App Frameworks

#WWDC17

Building Visually Rich User Experiences

Session 235

Noah Witherspoon, Software Engineer
Warren Moore, Software Engineer

Platform overview

Core Animation best practices

Tips and tricks

Color space

Render server

Timing functions

Pixel format

Interpolation

Projection

Filter chain

Convolution

Dithering

Transform matrix

Shader

Multisampling

Image unit

Blend mode

Transition

Presentation layer

Fill mode

Texture atlas

Content gravity

Compositing

Post-processing

Depth buffer

Graphics context

Rasterization

Mask

Framebuffer

Vibrancy

Shadow mapping

Offscreen pass

Interpolator

Winding order

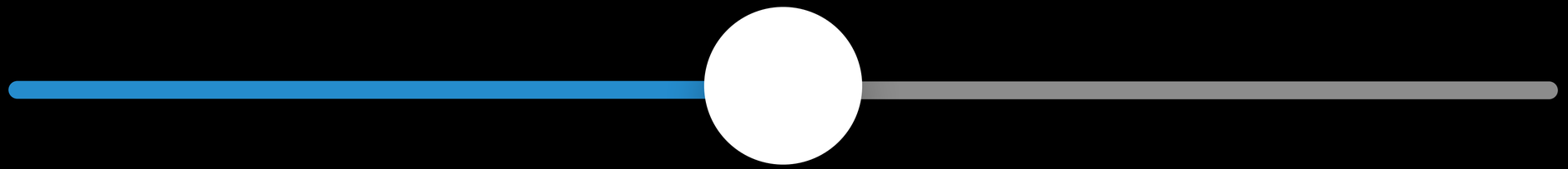
Pipeline descriptor

Sprite

Minification filter

Platform Overview

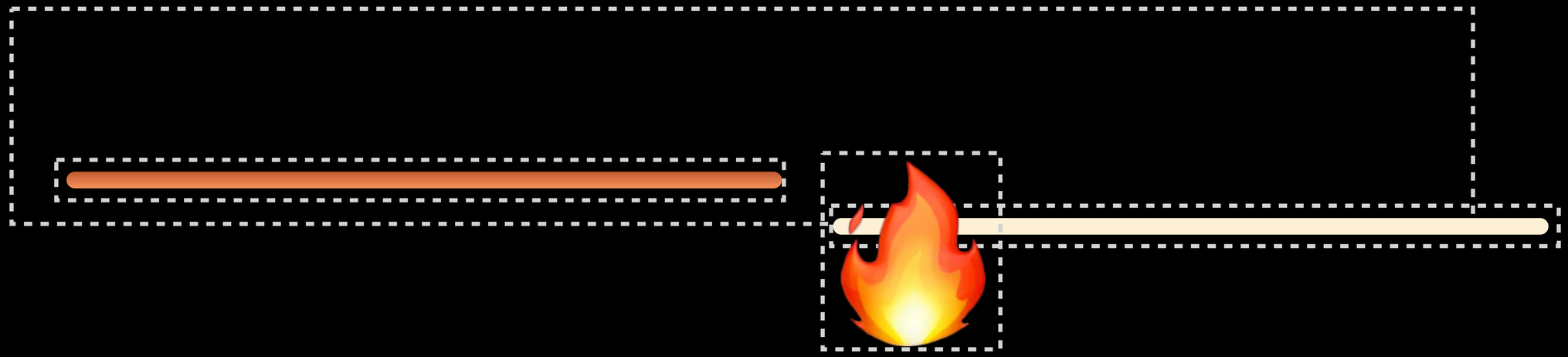
UIKit and AppKit



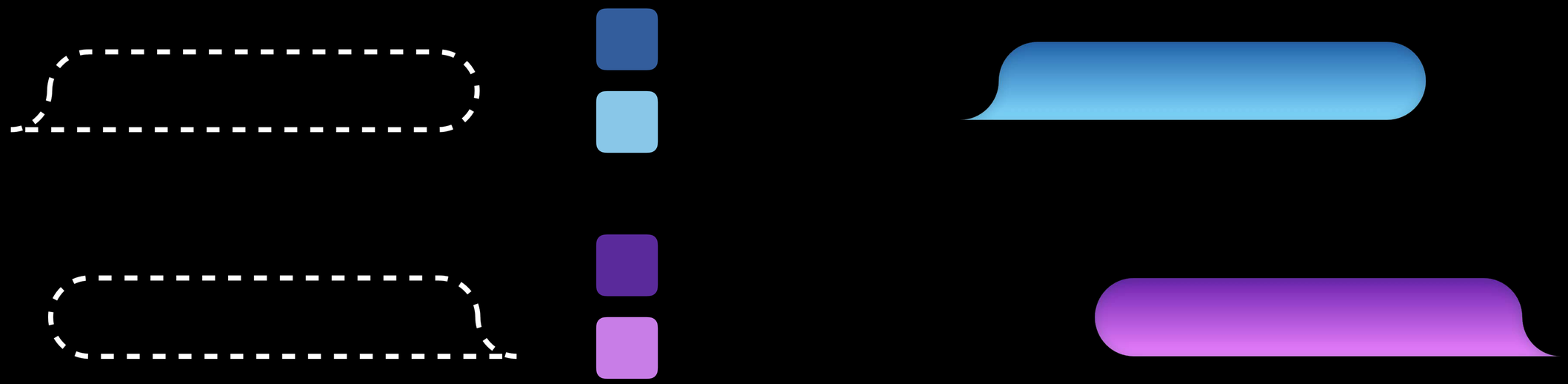
UIKit and AppKit

Core Animation

Core Animation

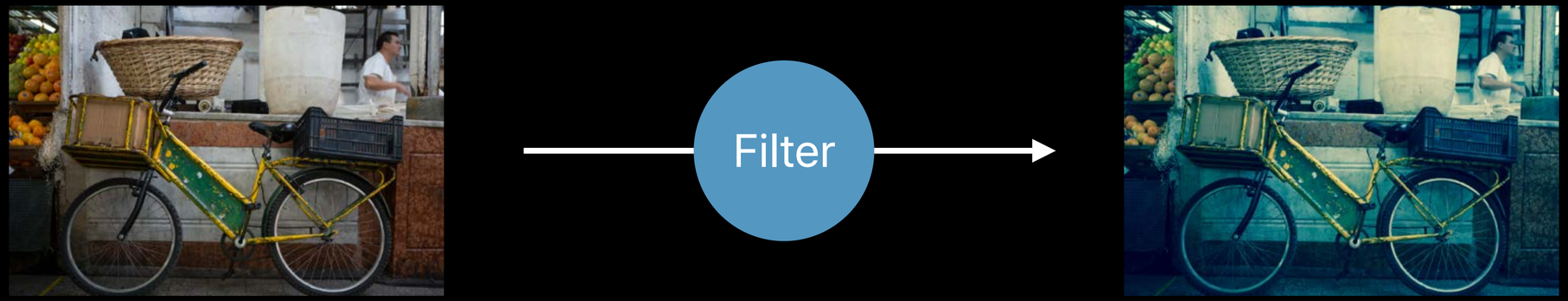


Core Graphics



Core Image

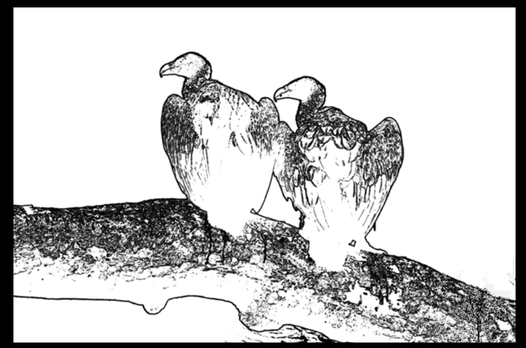
Core Image



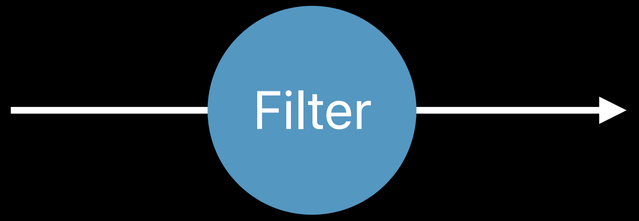
Core Image

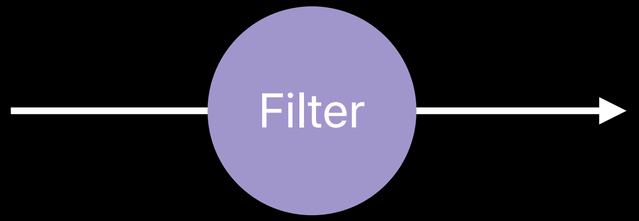
Filter

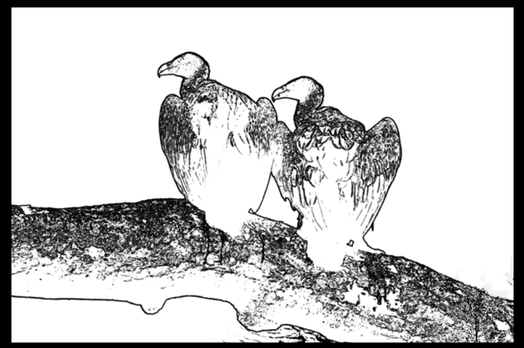
Filter

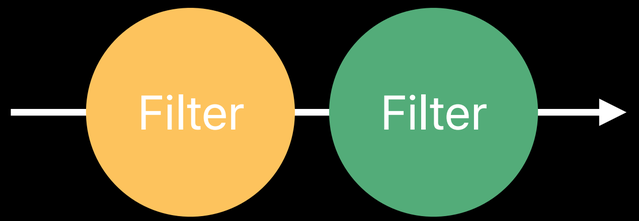


