iPhone Specialist Lab
L08:Quartz & Animation

Prof. Dr. Jan Borchers, Florian Heller, Jonathan Diehl
Media Computing Group, RWTH Aachen University

2011

http://hci.rwth-aachen.de/iphone

Quartz

The View Drawing Cycle

- When does drawRect: get called?
  - A part of the view was revealed
  - Unhiding a view
  - The view was scrolled off the screen and back on
  - setNeedsDisplay was set
- Parameter defines the area to be redrawn
  - Full view at first call
  - Can be smaller in subsequent calls

Quartz & CoreGraphics

- C-based
- 2D drawing engine
- Path-based drawing
- Transparency, shading, shadows, layers
- Hardware acceleration whenever possible

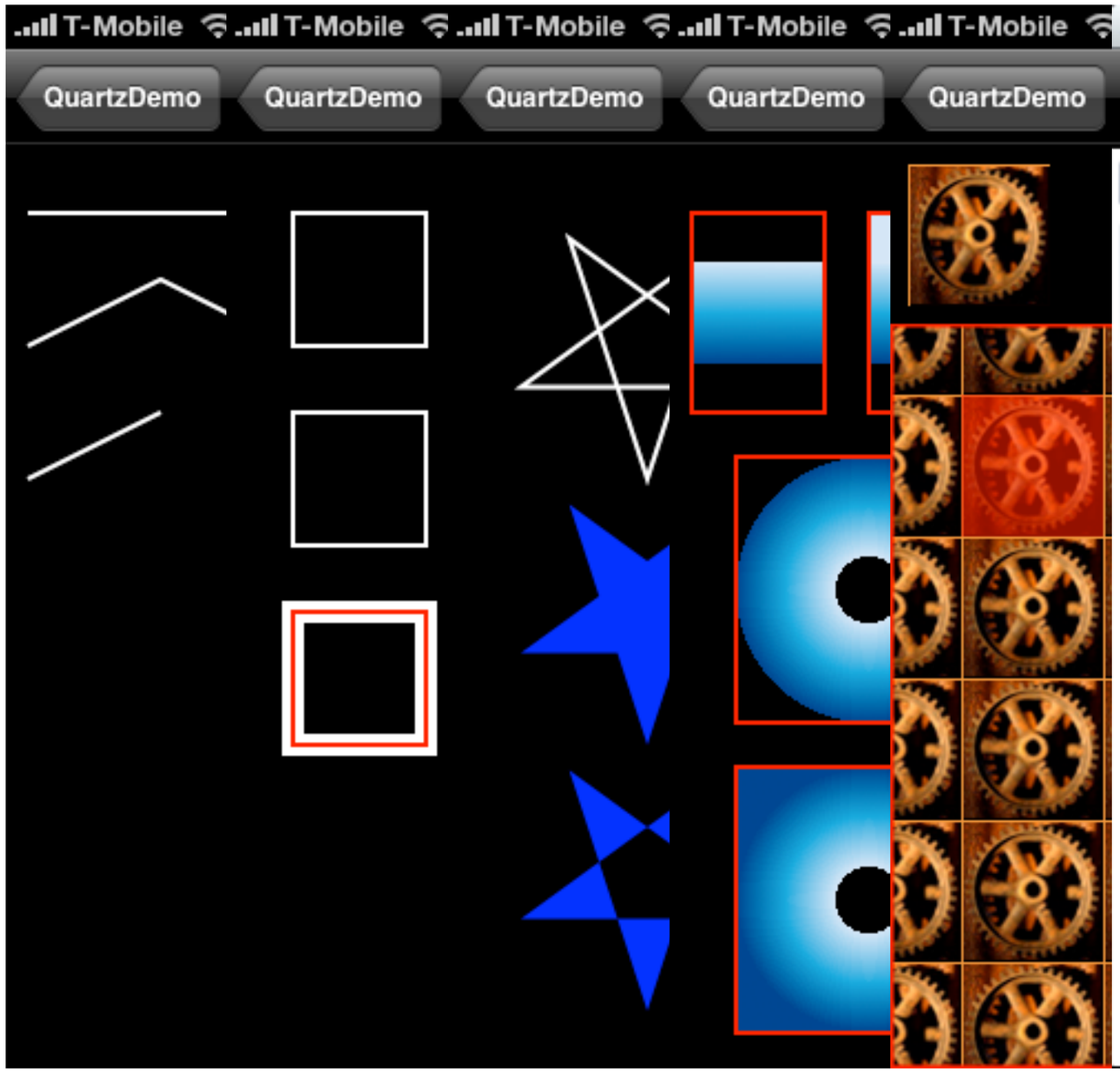
CoreGraphics Primitives

- Graphics context
- Paths
- Transformations
- Colors & Fonts
- Images & PDF

The Graphics Context

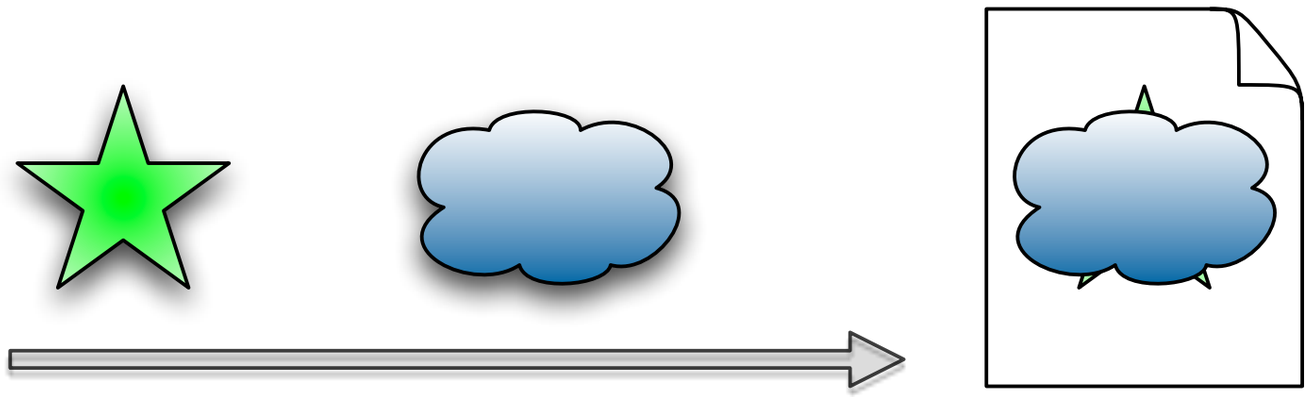
- Opaque data type (CGContextRef)
- Window, view, bitmap, PDF document
- Encapsulates drawing
  - Color
  - Line width
  - ...

CoreGraphics Examples



QuartzDemo Sample Code

Painters Drawing Model



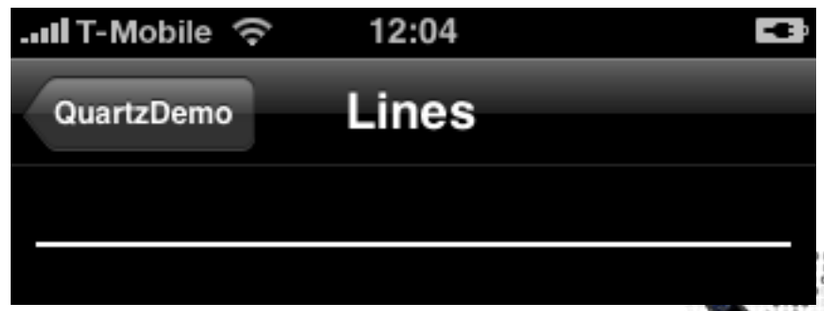
Simple Drawing Example

```
-(void)drawRect:(CGRect)rect
{
    //Get the current drawing context
    CGContextRef context = UIGraphicsGetCurrentContext();

    // Drawing lines with a white stroke color
    [[UIColor whiteColor] setStroke];

    // Draw a single line from left to right
    CGContextMoveToPoint(context, 10.0, 30.0);
    CGContextAddLineToPoint(context, 310.0, 30.0);
    CGContextStrokePath(context);

}
```



UIColor

- Simple model class
