



Subclassing UIControl

... using CoreAnimation

Nicolas Seriot @nst021

Sébastien Cloux @sbclx

iOS dev meeting
February 2nd, 2011

Motivation

- Translate graphical sketches into Cocoa
- Create animated controls \neq PNG
- Example: Swissquote ePrivate Banking



Settings

Total account value

45'434.97 CHF

Define your maximum risk level

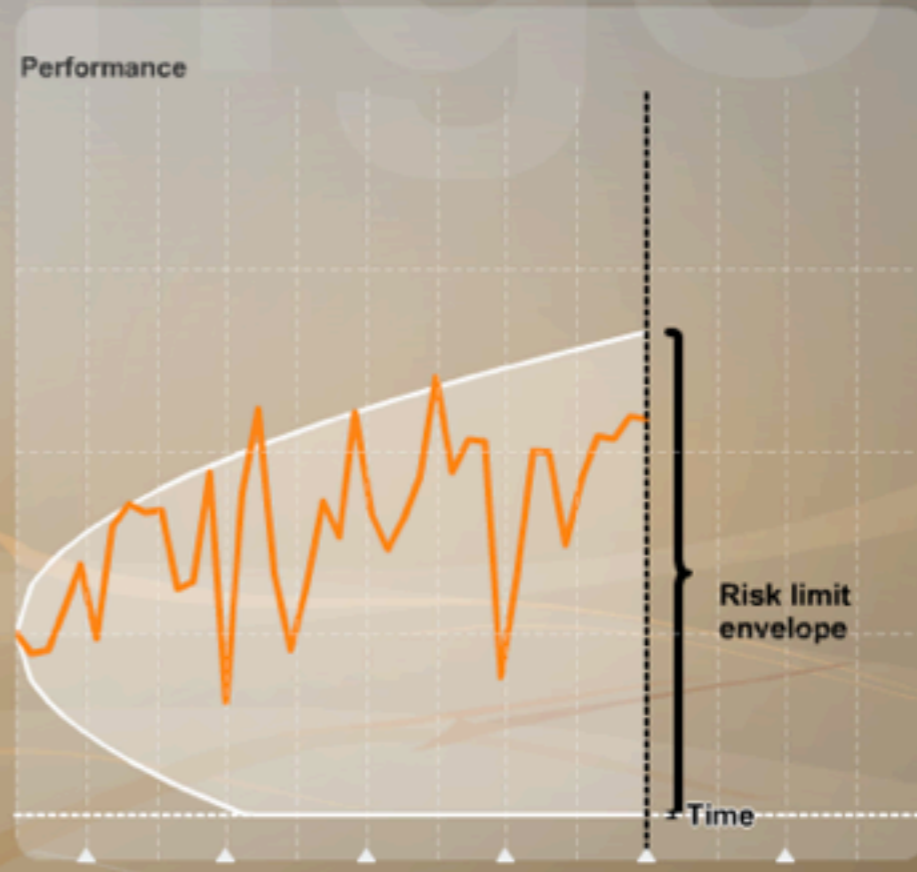
High

Potential performance chart according to the maximum risk level

Define your reference currency

CHF

Risk level:
You accept a high risk (Weekly CVaR from 0 to -5 %).
You accept the effective risk for one week:
in 95% of cases you are not going to loose more than 2271 CHF (according to your actual amount)



[Important information on risk](#)

Cancel Validate

Graphical Components



Fond bloc

Dégradé vertical partant du gris R:77/V:77/B:77 (#4D4D4D) au noir R:00/V:00/B:00 (#000000); Arrondi = 10px

Header

Dégradé vertical partant du orange R:255/V:116/B:0 (#FF7400) au noir R:00/V:00/B:00 (#000000)

Jauge




Contour blanc R:255/V:255/B:255 (#FFFFFF); Transparence 30%; Arrondi 12px; Epaisseur: 3px

Contour gris R:77/V:77/B:77 (#4D4D4D); Arrondi 5px; Epaisseur: 1px

Dégradé horizontal du vert R:69/V:207/B:64 (#45CF40) au rouge R:225/V:61/B:61 (#E13D3D) en passant par le orange R:255/V:146/B:48 (#FF9230)

How to translate this into Cocoa?