Core Image

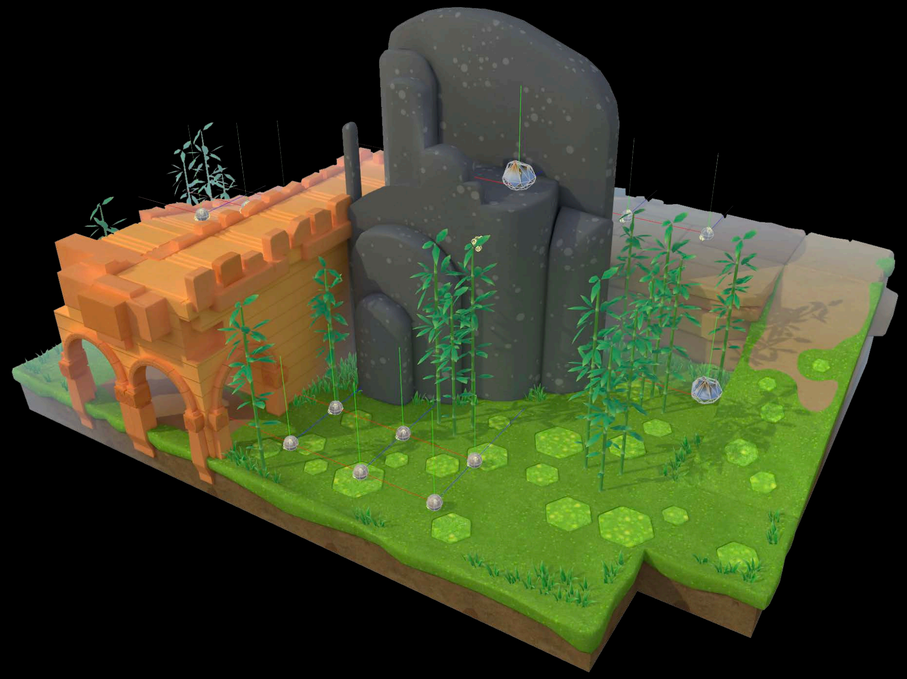








SceneKit and SpriteKit



Metal

Platform Overview

UIKit/AppKit

Core Animation

Core Graphics/Core Image

SceneKit/SpriteKit

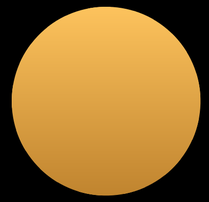
Metal

Demo

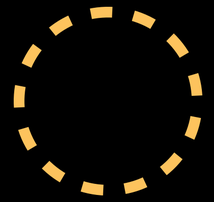
Warren Moore

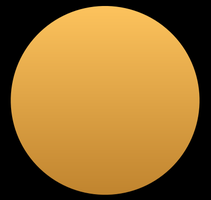
Core Animation Best Practices

Animations

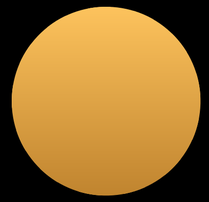


Animations

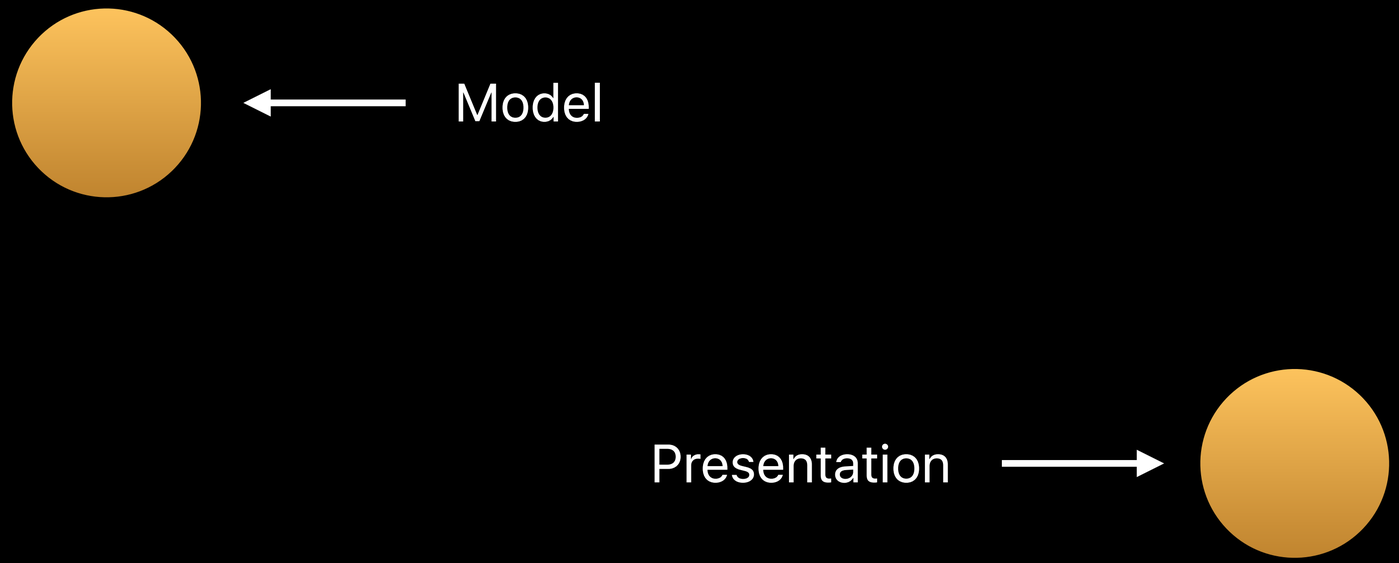




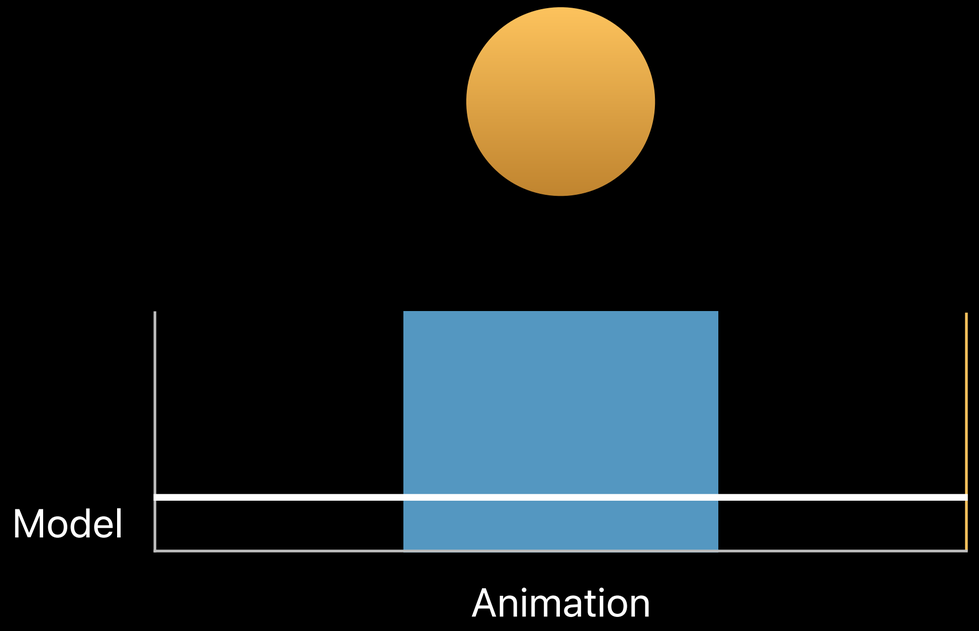
Animations

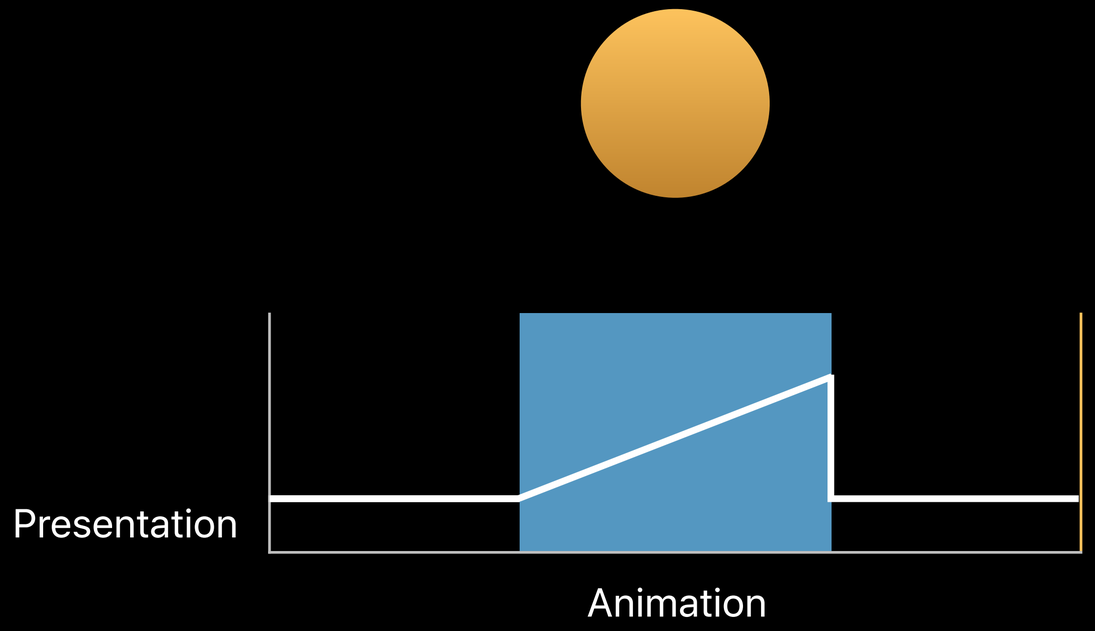


Animations

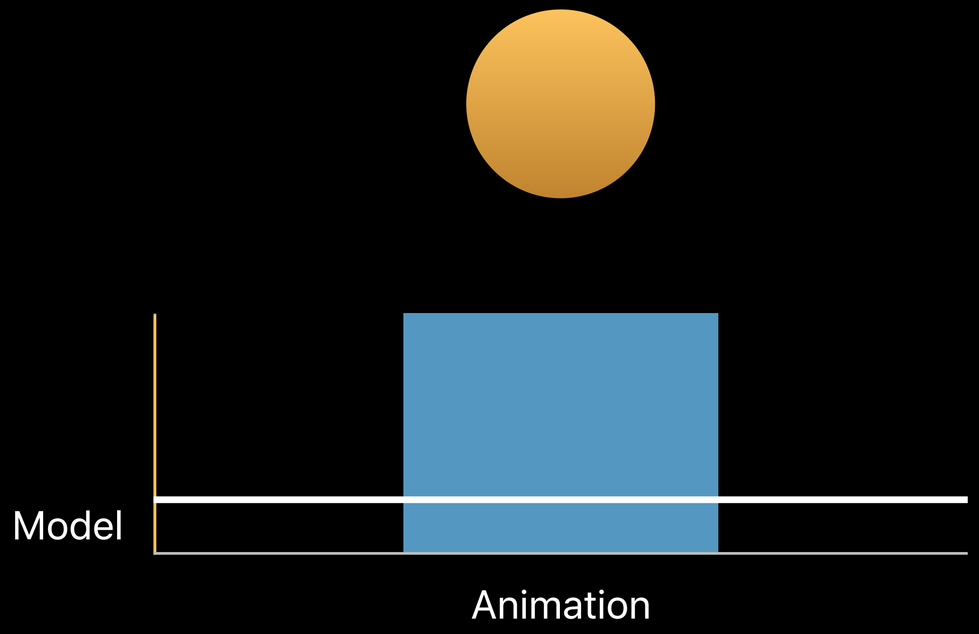


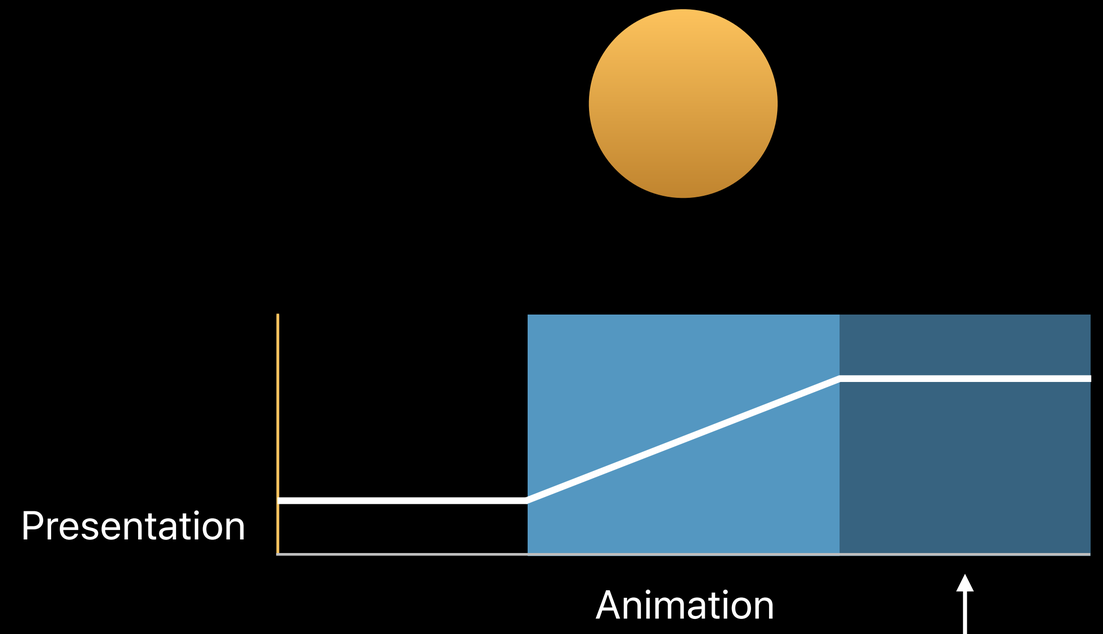
Animations





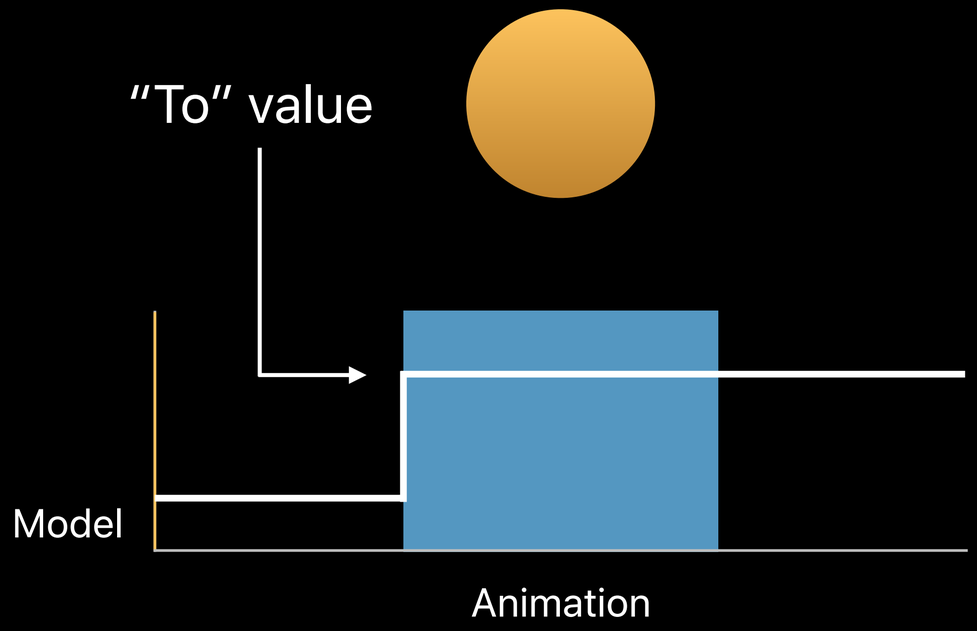
Animations

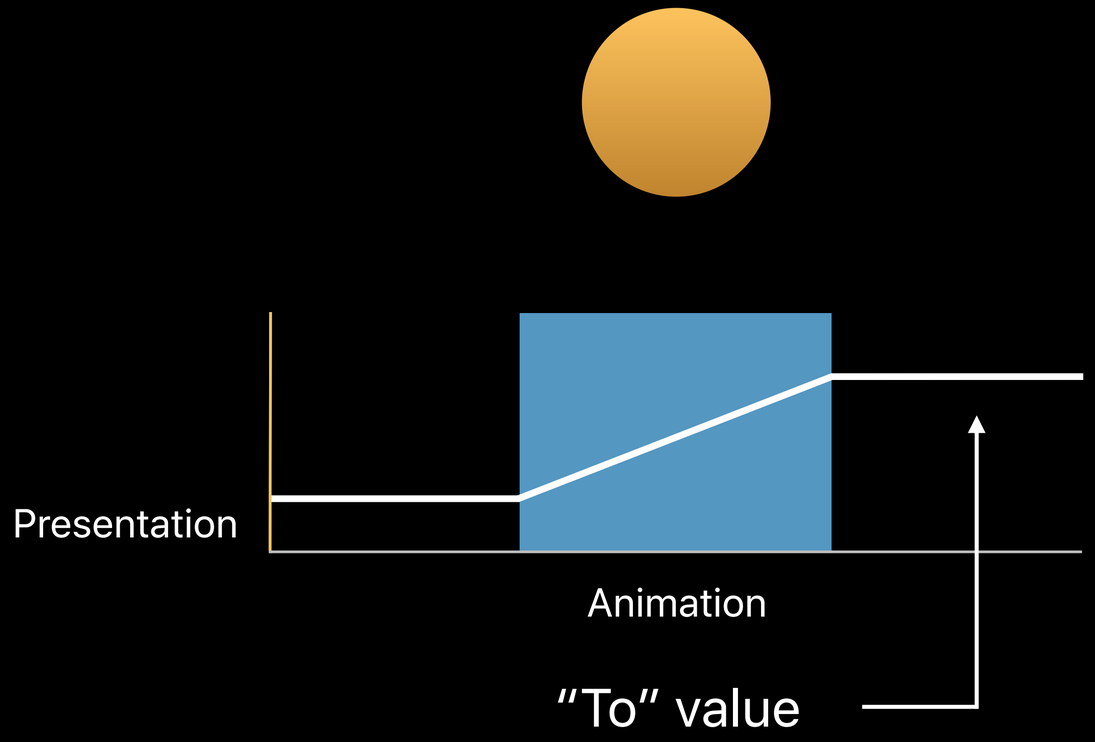





```
animation.removedOnCompletion = false
animation.fillMode = kCAFillModeForwards
```

Animations





Animations

```
let animation = let animation = CABasicAnimation(keyPath: "opacity")
animation.duration = 0.5
animation.fromValue = 1
animation.toValue = 0

layer.add(animation, forKey: nil)
layer.opacity = 0
```

Animations

Be aware of model/presentation layers

Apply final state when adding animations

