CS193P - Lecture 15

iPhone Application Development

iPhone Device APIs
Location, Accelerometer & Camera

デバイス API
位置・加速度・カメラ

Battery Life & Power Management

Announcements

- Paparazzi 4 due ***Friday*** night at 11:59PM
  - Late days: use 'em if you've got 'em
- Work on final projects!

Today's Topics

- Hardware features
  - Image Picker & Camera
  - Location
  - Accelerometer
- Battery Life & Power Management

イメージピッカーとカメラ

位置（センサ）

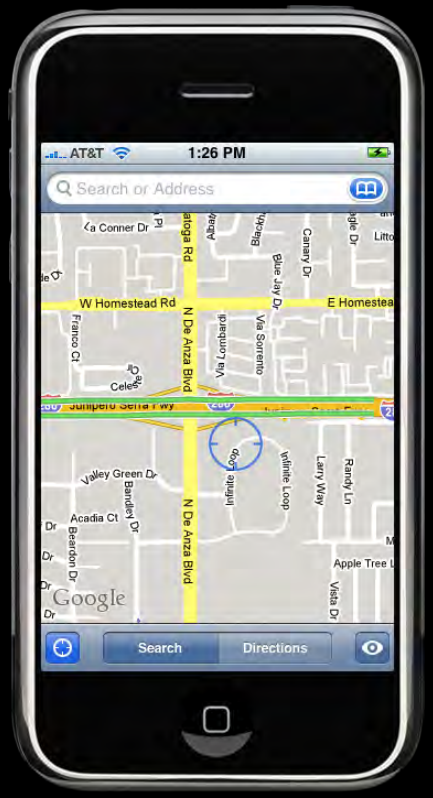
Lots of Cool Features

Device Hardware

Camera

Device Hardware

Core location



Device Hardware

Accelerometers

Limited Simulator Support

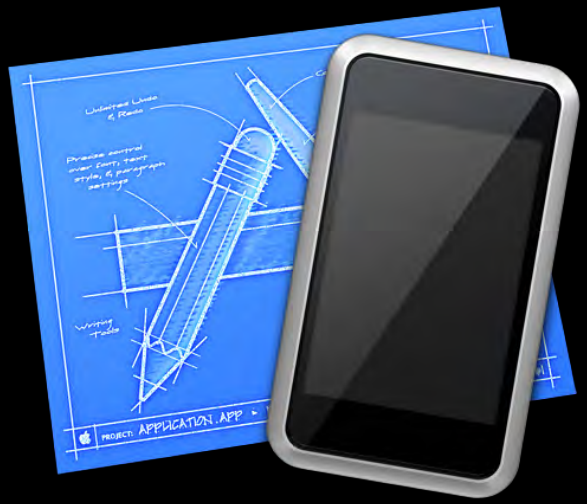


Image Picker

The Image Picker Interface

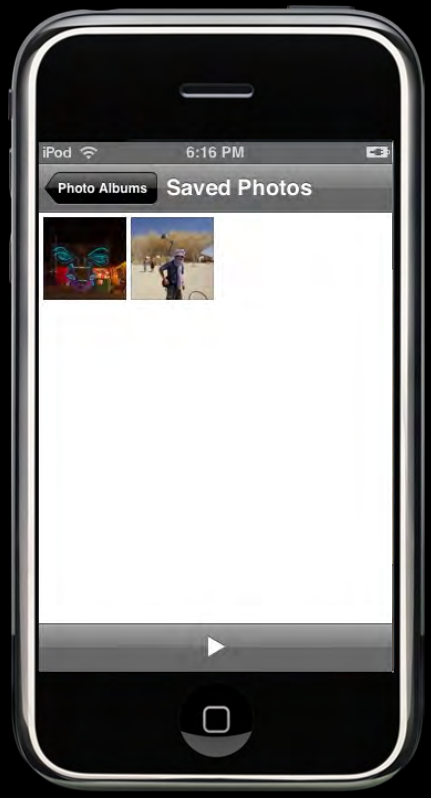
The camera

カメラから画像選択

The Image Picker Interface

Saved photos

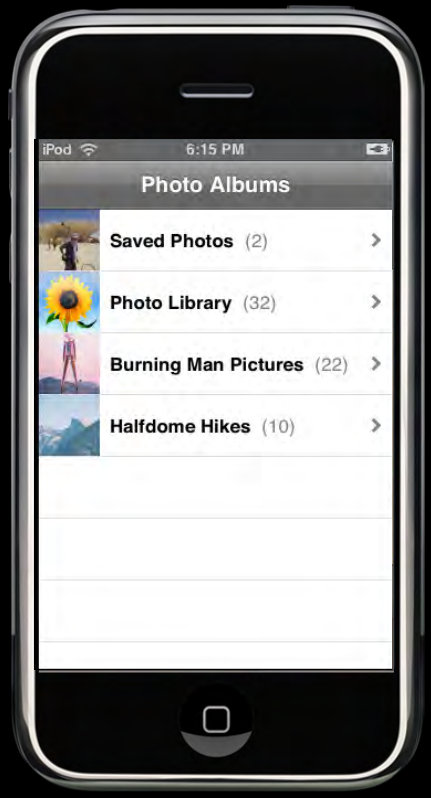
保存された写真から画像選択



The Image Picker Interface

The photo library

写真アルバムから画像選択



The Image Picker Interface

Displaying the interface

- UIImagePickerController class
  - Use as-is (no subclassing)
  - Handles all user and device interactions
  - UIViewController Subclass
- UIImagePickerControllerDelegate protocol
  - Implemented by your delegate object

サブクラス化せずにそのまま使う

デリゲートを指定して
自分のコントローラで
メソッドを処理

Displaying the Image Picker

イメージピッカーの表示

Steps for using

- Check the source availability
- Assign a delegate object
- Present the controller modally

画像取得元が利用可能かをチェック

デリゲートを指定

モーダルビューとしてピッカーコントローラを表示

Displaying the Image Picker

Called from a view controller object

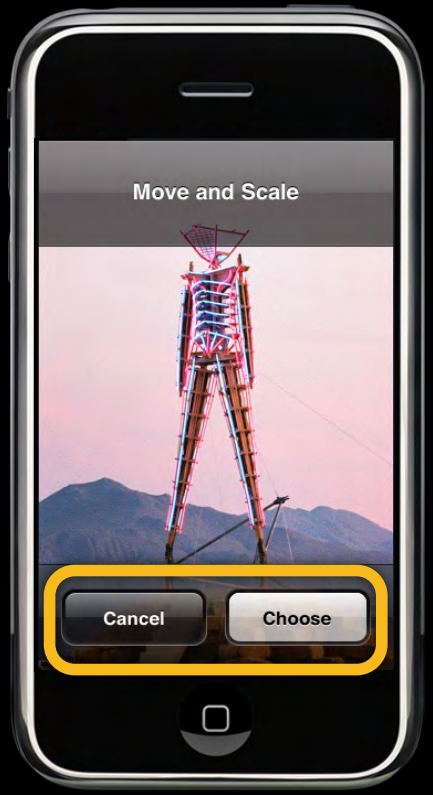
```
if ([UIImagePickerController isSourceTypeAvailable:
                UIImagePickerControllerSourceTypeCamera])
{
    UIImagePickerController* picker =
                [[UIImagePickerController alloc] init];
    picker.sourceType = UIImagePickerControllerSourceTypeCamera;
    picker.delegate = self;

    [self presentModalViewController:picker animated:YES];
}
```

カメラの存在をチェック

ピッカーを生成設定

モーダルビューとしてピッカーを表示

Selecting an Image



Defining Your Delegate Object

The UIImagePickerControllerDelegate protocol

- Two methods:

画像が選択されたときに呼び出されるメソッド

```
- (void)imagePickerController:(UIImagePickerController*)picker
      didFinishPickingImage:(UIImage*)image
                editingInfo:(NSDictionary*)editingInfo;
```

キャンセル時に呼び出されるメソッド

```
- (void)imagePickerControllerDidCancel:
                (UIImagePickerController*)picker;
```

Defining Your Delegate Object

The accept case

画像が選択されたとき

```
- (void)imagePickerController:(UIImagePickerController*)picker
        didFinishPickingImage:(UIImage*)image
                  editingInfo:(NSDictionary*)editingInfo
{
    // Save or use the image here.

    // Dismiss the image picker.
    [self dismissModalViewControllerAnimated:YES];
    [picker release];
}
```

画像を煮るなり焼くなり...