UIControl

 A horizontal segmented control with two segments: a blue segment on the left labeled "First" and a light gray segment on the right labeled "Second".	UISegmentedControl
 A horizontal slider with a blue track and a white knob positioned at approximately one-third of the track.	UISlider
 A horizontal switch with a blue track and a white knob on the left side, labeled "ON".	UISwitch

UIControl : UIView

- defines an interface for your own controls
 - interpret touches
 - hold states (enabled, selected, highlighted)
 - send events

UIControl API

```
// ...

// register and dispatch events to targets
// ... in code / Interface Builder

// handle touches
- (BOOL)beginTrackingWithTouch:(UITouch *)touch
                          withEvent:(UIEvent *)event;

- (BOOL)continueTrackingWithTouch:(UITouch *)touch
                          withEvent:(UIEvent *)event;

- (void)endTrackingWithTouch:(UITouch *)touch
                          withEvent:(UIEvent *)event;

- (void)cancelTrackingWithEvent:(UIEvent *)event;

// drawing
- (void)drawRect:(CGRect)aRect;
- (void)layoutSubviews;
```

CALayer Basics (I)

- each UIView has CALayer
- a CALayer can hold several sublayers
- CALayer properties are animatable
- CAGradientLayer for gradients
- CAShapeLayer to add a UIBezierPath

CALayer Basics (2)

```
// create myLayer and add it to the view
self.myLayer = [CALayer layer];
myLayer.frame = CGRectMake(20, 20, 100, 60);
myLayer.backgroundColor = [UIColor redColor].CGColor;
myLayer.borderWidth = 1.0;
myLayer.borderColor = [UIColor blackColor].CGColor;
[self.view.layer insertSublayer:myLayer atIndex:0];
```



```
// implicit animation
myLayer.backgroundColor = [UIColor greenColor].CGColor;
```



```
// key path support for structure fields
NSNumber *n = [NSNumber numberWithFloat:70.0];
[myLayer setValue:n forKeyPath:@"frame.size.width"];
```



SQRiskCursor



sliderGradientLayer

+



shadowArcLayer

+



handleGradientLayer

=



Demo

```

- (CAGradientLayer *)buildSliderGradientLayer {

    // create gradient layer (incomplete)
    CAGradientLayer *gl = [CAGradientLayer layer];
    gl.colors = [NSArray arrayWithObjects:
        (id)[[UIColor colorWithWebColor:0x45CF40] CGColor],
        (id)[[UIColor colorWithWebColor:0xFF9230] CGColor],
        (id)[[UIColor colorWithWebColor:0xE13D3D] CGColor], nil];

    gl.locations = [NSArray arrayWithObjects:
        [NSNumber numberWithFloat:0.0],
        [NSNumber numberWithFloat:0.5],
        [NSNumber numberWithFloat:1.0], nil];

    // create glossLayer (incomplete)
    CALayer *glossLayer = [CALayer layer];
    glossLayer.backgroundColor = [UIColor whiteColor].CGColor;
    glossLayer.opacity = SHADOW_LAYER_OPACITY;
    glossLayer.cornerRadius = gl.cornerRadius;
    [gl addSublayer:glossLayer];

    // add ticks (incomplete)
    for(NSUInteger i = 0; i <= maxValue; i++) {
        CGFloat x = [self xPositionForHandleValue:(float)i offset:cornerRadius];
        CGRect frame = CGRectMake(x, 0.0, 1.0, gl.bounds.size.height);
        CALayer *l = [[self class] tickLayerWithFrame:frame];
        [gl addSublayer:l];
    }

    return gl;
}

