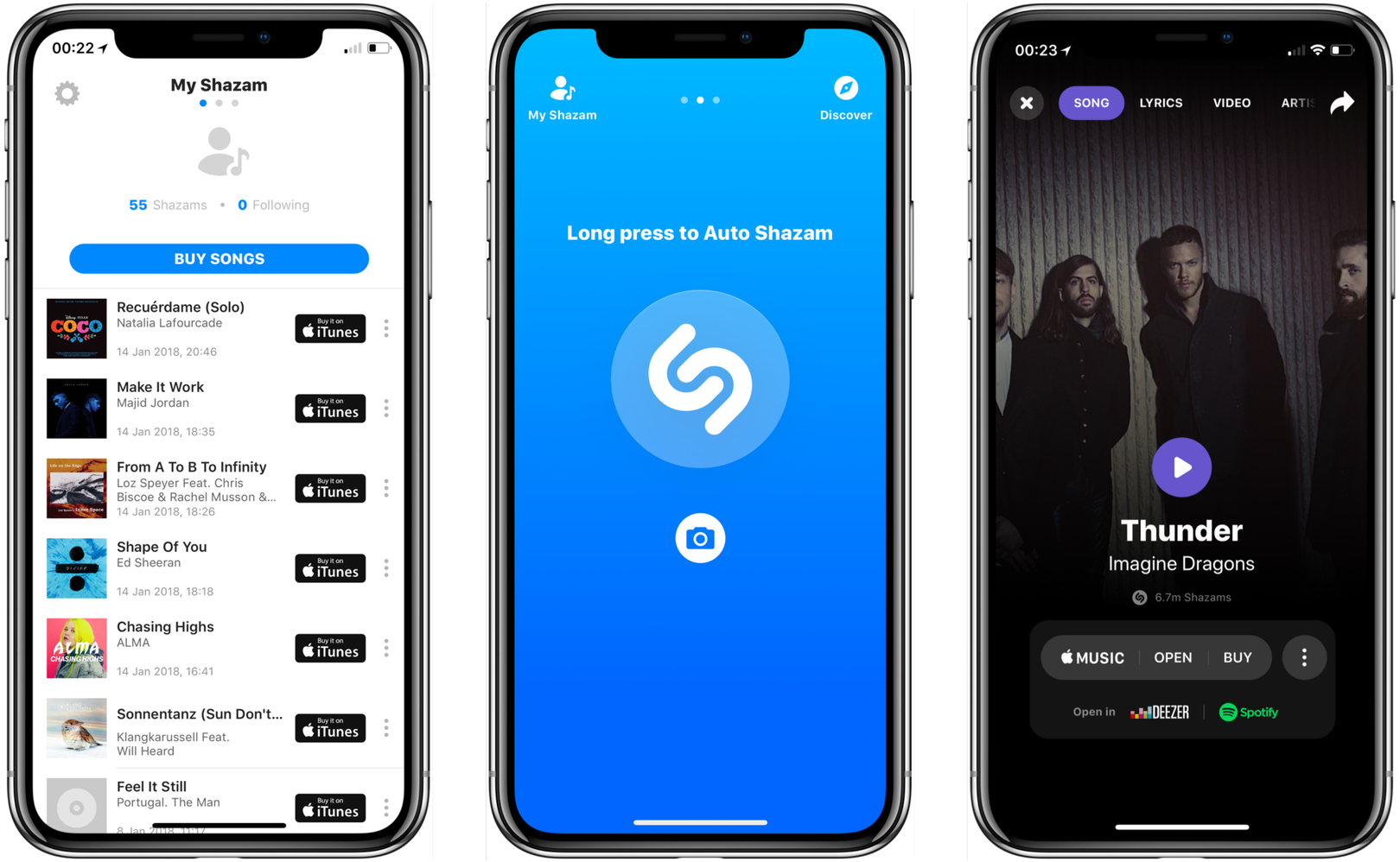
# Introduction to iOS Development

Session 101

Alex Telek
Shazam iOS Engineer

# Shazam


00:22
My Shazam
55 Shazams • 0 Following
BUY SONGS
Recuérdame (Solo)
Natalia Lafourcade
14 Jan 2018, 20:46
Make It Work
Majid Jordan
14 Jan 2018, 18:35
From A To B To Infinity
Loz Speyer Feat. Chris Biscoe & Rachel Musson &...
14 Jan 2018, 18:26
Shape Of You
Ed Sheeran
14 Jan 2018, 18:18
Chasing Highs
ALMA
14 Jan 2018, 16:41
Sonnentanz (Sun Don't...
Klangkarussell Feat. Will Heard
Feel It Still
Portugal. The Man
Buy it on iTunes
My Shazam
Discover
Long press to Auto Shazam
00:23
SONG
LYRICS
VIDEO
Thunder
Imagine Dragons
6.7m Shazams
MUSIC
OPEN
BUY
Open in
DEEZER
Spotify

