In this project we will examine **how to reverse a string in an iOS application using Category**. The concept is to experiment with creating a Category for a project.

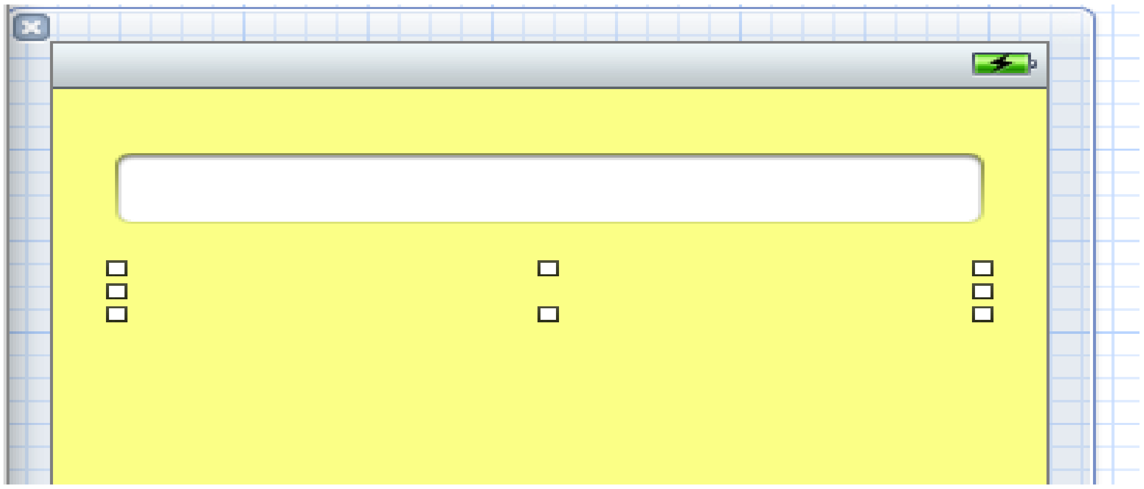
Sometimes we want to perform some task on an object that is not included in the methods of that class. An example of this might be reversing the order of characters in an NSString object. While we could certainly do this in code, it would be better if we could add a class method to NSString itself that would reverse any string. We will use a category to add the string reversal method to NSString.

What is a Category?

A category is an Objective-C construct that allows us to add methods to any class, regardless of whether we have the source code for that class. Once we add a category to a class, all the methods in the category are added to all instances of the class in our code. Since categories are maintained in separate files, it is a simple matter to include them in any project where the extended features of the class are desired.

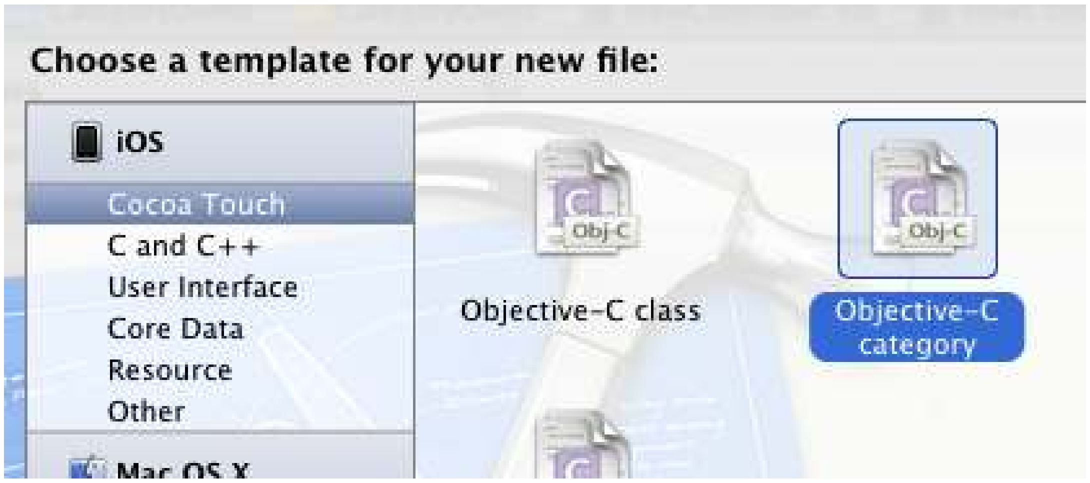
Step 1: Open a new **Single View Application** and call it **Category** or something that you like. Use Auto Reference Counting (ARC). Don't select the storyboard option. Don't really need it for this application.

