

socket io swift

Bitbucket Devslopes - Dropbox Android Developers SurePay Apple Twitter

socket io swift - Google Search Untitled

Google socket io swift

All News Videos Shopping Images More Settings Tools

About 1,260,000 results (0.47 seconds)

GitHub - socketio/socket.io-client-swift
<https://github.com/socketio/socket.io-client-swift> ▾
Contribute to socket.io-client-swift development by creating an account on GitHub .

Socket.IO — Socket.IO on iOS
<https://socket.io/blog/socket-io-on-ios/> ▾
Mar 9, 2015 - We are pleased to announce the immediate availability of the Socket.IO Swift Client!
You'll now be able to write code that runs natively on iOS ...

Getting started with Socket.io in Swift on iOS - Twilio
<https://www.twilio.com/blog/2016/.../getting-started-with-socket-io-in-swift-on-ios.html> ▾
Sep 30, 2016 - Let's demonstrate how to work with the Swift Socket.io client library by building an application to monitor the status of Twilio phone calls.

Swift Tutorial: Building an iOS Chat App Using Socket.IO - AppCoda
<https://www.appcoda.com/socket-io-chat-app/> ▾
Feb 22, 2016 - The first thing we have to do is to download the Socket.IO Swift Client library and add it to the project. The link I just gave you will take you to a ...

CocoaDocs.org - Socket.IO-Client-Swift
cocoadocs.org/docsets/Socket.IO-Client-Swift/1.3.1 ▾
or Swift. Supports ws/wss/polling connections and binary. For socket.io 1.0+ and 1.2 use the 1.2 branch.

Socket.IO-Client-Swift - CocoaDocs
cocoadocs.org/docsets/Socket.IO-Client-Swift/2.2.1 ▾
Socket.IO-Client-Swift. Socket.IO-client for iOS/OS X. Example. let socket = SocketIOClient(socketURL: "http://localhost:8080") socket.on("connect") { ... }

1x 02:01 / 26:29 +

for messages using socket.io and Swift 3 - Stack Overflow

The screenshot shows a video player interface with a GitHub page for "socketio/socket.io-client-swift" open in the background. The GitHub page title is "Socket.IO-Client-Swift" and it describes the project as "Socket.IO-client for iOS/OS X." Below the title is a section titled "Example" containing Swift code. The video player has a video frame showing a developer speaking, with the text "Objective-C Example" overlaid. The video controls at the bottom include a play button, volume, and settings icons.

Socket.IO-Client-Swift

Socket.IO-client for iOS/OS X.

Example

```
import SocketIO

let socket = SocketIOClient(socketURL: URL(string: "http://localhost:8080")!, config: [.log(true), .compress])

socket.on(clientEvent: .connect) {data, ack in
    print("socket connected")
}

socket.on("currentAmount") {data, ack in
    if let cur = data[0] as? Double {
        socket.emitWithAck("canUpdate", cur).timingOut(after: 0) {data in
            socket.emit("update", ["amount": cur + 2.50])
        }
    }

    ack.with("Got your currentAmount", "dude")
}

socket.connect()
```

Objective-C Example

```
@import SocketIO;
NSURL *url = [[NSURL alloc] initWithString:@"http://localhost:8080"];
SocketIOClient *socket = [[SocketIOClient alloc] initWithSocketURL:url config:@{@"log": @YES, @"compress": @YES}];
```

96. Sockets and Channels

The screenshot shows a Xcode interface with a file creation dialog open. The left sidebar displays a project structure for a project named "Smack". The "Services" folder contains several Swift files: AuthService.swift, UserDataService.swift, and MessageService.swift. The "View" folder contains GradientView.swift, RoundedButton.swift, AvatarCell.swift, CircleImage.swift, and ChannelCell.swift. The "Controller" folder contains ChatVC.swift, ChannelVC.swift, LoginVC.swift, CreateAccountVC.swift, AvatarPickerVC.swift, ProfileVC.swift, and AddChannelVC.swift. Supporting Files include SWRevealViewController.h, SWRevealViewController.m, and Smack-Bridging-Header.h.

