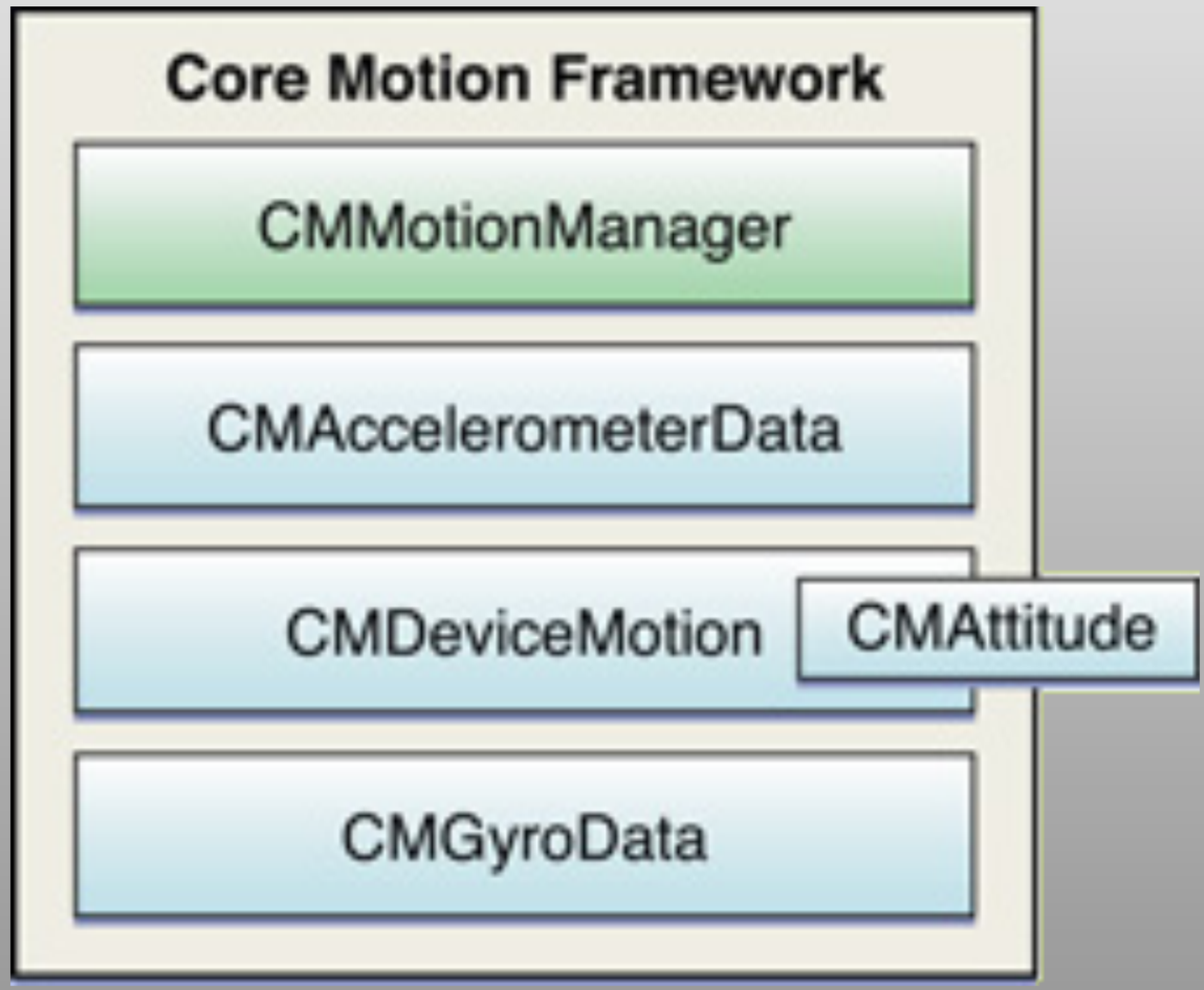
MSDOSX

Core Motion

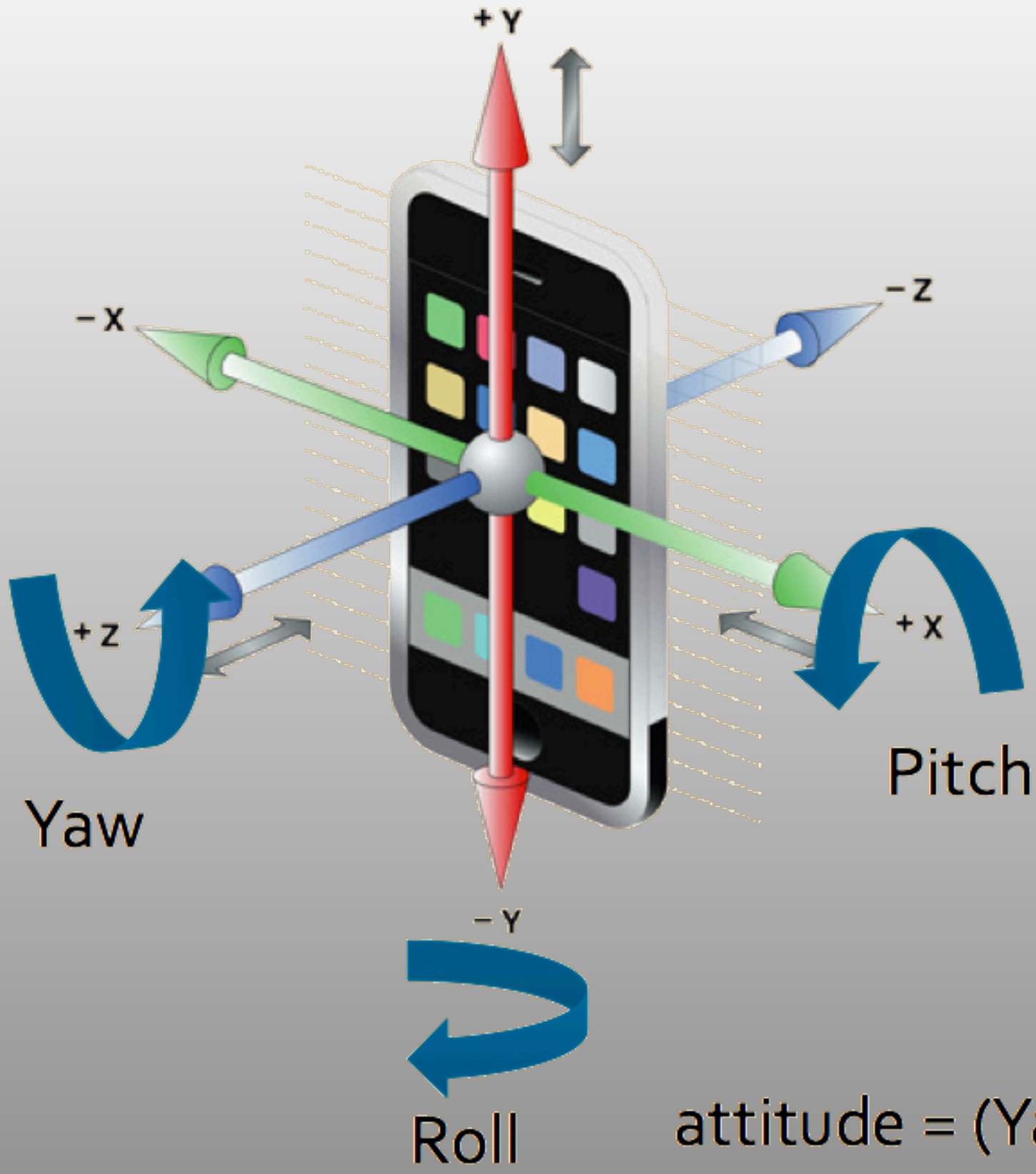
CMMotionManager Overview

- A CMMotionManager object is the gateway
  - Accelerometer data
  - Rotation-rate data
  - Magnetometer data
  - Other device-motion data such as attitude
- Create *one* instance of CMMotionManager
- Receive raw data and processed device-motion data
  - Sensor fusion algorithms gives the device's attitude, rotation rate, calibrated magnetic fields, the direction of gravity, and the acceleration.

CMMotionManager Overview



CMMotionManager



attitude = (Yaw * Pitch * Roll)^{T}

CMMotionManager Overview

- Two approaches when receiving motion data
  - Specified update intervals
  - Periodically sampling the motion data
  - With both of these approaches, the application should call the appropriate stop method (stopAccelerometerUpdates, stopGyroUpdates, stopMagnetometerUpdates, and stopDeviceMotionUpdates) when it has finished processing accelerometer, rotation-rate, magnetometer, or device-motion data.

