ANIMATION

SLAMINATE

- and why?

A BRIEF HISTORY OF ANIMATION APIs ON iOS

iOS 2

UIKit

```
[UIView beginAnimations:@"alpha" context:nil];

[UIView setAnimationDidStopSelector:@selector(animationDidStop:context:)];
[UIView setAnimationDelegate:self];
[UIView setAnimationDuration:0.3];
[UIView setAnimationCurve:UIViewAnimationCurveEaseOut];

self.alpha = .0;

[UIView commitAnimations];
```

Core Animation

```objc
CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"alpha"];

animation.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];
animation.duration = 0.3;
animation.delegate = self;
animation.keyPath = @"opacity";
animation.fromValue = [NSNumber numberWithDouble:1.0];
animation.toValue = [NSNumber numberWithDouble:0.0];

[self.view.layer addAnimation:animation forKey:@"alpha"];
```

iOS 4

iOS 4

UIKit

```
[UIView animateWithDuration:0.3
                      delay:0.0
                    options:UIViewAnimationOptionCurveEaseOut
                 animations:^{
                     self.alpha = 0.0;
                 } completion:^(BOOL finished) {
                     // Some completion action.
                 }];
```

Core Animation

```
CABasicAnimation *animation = [CABasicAnimation animationWithKeyPath:@"opacity"];

animation.timingFunction = [CAMediaTimingFunction functionWithName:kCAMediaTimingFunctionEaseOut];
animation.duration = .3;
animation.delegate = self;
animation.keyPath = @"opacity";
animation.fromValue = [NSNumber numberWithDouble:1.0];
animation.toValue = [NSNumber numberWithDouble:0.0];

[self.view.layer addAnimation:animation forKey:@"opacity"];
```

WHAT IS MISSING?

```
UIView.animateWithDuration(
    2.0,
    delay: 0.0,
    options: .CurveEaseOut,
    animations: {
        self.aConstraint.constant = 400.0
    },
    completion: nil
)
```

```
UIView.animateWithDuration(
    2.0,
    delay: 0.0,
    options: .CurveEaseOut,
    animations: {
        self.aConstraint.constant = 400.0
        self.view.updateConstraintsIfNeeded()
        self.view.layoutIfNeeded()
    },
    completion: nil
)
```

```
UIView.animateWithDuration(
    2.0,
    delay: 0.0,
    options: .CurveEaseOut,
    animations: {
        self.view.removeConstraint(aConstraint)
        self.view.addConstraint(aNewConstraint)
    },
    completion: nil
)
```

SLAMINATE

WHAT IS IT?

- it's a lightweight framework designed
to make animations easy.

- it's a lightweight framework designed to make **advanced** animations easy.

- it's a lightweight framework designed
to make **advanced** animations **very** easy.

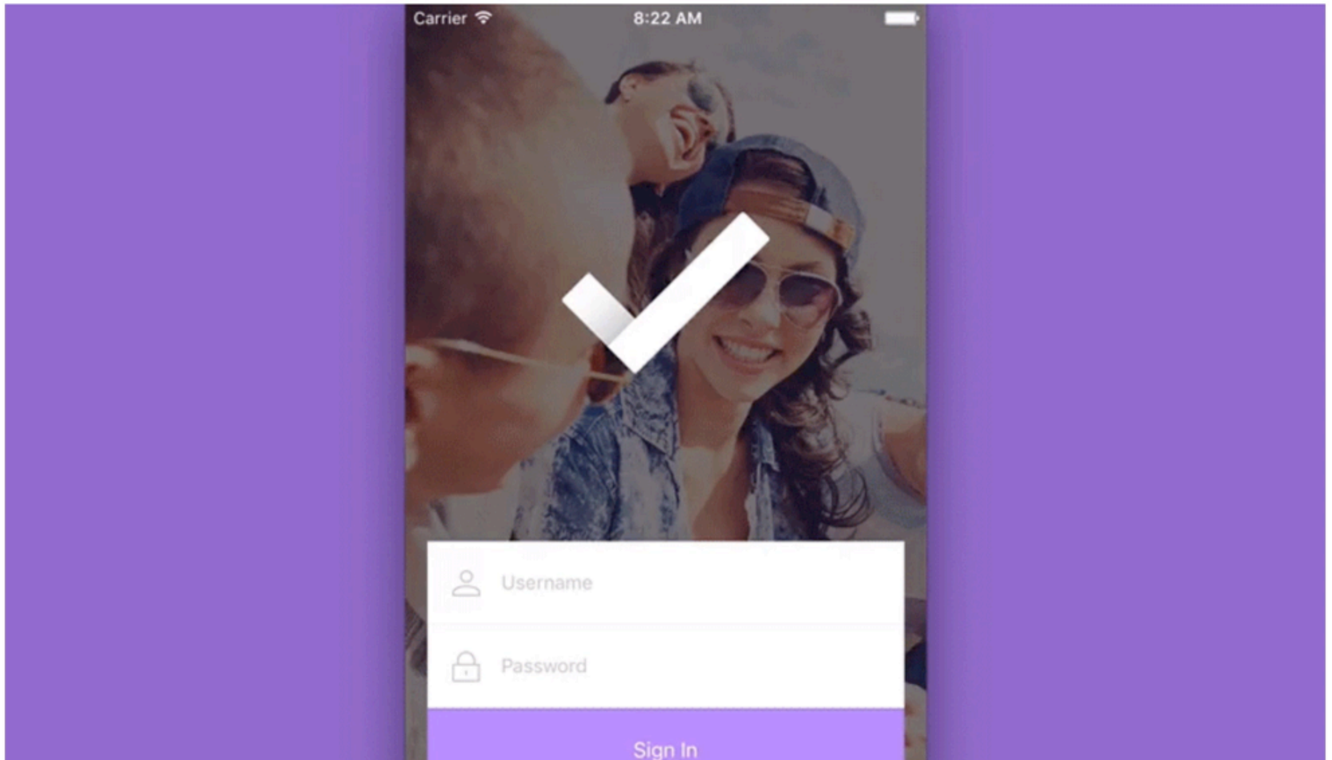
- it combines the **complexity of Core Animation**

- it combines the **complexity of Core Animation**,
with an **easy-to-use API**.

README.md

IBAnimatable

Design and prototype UI, interaction, navigation, transition and animation for App Store ready Apps in Interface Builder with IBAnimatable.



github.com/JakeLin/IBAnimatable

README.md

IBAnimatable

Design and prototype UI, interaction, navigation, transition and animation for App Store ready Apps in Interface Builder with IBAnimatable.