モーダルビューを閉じる

Defining Your Delegate Object

The cancel case

キャンセル時

```
- (void)imagePickerControllerDidCancel:
                (UIImagePickerController*)picker
{
    // Dismiss the image picker.
    [self dismissModalViewControllerAnimated:YES];
    [picker release];
}
```

Manipulating the Returned Image

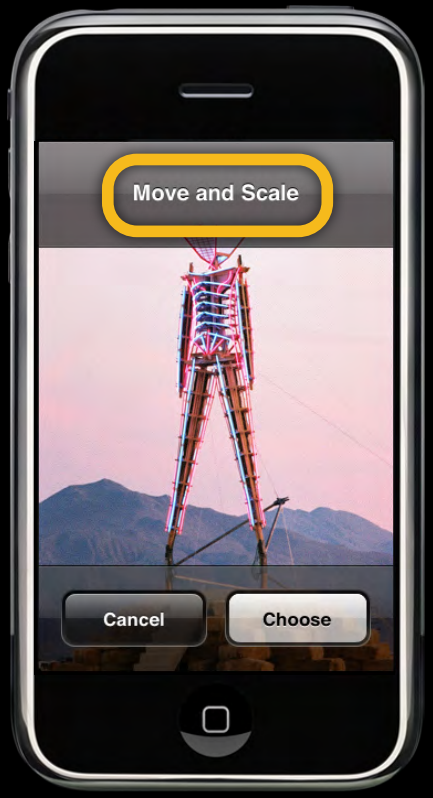
Allowing users to edit returned images

- If allowsImageEditing property is YES:
  - User allowed to crop the returned image
  - Image metadata returned in editingInfo

編集可能フラグが YES なら
ユーザは部分選択可能

Manipulating the Returned Image

Allowing users to edit returned images



Manipulating the Returned Image

The editingInfo dictionary

```
- (void)imagePickerController:(UIImagePickerController*)picker
        didFinishPickingImage:(UIImage*)image
                  editingInfo:(NSDictionary*)editingInfo
{
    // Save or use the image here.

    // Dismiss the image picker.
    [self dismissModalViewControllerAnimated:YES];
    [picker release];
}
```

Manipulating the Returned Image

The editingInfo dictionary

- Original image in UIImagePickerControllerOriginalImage key
- Crop rectangle in UIImagePickerControllerCropRect key

これらのキーを与えれば，
　　オリジナル画像，編集後の Rect
が得られる

Augmented Reality

Walk around looking through a camera.
What could possibly go wrong?

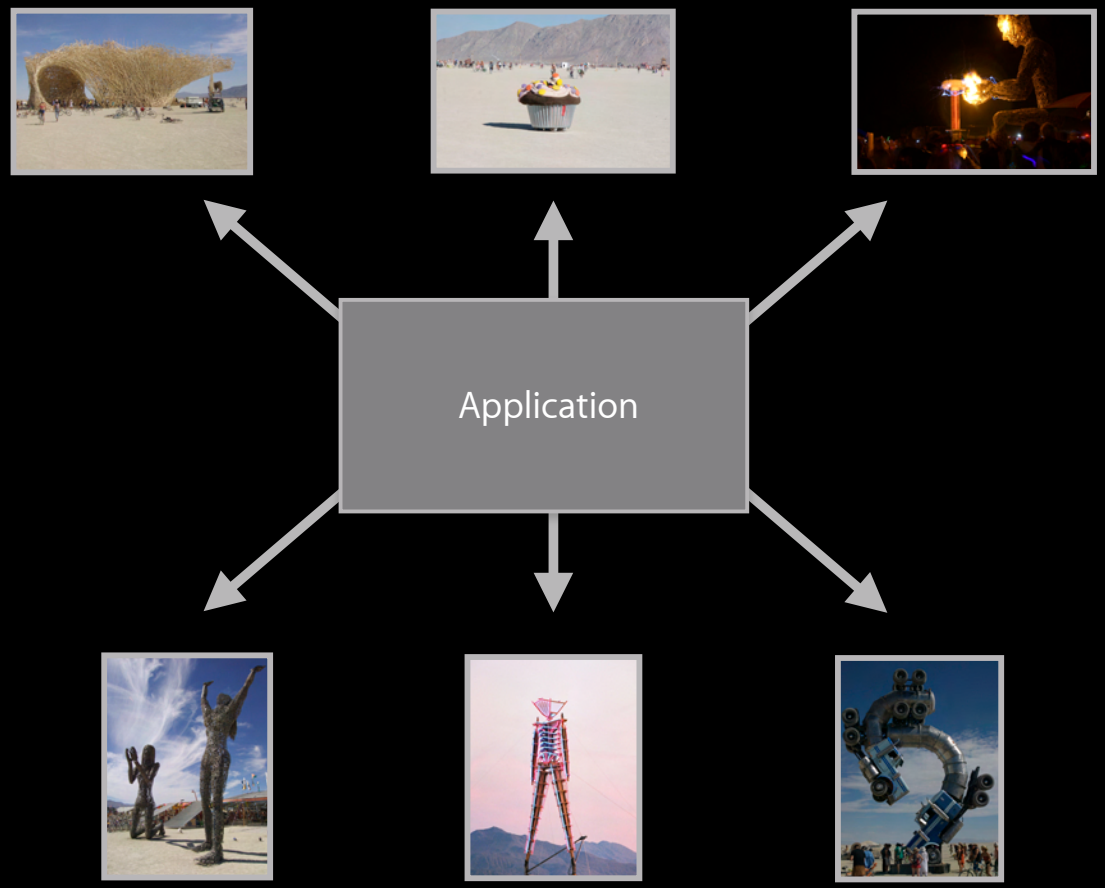
```
@property          BOOL                 showsCameraControls;
@property(retain)  UIView               cameraOverlayView;
@property          CGAffineTransform    cameraViewTransform;
```

The Image Picker Interface

Custom Camera Interface

Managing Image Data

Avoid retaining images



Saving Images

Writing to the saved photos album 写真アルバムへの保存

- UIImageWriteToSavedPhotosAlbum function
  - Photos can be downloaded to iPhoto by user
  - Optional completion callback

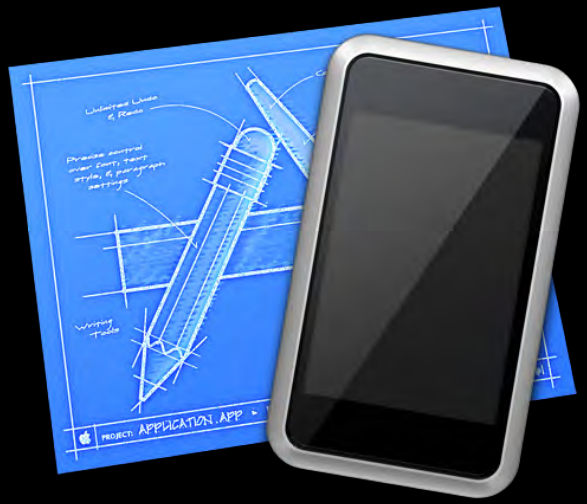
Saving Videos

Writing to the saved photos album 写真アルバムへの保存

- UIVideoAtPathIsCompatibleWithSavedPhotosAlbum

- UISaveVideoAtPathToSavedPhotosAlbum function
  - Videos can be downloaded to iPhoto by user
  - Optional completion callback

Available in the Simulator



Key Tips

Using UIImagePickerController effectively

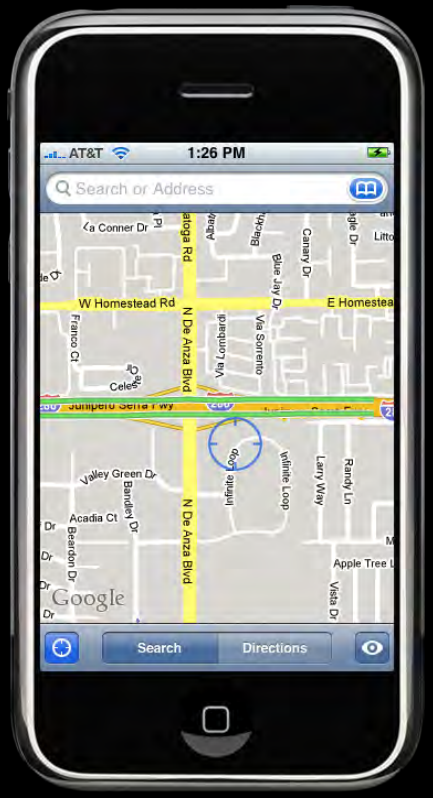
- Always check the source availability
- Your delegate methods do the cleanup
- Be frugal with images
- Available in the simulator

