CSC 471 / 371 Mobile Application Development for iOS

Prof. Xiaoping Jia
chool of Computing, CDM
DePaul University
xjia@cdm.depaul.edu
@DePaulSWEng

Animations & Transitions

Outline

- View animations
- View transitions

DePaul University 3

View Animation

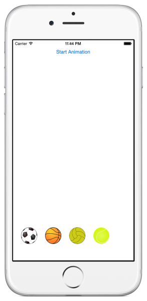
View Animation

- Many properties of the view objects are *animatable*.
  - center, frame, bounds
  - alpha, backgroundColor
  - transform
- Any changes to these properties can be animated automatically by the system
  - Through *interpolation*

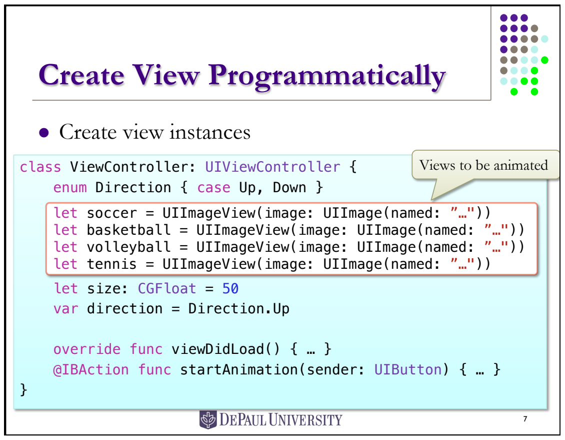
DePaul University 5

View Animation Demo

- Simple view animation
  - Movement of view object
  - Changing the frame property
- Animation parameters:
  - Duration, delay
  - Animation curve
- Spring animation



DePaul University 6



Create View Programmatically

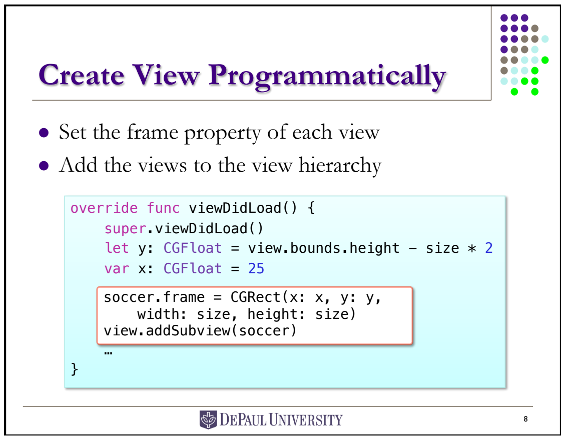
- Create view instances

```
class ViewController: UIViewController {
    enum Direction { case Up, Down }
    let soccer = UIImageView(image: UIImage(named: "…"))
    let basketball = UIImageView(image: UIImage(named: "…"))
    let volleyball = UIImageView(image: UIImage(named: "…"))
    let tennis = UIImageView(image: UIImage(named: "…"))
    let size: CGFloat = 50
    var direction = Direction.Up

    override func viewDidLoad() { … }
    @IBAction func startAnimation(sender: UIButton) { … }
}
```

Views to be animated

DePaul University 7



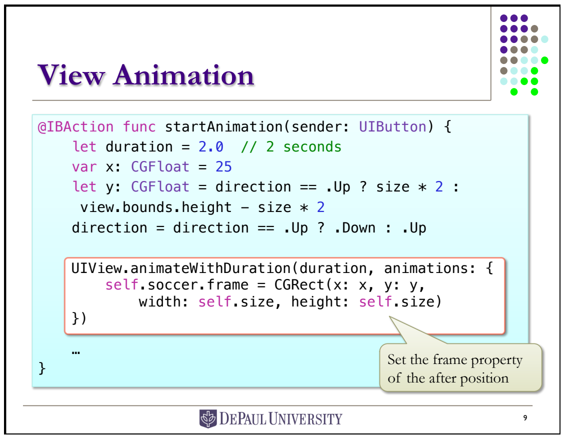
Create View Programmatically

- Set the frame property of each view
- Add the views to the view hierarchy

```
override func viewDidLoad() {
    super.viewDidLoad()
    let y: CGFloat = view.bounds.height - size * 2
    var x: CGFloat = 25

    soccer.frame = CGRect(x: x, y: y,
        width: size, height: size)
    view.addSubview(soccer)
    …
}
```