# What You Will Learn

# What You Will Learn

iOS 11

# What You Will Learn

iOS 11

OS X 10.13

# What You Will Learn


iOS 11


OS X 10.13


watchOS 4

# What You Will Learn


iOS 11


OS X 10.13


watchOS 4


tvOS 11

# What You Will Learn


iOS 11


OS X 10.13


watchOS 4


tvOS 11

Swift 4

# What You Will Learn

iOS 11

OS X 10.13

watchOS 4


tvOS 11

Swift 4

Xcode 9

# iOS Sessions

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

Intermediate Swift

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

Intermediate Swift

How iOS works, Life cycle, MVC

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

Intermediate Swift

How iOS works, Life cycle, MVC

UI - Auto Layout

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

Intermediate Swift

How iOS works, Life cycle, MVC

UI - Auto Layout

Data & Parsing

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

Intermediate Swift

How iOS works, Life cycle, MVC

UI - Auto Layout

Data & Parsing

Networking, Threading

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

Intermediate Swift

How iOS works, Life cycle, MVC

UI - Auto Layout

Data & Parsing

Networking, Threading

Location

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

Intermediate Swift

How iOS works, Life cycle, MVC

UI - Auto Layout

Data & Parsing

Networking, Threading

Location

ARKit

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

Intermediate Swift

How iOS works, Life cycle, MVC

UI - Auto Layout

Data & Parsing

Networking, Threading

Location

ARKit

Quickly becomes challenging

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

Intermediate Swift

How iOS works, Life cycle, MVC

UI - Auto Layout

Data & Parsing

Networking, Threading

Location

ARKit

Quickly becomes challenging

Learn by doing, not by listening!

# iOS Sessions

Every week - Wednesdays @ 6:30pm - Facebook

Start with basics (iOS, Swift, Xcode)

Intermediate Swift

How iOS works, Life cycle, MVC

UI - Auto Layout

Data & Parsing

Networking, Threading

Location

ARKit

Quickly becomes challenging

Learn by doing, not by listening!

Public Slack kcltechhq.slack.com - #ios-programming, ios.kcl.tech

# GitHub

# GitHub

KCLTech iOS sessions on GitHub

# GitHub

KCLTech iOS sessions on GitHub

Weekly update

# GitHub

KCLTech iOS sessions on GitHub

Weekly update

Materials

# GitHub

KCLTech iOS sessions on GitHub

Weekly update

Materials

Videos

# GitHub

KCLTech iOS sessions on GitHub

Weekly update

Materials

Videos

Discussions

# GitHub

KCLTech iOS sessions on GitHub

Weekly update

Materials

Videos

Discussions

https://goo.gl/5ofrVq

# WWDC18

# WWDC18

World Wide Developer Conference in San Francisco

# WWDC18

World Wide Developer Conference in San Francisco

~300 Student Scholarship Recipients

# WWDC18

World Wide Developer Conference in San Francisco

~300 Student Scholarship Recipients

1.Swift Playground

- Visually interactive
- Experienced within three minutes
- Book
- Graphics, Audio and more

# WWDC18

World Wide Developer Conference in San Francisco

~300 Student Scholarship Recipients

1.Swift Playground

- Visually interactive
- Experienced within three minutes
- Book
- Graphics, Audio and more

2. Essay (500 words each)

- Describe your Swift Playground
- Beyond WWDC
- Assistance

# WWDC18

World Wide Developer Conference in San Francisco

~300 Student Scholarship Recipients

1.Swift Playground
- Visually interactive
- Experienced within three minutes
- Book
- Graphics, Audio and more

2. Essay (500 words each)
- Describe your Swift Playground
- Beyond WWDC
- Assistance

3. Judging
- Technical accomplishment
- Creativity of ideas
- Content of written responses

# WWDC18

World Wide Developer Conference in San Francisco

~300 Student Scholarship Recipients

1.Swift Playground
- Visually interactive
- Experienced within three minutes
- Book
- Graphics, Audio and more