カメラ存在のチェックを
忘れずに

Core Location

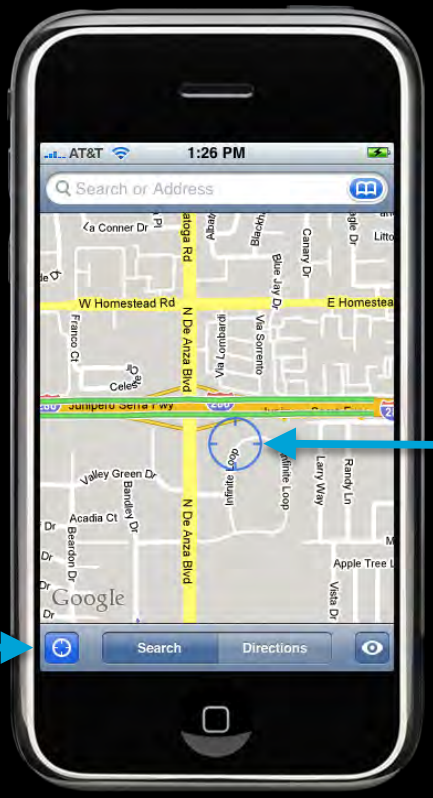
Core Location

What is it?



Core Location

What is it?



Location ring

Activate service

Core Location

How?

携帯基地局

Core Location

How?

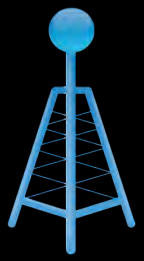
Wifi の
アクセスポイント

Core Location

How?

Core Location

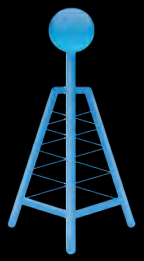
Location Technologies



Bootstrap

Core Location

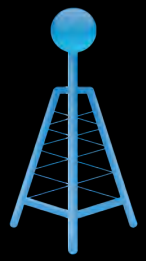
Location Technologies



Cross-check

Core Location

Location Technologies



Complement

Core Location Framework

Core Location Framework

The core classes and protocols

- Classes
  - CLLocationManager
  - CLLocation
- Protocol
  - CLLocationManagerDelegate　デリゲートのプロトコル

Core Location Framework

CLLocationManagerDelegate protocol

- Two optional methods

```
- (void)locationManager:(CLLocationManager*)manager
       didUpdateToLocation:(CLLocation*)newLocation
              fromLocation:(CLLocation*)oldLocation;
```

位置に変化があったとき

```
- (void)locationManager:(CLLocationManager*)manager
         didFailWithError:(NSError*)error;
```

失敗したとき

- Called asynchronously on main thread
- Issues movement-based updates

Getting a Location

Starting the location service

```
CLLocationManager* locManager =
                   [[CLLocationManager alloc] init];

locManager.delegate = self;
[locManager startUpdatingLocation];
```

位置管理者の生成

通知先を設定スタート

Getting a Location

Using the event data

```
- (void)locationManager:(CLLocationManager*)manager
       didUpdateToLocation:(CLLocation*)newLocation
              fromLocation:(CLLocation*)oldLocation
{
    NSTimeInterval howRecent =
             [newLocation.timestamp timeIntervalSinceNow];
    if (howRecent < -10) return;

    if (newLocation.horizontalAccuracy > 100) return;

   // Use the coordinate data.
   double lat = newLocation.coordinate.latitude;
   double lon = newLocation.coordinate.longitude;
}
```

位置に変化があったとき

誤差100m超

Getting a Heading

Using the event data

```
- (void)locationManager:(CLLocationManager *)manager
      didUpdateHeading:(CLHeading *)newHeading
{
   // Use the coordinate data.
   CLLocationDirection heading = newHeading.trueHeading;
}
```

コンパス
方角に
変化

Power Play (beat Canada again): CLLocationManager Properties

Desired Accuracy

Choosing an appropriate accuracy level

```
CLLocationManager* locManager =
                   [[CLLocationManager alloc] init];

locManager.desiredAccuracy = kCLLocationAccuracyBest;
```

- Choose an appropriate accuracy level
  - Higher accuracy impacts power consumption
  - Lower accuracy is "good enough" in most cases
- Can change accuracy setting later if needed
- Actual accuracy reported in CLLocation object

Distance Filter

Choosing an appropriate update threshold

```
CLLocationManager* locManager =
                   [[CLLocationManager alloc] init];

locManager.distanceFilter = 3000;
```

- New events delivered when threshold exceeded

Stopping the Service

```
CLLocationManager* locManager =
                   [[CLLocationManager alloc] init];
[locManager startUpdatingLocation];

...

[locManager stopUpdatingLocation];
```

- Restart the service later as needed

Responding to Errors

User may deny use of the location service

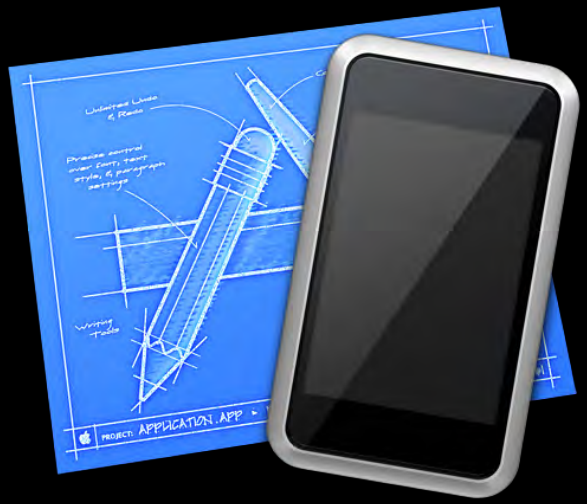
- Results in a `kCLErrorDenied` error
- Protects user privacy
- Occurs on a per-application basis

Responding to Errors

Location may be unavailable

- Results in a `kCLErrorLocationUnknown` error
- Likely just temporary
- Scan continues in background

Limited Simulator Support



Accelerometers

加速度センサ

What Are Accelerometers?

Measure changes in force

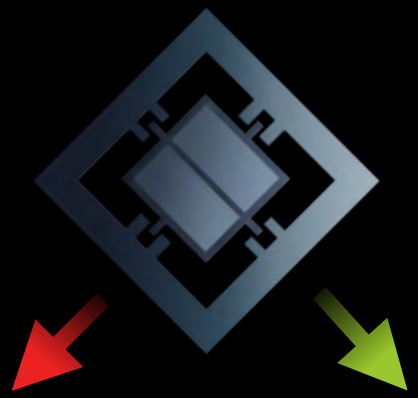
（おもりにかかる）力を計測

What Are Accelerometers?

Measure changes in force

What Are Accelerometers?

Measure changes in force



Accelerometers

What are the uses?

