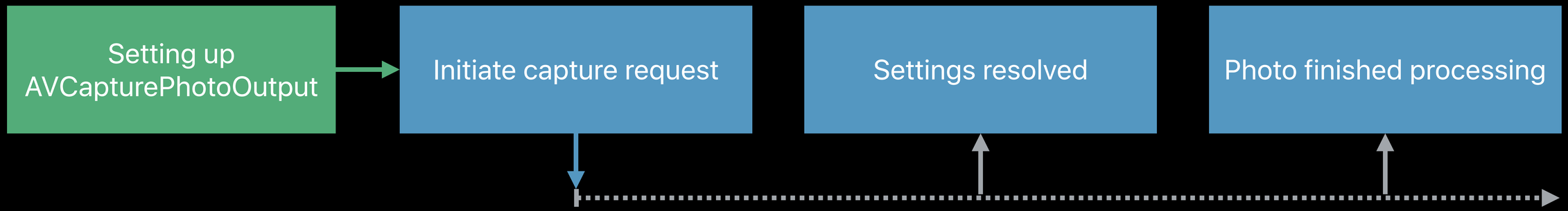
#WWDC19

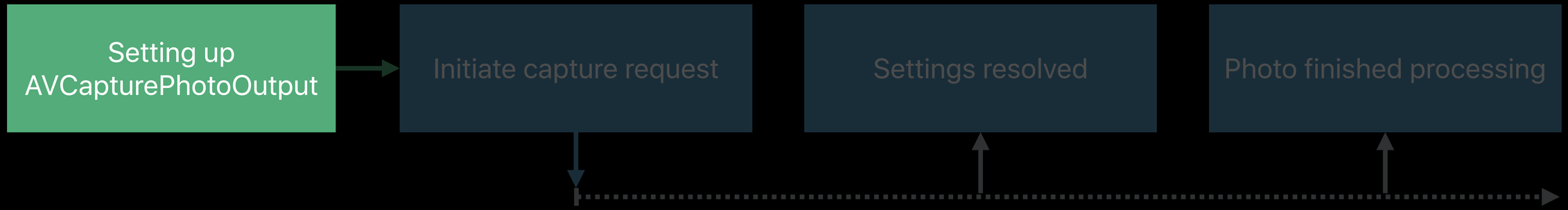
Advances in Camera Capture and Photo Segmentation

Brad Ford, Camera Software
Jacob Schack Vestergaard, Camera Software
David Hayward, Core Image

Capturing Segmentation Mattes

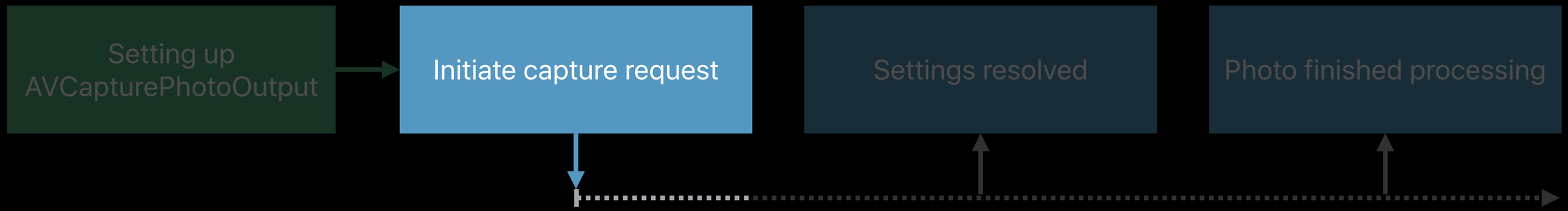

Setting up
AVCapturePhotoOutput
Initiate capture request
Settings resolved
Photo finished processing

Setting up AVCapturePhotoOutput



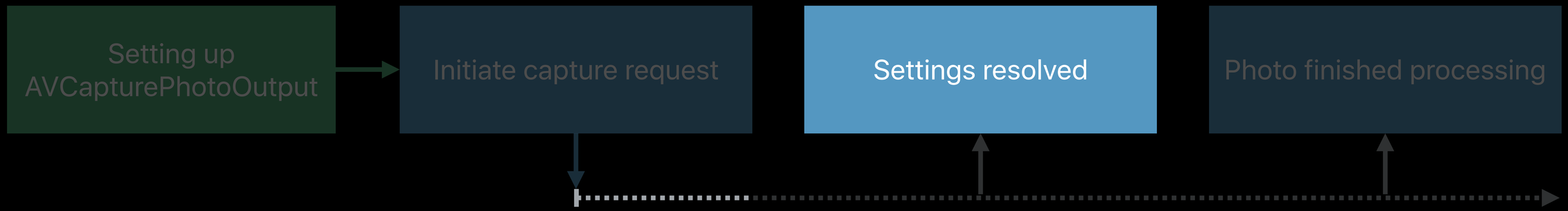

```
// begin configuration, set preset, add device input, ...
if session.canAddOutput(output) {
    session.addOutput(output)
    // what you usually do…

    output.enabledSemanticSegmentationMatteTypes = output.availableSemanticSegmentationMatteTypes
}
```