- it combines the **complexity of Core Animation**,
with an **easy-to-use API**.

```
Slaminate(
    duration: 2.0,
    curve: Curve.easeOut,
    animation: {
        self.view.alpha = 0.0
    }
)
```

```
Slaminate(
    duration: 2.0,
    curve: Curve.easeOut,
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
    }
)
```

```
Slaminate(
    duration: 2.0,
    curve: Curve.easeOut,
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
    }
)
```

```
Slaminate(
    duration: 2.0,
    curve: Curve.easeOut,
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
        self.view.addConstraint(aNewConstraint)
    }
)
```

```
Slaminate(
    duration: 2.0,
    curve: Curve.easeOut,
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
        self.view.addConstraint(aNewConstraint)
    }
)
.completed({ (animation)  in
    print("Complete")
})
```

```
Slaminate(
    duration: 2.0,
    curve: Curve.easeOut,
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
        self.view.addConstraint(aNewConstraint)
    }
)
.completed({ (animation)  in
    print("Complete")
})
.delayed(2.0)
```

```
Slaminate(
    duration: 2.0,
    curve: Curve.easeOut,
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
        self.view.addConstraint(aNewConstraint)
    }
)
.completed({ (animation)  in
    print("Complete")
})
.delayed(2.0)
.started({ _ in
    print("Started")
})
```

```
Slaminate(
    duration: 2.0,
    curve: Curve.easeOutBack,
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
        self.view.addConstraint(aNewConstraint)
    }
)
.completed({ (animation)  in
    print("Complete")
})
.delayed(2.0)
.started({ _ in
    print("Started")
})
```

```
Slaminate(
    duration: 2.0,
    curve: Curve.easeOutBounce,
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
        self.view.addConstraint(aNewConstraint)
    }
)
.completed({ (animation)  in
    print("Complete")
})
.delayed(2.0)
.started({ _ in
    print("Started")
})
```

```
Slaminate(
    duration: 2.0,
    curve: Curve.easeOutElastic,
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
        self.view.addConstraint(aNewConstraint)
    }
)
.completed({ (animation)  in
    print("Complete")
})
.delayed(2.0)
.started({ _ in
    print("Started")
})
```

```
Slaminate(
    duration: 2.0,
    curve: Curve(transform: { pow($0, 2) }),
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
        self.view.addConstraint(aNewConstraint)
    }
)
.completed({ (animation)  in
    print("Complete")
})
.delayed(2.0)
.started({ _ in
    print("Started")
})
```

```
Slaminate(
    duration: 2.0,
    curve: Curve(transform: { pow($0, 2) }),
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
        self.view.addConstraint(aNewConstraint)
    }
)
.completed({ (animation)  in
    print("Complete")
})
.delayed(2.0)
.started({ _ in
    print("Started")
})
.then(
    duration: 0.3,
    curve: Curve.easeInCirc,
    animation: {
        self.view.alpha = 1.0
    }
)
.completed({ _ in print(“All done”) })
```

```
Slaminate(
    duration: 2.0,
    curve: Curve(transform: { pow($0, 2) }),
    animation: {
        self.view.alpha = 0.0
        self.aConstraint.constant = 400.0
        self.view.removeConstraint(self.anotherConstraint)
        self.view.addConstraint(aNewConstraint)
    }
)
.completed({ (animation)  in
    print("Complete")
})
.delayed(2.0)
.started({ _ in
    print("Started")
})
.then(
    duration: 0.3,
    curve: Curve.easeInCirc,
    animation: {
        self.view.alpha = 1.0
    }
)
.completed({ _ in print(“All done”) })
```

```
let animation = Slaminate(
                    duration: 2.0,
                    curve: Curve(transform: { pow($0, 2) }),
                    animation: {
                        self.view.alpha = 0.0
                        self.aConstraint.constant = 400.0
                        self.view.removeConstraint(self.anotherConstraint)
                        self.view.addConstraint(aNewConstraint)
                    }
                )
                .completed({ (animation)  in
                    print("Complete")
                })
                .delayed(2.0)
                .started({ _ in
                    print("Started")
                })
                .then(
                    duration: 0.3,
                    curve: Curve.easeInCirc,
                    animation: {
                        self.view.alpha = 1.0
                    }
                )
                .completed({ _ in print(“All done”) })
```

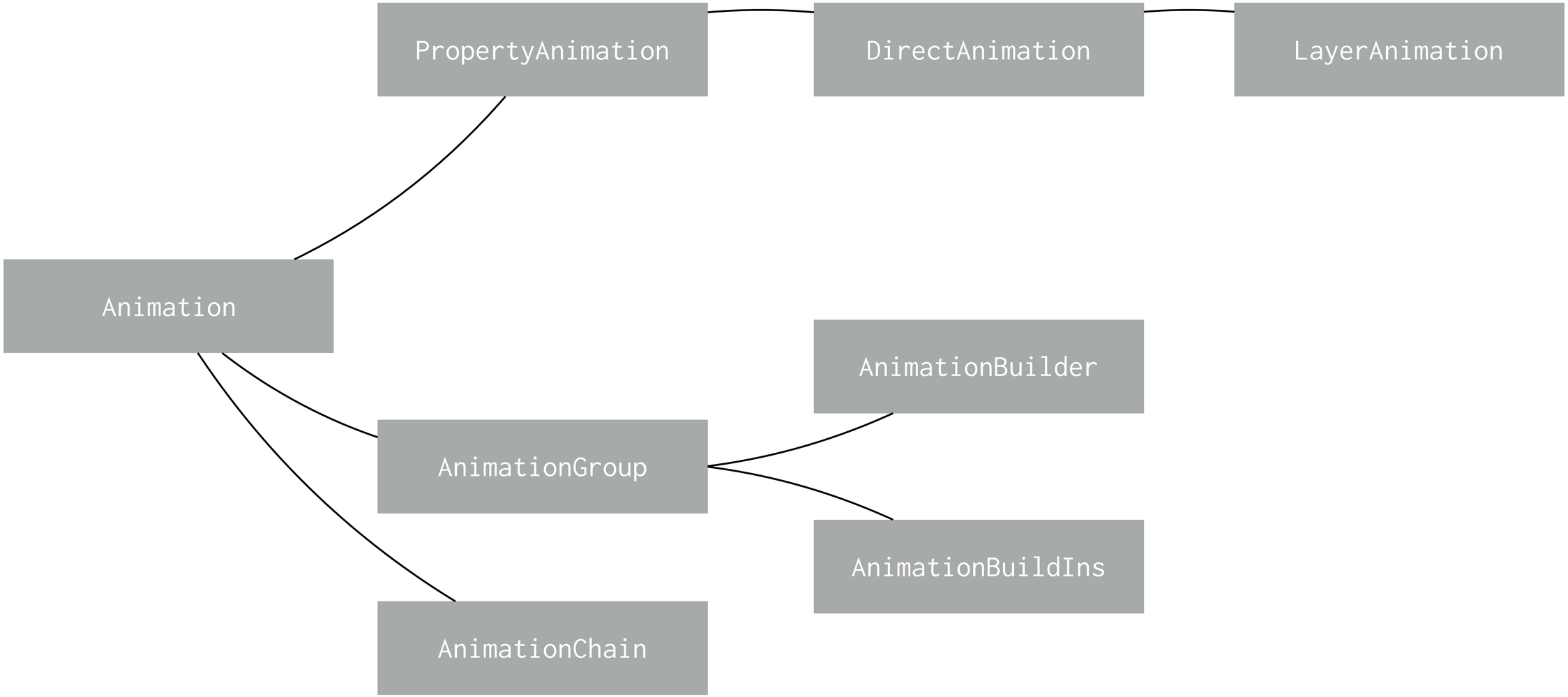
```
let animation = Slaminate(
                    duration: 2.0,
                    curve: Curve(transform: { pow($0, 2) }),
                    animation: {
                        self.view.alpha = 0.0
                        self.aConstraint.constant = 400.0
                        self.view.removeConstraint(self.anotherConstraint)
                        self.view.addConstraint(aNewConstraint)
                    }
                )
                .completed({ (animation)  in
                    print("Complete")
                })
                .delayed(2.0)
                .started({ _ in
                    print("Started")
                })
                .then(
                    duration: 0.3,
                    curve: Curve.easeInCirc,
                    animation: {
                        self.view.alpha = 1.0
                    }
                )
                .completed({ _ in print(“All done”) })
return animation
```

```
self.aView.transitionOut().then(animation: self.anotherView.transitionIn())
```