Transforms and Frames

```
layer.frame = CGRect(x: 0, y: 0, width: 200, height: 100)
```

Transforms and Frames

```
layer.transform = CGAffineTransform(scaleX: 0.25, y: 0.25)
```

Transforms and Frames

```
layer.transform = CGAffineTransform(scaleX: 0.25, y: 0.25)
```

Transforms and Frames

```
layer.frame = CGRect(x: 0, y: 0, width: 200, height: 100)
```

Transforms and Frames

```
layer.frame = CGRect(x: 0, y: 0, width: 200, height: 100)
```

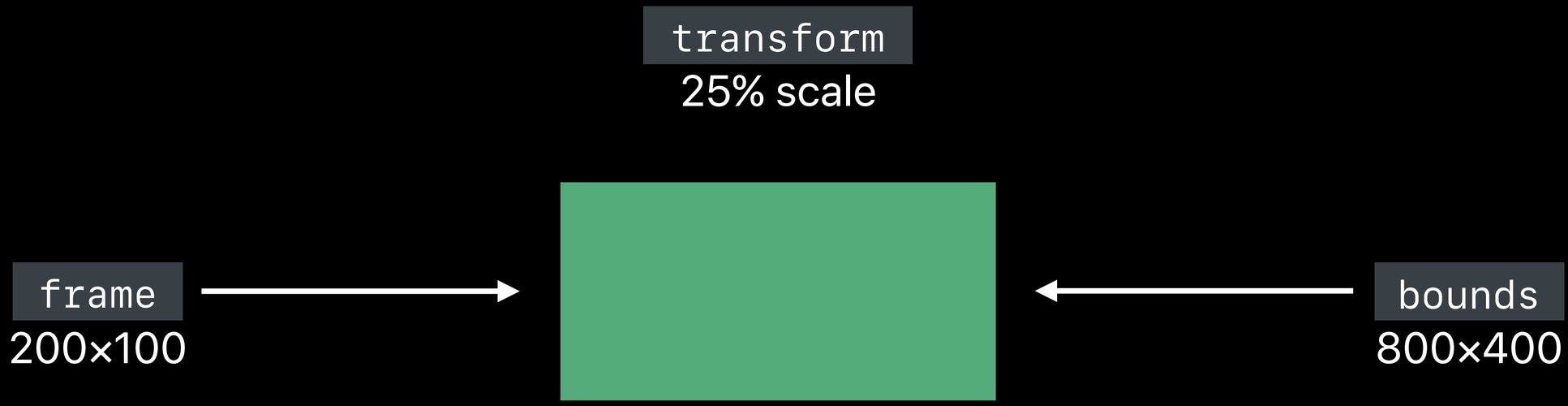
Transforms and Frames

```
layer.transform = CGAffineTransform(rotationAngle: 0.2)
```

Transforms and Frames

```
layer.transform = CGAffineTransform(rotationAngle: 0.2)
```

Transforms and Frames



Transforms and Frames

```
layer.transform = CATransform3D(scaleX: 0.25, y: 0.25)
// …
let newFrame = CGRect(x: 0, y: 0, width: 200, height: 100)

layer.frame = newFrame
```

Transforms and Frames

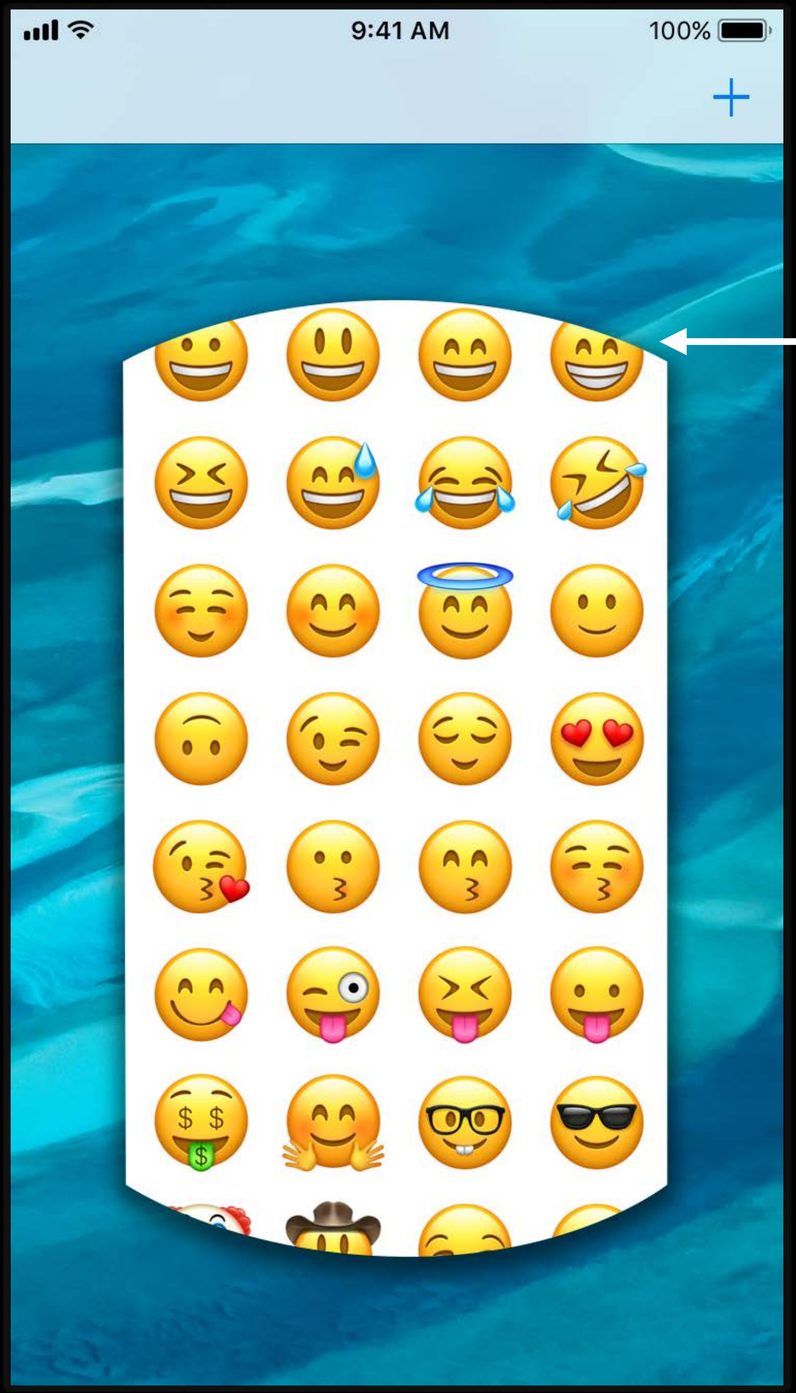
```
layer.transform = CATransform3D(scaleX: 0.25, y: 0.25)
// …
let newFrame = CGRect(x: 0, y: 0, width: 200, height: 100)

layer.bounds = CGRect(x: 0, y: 0, width: newFrame.width, height: newFrame.height)
layer.position = CGPoint(x: someRect.midX, y: someRect.midY)
```