- RGBA (red, green, blue, alpha)
  - [UIColor colorWithRed:1.0 green:1.0 blue:1.0 alpha:1.0]
- Many standard colors available
  - [UIColor redColor] ...
- Can integrate with Quartz
  - [[UIColor whiteColor] setStroke]
  - [[UIColor whiteColor] setFill]

Drawing Shapes

```
// Set Stroke and Fill Color
[[UIColor blackColor] setStroke];
[[UIColor redColor] setFill];

// Draw a single line from left to right
CGContextMoveToPoint(context, 10.0, 30.0);
CGContextAddLineToPoint(context, 310.0, 30.0);
CGContextStrokePath(context);

// Draw a rectangle
CGContextMoveToPoint(context, 10.0, 80.0);
CGContextAddLineToPoint(context, 310.0, 80.0);
CGContextAddLineToPoint(context, 310.0, 380.0);
CGContextAddLineToPoint(context, 10.0, 380.0);
CGContextAddLineToPoint(context, 10.0, 80.0);
CGContextFillPath(context);
CGContextStrokePath(context);
```

Predefined Shapes

- Rectangle
  - CGContextStrokeRect(context, rect)
  - CGContextFillRect(context, rect)
- Circle / Ellipse
  - CGContextStrokeEllipseInRect(context, rect)
  - CGContextFillEllipseInRect(context, rect)
- Arcs
  - CGContextAddArc(context, center.x, center.y, radius, startAngle, endAngle, direction)

UIBezierPath

- Construct a reusable path
  - path = [[UIBezierPath alloc] init]
- Building blocks:
  - [path moveToPoint:p1]
  - [path addLineToPoint:p2]
  - Arcs / Curves
- Integrates with Quartz
  - [path stroke]
  - [path fill]

CoreAnimation

Core Animation

- Collection of Objective-C classes for animation
- High level of abstraction
  - Dynamic (animatable) attributes
  - CAAnimation class

