

Introduction to iOS Development

Session 101

Alex Telek
Clarence Ji
KCL Tech iOS Engineers

What You Will Learn



What You Will Learn



iOS 9



What You Will Learn



iOS 9



watchOS 2



What You Will Learn



iOS 9



watchOS 2



tvOS 9



What You Will Learn



iOS 9



OS X 10.11



watchOS 2



tvOS 9



What You Will Learn



iOS 9



OS X 10.11



watchOS 2



tvOS 9



Swift 2



What You Will Learn



iOS 9



OS X 10.11



watchOS 2



tvOS 9



Swift 2



Xcode 7



iOS Sessions



iOS Sessions

Every week - Tuesday @ 6:30pm, JCMB B.17



iOS Sessions

Every week - Tuesday @ 6:30pm, JCMB B.17

Start with basics (iOS, OS X, watchOS, tvOS, Swift, Xcode)



iOS Sessions

Every week - Tuesday @ 6:30pm, JCMB B.17

Start with basics (iOS, OS X, watchOS, tvOS, Swift, Xcode)

UI



iOS Sessions

Every week - Tuesday @ 6:30pm, JCMB B.17

Start with basics (iOS, OS X, watchOS, tvOS, Swift, Xcode)

UI

Custom Views and Animations



iOS Sessions

Every week - Tuesday @ 6:30pm, JCMB B.17

Start with basics (iOS, OS X, watchOS, tvOS, Swift, Xcode)

UI

Custom Views and Animations

Concurrency and Multithreading



iOS Sessions

Every week - Tuesday @ 6:30pm, JCMB B.17

Start with basics (iOS, OS X, watchOS, tvOS, Swift, Xcode)

UI

Custom Views and Animations

Concurrency and Multithreading

3D Touch



iOS Sessions

Every week - Tuesday @ 6:30pm, JCMB B.17

Start with basics (iOS, OS X, watchOS, tvOS, Swift, Xcode)

UI

Custom Views and Animations

Concurrency and Multithreading

3D Touch

Foundation, UIKit, CoreLocation, CoreImage, CoreData, MapKit, CloudKit, WatchConnectivity, ClockKit....



iOS Sessions

Every week - Tuesday @ 6:30pm, JCMB B.17

Start with basics (iOS, OS X, watchOS, tvOS, Swift, Xcode)

UI

Custom Views and Animations

Concurrency and Multithreading

3D Touch

Foundation, UIKit, CoreLocation, CoreImage, CoreData, MapKit, CloudKit, WatchConnectivity, ClockKit....

Quickly becomes challenging



iOS Sessions

Every week - Tuesday @ 6:30pm, JCMB B.17

Start with basics (iOS, OS X, watchOS, tvOS, Swift, Xcode)

UI

Custom Views and Animations

Concurrency and Multithreading

3D Touch

Foundation, UIKit, CoreLocation, CoreImage, CoreData, MapKit, CloudKit, WatchConnectivity, ClockKit....

Quickly becomes challenging

Learn by doing, not by listening!



iOS Sessions

Every week - Tuesday @ 6:30pm, JCMB B.17

Start with basics (iOS, OS X, watchOS, tvOS, Swift, Xcode)

UI

Custom Views and Animations

Concurrency and Multithreading

3D Touch

Foundation, UIKit, CoreLocation, CoreImage, CoreData, MapKit, CloudKit, WatchConnectivity, ClockKit....

Quickly becomes challenging

Learn by doing, not by listening!

Public Slack kcltechhq.slack.com - #ios-programming



iTunes U



© 2016 KCL Tech Society. All rights reserved.

iTunes U



KCLTech iOS sessions on iTunes U



iTunes U



KCLTech iOS sessions on iTunes U
Weekly update



iTunes U



KCLTech iOS sessions on iTunes U

Weekly update

Materials



iTunes U



KCLTech iOS sessions on iTunes U

Weekly update

Materials

Videos



iTunes U



KCLTech iOS sessions on iTunes U

Weekly update

Materials

Videos

Discussions



iTunes U



KCLTech iOS sessions on iTunes U

Weekly update

Materials

Videos

Discussions

Assignments



iTunes U



KCLTech iOS sessions on iTunes U

Weekly update

Materials

Videos

Discussions

Assignments

The screenshot shows the 'My Courses' section of the iTunes U app. At the top, there are navigation icons: a back arrow, a plus sign, the text 'My Courses', a user profile icon labeled 'AT', and a search icon. Below these are buttons for 'Edit' and a magnifying glass. The main area displays three course cards:

- KCL TECH SOCIETY**: Building iOS 9 apps with Swift 2.0. This card has a purple background and shows a small icon of a person with a gear.
- Stanford**: Developing iOS 7 Apps for iPhone and iPad. This card has a blue background and shows a globe.
- Stanford**: Developing iOS 8 Apps with Swift. This card has a dark background and shows a white bird.

Each course card has a red circular badge with a number (25 or 18) in the top right corner, indicating new content. At the bottom of the screen are five navigation icons: 'My Courses' (selected), 'Featured', 'Top Charts', 'Browse', and 'Search'.



iTunes U



KCLTech iOS sessions on iTunes U

Weekly update

Materials

Videos

Discussions

Assignments

A screenshot of the iTunes U mobile application. At the top, there's a navigation bar with a back arrow, a plus sign for adding courses, the title 'My Courses', a user profile icon labeled 'AT', and search/filter icons. Below the navigation is a toolbar with five items: 'My Courses' (highlighted in blue), 'Featured', 'Top Charts', 'Browse', and 'Search'. The main content area displays three course cards: 'Building iOS 9 apps with Swift 2.0' by KCL TECH SOCIETY (9 lessons, 2 quizzes), 'Developing iOS 7 Apps for iPhone and iPad' by Stanford (25 lessons), and 'Developing iOS 8 Apps with Swift' by Stanford (18 lessons).

My Courses

Edit

AT

Search

My Courses

Featured

Top Charts

Browse

Search

KCL TECH SOCIETY

Building iOS 9 apps with Swift 2.0

Stanford 25

Stanford 18

Developing iOS 7 Apps for iPhone and iPad

Developing iOS 8 Apps with Swift

FWJ-BTY-PSJ



WWDC16



WWDC16



WWDC16

World Wide Developer Conference in San Francisco



WWDC16

World Wide Developer Conference in San Francisco

300 Student Scholarship Recipients



WWDC16

World Wide Developer Conference in San Francisco

300 Student Scholarship Recipients

1. Create an App (Developer Account)

- Development Projects
- Educational background
- Professional background
- Technical skills
- Interest



WWDC16

World Wide Developer Conference in San Francisco

300 Student Scholarship Recipients

1. Create an App (Developer Account)

- Development Projects
- Educational background
- Professional background
- Technical skills
- Interest

2. Judging

- Technical accomplishment
- Creativity of ideas expressed in the app
- Technical/work experience



WWDC16

World Wide Developer Conference in San Francisco

300 Student Scholarship Recipients

1. Create an App (Developer Account)

- Development Projects
- Educational background
- Professional background
- Technical skills
- Interest

2. Judging

- Technical accomplishment
- Creativity of ideas expressed in the app
- Technical/work experience