Initiating a Capture Request



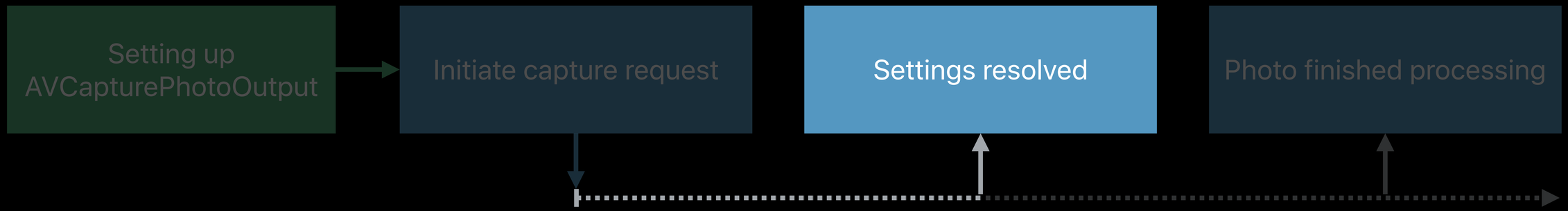

```
let settings = AVCapturePhotoSettings()

settings.enabledSemanticSegmentationMatteTypes = output.enabledSemanticSegmentationMatteTypes
// or
// settings.enabledSemanticSegmentationMatteTypes = [.hair, .skin]

photoOutput.capturePhoto(with: settings, delegate: self)
```

Resolved Capture Settings



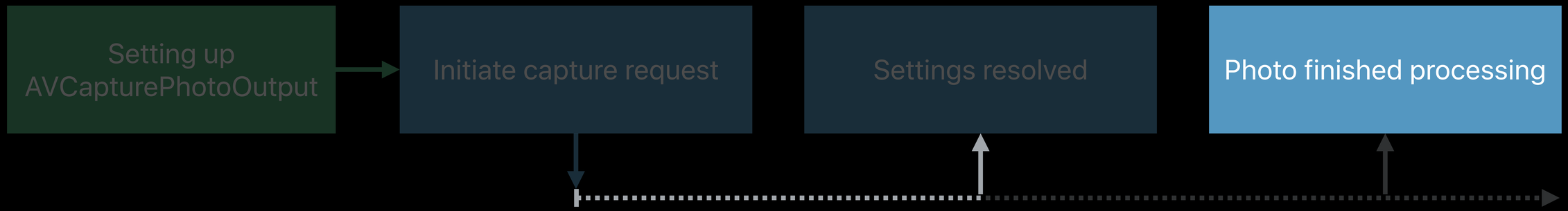

```
func photoOutput(_ output: AVCapturePhotoOutput,
    willBeginCaptureFor settings: AVCaptureResolvedPhotoSettings) {

    let matteDimensions = settings.dimensionsForSemanticSegmentationMatte(ofType: .hair)
}
```

Resolved Capture Settings



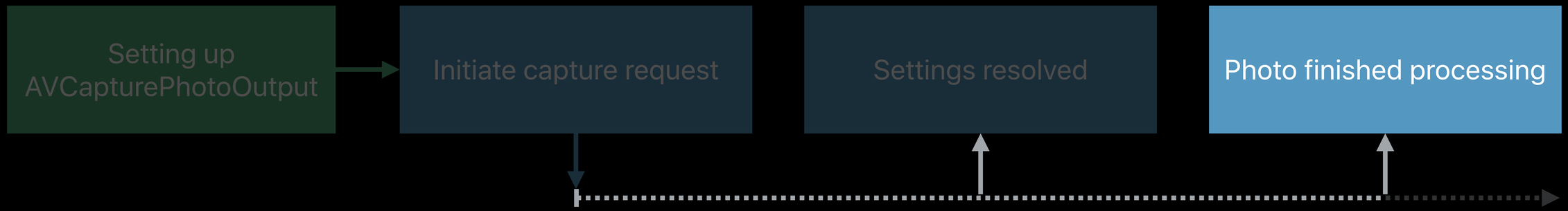

```
func photoOutput(_ output: AVCapturePhotoOutput,
    willBeginCaptureFor settings: AVCaptureResolvedPhotoSettings) {

    let matteDimensions = settings.dimensionsForSemanticSegmentationMatte(ofType: .hair)
}
```

Retrieving Matte on Capture




```
func photoOutput(_ output: AVCapturePhotoOutput,
    didFinishProcessingPhoto photo: AVCapturePhoto,
    error: Error?) {
        if var matte = photo.semanticSegmentationMatte(forType: .teeth) {
            let teethBuffer = matte.mattingImage
     }
}
```

Retrieving Matte on Capture




```
func photoOutput(_ output: AVCapturePhotoOutput,
    didFinishProcessingPhoto photo: AVCapturePhoto,
    error: Error?) {
        if var matte = photo.semanticSegmentationMatte(forType: .teeth) {
            let teethBuffer = matte.mattingImage
     }
}
```