```



```
- (CAShapeLayer *)buildShadowArcLayerWithSlider:(CALayer *)slider {
// incomplete

UIBezierPath *path = [UIBezierPath bezierPath];

[path addArcWithCenter:CGPointMake(0.0, slider.bounds.size.height/2.0)
      radius:slider.bounds.size.height/2.0
      startAngle:M_PI_2 * -1.0
      endAngle:M_PI_2
      clockwise:YES];

CAShapeLayer *sl = [CAShapeLayer layer];
sl.fillColor = [UIColor blackColor].CGColor;
sl.path = path.CGPath;

CALayer *rectangleLayer = [CALayer layer];
rectangleLayer.backgroundColor = [UIColor blackColor].CGColor;
rectangleLayer.frame = ...

[sl addSublayer:rectangleLayer];

return sl;
}
```



```
- (CAGradientLayer *)buildHandleGradient {  
  
    // create handle (incomplete)  
  
    CAGradientLayer *gl = [CAGradientLayer layer];  
    gl.frame = CGRectMake(0.0, 0.0, handleWidth, self.bounds.size.height);  
  
    gl.colors = [NSArray arrayWithObjects:  
        (id)[[UIColor lightGrayColor] CGColor],  
        (id)[[UIColor whiteColor] CGColor], nil];  
  
    gl.locations = [NSArray arrayWithObjects:  
        [NSNumber numberWithFloat:0.0],  
        [NSNumber numberWithFloat:1.0], nil];  
  
    gl.cornerRadius = handleWidth / 3.0;  
    gl.borderWidth = 1.0;  
    gl.borderColor = [UIColor colorWithWebColor:0x4D4D4D].CGColor;  
  
    // add tick (incomplete)  
  
    CALayer *tick = [CALayer layer];  
    tick.frame = CGRectMake(handleWidth / 2.0 - 1, 1.0/6.0 * gl.bounds.size.height,  
        2, 4.0/6.0 * gl.bounds.size.height);  
    tick.borderWidth = 1.0;  
    tick.borderColor = [UIColor grayColor].CGColor;  
  
    [gl addSublayer:tick];  
  
    return gl;  
}
```



Handle Touches (I)

```
- (BOOL)beginTrackingWithTouch:(UITouch *)touch withEvent:(UIEvent *)event {
    [self sendActionsForControlEvents:UIControlEventTouchUpInside];

    CGPoint p = [touch locationInView:self];
    [self didTouchPoint:p animated:YES];

    return YES;
}

- (BOOL)continueTrackingWithTouch:(UITouch *)touch withEvent:(UIEvent *)event {
    CGPoint p = [touch locationInView:self];

    if (CGRectContainsPoint(self.frame, p)) [self sendActionsForControlEvents:UIControlEventTouchUpInside];
    else [self sendActionsForControlEvents:UIControlEventTouchDragOutside];

    [self didTouchPoint:p animated:NO];

    return YES;
}

- (void)endTrackingWithTouch:(UITouch *)touch withEvent:(UIEvent *)event {
    CGPoint p = [touch locationInView:self];

    if (CGRectContainsPoint(self.bounds, p)) [self sendActionsForControlEvents:UIControlEventTouchUpInside];
    else [self sendActionsForControlEvents:UIControlEventTouchUpOutside];

    [self adjustTouchValueFromPoint:p];
}

- (void)cancelTrackingWithEvent:(UIEvent *)event {
    [self sendActionsForControlEvents:UIControlEventTouchCancel];
}
```

Handle Touches (2)

```
- (void)moveHandleToValue:(CGFloat)aValue animated:(BOOL)animated {
```

```
    CGFloat x = [self xPositionForHandleValue:aValue offset:sliderGradient.cornerRadius];
```

```
    if(animated == NO) {  
        [CATransaction begin];  
        [CATransaction setValue:(id)kCFBooleanTrue  
            forKey:kCATransactionDisableActions];  
    }
```

```
    handleGradient.position = CGPointMake(x,  
        (sliderGradient.bounds.size.height + sliderGradient.cornerRadius) / 2.0);
```

```
    [self moveShadowLimitToStartPositionX:x];
```

```
    if(animated == NO) {  
        [CATransaction commit];  
    }
```

```
}
```

```
- (void)setValue:(NSInteger)aValue {
```

```
    value = MIN(aValue, maxValue);  
    [self moveHandleToValue:(float)value animated:YES];  
    [self sendActionsForControlEvents:UIControlEventValueChanged];  
}
```

```
- (void)didTouchPoint:(CGPoint)p animated:(BOOL)animated {
```

