Bit Logic Operators

AND	OR	XOR	NOT
10101010	10101010	10101010	10101010
11110000	11110000	11110000	
10100000	11111010	01011010	01010101

AND &	OR |	XOR ^	NOT !

shl (<<)	shr (>>)
1011100	1011100
111000	0101110

Sample Obj-C Code

```
//
//  main.m
//  test
//
//  Created by Arend on 9/12/11.
//  Copyright 2011 __MyCompanyName__. All rights reserved.
//

#import <Foundation/Foundation.h>

int main (int argc, const char * argv[])
{
```

```
    NSAutoreleasePool * pool = [[NSAutoreleasePool alloc] init];

    // insert code here...
    NSLog(@"Hello, World!");

    [pool drain];
    return 0;
}
```

NSString

```
S=[NSString alloc]; alloc is part of NSString ... in fact it is “part” of
NSObject
But: alloc is class method and does not create an instance
init does!

S=[[NSString alloc] init]
init is one of many constructors!

initWithString is another

S=[[NSString alloc] initWithString:@"Let there be light!"];
NSLog(@"Hello, World! %@",S);

NSLog can print NSStrings, %@ is the new identifier

S=[[NSString alloc] initWithFormat:@"Hallo %i",100];
```

NSMutableString

```
NSMutableString *M=[[NSMutableString alloc] initWithString:@"first part "];
[M appendString:@"second part"];

[M appendString:@"second part"];

is similar to c++:

M->appendString(@”second part”);

[object what] means:
```

```
broadcast what to object
c++ = object->what();

[object what:number] means:
broadcast what to object using number as a parameter
c++ = object->what(number)

[object what:numberOne another:numberTwo] means:
broadcast what to object using numberOne and numberTwo as a parameter
c++ = object->what(numberOne,numberTwo);
```

In other words:

```
class myClass{
private:
    int privateVariable;
    void aFunction(int aVariable,int bVariable);
};
void myClass::aFunction(int aVariable,int bVariable){
    privateVariable=aVariable*bVariable;
}

instance.aFunction(5,5);
instance->aFunction(5,5);

@interface myClass : NSObject {
@private
    int privateVariable;
}
-(void) aFunction:(int)a bVariable:(int)b;
@end
@implementation myClass
-(void) aFunction:(int)a bVariable:(int)b{
    privateVariable=a*b;
}
@end

[instance aFunction:5 bVariable:5];
```

Scheme

```
instance->aFunction(5,5);

[instance aFunction:5 bVariable:5];

-(void) aFunction:(int)a bVariable:(int)b;
```

NSArray

```
NSArray *A=[[NSArray alloc] initWithArray:[M componentsSeparatedByString:@"
"]];

NSLog(@"Hello, World! %@",A);
@ calls the string serialization method of NSArray

[A writeToFile:@"array.xml" atomically:YES];
writeToFile calls the file serialization method and produces a XML

for(NSString *S in B)
    NSLog(@"%@",S);
these classes have built in iterators which can be used in for loops
```

NSMutableArray

```
NSMutableArray *M=[[NSMutableArray alloc] init];
for(int i=0;i<10;i++)
    [M addObject:[[NSNumber alloc] initWithInt:i]];
for(NSNumber *N in M)
    NSLog(@"%@",[N stringValue]);

    NSLog(@"%i",[N intValue]);
    NSLog(@"%f",[N floatValue]);
```

XML Serialization

```
[M writeToFile:@"mutArray.xml" atomically:YES];
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/
DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<array>
      <integer>0</integer>
      <integer>1</integer>
      <integer>2</integer>
      <integer>3</integer>
      <integer>4</integer>
      <integer>5</integer>
      <integer>6</integer>
      <integer>7</integer>
      <integer>8</integer>
      <integer>9</integer>
</array>
</plist>
```

```
NSMutableArray *M=[[NSMutableArray alloc]
initWithContentsOfFile:@"mutArray.xml"];
```