The main area shows a "Choose options for your new file:" dialog. The "Class:" field is set to "SocketSel". The "Subclass of:" dropdown is set to "NSObject". The "Language:" dropdown is set to "Swift". There is also an unchecked checkbox for "Also create XIB file".

Below the dialog, a portion of a Swift file is visible:

```
frame.size.width - 60  
USER_DATA_DID_CHANGE, object:  
  
let profile = ProfileVC()  
profile.modalPresentationStyle = .custom  
present(profile, animated: true, completion: nil)  
} else {  
    let channelVC = ChannelVC()  
    channelVC.modalPresentationStyle = .custom  
    present(channelVC, animated: true, completion: nil)
```

The bottom of the screen shows the Xcode toolbar with various icons for navigation and search.

96. Sockets and Channels



Smack | Build Smack: **Succeeded** | Yesterday at 11:11 PM

Smack | Services | SocketService.swift | socket

33

```
1 //  
2 // SocketService.swift  
3 // Smack  
4 //  
5 // Created by Jonny B on 7/20/17.  
6 // Copyright © 2017 Jonny B. All rights reserved.  
7 //  
8  
9 import UIKit  
10 import SocketIO  
11  
12 class SocketService: NSObject {  
13  
14     static let instance = SocketService()  
15  
16     override init() {  
17         super.init()  
18     }  
19  
20     var socket : SocketIOClient = SocketIOClient(socketURL: URL(string: BASE_URL)!)  
21  
22 }  
23
```

1x 05:13 / 26:29

CC

96. Sockets and Channels

The screenshot shows the Xcode IDE interface with the following details:

- Project Navigator:** Shows the project structure with "Smack" selected.
- Build Issues:** A sidebar titled "Buildtime (34) Runtime" lists issues across several frameworks:
 - Alamofire:** 5 issues (Swift Compiler Warning)
 - SwiftyJSON:** 1 issue (Swift Compiler Warning)
 - Starscream:** 1 issue (Swift Compiler Warning)
 - Smack:** 27 issues (Semantic Issue, Class 'SWContextTransitionObject' does not conform to protocol 'UIViewControllerContextTransitioning', Add stubs for missing protocol requirements, Possible misuse of comma operator here)
- Code Editor:** The main window displays the file `SocketService.swift` with the following content:

```
3 // Smack
4 //
5 // Created by Jonny B on 7/20/17.
6 // Copyright © 2017 Jonny B. All rights reserved.
7 //

8

9 import UIKit
10 import SocketIO

11 class SocketService: NSObject {

12     static let instance = SocketService()

13     let manager: SocketManager
14     let socket: SocketIOClient
15
16     override init() {
17         self.manager = SocketManager(socketURL: URL(string: BASE_URL)!)
18         self.socket = manager.defaultSocket
19         super.init()
20     }
21
22
23 }

24
25
26
27
28
29
30
31
32
33
34
35}
```

A purple annotation "new version code in init() method" is placed next to the line `self.socket = manager.defaultSocket`.
- Identity and Type:** A panel on the right shows the file's properties:
 - Name: `SocketService.swift`
 - Type: Default - Swift Source
 - Location: Relative to Group
 - Full Path: /Users/devslopes/Desktop/smack/Smack/Services/SocketService.swift
- Text Settings:** Includes options for Text Encoding (No Explicit Encoding), Line Endings (No Explicit Line Endings), Indent Using (Spaces), and Widths (Tab width 4, Indent width 4). The "Wrap lines" checkbox is checked.
- Search Results:** A "No Matches" message is displayed at the bottom right.