DePaul University 8



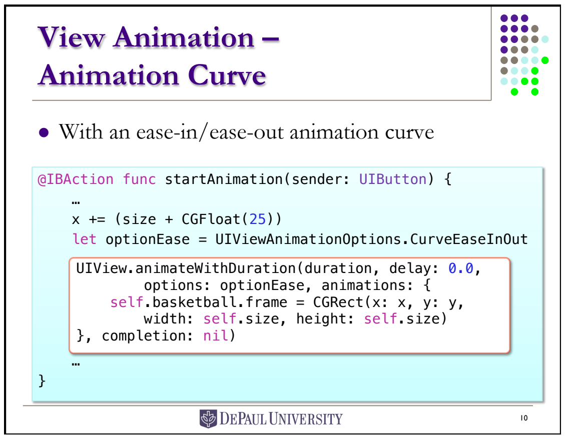
View Animation

```
@IBAction func startAnimation(sender: UIButton) {
    let duration = 2.0  // 2 seconds
    var x: CGFloat = 25
    let y: CGFloat = direction == .Up ? size * 2 :
     view.bounds.height - size * 2
    direction = direction == .Up ? .Down : .Up

    UIView.animateWithDuration(duration, animations: {
        self.soccer.frame = CGRect(x: x, y: y,
            width: self.size, height: self.size)
    })
    …
}
```

Set the frame property of the after position

DePaul University 9



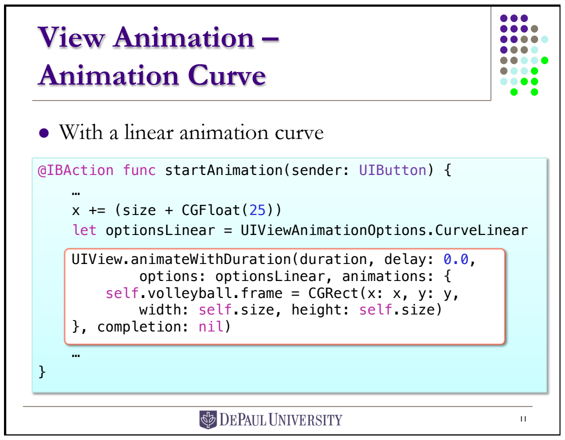
View Animation – Animation Curve

- With an ease-in/ease-out animation curve

```
@IBAction func startAnimation(sender: UIButton) {
    …
    x += (size + CGFloat(25))
    let optionEase = UIViewAnimationOptions.CurveEaseInOut

    UIView.animateWithDuration(duration, delay: 0.0,
            options: optionEase, animations: {
        self.basketball.frame = CGRect(x: x, y: y,
            width: self.size, height: self.size)
    }, completion: nil)
    …
}
```

DePaul University 10



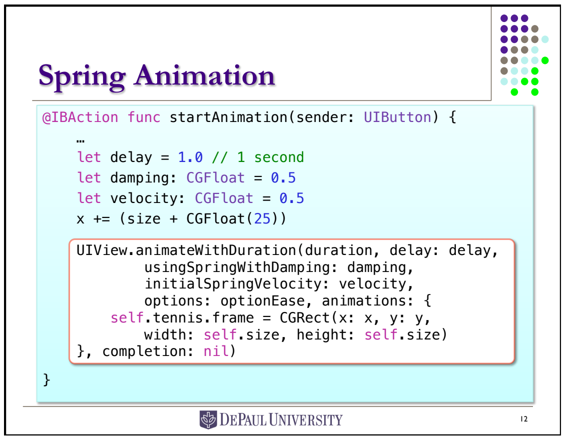
View Animation – Animation Curve

- With a linear animation curve

```
@IBAction func startAnimation(sender: UIButton) {
    …
    x += (size + CGFloat(25))
    let optionsLinear = UIViewAnimationOptions.CurveLinear

    UIView.animateWithDuration(duration, delay: 0.0,
            options: optionsLinear, animations: {
        self.volleyball.frame = CGRect(x: x, y: y,
            width: self.size, height: self.size)
    }, completion: nil)
    …
}
```

DePaul University 11



Spring Animation

```
@IBAction func startAnimation(sender: UIButton) {
    …
    let delay = 1.0 // 1 second
    let damping: CGFloat = 0.5
    let velocity: CGFloat = 0.5
    x += (size + CGFloat(25))

    UIView.animateWithDuration(duration, delay: delay,
            usingSpringWithDamping: damping,
            initialSpringVelocity: velocity,
            options: optionEase, animations: {
        self.tennis.frame = CGRect(x: x, y: y,
            width: self.size, height: self.size)
    }, completion: nil)
}
```

