Powering an App 📱 with Swift Microservices

~

Who: Jarrod Parkes
What: Swift Cloud Workshop 2
When: Sept 30th, 2017 @ 2:10-2:40 PM CST
Where: Amazon Inc, Austin, TX, USA 🇺🇸
Why: We ❤️ Swift!

👋 Hey, I'm Jarrod

- I build iOS/Swift courses at Udacity
- I've been working on a "Server-Side Swift" course
- Course features "Game Night!"
  - An iOS app powered by Swift microservices

Talk Outline 📋

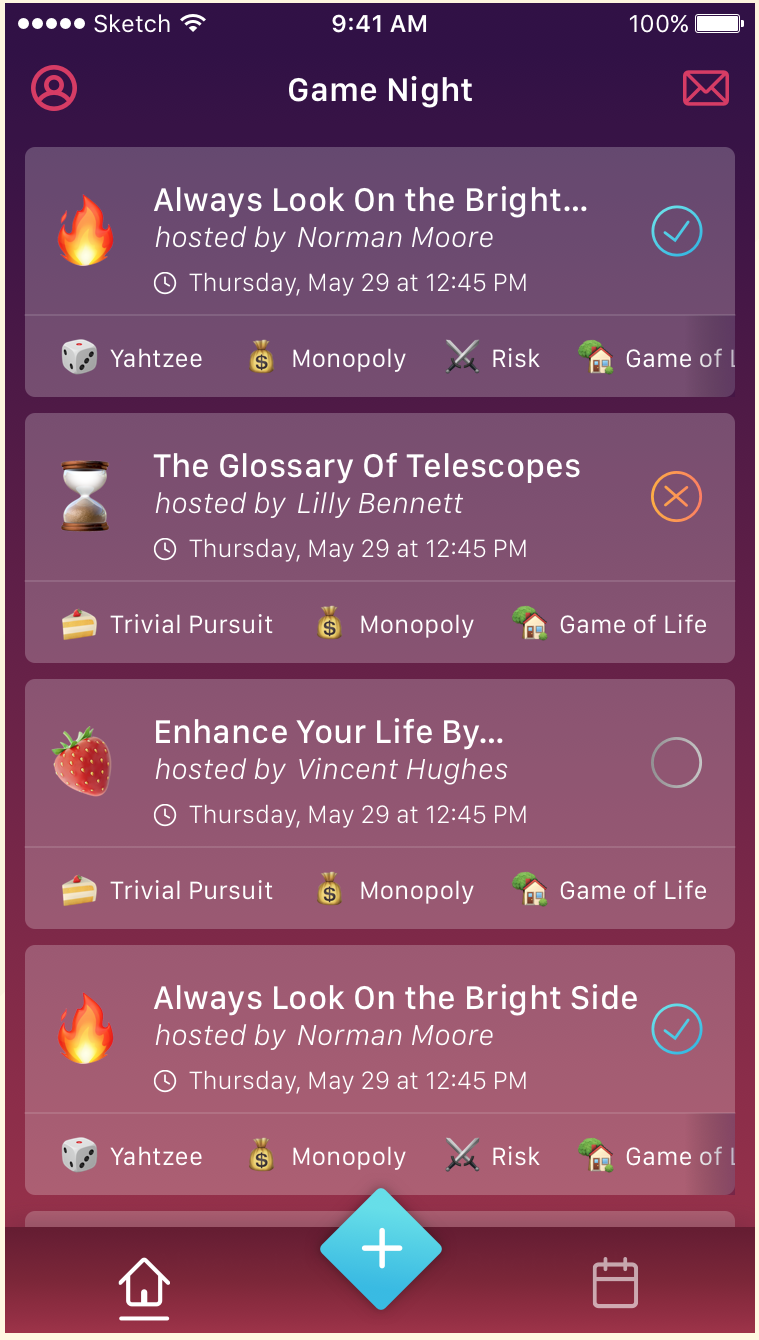
- Developing Game Night! 🎲
- The ups and downs 📶
- Future development 🚀