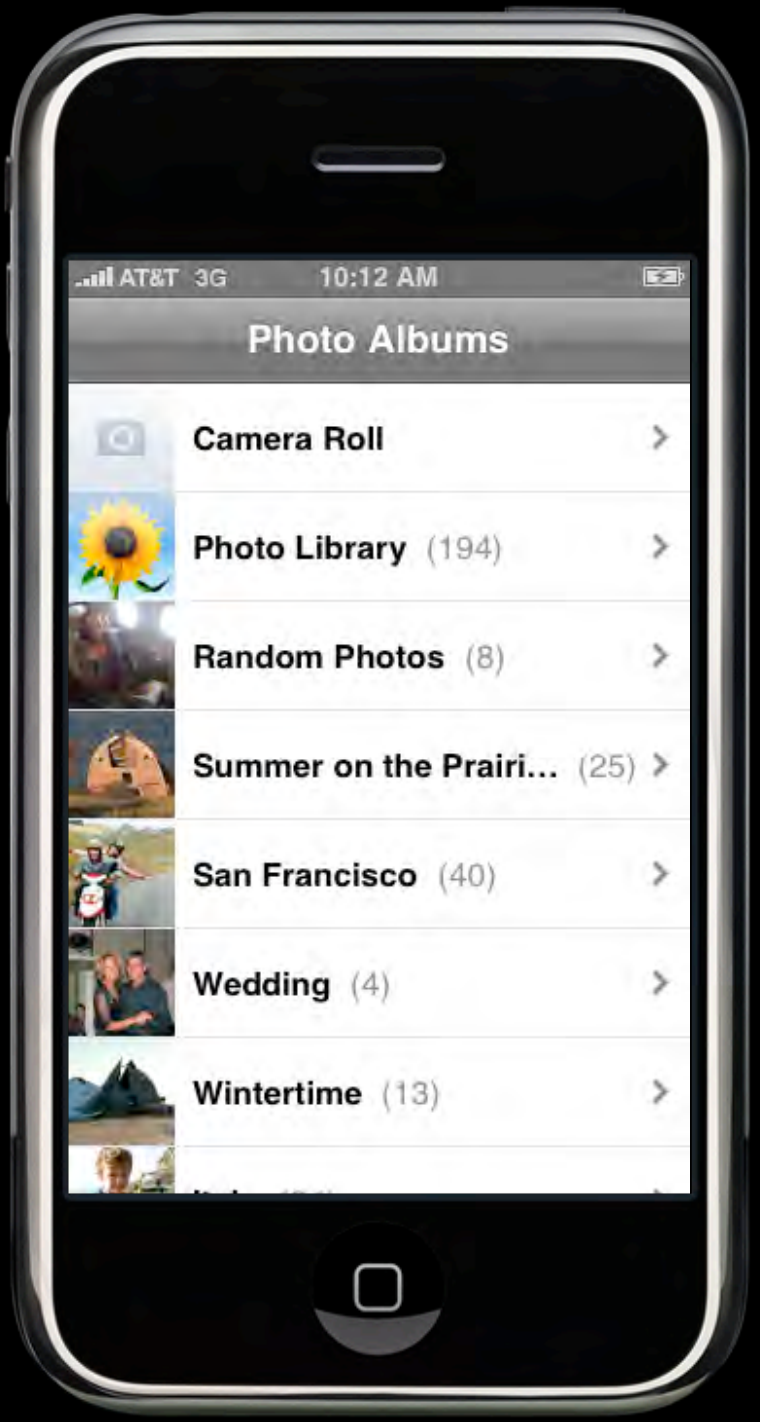
Accelerometers

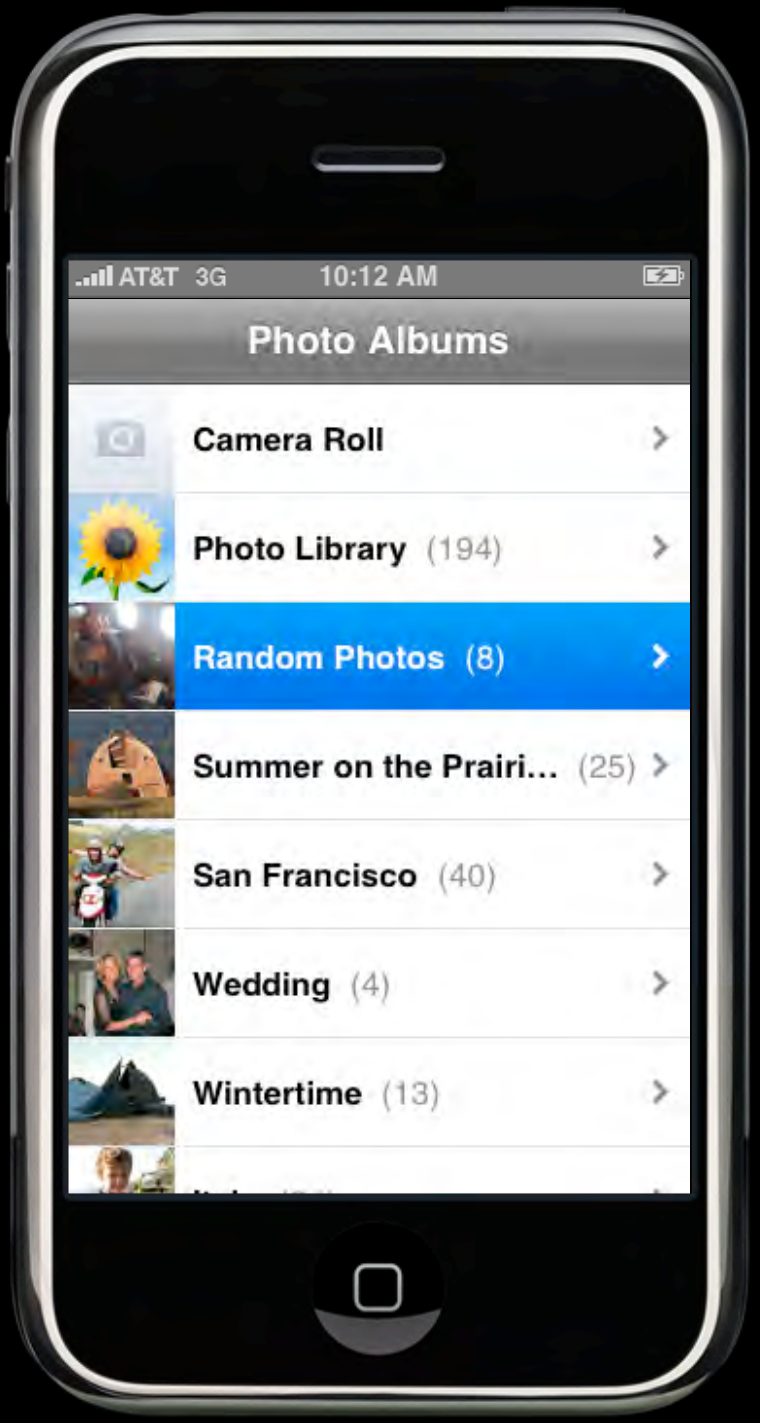
What are the uses?