AVCam

AVCam

Leveraging Core Image

David Hayward, Core Image

Demo

Coulrophobia
[kool-ruh-foh-bee-uh]
An extreme or irrational fear of clowns

Using Segmentation Mattes with Core Image

Using Segmentation Mattes with Core Image

Creating matte images

Using Segmentation Mattes with Core Image

Creating matte images

Filtering matte images

Using Segmentation Mattes with Core Image

Creating matte images

Filtering matte images

Saving matte images

Creating Segmentation Mattes with Core Image

NEW

Creating a matte CIImage from AVSemanticSegmentationMatte

Creating Segmentation Mattes with Core Image

NEW

Creating a matte CIImage from AVSemanticSegmentationMatte


```
let matte = photo.semanticSegmentationMatte(forType: .hair)  // or .skin or .teeth
```

Creating Segmentation Mattes with Core Image

NEW

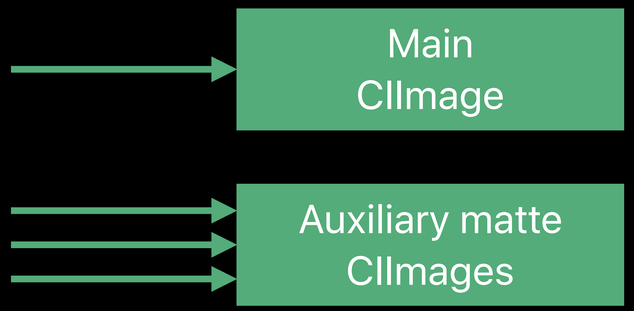
Creating a matte CIImage from AVSemanticSegmentationMatte


```
let matte = photo.semanticSegmentationMatte(forType: .hair)  // or .skin or .teeth
let img = CIImage(semanticSegmentationMatte: matte)
```

Creating Segmentation Mattes with Core Image

NEW

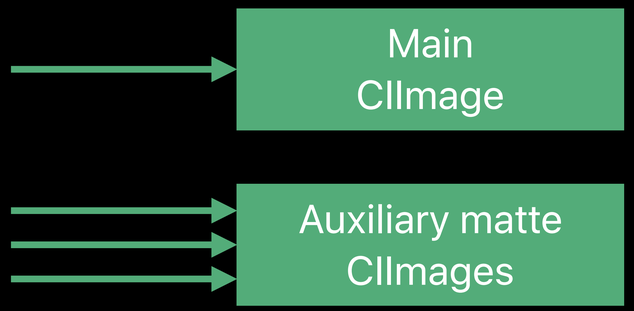
Loading a matte CIImage from HEIF



Creating Segmentation Mattes with Core Image

NEW

Loading a matte CIImage from HEIF

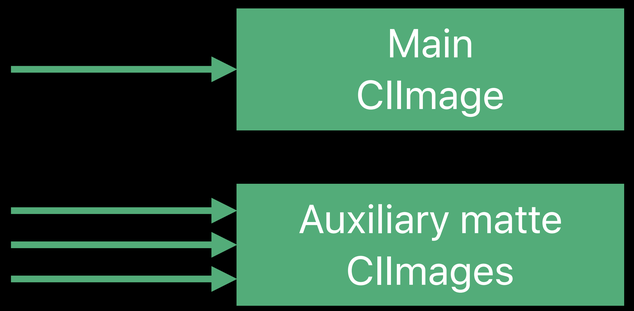



```
let main = CIImage(contentsOf: url)
```

Creating Segmentation Mattes with Core Image

NEW

Loading a matte CIImage from HEIF

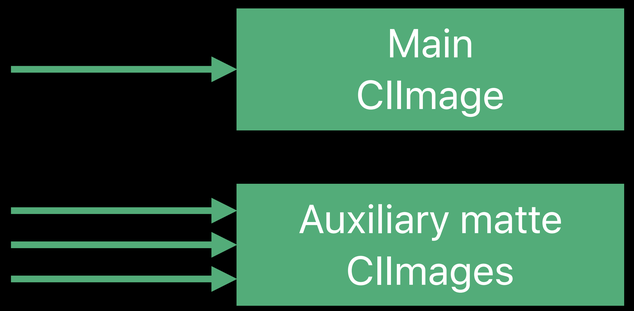



```
let main = CIImage(contentsOf: url)
let hair = CIImage(contentsOf: url,
                   options: [.auxiliarySemanticSegmentationHairMatte : true])
```

Creating Segmentation Mattes with Core Image

NEW

Loading a matte CIImage from HEIF

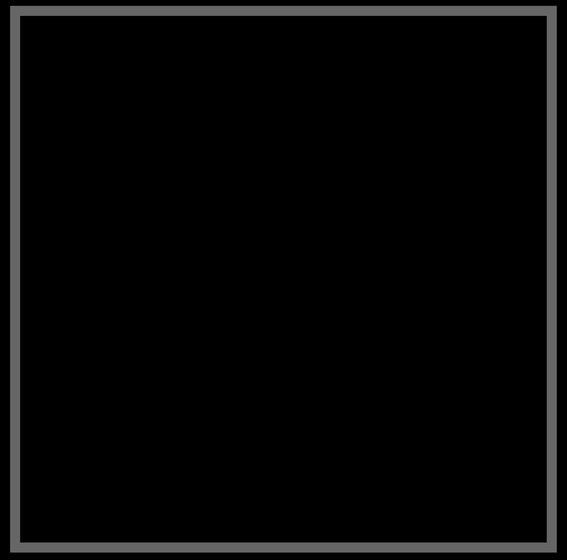



```
let main = CIImage(contentsOf: url)
let hair = CIImage(contentsOf: url,
                   options: [.auxiliarySemanticSegmentationHairMatte : true])
                       // or .auxiliarySemanticSegmentationSkinMatte
                       // or .auxiliarySemanticSegmentationTeethMatte
```

