**Movie Player:**

**Create a new iOS iPhone application project based on the Single View Application template, naming the product and class prefix Movie**

**Add the Media Framework to the Project:**

**The MPMoviePlayerController class is contained within the MediaPlayer framework. This framework must be included in any projects that make use of this class. This can be achieved by selecting the product target entry from the project navigator panel (the top item named Movie) and clicking on the Build Phases tab in the main panel. In the Link Binary with Libraries section click on the ‘+’ button, select the MediaPlayer.framework entry from the resulting panel and click on the Add button.**

**Design the Interface:**

**Drag a single Button instance to the view window and change the text on the button to “Play Movie”.**

**Display the Assistant Editor, Ctrl-click on the button object and drag the line to the area immediately beneath the @interface line in the Assistant Editor panel. Release the line and, within the resulting connection dialog, establish an Action method on the Touch Up Inside event configured to call a method named playMovie.**

**Before proceeding, it is imperative that the <MediaPlayer/MediaPlayer.h> file be imported to avoid a catalog of unresolved references when the application is compiled. A reference to an instance of the MPMoviePlayerController class is also going to required. Select the MovieViewController.h file, therefore, and modify it as follows:**

#import <UIKit/UIKit.h>

#import <MediaPlayer/MediaPlayer.h>

@interface MovieViewController : UIViewController

@property (strong, nonatomic) MPMoviePlayerController \*moviePlayer;

- (IBAction)playMovie:(id)sender;

@end

**Implementing the Action Method**

**The next step is to write the code for the action method so that video playback is initiated when the button is touched by the user. The stub method for the playMovie action has been created in MovieViewController.m and should be modified as follows**

-(void)playMovie:(id)sender

{

NSURL \*url = [NSURL URLWithString:

@"http://mm214.com/surf/70/video/hydromini.mov"];

\_moviePlayer = [[MPMoviePlayerController alloc]

initWithContentURL:url];

[[NSNotificationCenter defaultCenter] addObserver:self

selector:@selector(moviePlayBackDidFinish:)

name:MPMoviePlayerPlaybackDidFinishNotification

object:\_moviePlayer];

\_moviePlayer.controlStyle = MPMovieControlStyleDefault;

\_moviePlayer.shouldAutoplay = YES;

[self.view addSubview:\_moviePlayer.view];

[\_moviePlayer setFullscreen:YES animated:YES];

}

**The above method constructs an NSURL object using the URL of a web based video file. This is then used in the creation of a new instance of the MPMoviePlayerController class. A notification is then configured such that the moviePlaybackDidFinish method is called when the playback finishes. Next, properties are set to ensure that the standard movie controls are available to the user and that the movie automatically starts playing once it is ready. Finally the movie player object is added as a subview to the current view and displayed to the user in full screen mode.**

**The Target-Action Notification Method**

**The playButton action method declared that when the movie has finished playing, the movie player object is to call a method named moviePlaybackDidFinish. It is the responsibility of this method to cancel the notification and remove the movie player interface from display. Edit the MovieViewController.m file and add this method as follows:**

- (void) moviePlayBackDidFinish:(NSNotification\*)notification {

MPMoviePlayerController \*player = [notification object];

[[NSNotificationCenter defaultCenter]

removeObserver:self

name:MPMoviePlayerPlaybackDidFinishNotification

object:player];

if ([player

respondsToSelector:@selector(setFullscreen:animated:)])

{

[player.view removeFromSuperview];

}

}

**\*** **The MPMoviePlayerController class supports the playback of movies of type .mov, .mp4, .mpv and .3gp. In terms of compression, the class supports H.264 Baseline Profile Level 3.0 and MPEG-4 Part 2 video.**