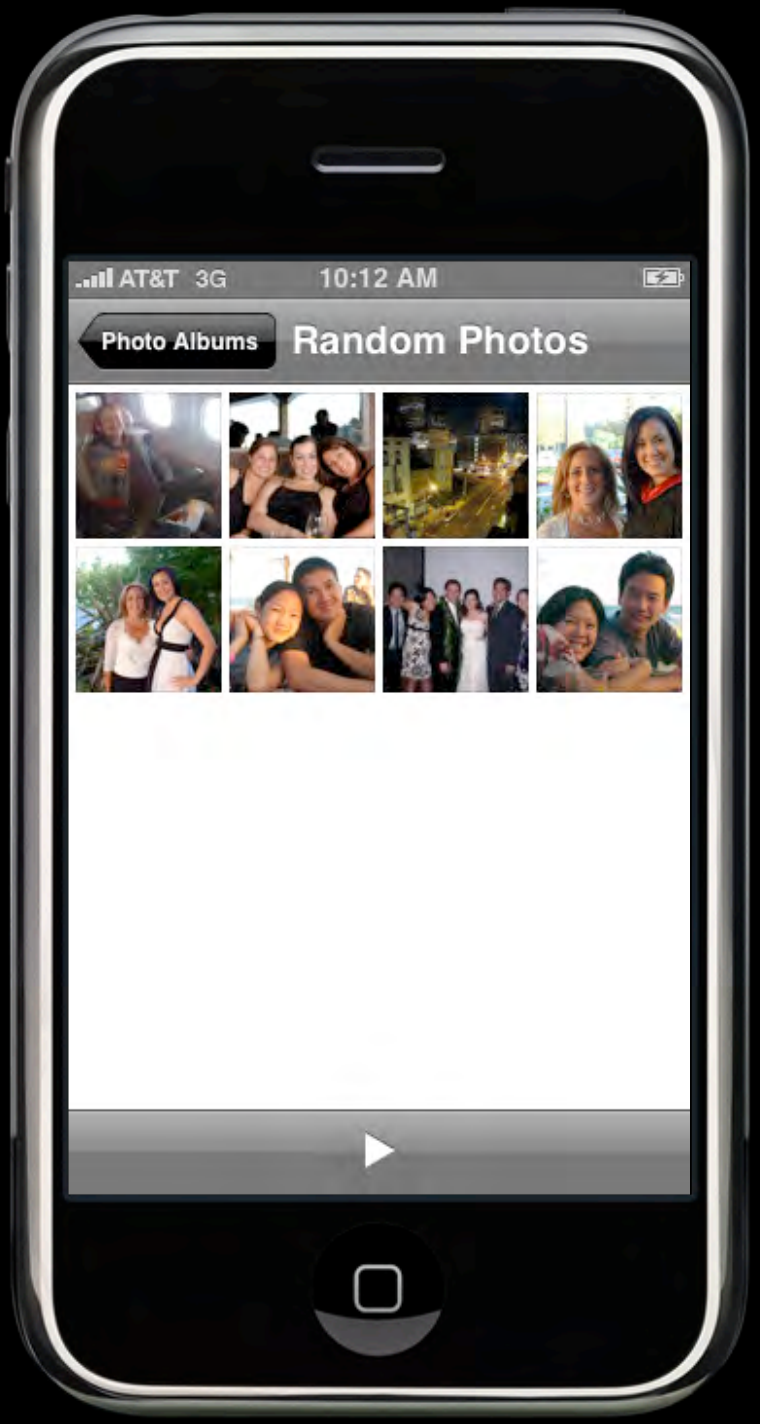
Kinds of Orientation

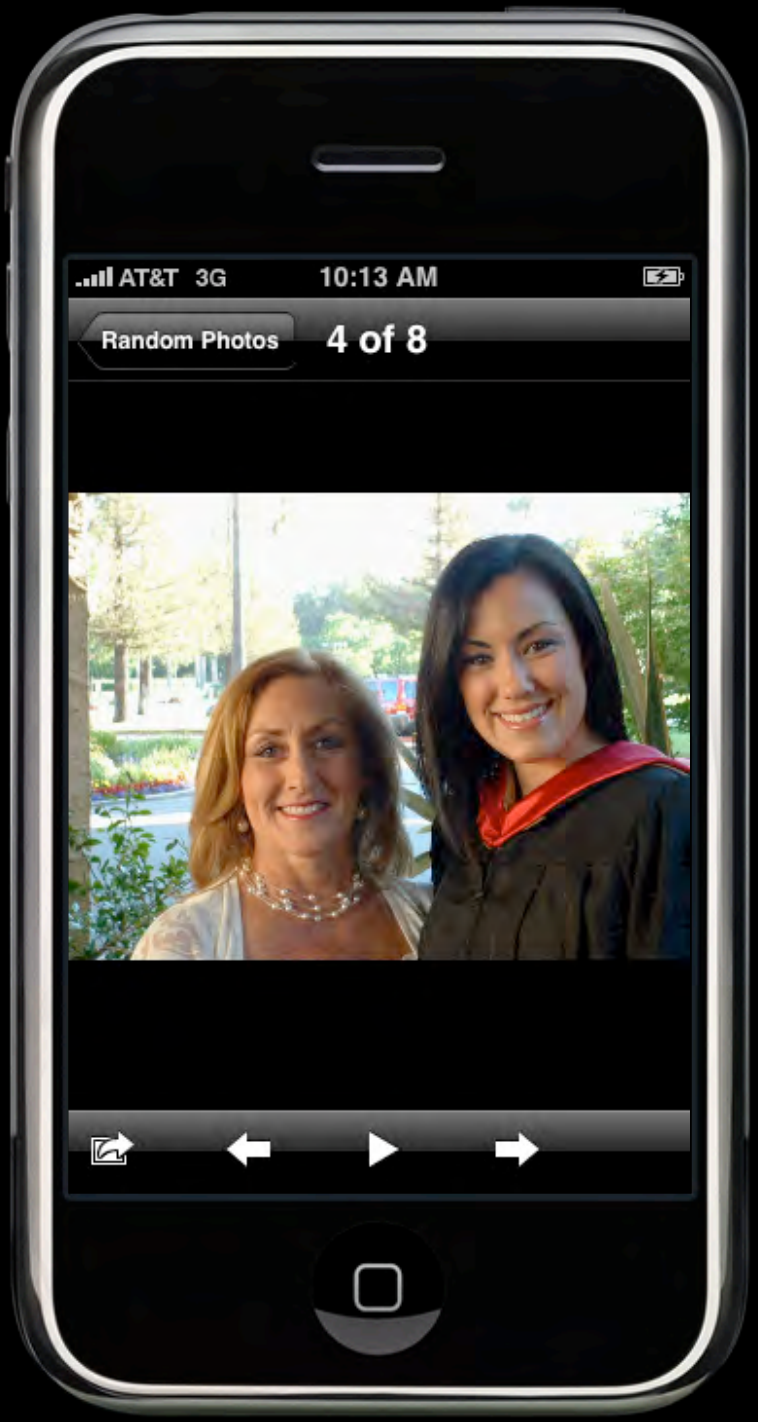
The physical vs the interface

- Physical Orientation
  - How is the device positioned?
- Interface Orientation
  - Where is the status bar?
- Examples: Photos & Safari


AT&T 3G
10:12 AM
Photo Albums
Camera Roll
Photo Library (194)
Random Photos (8)
Summer on the Prairi... (25)
San Francisco (40)
Wedding (4)
Wintertime (13)


AT&T 3G
10:12 AM
Photo Albums
Camera Roll
Photo Library (194)
Random Photos (8)
Summer on the Prairi... (25)
San Francisco (40)
Wedding (4)
Wintertime (13)


AT&T 3G
10:12 AM
Photo Albums
Random Photos


AT&T 3G
10:13 AM
Random Photos
4 of 8


AT&T 3G
10:13 AM
4 of 8
Random Photos


AT&T 3G
10:13 AM
Apple - iPhone
www.apple.com/iphone/
Google
iPhone 3G
35,000 apps. And counting.


AT&T 3G
10:13 AM
Apple - iPhone
www.apple.com/iphone/
Google
iPhone 3G
35,000 apps. And counting.


AT&T 3G
10:13 AM
Apple - iPhone
www.apple.com/iphone/
Google

Orientation-Related Changes

Getting the physical orientation

- UIDevice class
  - Start notifications
    - `beginGeneratingDeviceOrientationNotifications`
  - Get Orientation
    - `UIDeviceOrientationDidChangeNotification` delivered to registered observers
    - `orientation` property
  - Stop notifications
    - `endGeneratingDeviceOrientationNotifications`

Orientation-Related Changes

Getting the interface orientation

- UIApplication class
  - `statusBarOrientation` property
  - Defines interface orientation, not device orientation

- UIViewController class
  - `interfaceOrientation` property

```
- (BOOL)shouldAutorotateToInterfaceOrientation:
                    (UIInterfaceOrientation)interfaceOrientation
```

Shake 振る（シェイク！＝Undo！）

Undo!

- UIEvent type
  - `@property(readonly) UIEventType    type;`
  - `@property(readonly) UIEventSubtype subtype;`