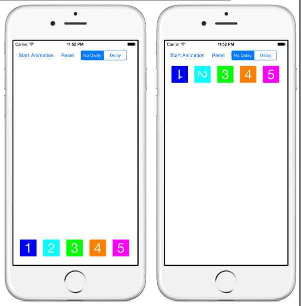
DePaul University 12

Key Frame Animation

- Allows an animation to be divided into several segments.
- The consecutive segments are divided by *key frames*.
- You may specify the relative start time and duration of each stage
- How to define key frame animations?
  - Insert/add key frames at the appropriate points of the animation

Key Frame Animation Demo

- Animating movements of numbered squares
- Move from bottom to top
- Rotate
  - ½ rotation
  - ¼ rotation
  - Full rotation, and animation



Key Frame Animation – Half Rotation

```
@IBAction func startAnimation(sender: UIButton) {
    let duration: Double = 2.0
    var delay: Double = 0.0
    let dt: Double =
        delayOptions.selectedSegmentIndex == 0 ? 0.0 : 2.0

    var x: CGFloat = 20

    UIView.animateWithDuration(duration, animations: {
        self.objects[0].frame = CGRect(x: x, y: 75,
            width: 50, height: 50)
        self.objects[0].transform =
            CGAffineTransformMakeRotation(CGFloat(M_PI))
    })
    …
```

A half rotation

Key Frame Animation – Quarter Rotation

```
@IBAction func startAnimation(sender: UIButton) {
    …
    x += 70
    delay += dt
    UIView.animateWithDuration(duration, delay: delay,
            options: UIViewAnimationOptions.CurveEaseInOut,
            animations: {
        self.objects[1].frame = CGRect(x: x, y: 75,
            width: 50, height: 50)
        self.objects[1].transform =
            CGAffineTransformMakeRotation(CGFloat(M_PI / 2))
    }, completion: nil)
    …
```

A quarter rotation

Key Frame Animation – Full Rotation, 1st Attempt

```
@IBAction func startAnimation(sender: UIButton) {
    …
    x += 70
    delay += dt
    UIView.animateWithDuration(duration, delay: delay,
            options: UIViewAnimationOptions.CurveEaseInOut,
            animations: {
        self.objects[2].frame = CGRect(x: x, y: 75,
            width: 50, height: 50)
        self.objects[2].transform =
            CGAffineTransformMakeRotation(CGFloat(2 * M_PI))
    }, completion: nil)
…
```

The start and the end positions are the same

A full rotation. But no animation.

Key Frame Animation – Full Rotation, 2nd Attempt

```
UIView.animateKeyframesWithDuration(duration,
delay: delay, options: .CalculationModeLinear,
animations: {
    self.objects[3].frame = CGRect(x: x, y: 75,
        width: 50, height: 50)
    UIView.addKeyframeWithRelativeStartTime(0,
            relativeDuration: 1/2, animations: {
        self.objects[3].transform =
            CGAffineTransformMakeRotation(CGFloat(M_PI))
    })
    UIView.addKeyframeWithRelativeStartTime(1/2,
            relativeDuration: 1/2, animations: {
        self.objects[3].transform =
            CGAffineTransformMakeRotation(CGFloat(2 * M_PI))
    })
}, completion: nil)
```

Key Frame Animation – Full Rotation, 3rd Attempt

- Divide into three segments.
- Each segment performs a 1/3 rotation.

```
UIView.animateKeyframesWithDuration(duration,
delay: delay, options: .CalculationModeLinear,
animations: {
    self.objects[4].frame = CGRect(x: x, y: 75,
        width: 50, height: 50)

    Add key frames with 1/3 , 2/3, and full rotations

}, completion: nil)
```

The Key Frames with 1/3 Rotation

```
UIView.addKeyframeWithRelativeStartTime(0,
        relativeDuration: 1/3, animations: {
    self.objects[4].transform =
        CGAffineTransformMakeRotation(CGFloat(M_PI * 2 / 3))
})
UIView.addKeyframeWithRelativeStartTime(1/3,
        relativeDuration: 1/3, animations: {
    self.objects[4].transform =
        CGAffineTransformMakeRotation(CGFloat(M_PI * 4 / 3))
})
UIView.addKeyframeWithRelativeStartTime(2/3,
        relativeDuration: 1/3, animations: {
    self.objects[4].transform =
        CGAffineTransformMakeRotation(CGFloat(M_PI * 2))
})
```