Transforms and Frames

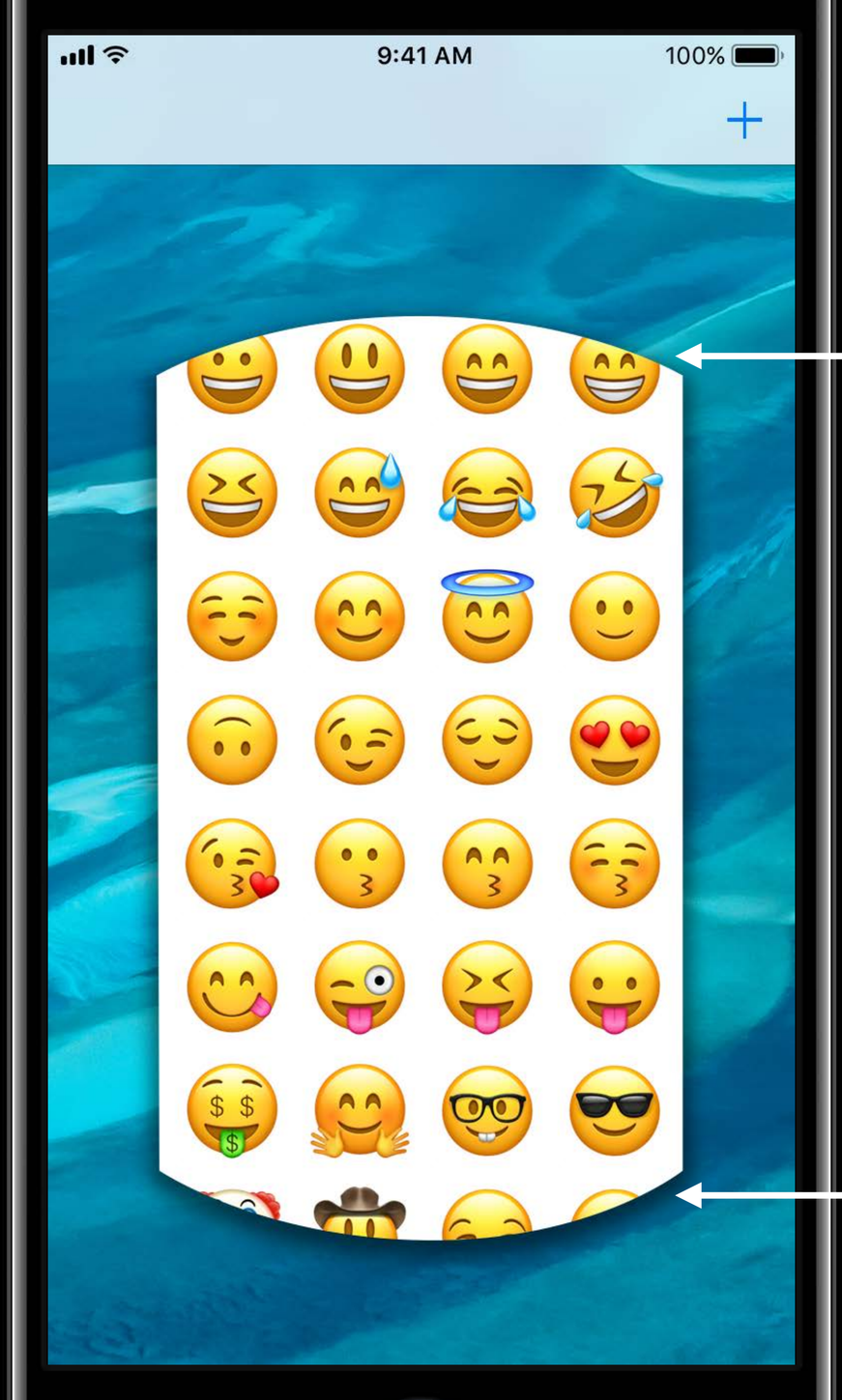
Frame is affected by transform

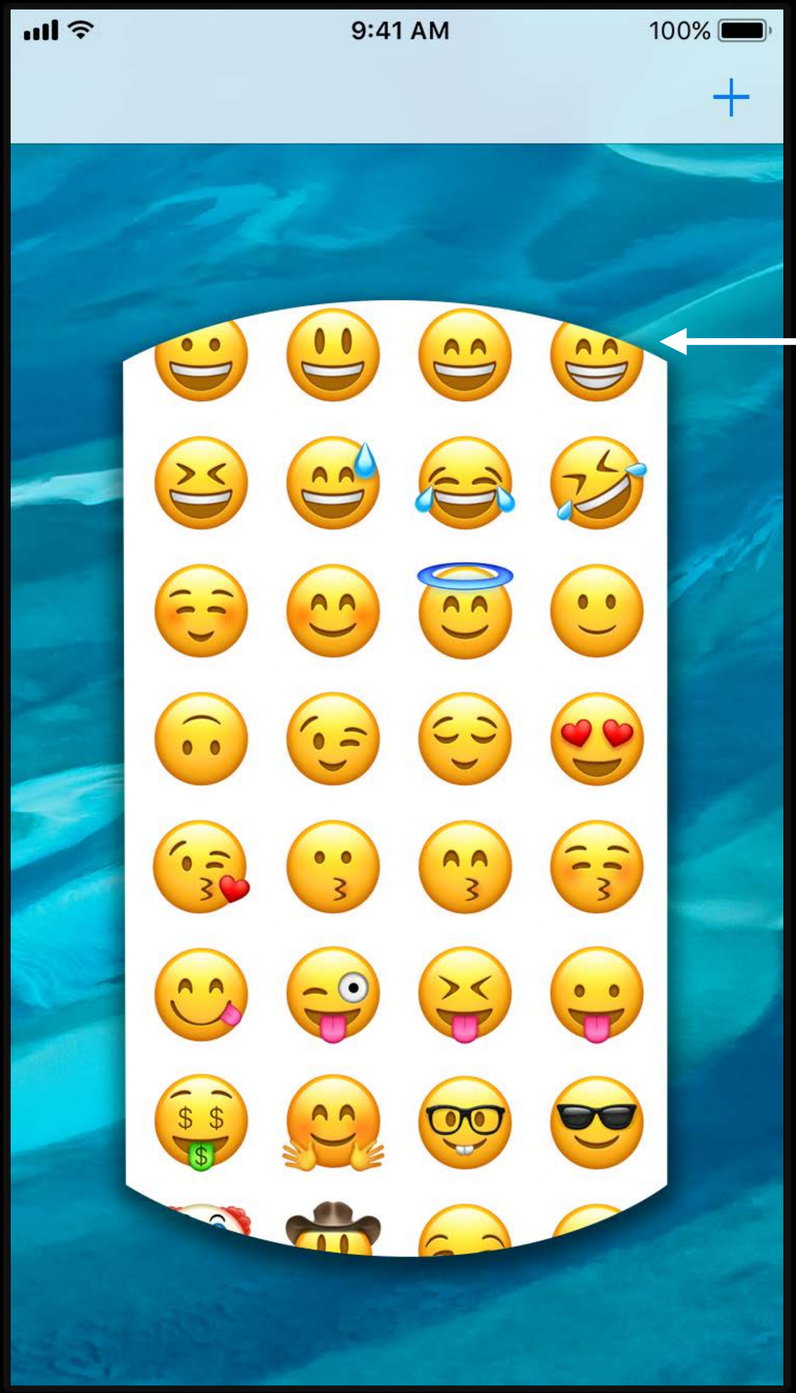
Set bounds/position directly instead


9:41 AM
100%

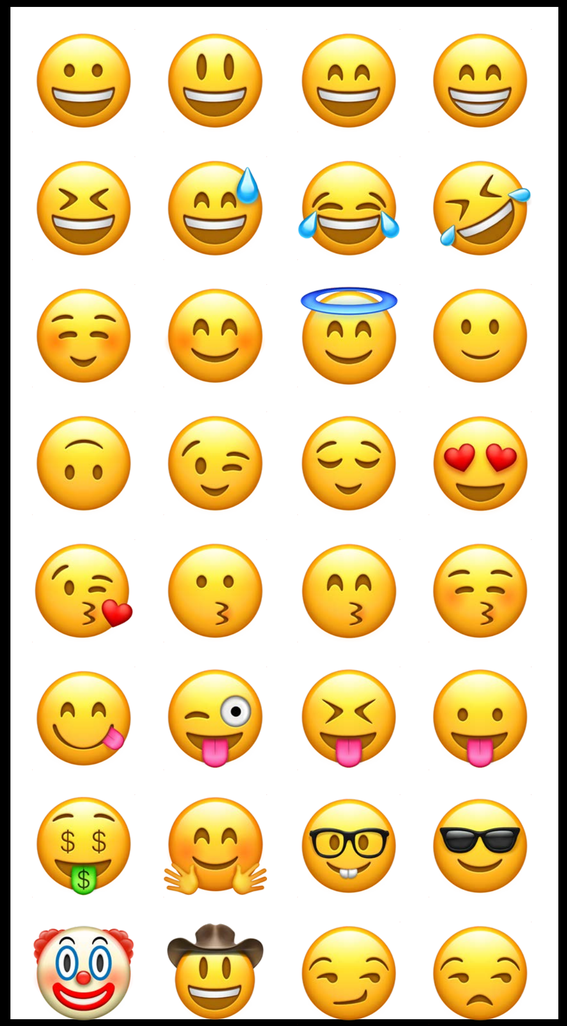


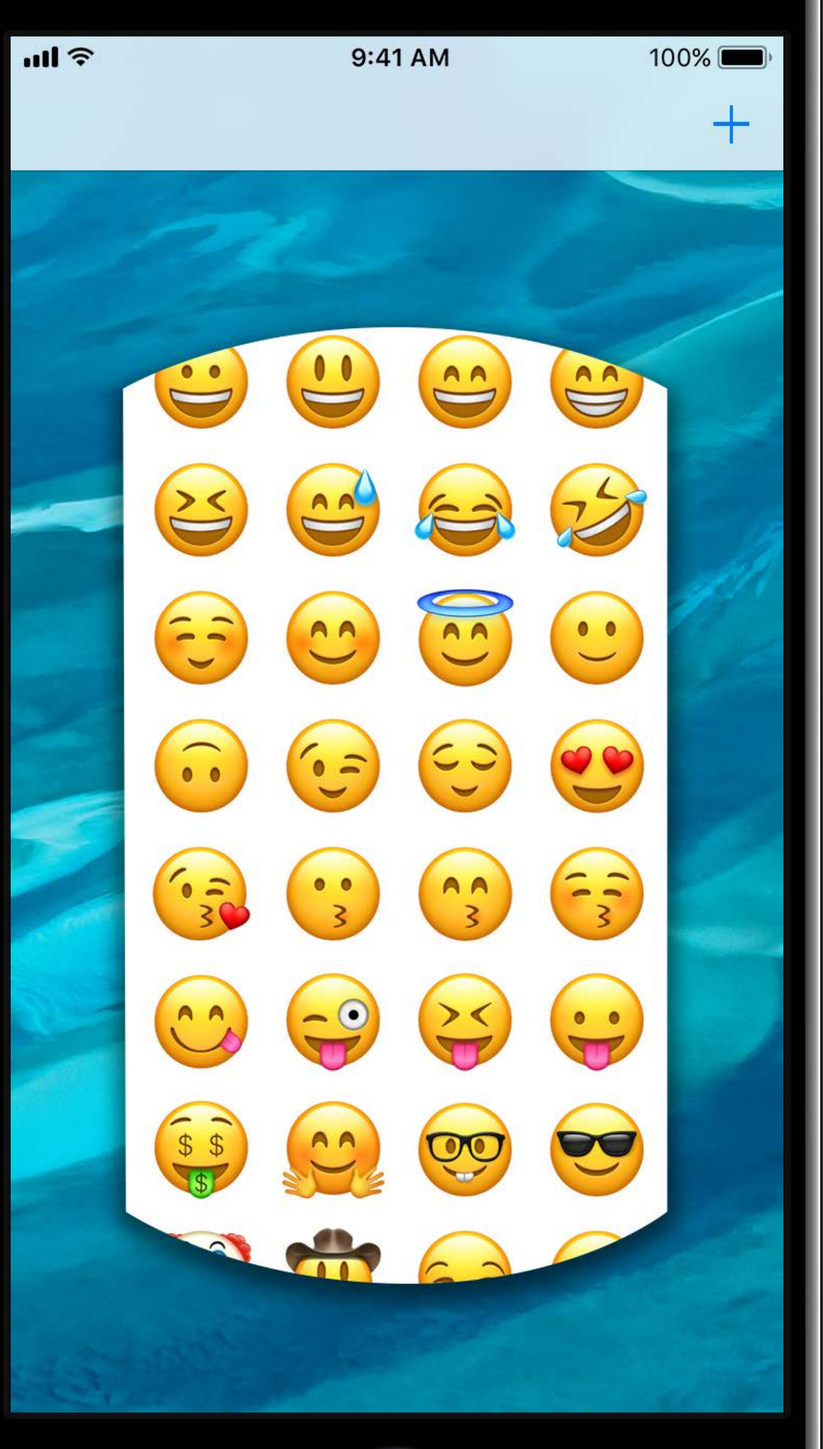
Mask

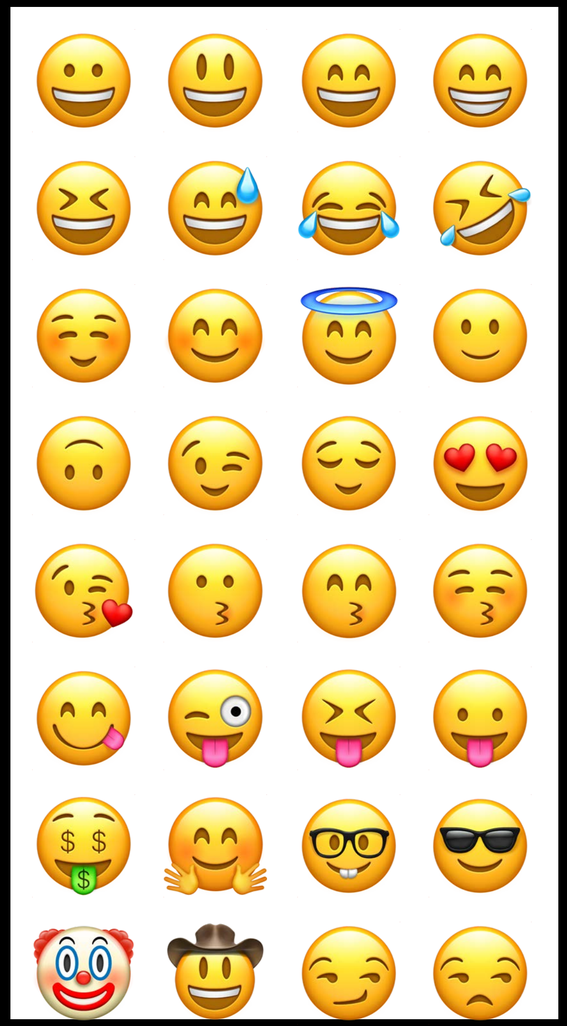