  - `UIEventTypeMotion`
  - `UIEventSubtypeMotionShake`

Orientation changes are nice, but…

向きが分かるのもいいけれど…

Wii™ Want Raw Data

生データを欲しいよね

0.75g

1.0g

0.5g

Wii™ Want Raw Data

0.75g

1.0g

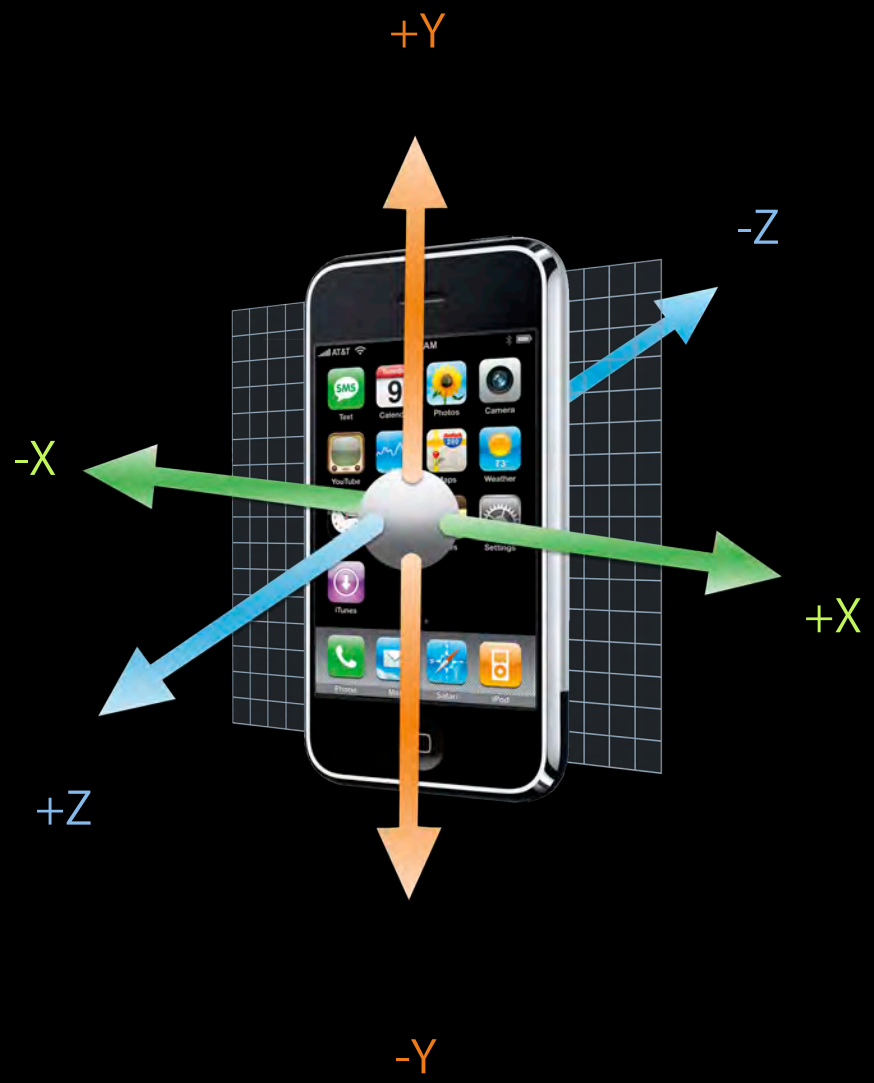
0.5g

The Accelerometer Interface

Getting the raw accelerometer data

- Part of the UIKit framework
- Delivers 3-axis data　３軸それぞれの生データ
- Configurable update frequency (approx 10–100Hz)
- Delegate-based event delivery　デリゲートで通知される

Device Axis Orientation



The Accelerometer Interface

Getting the raw accelerometer data

- Classes
  - UIAccelerometer
  - UIAcceleration
- Protocol
  - UIAccelerometerDelegate　デリゲートのプロトコル

Configuring the Accelerometer

Starting the event delivery

```
- (void)enableAccelerometerEvents
{
    UIAccelerometer* theAccel =
                        [UIAccelerometer sharedAccelerometer];
    theAccel.updateInterval = 1/50;  // 50 Hz
    theAccel.delegate = self;
}
```

まずは仕込み
（イベント通知を設定）

Event delivery begins as soon as you assign the delegate

Defining Your Delegate Object

Processing the accelerometer data

```
- (void)accelerometer:(UIAccelerometer*)accelerometer
        didAccelerate:(UIAcceleration*)acceleration
{
    // Get the event data
    UIAccelerationValue  x, y, z;

    x = acceleration.x;
    y = acceleration.y;
    z = acceleration.z;

    // Process the data...
}
```

設定しておいた
時間間隔で
このメソッドが
呼び出される

- Only one delegate per application
- Delivered asynchronously to main thread

Configuring the Accelerometer

Choosing an appropriate update frequency

- System range is approximately 10–100Hz
- Frequency should be based on need 必要に応じた時間間隔で
  - Determine the minimum frequency for your needs
  - Don't update too frequently
- Target ranges
  - Game input: 30–60 Hz
  - Orientation detection: 10–20 Hz

Disabling Event Delivery

Stopping the event delivery

イベント通知をストップさせるには

```
- (void)disableAccelerometerEvents
{
    UIAccelerometer* theAccel =
                        [UIAccelerometer sharedAccelerometer];

    theAccel.delegate = nil;
}
```

Filtering Accelerometer Data

Use filters to isolate data components

- Low-pass filter
  - Isolates constant acceleration
  - Used to find the device orientation
- High-pass filter
  - Shows instantaneous movement only
  - Used to identify user-initiated movement

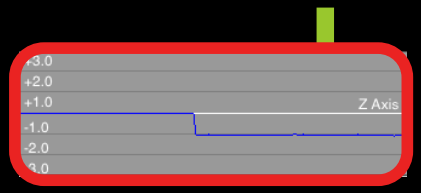
向きを知るには
ローパスフィルタ

ジェスチャを知るには
ハイパスフィルタ

Filtering Accelerometer Data

Examining the accelerometer data

-1.0g



f(t)

Filtering Accelerometer Data

But, to apply a filter…

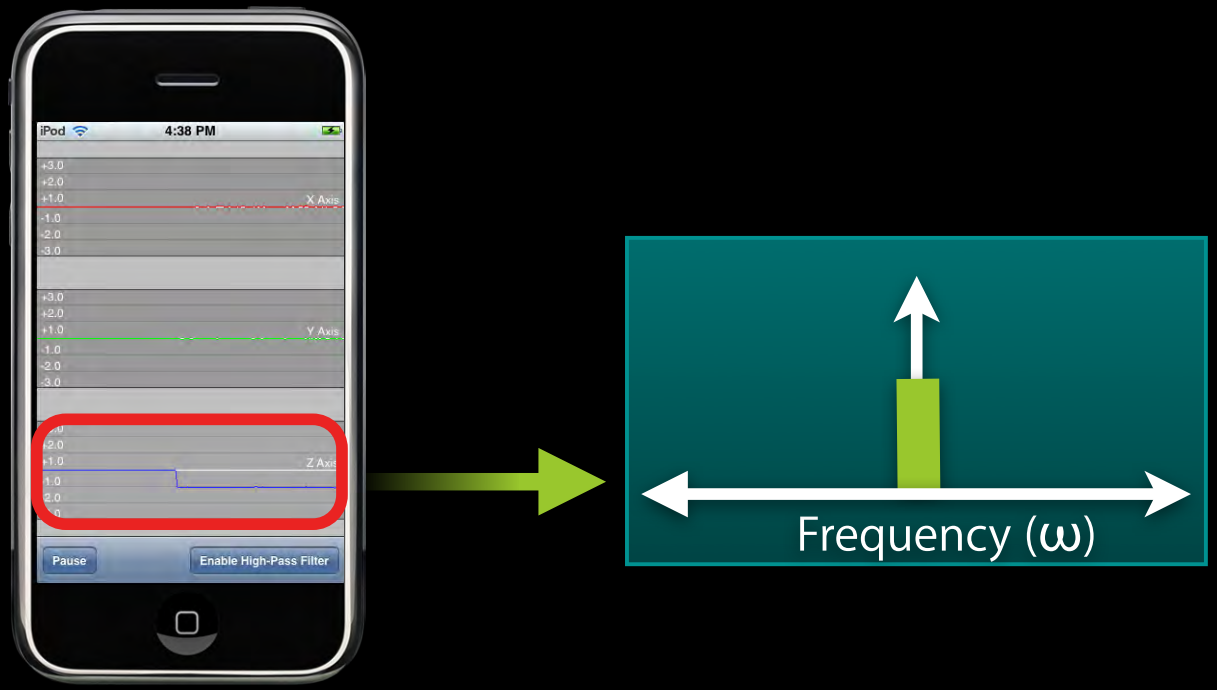
$$f(t) \Rightarrow F(\omega)$$

Fourier Transform

フーリエ変換すると

Filtering Accelerometer Data

Changing to the frequency domain

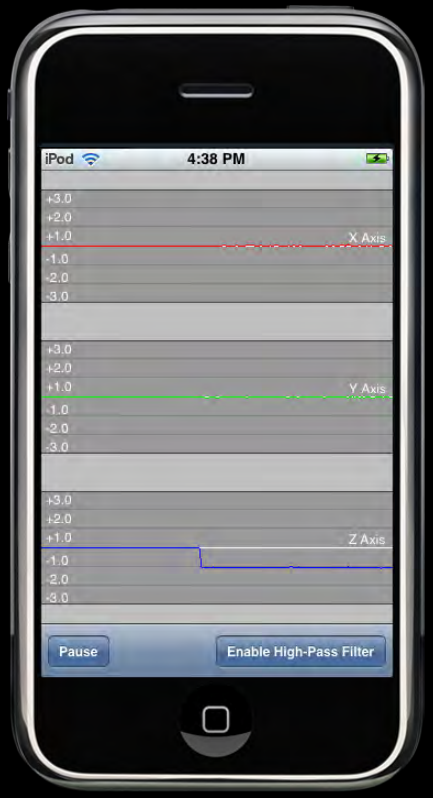


f(t)

F(ω)