DEMO

HOW IT WORKS

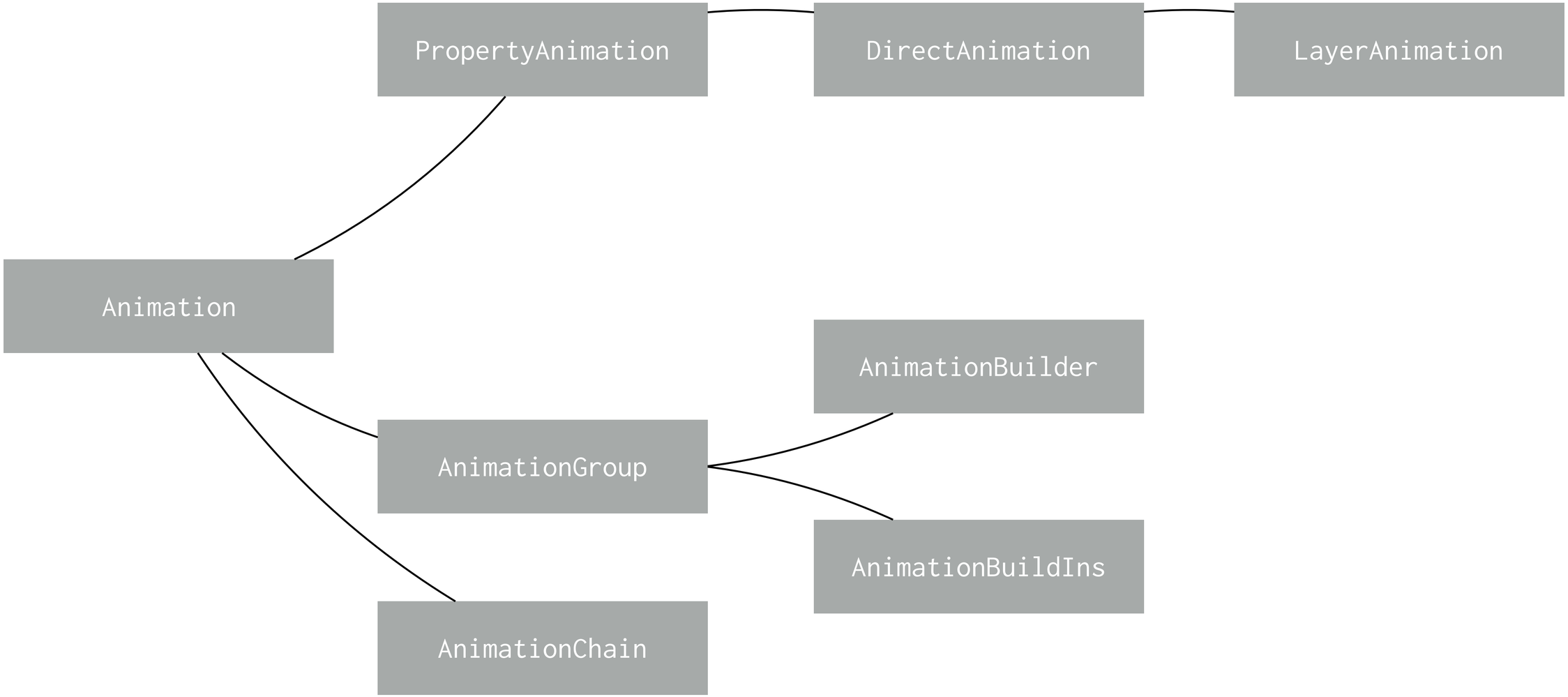
Class Hierarchy



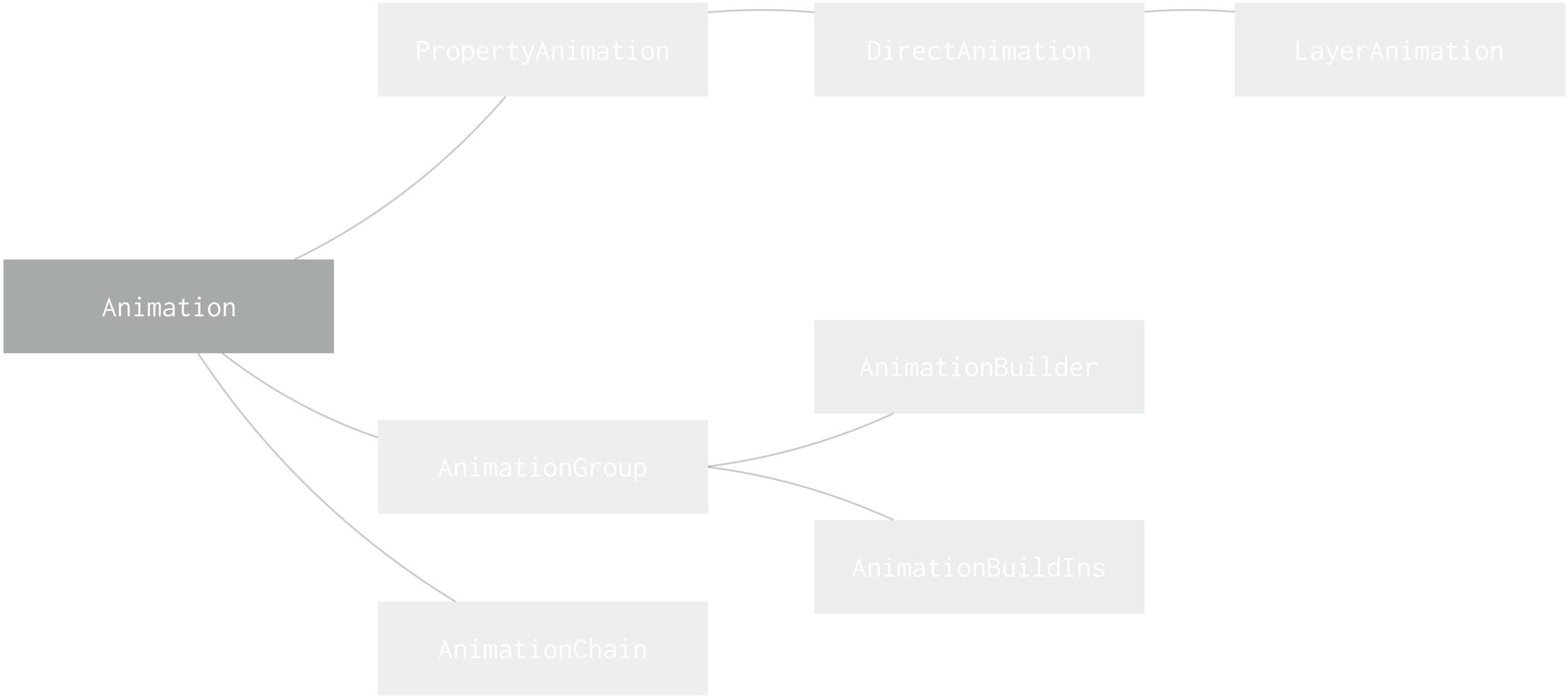
- all Objective-C compatible.