Deadline: April, 2016

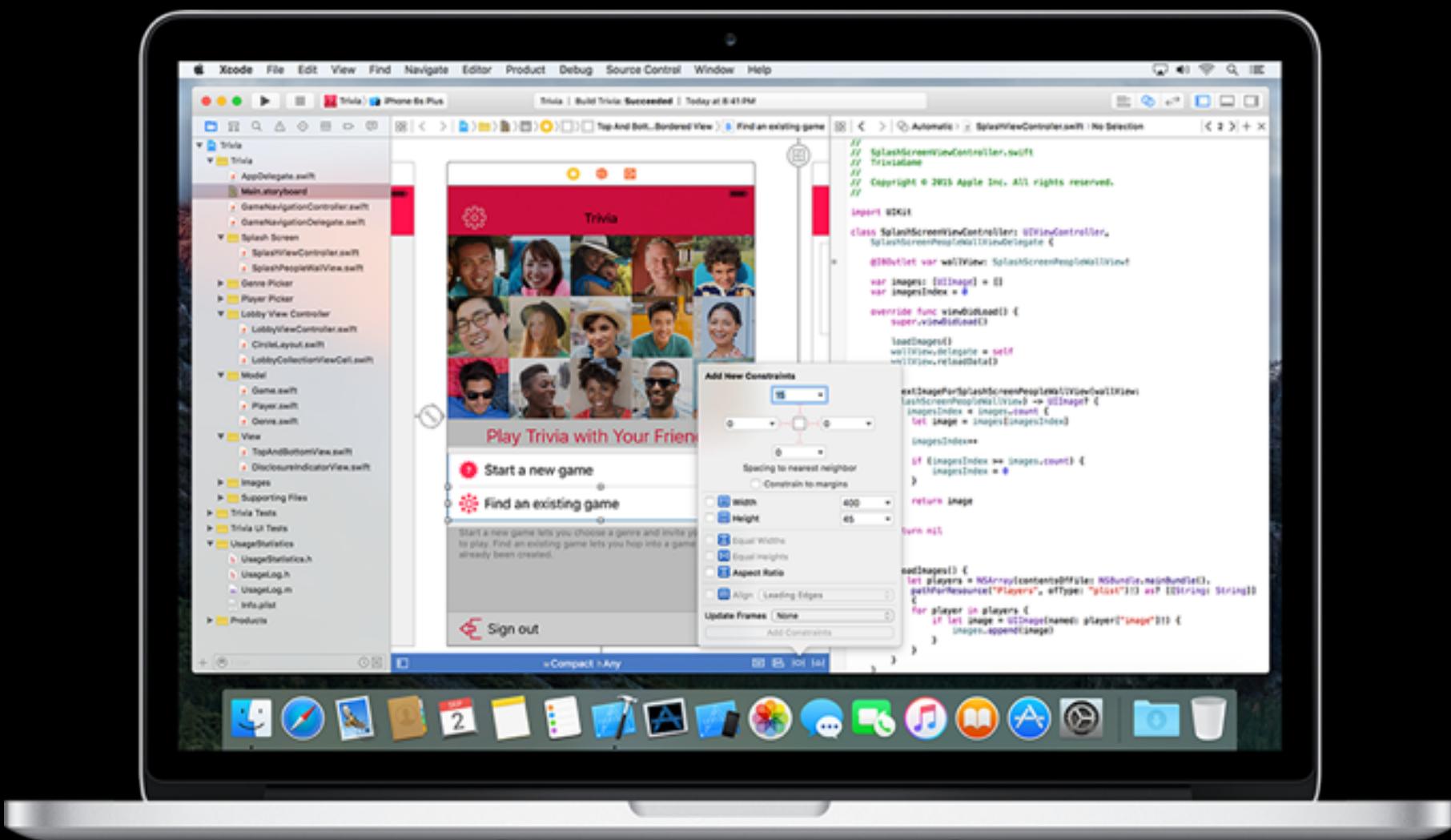


Xcode 7



Xcode 7

Swift 2



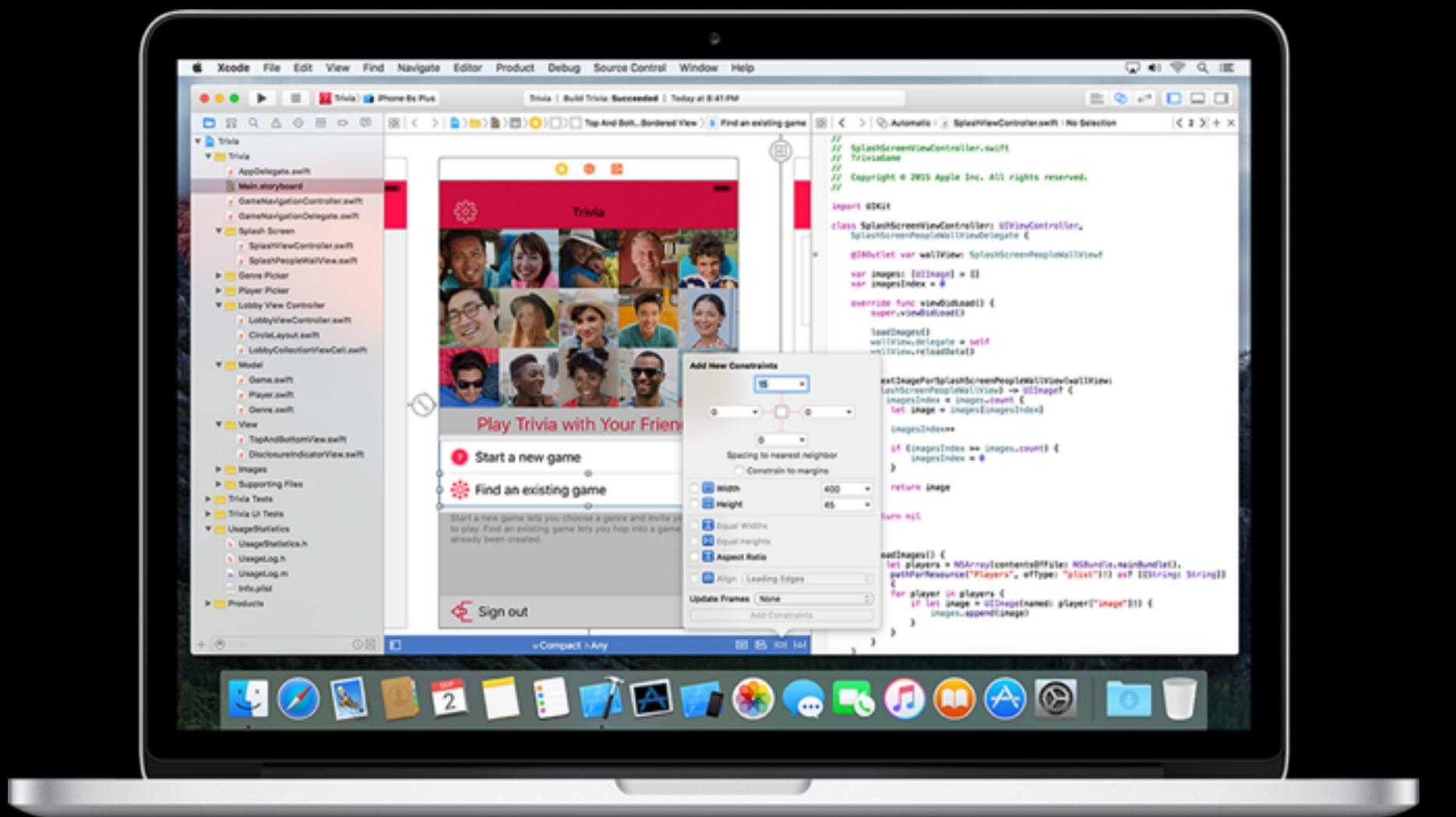
Xcode 7



Swift 2
watchOS 2



Xcode 7



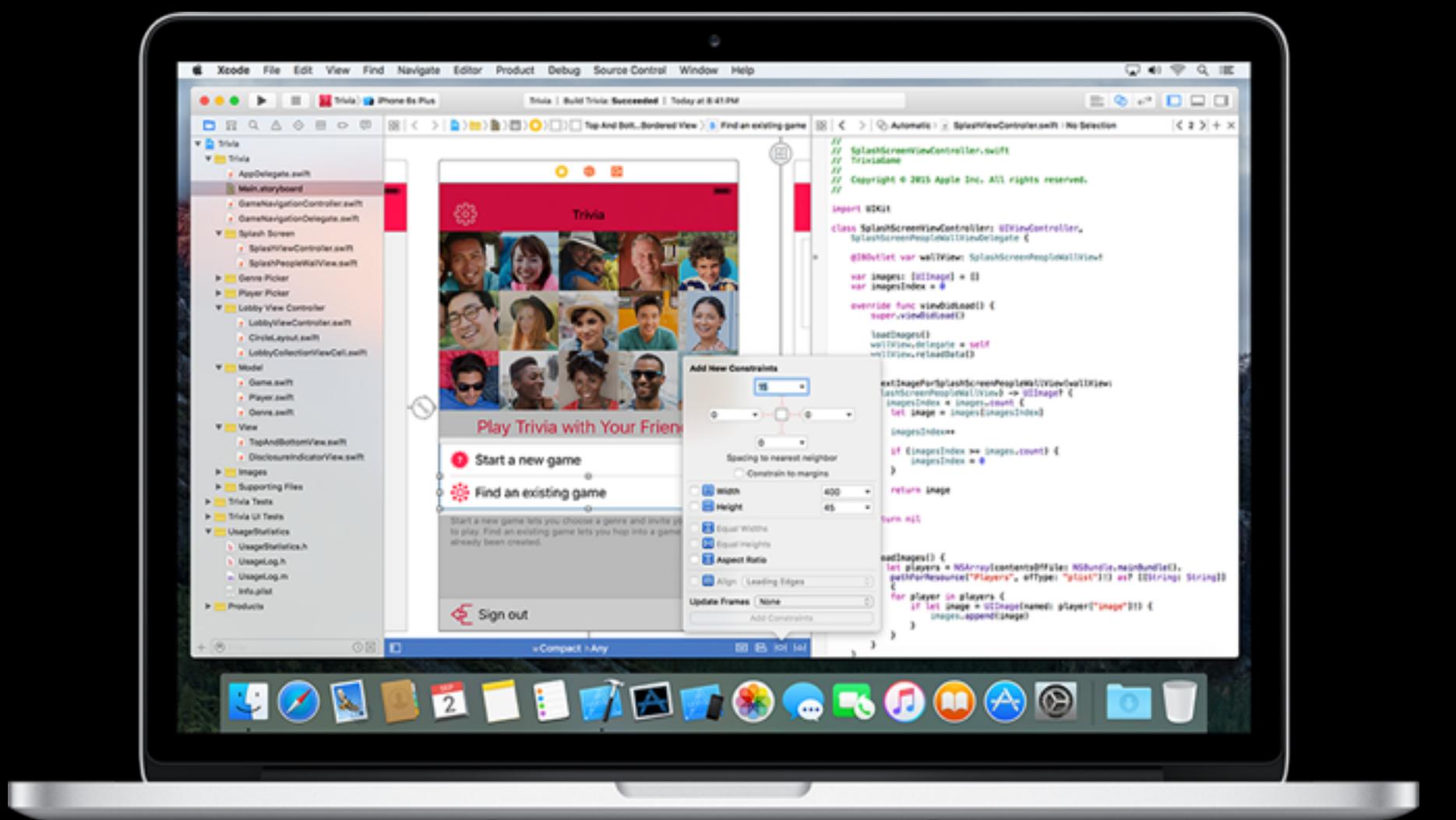
Swift 2

watchOS 2

tvOS 9



Xcode 7



Swift 2

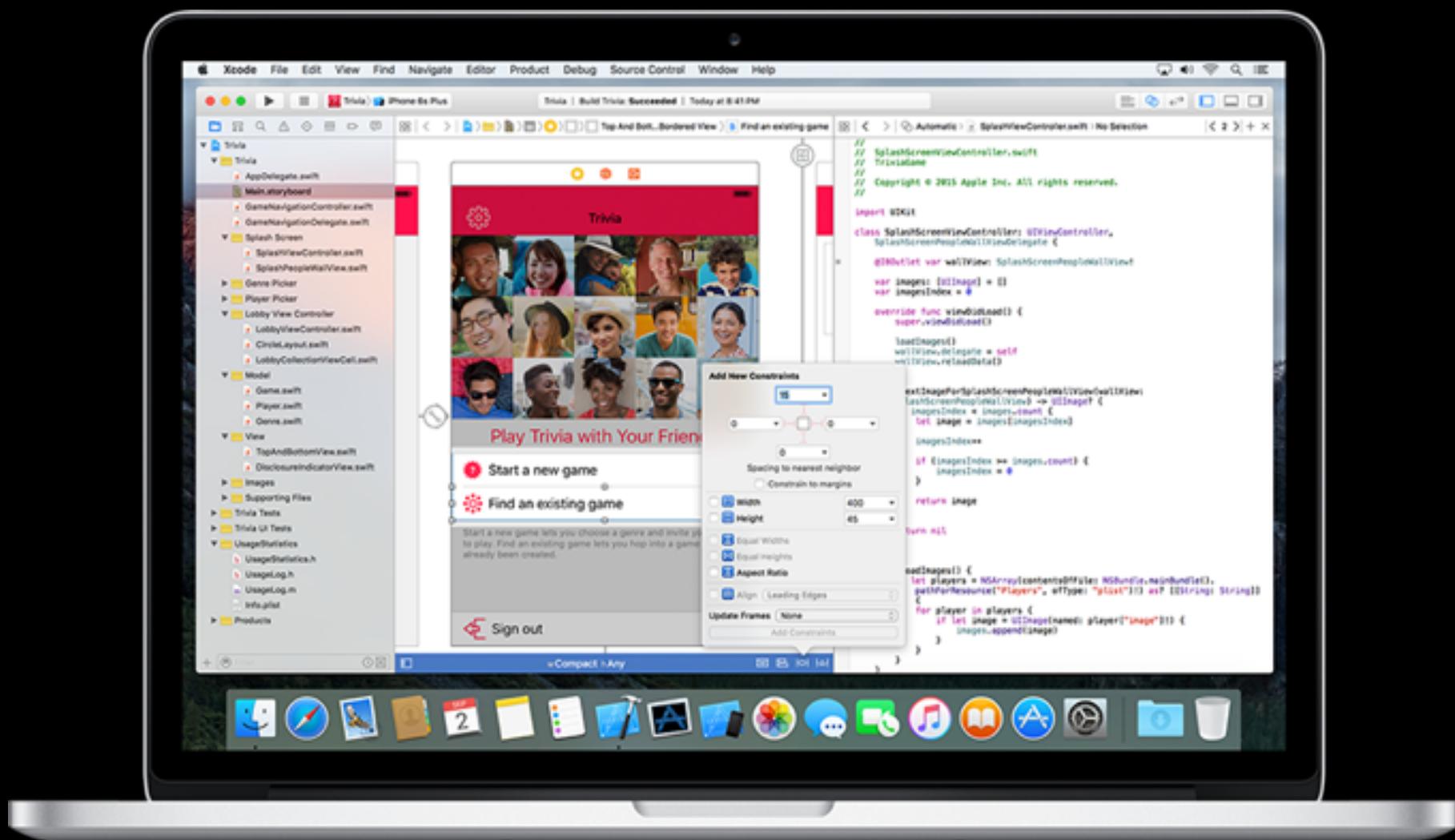
watchOS 2

tvOS 9

OS X



Xcode 7



Swift 2

watchOS 2

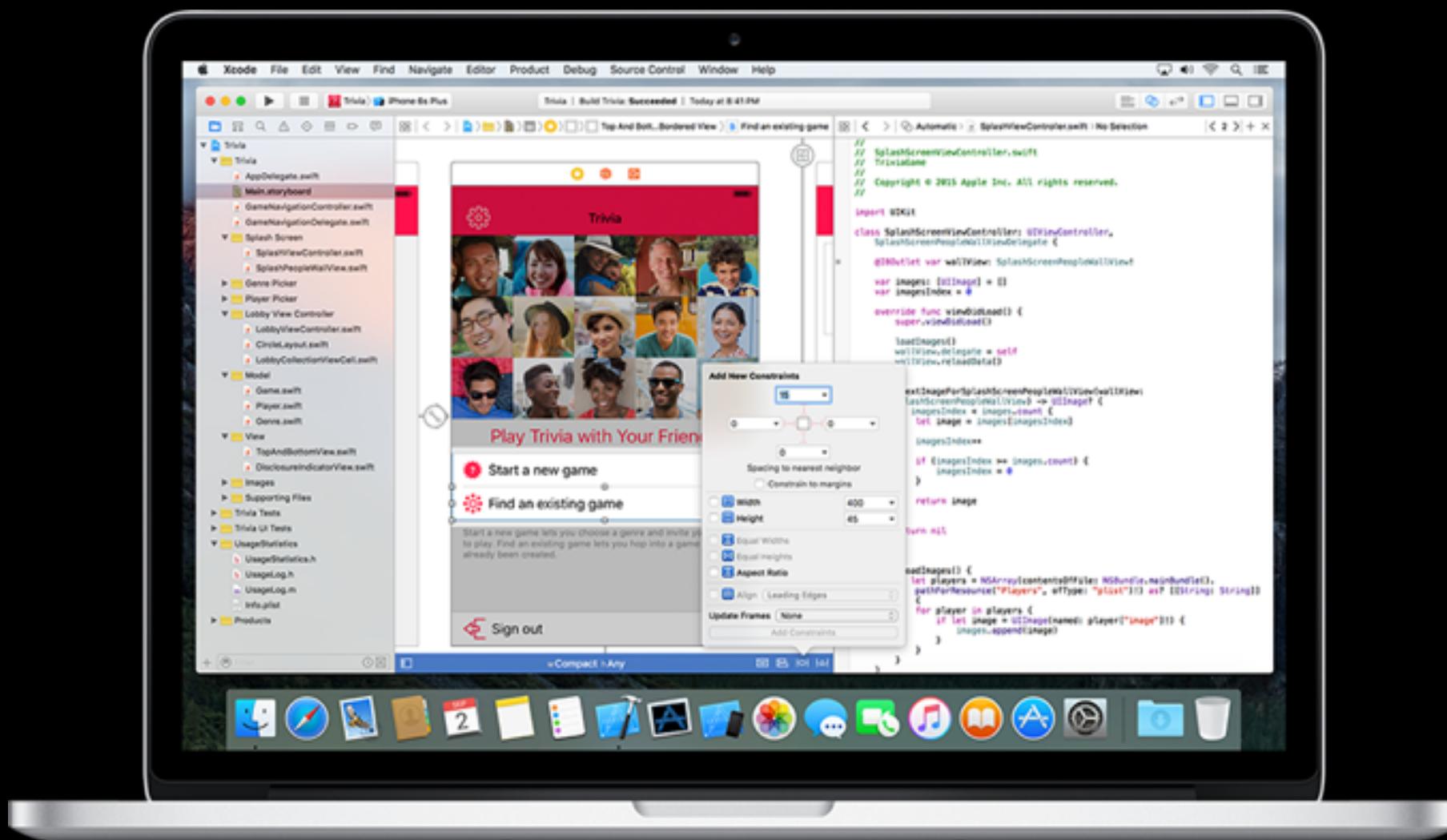
tvOS 9

OS X

Live design



Xcode 7



Swift 2
watchOS 2
tvOS 9
OS X
Live design
Visual debugging



Xcode 7



Swift 2