Where It Started 

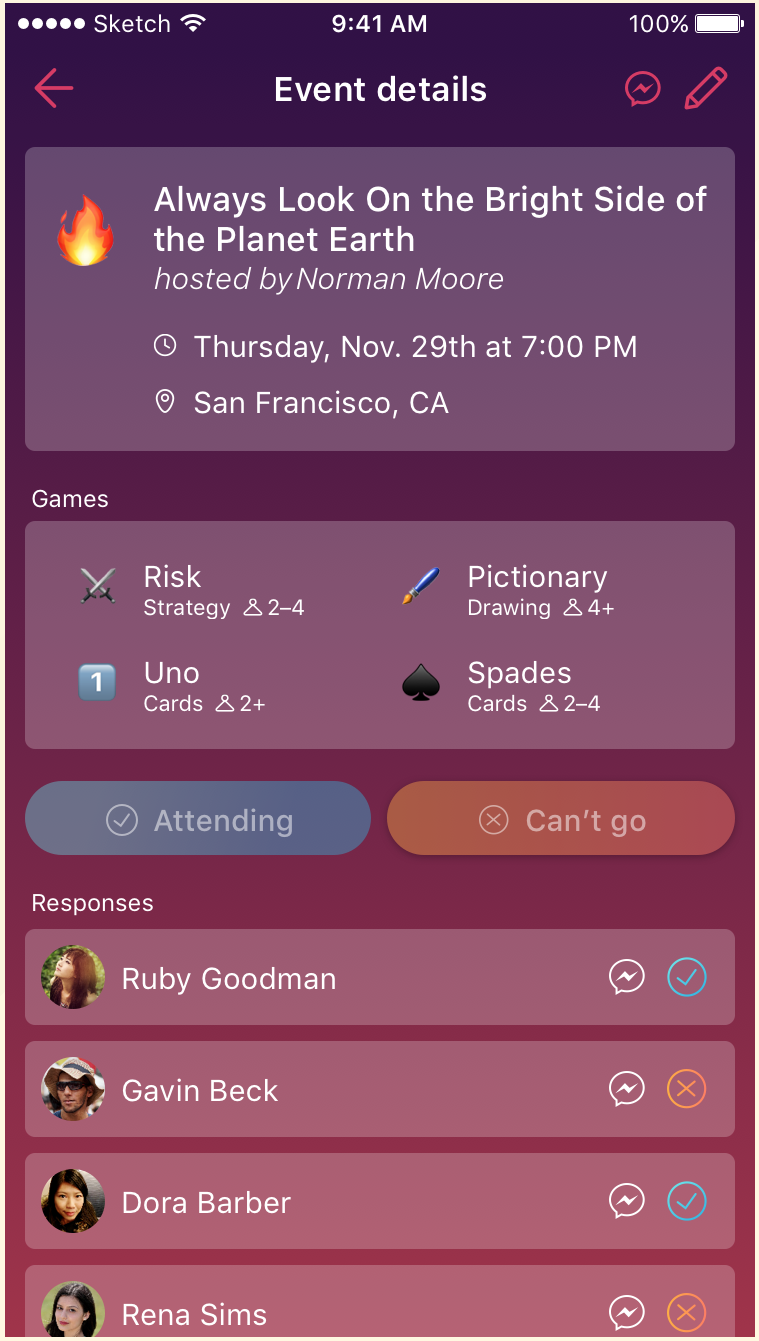
- Course v1 launched @ IBM InterConnect 2017
  - Swift on Linux
  - Swift Package Manager
  - IBM Cloud Tools, ToDo app
  - Built with Kitura

