2008 18 year(s), 0 month(s) .
(Personal contracts since 1998) 28 year(s), 0 month(s) .

Recent
Academic projects.
Business projects.
Personal software projects.
Real estate projects.
Working with publishers on technical manuscripts as part of the editorial cycle.

Cross-platform mobile developer with Flutter / PHP Laravel (Lumens) developer (Contract)
Charitable Organisation
Flutter (3.10.6), Flutter Realm (1.3.0), Dart, Stripe (flutter_stripe:9.3.0), credit_card_scanner (1.0.5), PHP, Laravel. Lumens.

GDPR (privacy by design) and PCI DSS (card data handling) compliant as possible. Only static content is kept in an in-memory database, not even in storage.
Over 12,000 lines of Flutter code. Over 800 lines of PHP web service code.
I read somewhere that the average US programmer averages 200 lines of code per day. In the first month, I averaged over 500 lines of code per day. **Greenfield** project from start to finish. Donation app for water and food aid projects. **Cross platform** with **flutter**. Did all the web services with Laravel/Lumens and MySQL database.

Payment gateway integration with **Stripe**.

---

iOS and Android Developer (Native)
Company **Humm Group**
https://humm-group.com
https://apps.apple.com/au/app/humm/id1455391873
https://apps.apple.com/au/developer/humm-pro-pty-ltd/id1532343413
https://apps.apple.com/ca/app/humm-ca/id1586818651

Apps: Bundll, QANTAS Pay, Humm Pro, Humm Canada, Humm Cards and legacy apps which included the following tools and libraries:

AFNetworking[1], FLAnimatedImage[2], Auth0[3], NVActivityIndicatorView[4], Floating Panel, JWTDecode[5], Kingfisher[6], Firebase, Dynatrace, AppFlyerFramework, CardIO[7], EMSMobileSDK, Alamofire[8], Lottie[9], KeychainSwift[10], Branch, Swipe, Validator, SwiftEntryKit[11], SVProgressHUD[12], SwiftLint, Quick, Nimble, IOSSecuritySuite, Snapkit, SwipeCellKit[13], ObjectMapper, SwiftyJSON[14], DateToolsSwift[15], RangeSeekSlider, NVActivityView, IQKeyboardManager, **SalesForce** market-

[1] https://cocoapods.org/pods/AFNetworking
[2] https://cocoapods.org/pods/FLAnimatedimage
[3] https://auth0.com/
[4] https://cocoapods.org/pods/NVActivityIndicatorView
[5] https://cocoapods.org/pods/JWTDecode
[6] https://cocoapods.org/pods/Kingfisher
[7] https://cocoapods.org/pods/CardIO
[8] https://cocoapods.org/pods/Alamofire
[9] https://cocoapods.org/pods/lottie-ios
[10] https://cocoapods.org/pods/KeychainSwift
[11] https://cocoapods.org/pods/SwiftEntryKit
[12] https://cocoapods.org/pods/SVProgressHUD-0.8.1
[13] https://cocoapods.org/pods/SwipeCellKit
[14] https://cocoapods.org/pods/SwiftyJSON
[15] https://cocoapods.org/pods/DateToolsSwift

Latest CV: https://www.pymblesoftware.com/cv.pdf

ing cloud[16], kotlinkit, cardview, androidx, Gson[17], threetenabp[18], brentrielly navigation, picasso[19], retrofit2[20], OkHttp3[21], biometric, rootbeer[22], fillrembedded, viewpagerdots[23], filetree, play services, swipe reveal layout, EML payments[24], RxAndroid/RxJava[25], epoxy, TMX, securebank, swiperefreshlayout[26], liveness

Humm Group white labels credit cards and provides financial services such as small to medium-enterprise equipment leasing. Specifically, Humm provides consumer credit cards / Store cards such as the Farmers[27] chain in New Zealand and the QANTAS Pay card.

Since I have had experience with financial services and credit card processing online in one form or another since working with Plink libraries at BDE in 1997; I was able to quickly adapt to the workflows, codebases/business practices of the credit card services provided by Humm Group to provide bug fixes and minor maintenance on 192342 lines of Swift, 879 lines of Objective C, 334037 lines of Kotlin, 47,982 lines of java, excluding libraries. To develop the changes to these codebases I reviewed Figma diagrams and incorporated changes to UI elements in Android and iOS code as per the requirements of the JIRA tickets.

To modernise the build environment, I added Azure DevOps / Fastlane CI/CD build pipelines for a few of the apps.