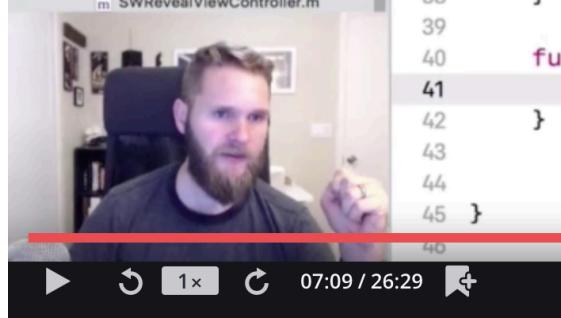
96. Sockets and Channels

The screenshot shows the Xcode interface with the project 'Smack' open. The file 'SocketService.swift' is selected in the left sidebar. The code editor displays the following Swift code:

```
1 // SocketService.swift
2 // Smack
3 // Created by Jonny B on 7/20/17.
4 // Copyright © 2017 Jonny B. All rights reserved.
5
6 import UIKit
7 import SocketIO
8
9 class SocketService: NSObject {
10
11     static let instance = SocketService()
12
13     override init() {
14         super.init()
15     }
16
17     var socket : SocketIOClient = SocketIOClient(socketURL: URL(string: BASE_URL)!)
18
19     func establishConnection() {
20         socket.connect()
21     }
22
23     func closeConnection() {
24         socket.disconnect()
25     }
26
27 }
28
29
30 }
31
```

The line 'socket.connect()' is highlighted with a light blue background. The status bar at the bottom shows the time as 06:13 / 26:29.

96. Sockets and Channels



The screenshot shows the Xcode IDE with the project "Smack" open. The file `AppDelegate.swift` is currently selected and displayed in the main editor area. The code in the editor is as follows:

```
Smack | Build Smack: Succeeded | Yesterday at 11:11 PM 33
Smack > Smack > AppDelegate.swift > applicationWillTerminate(_:) < >
or when the user quits the application and it begins the
transition to the background state.
24 // Use this method to pause ongoing tasks, disable timers, and
// invalidate graphics rendering callbacks. Games should use this
// method to pause the game.
25 }
26
27 func applicationDidEnterBackground(_ application: UIApplication) {
28     // Use this method to release shared resources, save user data,
29     // invalidate timers, and store enough application state
30     // information to restore your application to its current state in
31     // case it is terminated later.
32     // If your application supports background execution, this method
33     // is called instead of applicationWillTerminate: when the user
34     // quits.
35
36     func applicationWillEnterForeground(_ application: UIApplication) {
37         // Called as part of the transition from the background to the
38         // active state; here you can undo many of the changes made on
39         // entering the background.
40
41     func applicationWillTerminate(_ application: UIApplication) {
42         SocketService.instance.closeConnection()
43     }
44 }
45 }
```

The code highlights several methods related to application state transitions: `applicationDidEnterBackground`, `applicationWillEnterForeground`, and `applicationWillTerminate`. The `applicationWillTerminate` method specifically calls `SocketService.instance.closeConnection()`.

The Xcode interface includes a sidebar with project files, a top bar showing the build status, and a bottom bar with various icons.

