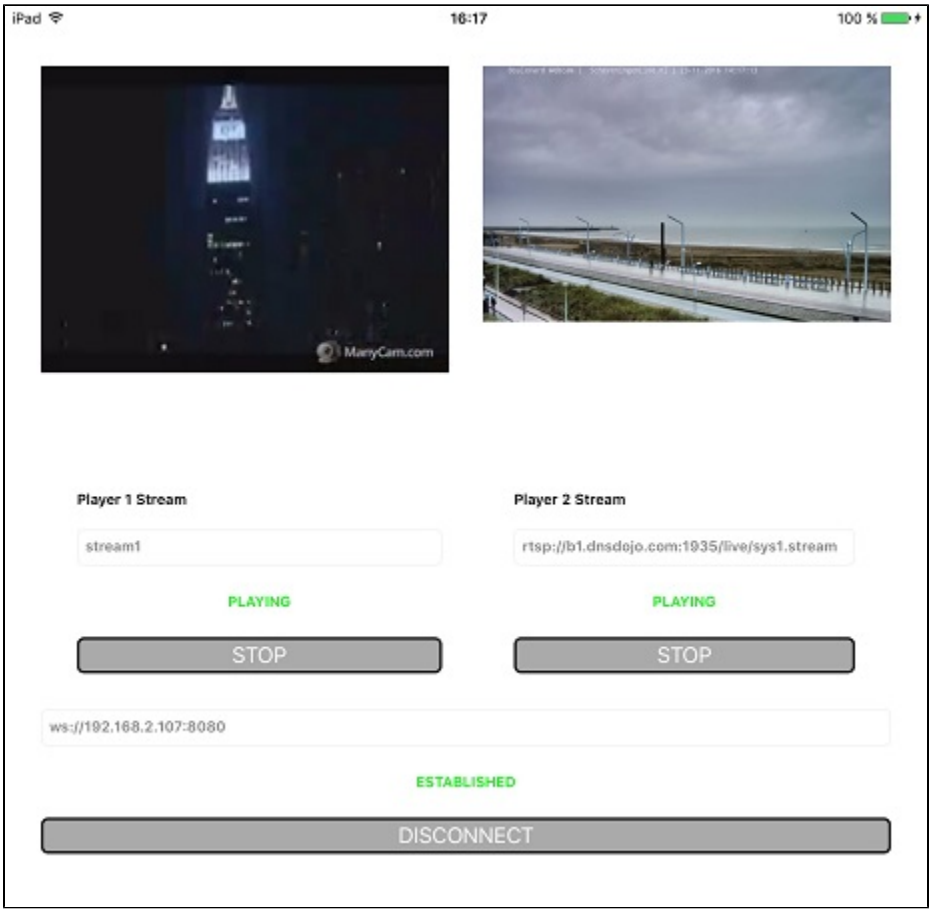
iOS 2 Players

Example of iOS application with two players

This example demonstrates how two or more players can be displayed in one application. Each of the players can be used to play a different stream.



Work with code of the example

To analyze the code, let's take TwoPlayers example version with hash 088a74d, which can be downloaded with 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.Session creation

FPWCSApi2 createSessioncode

The options include:

- URL of WCS server
- appKey of internal server-side application (defaultApp)

```
FPWCSApi2SessionOptions *options = [[FPWCSApi2SessionOptions alloc] init];
options.urlServer = _connectUrl.text;
options.appKey = @"defaultApp";
NSError *error;
FPWCSApi2Session *session = [FPWCSApi2 createSession:options error:&error];
```

3.Connection to the server

FPWCSApi2Session connectcode

```
[session connect];
```

4. Receiving the event confirming successful connection.

ViewController onConnectedcode

```
- (void)onConnected:(FPWCSApi2Session *)session {
    [_connectButton setTitle:@"DISCONNECT" forState:UIControlStateNormal];
    [self changeViewState:_connectButton enabled:YES];
    [self onStopped1];
    [self onStopped2];
}
```

5. Playback of video stream 1.

FPWCSApi2Session createStream, FPWCSApi2Stream playcode

Object with next stream options is passed to createStream method:

- stream name
- view to display video

```
- (FPWCSApi2Stream *)play1Stream {
    FPWCSApi2Session *session = [FPWCSApi2 getSessions][0];
    FPWCSApi2StreamOptions *options = [[FPWCSApi2StreamOptions alloc] init];
    options.name = _player1StreamName.text;
    options.display = _player1Display;
    NSError *error;
    player1Stream = [session createStream:options error:nil];
    ...
    if(![player1Stream play:&error]) {
        UIAlertController * alert = [UIAlertController
                                     alertControllerWithTitle:@"Failed to play"
                                     message:error.localizedDescription
                                     preferredStyle:UIAlertControllerStyleAlert];

        UIAlertAction* okButton = [UIAlertAction
                                   actionWithTitle:@"Ok"
                                   style:UIAlertActionStyleDefault
                                   handler:^(UIAlertAction * action) {

                                   }];

        [alert addAction:okButton];
        [self presentViewController:alert animated:YES completion:nil];
    }
    return player1Stream;
}
```

6. Playback of video stream 2.

FPWCSApi2Session createStream, FPWCSApi2Stream playcode

Object with next stream options is passed to createStream method:

- stream name
- view to display video

```
- (FPWCSApi2Stream *)play2Stream {
    FPWCSApi2Session *session = [FPWCSApi2 getSessions][0];
    FPWCSApi2StreamOptions *options = [[FPWCSApi2StreamOptions alloc] init];
    options.name = _player2StreamName.text;
    options.display = _player2Display;
    NSError *error;
    player2Stream = [session createStream:options error:nil];
    ...
    if(![player2Stream play:&error]) {
        UIAlertController * alert = [UIAlertController
                                     alertControllerWithTitle:@"Failed to play"
                                     message:error.localizedDescription
                                     preferredStyle:UIAlertControllerStyleAlert];

        UIAlertAction* okButton = [UIAlertAction
                                   actionWithTitle:@"Ok"
                                   style:UIAlertActionStyleDefault
                                   handler:^(UIAlertAction * action) {

                                   }];

        [alert addAction:okButton];
        [self presentViewController:alert animated:YES completion:nil];
    }
    return player2Stream;
}
```

7. Stream 1 playback stop.

FPWCSApi2Stream stopcode

```
- (void)player1Button:(UIButton *)button {
    [self changeViewState:button enabled:NO];
    if ([button.titleLabel.text isEqualToString:@"STOP"]) {
        if ([FPWCSApi2 getSessions].count) {
            NSError *error;
            [player1Stream stop:&error];
        } else {
            NSLog(@"Stop playing, no session");
            [self onStopped1];
        }
        ...
    }
}
```

8. Stream 2 playback stop.

FPWCSApi2Stream stopcode

```
- (void)player2Button:(UIButton *)button {
    [self changeViewState:button enabled:NO];
    if ([button.titleLabel.text isEqualToString:@"STOP"]) {
        if ([FPWCSApi2 getSessions].count) {
            NSError *error;
            [player2Stream stop:&error];
        } else {
            NSLog(@"Stop playing, no session");
            [self onStopped2];
        }
        ...
    }
}
```

9. Disconnection.

FPWCSApi2Session disconnectcode

```
- (void)connectButton:(UIButton *)button {
    [self changeViewState:button enabled:NO];
    if ([button.titleLabel.text isEqualToString:@"DISCONNECT"]) {
        if ([FPWCSApi2 getSessions].count) {
            FPWCSApi2Session *session = [FPWCSApi2 getSessions][0];
            NSLog(@"Disconnect session with server %@", [session getServerUrl]);
            [session disconnect];
        } else {
            NSLog(@"Nothing to disconnect");
            [self onDisconnected];
        }
    } else {
        //todo check url is not empty
        [self changeViewState:_connectUrl enabled:NO];
        [self connect];
    }
}
```

10. Receiving the event confirming successful disconnection.

ViewController onDisconnectedcode

```
- (void)onDisconnected {
    [_connectButton setTitle:@"CONNECT" forState:UIControlStateNormal];
    [self changeViewState:_connectButton enabled:YES];
    [self changeViewState:_connectUrl enabled:YES];
    [self onStopped1];
    [self onStopped2];
}
```