(Native) iOS and Android Developer
Company **Milieu Labs**. (Contract)
https://apps.apple.com/au/app/milieu-climate/id1566271872

Consumer thermostat systems.

Pre-existing tech stack: Amazon **IoT** [28], AWS, SwiftFromat, RxSwift[29], RxCocoa[30], ReSwift[31], AWS SDK iOS, Alamofire, Crashlytics, Freddy, Butterknife[32], MQTT[33], RxJava[25], Redux[34], Mockito, Jetpack. AWS Cognito, AWS IoT, Dagger2, OkHttp[21], retrofit2[20]

Milieu Labs provides controls for domestic air conditioning systems. The control unit of the air conditioner is replaced with a module that receives MQTT messages from a server which is indirectly managed through a user's account on the app. The project added a Zone information screen. Zone state change handling which entailed unpacking MQTT Zone state messages, and translating that through a middleware layer onto the screen through the UI controls.

Android Developer
Company **Oracle** (Contract)

Worked on JSON parsing code for an Oracle internal project for external clients.
The project was under strict NDA. There was an intense level of security around the project. The project had international security implications.

[16] https://developer.salesforce.com/docs/marketing/marketing-cloud/guide/mobile-push-sdk.html
[17] https://github.com/google/gson
[18] https://github.com/JakeWharton/ThreeTenABP
[19] https://github.com/square/picasso
[20] https://github.com/square/retrofit
[21] https://square.github.io/okhttp/
[22] https://github.com/scottyab/rootbeer
[23] https://github.com/afollestad/viewpagerdots
[24] https://www.emlpayments.com/
[25] https://github.com/ReactiveX/RxAndroid
[26] https://developer.android.com/jetpack/androidx/releases/swiperefreshlayout
[27] https://www.farmers.co.nz/
[28] https://aws.amazon.com/iot/
[29] https://github.com/ReactiveX/RxSwift
[30] https://cocoapods.org/pods/RxCocoa
[31] https://cocoapods.org/pods/ReSwift
[32] https://github.com/JakeWharton/butterknife
[33] https://mqtt.org
[34] https://cocoapods.org/pods/Redux

Latest CV: https://www.pymblesoftware.com/cv.pdf

Technologies employed included OkHttp[21], Retrofit[20], bumptech/glide, Appsflyer, Facebook SDK, Firebase, Kakao, Line SDK, SnapChat SDK, Twitter SDK, VK[35] SDK.

Android Developer
Company **Australian Retirement Trust/Sunsuper** (Contract)

https://play.google.com/store/apps/details?id=com.sunsuper.prod

Java, Kotlin (Android)

Minor maintenance work.
Took Figma diagrams and updated UI elements in Android code.

Developer
Company **BP (British Petroleum)** (Contract)
No link: The company's internal app is only available on the corporate portal.

Flutter, Kotlin (Android), AWS CloudWatch, S3, node.js Lambdas, DynamoDB,
Azure DevOps

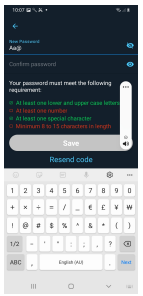
This was a suite of 3 apps that are four internal use by point of sale BP employees.

I introduced the **feature** of local notifications in Kotlin/Android with an Android AlarmManager boot time background service in less than 500 lines of MVVM architected code.
This was so that station employees could be reminded to rotate stock at certain points in the day.
Bug fixing in Kotlin, Flutter and node.js Lambdas.
Meetings with the UK are scheduled at 8 pm. Late-night meetings are scheduled for every night of the week. Dealing with Teams messages at 12:46 am

---

Android developer
Company **CoverMore** (Contract)

Kotlin, LiveData, Navigation graph, MVVM.
I **rewrote** the password reset screen. The design was such that the business logic was in a view model and as the user entered values the view model validated the input against the password rules. Publishing/observing, as the input matched the rules the label of each rule turned from red to green.
I introduced the **feature** of data binding into the app. Worked on the common button and some of the reusable widgets.
Android Material design, reusable software components and UI widgets.

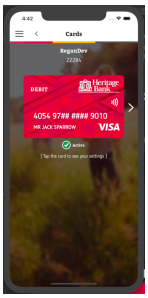


CoverMoore Password Screen

Company **Nightlife music** (Contract)
https://apps.apple.com/us/app/crowddj/id911666442?ls=1
Java, Objective-C maintenance work and small projects including work on 8-year-old code base in Objective-C on iOS and Java on Android.