Filtering Segmentation Mattes with Core Image

Filtering Segmentation Mattes with Core Image

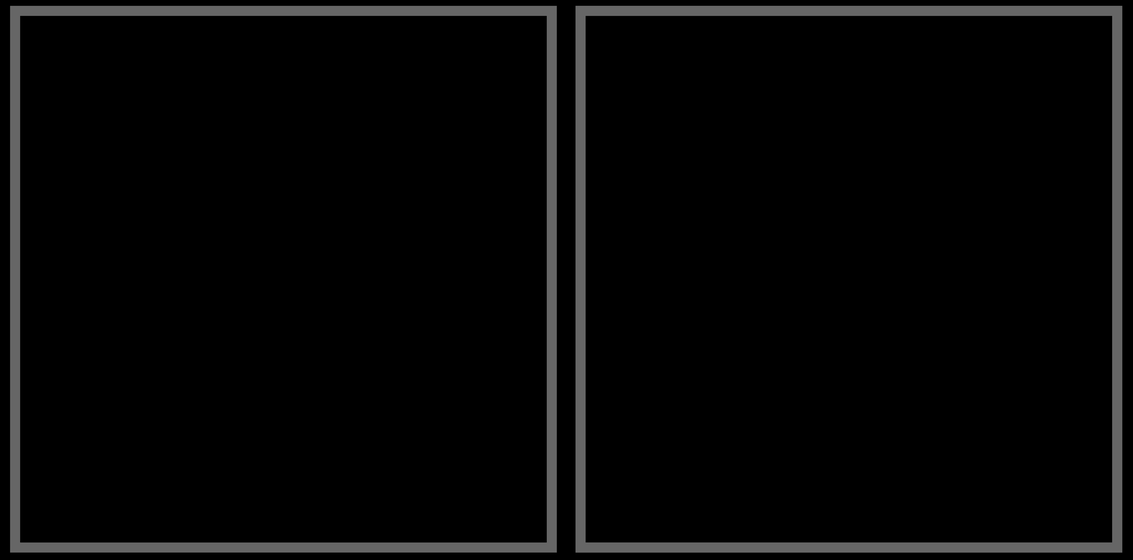
Base



Filtering Segmentation Mattes with Core Image

Base

Adjusted

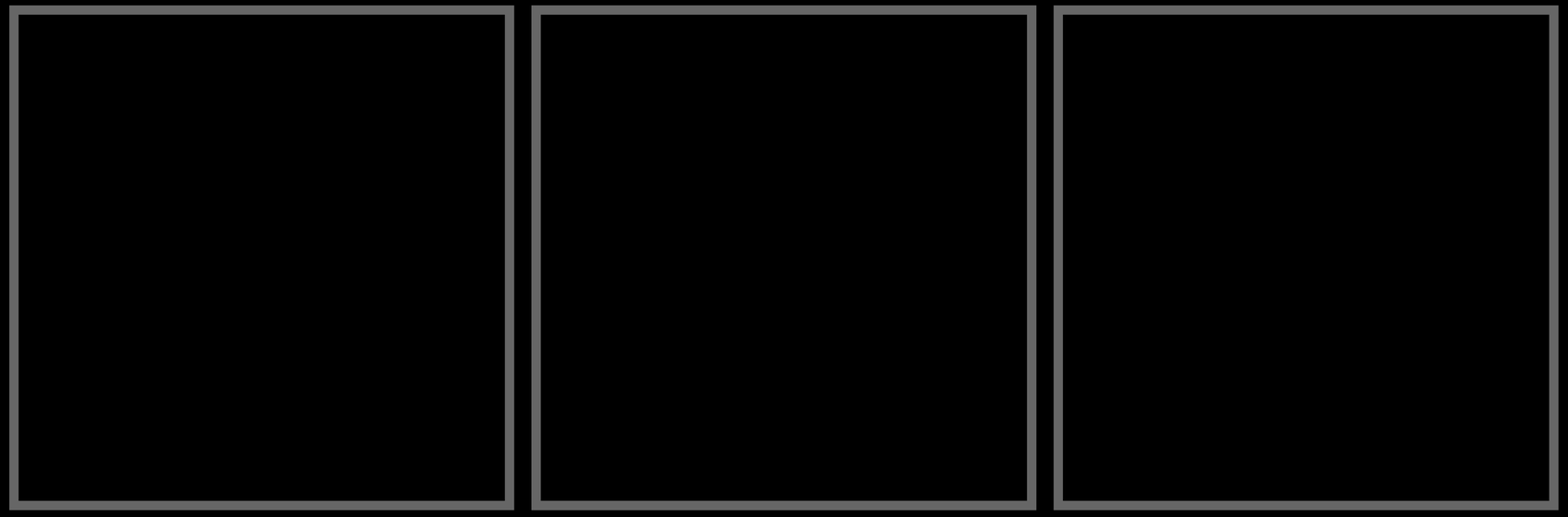


Filtering Segmentation Mattes with Core Image

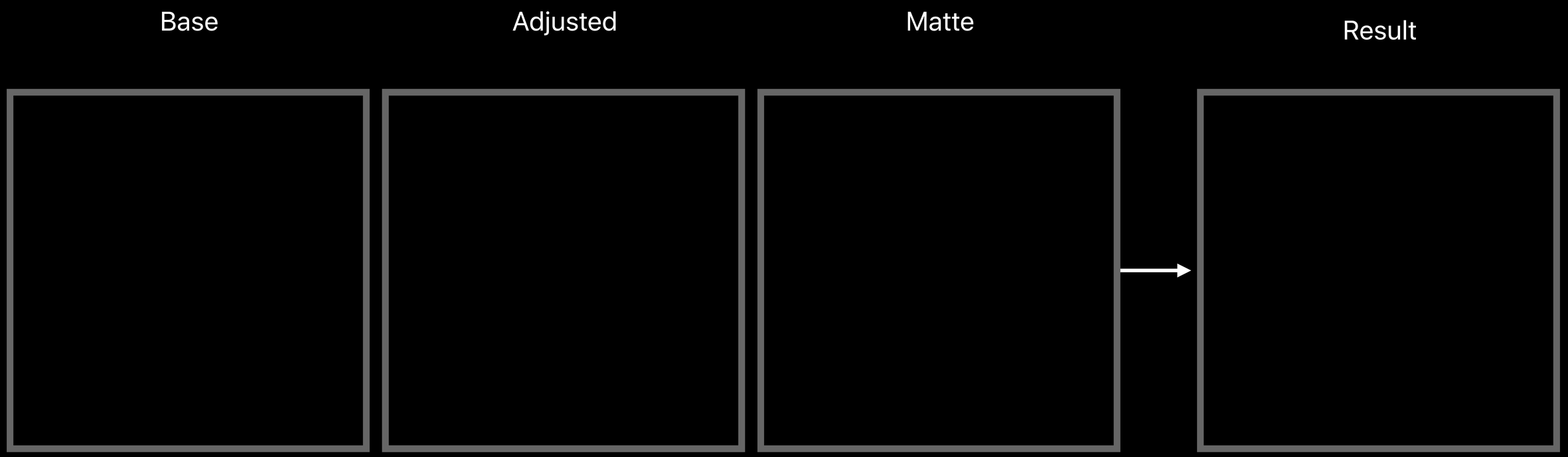
Base

Adjusted

Matte



Filtering Segmentation Mattes with Core Image



```
// Filtering Segmentation Mattes with Core Image
```

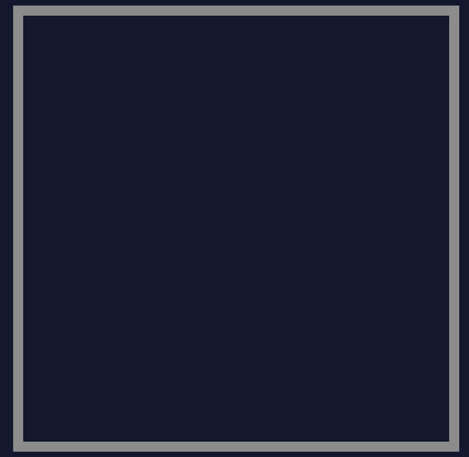
```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins
```

NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let base = CIImage( contentsOf : url )
```

NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let base = CIImage( contentsOf : url )
```



NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let base = CIImage( contentsOf : url )

let maxcomp = CIFilter.maximumComponent
    maxcomp.inputImage = base
```

NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let base = CIImage( contentsOf : url )

let maxcomp = CIFilter.maximumComponent
    maxcomp.inputImage = base
var makeup = maxcomp.outputImage
```

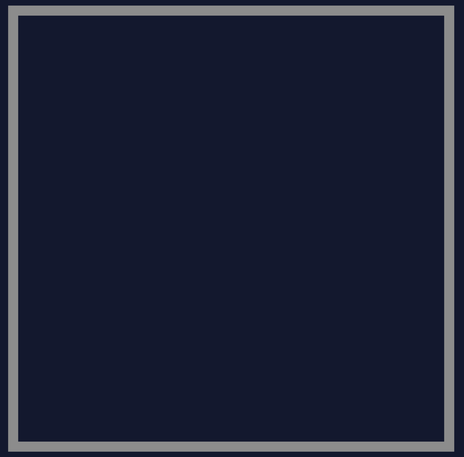
NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let base = CIImage( contentsOf : url )