NSMutable Dictionary

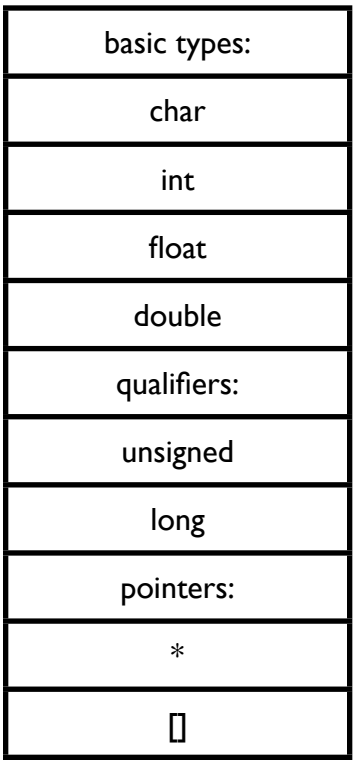
```
NSMutableDictionary *D=[[NSMutableDictionary alloc] init];
for(int i=0;i<10;i++)
    [D setObject:(id) forKey:(id)];
[D
    setObject:[[NSString alloc] initWithFormat:@"The string for element:%i",i]
    forKey:[[NSNumber alloc] initWithInt:i]
];
```

```
remember:
[object what:numberOne another:numberTwo] means:
broadcast what to object using numberOne and numberTwo as a parameter
c++ = object->what(numberOne,numberTwo);
```

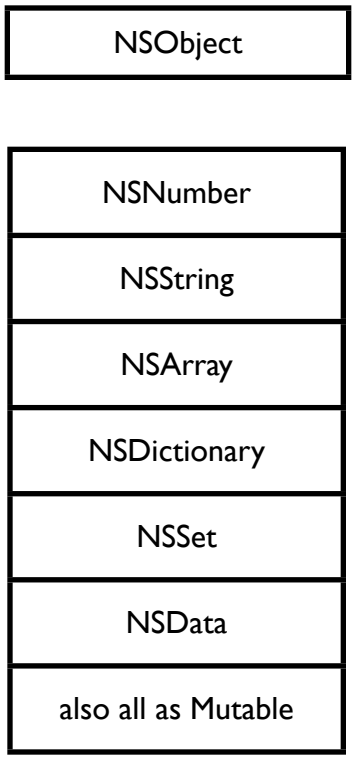
Iterators of NSDictionary

```
for(NSNumber *N in D)
    NSLog(@"%@",[N stringValue]);
for(NSString *S in D)
    NSLog(@"%@",S);

for(NSNumber *N in D)
    NSLog(@"%@",[D objectForKey:N]);
for(NSString *S in D)
    NSLog(@"%@",[D objectForKey:S]);

NSLog(@"%@",[D allKeys]);
NSLog(@"%@",[D allValues]);
```

Types



Objects



Creating Objective C Objects

```
@interface myClass : NSObject {
@private
    int privateVariable;
```

```
}

-(void) aFunction:(int)a bVariable:(int)b;

@end

@implementation myClass

-(void) aFunction:(int)a bVariable:(int)b{
    privateVariable=a*b;
}

@end
```

Calls

Basic Calls:

```
[object method];
[object methodWithInput:input];
[object methodWithInputA:inputA inputB:inputB];
```

Methods can return a value:

```
output = [object method];
output = [object methodWithInput:input];
```

Nesting is Desired:

```
output = [objectA methodWithInput:[objectB methodWithInput:input]];
```

Instances

Objects have special “pointers” called id:

```
id myObject = [NSString string];
```

Problem:

```
id myObject = [NSString string];
myObject = [NSArray array];
```

Solution:

```
NSString* myString = [NSString string];
```

Advantage:

```
(di) autorelease [NSString string];
```

Release and Autorelease

```
NSString* myObject = [[NSString alloc] init];
     ...
[myObject release];
```

```
{
    NSString* myObject = [[NSString alloc] init];
    NSString* myString=[NSString string];
          ...
    [myObject release];
    //myString will be autoreleased
}
```

```
{
     NSString* myObject = [[[NSString alloc] init] autorelease];
   //myObject will be autoreleased
}
```

Never Dealloc Except:

```
-(void) dealloc{
    [super dealloc];
}
```

Set and Get Oldschool:

//.h

```
@interface myClass : NSObject {
@private
    int variable;
}
```

```
-(int) getVariable;
-(void) setVariable:(int)newValue;
@end


//.m
-(int) getVariable{
    return variable;
}
-(void) setVariable:(int)newValue{
    variable=newValue;
}

//main
myClass *test=[[myClass alloc] init];
[test setVariable:15];
NSLog(@"%i",[test getVariable]);
```

Set and Get Newschool

//.h

```
@interface myClass : NSObject {
@private
    int variable;
}
@property (readwrite) int variable;
@end

//.m
@synthesize variable;

//main
myClass *test=[[myClass alloc] init];
test.variable=15;
NSLog(@"%i",test.variable);

@property (readonly) int variable;
```