watchOS 2

tvOS 9

OS X

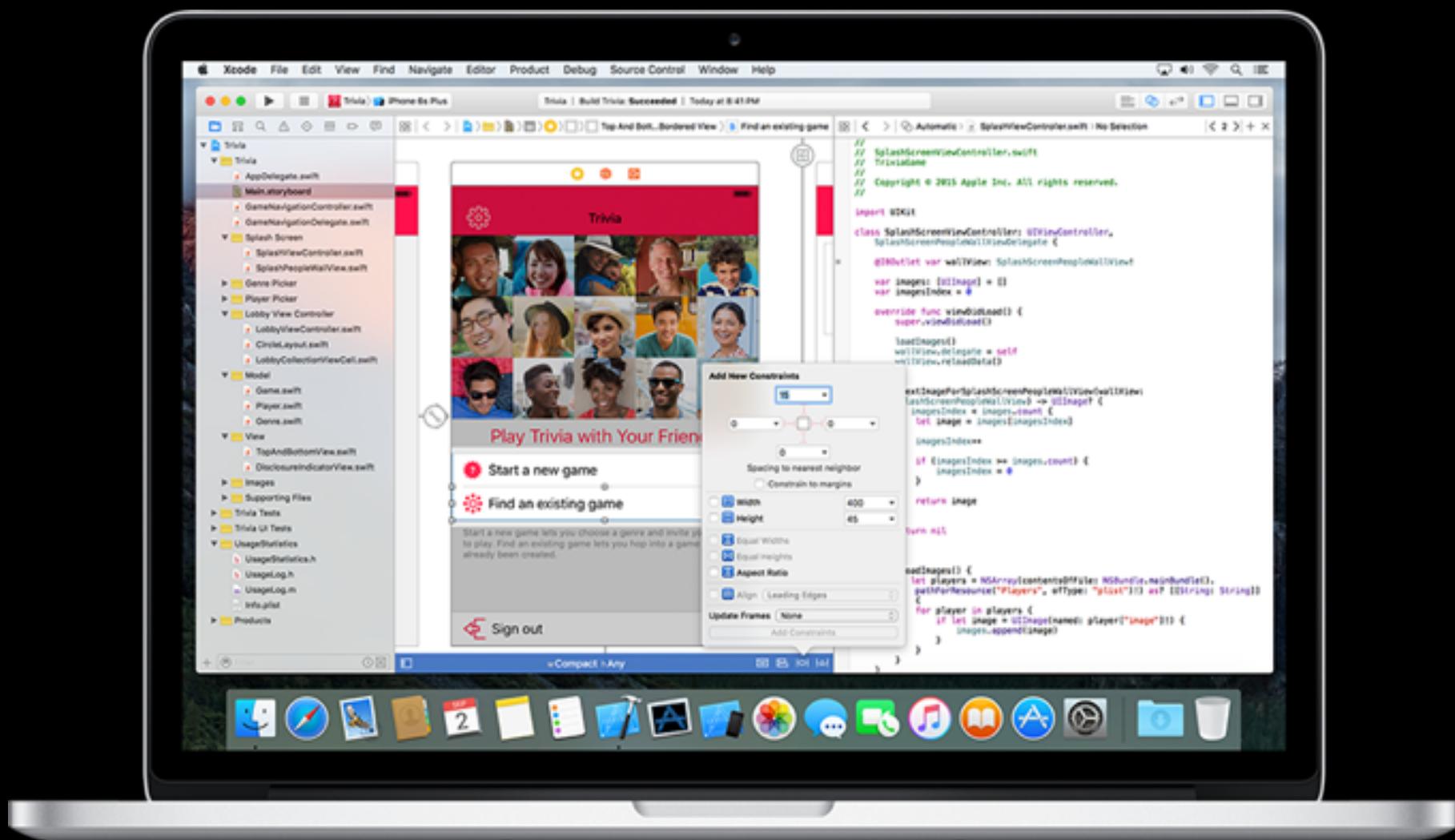
Live design

Visual debugging

Performance testing



Xcode 7



Swift 2

watchOS 2

tvOS 9

OS X

Live design

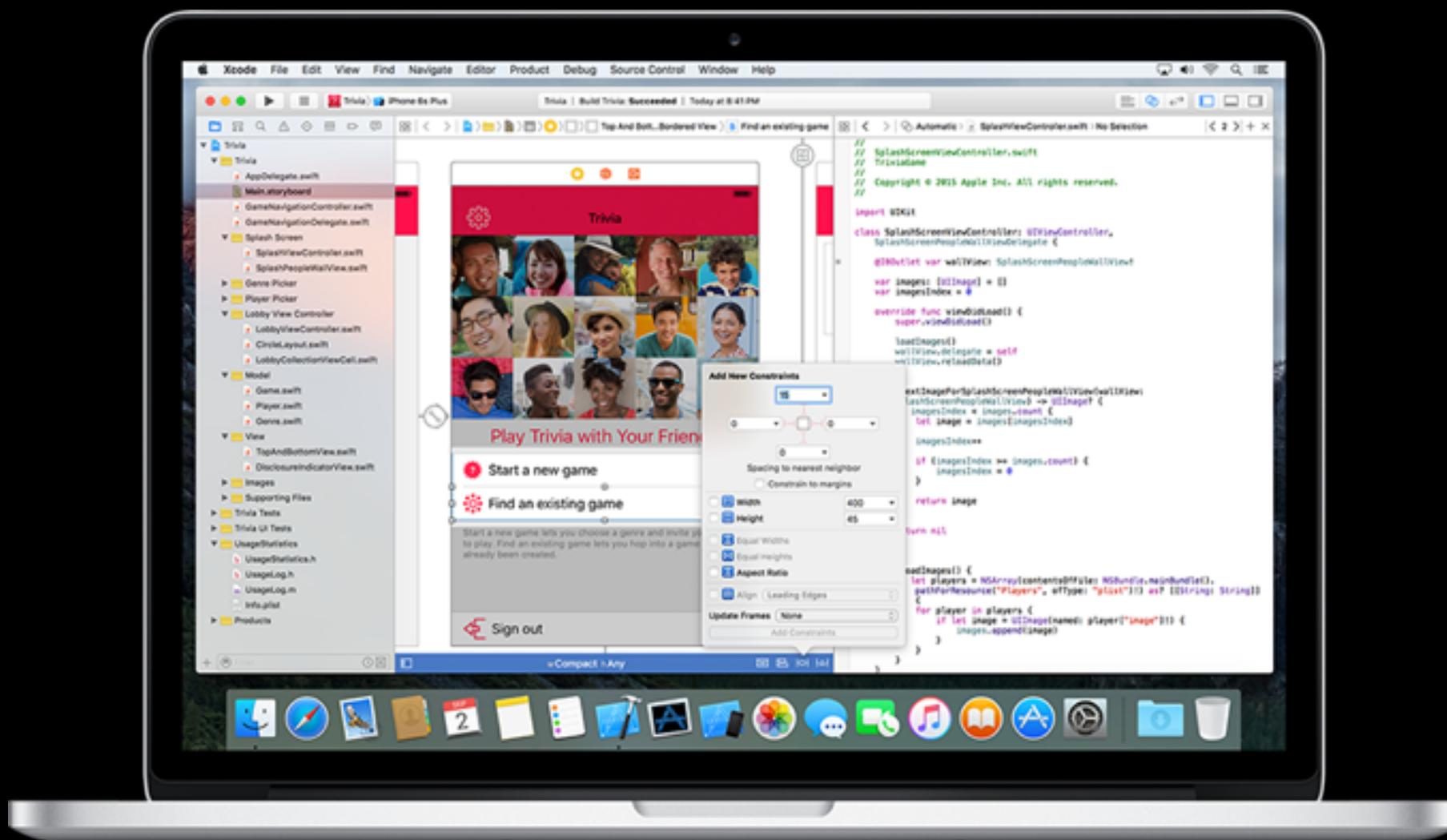
Visual debugging

Performance testing

UI Testing



Xcode 7



Swift 2

watchOS 2

tvOS 9

OS X

Live design

Visual debugging

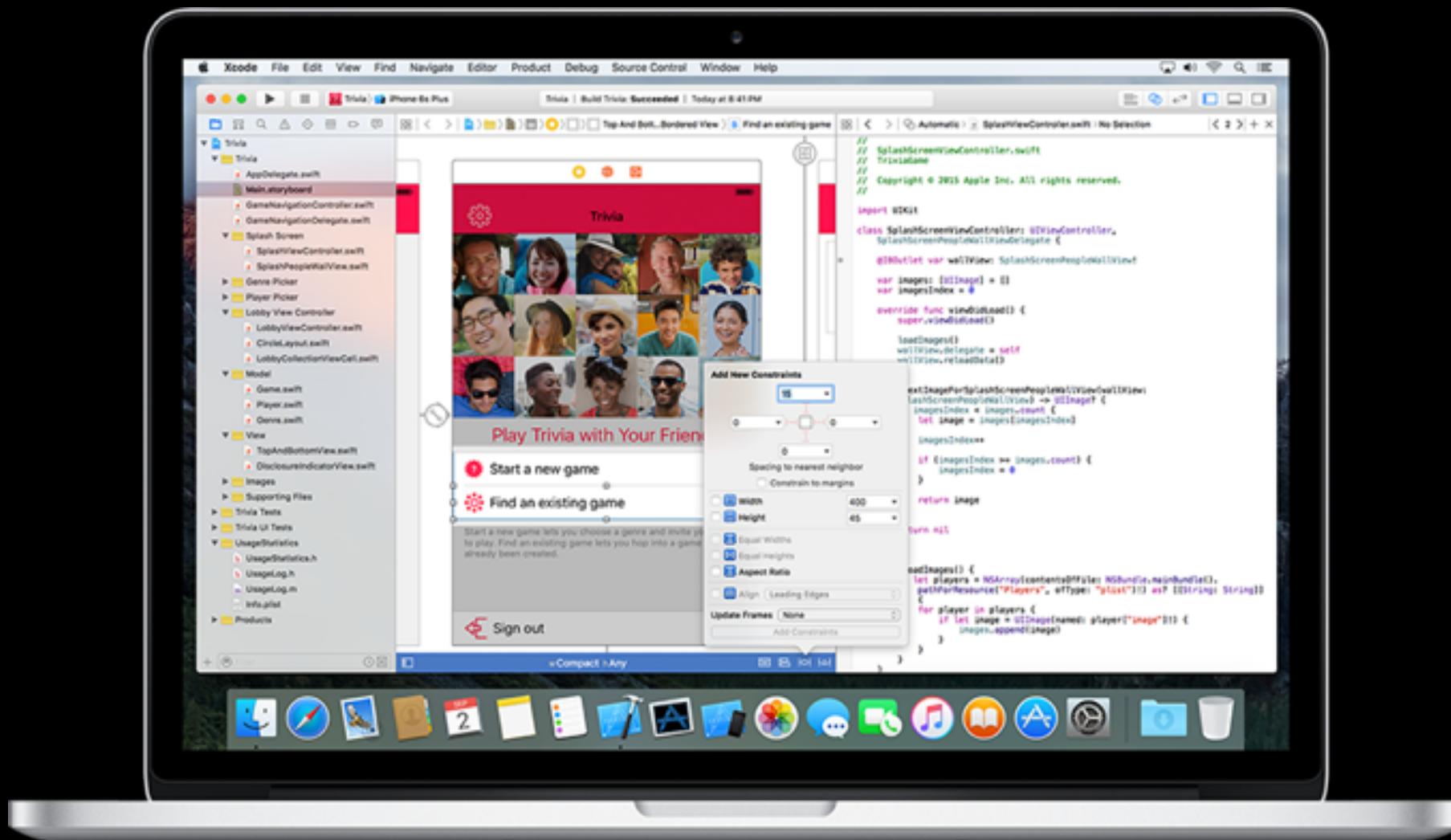
Performance testing

UI Testing

Code Coverage



Xcode 7



Swift 2

watchOS 2

tvOS 9

OS X

Live design

Visual debugging

Performance testing