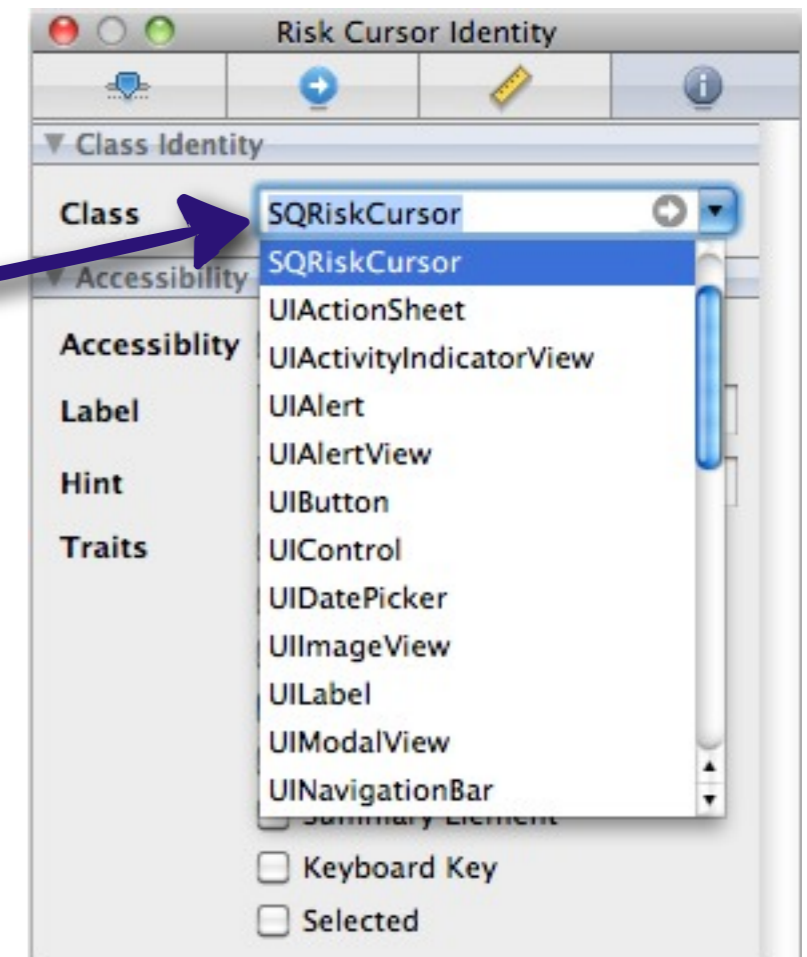
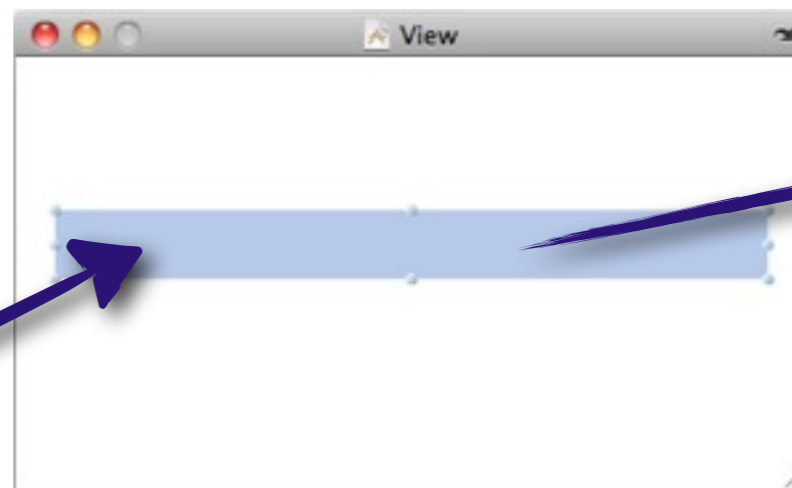
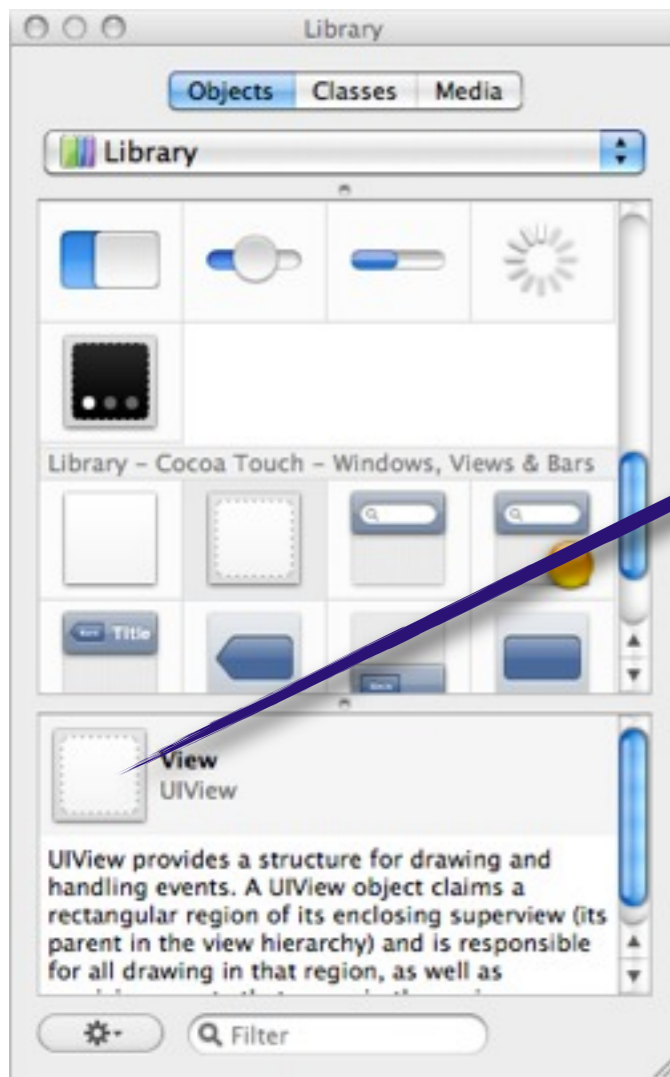
```
    CGFloat touchValue = [self touchValueForPoint:p];
```

```
    [self moveHandleToValue:(float)touchValue animated:animated];
```

```
}
```

