References

Beginning iPhone 3 Development: Exploring the iPhone SDK by Jeff LaMarche, and David Mark, Chapter 4

Various Apple iOS documentation
Existing Views & Controls

Buttons
Text Fields
Text view
Switch
Slider
Image view
Search bar
Pickers
Segment Control
Toolbar
Alerts & Actionsheets
Web view
Views, Controls & Controllers

Views
   Subclass of UIView
   Label, Web view, Toolbar, etc

Controls
   Subclass of UIControl
   Button, date picker, slider, text field, switch

Controllers
   Subclass of UIViewController
UICatalog

Buttons
Various uses of UIButton

Controls
Various uses of UIControl

TextFields
Uses of UITextField

SearchBar
Use of UISearchBar

TextView
Use of UITextField

Pickers
Uses of UIDatePicker, UIPickerView

Images
Use of UIImageView

Web
Use of UIWebView

Segment
Various uses of UISegmentedControl

ToolBar

Active, Static and Passive (Views & Controls)

Active
  User does something
  Code executes in response
  Button

Static
  Displays something
  User cannot interact with it
  Program can change it
  Label

Passive
  User can
TextField Example
@interface TextAlertViewController : UIViewController {

@property (nonatomic, retain) IBOutlet UITextField *sampleField;
@property (nonatomic, retain) IBOutlet UITextField *echoField;

- (IBAction) donePressed;
- (IBAction) editDone;

@end
#import "TextAlertViewController.h"

@implementation TextAlertViewController

@synthesize sampleField;
@synthesize echoField;

- (IBAction) donePressed {
    [self editDone];
}

- (IBAction) editDone {
    echoField.text = sampleField.text;
}
How to know when user is done editing?

TextField events

Edit Did Begin
    When field gets focus

Edit Changed
    When any change is made
    Characters added/removed, cursor changes position

Edit Did End
    When focus leaves field
    Tab to another field

Did End On Exit
    When user tabs on "return" or "done" key on keyboard
How to know when user is done editing?

Hard to tell by just text field events

Provide some other way for user to indicate done
Keyboard

When textfield gets focus keyboard appears

Your code has to hide it

Multiple keyboards available
First Responder

UI element the user is interacting with

Window currently the focus for user events
Hiding the Keyboard

When text field is notified that should stop being first responder

Send resignFirstResponder to active text field
- (IBAction) donePressed {
    [self editDone];
    [sampleField resignFirstResponder];
    [echoField resignFirstResponder];
}
Do we have to list all text fields?

UIView class has method "firstResponder"

- (IBAction) donePressed {
    [self editDone];
    UIView * firstResponder = [[self view] firstResponder];
    if( [firstResponder isKindOfClass:[UITextField class]] )
        [firstResponder resignFirstResponder];
}

But ...
firstResponder is not in public API

Apple rejects apps that use non public API methods
Question for thought

How does one find out about methods not in public API?
```objc
#import <Foundation/Foundation.h>

@interface UIView (FirstResponder)
- (UIView *) getFirstResponder;
@end

@implementation UIView (FirstResponder)

- (UIView *) getFirstResponder
{
    if (self.isFirstResponder) {
        return self;
    }

    for (UIView *subView in self.subviews) {
        UIView *firstResponder = [subView getFirstResponder];
        if (firstResponder != nil) {
            return firstResponder;
        }
    }

    return nil;
}
@end
```

Adapted from http://stackoverflow.com/questions/949806/is-there-a-way-to-detect-what-uiview-is-currently-visible
Legal Solution

- (IBAction) donePressed {
    [self editDone];
    UIView * firstResponder = [[self view] getFirstResponder];
    if( [firstResponder isKindOfClass:[UITextField class]] )
        [firstResponder resignFirstResponder];
}

- (IBAction) editDone {
    echoField.text = sampleField.text;
}
- (IBAction) backgroundTapped {
    [self hideKeyboard];
}

- (void) hideKeyboard {
    UIView * firstResponder = [[self view] getFirstResponder];
    if( [firstResponder isKindOfClass:[UITextField class]] )
        [firstResponder resignFirstResponder];
}

- (IBAction) donePressed {
    [self editDone];
    [self hideKeyboard];
}

- (IBAction) editDone {
    echoField.text = sampleField.text;
}
You have to change the class of the View to be UIControl
Some Keyboard Types

UIKeyboardTypeASCIICapable

URL

Numbers

Number Pad
Keyboard Type in IB
- (void)viewDidLoad {
    [super viewDidLoad];
    sampleField.keyboardType = UIKeyboardTypeDefault;
    sampleField.returnKeyType = UIReturnKeyGo;
}
Keyboard Types

UIKeyboardTypeDefault,
UIKeyboardTypeASCIICapable,
UIKeyboardTypeNumbersAndPunctuation,
UIKeyboardTypeURL,
UIKeyboardTypeNumberPad,
UIKeyboardTypePhonePad,
UIKeyboardTypeNamePhonePad,
UIKeyboardTypeEmailAddress,
UIKeyboardTypeAlphabet = UIKeyboardTypeASCIICapable
Return Key Types

UIReturnKeyDefault,
UIReturnKeyGo,
UIReturnKeyGoogle,
UIReturnKeyJoin,
UIReturnKeyNext,
UIReturnKeyRoute,
UIReturnKeySearch,
UIReturnKeySend,
UIReturnKeyYahoo,
UIReturnKeyDone,
UIReturnKeyEmergencyCall,
Other Keyboard Properties

autocapitalizationType
autocorrectionType
enablesReturnKeyAutomatically
keyboardAppearance
returnKeyType
secureTextEntry

See UITextInputTraits Protocol for more details
TextField Delegate

Messages that are part of TextField editing sequence
Allows one to control text field edits

- textFieldShouldBeginEditing:
- textFieldDidBeginEditing:
- textFieldShouldEndEditing:
- textFieldDidEndEditing:

- textField:shouldChangeCharactersInRange:replacementString:
- textFieldShouldClear:
- textFieldShouldReturn:
Controller as TextField Delegate

@interface TextAlertViewController : UIViewController <UITextFieldDelegate>
{
}@property (nonatomic, retain) IBOutlet UITextField *sampleField;
@property (nonatomic, retain) IBOutlet UITextField *echoField;
- (IBAction) backgroundTapped;
- (IBAction) donePressed;
- (IBAction) editDone;
- (IBAction)textFieldDoneEditing:(id)sender;
@end
API does not support multi-line text fields

See:
http://www.hanspinckaers.com/multi-line-uitextview-similar-to-sms