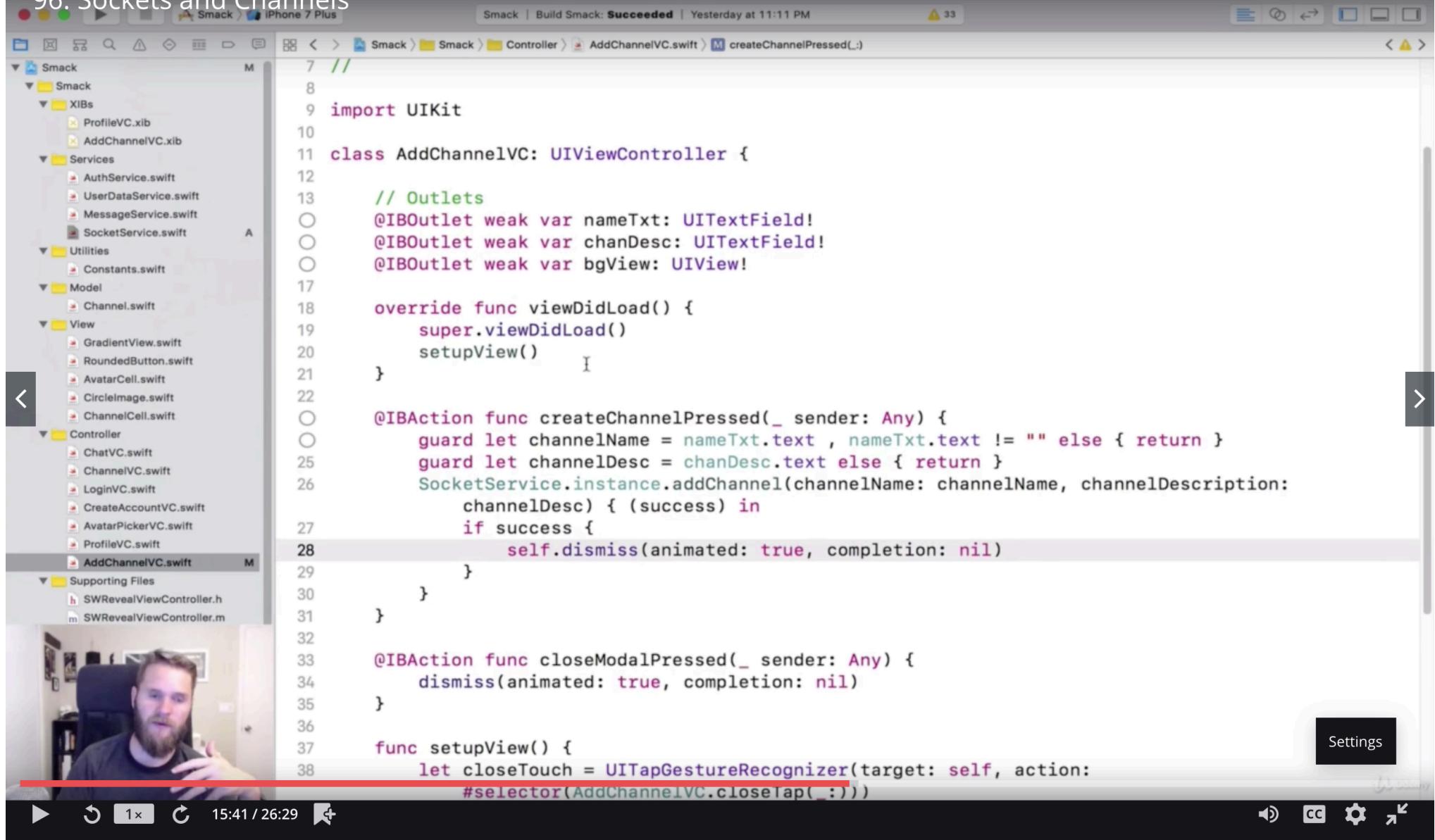
96. Sockets and Channels

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure under the "Smack" project. The "SocketService.swift" file is selected in the Services group.
- Editor:** Displays the code for the `SocketService` class. The code uses SocketIOClient to handle socket connections and emit events like "newChannel".
- Build Bar:** Shows "Smack" | Build Smack: **Succeeded** | Yesterday at 11:11 PM
- Document Outline:** Shows the file tree with icons for files and folders.
- Bottom Bar:** Includes standard Xcode controls like play/pause, refresh, zoom, and a progress bar indicating the video is at 13:33 / 26:29.
- Bottom Right:** A "Settings" button.

```
11
12 class SocketService: NSObject {
13
14     static let instance = SocketService()
15
16     override init() {
17         super.init()
18     }
19
20     var socket : SocketIOClient = SocketIOClient(socketURL: URL(string: BASE_URL)!)
21
22     func establishConnection() {
23         socket.connect()
24     }
25
26     func closeConnection() {
27         socket.disconnect()
28     }
29
30     func addChannel(channelName: String, channelDescription: String, completion: @escaping
31                     CompletionHandler) {
32         socket.emit("newChannel", channelName, channelDescription)
33         completion(true)
34     }
35
36
37
38
39
40
41
42
```