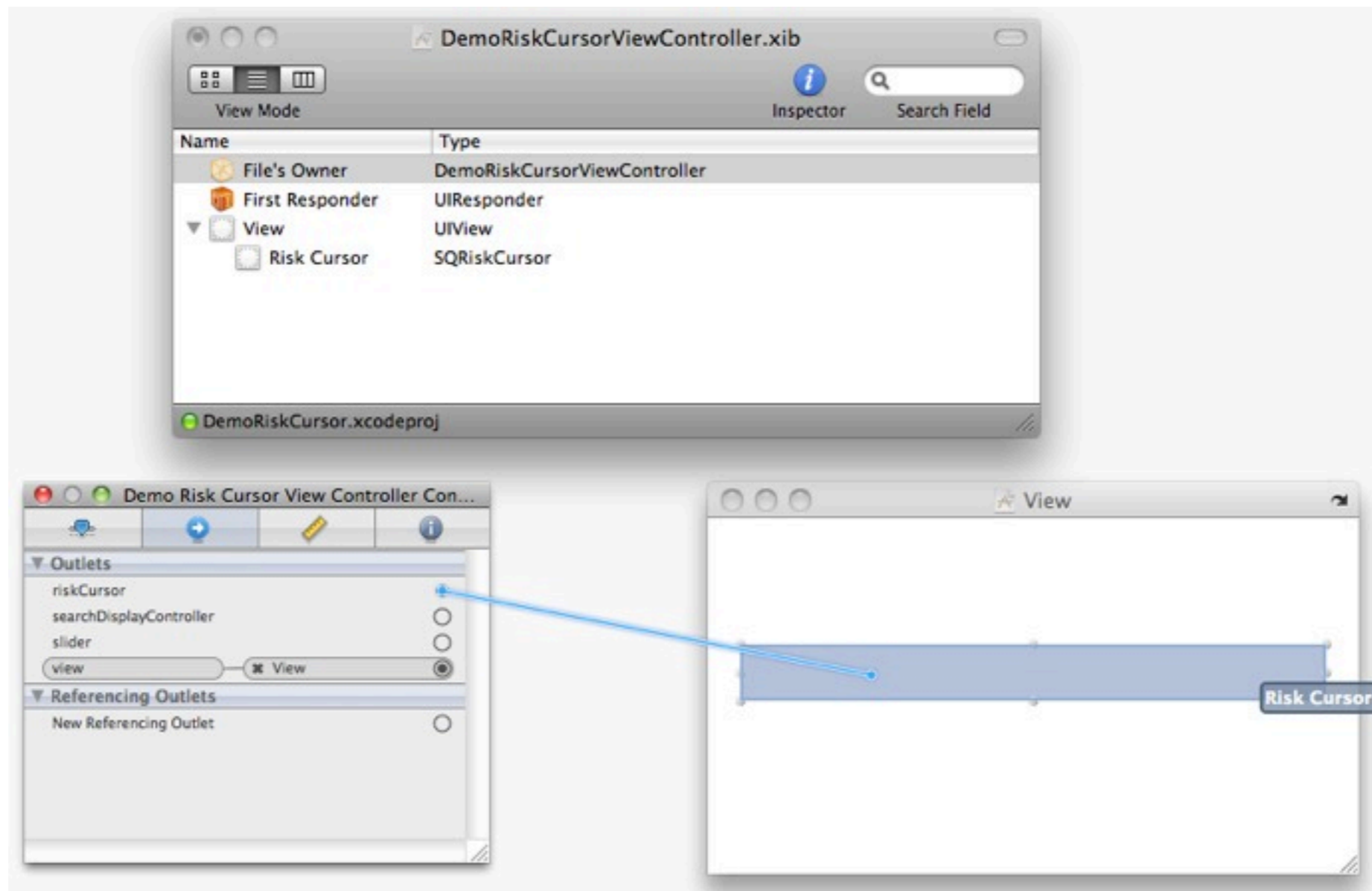
Using Interface Builder

- Insert a UIView
- Set the right Class Identity



Interface Builder

- Connect it





Handle Touches (again)

```
- (BOOL)beginTrackingWithTouch:(UITouch *)touch withEvent:(UIEvent *)event {
    [self sendActionsForControlEvents:UIControlEventTouchUpInside];

    CGPoint p = [touch locationInView:self];
    [self didTouchPoint:p animated:YES];

    return YES;
}

- (BOOL)continueTrackingWithTouch:(UITouch *)touch withEvent:(UIEvent *)event {
    CGPoint p = [touch locationInView:self];

    if (CGRectContainsPoint(self.frame, p)) [self sendActionsForControlEvents:UIControlEventTouchUpInside];
    else [self sendActionsForControlEvents:UIControlEventTouchUpInside];

    [self didTouchPoint:p animated:NO];

    return YES;
}

- (void)endTrackingWithTouch:(UITouch *)touch withEvent:(UIEvent *)event {
    CGPoint p = [touch locationInView:self];

    if (CGRectContainsPoint(self.bounds, p)) [self sendActionsForControlEvents:UIControlEventTouchUpInside];
    else [self sendActionsForControlEvents:UIControlEventTouchUpInside];

    [self adjustTouchValueFromPoint:p];
}

- (void)cancelTrackingWithEvent:(UIEvent *)event {
    [self sendActionsForControlEvents:UIControlEventTouchUpInside];
}
```

User Interactions

- UIControl have UIControlEvents

```
enum {
    UIControlEventTouchDown           = 1 << 0,      // on all touch downs
    UIControlEventTouchDownRepeat     = 1 << 1,      // on multiple touchdowns (tap count > 1)
    UIControlEventTouchDragInside     = 1 << 2,
    UIControlEventTouchDragOutside    = 1 << 3,
    UIControlEventTouchDragEnter      = 1 << 4,
    UIControlEventTouchDragExit       = 1 << 5,
    UIControlEventTouchUpInside       = 1 << 6,
    UIControlEventTouchUpOutside      = 1 << 7,
    UIControlEventTouchCancel         = 1 << 8,

    UIControlEventValueChanged        = 1 << 12,     // sliders, etc.
    //...
};
typedef NSUInteger UIControlEvents;
```