let maxcomp = CIFilter.maximumComponent
    maxcomp.inputImage = base
var makeup = maxcomp.outputImage
```

NEW

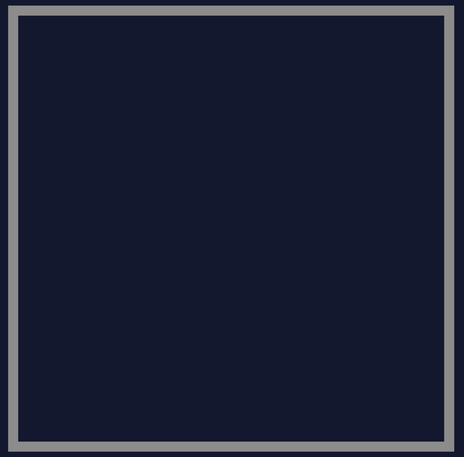
```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let base = CIImage( contentsOf : url )

let maxcomp = CIFilter.maximumComponent
    maxcomp.inputImage = base
var makeup = maxcomp.outputImage

let gamma = CIFilter.gammaAdjust
    blend.inputImage = makeup
    blend.power = 0.5
```

NEW



```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let base = CIImage( contentsOf : url )

let maxcomp = CIFilter.maximumComponent
    maxcomp.inputImage = base
var makeup = maxcomp.outputImage

let gamma = CIFilter.gammaAdjust
    blend.inputImage = makeup
    blend.power = 0.5
makeup = gamma.outputImage
```

NEW



```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let base = CIImage( contentsOf : url )

let maxcomp = CIFilter.maximumComponent
    maxcomp.inputImage = base
var makeup = maxcomp.outputImage

let gamma = CIFilter.gammaAdjust
    blend.inputImage = makeup
    blend.power = 0.5
makeup = gamma.outputImage
```

NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins
```

NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

var matte = CIImage( contentsOf : url,
            options : [.auxiliarySemanticSegmentationSkinMatte : true] )
```

NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

var matte = CIImage( contentsOf : url,
            options : [.auxiliarySemanticSegmentationSkinMatte : true] )
```

NEW

NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

var matte = CIImage( contentsOf : url,
            options : [.auxiliarySemanticSegmentationSkinMatte : true] )

let scale = CGAffineTransformMakeScale(
            base.extent.size.width  / matte.extent.size.width,
            base.extent.size.height / matte.extent.size.height)
```

NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

var matte = CIImage( contentsOf : url,
            options : [.auxiliarySemanticSegmentationSkinMatte : true] )

let scale = CGAffineTransformMakeScale(
            base.extent.size.width  / matte.extent.size.width,
            base.extent.size.height / matte.extent.size.height)

matte = matte.transformed( by: scale )
```

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

var matte = CIImage( contentsOf : url,
            options : [.auxiliarySemanticSegmentationSkinMatte : true] )

let scale = CGAffineTransformMakeScale(
            base.extent.size.width  / matte.extent.size.width,
            base.extent.size.height / matte.extent.size.height)

matte = matte.transformed( by: scale )
```

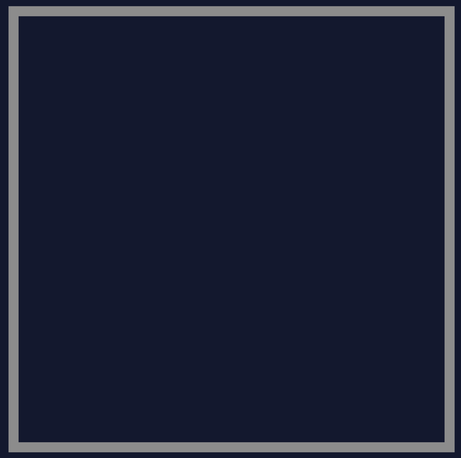
NEW

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins
```

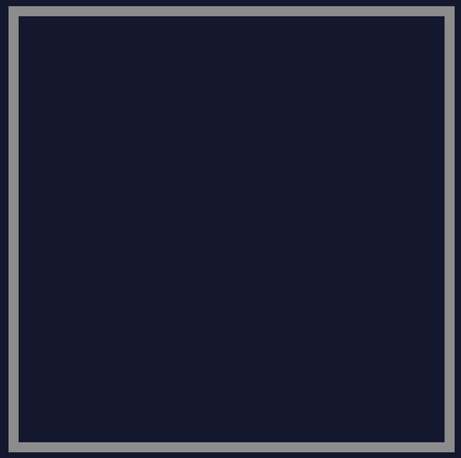
```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let blend = CIFilter.blendWithMask
```

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let blend = CIFilter.blendWithMask
    blend.backgroundImage = base
```

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let blend = CIFilter.blendWithMask
    blend.backgroundImage = base
```



