iOS Two-way Streaming

Пример iOS-приложения с плеером и стримером

Данное приложение может использоваться для публикации WebRTC-видеопотока и воспроизведения любого из следующих типов потоков с Web Call Server:

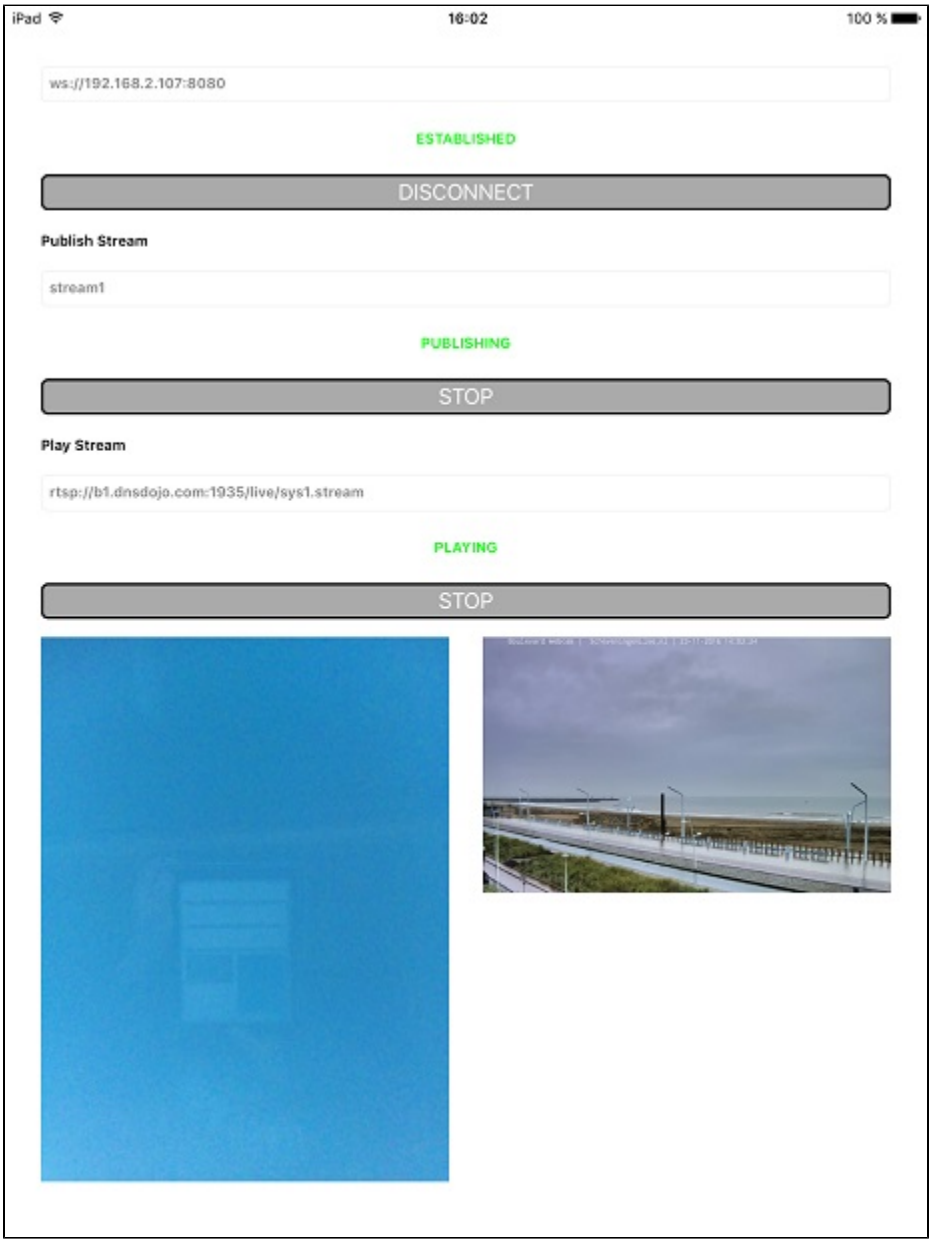
- RTSP
- WebRTC
- RTMP
- RTMFP

На скриншоте ниже представлен пример во время публикации и воспроизведени двух разных потоков.

Поля ввода

- 'WCS URL', где 192.168.2.107 - адрес WCS-сервера
- 'Publish Stream' - для имени публикуемого потока
- 'Play Stream' - для имени воспроизводимого потока

Слева отображается видео с камеры, справа воспроизводится другой поток.



Работа с кодом примера

Для разбора кода возьмем версию примера TwoWayStreaming, которая доступна для скачивания в соответствующей сборке 2.5.2.

Класс для основного вида приложения: ViewController (заголовочный файл ViewController.h; файл имплементации ViewController.m).

1. Импорт API. код

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

2. Создание сессии и подключение к серверу.