- Implement UIControl methods for sending Actions for Events

```
[self sendActionsForControlEvents:UIControlEventTouchUpInside];
[self sendActionsForControlEvents:UIControlEventTouchCancel];
[self sendActionsForControlEvents:UIControlEventTouchUpOutside];

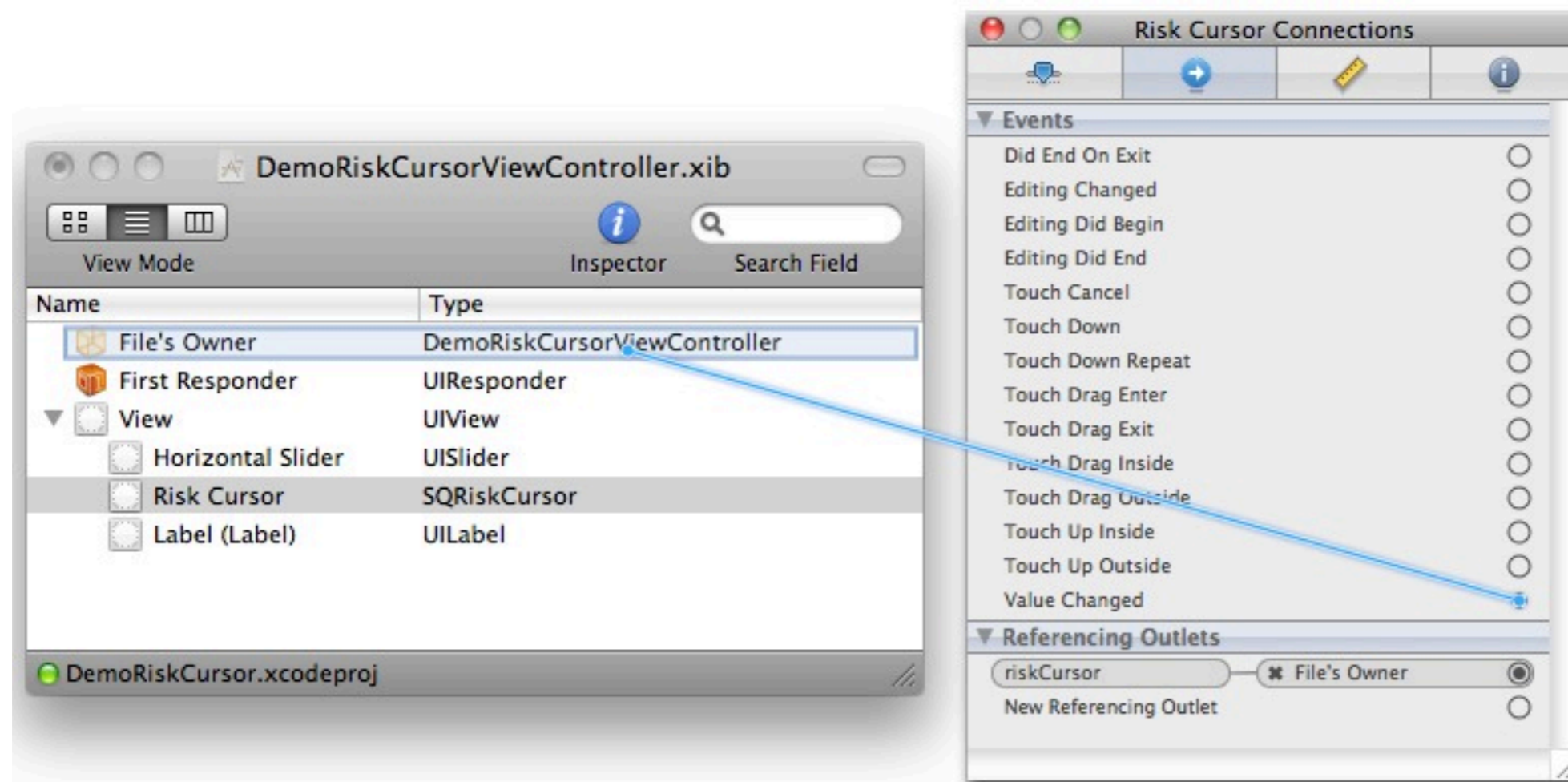
- (void)setValue:(NSUInteger)aValue {
    value = MIN(aValue, maxValue);

    [self moveHandleToValue:(float)value animated:YES];

    [self sendActionsForControlEvents:UIControlEventValueChanged];
}
```

UIControl Events in IB

- Connect IBActions



Value & Rotation

```
riskCursor.maxValue = 2;  
  
riskCursor.transform =  
    CGAffineTransformMakeRotation(M_PI/2);
```



Using SQRiskCursor

- resizing
- set cursor value
- get cursor value
- change max value
- rotate cursor

Demo

Conclusion

- Cocoa makes creating custom controls easy
- Subclass, draw, handle touches, send events
- Interface Builder makes reusing controls easy
- Improve user experience
- Increase application perceived value

Can I download it for free?

<http://github.com/nst/SQRiskCursor>

```
// You may use this code for information purpose,  
// but not reuse it as such without Swissquote written authorization.
```

Thank you, and happy coding!