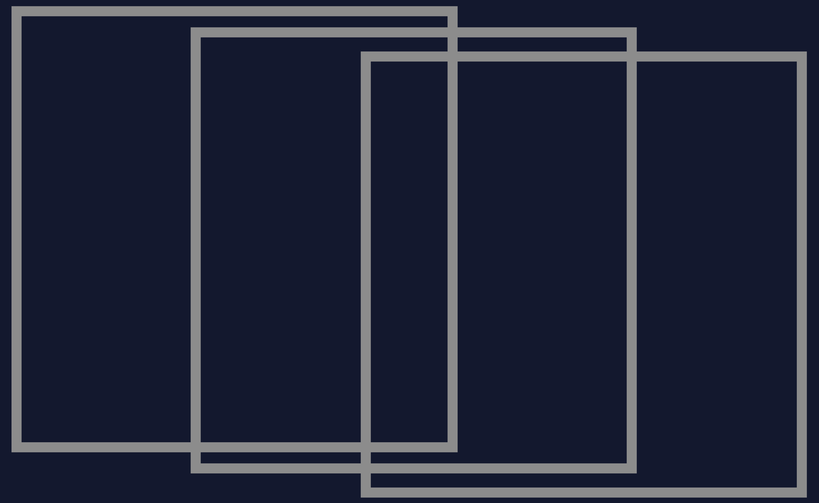
```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let blend = CIFilter.blendWithMask
    blend.backgroundImage = base
    blend.inputImage = makeup
```



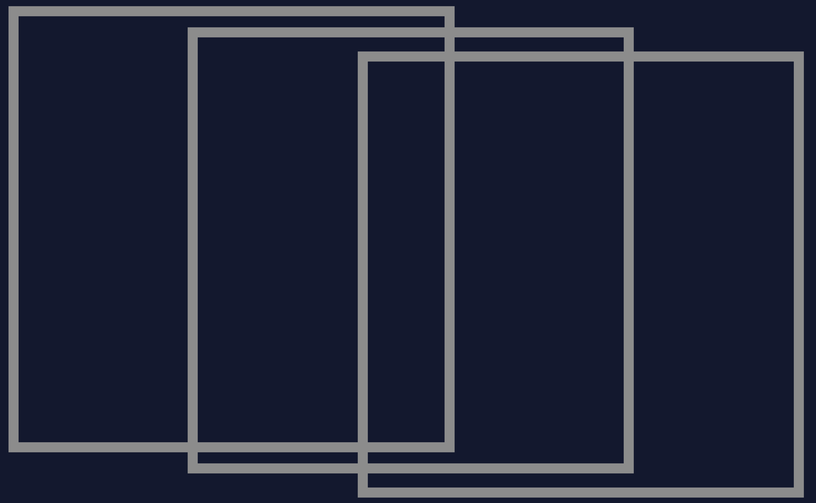
```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let blend = CIFilter.blendWithMask
    blend.backgroundImage = base
    blend.inputImage = makeup
```

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let blend = CIFilter.blendWithMask
    blend.backgroundImage = base
    blend.inputImage = makeup
    blend.maskImage = matte
```

```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let blend = CIFilter.blendWithMask
    blend.backgroundImage = base
    blend.inputImage = makeup
    blend.maskImage = matte
```



```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let blend = CIFilter.blendWithMask
    blend.backgroundImage = base
    blend.inputImage = makeup
    blend.maskImage = matte

let result = blend.outputImage
```



```
// Filtering Segmentation Mattes with Core Image

import CoreImage.CIFilterBuiltins

let blend = CIFilter.blendWithMask
    blend.backgroundImage = base
    blend.inputImage = makeup
    blend.maskImage = matte

let result = blend.outputImage
```

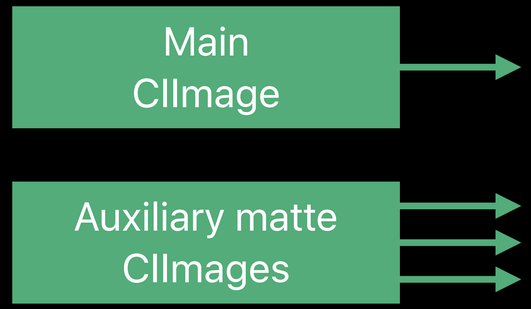
Saving Segmentation Mattes with Core Image

NEW

Saving Segmentation Mattes with Core Image

NEW

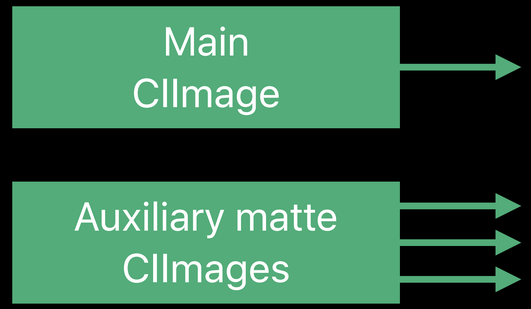
Saving to HEIF with matte CIImages



Saving Segmentation Mattes with Core Image

NEW

Saving to HEIF with matte CIImages

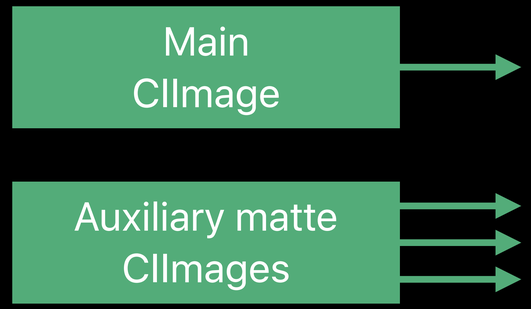



