

iOS GPUImageDemo

Example of video capturing using GPUImage library

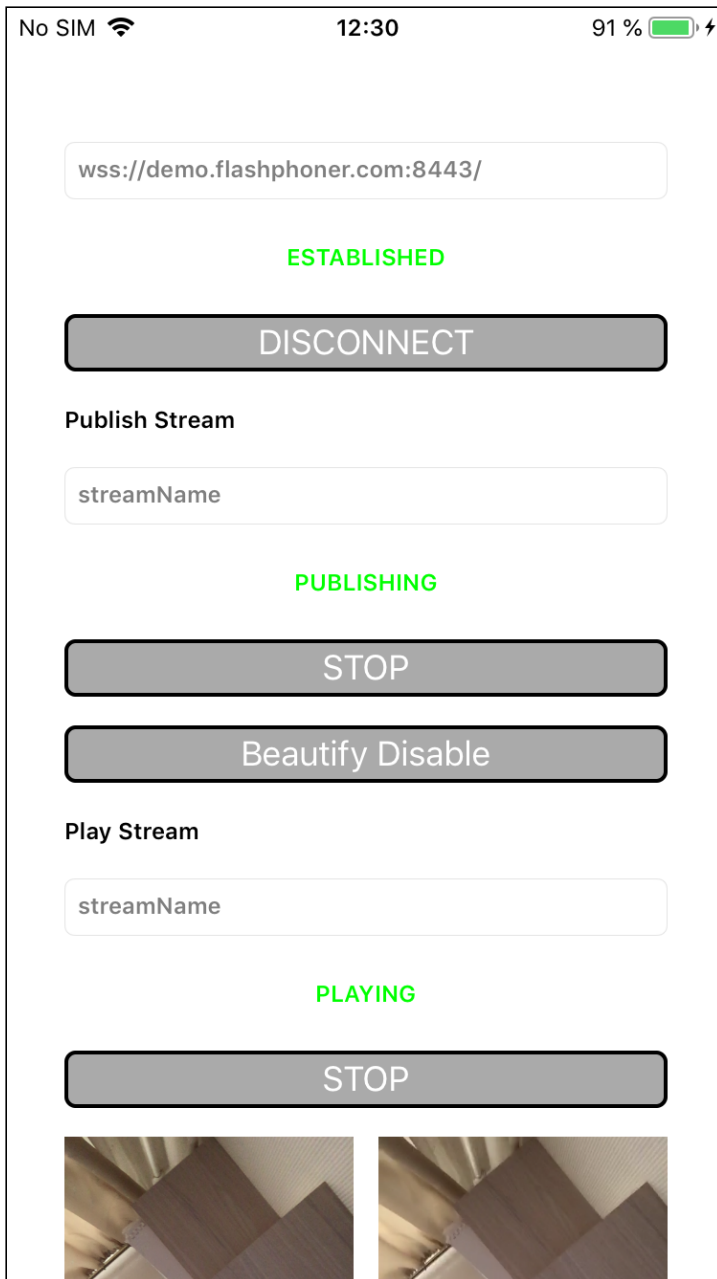
The application shows how to capture video from custom source. The source is GPUImage library with beautifying filter applied. This example works with SDK build [2.6.1](#) and newer.

On screenshot below, video is publishing with beautifying filter applied and playing

Inputs

- `WCS URL`, where `demo.flashphoner.com` is WCS server address
- `Publish Stream` to input stream name to publish
- `Play Stream` to input stream name to play

`Beautify Enable/Disable` button enables and disables beautifying filter (the filter is enabled on screenshot)



Analyzing the example code

To describe the code let's take GPUImageDemo application source code which is available [here](#).

Main application view class: ViewController (header file [ViewController.h](#); implementation file [ViewController.m](#)).

1. API import

[code](#)


```

    }];
    [self.videoCamera addTarget:_rawDataOutput];
    [self.videoCamera addTarget:_localDisplay];
  }
  ...
}

```

4. Stream publishing

`FPWCSApi2Session.createStream`, `FPWCSApi2Stream.publish`

code

The following parameters are passed to `createStream` method:

- stream name to publish
- local view to display
- video capture source

If `publish` is called successfully, video capture is started using `startCameraCapture` method

```

- (FPWCSApi2Stream *)publishStream {
    ...
    options.display = _localNativeDisplay;
    options.constraints = [[FPWCSApi2MediaConstraints alloc]
initWithAudio:YES videoCapturer:self.videoCapturer];

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

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

    [self.videoCamera startCameraCapture];

    return stream;
}

```

5. Enabling and disabling beautifying filter

code

```
- (void)beautify:(UIButton *)button {
    if (self.beautifyEnabled) {
        self.beautifyEnabled = NO;
        [_beautyButton setTitle:@"Beautify Enable"
forState:UIControlStateNormal];
        [self.videoCamera removeAllTargets];
        [self.videoCamera addTarget:_rawDataOutput];
        [self.videoCamera addTarget:_localDisplay];
    }
    else {
        self.beautifyEnabled = YES;
        [_beautyButton setTitle:@"Beautify Disable"
forState:UIControlStateNormal];
        [self.videoCamera removeAllTargets];
        GPUImageBeautifyFilter *beautifyFilter = [[GPUImageBeautifyFilter
alloc] init];
        [self.videoCamera addTarget:beautifyFilter];
        [beautifyFilter addTarget:_localDisplay];
        [beautifyFilter addTarget:_rawDataOutput];
    }
}
```

6. Stream playback

```
FPWCSApi2Session.createStream, FPWCSApi2Stream.play
```

code

The following parameters are passed to `createStream` method:

- stream name to play
- view to display remote stream

```
- (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
            initWithTitle:@"Ok"
            style:UIAlertActionStyleDefault
            handler:^(UIAlertAction * action) {

                }];

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

```

7. Playback stopping

`FPWCSApi2Stream.stop`

code

```

- (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];
        }
        ...
    }
}

```

8. Publishing stopping

`FPWCSApi2Stream.stop`

code

```

- (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];
        }
        ...
    }
}

```

9. Connection closing

`FPWCSApi2Session.disconnect`

code

```

- (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];
        }
        ...
    }
}

```