9:41 AM
100%
Mask
Shadow

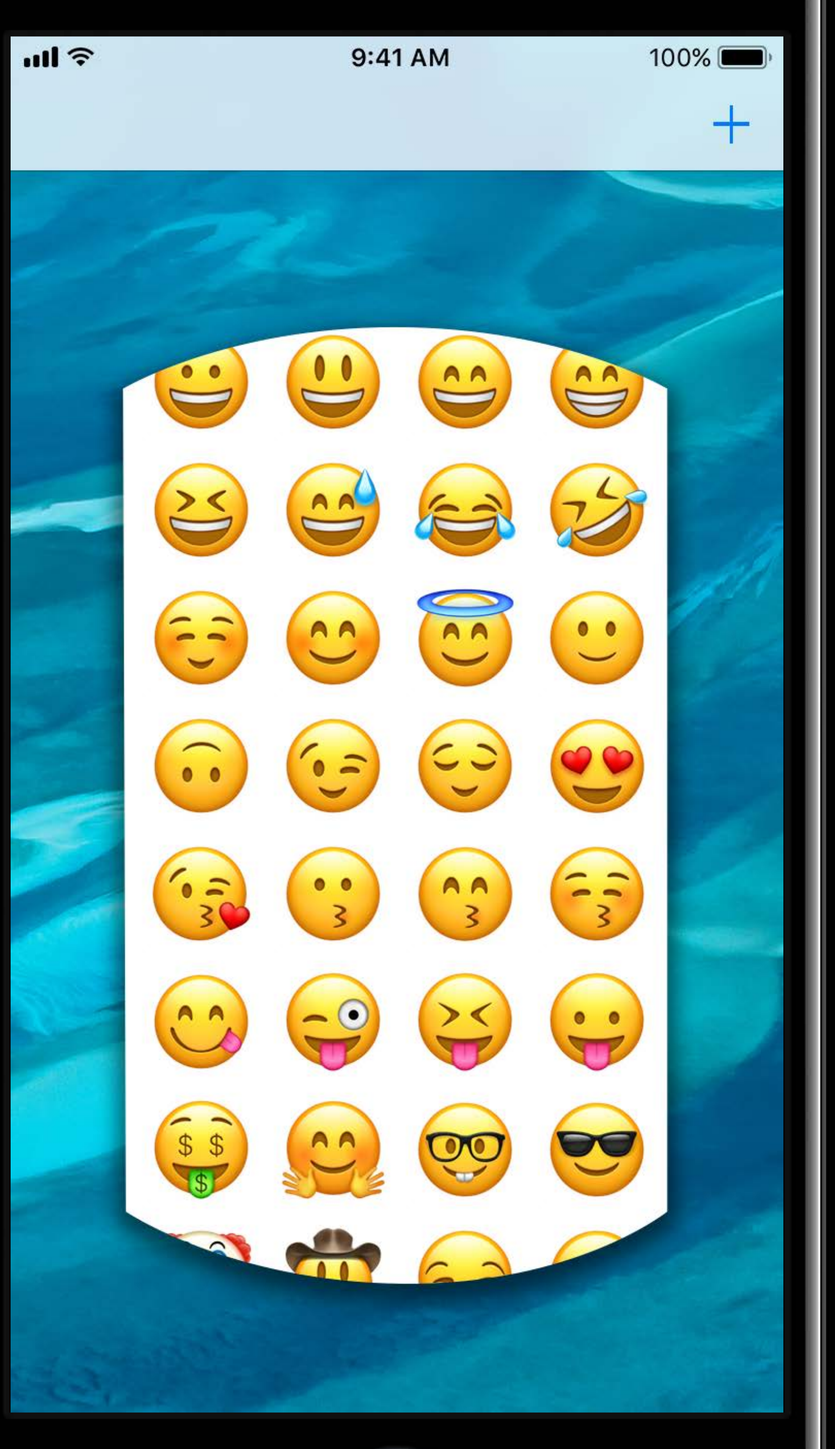


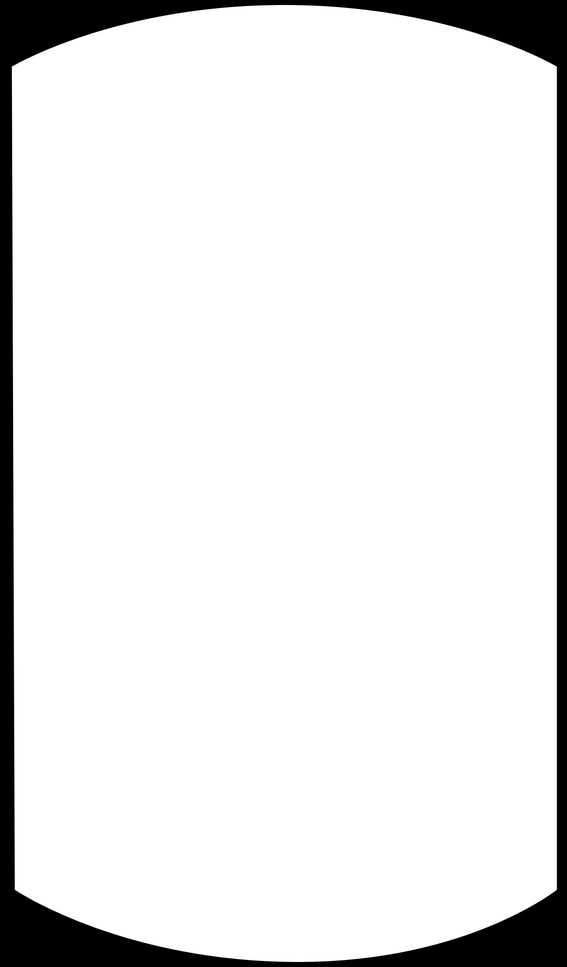
Mask




9:41 AM
100%

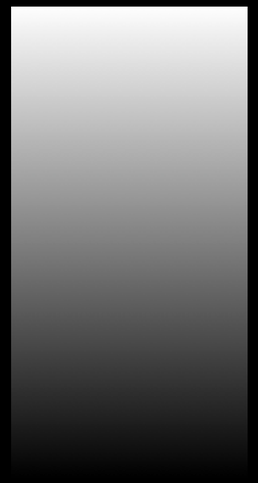



9:41 AM
100%

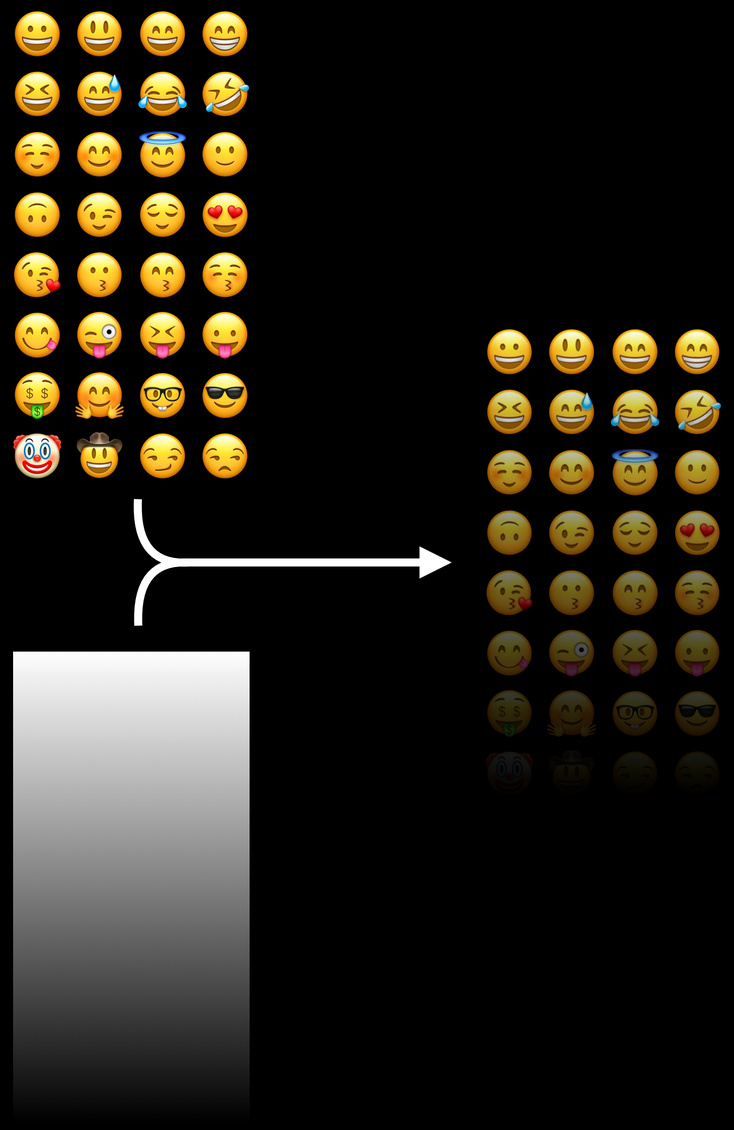



9:41 AM
100%

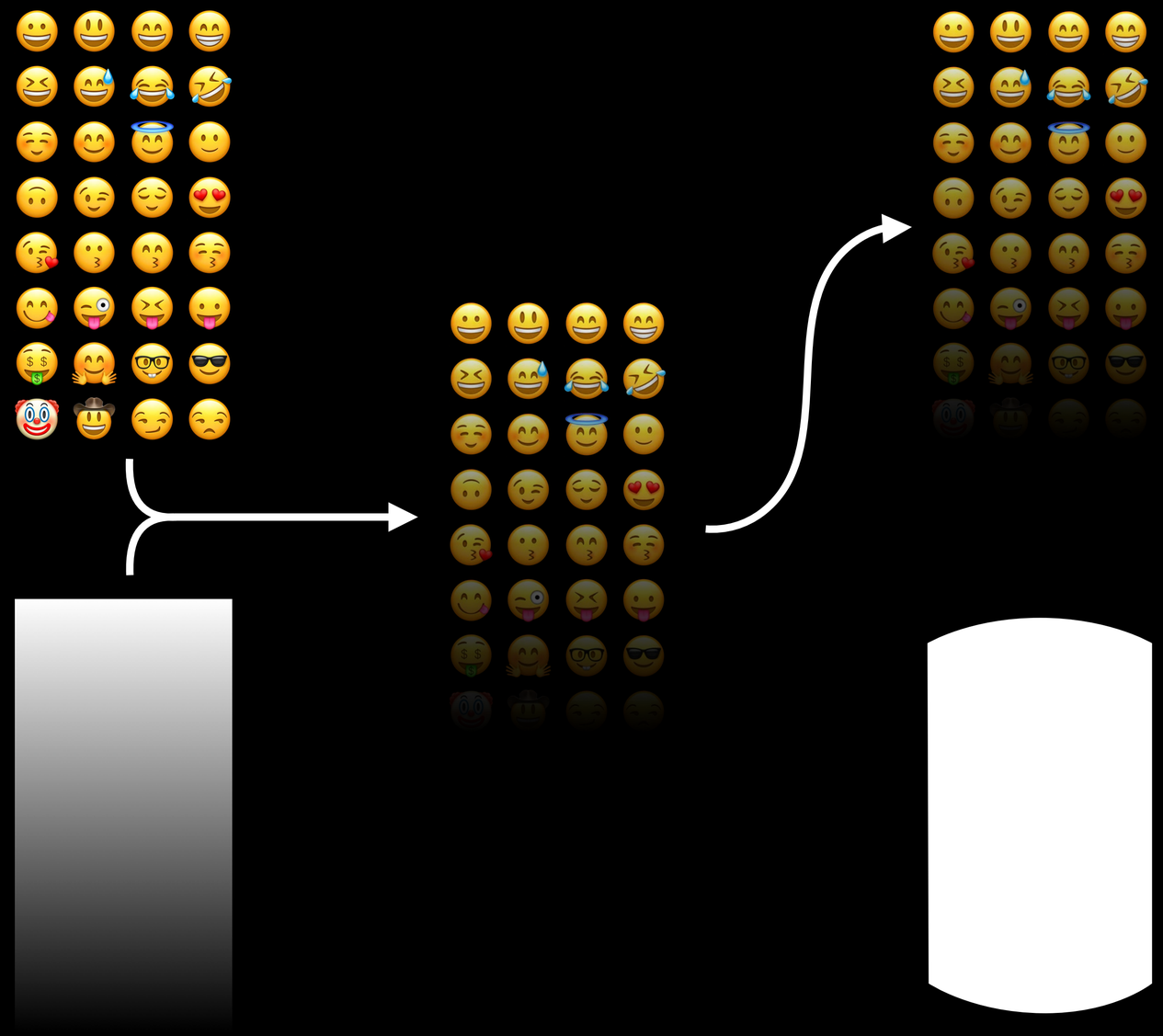