[35] VK is the Russian version of Facebook

Latest CV: https://www.pymblesoftware.com/cv.pdf



Heritage Bank

Company **Heritage Bank** (Contract)
https://play.google.com/store/apps/details?id=au.com.heritage.app .
https://apps.apple.com/au/app/heritage-mobile-banking/id386772598?ls=1 .

Originally 100 per cent Objective-C, I **rewrote** scheduled payments screen in Swift adding new sections and **refactored** the design into separate legacy and MVVM modules.
Merged all the complex build steps and frameworks into folders. Reduced the build time from 8 minutes to under one minute.

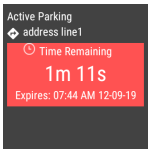
Accessibility audit for vision impaired. Added **features** for Apple's VoiceOver support.
Provisioning credit cards into Apple Wallet via Apple APIs, encrypting on bank servers with the encryption keys from Apple's API. Leading and mentoring junior developers.
Cocoapod
Fixing the build system.
PassKit

Company **Xinja Bank** (Contract)
https://xinja.com.au/
Android Developer.
Technologies: Kotlin

Maintenance and minor **bug fixing** of Kotlin NeoBank banking app. Koin and View Model, Live Data, Android material design.
There were over 20 onboarding screens in creating an account. I was tasked with adding more.

Technologies employed included androidx, biometric, firebase, Lottie, GSon[17], jodaTime, newRelic, ok2Curl, retrofit[20], zendesk

---



Watch App

Client **NSW Government DFSI** (Contract)
iOS and Android developer
Technologies: Swift / Kotlin
https://play.google.com/store/apps/details?id=au.gov.nsw.onegov.parknpay.release
https://apps.apple.com/au/app/parknpay/id1453474761?mt=8

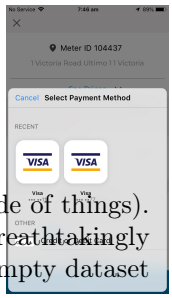
Added feature of Braintree payments integration in Swift and Kotlin in an app for NSW Government including Apple Pay and Google Pay.
The previous payment **integration** had reportedly been problematic and the payment gateway vendor for Braintree was almost effortless.
It took less effort and resources than expected so I added the **features** of Apple Watch and Android Gear Watch integration into the app for the NSW government.
Did a custom notification where the countdown timer and buttons appear in the phone notifications even on the lock screen.

The app communicates with an internal server, which acts as a proxy for another server that manages parking meters.
Part of the credit card processing integration occurred on the parking meter servers and part was within the app.

The idea of the app is to pay for parking meters from the phone and be notified of the need to top up the parking meter if it is about to expire by tapping on the watch and the phone notifications.
Complex integration due to the number of servers involved and the number of moving parts and the level of security.
All the watch app extensions, all the app to payment gateway integration.

Latest CV: https://www.pymblesoftware.com/cv.pdf

Client **Rydo Taxi app** . (Contract)
https://apps.apple.com/au/app/rydo/id1150318596
https://play.google.com/store/apps/details?id=com.Rydotechnologies.Rydo&hl=en
iOS and Android developer.
Technologies: Swift / Kotlin



Payments

Maintenance work and small projects involving C#, Swift, Java, and Kotlin (Mostly Java on the Android side of things). Fixed about a dozen bugs each week on Android then about a dozen on iOS the next week. The code was breathtakingly bad, if the server found no results in the database it would return 404 instead of 200 and an indication of an empty dataset in the JSON payload.
The server would 500 or 400 frequently on requests that worked a minute ago.

Somehow previous developers had considered the foreground and background of the app to be handled in view controllers instead of the app delegate, there was a lot of surreal code.
Whoever wrote it had no idea of the application life cycle or standard industry practices like not abusing HTTP error codes.

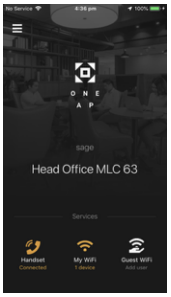
I was doing work on iOS, Android and ASP.NET and completing **features** in two weeks they had waited 2 years to be completed.

Client **ServCorp** (Contracts)
https://www.servcorp.com.au/en/oneap/
Technologies: Swift / Kotlin / Objective-C / Java / PHP / Larvel / CodeIgniter / LaTeX

Comments: Replaced back-end login process with Auth0[3] (an OAuth implementation). **Maintenance** work in PHP and iOS and Android code to replace naive login and session verification.
Did a **localisation** for the Japanese (日本語) version. I have some understanding of Japanese having once worked in (Citrix 2005/2006) and visited the country several times.
Wrote extensive documentation, and fixed bugs. **Refactored** local login to an OAuth login with Auth0[3].