UI Testing

Code Coverage

Address Sanitizer



iOS 9



iOS 9



3D Touch



iOS 9



3D Touch Slide Over



iOS 9



3D Touch
Slide Over
Split View



iOS 9



3D Touch
Slide Over
Split View
Picture in Picture



iOS 9



3D Touch
Slide Over
Split View
Picture in Picture
App Thinning



iOS 9



3D Touch
Slide Over
Split View
Picture in Picture
App Thinning
Safari View Controller



iOS 9



3D Touch
Slide Over
Split View
Picture in Picture
App Thinning
Safari View Controller
Search API



watchOS 2



watchOS 2

Digital Crown



watchOS 2

Digital Crown
Audio



watchOS 2



Digital Crown
Audio
Animation APIs



watchOS 2



Digital Crown

Audio

Animation APIs

Taptic Engine



watchOS 2



Digital Crown

Audio

Animation APIs

Taptic Engine
Core Motion



watchOS 2



Digital Crown

Audio

Animation APIs

Taptic Engine

Core Motion

ClockKit



watchOS 2



Digital Crown

Audio

Animation APIs

Taptic Engine

Core Motion

ClockKit

Time Travel



watchOS 2



Digital Crown
Audio
Animation APIs
Taptic Engine
Core Motion
ClockKit
Time Travel
WatchConnectivity