Multiple Masks



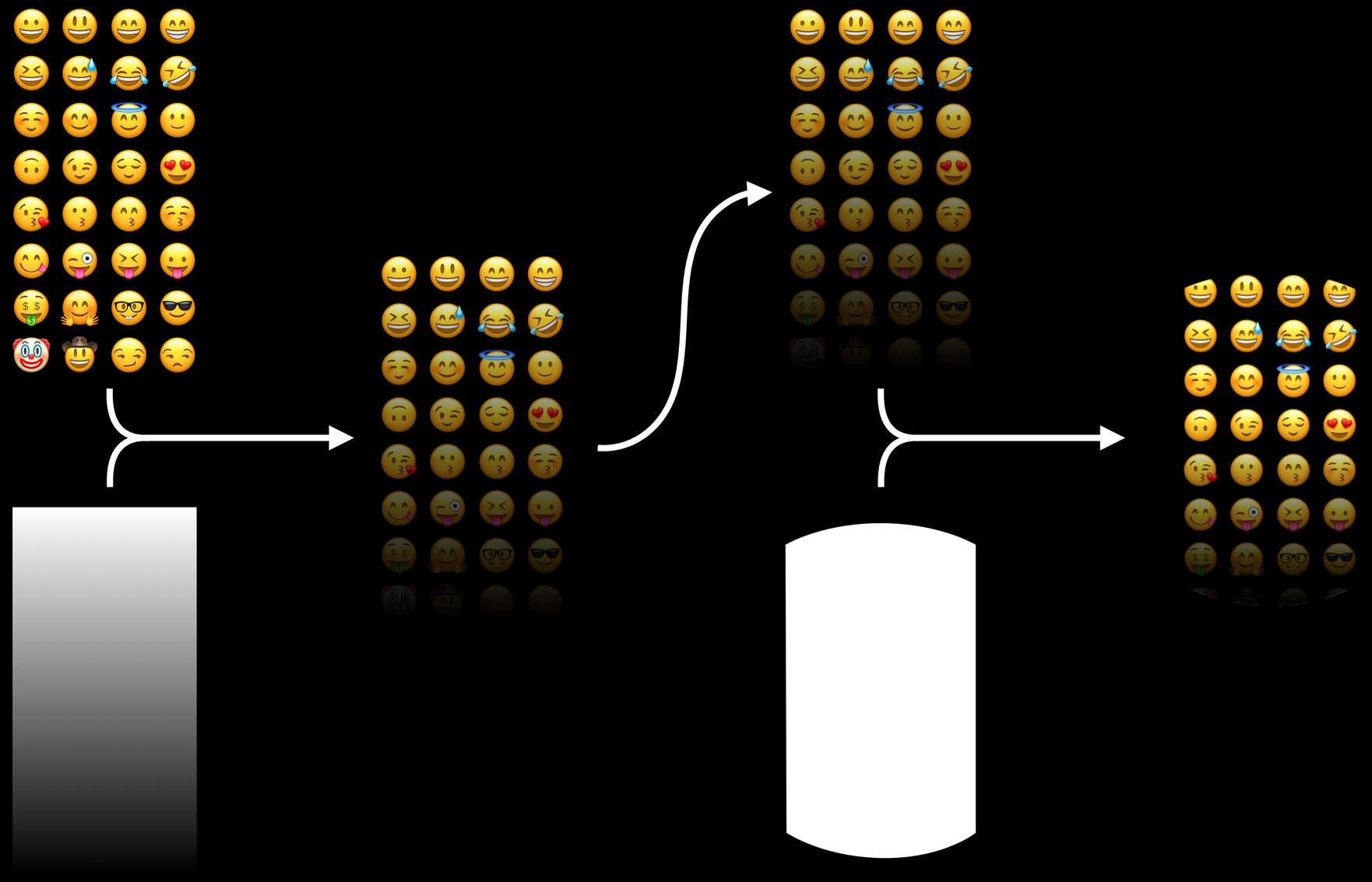
Multiple Masks



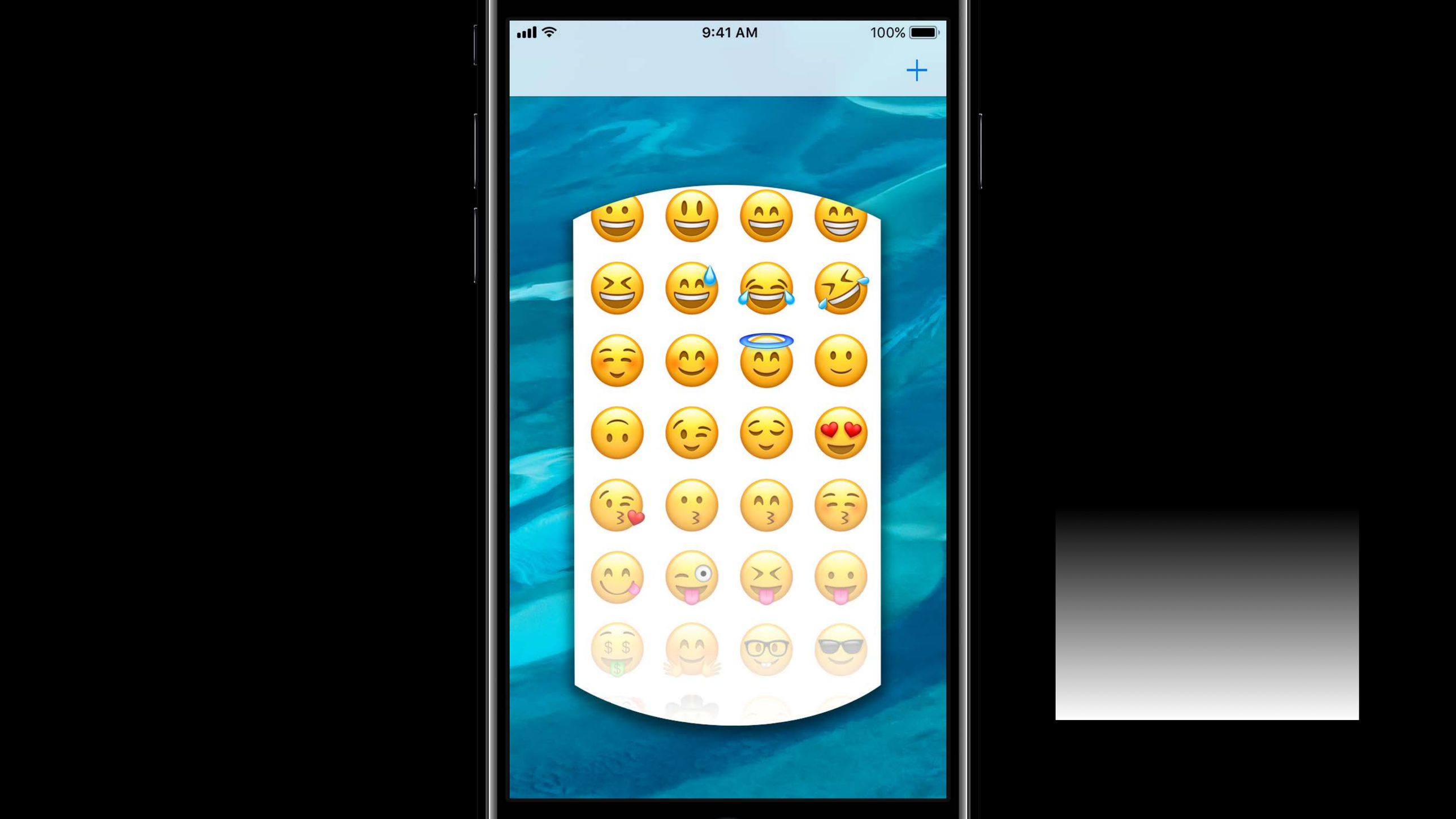
Multiple Masks



Multiple Masks




9:41 AM
100%

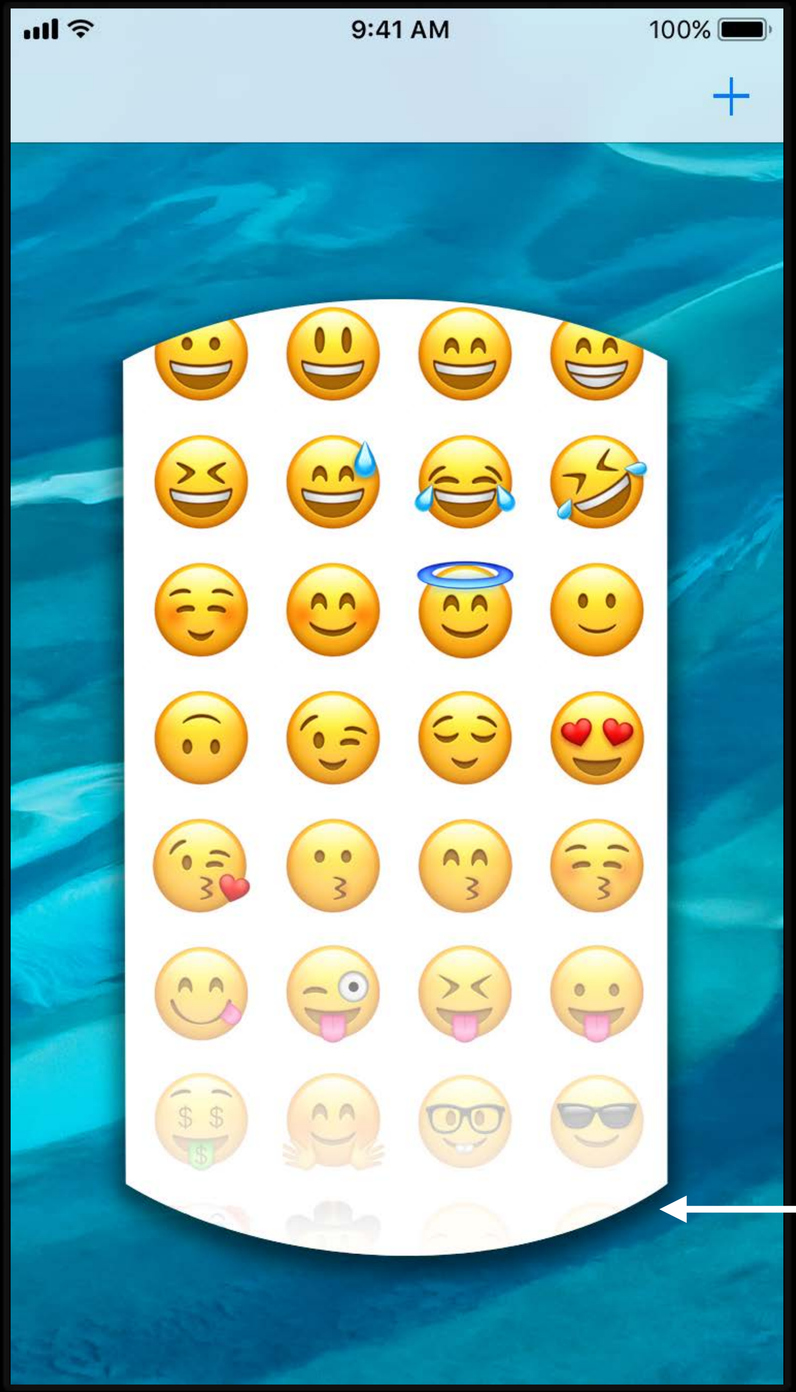

9:41 AM
100%

Masks

Each mask has a cost

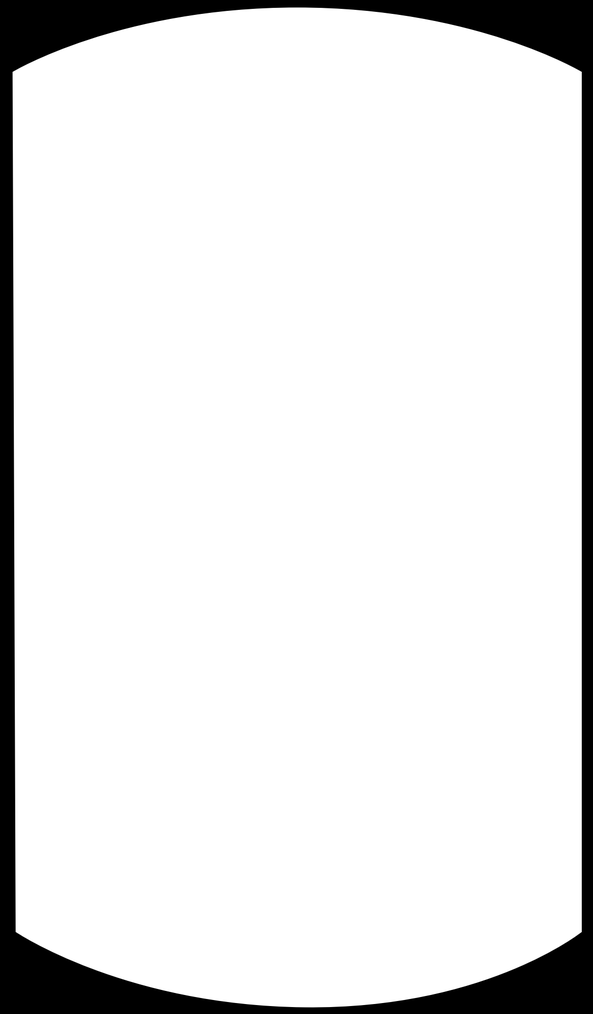