Set and Get Mashup

//.h

```
@interface myClass : NSObject {
@private
    int variable;
```

```
}
@property (readwrite) int variable;
@end

//.m
@synthesize variable;

//main
[test setVariable:15];
NSLog(@"%i",[test variable]);
```

New School Example:

```
@interface myClass : NSObject {
@private
    NSMutableString *S1;
    NSMutableString *S2;
}
@property (readwrite) NSMutableString* S1;
@property (readwrite) NSMutableString* S2;
-(unsigned long) totalLength;
@end

@implementation myClass
@synthesize S1;
@synthesize S2;
-(unsigned long) totalLength{
    return [S1 length]+[S2 length];
}
@end

[test setS1:@"Hello"];
[test setS2:@" this is a test!"];
NSLog(@"%i",(int)[test totalLength]);
```

Old School Example

```
@interface myClass : NSObject {
@private
    NSMutableString *S1;
    NSMutableString *S2;
}
-(void) setS1:(NSMutableString*)newString;
```

```
-(void) setS2:(NSMutableString*)newString;
-(unsigned long) totalLength;
@end

@implementation myClass
-(void) setS1:(NSMutableString*)newString{ S1=newString;}
-(void) setS2:(NSMutableString*)newString{ S2=newString;}
-(unsigned long) totalLength{
    return [S1 length]+[S2 length];
}
@end

[test setS1:@"Hello"];
[test setS2:@" this is a test!"];
NSLog(@"%i",(int)[test totalLength]);
```

SPQuad

width, height
x,y
alpha
scaleX,scaleY
color
color: ofVertex:
rotation
[self addChild:] ->display tree

```
SPStage has children
-order matters, first added first painted

SPQuads don't have children...
[quad addChild:[SPQuad quadWithWidth:50 height:50 color:0x0000ff]];
```

SPImage (SPTexture)

texture
width, height
x, y
alpha
scaleX, scaleY
color
color: ofVertex:
rotation
[self addChild:]->display tree

```
SPImage *I =[SPImage imageWithTexture:
                    [SPTexture textureWithContentsOfFile:@"megaMan.png"]
                ];

[I setName:[NSString stringWithFormat:@"miniman %i",i]];

            [self childAtIndex:(int)];
            [self childByName:(NSString *)];
```

Events - Tell you that something has happened

onEnterFrame
SP_EVENT_TYPE_ENTER_FRAME
SPEnterFrameEvent

```
[self addEventListener:@selector(onEnterFrame:) atObject:self
forType:SP_EVENT_TYPE_ENTER_FRAME];

requires:

-(void) onEnterFrame:(SPEnterFrameEvent*)event
```

Setting up the field:

```
          for(int x=0;x<8;x++)
            for(int y=0;y<8;y++){
                SPQuad *Q=[SPQuad quadWithWidth:40 height:40];
                Q.x=x*40;
                Q.y=y*40;
                [Q setName:[NSString stringWithFormat:@"quad_%i_%i",x,y]];
                for(int z=0;z<4;z++)
                    [Q setColor:SP_COLOR(rand()&255,rand()&255,rand()&255)
                       ofVertex:z];
                [self addChild:Q];
            }
          [self addEventListener:@selector(onEnterFrame:)
                      atObject:self
                      forType:SP_EVENT_TYPE_ENTER_FRAME];
```

Let's do something:

```
-(void) onEnterFrame:(SPEnterFrameEvent*) event{
```

```
    int x1,y1,x2,y2;
    float spx,spy;
    do{
        x1=rand()&7;
        y1=rand()&7;
        x2=rand()&7;
        y2=rand()&7;
    }while((x1==x2)||(y1==y2));
    SPDisplayObject *Q1,*Q2;
    Q1=[self childByName:[NSString stringWithFormat:@"quad_%i_%i",x1,y1]];
    Q2=[self childByName:[NSString stringWithFormat:@"quad_%i_%i",x2,y2]];
    spx=Q1.x;
    spy=Q1.y;
    Q1.x=Q2.x;
    Q1.y=Q2.y;
    Q2.x=spx;
    Q2.y=spy;
}
```