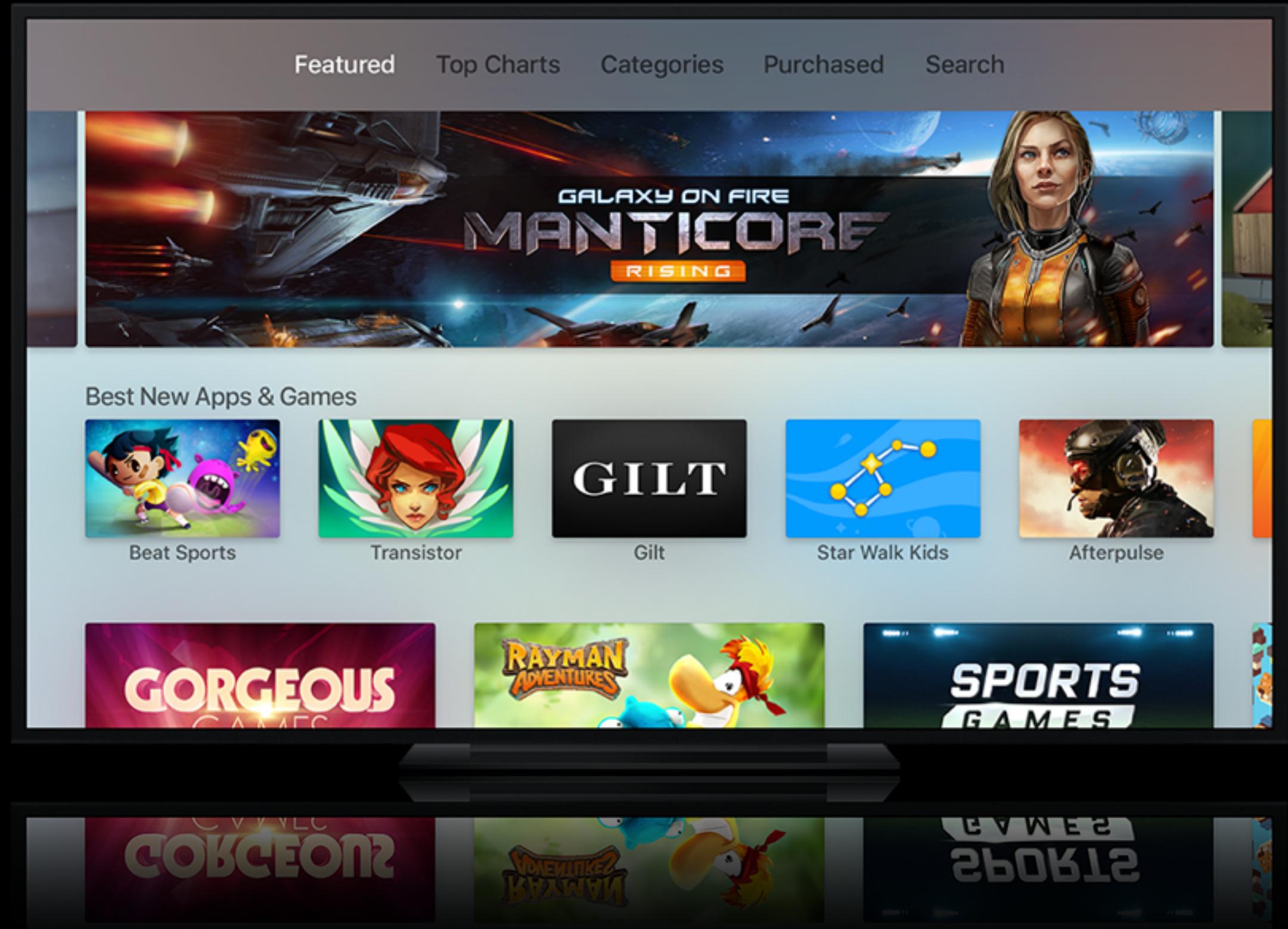
watchOS 2



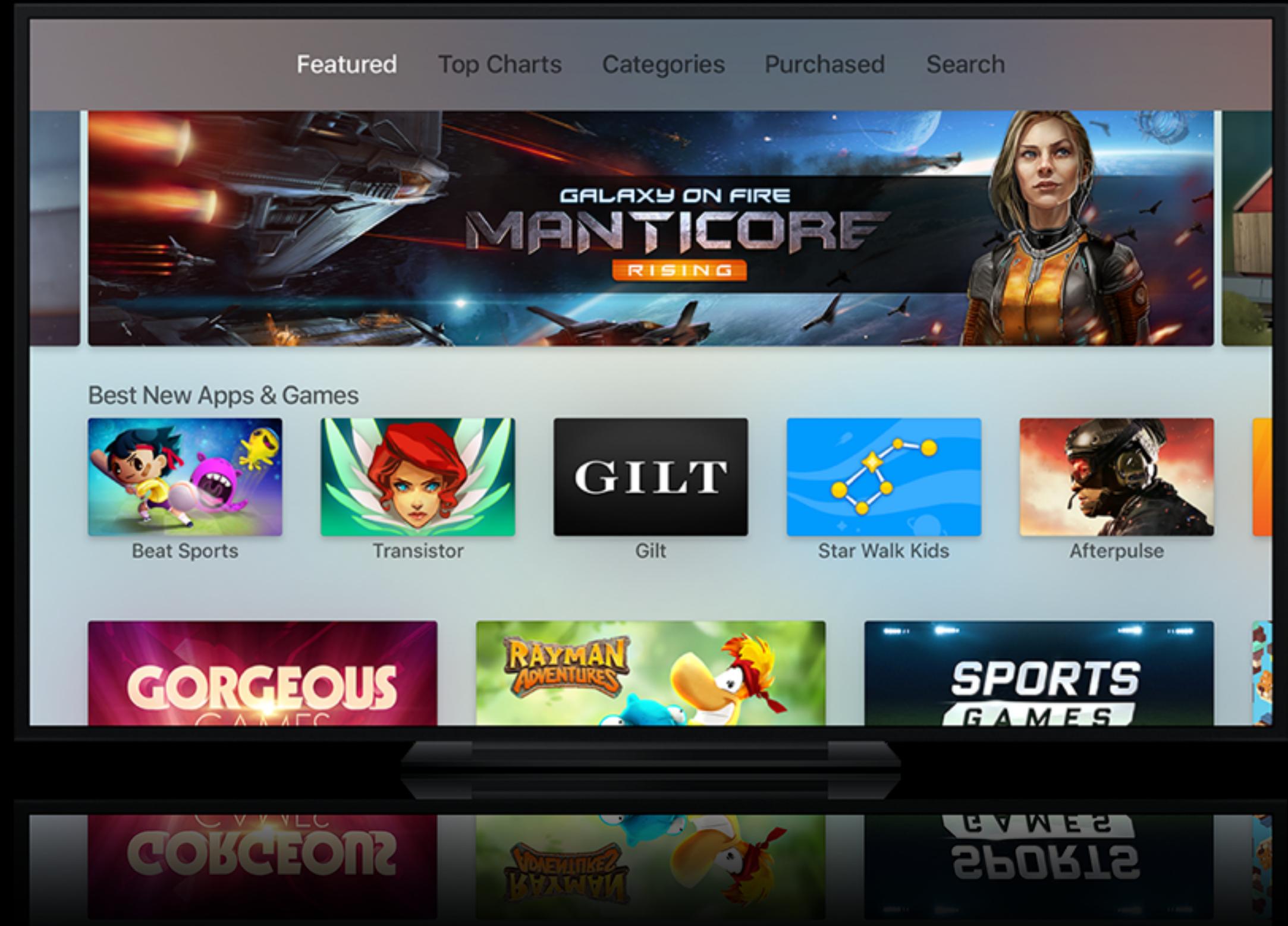
Digital Crown
Audio
Animation APIs
Taptic Engine
Core Motion
ClockKit
Time Travel
WatchConnectivity
NSURLSession