2. Essay (500 words each)
- Describe your Swift Playground
- Beyond WWDC
- Assistance


3. Judging
- Technical accomplishment
- Creativity of ideas
- Content of written responses

# WWDC18

World Wide Developer Conference in San Francisco

~300 Student Scholarship Recipients

1.Swift Playground
- Visually interactive
- Experienced within three minutes
- Book
- Graphics, Audio and more

2. Essay (500 words each)
- Describe your Swift Playground
- Beyond WWDC
- Assistance


3. Judging
- Technical accomplishment
- Creativity of ideas
- Content of written responses

Deadline: April, 2018

# Xcode 9


Trivia
Play Trivia with Your Frien
Start a new game
Find an existing game
Sign out
Add New Constraints

# Xcode 9

Swift 4

# Xcode 9

Swift 4

watchOS 4

# Xcode 9


Swift 4

watchOS 4

tvOS 11

# Xcode 9


Swift 4

watchOS 4

tvOS 11

OS X

# Xcode 9


Swift 4

watchOS 4

tvOS 11

OS X

Live design

# Xcode 9


Swift 4

watchOS 4

tvOS 11

OS X

Live design

Visual debugging

# Xcode 9


Swift 4

watchOS 4

tvOS 11

OS X

Live design

Visual debugging

Performance  testing

# Xcode 9


Swift 4

watchOS 4

tvOS 11

OS X

Live design

Visual debugging

Performance  testing

UI Testing

# Xcode 9


Swift 4

watchOS 4

tvOS 11

OS X

Live design

Visual debugging

Performance  testing

UI Testing

Code Coverage

# Xcode 9


Swift 4

watchOS 4

tvOS 11

OS X

Live design

Visual debugging

Performance  testing

UI Testing

Code Coverage

Refactor and Transform

# Xcode 9


Swift 4

watchOS 4

tvOS 11

OS X

Live design

Visual debugging

Performance  testing

UI Testing

Code Coverage

Refactor and Transform

Cut The Cord

# iOS 11


9:41
Messages
Calendar
Photos
Camera
Weather
Clock
Maps
News
Wallet
Notes
Reminders
Stocks
iTunes Store
App Store
Books
Health
Settings
9:41
Search
WEATHER
Show More
San Francisco
Sunny
Chance of Rain: 0%
65°
66° / 55°
UP NEXT
Lunch
12:00 – 1:00 PM
SIRI APP SUGGESTIONS
Show More
Photos
Weather
Settings
Maps
FAVORITES
Show More
Katie
Kelly
Yumi
John
Edit
Weather
San Francisco
Sunny
75°
Thursday TODAY
Now 1PM 2PM 3PM 4PM
75° 77° 78° 78° 76°
Friday
Saturday
Sunday
Monday
Tuesday
Wednesday
Thursday
Mailboxes
Inbox

# iOS 11


iPhone X

# iOS 11


iPhone X

Drag and Drop

# iOS 11


iPhone X

Drag and Drop

Apple Pencil

# iOS 11


iPhone X

Drag and Drop

Apple Pencil

Augmented Reality

# iOS 11


iPhone X

Drag and Drop

Apple Pencil

Augmented Reality

SiriKit

# iOS 11


iPhone X

Drag and Drop

Apple Pencil

Augmented Reality

SiriKit

Camera

# iOS 11


iPhone X

Drag and Drop

Apple Pencil

Augmented Reality

SiriKit

Camera

AirPlay 2

# iOS 11


iPhone X

Drag and Drop

Apple Pencil

Augmented Reality

SiriKit

Camera

AirPlay 2

MusicKit

*Demo*

Xcode 9

# Swift 4

# Swift 4


# Swift 4



# Swift 4




```c
#include <stdio.h>

int main()
{
  printf("Hello, KCL Tech\n");
  return 0;
}
```

```
print(“Hello, KCL Tech”)
```

# Variables

# Variables

```
var languageName: String = “Swift”
```

# Constants and Variables

```
let languageName: String = “Swift”
```

# Constants and Variables

SAFE

```
let languageName: String = “Swift”
```

# Constants and Variables

SAFE

```
let languageName: String = “Swift”
var version: Double = 4.0
```

# Constants and Variables

SAFE

```
let languageName: String = “Swift”
var version: Double = 4.0
var introducedIn: Int = 2014
```

# Constants and Variables

SAFE

```
let languageName: String = “Swift”
var version: Double = 4.0
var introducedIn: Int = 2014
var isAwesome: Bool = true
```

# Constants and Variables

SAFE

```
let languageName: String = “Swift”
var version: Double = 4.0
let introducedIn: Int = 2014
let isAwesome: Bool = true
```