Step 2: Add a Text Field and a Label to the ViewController.xib file.

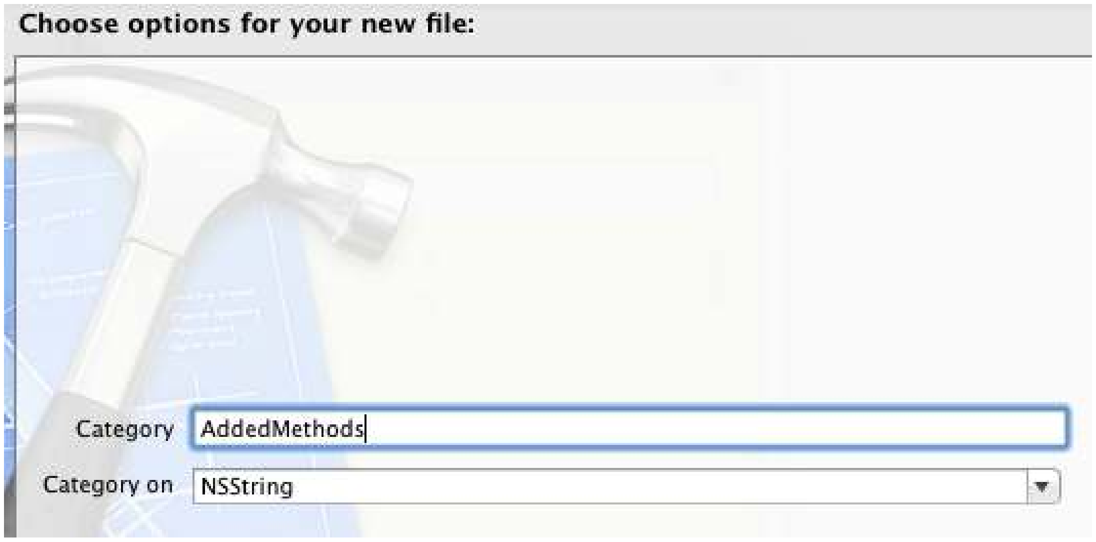


Step 3: In the properties for the Text Field, change the **Return Key** to **Done**.

Step 4: In Xcode, select File->New->File and then select the option to create an **Objective-C Category**.



Step 5: Name the new category **AddedMethods**. Make sure that the category is on **NSString.**



Category: AddedMethods
Category on: NSString

Step 6: Click Next, then Create. Two new files will be generated with the names:

NSString+AddedMethods.h
NSString+AddedMethods.m

This file naming convention makes it easy to see categories that we have added.

Step 7: Open **NSString+AddedMethods.h** and add the declaration for our new class method in between @interface and @end as shown below:

```
+ (NSString *)stringByReversingString:(NSString *)str;
```

Save the file.

Step 8: Open **NSString+AddedMethods.m** and add the following method implementation between @implementation and @end as show below:

```
+ (NSString *)stringByReversingString:(NSString *)str
{
    char cString[50];
    char revString[50];
    NSString *retval;
    [str getCString:cString
          maxLength:[str
lengthOfBytesUsingEncoding:NSUTF8StringEncoding] + 1
           encoding:NSUTF8StringEncoding];
    printf("%s", cString);

    for (int i = 0; i < strlen(cString); i++) {
        revString[i] = cString[strlen(cString) - (i + 1)];
    }
    revString[strlen(cString)] = '\0';
    retval = [NSString stringWithCString:revString
encoding:NSUTF8StringEncoding];

    return retval;
}
```

It is much easier to index into C strings and build them up character by character than it would be for NSString objects.

In this method, we set up two buffers (char arrays). Then we get a C string representation of the str argument by using the getCString:maxLength:encoding method. The maxLength parameter must be one more than the number of bytes in the NSString to account for the NULL terminator on C strings.

We then copy each character from the cString to the revString in reverse order (in the for loop). After doing this, we put a NULL terminator ('\0') at the end of the reversed string, create a new NSString (retval) from the contents of the reversed string, and return it.