Client: **eDale** / **ServCorp** (Contracts)
Technologies: Swift / Kotlin / Objective-C / Java / Node.js / Heroku / PHP / Larvel / CodeIgniter / LaTeX/ Android

Comments: Several overlapping short-term contracts. Some clean-up of previous work for Edale, bug fixing, and random ad hoc changes. Hand over and manage a developer I hired through my own company to take over work.
Short-term work on Bluetooth tile beacons for finding car keys - type tag beacons. Another USB-based Bluetooth beacon (Sensoro) and QR code project for allocating desktop handset phones and Wifi access. Worked 3 days a week at Servcorp doing maintenance work on an iOS and Android app that allows for provisioning of desktop handsets and (guest) wifi access



Servcorp OneAp

Sensoro beacons, translating Chinese comments into English. **Implemented feature of** Android circular reveal animation like the iOS animation. completed button handlers, differentiated between my wifi and guest wifi buttons and called different web service endpoints accordingly. Scale and slide up login animation to match the iOS version. Made UI consistent with the iOS version. Added disconnect handset screen in Kotlin. Added **feature** of pull to refresh the menu on the Android version to be consistent with the iOS version. Added site features refresh endpoint call and UI update. Added background handset polling. Used to be location dependent. Fixed refresh, device count wrong and other bugs in. both iOS and Android versions. A branch was created in the source code control system called allow_handset_choice to allow the choice of a beacon/handset other than the nearest found beacon. An extra button was added to the found handset screen. Touching this button launches another screen with a list of handsets detected. On both the iOS and Android versions all the beacons that were available in the area are stored in collections in the bowels of the Bluetooth code. In both versions, the collections were propagated up through the software layers until they are presented in a UITableView or a recycler view and the on-item selection handlers then put the selected MAC address in place of the nearest found beacon MAC address and then call the occupy handset code which calls the web service endpoint (Function 16).

Half of the endpoints of the ServCorp server were in CodeIgniter and half were in Larvel in another repository. I set up a local copy of the code bases on my development machine for debugging server-side and client-side issues.

Latest CV: https://www.pymblesoftware.com/cv.pdf

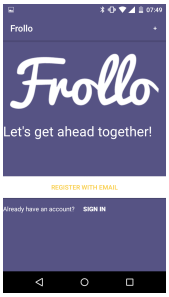
Client: **Edale Holdings Pty Ltd** (Master contract)
The app has been removed from the app store.
Technologies: Android/iOS/Heroku/Amazon S3, Kotlin, retrofit2[20], Swift4, node.js, mongoDB.
Front-end and back-end developer. Team leader
Comments: **Greenfield development** Created a node.js web service with about a dozen endpoints. Created an iOS app that allows users to register, log in and like other users of a dating app. User images on S3 displayed on a dating app. Created an Android version of the same app to talk to the same web service. Full control over everything changing web services to fit the needs of mobile apps and changing mobile apps to deal with web service restrictions. Next to no formal specification. Lots of client hand-holding, random inconsistent changes, About 2700 lines of Kotlin, about 1000 lines of node.js and about 2200 lines of **Swift4** in the first week.
I eventually hired another developer through my own company. **Complete app start to finish**

Client: **Object consulting** (Contract)

Bluetooth cattle scanner

Technologies: Kotlin, Android, Bluetooth, Retrofit2[20]
Android developer
Comments: **Greenfield development**. Created an app that read from an RFID tag "wand". The RFID tags would be assigned as a collar or an animal tag. The list of tags was uploaded as JSON to the RESTful web service. Used retrofit2 for accessing the web service. Used Realm to store the tags. Used some example Bluetooth code to read the wand and send it simple commands.
Complete app start to finish

Client: **Frollo** (Contract)



Frollo

https://apps.apple.com/app/id1179563005

Technologies: Kotlin,
Android developer
Comments: **Greenfield development**. Created an app that integrates financial data from all bank accounts from all banks and hits about 60 endpoints on a single web service. Created multi-dimensional build variants so there was product A, production, staging, and develop and the same for product B. Wrote about 21,000 lines of Kotlin and about 800 lines of Java in 2 months. Initially not very idiomatic Kotlin but progressively more stylistic Kotlin. Wrote JUnit tests. Used com.github.PhilJay:MPAndroidChart to present data pulled from
com. square-up.retrofit2:retrofit:2.3.0 into recycler views. Used
com.github.vicpinm:krealmextensions:1.1.5 to store data from the endpoints. Used Picasso to fetch and display bank icons. Used bouncy castle to encrypt data put or posted to endpoints. Used Android Account manager to store account information. Created one-time passwords to send credentials and financial data to the server. Used SSL-pinning. Created code that read data from an end-point and dynamically created controls from it almost like a web browser. Used an Airbnb library on GitHub to do deep linking to handle frollo:// and https://m.frollo.us
Complete app start to finish The entire app had already between done on the other platform and all decisions had been made so it was a matter of making like for like.