- prefixed with SLA.

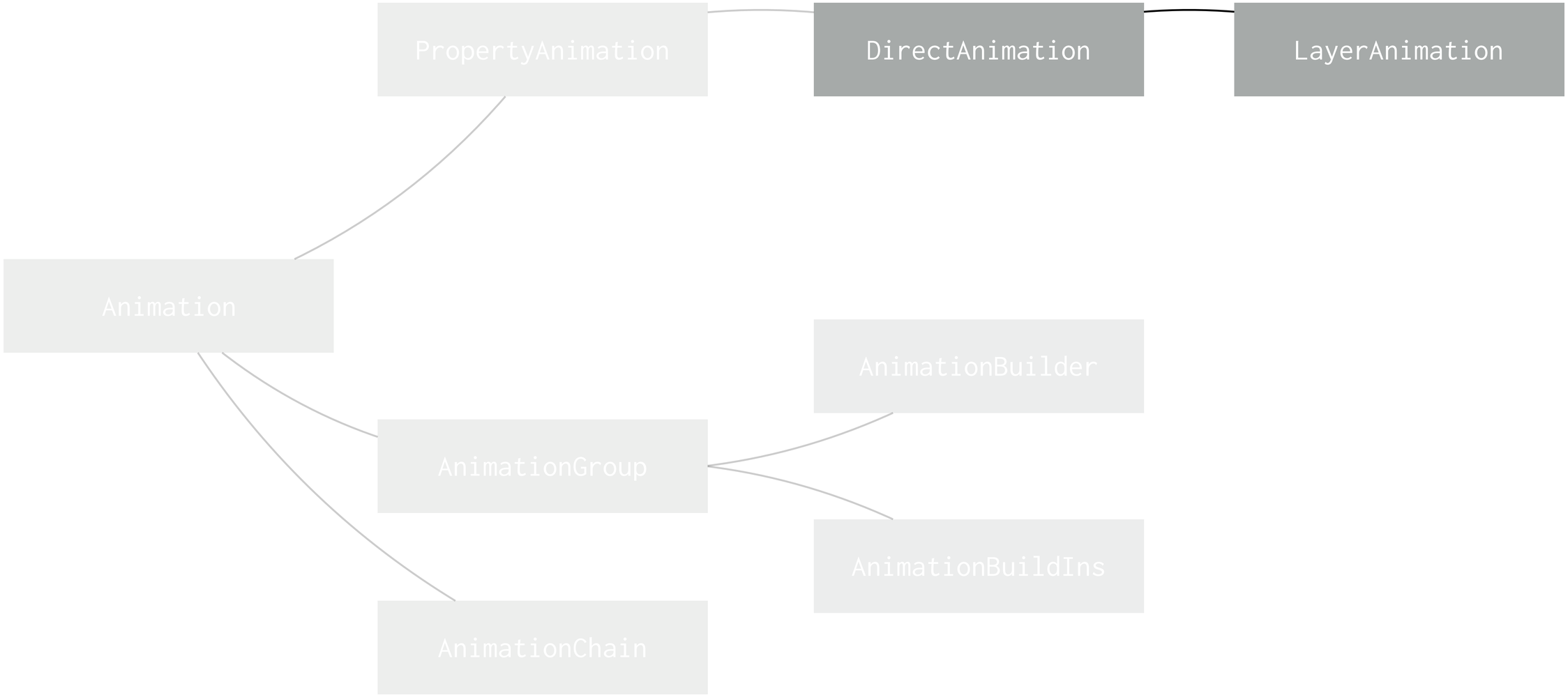
Class Hierarchy



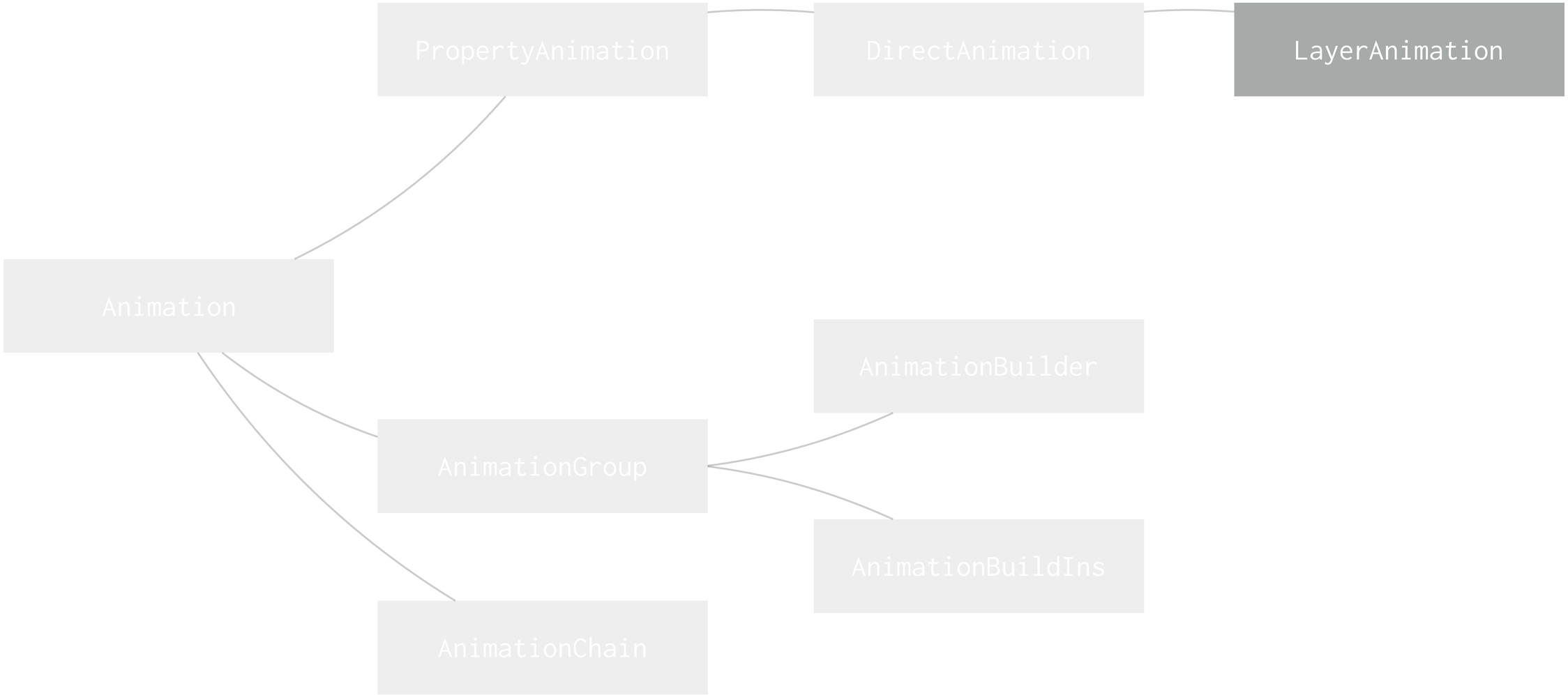
Class Hierarchy



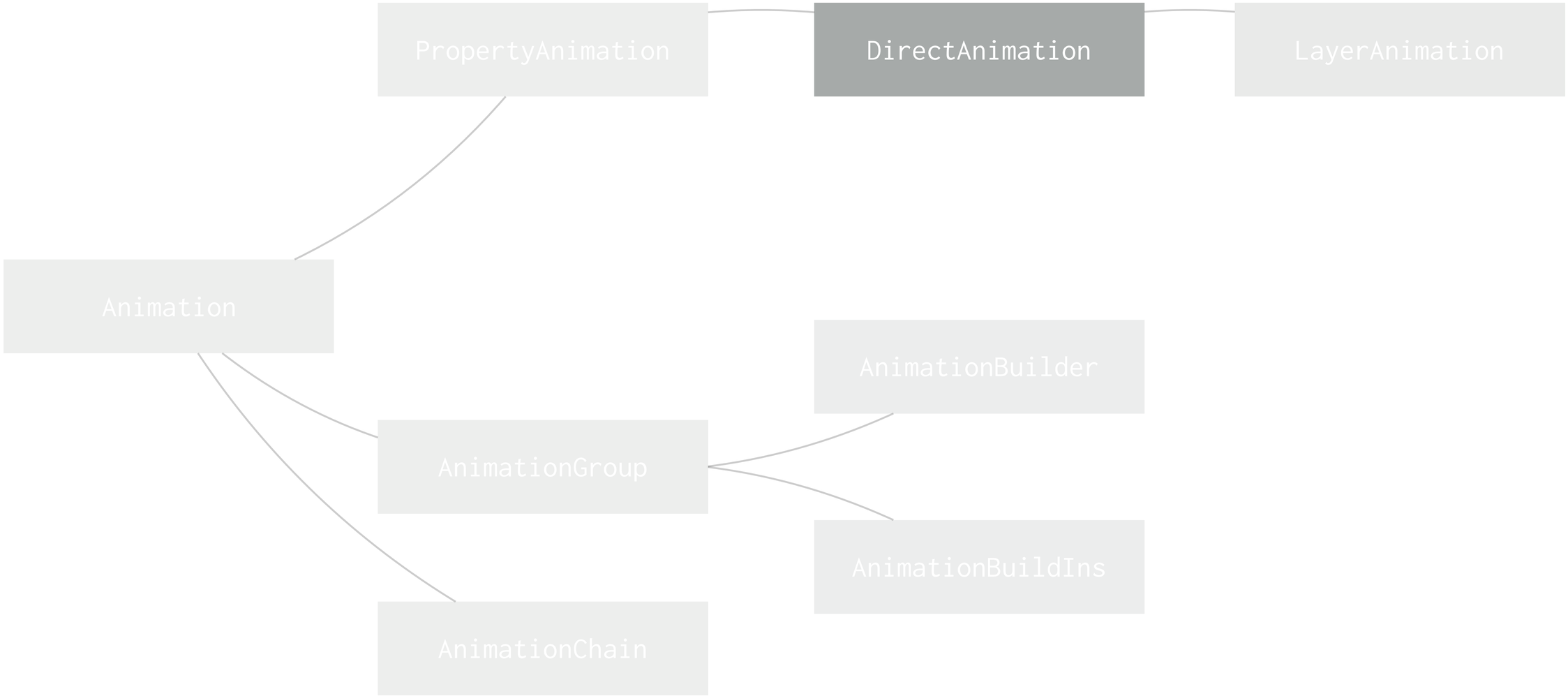
Class Hierarchy



Class Hierarchy

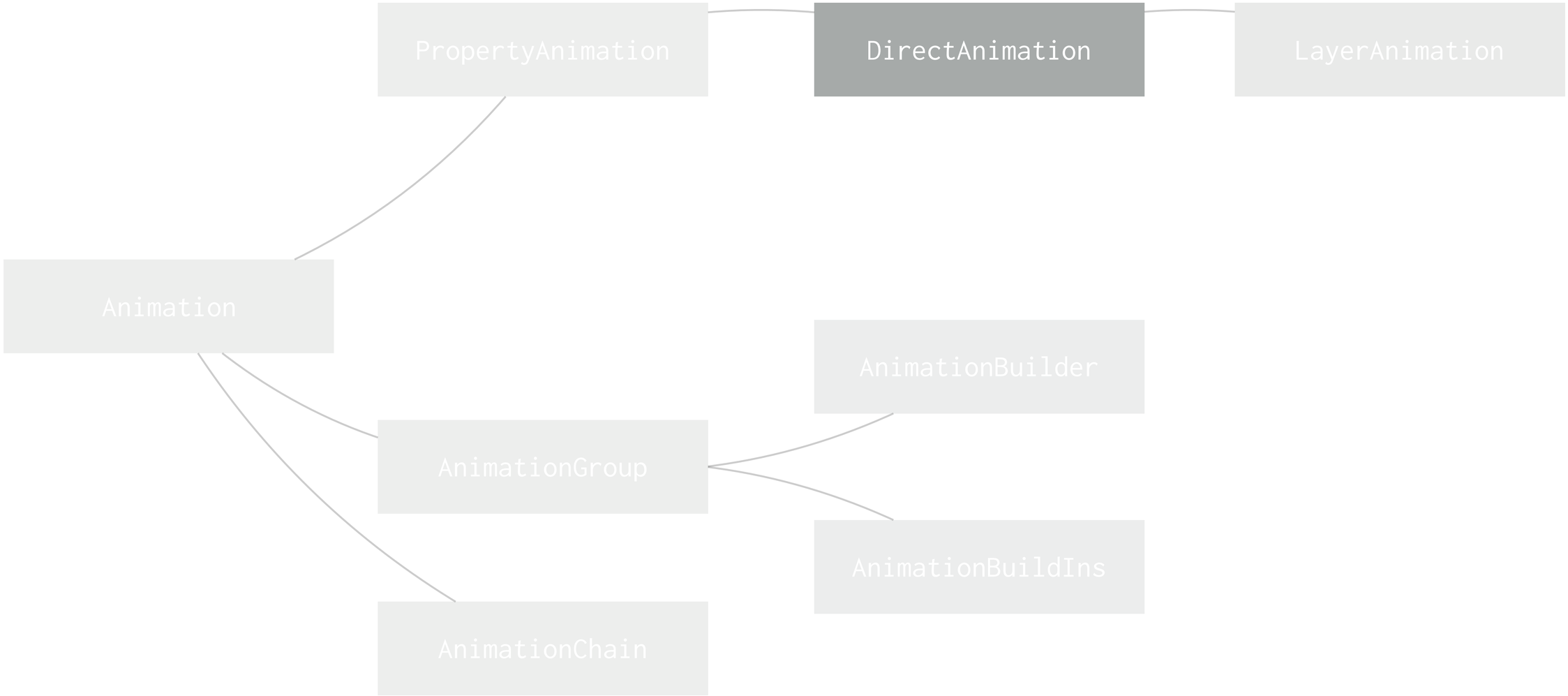