CMMotionManager

Handing Motion Updates at Specified Intervals

- Accelerometer.
  - Set the accelerometerUpdateInterval property to specify an update interval.
  - Call the startAccelerometerUpdatesToQueue:withHandler: method, passing in a block of type CMAccelerometerHandler.
  - Accelerometer data is passed into the block as CMAccelerometerData objects.

CMMotionManager

Handing Motion Updates at Specified Intervals

- Gyroscope.
  - Set the gyroUpdateInterval property to specify an update interval.
  - Call the startGyroUpdatesToQueue:withHandler: method, passing in a block of typeCMGyroHandler.
  - Rotation-rate data is passed into the block as CMGyroData objects.

CMMotionManager

Handing Motion Updates at Specified Intervals

- Magnetometer.
  - Set the magnetometerUpdateInterval property to specify an update interval.
  - Call the startMagnetometerUpdatesToQueue:withHandler: method, passing a block of type CMMagnetometerHandler.
  - Magnetic-field data is passed into the block as CMMagnetometerData objects.

CMMotionManager

Handing Motion Updates at Specified Intervals

- Device motion.
  - Set the deviceMotionUpdateInterval property to specify an update interval.
- Call startDeviceMotionUpdatesUsingReferenceFrame:toQueue:withHandler: or startDeviceMotionUpdatesToQueue:withHandler: method, passing in a block of type CMDeviceMotionHandler.
  - With the former method (new in iOS 5.0), you can specify a reference frame to be used for the attitude estimates.
- Rotation-rate data is passed into the block as CMDeviceMotion objects.

CMMotionManager
Periodic Sampling of Motion Data

- Accelerometer.
  - Call startAccelerometerUpdates to begin updates and periodically access CMAccelerometerData objects by reading the accelerometerData property.

CMMotionManager
Periodic Sampling of Motion Data

- Gyroscope.
- Call startGyroUpdates to begin updates and periodically access CMGyroData objects by reading the gyroData property.

CMMotionManager
Periodic Sampling of Motion Data

- Magnetometer.
  - Call startMagnetometerUpdates to begin updates and periodically access CMMagnetometerData objects by reading the magnetometerData property.

CMMotionManager
Periodic Sampling of Motion Data

- Device motion.
  - Call the startDeviceMotionUpdatesUsingReferenceFrame: or startDeviceMotionUpdates method to begin updates and periodically access CMDeviceMotion objects by reading the deviceMotion property.
  - The startDeviceMotionUpdatesUsingReferenceFrame: method (new in iOS 5.0) lets you specify a reference frame to be used for the attitude estimates.

CMMotionManager
Hardware Availability

- Hardware not available?
  - No effect
- How do we know if a feature is available?
  - Properties
    - E.G.: gyroAvailable, gyroActive

CMMotionManager Motion Event Blocks

```
typedef void (^CMAccelerometerHandler)(CMAccelerometerData
*accelerometerData, NSError *error);

typedef void (^CMGyroHandler)(CMGyroData *gyroData,
NSError *error);

typedef void (^CMMagnetometerHandler)(CMMagnetometerData
*magnetometerData, NSError *error);

typedef void (^CMDeviceMotionHandler)(CMDeviceMotion
*motion, NSError *error);
```

CMAccelerometerData

The type of a structure containing 3-axis acceleration values. Acceleration measured in G's (gravitational force) (9.81 m s^{-2}).

```
typedef struct {
        double x;
        double y;
        double z;
} CMAcceleration;
```

CMAttitude

- Represents a measurement of the device's attitude at a point in time.
- "Attitude" refers to the orientation of a body relative to a given frame of reference.
- Three different representations of attitude
  - A rotation matrix
  - A quaternion
  - Euler angles (roll, pitch, and yaw values).

CMAttitude: CMQuaternion

The type for a quaternion representing a measurement of attitude.

```
typedef struct {
        double x, y, z, w;
} CMQuaternion;
```

Rotation of {w} around unit vector of {x,y,z}

CMAttitude: CMReferenceFrame

Enum constants for indicating the reference frames from which all attitude samples are referenced.