FPWCSApi2 createSession, FPWCSApi2Session connect код

В параметрах сессии указываются:

- URL WCS-сервера
- имя серверного приложения defaultApp

```
- (FPWCSApi2Session *)connect {
    FPWCSApi2SessionOptions *options = [[FPWCSApi2SessionOptions alloc] init];
    options.urlServer = _connectUrl.text;
    options.appKey = @"defaultApp";
    NSError *error;
    FPWCSApi2Session *session = [FPWCSApi2 createSession:options error:&error];
    ...
    [session connect];
    return session;
}
```

3. Публикация видеопотока.

FPWCSApi2Session createStream, FPWCSApi2Stream publish код

Методу createStream передаются параметры:

- имя публикуемого потока
- вид для локального отображения
- параметр record = true для записи потока при публикации
- размеры и FPS публикуемого видео при публикации с iPad

```
- (FPWCSApi2Stream *)publishStream {
    FPWCSApi2Session *session = [FPWCSApi2 getSessions][0];
    FPWCSApi2StreamOptions *options = [[FPWCSApi2StreamOptions alloc] init];
    options.name = _localStreamName.text;
    options.display = _localDisplay;
    if ( UI_USER_INTERFACE_IDIOM() == UIUserInterfaceIdiomPad ) {
        options.constraints = [[FPWCSApi2MediaConstraints alloc] initWithAudio:YES videoWidth:640 videoHeight:
480 videoFps:15];
    }
    NSError *error;
    FPWCSApi2Stream *stream = [session createStream:options error:&error];
    ...
    if(![stream publish:&error]) {
        UIAlertController * alert = [UIAlertController
                                     alertControllerWithTitle:@"Failed to publish"
                                     message:error.localizedDescription
                                     preferredStyle:UIAlertControllerStyleAlert];

        UIAlertAction* okButton = [UIAlertAction
                                   actionWithTitle:@"Ok"
                                   style:UIAlertActionStyleDefault
                                   handler:^(UIAlertAction * action) {
                                       [self onUnpublished];
                                   }];

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

4. Переключение камеры во время публикации потока

FPWCSApi2Stream switchCamera код

```
- (void)switchCameraButton:(UIButton *)button {
    if ([FPWCSApi2 getSessions].count) {
        FPWCSApi2Session *session = [FPWCSApi2 getSessions][0];
        NSArray *streams = [session getStreams];
        for (FPWCSApi2Stream *stream in streams ) {
            if ([stream isPublished]) {
                NSLog(@"Found published stream, switching camera");
                [stream switchCamera];
            }
        }
    } else {
        NSLog(@"No active sessions found");
    }
}
```

5. Воспроизведение видеопотока.

FPWCSApi2Session createStream, FPWCSApi2Stream play код

Методу createStream передаются параметры:

- имя воспроизводимого потока
- вид для отображения потока

```
- (FPWCSApi2Stream *)playStream {
    FPWCSApi2Session *session = [FPWCSApi2 getSessions][0];
    FPWCSApi2StreamOptions *options = [[FPWCSApi2StreamOptions alloc] init];
    options.name = _remoteStreamName.text;
    options.display = _remoteDisplay;
    NSError *error;
    FPWCSApi2Stream *stream = [session createStream:options error:nil];
    ...
    if(![stream 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 stream;
}
```

6. Остановка воспроизведения видеопотока.

FPWCSApi2Stream stop код

```
- (void)playButton:(UIButton *)button {
    [self changeViewState:button enabled:NO];
    if ([button.titleLabel.text isEqualToString:@"STOP"]) {
        if ([FPWCSApi2 getSessions].count) {
            FPWCSApi2Stream *stream;
            for (FPWCSApi2Stream *s in [[FPWCSApi2 getSessions][0] getStreams]) {
                if ([[s getName] isEqualToString:_remoteStreamName.text]) {
                    stream = s;
                    break;
                }
            }
            if (!stream) {
                NSLog(@"Stop playing, nothing to stop");
                [self onStopped];
                return;
            }
            NSError *error;
            [stream stop:&error];
        } else {
            NSLog(@"Stop playing, no session");
            [self onStopped];
        }
        ...
    }
}
```

7. Остановка публикации видеопотока.

FPWCSApi2Stream stop код

```
- (void)publishButton:(UIButton *)button {
    [self changeViewState:button enabled:NO];
    if ([button.titleLabel.text isEqualToString:@"STOP"]) {
        if ([FPWCSApi2 getSessions].count) {
            FPWCSApi2Stream *stream;
            for (FPWCSApi2Stream *s in [[FPWCSApi2 getSessions][0] getStreams]) {
                if ([[s getName] isEqualToString:_localStreamName.text]) {
                    stream = s;
                    break;
                }
            }
            if (!stream) {
                NSLog(@"Stop publishing, nothing to stop");
                [self onUnpublished];
                return;
            }
            NSError *error;
            [stream stop:&error];
        } else {
            NSLog(@"Stop publishing, no session");
            [self onUnpublished];
        }
        ...
    }
}
```

8. Закрытие соединения.

FPWCSApi2Session disconnect код

```
- (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];
        }
        ...
    }
}
```