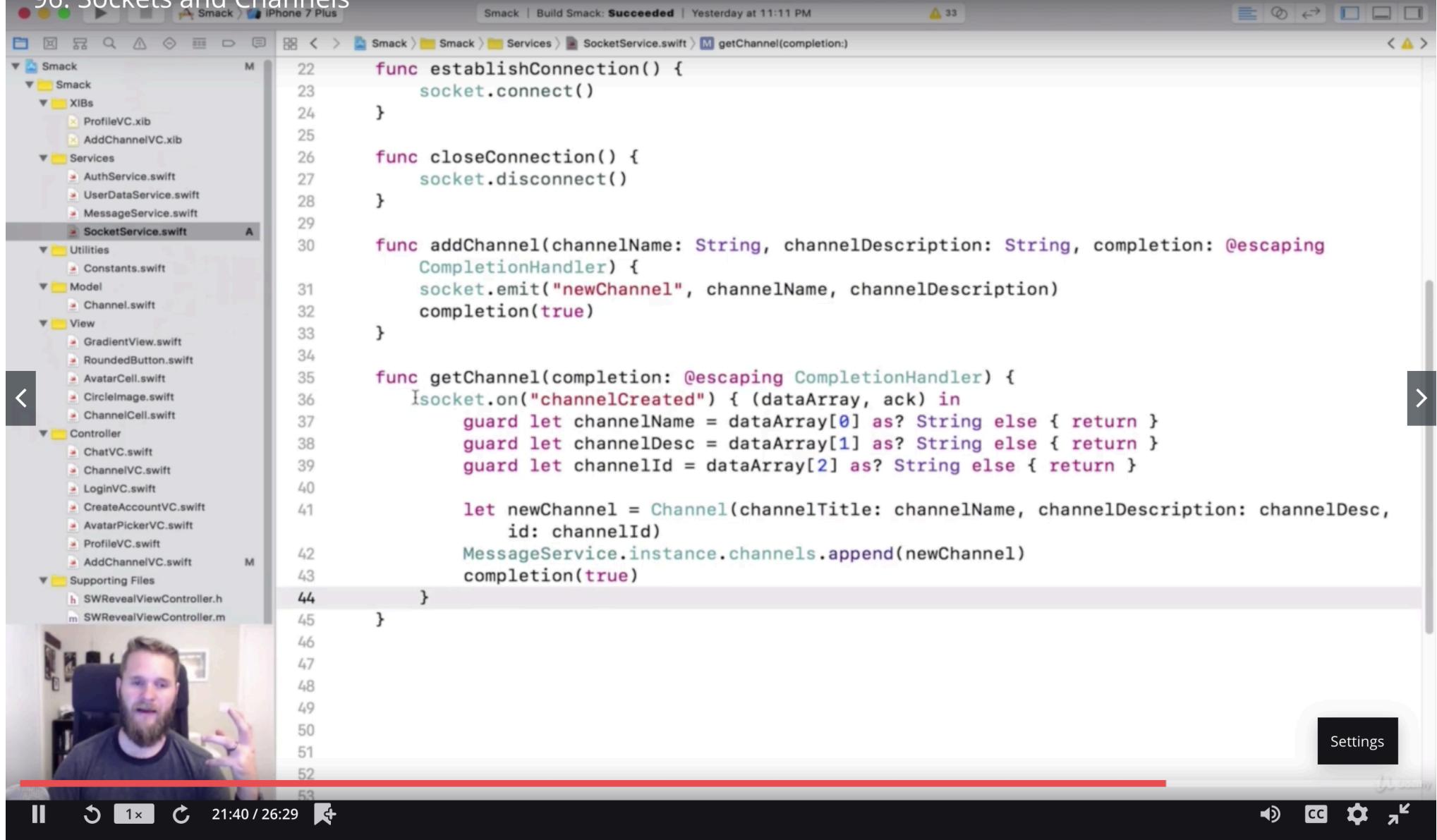
96. Sockets and Channels



The screenshot shows the Xcode interface with the project 'Smack' open. The file 'AddChannelVC.swift' is selected in the project navigator. The code editor displays the implementation of the `AddChannelVC` class, which handles channel creation and closure logic.

```
7 //|
8
9 import UIKit
10
11 class AddChannelVC: UIViewController {
12
13     // Outlets
14     @IBOutlet weak var nameTxt: UITextField!
15     @IBOutlet weak var chanDesc: UITextField!
16     @IBOutlet weak var bgView: UIView!
17
18     override func viewDidLoad() {
19         super.viewDidLoad()
20         setupView()
21     }
22
23     @IBAction func createChannelPressed(_ sender: Any) {
24         guard let channelName = nameTxt.text, nameTxt.text != "" else { return }
25         guard let channelDesc = chanDesc.text else { return }
26         SocketService.instance.addChannel(channelName: channelName, channelDescription:
27             channelDesc) { (success) in
28             if success {
29                 self.dismiss(animated: true, completion: nil)
30             }
31         }
32
33     @IBAction func closeModalPressed(_ sender: Any) {
34         dismiss(animated: true, completion: nil)
35     }
36
37     func setupView() {
38         let closeTouch = UITapGestureRecognizer(target: self, action:
#selector(AddChannelVC.closeTap(_:)))
```