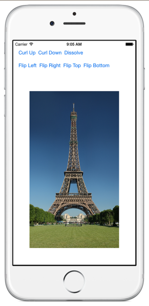
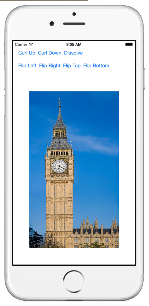
View Transition

View Transitions

- Replace one view with another in the view hierarchy
- Animate the transition from one to the other
- Transition animation options:
  - Curl up or down
  - Flip from left to right or from right to left
  - Flip from top or from bottom
  - Dissolve

View Transition Demo

- A container view contains one of the two views.
- The transition from one to the other is animated


Curl Up Curl Down Dissolve
Flip Left Flip Right Flip Top Flip Bottom

Curl Up Curl Down Dissolve
Flip Left Flip Right Flip Top Flip Bottom

The Views

- Create a container view and two views

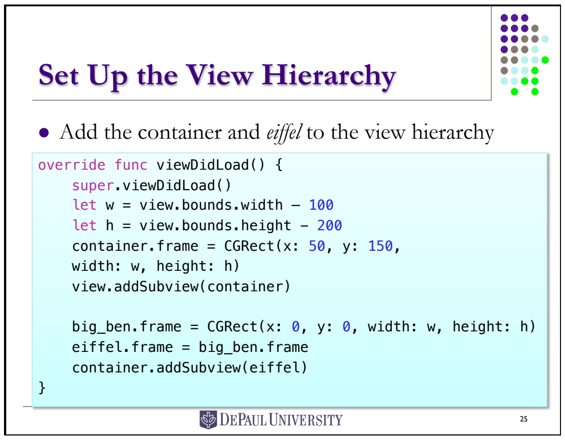
```
class ViewController: UIViewController {

    let container = UIView()
    let big_ben = UIImageView(image: UIImage(named: "…"))
    let eiffel = UIImageView(image: UIImage(named: "…"))

    override func viewDidLoad() { … }

    @IBAction func changeView(sender: UIButton) { … }

}
```



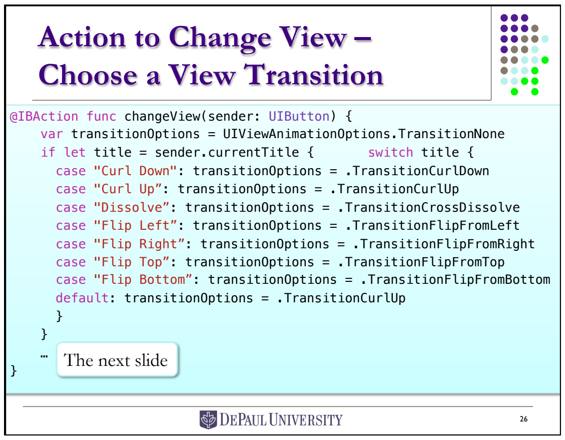
Set Up the View Hierarchy

- Add the container and *eiffel* to the view hierarchy

```
override func viewDidLoad() {
    super.viewDidLoad()
    let w = view.bounds.width - 100
    let h = view.bounds.height - 200
    container.frame = CGRect(x: 50, y: 150,
    width: w, height: h)
    view.addSubview(container)

    big_ben.frame = CGRect(x: 0, y: 0, width: w, height: h)
    eiffel.frame = big_ben.frame
    container.addSubview(eiffel)
}
```

DEPAUL UNIVERSITY 25

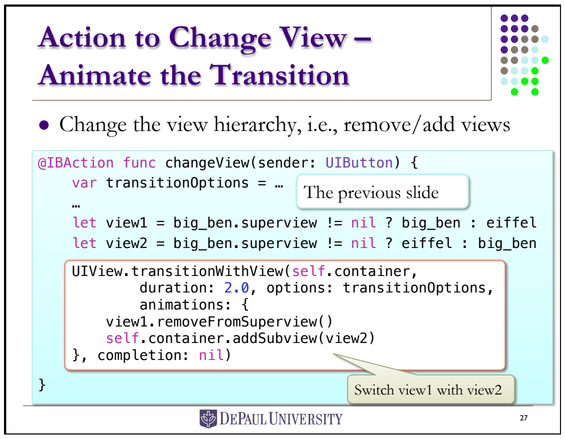


Action to Change View – Choose a View Transition

```
@IBAction func changeView(sender: UIButton) {
    var transitionOptions = UIViewAnimationOptions.TransitionNone
    if let title = sender.currentTitle {       switch title {
      case "Curl Down": transitionOptions = .TransitionCurlDown
      case "Curl Up”: transitionOptions = .TransitionCurlUp
      case "Dissolve”: transitionOptions = .TransitionCrossDissolve
      case "Flip Left”: transitionOptions = .TransitionFlipFromLeft
      case "Flip Right”: transitionOptions = .TransitionFlipFromRight
      case "Flip Top”: transitionOptions = .TransitionFlipFromTop
      case "Flip Bottom”: transitionOptions = .TransitionFlipFromBottom
      default: transitionOptions = .TransitionCurlUp
      }
    }
    …
}
```