- **`CMAttitudeReferenceFrameXArbitraryZVertical`**
  - Z axis is vertical and the X axis points in an arbitrary direction in the horizontal plane.
- **`CMAttitudeReferenceFrameXArbitraryCorrectedZVertical`**
  - Describes the same reference frame as **`CMAttitudeReferenceFrameXArbitraryZVertical`** except that the magnetometer, when available and calibrated, is used to improve long-term yaw accuracy. Using this constant instead of **`CMAttitudeReferenceFrameXArbitraryZVertical`** results in increased CPU usage.
- **`CMAttitudeReferenceFrameXMagneticNorthZVertical`**
  - Describes a reference frame in which the Z axis is vertical and the X axis points toward magnetic north. Note that using this reference frame may require device movement to calibrate the magnetometer.
- **`CMAttitudeReferenceFrameXTrueNorthZVertical`**
  - Describes a reference frame in which the Z axis is vertical and the X axis points toward true north. Note that using this reference frame may require device movement to calibrate the magnetometer. It also requires the location to be available in order to calculate the difference between magnetic and true north.

CMGyroData

The CMGyroData class contains a single measurement of the device's rotation rate.

CMGyroData
CMRotationRate

CMRotationRate

```
typedef struct {
        double x;
        double y;
        double z;
} CMRotationRate;
```

“The sign follows the right hand rule: If the right hand is wrapped around the axis such that the tip of the thumb points toward the positive, a positive rotation is one toward the tips of the other four fingers.”

CMMagnetometerData

Instances of the CMMagnetometerData class encapsulated measurements of the magnetic field made by the device's magnetometer.

CMMagnetometerData
CMMagneticField

CMMagneticField

```
typedef struct {
    double x;
    double y;
    double z;
} CMMagneticField;
```

Magnetic fields in microteslas.

CMDeviceMotion

- Encapsulates: attitude, rotation rate, and acceleration.
- Measures sum of gravity and user acceleration vectors.
  - *User acceleration* is the acceleration that the user imparts to the device.
- Because Core Motion is able to track a device's attitude using both the gyroscope and the accelerometer, it can differentiate between gravity and user acceleration. A CMDeviceMotion object provides both measurements in the gravity and userAcceleration *properties*.

CMDeviceMotion
CMCalibratedMagneticField

CMCalibratedMagneticField

```
typedef struct {
    CMMagneticField field;
    CMMagneticFieldCalibrationAccuracy accuracy;
} CMCalibratedMagneticField;
```

field
A CMMagneticField containing 3-axis calibrated magnetic field data.

accuracy
The accuracy of the magnetic field estimate:

- CMMagneticFieldCalibrationAccuracyUncalibrated
- CMMagneticFieldCalibrationAccuracyLow
- CMMagneticFieldCalibrationAccuracyMedium
- CMMagneticFieldCalibrationAccuracyHigh

CMLogItem

- Base class for Core Motion classes
- Handles specific types of motion events.
- Represent a piece of time-tagged data
    - Usually logged to a file.
- Read-only timestamp property
    - Records the time a measurement was taken.
    - Time in seconds since the phone booted.

CoreMotion
CMError

```
typedef enum {
   CMErrorNULL = 100,
   CMErrorDeviceRequiresMovement,
   CMErrorTrueNorthNotAvailable
} CMError;
```

CMErrorNULL

No error.

CMErrorDeviceRequiresMovement

The device must move for a sampling of motion data to occur.

CMErrorTrueNorthNotAvailable

True north is not available on this device. This usually indicates that the device's location is not yet available.

Lab (Optional)

- Get AccelerometerGraph project working on your iOS device
- Must “Code Sign” the target application
  - Build Settings : Code Signing
  - Talk to Kevin if you don’t have an “identity” (profile) yet.

References I

- Core Motion (Apple)

  http://developer.apple.com/library/IOS/#documentation/CoreMotion/Reference/CoreMotion_Reference/_index.html

- Motion Events

  http://developer.apple.com/library/ios/#documentation/EventHandling/Conceptual/EventHandlingiPhoneOS/MotionEvents/MotionEvents.html

References II

- CMMotionManager Class Reference

  http://developer.apple.com/library/IOS/#documentation/CoreMotion/Reference/CMMotionManager_Class/Reference/Reference.html

- Block Programming Topics

  http://developer.apple.com/library/ios/#documentation/Cocoa/Conceptual/Blocks/Articles/00_Introduction.html