

iOS Phone Video

Example of iOS application for video calls

On the screenshot below the example is displayed before a call will be established.

The interface of the application is the same as in the example [Phone](#), except that two videos are played

- left - video from the camera of this user
- right - video from the other call party

No SIM

16:53

42 %

wss://wcs5-eu.flashphoner.com:8443

Sip Login

1000

Sip Auth Name

1000

Sip Password

1234

Sip Domain

192.168.0.1

Sip Outbound Proxy

192.168.0.1

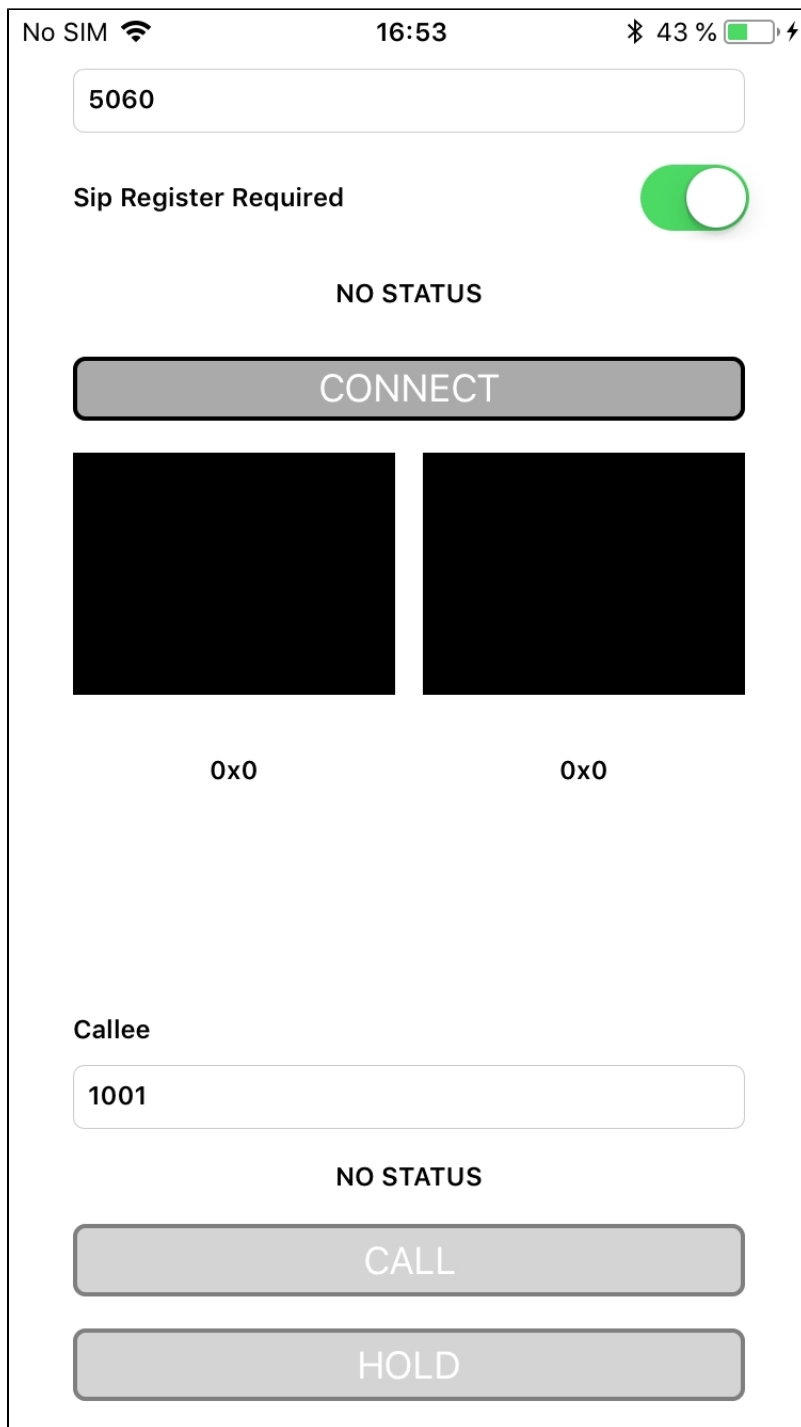
Sip Port

5060

Sip Register Required

☒

NO STATUS



Analyzing the code of the examle

To analyze the code, let's takePhoneMinVideo example, which can be downloaded with corresponding build[2.5.2](#).

View class for the main view of the application: ViewController (header file[ViewController.h](#); implementation file[ViewController.m](#)).

1. Import of API.[code](#)

```
#import <FPWCSApi2/FPWCSApi2.h>
```

2. Connection to the server.

FPWCSApi2 createSession, FPWCSApi2Session connect[code](#)

FPWCSApi2SessionOptions object with the following parameters is passed to createSession() method

- URL of WCS server
- SIP parameters to make outgoing call and to receive incoming calls
- appKey of internal server-side application(defaultApp)

```
FPWCSApi2SessionOptions *options = [[FPWCSApi2SessionOptions alloc] init];
options.urlServer = _connectUrl.text;
options.sipRegisterRequired = _sipRegRequired.control.isOn;
options.sipLogin = _sipLogin.input.text;
options.sipAuthenticationName = _sipAuthName.input.text;
options.sipPassword = _sipPassword.input.text;
options.sipDomain = _sipDomain.input.text;
options.sipOutboundProxy = _sipOutboundProxy.input.text;
options.sipPort = [NSNumber numberWithInt: [_sipPort.input.text integerValue]];
options.appKey = @"defaultApp";
NSError *error;
...
session = [FPWCSApi2 createSession:options error:&error];
...
[session connect];
```