CALayer

- UIView equivalent for animation
  - All animation is performed in CALayers
- All UIViews are backed up by CALayers
  - (only Cocoa Touch, on demand for Cocoa)
  - Layer hierarchy in parallel to view hierarchy
  - view.layer
- You can create and animate your own layers
  - No need for a view

Custom CALayers

- Do not subclass CALayer
- Assign content or a delegate
- Content or delegate is queried for drawing
  - - (void)drawLayer:(CALayer *)layer inContext: (CGContextRef)context

Example: Custom CALayer

```
@implementation BoxLayerDelegate
- (void)drawLayer:(CALayer *)layer inContext:(CGContextRef)context
{
   CGContextSetRGBFillColor(context, 1.0, 0.0, 0.0, 1.0);
   CGContextFillRect(context, layer.bounds);
}
@end

// in any view
- (void)awakeFromNib;
{
   // create the box layer
   boxLayer = [[CALayer alloc] init];
   // give it a size and location
   boxLayer.bounds = CGRectMake(0.0, 0.0, 85.0, 85.0);
   boxLayer.position = CGPointMake(160.0, 100.0);
   // set the delegate
   boxLayerDelegate = [[BoxLayerDelegate alloc] init];
   boxLayer.delegate = boxLayerDelegate;
   [boxLayer setNeedsDisplay];
   // make it a sublayer
   [self.layer addSublayer:boxLayer];
}
```

Implicit Animations

- Layers offer many animatable properties
- Changing their value creates an implicit animation
  - The presented value is changed over time
- Every layer has a presentation and a model layer
  - presentationLayer: displayed (animated) values
  - modelLayer: target values

List of Animatable Properties

- Geometric: frame, bounds, position, transform...
- Background: backgroundColor, backgroundFilters
- Content: contents, contentsGravity
- Sublayers: sublayers, sublayerTransform...
- Border: borderColor, borderWidth
- Filters, Shadow, Composing, Mask

Demo

Explicit Animation

- Create animation object
  - CABasicAnimation
  - CAKeyframeAnimation
- Configure animation
  - Duration
  - Timing function
- Configure animation target
  - Key-path of animated property
  - From and to value

Example: Move Animation

```
- (void)startMoveAnimation;
{
   CGPoint orgPoint = timeLabel.layer.position;
   CGPoint targetPoint = CGPointMake(orgPoint.x, orgPoint.y + 100.0);

   CABasicAnimation *move = [[CABasicAnimation alloc] init];
   move.keyPath = @"position";
   move.fromValue = [NSValue valueWithCGPoint:orgPoint];
   move.toValue = [NSValue valueWithCGPoint:targetPoint];
   move.duration = 0.5;

   timeLabel.layer.position = targetPoint;

   // animate
   [timeLabel.layer addAnimation:move forKey:@"moveAnimation"];

}
```

Example: Spin Animation

```
- (void)startSpinAnimation;
{
   // create the spin animation
   CABasicAnimation *spin = [[CABasicAnimation alloc] init];
   spin.keyPath = @"transform.rotation";
   spin.toValue = [NSNumber numberWithFloat:M_PI * 4.0];
   spin.duration = 1.0;

   // set ease-in, ease-out as timing function
   spin.timingFunction = [CAMediaTimingFunction
     functionWithName:kCAMediaTimingFunctionEaseInEaseOut];

   // set the delegate
   spin.delegate = self;

   // set the spin animation
   [timeLabel.layer addAnimation:spin forKey:@"spinAnimation"];
   [spin release];
}
```

Example: Bounce Animation

```
- (void)startBounceAnimation;
{
   CAKeyframeAnimation *bounce = [[CAKeyframeAnimation alloc] init];
   bounce.keyPath = @"transform";

   // create the values it will pass through
   CATransform3D forward  = CATransform3DMakeScale(1.3, 1.3, 1.0);
   CATransform3D back     = CATransform3DMakeScale(0.7, 0.7, 1.0);
   CATransform3D forward2 = CATransform3DMakeScale(1.2, 1.2, 1.0);
   CATransform3D back2    = CATransform3DMakeScale(0.9, 0.9, 1.0);
   bounce.values = [NSArray arrayWithObjects:
                    [NSValue valueWithCATransform3D:CATransform3DIdentity],
                    [NSValue valueWithCATransform3D:forward],
                    [NSValue valueWithCATransform3D:back],
                    [NSValue valueWithCATransform3D:forward2],
                    [NSValue valueWithCATransform3D:back2],
                    [NSValue valueWithCATransform3D:CATransform3DIdentity],nil];

   // start animation
   [timeLabel.layer addAnimation:bounce forKey:@"bounceAnimation"];
}
```

Transformations

- Animations can use affine transformations to manipulate the layer's geometry
  - CGAffineTransform
    - Every rendered pixel is transformed
- Common transformations:
  - CGContextTranslateCTM: move origin
  - CGContextScaleCTM: change size
  - CGContextRotateCTM: rotate around anchorPoint

Demo