# Type Inference

SAFE

```
let languageName: String = “Swift”
var version: Double = 4.0
let introducedIn: Int = 2014
let isAwesome: Bool = true
```

# Type Inference

SAFE

```
let languageName = “Swift”
var version = 4.0
let introducedIn = 2014
let isAwesome = true
```

# Unicode Names

```
let languageName = “Swift”
var version = 4.0
let introducedIn = 2014
let isAwesome = true
```

# Unicode Names

```
let languageName = “Swift”
var version = 4.0
let introducedIn = 2014
let isAwesome = true
let π = 3.14159265
```

# Unicode Names

```
let languageName = “Swift”
var version = 4.0
let introducedIn = 2014
let isAwesome = true
let π = 3.14159265
let 🐶🐮 = “dogcow”
```

# Combining Strings and Characters

# Combining Strings and Characters

```
let dog = “dog”
```

# Combining Strings and Characters

```
let dog = “dog”
let cow = “cow”
```

# Combining Strings and Characters

```
let dog = “dog”
let cow = “cow”
let dogCow = dog + cow
// dogCow is “dogcow”
```

# Building Complex Strings

POWER

# Building Complex Strings

POWER

```
let a = 3
let b = 5

// “3 times 5 is 15”
```

# Building Complex Strings

POWER

```
let a = 3
let b = 5

// “3 times 5 is 15”

let result = “\(a) times \(b) is \(a * b)”
```

# String Mutability

# String Mutability

```
var variableString = “Horse”
```

# String Mutability

```
var variableString = “Horse”
variableString += “ and carriage”
//variableString is now “Horse and carriage”
```

# String Mutability

```
var variableString = “Horse”
variableString += “ and carriage”
//variableString is now “Horse and carriage”

let constantString = “Horse”
```

# String Mutability

```
var variableString = “Horse”
variableString += “ and carriage”
//variableString is now “Horse and carriage”

let constantString = “Horse”
constantString += “ and carriage”
//error
```

# Array and Dictionary

```
var names = [“Anna”, “Brian”, “Jack”]
```

```
var numberOfLegs = [“ant”: 6, “snake”: 0]
```

# Typed Collections

```
var names = [“Anna”, “Brian”, “Jack”]
```

# Typed Collections

```
var names = [“Anna”, “Brian”, “Jack”, 42]
```

# Typed Collections

```
var names = [“Anna”, “Brian”, “Jack”, true]
```

# Typed Collections

```
var names: String[] = [“Anna”, “Brian”, “Jack”]
```

```
var ages = Int[]()
let values = Int[](count: 5, repeatedValue: 1)
```

# Loops

# Loops

```
while hungry {
  eatCake()
}
```

# Loops

```
while hungry {
  eatCake()
}
```

```
for var i = 0; i < 10; i++ {
  eat(i)
}
```

# Loops

```
while hungry {
  eatCake()
}
```

```
var index = 0
repeat {
  index++
} while index < 5
```

```
for var i = 0; i < 10; i++ {
  eat(i)
}
```

# For-In: Strings and Characters


# For-In: Strings and Characters


```
for character in "🐭🐭🐭🐭🐭" {
  print(character)
}
```

# For-In: Strings and Characters

POWER

```
for character in "🐭🐭🐭🐭🐭" {
  print(character)
}
```

```
🐭
🐭
🐭
🐭
🐭
```