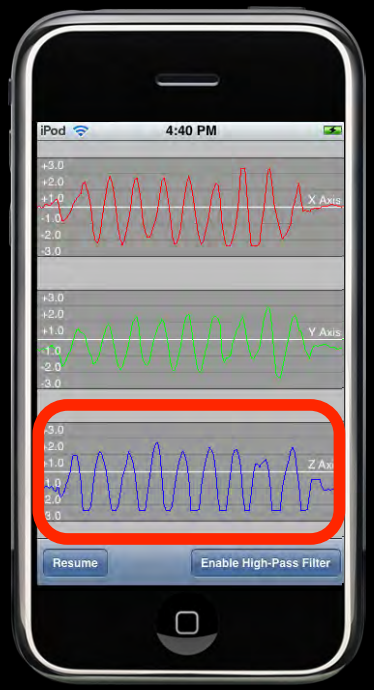
Filtering Accelerometer Data

But if we shake the device…

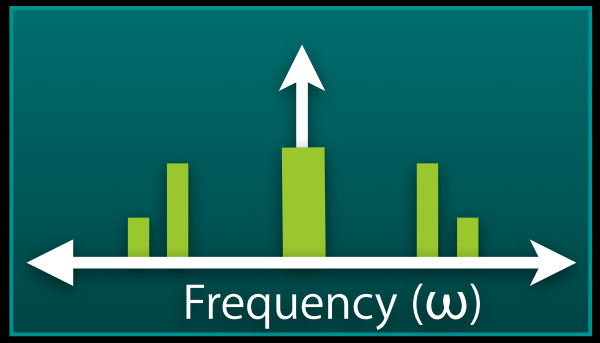


Filtering Accelerometer Data

We see something more interesting…



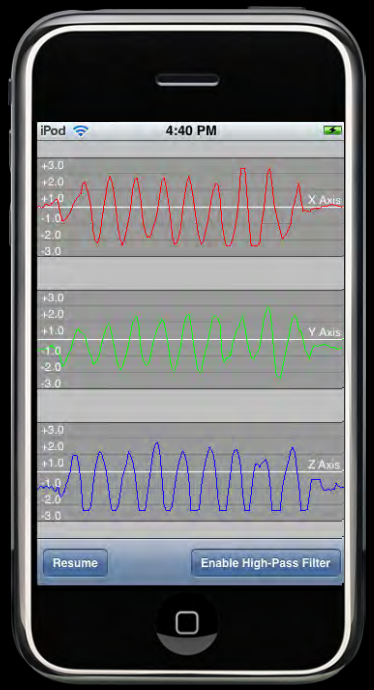
f(t)



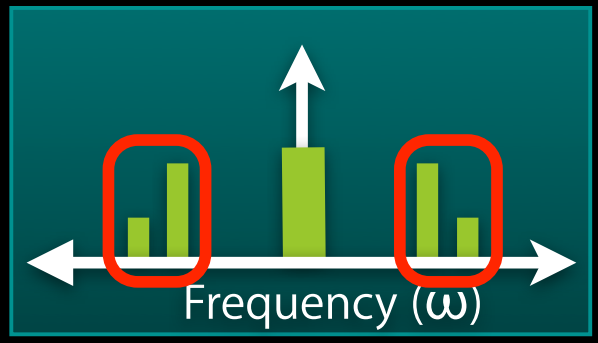
F(ω)

Filtering Accelerometer Data

We see something more interesting…



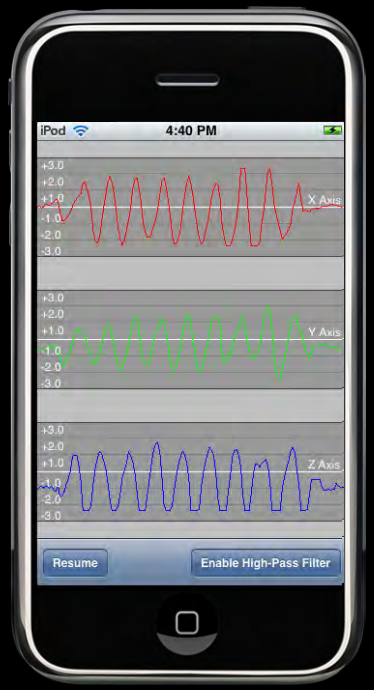
f(t)



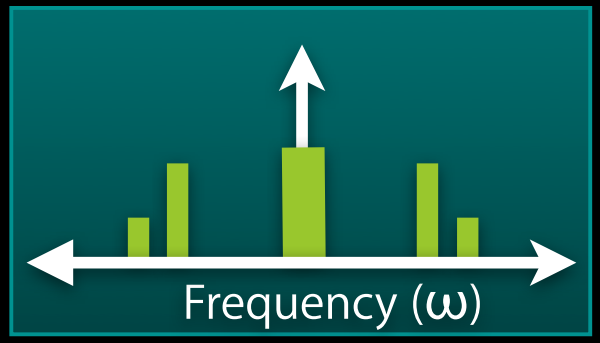
F(ω)

Filtering Accelerometer Data

Applying a low-pass filter



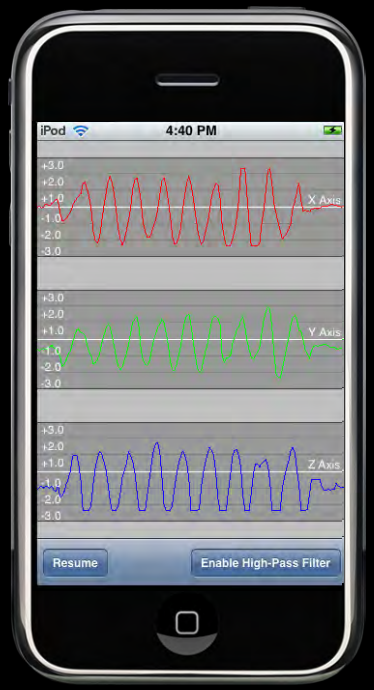
f(t)



F(ω)

Filtering Accelerometer Data

Applying a low-pass filter

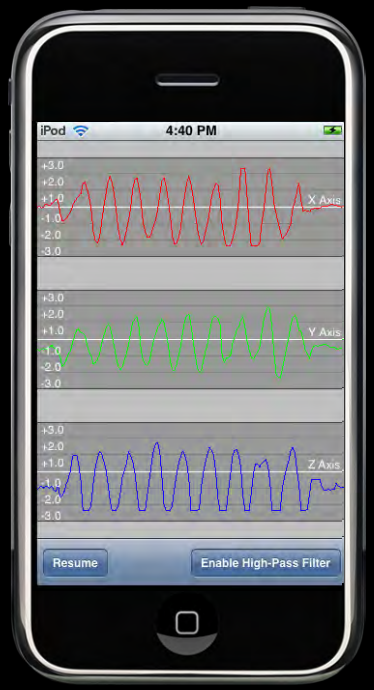


f(t)

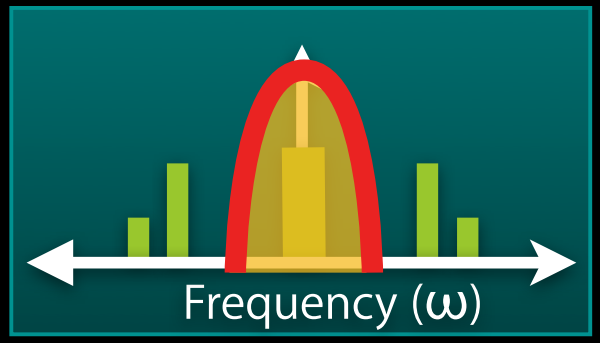
F(ω)

Filtering Accelerometer Data

Applying a low-pass filter



f(t)



F(ω)

Filtering Accelerometer Data

Applying a low-pass filter

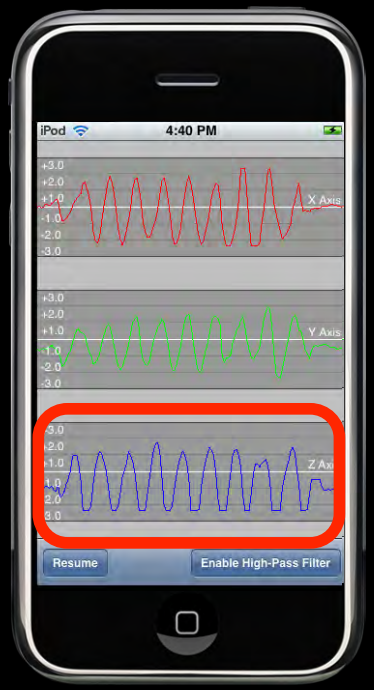
- Simple low-pass filter example

```
#define FILTERFACTOR  0.1

value = (newAcceleration * FILTERFACTOR) +
        (previousValue * (1.0 - FILTERFACTOR));

previousValue = value;
```