Class Hierarchy

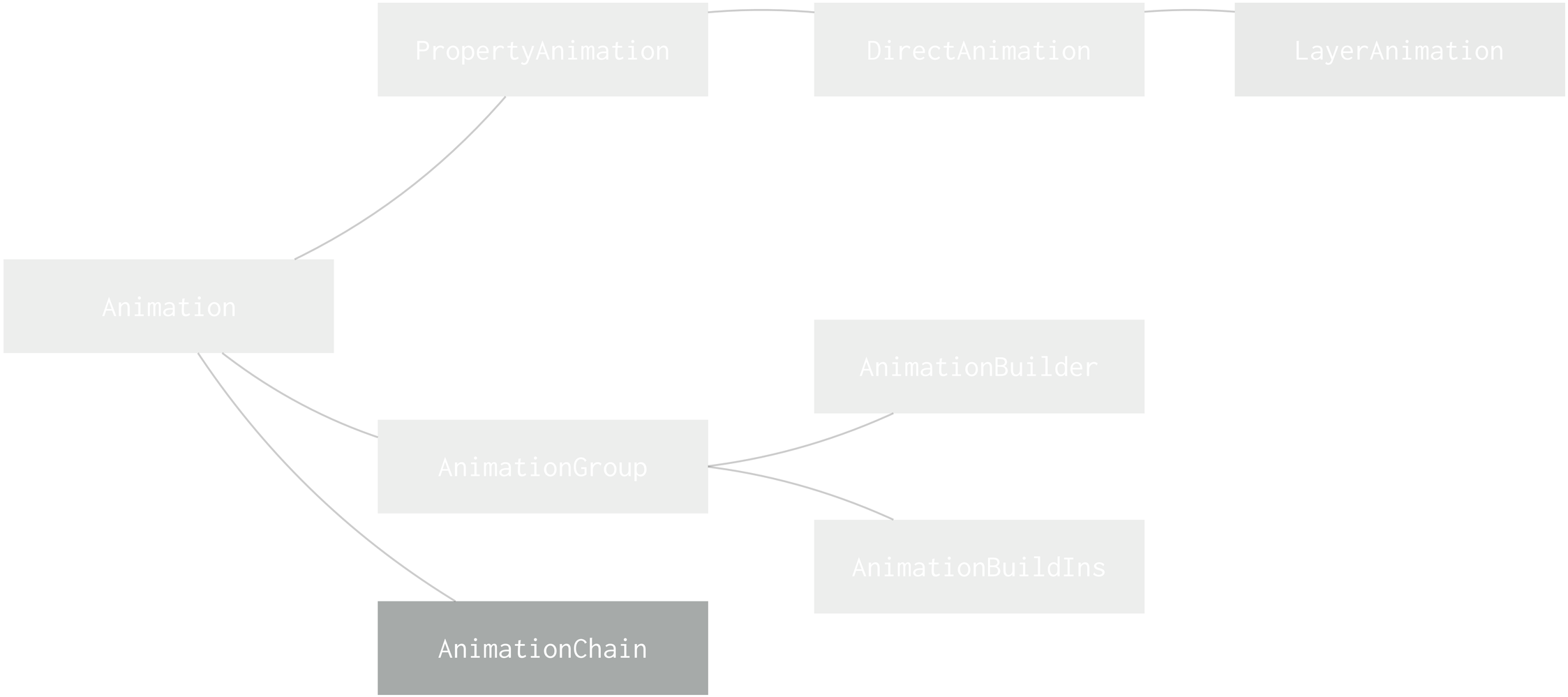


```
Slaminate(
    duration: 1.0,
    curve: Curve.easeOutSine,
    animation: {
        self.anAudioPlayer.setValue(0.0, forKey: "volume")
    }
)
.completed({ _ in
    self.anAudioPlayer.stop()
})
```