# For-In: Ranges

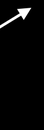
POWER

# For-In: Ranges


```
for number in 1…5 {
  print(“\(number) times 4 is \(number*4)”)
}
```

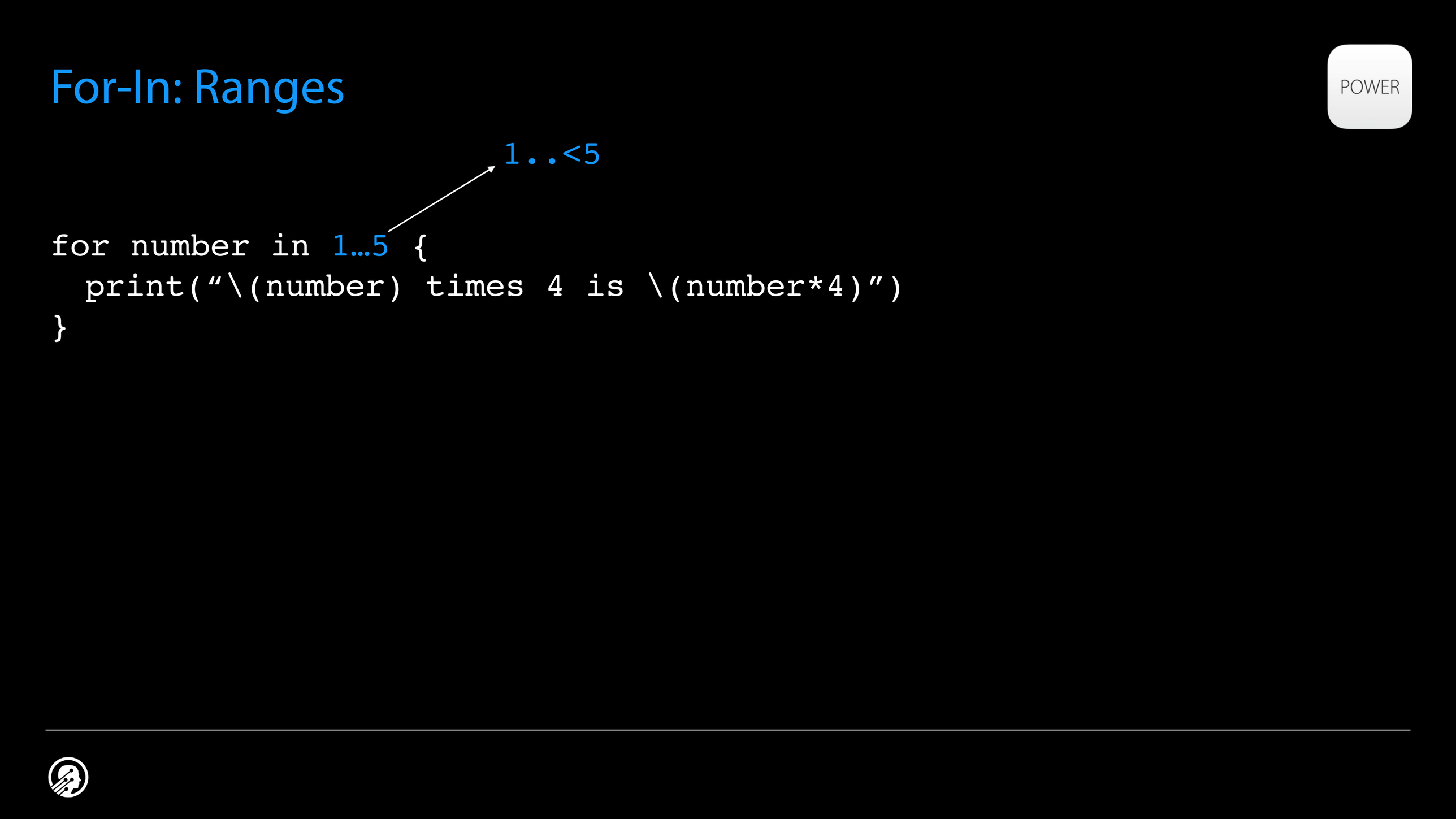
# For-In: Ranges

POWER



```
for number in 1…5 {
  print(“\(number) times 4 is \(number*4)”)
}
```

# For-In: Ranges




1..<5

```
for number in 1…5 {
  print(“\(number) times 4 is \(number*4)”)
}
```

# For-In: Ranges

POWER

1..<5

```
for number in 1…5 {
  print(“\(number) times 4 is \(number*4)”)
}
```

```
1 times 4 is 4
2 times 4 is 8
3 times 4 is 12
4 times 4 is 16
5 times 4 is 20
```

# For-In: Arrays

POWER

# For-In: Arrays

POWER

```
for name in [“Anna”, “Brian”, “Jack”] {
  print(“Hello, \(name)!”)
}
```

# For-In: Arrays

POWER

```
for name in [“Anna”, “Brian”, “Jack”] {
  print(“Hello, \(name)!”)
}
```

```
Hello, Anna!
Hello, Brian!
Hello, Jack!
```

# For-In: Dictionaries

POWER

# For-In: Dictionaries


```
var numberOfLegs = [“ant”: 6, “snake”: 0, “cheetah”: 4]
```

# For-In: Dictionaries


```
var numberOfLegs = [“ant”: 6, “snake”: 0, “cheetah”: 4]


for (animal, leg) in numberOfLegs {
  print(“\(animal)s have \(leg) legs”)
}
```