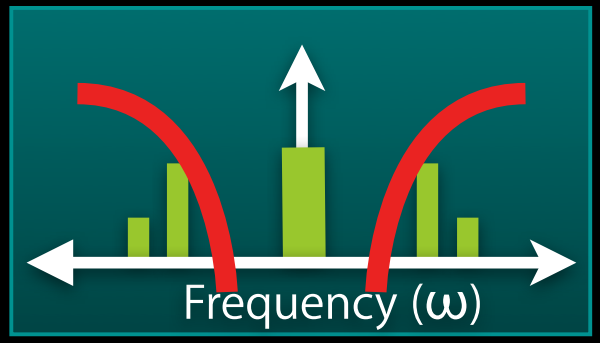
注）ちょっとシンプルすぎるローパスフィルタ
（興味あるひとは FIR をググる）

Filtering Accelerometer Data

Applying a high-pass filter



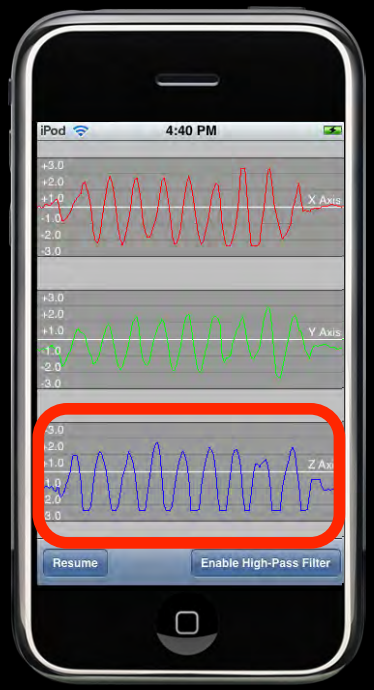
f(t)



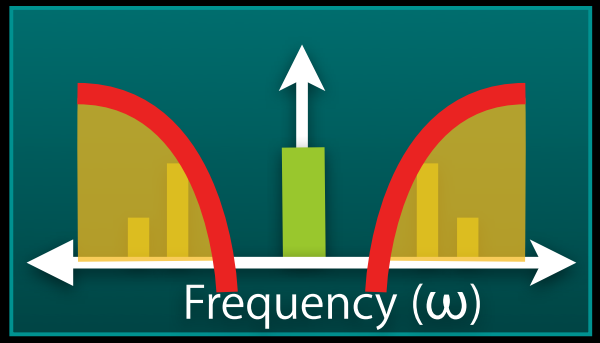
F(ω)

Filtering Accelerometer Data

Applying a high-pass filter



f(t)



F(ω)

Filtering Accelerometer Data

Applying a high-pass filter

- Simple high-pass filter example

```
#define FILTERFACTOR  0.1

lowPassValue = (newAcceleration * FILTERFACTOR) +
               (previousValue * (1.0 - FILTERFACTOR));

previousLowPassValue = lowPassValue;

highPassValue = newAcceleration - lowPassValue;
```

低周波成分を引けば
高周波成分が残る

Filtering Accelerometer Data

Bubble Level sample (low-pass filter)

水平器

Demo

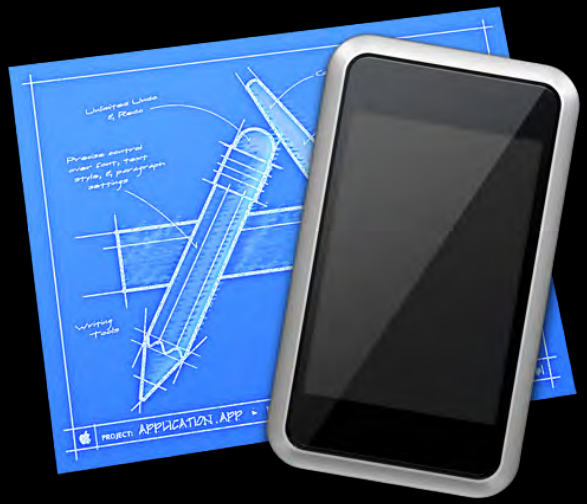
Filtering Accelerometer Data

Bubble Level sample (low-pass filter)

```
- (void)accelerometer:(UIAccelerometer*)accelerometer
        didAccelerate:(UIAcceleration*)acceleration
{
    accelerationX = acceleration.x * kFilteringFactor +
        accelerationX * (1.0 - kFilteringFactor);
    accelerationY = acceleration.y * kFilteringFactor +
        accelerationY * (1.0 - kFilteringFactor);

    currentRawReading = atan2(accelerationY,accelerationX);
    float calibtratedAngle = [self calibratedAngleFromAngle:
                currentRawReading];

    [levelView updateToInclinationInRadians:calibratedAngle];
}
```

Demo

No Simulator Support



Key Tips

Using the Accelerometers Effectively

- Use UIViewControllers
- Use filters to isolate raw data components
- Disable accelerometer updates when not needed
  - Set your accelerometer delegate to nil

Summary

- Take advantage of the device APIs, but…
- For image picker, always check source availability
- For hardware-based features, turn them off when not needed

Battery Life & Power Management

これについては割愛します

Power Management

Small devices need advanced power management

- Total power consumption
  - Laptops: ~20-60W
  - iPhone: 500 mW to 2.5W
- Dynamic clocking
- Clock gating and power gating
  - Turning blocks on and off continuously

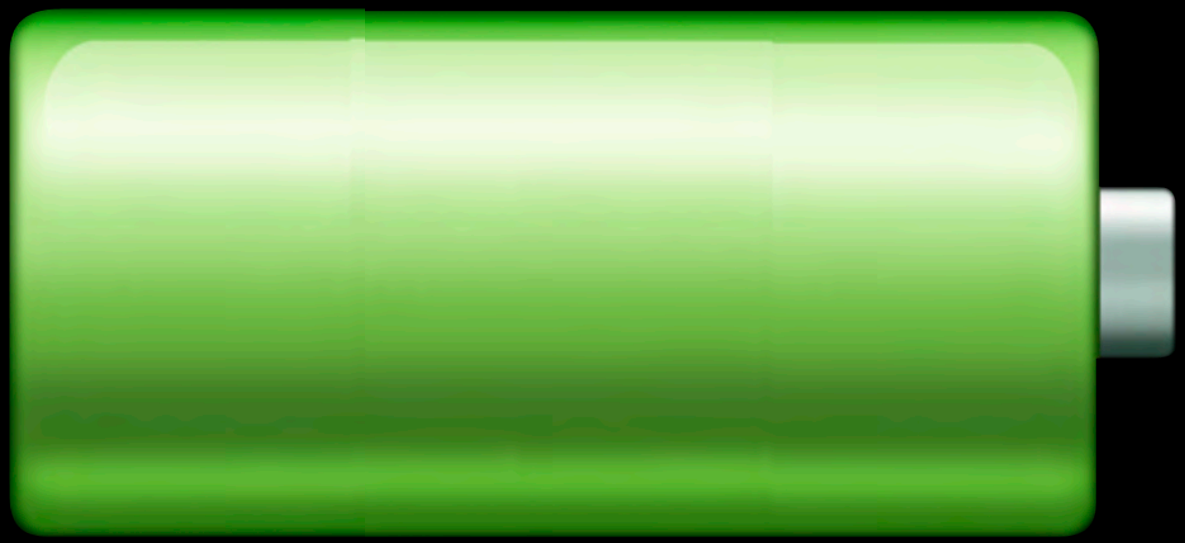
Power Consumption

Everything consumes power

- Radios – up to ~2W
  - Baseband, Wi-Fi, Bluetooth, GPS
- CPU/GPU – up to ~800 mW
- Display – up to ~200 mW
- Hardware modules – ~10s of mWs
- Keeping the system awake – enormous impact

Battery Life

Be aware of power consumption



Battery Life

Be aware of power consumption

Power Consumption - Radios

The network

- Transmitting is the most expensive operation
- Minimize the amount of transmitted data
- Avoid chatty protocols
- Transmit/receive in bursts
- Use compact data formats
- Core Location
  - Stop the location service once you have a location fix
  - Request only the location accuracy that you need

Power Consumption - CPU/GPU

All about performance

- Reduce CPU usage
- Use Sample or Shark
- Stress the GPU less – fewer layers, smaller textures, etc.

Power Consumption - Hardware Modules

Accelerometer, NAND, others

- Turn off what you don't need
- Accelerometer
  - Set the `UIAccelerometer` delegate to `nil`
  - Support orientation changes only as needed
- NAND
  - Access the disk less – use the System Usage instrument

Power Consumption - Standby

Let the system sleep

- Battery life drops from 250+ hours to <12 hours without sleep
- Don’t disable the idle timer
- Don’t play audio except when you need to

Questions?