Client: **Seven Studio at the Easter Show (2 weeks), AMP Banking App** (Multiple Contracts)
Technologies: Android, iOS (iPad), AlamoFire, ffmpeg, Swift 3, AWS, S3., Carthage, mongoose.js, node.js, MongoDB, Kotlin, **iOS/Node.js, MongoDB, Android developer.**

Latest CV: https://www.pymblesoftware.com/cv.pdf

Comments:
AMP Android banking app - Kotlin, RxJava, Dagger dependency injection. A mess of deeply nested Kotlin templates. “Clean design” – model, view, presenter, use case, contracts.

7Studio, an App that records videos, displays a teleprompter. The recorded video gets an overlay graphic applied and then merged with a news break intro and news break closing theme. RecordedUI XC unit test. Memory leak debugging with instruments. Run at the Easter show for kids to be a reporter for a day.

Client: **Accenture/Foxtel** (Contract)
https://play.google.com/store/apps/details?id=com.foxtel.epg&hl=en_AU&gl=US
Technologies: Android, Web Services
Android developer
Comments: **Maintenance work** on 100,000 plus line of code program guide / billing-account management app with hundreds of thousands of users. Read crash reports, and patched code with defensive programming techniques like checking if things are null before attempting to use them. Fairly basic maintenance work. The code at times was textbook on what not to do.

Client: **RaceNet** (Contract)
No link: The app was replaced after several years.
Technologies: Android and iOS. Java and Swift.
iOS, Android, Web Services developer.
Comments: **Greenfield development.** For the Android version I wrote about 20,000 lines of code in the first month. About 5,000 lines a week. GSon, Android studio. Got the basic version of the Android app completed. Worked with Web services developer on defining data models and endpoints. Used about 30 web service endpoints. I used Cocoapods and SwiftyJSON. Wrote the iOS version in Swift. I ended up churning out about 30,000 lines of Android code in Java and about 20,000 lines of code in Swift.
Complete app start to finish

Client: **SmartBill** (Contract)
No Link: Removed - App Store policies
Technologies: Android, Sqlite3, iOS 7, CoreData, Xcode 5s Unit testing framework.
iOS/Android developer.
Comments: **Greenfield development**. Developed an Android and iOS app to gather data usage and call log data and send it to a server for ‘smart bill’ analysis against phone plans. Used silent push notifications to wake the app up for processing. AFNetworking[1], working on ASP.NET code for Apple push notification servers. SQL Lite.
Complete app start to finish