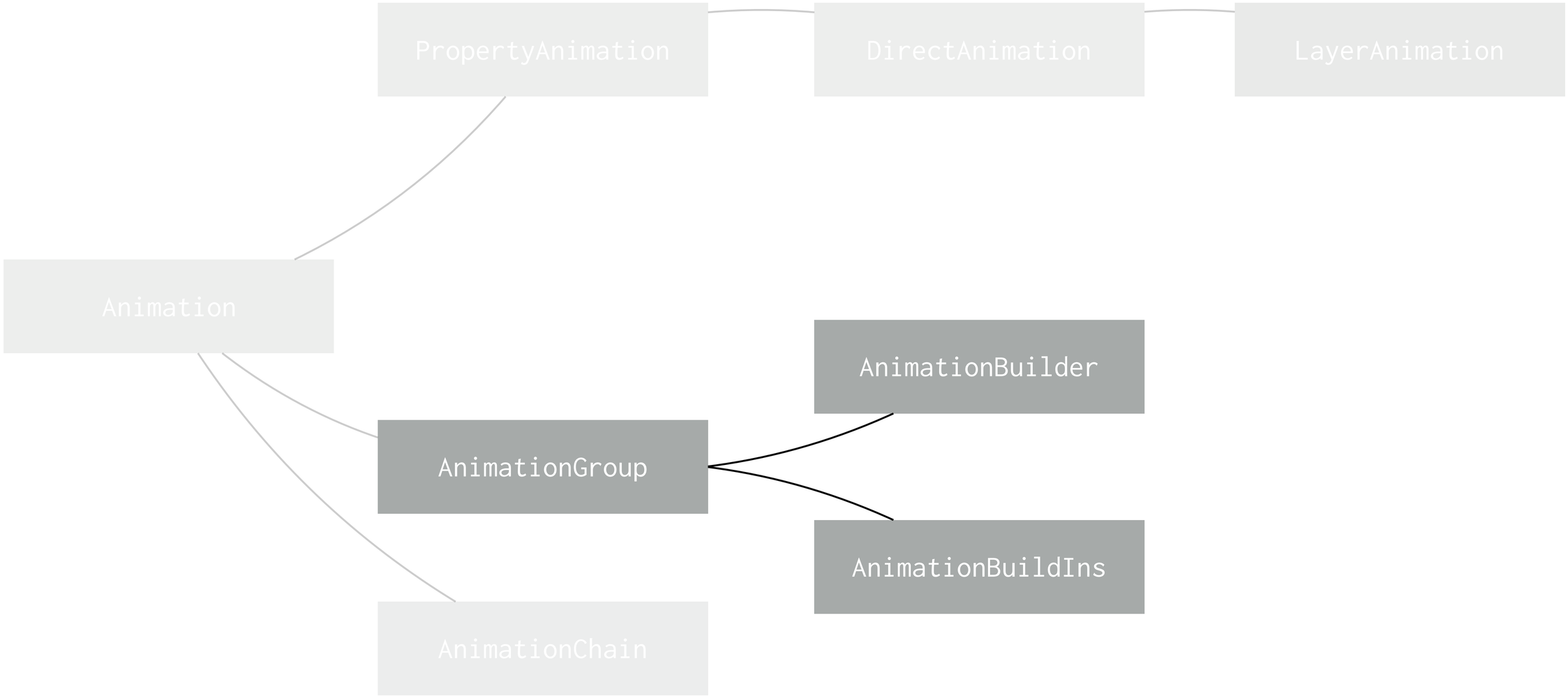
Class Hierarchy



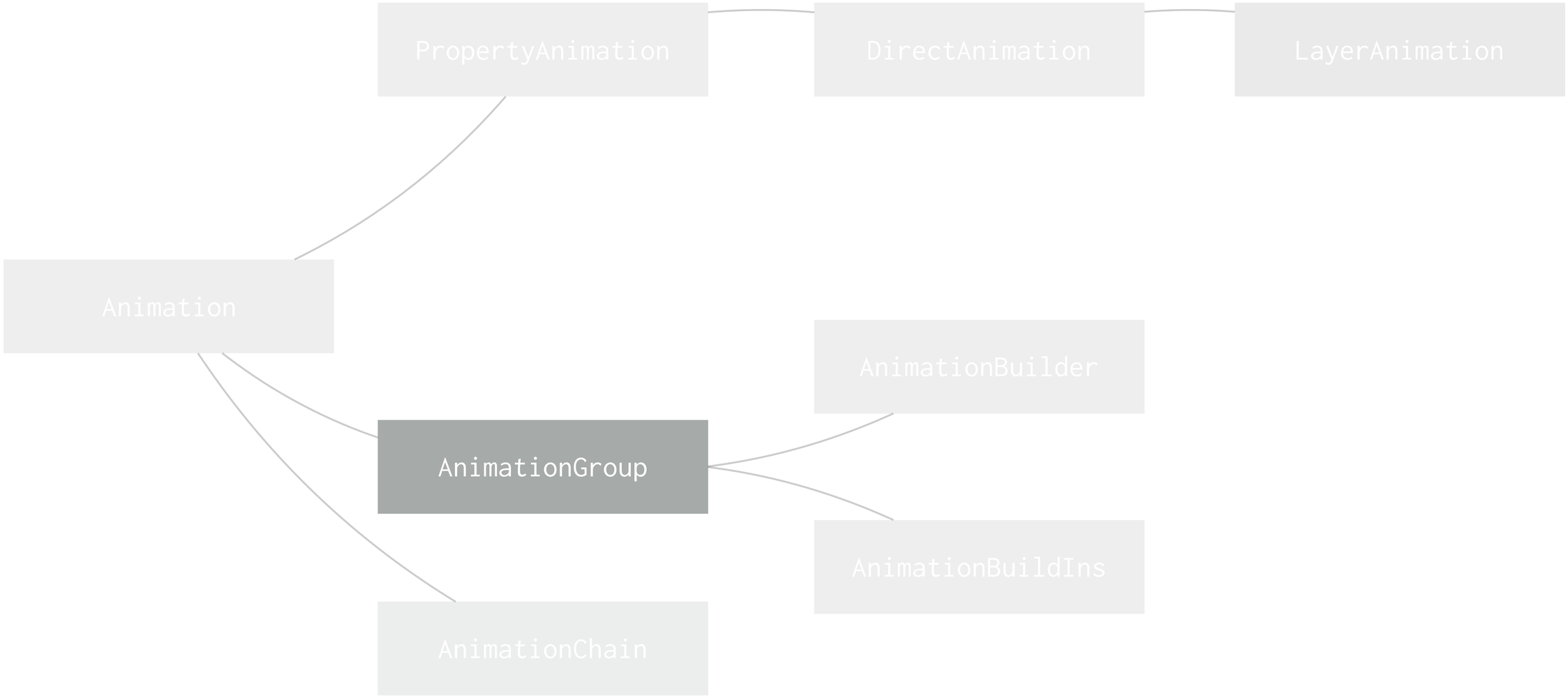
Class Hierarchy



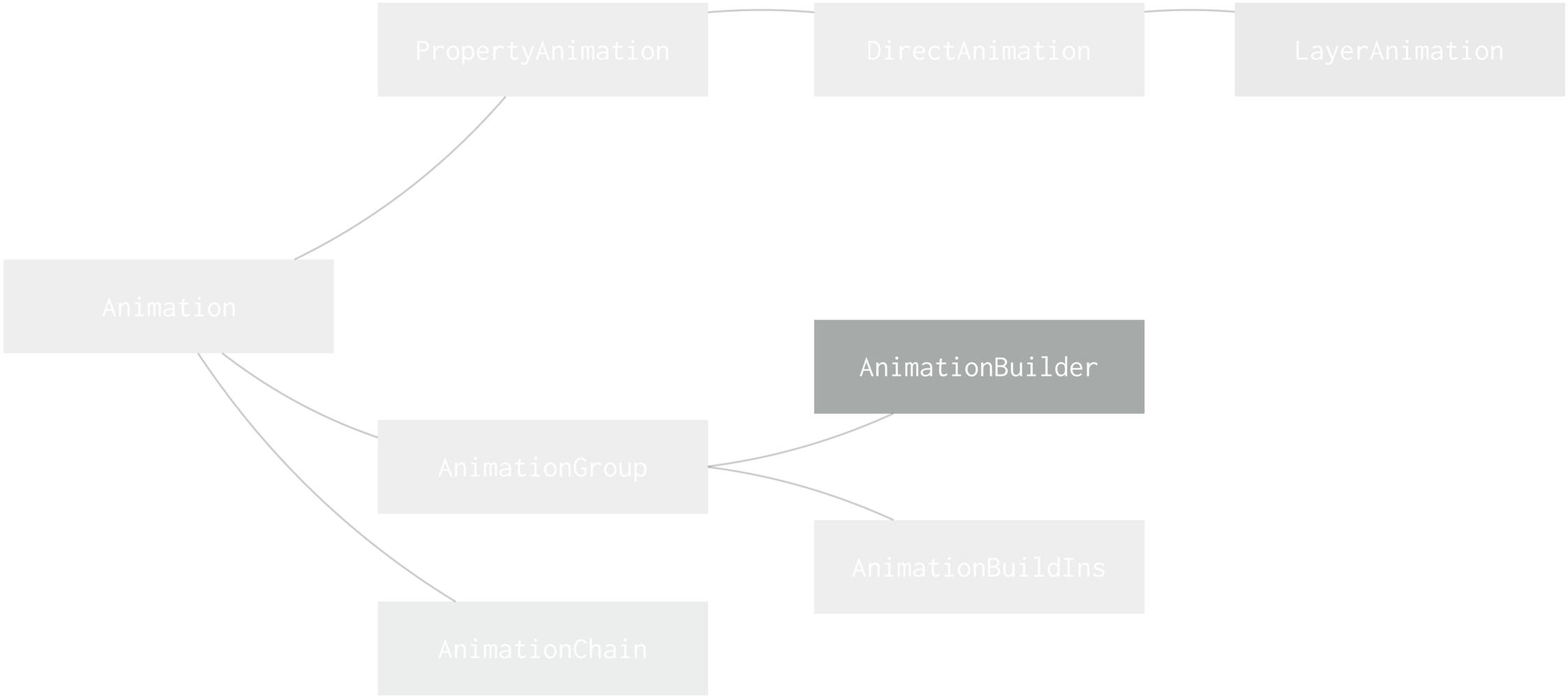
Class Hierarchy