tvOS



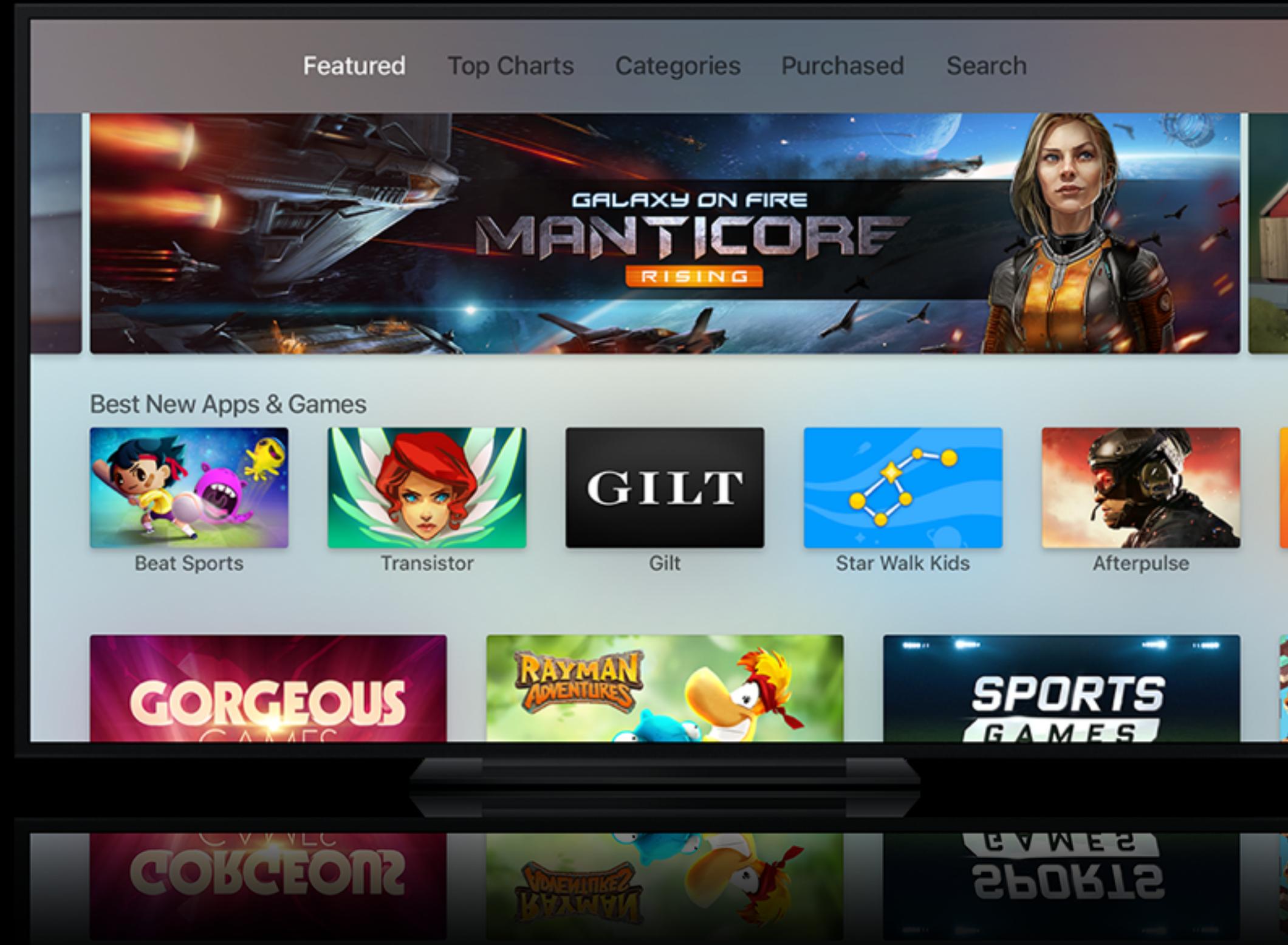
tvOS



Most Frameworks



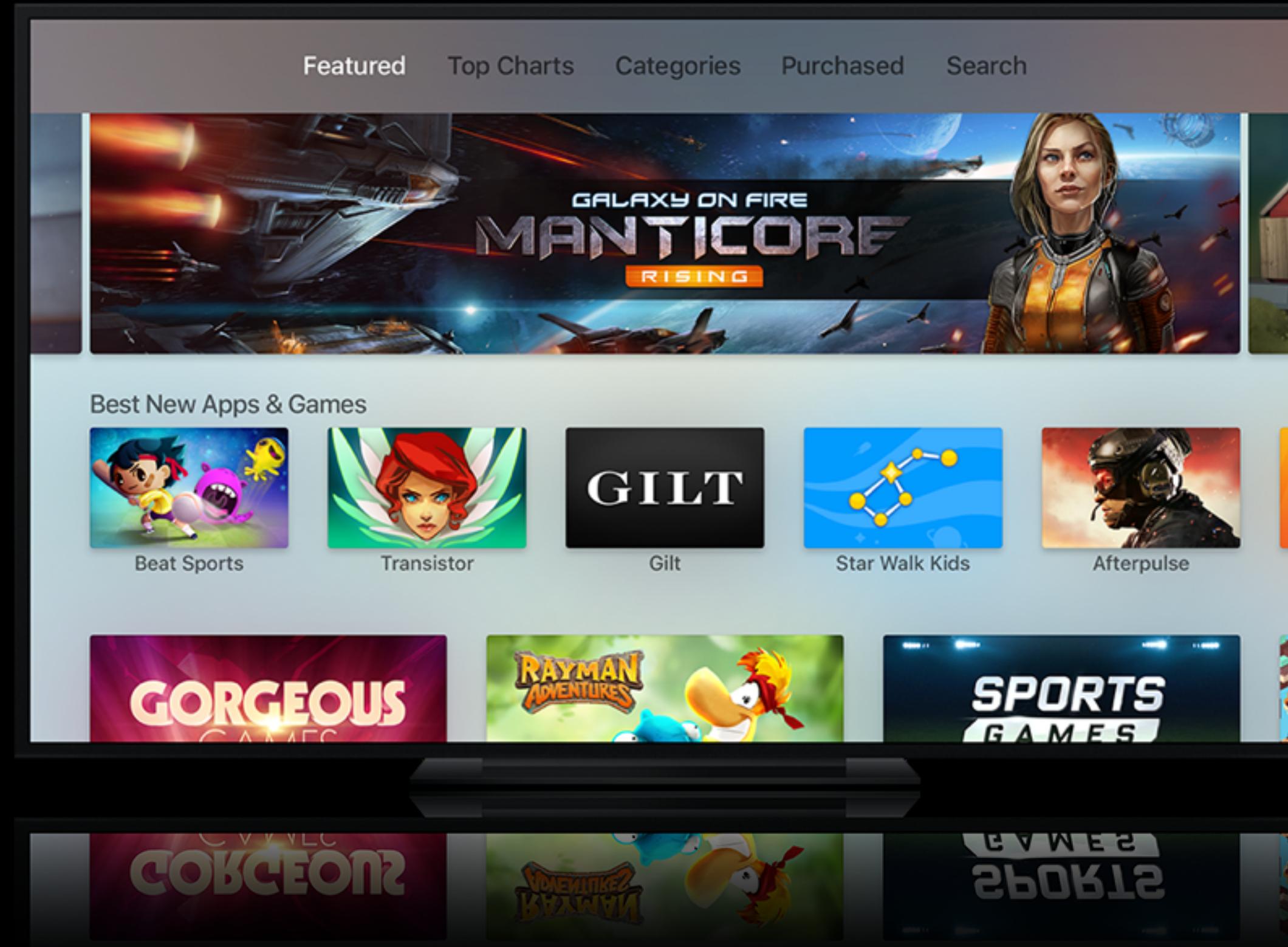
tvOS



Most Frameworks
Javascript + TVML



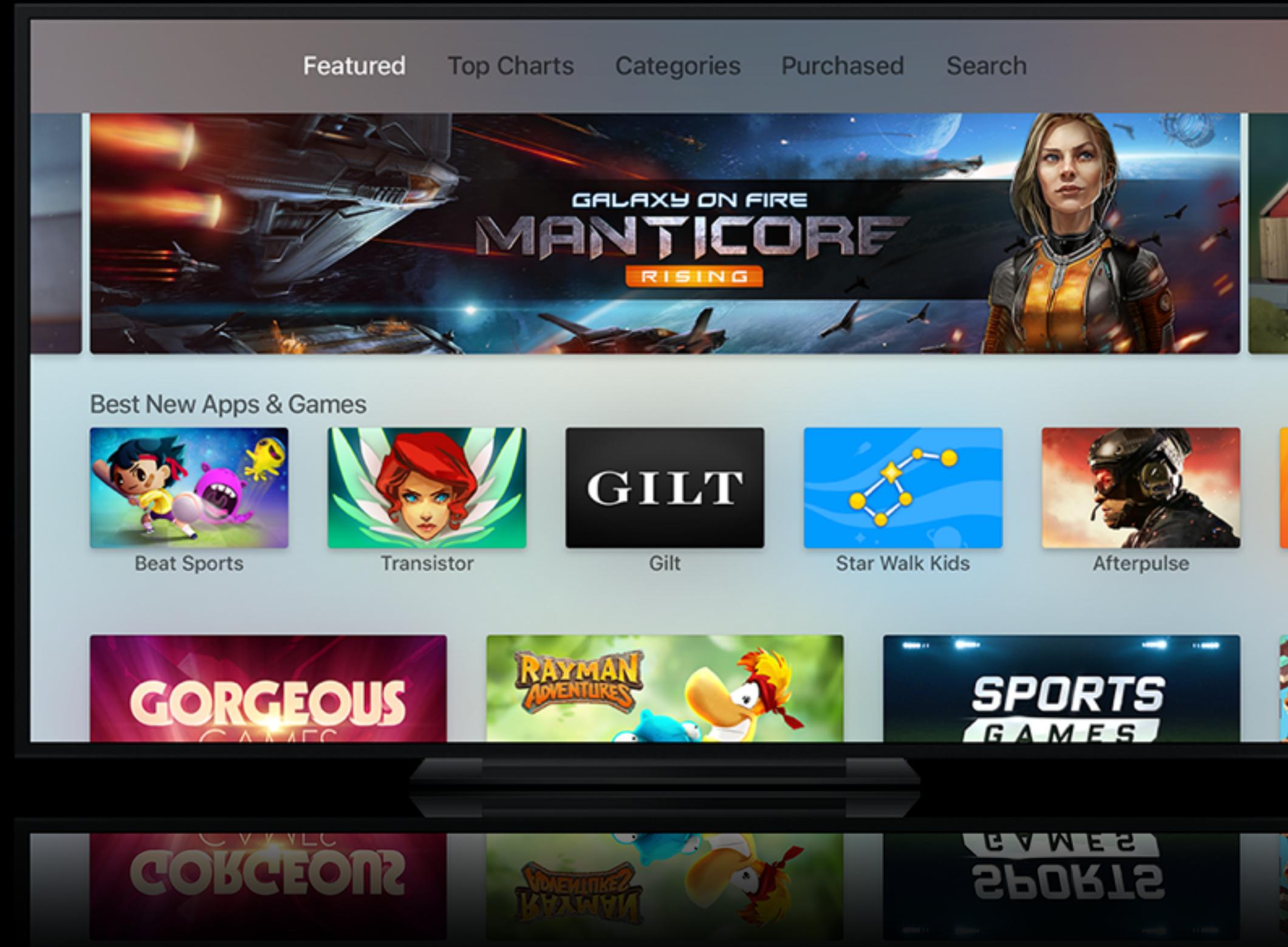
tvOS



Most Frameworks
Javascript + TVML
Remote



tvOS



Most Frameworks

Javascript + TVML

Remote
Siri



Demo
Xcode 7

Swift 2



Swift 2



Swift 2

SAFE

MODERN



Swift 2

SAFE

MODERN

POWER



```
#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

```
let languageName: String = "Swift"
```



Constants and Variables

```
let languageName: String = "Swift"  
var version: Double = 1.0
```



Constants and Variables

```
let languageName: String = "Swift"  
var version: Double = 1.0  
var introducedIn: Int = 2014
```



Constants and Variables

```
let languageName: String = "Swift"  
var version: Double = 1.0  
var introducedIn: Int = 2014  
var isAwesome: Bool = true
```



Constants and Variables

```
let languageName: String = "Swift"  
var version: Double = 1.0  
let introducedIn: Int = 2014  
let isAwesome: Bool = true
```



Type Inference

```
let languageName: String = "Swift"  
var version: Double = 1.0  
let introducedIn: Int = 2014  
let isAwesome: Bool = true
```



Type Inference

```
let languageName = "Swift"  
var version = 1.0  
let introducedIn = 2014  
let isAwesome = true
```



Unicode Names

```
let languageName = "Swift"  
var version = 1.0  
let introducedIn = 2014  
let isAwesome = true
```



Unicode Names

```
let languageName = "Swift"  
var version = 1.0  
let introducedIn = 2014  
let isAwesome = true  
let π = 3.14159265
```



Unicode Names

```
let languageName = "Swift"  
var version = 1.0  
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

```
let a = 3
```

```
let b = 5
```

```
// "3 times 5 is 15"
```



Building Complex Strings

```
let a = 3
```

```
let b = 5
```

```
// "3 times 5 is 15"
```

```
let result = "\u{a} times \u{b} is \u{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)
```



Typed Collections

SAFE



Typed Collections

```
var names = ["Anna", "Brian", "Jack"]  
// an array of String values