```
context.writeHEIFRepresentation(of: result,
                                to: url,
                            format: .RGBA8,
                        colorSpace: mainImage.colorSpace,
```

Saving Segmentation Mattes with Core Image

NEW

Saving to HEIF with matte CIImages

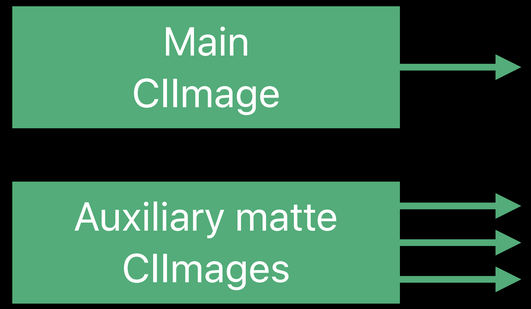



```
context.writeHEIFRepresentation(of: result,
                                to: url,
                            format: .RGBA8,
                        colorSpace: mainImage.colorSpace,
                           options: [.semanticSegmentationSkinMatteImage : skinImage,
```

Saving Segmentation Mattes with Core Image

NEW

Saving to HEIF with matte CIImages




```
context.writeHEIFRepresentation(of: result,
                                to: url,
                            format: .RGBA8,
                        colorSpace: mainImage.colorSpace,
                           options: [.semanticSegmentationSkinMatteImage : skinImage,
                                     .semanticSegmentationHairMatteImage : hairImage,
                                     .semanticSegmentationTeethMatteImage : teethImage])
```

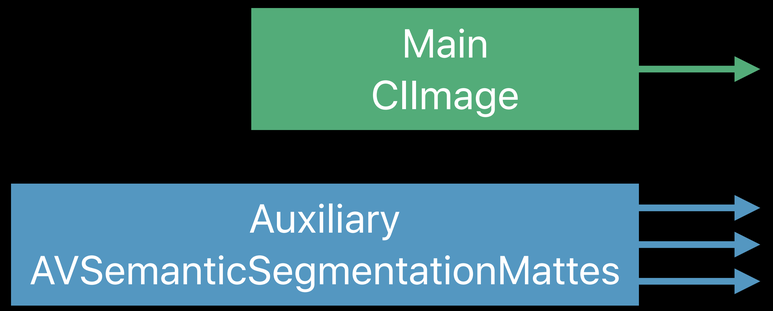
Saving Segmentation Mattes with Core Image

NEW

Saving Segmentation Mattes with Core Image

NEW

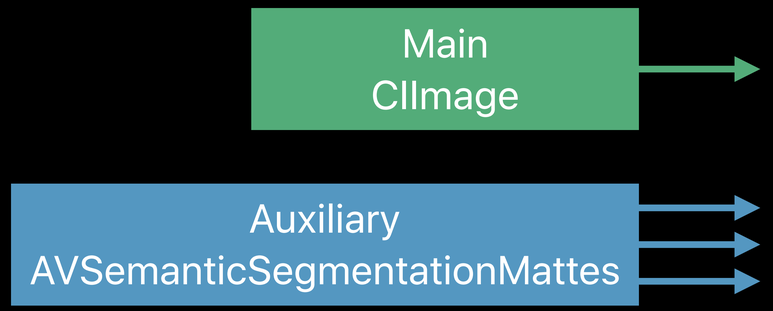
Saving to HEIF with AVSemanticSegmentationMattes



Saving Segmentation Mattes with Core Image

NEW

Saving to HEIF with AVSemanticSegmentationMattes

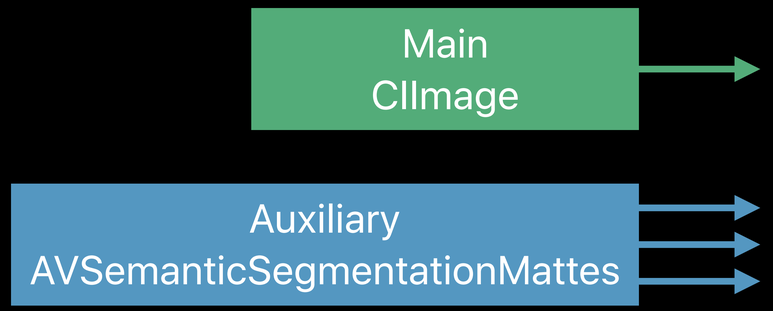



```
context.writeHEIFRepresentation(of: result,
                                to: url,
                            format: .RGBA8,
                        colorSpace: mainImage.colorSpace,
```

Saving Segmentation Mattes with Core Image

NEW

Saving to HEIF with AVSemanticSegmentationMattes




```
context.writeHEIFRepresentation(of: result,
                                to: url,
                            format: .RGBA8,
                        colorSpace: mainImage.colorSpace,
                           options: [.AVSemanticSegmentationMattes :
                                         [ skinMatte, hairMatte, teethMatte]]
```

Summary

Summary

Creating matte images

Filtering matte images

Saving matte images

More Information

developer.apple.com/wwdc19/260

Capturing Depth in iPhone Photography	WWDC 2017
Introducing the Photos Frameworks	WWDC 2014


WWDC19