3. Outgoing call.

FPWCSApi2Session createCall, FPWCSApi2Call call[code](#)

The next parameters are passed to createCall() method:

- callee SIP username
- view to display local video stream preview
- view to display remote video stream from callee
- additional SIP INVITE parameters from string set by user

```
- (FPWCSApi2Call *)call {
    FPWCSApi2Session *session = [FPWCSApi2 getSessions][0];
    FPWCSApi2CallOptions *options = [[FPWCSApi2CallOptions alloc] init];
    options.callee = _callee.input.text;
    options.localDisplay = _videoView.local;
    options.remoteDisplay = _videoView.remote;
    options.localConstraints = [[FPWCSApi2MediaConstraints alloc] initWithAudio:YES video:YES];
    options.remoteConstraints = [[FPWCSApi2MediaConstraints alloc] initWithAudio:YES video:YES];
    NSError *error;
    call = [session createCall:options error:&error];
    ...
    [call call];
    return call;
}
```

4.Receiving the event on incoming call

FPWCSApi2Session onIncomingCallCallback[code](#)

```

[session onIncomingCallCallback:^(FPWCSApi2Call *rCall) {
    call = rCall;

    [call on:kFPWCSCallStatusBusy callback:^(FPWCSApi2Call *call){
        [self changeCallStatus:call];
        [self toCallState];
    }];

    [call on:kFPWCSCallStatusFailed callback:^(FPWCSApi2Call *call){
        [self changeCallStatus:call];
        [self toCallState];
    }];

    [call on:kFPWCSCallStatusRing callback:^(FPWCSApi2Call *call){
        [self changeCallStatus:call];
        [self toHangupState];
    }];

    [call on:kFPWCSCallStatusHold callback:^(FPWCSApi2Call *call){
        [self changeCallStatus:call];
        [self changeViewState:_holdButton enabled:YES];
    }];

    [call on:kFPWCSCallStatusEstablished callback:^(FPWCSApi2Call *call){
        [self changeCallStatus:call];
        [self toHangupState];
        [self changeViewState:_holdButton enabled:YES];
    }];

    [call on:kFPWCSCallStatusFinish callback:^(FPWCSApi2Call *call){
        [self changeCallStatus:call];
        [self toCallState];
        [self dismissViewControllerAnimated:YES completion:nil];
    }];
    ...
}];

```

5. Answering incoming call.

FPWCSApi2Call answer [code](#)

```

alert = [UIAlertController
        alertControllerWithTitle:[NSString stringWithFormat:@"Incoming call from '%@'",
[rCall getCallee]]
        message:error.localizedDescription
        preferredStyle:UIAlertControllerStyleAlert];

UIAlertAction* answerButton = [UIAlertAction
        actionWithTitle:@"Answer"
        style:UIAlertActionStyleDefault
        handler:^(UIAlertAction * action) {
            [call getLocalConstraints].video = [[FPWCSApi2VideoConstraints alloc] init];
            [call setLocalDisplay:_videoView.local];
            [call setRemoteDisplay:_videoView.remote];
            [call answer];
        }];

[alert addAction:answerButton];
UIAlertAction* hangupButton = [UIAlertAction
        actionWithTitle:@"Hangup"
        style:UIAlertActionStyleDefault
        handler:^(UIAlertAction * action) {
            [call hangup];
        }];

[alert addAction:hangupButton];
[self presentViewController:alert animated:YES completion:nil];

```

6. Call hold and retrieve.

FPWCSApi2Call hold, unhold[code](#)

```

- (void)holdButton:(UIButton *)button {
    [self changeViewState:button enabled:NO];
    if ([button.titleLabel.text isEqualToString:@"UNHOLD"]) {
        if (call) {
            [call unhold];
            [_holdButton setTitle:@"HOLD" forState:UIControlStateNormal];
        }
    } else {
        if (call) {
            [call hold];
            [_holdButton setTitle:@"UNHOLD" forState:UIControlStateNormal];
        }
    }
}

```

7. Outgoing call hangup.

FPWCSApi2Call hangup[code](#)

```

- (void)callButton:(UIButton *)button {
    [self changeViewState:button enabled:NO];
    if ([button.titleLabel.text isEqualToString:@"HANGUP"]) {
        if ([FPWCSApi2 getSessions].count) {
            [call hangup];
        } else {
            [self toCallState];
        }
        ...
    }
}

```

8.Incoming call hangup.

FPWCSEApi2Call hangup[code](#)

```
UIAlertAction* hangupButton = [UIAlertAction
                               initWithTitle:@"Hangup"
                               style:UIAlertActionStyleDefault
                               handler:^(UIAlertAction * action) {
                                   [call hangup];
                               }];

[alert addAction:hangupButton];
```

9. Disconnection.

FPWCSEApi2Session disconnect[code](#)

```
- (void)connectButton:(UIButton *)button {
    [self changeViewState:button enabled:NO];
    if ([button.titleLabel.text isEqualToString:@"DISCONNECT"]) {
        if ([FPWCSEApi2 getSessions].count) {
            FPWCSEApi2Session *session = [FPWCSEApi2 getSessions][0];
            NSLog(@"Disconnect session with server %@", [session getServerUrl]);
            [session disconnect];
        } else {
            NSLog(@"Nothing to disconnect");
            [self onDisconnected];
        }
        ...
    }
}
```