# For-In: Dictionaries

POWER

```
var numberOfLegs = [“ant”: 6, “snake”: 0, “cheetah”: 4]

for (animal, leg) in numberOfLegs {
  print(“\(animal)s have \(leg) legs”)
}
```

```
ants have 6 legs
snakes have 0 legs
cheetahs have 4 legs
```

# If Statements

# If Statements

```
if legCount == 0 {

  print(“It slides”)

} else {

  print(“It walks”)
}
```

# More Complex If Statements

# More Complex If Statements

```
if legCount == 0 {

  print(“It slides”)

} else if legCount == 1 {

  print(“It hops”)

} else {

  print(“It walks”)
}
```

# Switch

# Switch

```
switch legCount {
  case 0:
    print(“It slides”)

  case 1:
    print(“It slides”)

  default:
    print(“It slides”)
}
```

# Switch

# Switch

```
switch legCount {
  case 0:
    print(“It slides”)

  case 1,3,5,7,9:
    print(“It hops”)

  case 2,4,6,8,10:
    print(“It walks”)

  default:
    print(“No idea”)
}
```

# Switch


# Switch


```
switch textField {
  case userNameTextField:
    print(“You tapped the username text field”)

  case passwordTextField:
    print(“You tapped the password text field”)

  default:
    print(“You tapped some other object”)
}
```

# Switch

# Switch

```
let point = (1, -1)

switch point {
  case let (x, y) where x == y:
    print(“x equals with y”)

  case let (x, y) where x == -y:
    print(“x is the abs of y”)

  case let (x, y):
    print(“x and y are two coordinates”)
}
```

# Functions

# Functions

```
func sayHello() {
  print(“Hello!”)
}
```

# Functions

```
func sayHello() {
  print(“Hello!”)
}

sayHello()
```

# Functions

```
func sayHello() {
  print(“Hello!”)
}

sayHello()
```

```
Hello
```

# Functions with Parameters

# Functions with Parameters

```
func sayHello(name: String) {
  print(“Hello \(name)!”)
}
```

# Functions with Parameters

```
func sayHello(name: String) {
  print(“Hello \(name)!”)
}

sayHello(“WWDC”)
```

# Functions with Parameters

```
func sayHello(name: String) {
  print(“Hello \(name)!”)
}

sayHello(“WWDC”)
```

```
Hello, WWDC!
```

# Functions with Parameters

# Functions with Parameters

```
func sayHello(name: String, age: Int) {
  print(“Hello \(name), \(age)!”)
}
```

# Functions with Parameters

```
func sayHello(name: String, age: Int) {
  print(“Hello \(name), \(age)!”)
}

sayHello(“Peter”, age: 20)
```

# Functions with Parameters

```
func sayHello(name: String, age: Int) {
  print(“Hello \(name), \(age)!”)
}

sayHello(“Peter”, age: 20)


func sayHello(name: String, _ age: Int) {
  print(“Hello \(name), \(age)!”)
}
```

# Functions with Parameters

```
func sayHello(name: String, age: Int) {
  print("Hello \(name), \(age)!")
}

sayHello("Peter", age: 20)


func sayHello(name: String, _ age: Int) {
  print("Hello \(name), \(age)!")
}

sayHello("Peter", 20)
```

# Functions with Parameters

# Functions with Parameters

```
func addListOfParams(params: String…) {
  // Do something with params
}
```

# Functions with Parameters

```
func addListOfParams(params: String…) {
  // Do something with params
}

addListOfParams(“Name”, “Peter”, “Age”, “Cat”)
```

# Returning Values

# Returning Values

```
func sayHello(name: String) -> String {
  return “Hello ” + name
}
```

# Returning Values

```
func sayHello(name: String) -> String {
  return “Hello ” + name
}

let greeting = sayHello(“WWDC”)
```

# Returning Values

```
func sayHello(name: String) -> String {
  return “Hello ” + name
}

let greeting = sayHello(“WWDC”)

println(greeting)
```

# Returning Values

```
func sayHello(name: String) -> String {
  return “Hello ” + name
}

let greeting = sayHello(“WWDC”)

println(greeting)
```

```
Hello, WWDC
```

# Returning Multiple Values

MODERN

# Returning Multiple Values

MODERN

```
func refreshWebPage() -> (Int, String) {
  //…try to refresh…

  return (200, “Success”)
}

let (code, status) = refreshWebPage()
print(status)
```

# Functions in General

# Functions in General

```
func functionName(variableName: ParamType) -> ReturnType {
  return Variable as ReturnType
}
```

*Demo*

Swift in Playground

KCL TECH SOCIETY