Layer size matters

Use shortcuts where possible


9:41 AM
100%
Shadow

Shadows

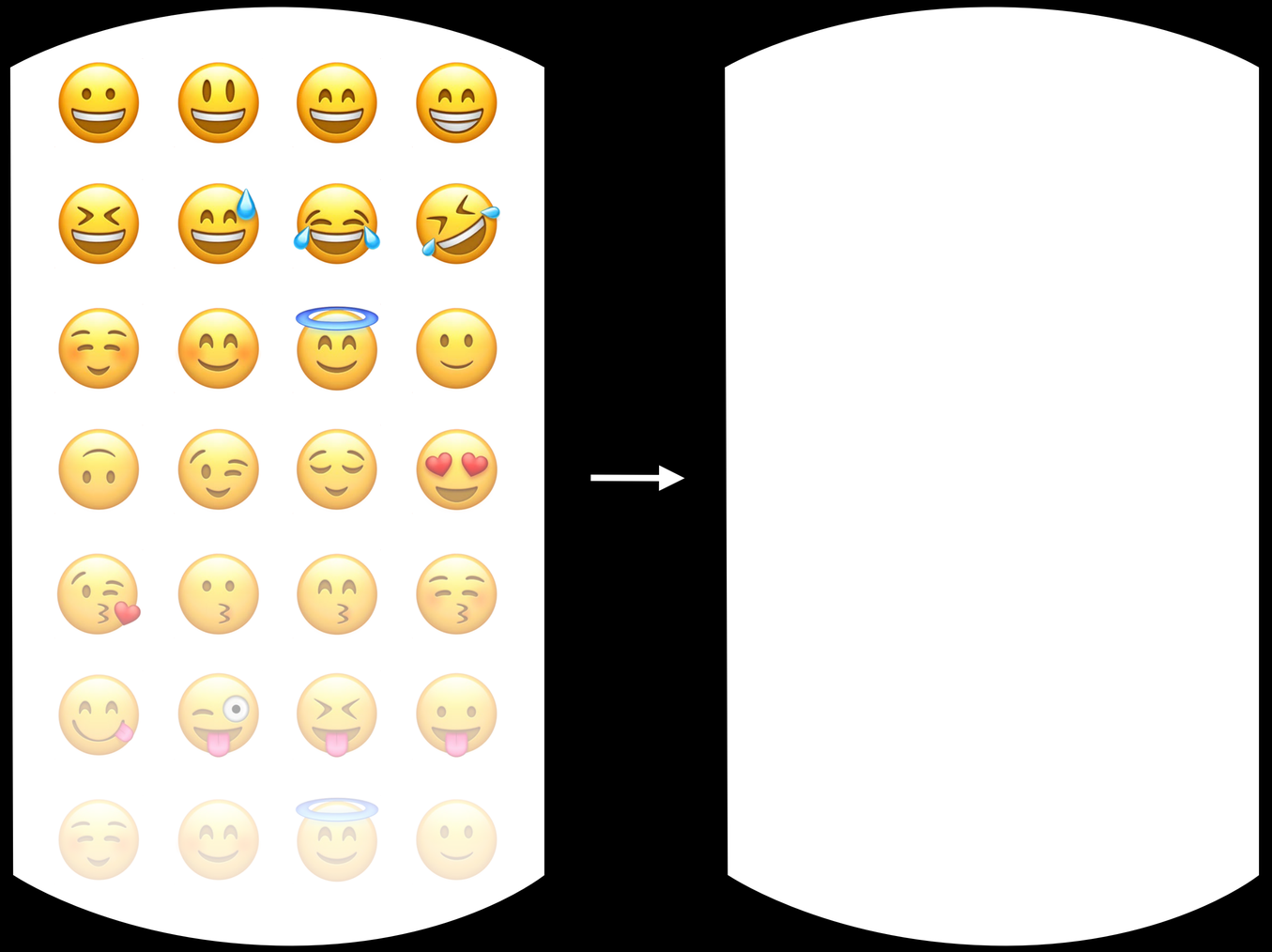
Shadows



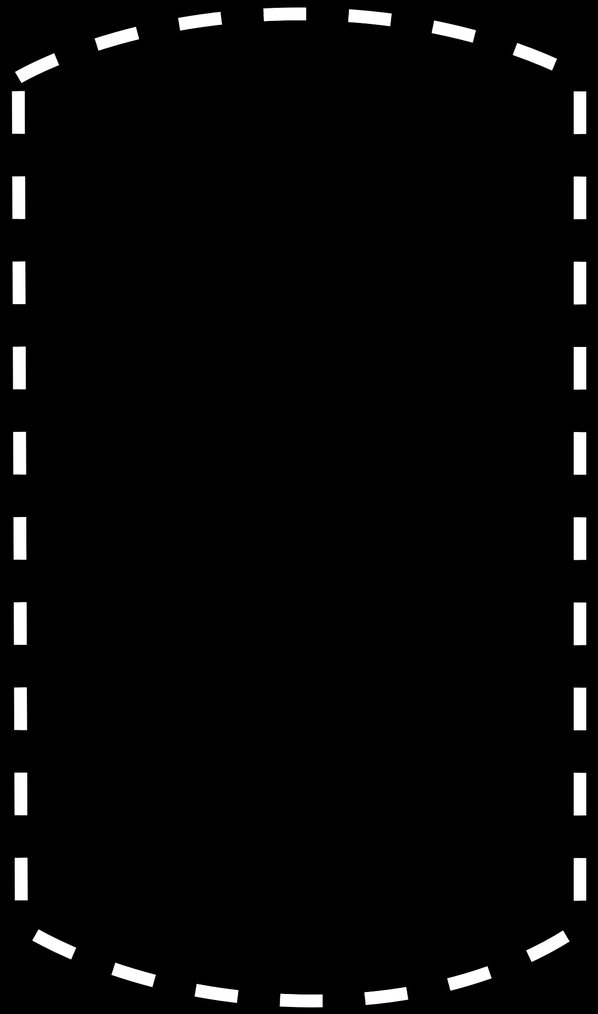
Shadows

Shadows

Shadows



Shadows



Shadows

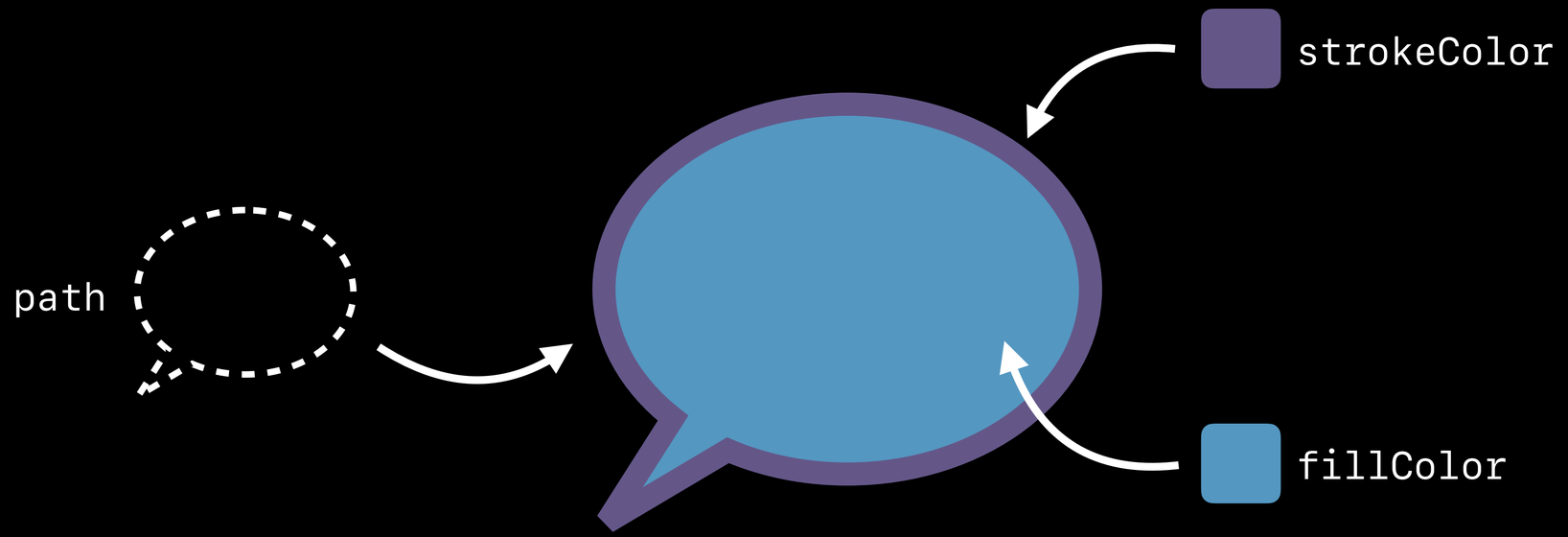
Shadows come from layer and all its sublayers