96. Sockets and Channels

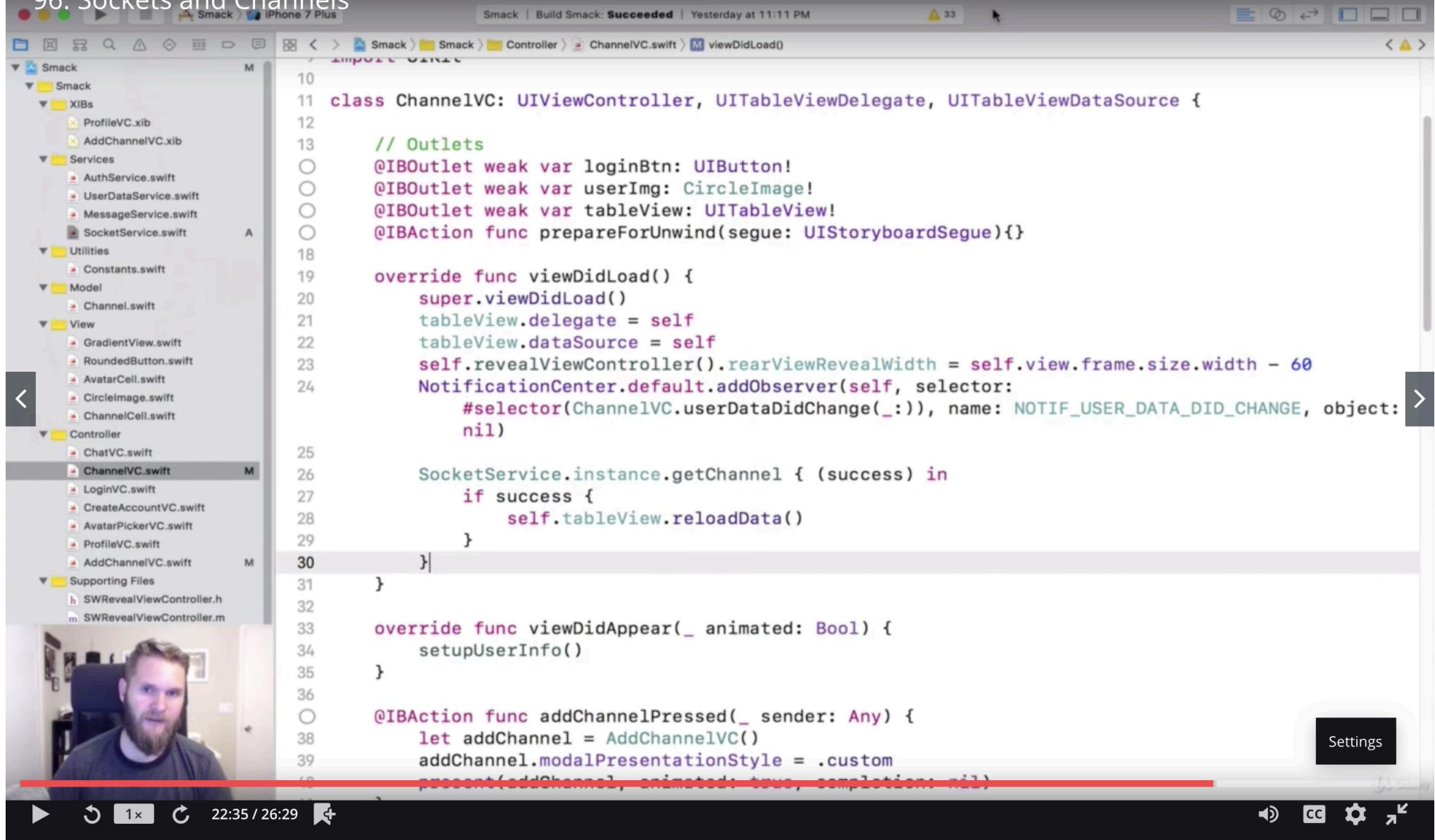


The screenshot shows the Xcode interface with the project "Smack" open. The file "SocketService.swift" is selected in the left sidebar, which displays the following Swift code:

```
22 func establishConnection() {
23     socket.connect()
24 }
25
26 func closeConnection() {
27     socket.disconnect()
28 }
29
30 func addChannel(channelName: String, channelDescription: String, completion: @escaping CompletionHandler) {
31     socket.emit("newChannel", channelName, channelDescription)
32     completion(true)
33 }
34
35 func getChannel(completion: @escaping CompletionHandler) {
36     socket.on("channelCreated") { (dataArray, ack) in
37         guard let channelName = dataArray[0] as? String else { return }
38         guard let channelDesc = dataArray[1] as? String else { return }
39         guard let channelId = dataArray[2] as? String else { return }
40
41         let newChannel = Channel(channelTitle: channelName, channelDescription: channelDesc,
42             id: channelId)
43         MessageService.instance.channels.append(newChannel)
44         completion(true)
45     }
46 }
47
48
49
50
51
52 }
```

The Xcode status bar at the bottom indicates the video is at 21:40 / 26:29. A camera icon with a red circle over it is visible, indicating the video is being recorded.

96. Sockets and Channels



The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure for "Smack". The "Controller" group contains "ChannelVC.swift".
- Editor:** Displays the code for "ChannelVC.swift".
- Code:** The code is written in Swift and handles socket operations and UI interactions.