Step 9: Import the category in order to use it. Add the following line to the top of **ViewController.h.**

```
#import "NSString+AddMethods.h"
```

Step 10: Wire the Label as a property. Call it **output**. The finished code looks like:

```
@property (weak, nonatomic) IBOutlet UILabel *output;
```

Step 11: Right-Click on the Text Field and wire the **Did End on Exit** event and call it **inputDone**. The finished code looks like:

```
- (IBAction)inputDone:(UITextField *)sender;
```

Step 12: Switch to ViewController.m and add the code to make the text field's "Did End on Exit" work. The finished method looks like:

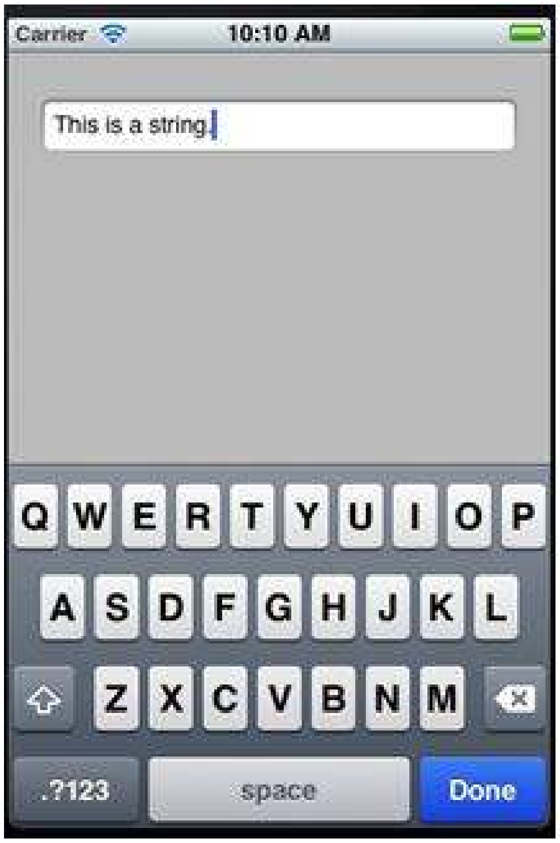
```
- (IBAction)inputDone:(UITextField *)sender {
    self.output.text = [NSString stringByReversingString:sender.text];
    [sender resignFirstResponder];
}
```

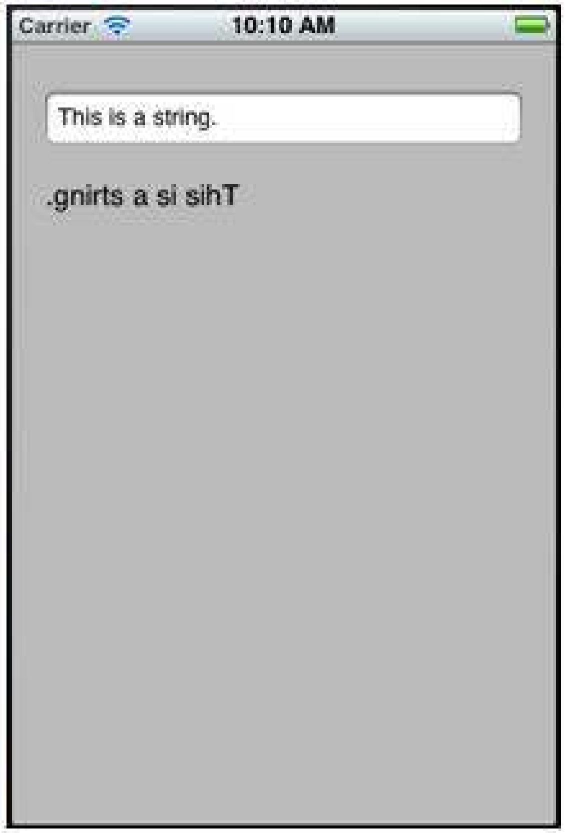
This action method sets the output label's text property to the reversed contents of the input text field's text property using the stringByReversingString method we just added to the NSString class.

After doing this, the method calls resignFirstResponder on the sender, which dismisses the keyboard.

Step 13: Run the program, you are done!

The program output looks like this:


Carrier
10:10 AM
This is a string.
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
.?123
space
Done


Carrier
10:10 AM
This is a string.
.gnirts a si sihT