Latest CV: https://www.pymblesoftware.com/cv.pdf

Education

Mobile

Certificate Kotlin for Android: Best Practices **LinkedIn Learning**
Certificate Advanced Kotlin Database Development **LinkedIn Learning**
Certificate Android Dependency Injection with Dagger 2 and Kotlin **LinkedIn Learning**
Certificate Android App Development: Enterprise Integration **LinkedIn Learning**
Certificate Android Development: Understanding Intents **LinkedIn Learning**
Certificate Building an Android App with Jetpack Libraries **LinkedIn Learning**
Certificate Android App Security: A Structured Approach to Pen Testing **LinkedIn Learning**
Course Jetpack Compose crash course **Udemy**
Certificate Android Development: Data Binding **LinkedIn Learning**
Certificate Android App Development: Accessibility **LinkedIn Learning**
Certificate Android Development: Retrofit with Kotlin **LinkedIn Learning**
Certificate Building Android Apps with AWS **LinkedIn Learning**
Certificate Android App Development: Design Patterns for Mobile Architecture **LinkedIn Learning**
Certificate Java: Testing with JUnit **LinkedIn Learning**
Certificate iOS Development Tips **LinkedIn Learning**
Certificate Coding Exercises: Swift **LinkedIn Learning**
Certificate iOS App Development: Accessibility **LinkedIn Learning**
Certificate iOS App Development: Test-Driven Development **LinkedIn Learning**
Certificate iOS Development: Security **LinkedIn Learning**
Certificate Building iOS Apps with CloudKit **LinkedIn Learning**
Certificate SwiftUI Essential Training **LinkedIn Learning**
Certificate Make SwiftUI Playgrounds Applications **LinkedIn Learning**
Certificate Apple watchOS App Development: Advanced APIs[36] **LinkedIn Learning**
Certificate Biometric Authentication for iOS in Swift **LinkedIn Learning**[37]
Certificate Biometrics: Security and Privacy Considerations **LinkedIn Learning**
Certificate iOS App Development: Design Patterns for Mobile Architecture **LinkedIn Learning**
Certificate iOS 17 Development Essential Training **LinkedIn Learning**
Certificate Swift 5 Essential Training **LinkedIn Learning**
Certificate iOS App Development: Core ML **LinkedIn Learning**
Certificate iOS and watchOS App Development: Notifications **LinkedIn Learning**
Certificate Swift: Delegations and Data Sources **LinkedIn Learning**
Certificate Composable SwiftUI Architecture Using Redux: 1 Introduction **LinkedIn Learning**
Certificate Composable SwiftUI Architecture Using Redux: 2 Building the App **LinkedIn Learning**
Certificate Building an App for All Apple Platforms **LinkedIn Learning**
Certificate Xamarin.Forms Essential Training **LinkedIn Learning**
Certificate Xamarin Essential Training: Create Your First App **LinkedIn Learning**
Certificate Moving from Xamarin.Forms to .NET MAUI **LinkedIn Learning**
Certificate Dart Clean Code: Writing High Efficiency, Maintainable Dart Programs **LinkedIn Learning**
Certificate React Native Essential Training **LinkedIn Learning**
Certificate Introduction to SQLite **LinkedIn Learning**
Certificate Learning React Native **LinkedIn Learning**
Course Create a tiny app with ReactNative **Udemy**
Course Reactive Programming in iOS with RxSwift **Udemy**

Publications

Russell R., (2024) "Thirty-five years in Software Development", Kindle from https://www.amazon.com.au/dp/B0DK4FHMZ4 .

Russell R., (2024) "Thirty-five years in Software Development", Paperback from https://www.amazon.com/Thirty-five-years-Software-Development-Leadership/dp/B0DP7CR7SV .
ISBN 979-8301444326

Latest CV: https://www.pymblesoftware.com/cv.pdf

35 Years
book

Russell R., (2024) “Thirty-five years in Software Development”, Hardcover from https://www.amazon.com/Thirty-five-years-Software-Development-Leadership/dp/B0DPDLWPRC/ .
ISBN 979-8301885365

bada
book

Russell R., (2012) “Programming bada”, Kindle, iBooks and PDF file from www.pymblesoftware.com/book .
www.pymblesoftware.com/book/bada-short.pdf .
http://itunes.apple.com/au/book/programming-bada/id543013439?mt=11&ls=1
https://www.amazon.com/Programming-bada-Regan-Russell-ebook/dp/B007LFX608

Russell R., (Nov 1999) "BeRays: A ray tracer for BeOS", Doctor Dobbs Journal.
Russell, R., & Ryan C (1994) “METAcoder for windows: real-time and multi-pass.
event logging and analysis in the social and behavioural
sciences.” Psychology Teaching Review.
Russell, R., & Ryan C. (1994) “METAcoder for windows: real-time and multi-pass.
event logging and analysis in the social and behavioural.
sciences.” Psychology Software News.

Doc Dobbs
Journal

Review of Windows NT Device Driver Development Doctor Dobbs electronic review of computer books (ERCB).
Review of The Windows NT Device Driver Book: A Guide for Programmers ERCB.
Review of Developing Windows NT Device Drivers ERCB.
Review of Writing a UNIX Device Driver, Second Edition. ERCB.
Review of Panic: Unix crash dump analysis, ERCB.
Review of Advanced animation and rendering techniques, ERCB
Review of Windows TCP/IP. ERCB
Review of Open source development with CVS. ERCB
Review of System performance tuning. ERCB
Review of Learning the vi editor. ERCB
Review of Ada for experienced programmers. ERCB.

BeRays
Article

Publications Officier, James Cook University SCUBA Dive Club 1992 - 1993.

Latest CV

Book: 35 Years in Software Development

Latest CV: https://www.pymblesoftware.com/cv.pdf

Book: Programming bada

Book a meeting with me

Latest CV: https://www.pymblesoftware.com/cv.pdf