- Annotations:** A red horizontal bar highlights the line starting with "let addChannel = AddChannelVC()".
- Bottom Bar:** Includes standard Xcode navigation icons (play, stop, zoom) and a timestamp "22:35 / 26:29".
- Top Bar:** Shows the project name "Smack", build status "Build Smack: Succeeded", and the date "Yesterday at 11:11 PM".
- Right Side:** Includes a "Settings" button and a video feed of a person with a beard.

```
10
11 class ChannelVC: UIViewController, UITableViewDelegate, UITableViewDataSource {
12
13     // Outlets
14     @IBOutlet weak var loginBtn: UIButton!
15     @IBOutlet weak var userImg: CircleImage!
16     @IBOutlet weak var tableView: UITableView!
17     @IBAction func prepareForUnwind(segue: UIStoryboardSegue){}
18
19     override func viewDidLoad() {
20         super.viewDidLoad()
21         tableView.delegate = self
22         tableView.dataSource = self
23         self.revealViewController().rearViewRevealWidth = self.view.frame.size.width - 60
24         NotificationCenter.default.addObserver(self, selector:
25             #selector(ChannelVC.userDataDidChange(_:)), name: NOTIF_USER_DATA_DID_CHANGE, object:
26             nil)
27
28         SocketService.instance.getChannel { (success) in
29             if success {
30                 self.tableView.reloadData()
31             }
32         }
33
34         override func viewDidAppear(_ animated: Bool) {
35             setupUserInfo()
36         }
37
38         @IBAction func addChannelPressed(_ sender: Any) {
39             let addChannel = AddChannelVC()
40             addChannel.modalPresentationStyle = .custom
41             present(addChannel, animated: true, completion: nil)
42 }
```

96. Sockets and Channels