The next slide

DEPAUL UNIVERSITY 26



Action to Change View – Animate the Transition

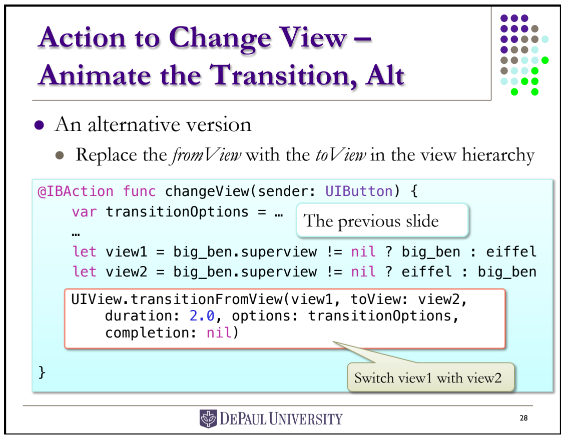
- Change the view hierarchy, i.e., remove/add views

```
@IBAction func changeView(sender: UIButton) {
    var transitionOptions = …
    …
    let view1 = big_ben.superview != nil ? big_ben : eiffel
    let view2 = big_ben.superview != nil ? eiffel : big_ben
    UIView.transitionWithView(self.container,
            duration: 2.0, options: transitionOptions,
            animations: {
        view1.removeFromSuperview()
        self.container.addSubview(view2)
    }, completion: nil)
}
```

The previous slide

Switch view1 with view2

DEPAUL UNIVERSITY 27



Action to Change View – Animate the Transition, Alt

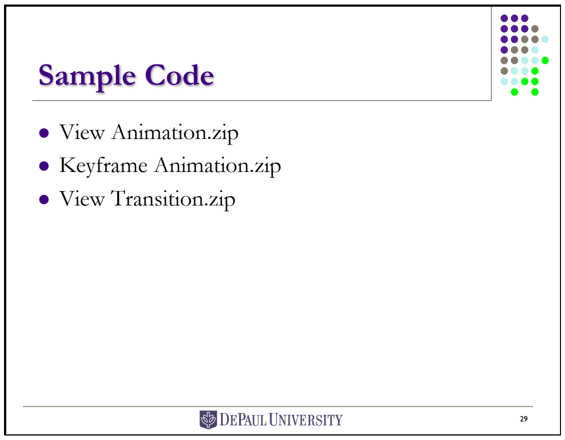
- An alternative version
  - Replace the *fromView* with the *toView* in the view hierarchy

```
@IBAction func changeView(sender: UIButton) {
    var transitionOptions = …
    …
    let view1 = big_ben.superview != nil ? big_ben : eiffel
    let view2 = big_ben.superview != nil ? eiffel : big_ben
    UIView.transitionFromView(view1, toView: view2,
        duration: 2.0, options: transitionOptions,
        completion: nil)
}
```

The previous slide

Switch view1 with view2

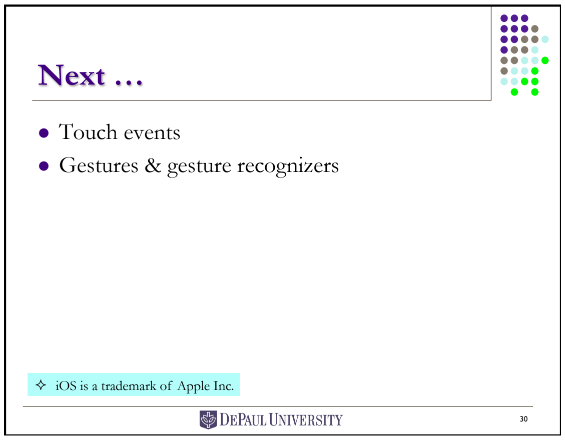
DEPAUL UNIVERSITY 28



Sample Code

- View Animation.zip
- Keyframe Animation.zip
- View Transition.zip

DEPAUL UNIVERSITY 29



Next …

- Touch events
- Gestures & gesture recognizers

✧ iOS is a trademark of Apple Inc.

DEPAUL UNIVERSITY 30