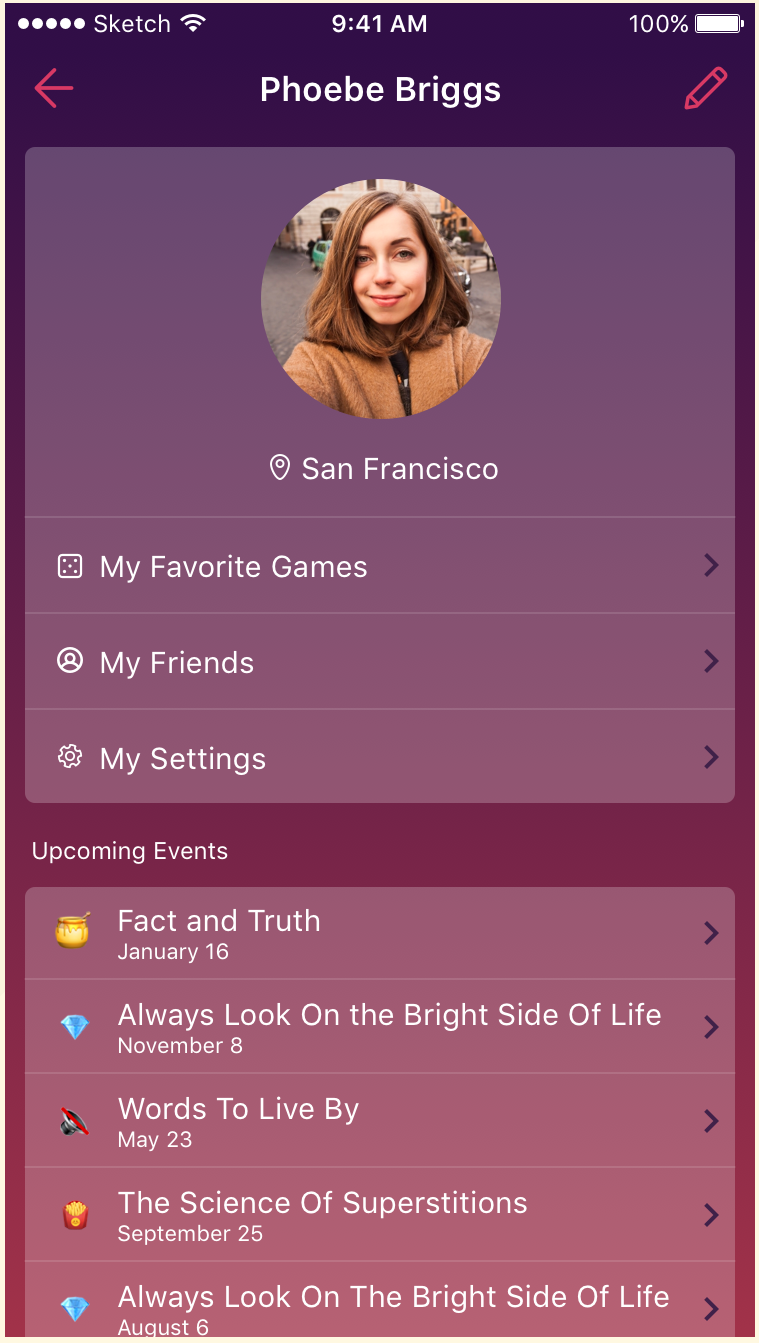
We Can Do More! 🎲

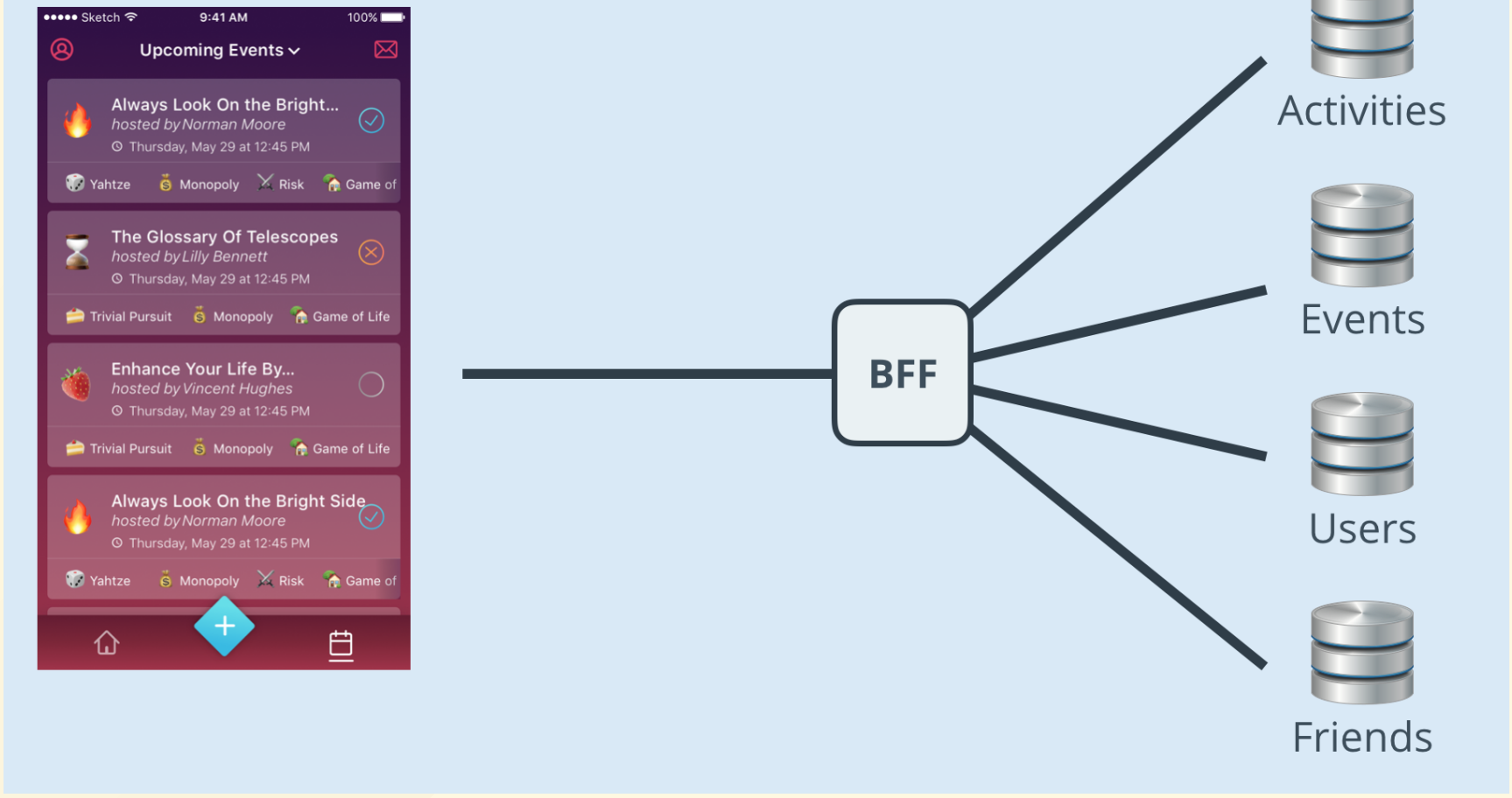
- Course v1 was "ok", but missing a spark
- Let's build something *compelling*!