Blending is expensive

Use shadow paths

Tips and Tricks

Shape Layers

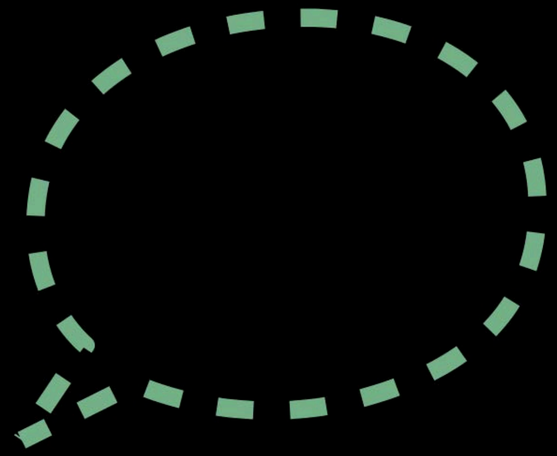


Shape Layers

Shape Layers

```
strokeStart / strokeEnd
```

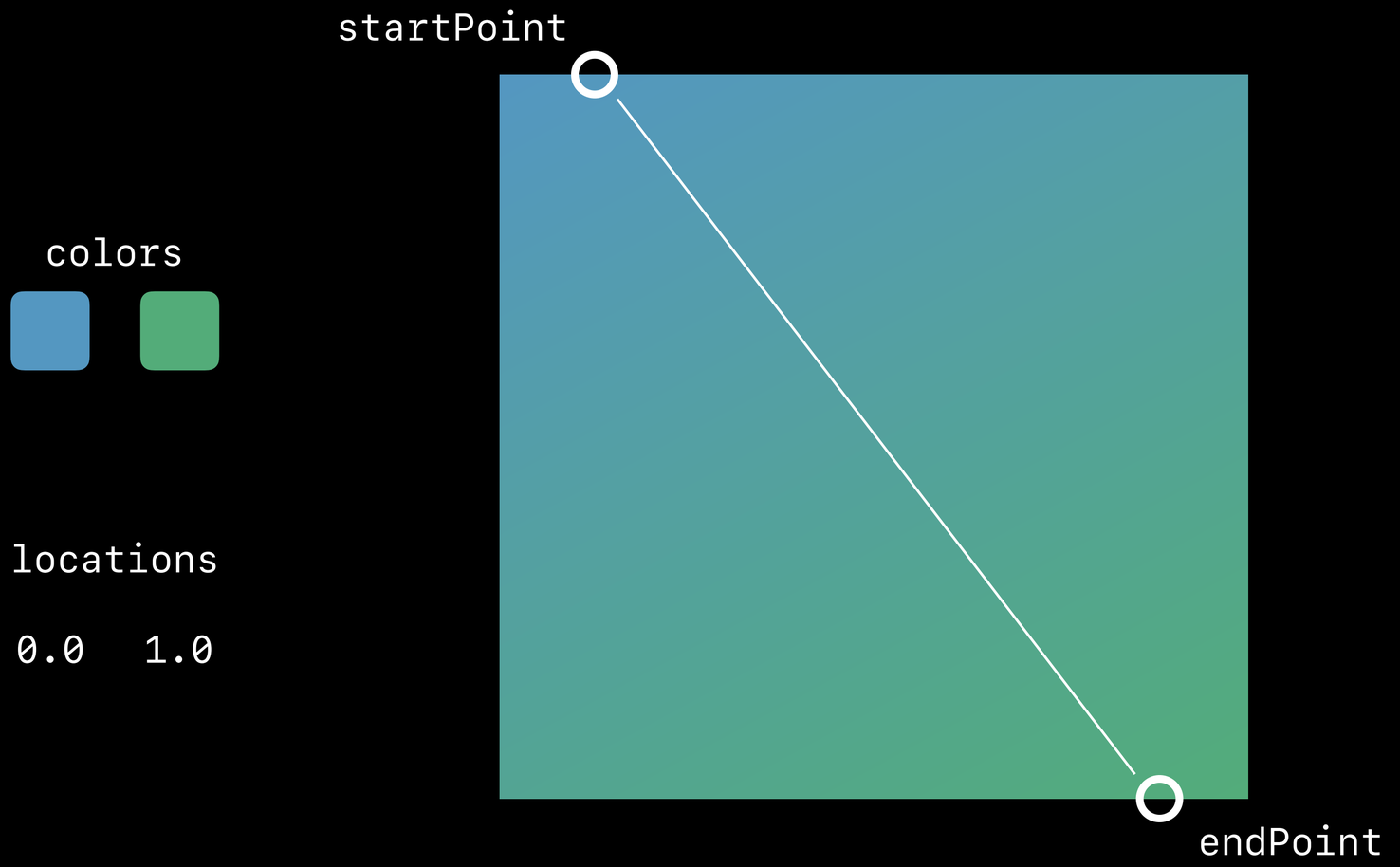
Shape Layers



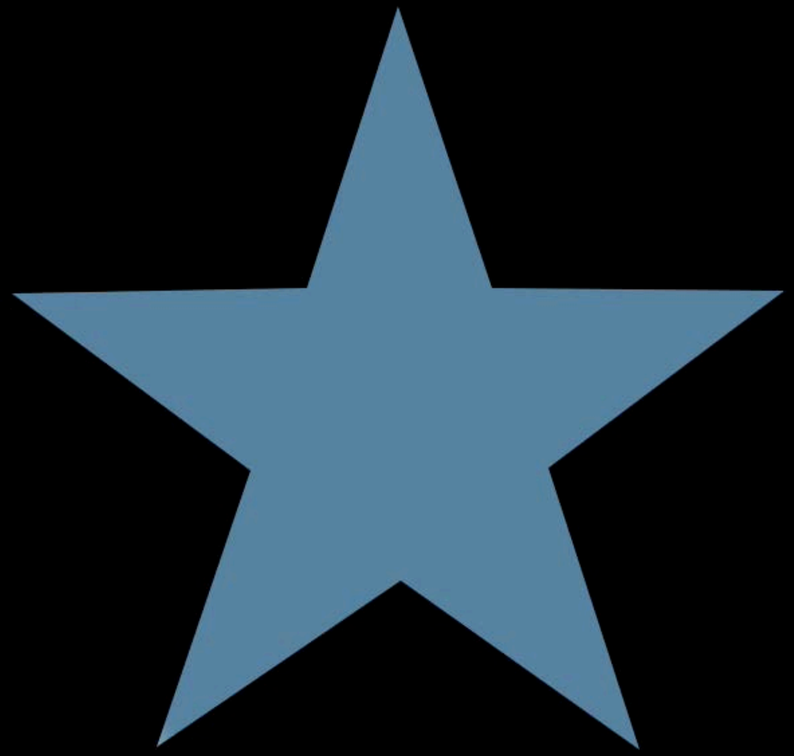
`strokeStart / strokeEnd`

`lineDashPhase`

Gradient Layers



Gradient Layers



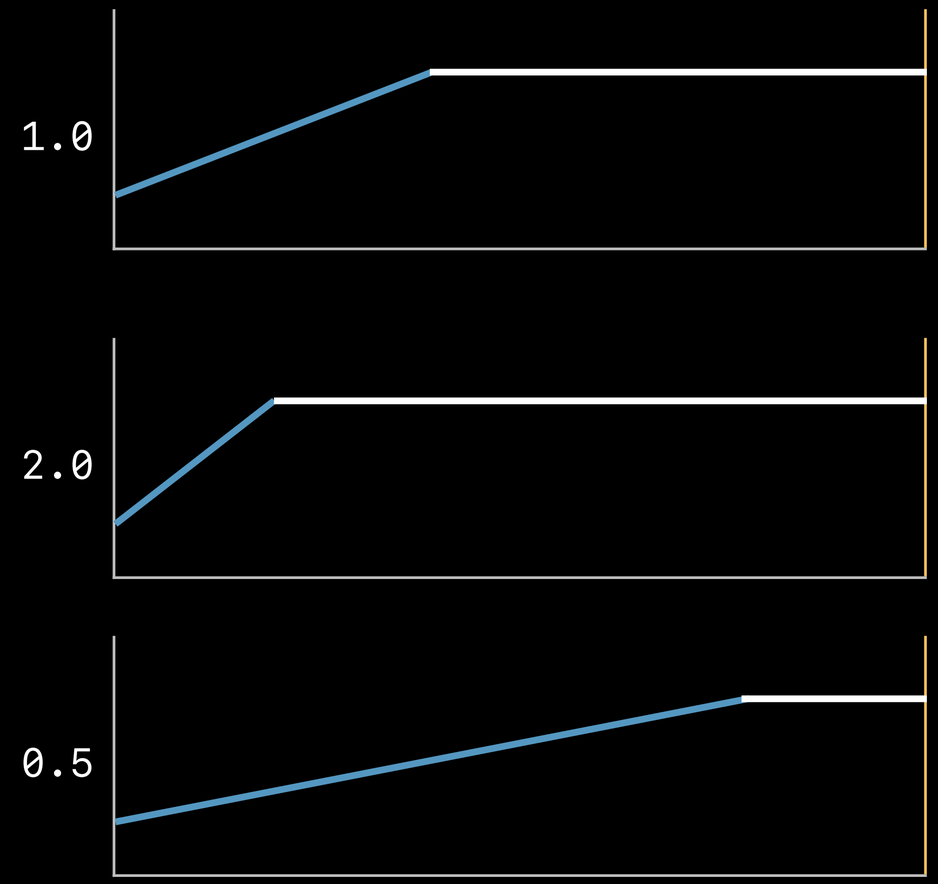
`startPoint/endPoint`

Layer Speed

Layers can have different time scales

Sublayers inherit time from superlayers

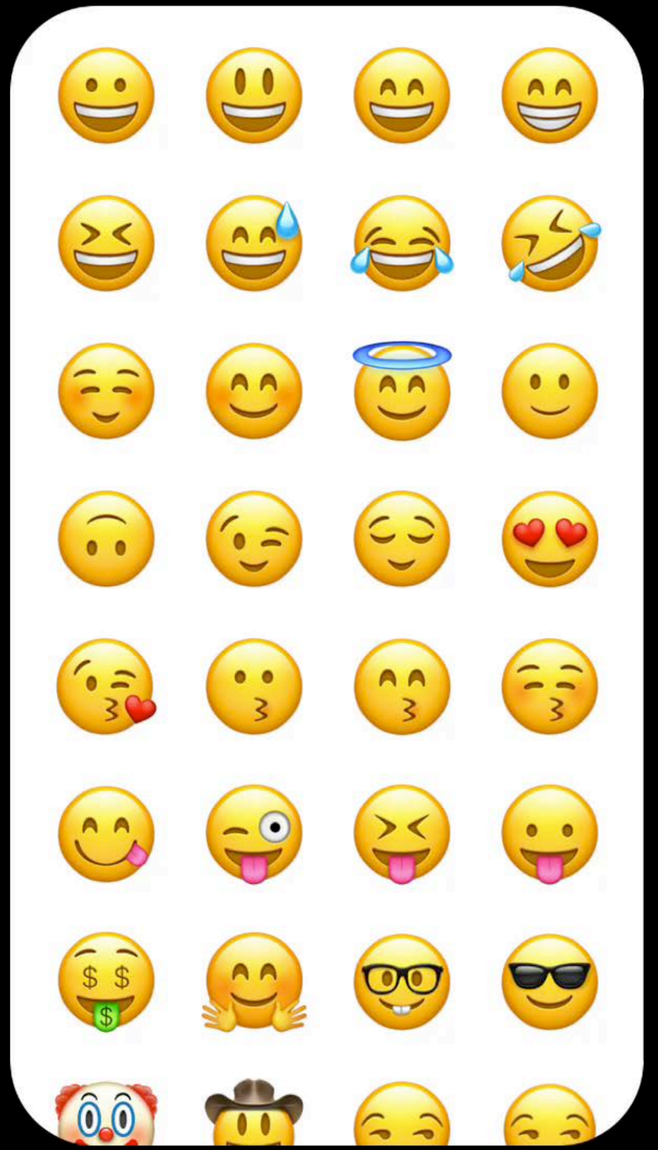
Animations use layer’s local time



Layer Speed

With `speed` at 0, animations pause

Interactive scrubbing with `timeOffset`



Demo

Noah Witherspoon

Summary

Broaden your toolbox

Know best practices

Experiment with the API

More Information

https://developer.apple.com/wwdc17/235

Related Sessions

Going Beyond 2D with SpriteKit	Executive Ballroom	Friday 9:00AM
Introducing Metal 2		WWDC 2017
SceneKit in Swift Playgrounds		WWDC 2017
Advances in Core Image: Filters, Metal, Vision, and More		WWDC 2017
Advanced Animations with UIKit		WWDC 2017

Labs

Metal 2 Lab	Technology Lab F	Fri 9:00AM–12:00PM
SpriteKit Lab	Technology Lab G	Fri 12:00PM–2:30PM
Cocoa Touch and Haptics Lab	Technology Lab C	Fri 12:00PM–1:50PM
Cocoa Lab	Technology Lab B	Fri 1:50PM–3:20PM


WWDC17