onTouch
SP_EVENT_TYPE_TOUCH
SPTouchEvent

Adding a touch:

```
          for(int x=0;x<8;x++)
            for(int y=0;y<8;y++){
                SPQuad *Q=[SPQuad quadWithWidth:40 height:40];
                Q.x=x*40;
                Q.y=y*40;
                [Q setName:[NSString stringWithFormat:@"quad_%i_%i",x,y]];
                for(int z=0;z<4;z++)
                    [Q setColor:SP_COLOR(rand()&255,rand()&255,rand()&255)
                       ofVertex:z];
                [self addChild:Q];
                [Q addEventListener:@selector(onTouch:)
                           atObject:self
                            forType:SP_EVENT_TYPE_TOUCH];
            }
          [self addEventListener:@selector(onEnterFrame:)
                        atObject:self
                         forType:SP_EVENT_TYPE_ENTER_FRAME];
```

Delete the source of the touch:

```
-(void) onTouch:(SPTouchEvent*)event{
    [self removeChild:event.target];
```

```
}
```

You can make your own events:

```
#define EVENT_TYPE_YES_TRIGGERED @"yesTriggered"
#define EVENT_TYPE_NO_TRIGGERED  @"noTriggered"

// the listener we created for the yes-button:
- (void)onYesButtonTriggered:(SPEvent *)event
{
    SPEvent *localEvent = [SPEvent eventWithType:EVENT_TYPE_YES_TRIGGERED
bubble:YES];
    [self dispatchEvent:localEvent];
}

// the listener we created for the no-button:
- (void)onNoButtonTriggered:(SPEvent *)event
{
    SPEvent *localEvent = [SPEvent eventWithType:EVENT_TYPE_NO_TRIGGERED
bubble:YES];
    [self dispatchEvent:localEvent];
}
```

Texture Atlas Example:

```
<TextureAtlas imagePath="textureAtlas.png">
      <SubTexture name="standRight" x="0" y="0" width="64" height="64" />
      <SubTexture name="standLeft" x="64" y="0" width="64" height="64" />
      <SubTexture name="sprintRight1" x="0" y="128" width="64" height="64" />
      <SubTexture name="sprintRight2" x="64" y="128" width="64" height="64" />
      <SubTexture name="sprintRight3" x="128" y="128" width="64" height="64" />
      <SubTexture name="sprintRight4" x="192" y="128" width="64" height="64" />
      <SubTexture name="sprintLeft1" x="256" y="128" width="64" height="64" />
      <SubTexture name="sprintLeft2" x="320" y="128" width="64" height="64" />
      <SubTexture name="sprintLeft3" x="384" y="128" width="64" height="64" />
      <SubTexture name="sprintLeft4" x="448" y="128" width="64" height="64" />
</TextureAtlas>
```

Implement Texture Atlas:

```
SPTextureAtlas *atlas=[SPTextureAtlas
atlasWithContentsOfFile:@"textureAtlas.xml"];
SPMovieClip *M=[ [SPMovieClip alloc]
                  initWithFrame:[atlas textureByName:@"standRight"]
                  fps:10.0
```

```
                    ];
[M addFrame:[atlas textureByName:@"sprintRight1"]];
[M addFrame:[atlas textureByName:@"sprintRight2"]];
[M addFrame:[atlas textureByName:@"sprintRight3"]];
[M addFrame:[atlas textureByName:@"sprintRight4"]];
[M addFrame:[atlas textureByName:@"standLeft"] withDuration:5.0];
[M addFrame:[atlas textureByName:@"sprintLeft1"]];
[M addFrame:[atlas textureByName:@"sprintLeft2"]];
[M addFrame:[atlas textureByName:@"sprintLeft3"]];
[M addFrame:[atlas textureByName:@"sprintLeft4"]];
[M addFrame:[atlas textureByName:@"standRight"] withDuration:2.0];
[M play];
[M setLoop:YES];
[self addChild:M];
[self.stage.juggler addObject:M];
```

SPMovieClip

```
    NSArray *A=[atlas texturesStartingWith:@"p"];
    SPMovieClip *M=[[SPMovieClip alloc] initWithFrames:A fps:8.0];
```

```
<TextureAtlas imagePath="textureAtlas.png">
      <SubTexture name="standRight" x="0" y="0" width="64" height="64" />
</TextureAtlas>
```

```
SPTextureAtlas *atlas=[SPTextureAtlas
atlasWithContentsOfFile:@"textureAtlas.xml"];
SPMovieClip *M=[ [SPMovieClip alloc]
                initWithFrame:[atlas textureByName:@"standRight"]
                fps:10.0
];
[M addFrame:[atlas textureByName:@"sprintRight1"]];
[M addFrame:[atlas textureByName:@"standLeft"] withDuration:5.0];
[M play];
[M setLoop:YES];
[self addChild:M];
[self.stage.juggler addObject:M];
```