●●●●● Sketch
9:41 AM
100%
Game Night
Always Look On the Bright...
hosted by Norman Moore
Thursday, May 29 at 12:45 PM
Yahtzee
Monopoly
Risk
Game of
The Glossary Of Telescopes
hosted by Lilly Bennett
Thursday, May 29 at 12:45 PM
Trivial Pursuit
Monopoly
Game of Life
Enhance Your Life By...
hosted by Vincent Hughes
Thursday, May 29 at 12:45 PM
Trivial Pursuit
Monopoly
Game of Life
Always Look On the Bright Side
hosted by Norman Moore
Thursday, May 29 at 12:45 PM
Yahtzee
Monopoly
Risk
Game of


Sketch
9:41 AM
100%
Monopoly
2–6 players
Adventure
4
Friends like this game.
6
Friends are playing this.
Game Description
Monopoly is a board game that originated in the United States in 1903 as a way to demonstrate that an economy which rewards wealth creation is better than one in which monopolists work under few constraints and to promote the economic theories of Henry George and in particular his ideas about taxation.
Players move around the game-board buying, trading, or selling properties, developing their properties with houses and hotels, and collecting rent from their opponents, with the goal being to drive them all into bankruptcy, leaving one monopolist in control of the economy.


●●●●● Sketch
9:41 AM
100%
Event details
Always Look On the Bright Side of the Planet Earth
hosted by Norman Moore
Thursday, Nov. 29th at 7:00 PM
San Francisco, CA
Games
Risk
Strategy 2–4
Pictionary
Drawing 4+
Uno
Cards 2+
Spades
Cards 2–4
Attending
Can't go
Responses
Ruby Goodman
Gavin Beck
Dora Barber
Rena Sims


●●●●● Sketch
9:41 AM
100%
Phoebe Briggs
San Francisco
My Favorite Games
My Friends
My Settings
Upcoming Events
Fact and Truth
January 16
Always Look On the Bright Side Of Life
November 8
Words To Live By
May 23
The Science Of Superstitions
September 25
Always Look On The Bright Side Of Life
August 6


Sketch
9:41 AM
100%
Upcoming Events
Always Look On the Bright...
hosted by Norman Moore
Thursday, May 29 at 12:45 PM
Yahtze
Monopoly
Risk
Game of
The Glossary Of Telescopes
hosted by Lilly Bennett
Thursday, May 29 at 12:45 PM
Trivial Pursuit
Monopoly
Game of Life
Enhance Your Life By...
hosted by Vincent Hughes
Thursday, May 29 at 12:45 PM
Trivial Pursuit
Monopoly
Game of Life
Always Look On the Bright Side
hosted by Norman Moore
Thursday, May 29 at 12:45 PM
Yahtze
Monopoly
Risk
Game of
BFF
Activities
Events
Users
Friends

Activities Microservice 🎲

- Student's first microservice from scratch
- CRUD microservice
- Uses MySQL client
  - https://github.com/nicholasjackson/swift-mysql
  - Supports pagination, transactions, and stored procedures
  - Imperative style, easy to test ✅

MySQL Client: Init 

```
// Create connection pool
var pool = MySQLConnectionPool(
    connectionString: connectionString,
    poolSize: 10,
    defaultCharset: "utf8mb4")

// Create data accessor
var dataAccessor = ActivityMySQLDataAccessor(pool: pool)
```