```



Typed Collections

```
var names = ["Anna", "Brian", "Jack"]  
// an array of String values
```

```
var numberOfLegs = {"ant": 6, "snake": 0}  
// a Dictionary with String keys and Int values
```



Loops



Loops

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



Loops

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

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



Loops

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

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

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



POWER

For-In: Strings and Characters



For-In: Strings and Characters

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



For-In: Strings and Characters

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



POWER

For-In: Ranges



For-In: Ranges

```
for number in 1...5 {  
    print("\$(number) times 4 is \$(number*4)")  
}
```



For-In: Ranges

```
for number in 1...5 {  
    print("\$(number) times 4 is \$(number*4)")  
}
```



For-In: Ranges

```
for number in 1..<5 {  
    print("\$number) times 4 is \$(\$number*4)")  
}
```



For-In: Ranges

```
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
```



POWER

For-In: Arrays



For-In: Arrays

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



For-In: Arrays

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

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



POWER

For-In: Dictionaries



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

```
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")
```

```
print(greeting)
```



Returning Values

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

```
let greeting = sayHello("WWDC")
```

```
print(greeting)
```

```
Hello, WWDC
```



Returning Multiple Values

MODERN



Returning Multiple Values

```
func refreshWebPage() -> (Int, String) {  
    //...try to refresh...  
  
    return (200, "Success")  
}
```



Functions in General



Functions in General

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



Demo

Swift in Playground



KCL TECH SOCIETY