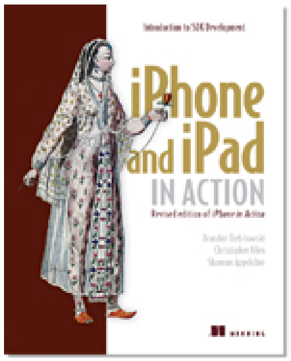

Accessing the UIAccelerometer

When you're using the iPhone's orientation notification, the frameworks are doing the work for you: they're taking low-level acceleration reports and turning them into more meaningful events. It's similar to the concept of iPhone actions, which turn low-level touch events into high-level control events.

WARNING

Accelerometer programs can't be tested on the iPhone Simulator. Instead you'll need to have a fully provisioned iPhone to test out your code. See appendix C for information on provisioning your iPhone.

Notifications aren't going to be sufficient if you would prefer to program entire interfaces that effectively use the iPhone's movement in three-dimensional space as a new user-input device. For that, you'll need to access two iPhone classes: `UIAccelerometer` and `UIAcceleration`. We'll talk about `UIAccelerometer`.

The `UIAccelerometer` is a class that you can use to receive acceleration-related data. It is a shared object, like `UIApplication` and `UIDevice`. The process of using it is shown in listing 1.

Listing 1 Accessing the UIAccelerometer takes just a few steps

```
- (void)viewDidLoad {
    UIAccelerometer *myAccel =
                          #1
        [UIAccelerometer sharedAccelerometer];                   #1
    myAccel.updateInterval = .1;                                 #2
    myAccel.delegate = self;                                     #3
    [super viewDidLoad];
}
```

#1 Creates shared accelerometer
#2 Updates 10 times a second
#3 Delegates accelerometer protocol

The first step is to access the accelerometer, which is done with another call to a shared-object method (#1). Having this step on its own line is probably unnecessary, because you could perform the other two steps as nested calls, but we find this a lot more readable.

Next, you select your update interval (#2), which specifies how often you'll receive information on acceleration. This is hardware limited, with a current default of 100 updates per second. That's most likely just right if you're creating a game using the accelerometer, but excessive for other purposes. We've opted for 10 updates per second, which is an `updateInterval` of `0.1`. You should always set the lowest acceptable input to preserve power on the iPhone.

Finally, you must set a delegate for the accelerometer (#3), which is how you receive data on accelerometer changes. The delegate will need to respond to only one method, `accelerometer:didAccelerate:`, which sends a message containing a `UIAcceleration` object whenever acceleration occurs (to the limit of the `updateInterval`).

Summary

The iPhone's accelerometers can give you access to a variety of information about where an iPhone exists in space. By measuring gravity, you can easily discover an iPhone's precise orientation. By measuring movement, you can see how an iPhone is being guided through space. Finally, you can build more complex movements into three-dimensional gestures, such as the shake.

Accelerometers allow users to make simple adjustments to a program. We can imagine game controls and painting programs both built entirely around the accelerometers.