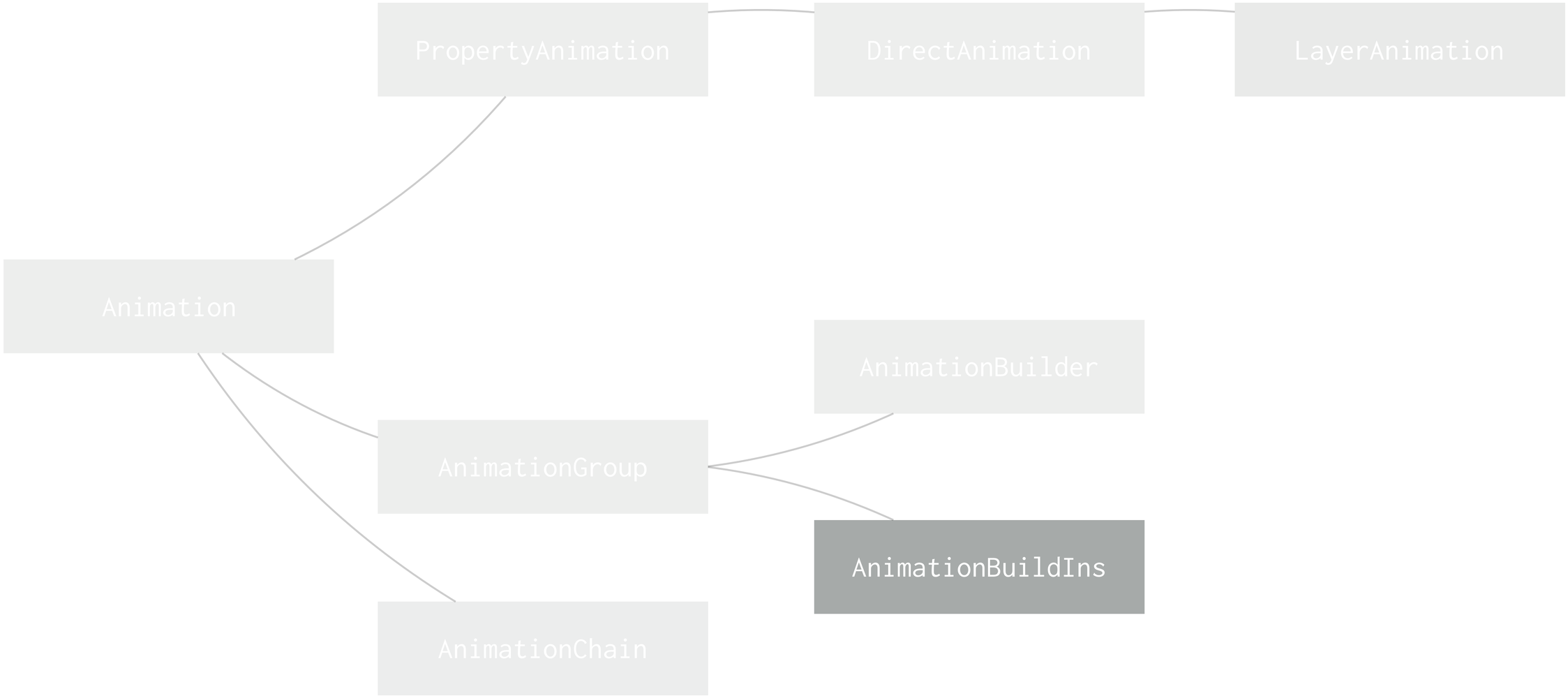
Class Hierarchy



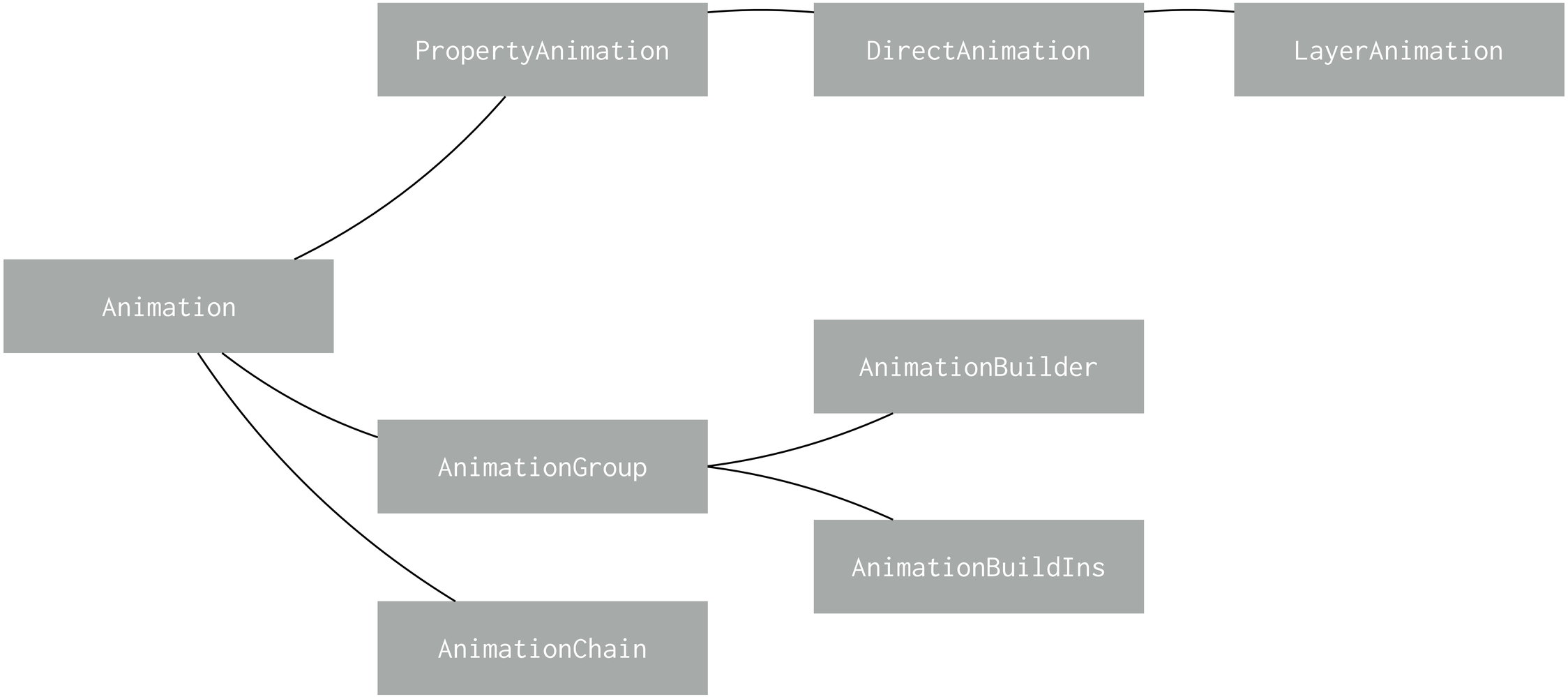
Class Hierarchy



Class Hierarchy



Class Hierarchy



WHAT I NEED FROM YOU...

- feedback

- suggestions to improve the API

- contributions! ;)

I AM KRISTIAN.

- I live in Aarhus, Denmark

- I'm freelance

- and I'm up for hire... ;)

@trenskow

github.com/trenskow

github.com/trenskow/Slaminate

THANKS!