MySQL Client: Querying 

```
// In data accessor...

do {
    // Get connection
    let connection = try pool.getConnection()
    defer { pool.releaseConnection(connection!) }

    // Execute query
    let result = try connection!.execute(builder: query)
    let activities = result.toActivities()

    // Return results...

} catch { /* Error */ }
```

Activities: Data 🎲

```
public struct Activity {
    public var id: Int?
    public var name: String?
    public var emoji: String?
    public var description: String?
    public var genre: String?
    public var minParticipants: Int?
    public var maxParticipants: Int?
    public var createdAt: Date?
    public var updatedAt: Date?
}
```

Activities: Example Query 

```
public func updateActivity(_ activity: Activity)
    throws -> Bool {

    // Build safe query
    let updateQuery = MySQLQueryBuilder()
        .update(data: activity.toMySQLRow(),
            table: "activities")
        .wheres(statement: "WHERE Id=?",
            parameters: "\(activity.id!)")

    // Execute query
    let result = try execute(builder: updateQuery)

    // Return results
    return result.affectedRows > 0
}
```

Activities Server 

```
// Create router
let router = Router()

// CRUD endpoints
router.get("/activities/:id", handler: handlers.getAct)
router.post("/activities", handler: handlers.postAct)
router.put("/activities/:id", handler: handlers.putAct)
router.delete("/activities/:id", handler: handlers.delAct)

// Add and start server
Kitura.addHTTPServer(onPort: 8080, with: router)
Kitura.run()
```

Activities Demo 

Users Microservice 

- The auth microservice
- Integrates with Facebook's AccountKit and Perfect-Crypto
- Generates JWTs

Users: Login 

Account Kit Login Demo

Users: Token Exchange 

```
// Create URL
guard let url = getURLWithPath("/access_token",
    withParameters: [
        "grant_type": "authorization_code",
        "code": code,
        "access_token": "AA|\(appID)|\(appSecret)"
    ]) else { /* Error */ }

// Perform token exchange
let task = session.dataTaskWithURL(url) {
    (data, response, error) in

        // Get Account Kit id
}
task.resume()
```

Users: Create JWT 🎲

```
 do {
    // Make payload
    let payload: [String: Any] = [
        "perms": "usersAll,activities,events,friends",
        "user": id,
        /* Reserved claims also included */
    ]

    // Create token
    let jwt = try self.jwtComposer.createToken(payload)

    // Send response
    try response.send(json: JSON(["jwt": jwt, "id": id]))
                .status(.OK).end()

} catch { /* Send error response */ }
```

Users: JWT Middleware 

```
do {
    // Get JWTVerifier
    let verifier = try composer.getVerifier(token)

    // Verify alg, key, and claims
    try composer.verifyAlgorithmAndKey(verifier)
    try composer.verifyReservedClaims(verifier,
        iss: "http://gamenight.udacity.com",
        sub: "users microservice"
    )
    try composer.verifyPrivateClaims(verifier)

    // Add user id to request data...

} catch { /* Send error response */ }

next()
```

Users Server 

```
// Create router
let router = Router()

// Use JWT middleware
router.get("/*", middleware: JWTMiddleware(
    jwtComposer: jwtComposer,
    permissions: [.usersProfile, .usersAll]
))
router.get("/users", handler: handlers.getUsers)

// Path without JWT middleware
router.post("/users/login", handler: handlers.login)

// Add and start server
Kitura.addHTTPServer(onPort: 8080, with: router)
Kitura.run()
```

Users Demo 

To Be Continued... 🎲

- BFF
- Deploy services with Terraform
- New services
  - Notifications
  - Event Queue
  - Search 🔍
- Heavier focus on testing

What Went Well 

- Easy to setup minimally complex CRUD services
- Sharing of model
  - Will become even better w/ Codable
- Containerization FTW
  - Services deployable to AWS, Bluemix, GCS, Azure

What Went Wrong 📶

- Had to sink time into tooling
  - This tooling already exists in other ecoystems
- Tons to learn for newcomers (backend newbies)
  - ^ Me included ✋
- Teaching around a moving target 🎯

Review

- New course launch 🎉
  - https://www.udacity.com/course/server-side-swift--ud1031
- Advocacy by building a *compelling* app 📱
- Tooling isn't the strongest, but is getting better
- Need watershed moment(s) 💧 !

Questions ✋?

Suggestions ✋?

Slides available @ jarrodparkes.com

Thank You 🙏!

- Game Night App (link coming soon)
- Activities Microservice
- Events Microservice
- Users Microservice
- Friends Microservice
- Let's Deploy Swift
  - Deployed to AWS, Bluemix, Heroku, GCS, Azure, and more