SPTween

```
        SPTween *T =[SPTween tweenWithTarget:M time:4.0];
        [T animateProperty:@"x" targetValue:rand()%(320-(int)M.width)];
        [T animateProperty:@"y" targetValue:rand()%(480-(int)M.height)];
        [self.stage.juggler addObject:T];


SPTween *T =[SPTween tweenWithTarget:M time:4.0
transition:SP_TRANSITION_EASE_OUT_IN_BOUNCE];
```

Touches

begin
moved
stationary
ended
cancelled
multitouch

```
-(void) onTouch:(SPTouchEvent*)event{
    NSArray *touches = [
                        [event
                        touchesWithTarget:self andPhase:SPTouchPhaseBegan]
                        allObjects];
    for(SPTouch *T in touches){
        SPQuad *Q=[SPQuad quadWithWidth:10 height:10];
        Q.x=[T locationInSpace:self].x;
        Q.y=[T locationInSpace:self].y;
        [self addChild:Q];
    }
}

        SPTween *T =[SPTween tweenWithTarget:M time:4.0];
        [T animateProperty:@"x" targetValue:rand()%(320-(int)M.width)];
        [T animateProperty:@"y" targetValue:rand()%(480-(int)M.height)];
        [self.stage.juggler addObject:T];
```

SPCompiledSprite

```
-(SPCompiledSprite*) makeMap{
    int x,y;
    SPCompiledSprite *dummy;
    dummy=[[SPCompiledSprite alloc] init];
      Loop stuff:
                [dummy addChild:q];

    [dummy compile];
    return dummy;
}
```

Interface Orientation

```
- (BOOL)shouldAutorotateToInterfaceOrientation: (UIInterfaceOrientation)
orientation
{
    return YES;
}

[[UIDevice currentDevice] beginGeneratingDeviceOrientationNotifications];
[[NSNotificationCenter defaultCenter]
                                    addObserver:self
                          selector:@selector(onOrientationDidChange:)
                              name:@"UIDeviceOrientationDidChangeNotification"
                                    object:nil];

-(void)onOrientationDidChange:(UIEvent *)event{
        NSLog(@"%i",[UIDevice currentDevice].orientation);
}
```

UIAcclerometer

```
UIAccelerometer *accelerometer = [UIAccelerometer sharedAccelerometer];
accelerometer.updateInterval = 1.0/60.0;
accelerometer.delegate = self;
```

```
@interface Game : SPStage <UIAccelerometerDelegate> {

- (void) accelerometer:(UIAccelerometer *)accelerometer didAccelerate:
(UIAcceleration *)acceleration{
    float accelX = acceleration.x;
    float accelY = acceleration.y;
    float accelZ = acceleration.z;
    NSLog(@"%f %f %f",accelX,accelY,accelZ);
}

@protocol myProtocol
@optional
-(int) getFeeling;
@required
-(bool) alive;
@end

@interface Game : SPStage

<UIAccelerometerDelegate,myProtocol> {
```

SPButton

```
SPButton *B =[SPButton buttonWithUpState:
                        [SPTexture textureWithContentsOfFile:@"joybutton.png"]
                    ];

[B addEventListener:@selector(buttonTriggered:)
     atObject:self
     forType:SP_EVENT_TYPE_TRIGGERED];
```

Properties:

```
1  - initWithUpState:downState:
2  - initWithUpState:text:
3  - initWithUpState:
4  + buttonWithUpState:downState:
5  + buttonWithUpState:text:
6  + buttonWithUpState:
```

```
    scaleWhenDown property
    alphaWhenDisabled property
    enabled property
    text property
    fontName property
    fontSize property
    fontColor property
    upState property
    downState property
    textBounds property
```

SPSound and SPSoundChannel

SPSound:

```
    [[SPSound soundWithContentsOfFile:@"sound.caf"] play];
```

SPSoundChannel:

```
    SPSound *S=[SPSound soundWithContentsOfFile:@"sound.caf"];
    SPSoundChannel *C=[S createChannel];
    [C play];
```

Properties:

```
- play
- stop
- pause
```

```
    isPlaying property
    isPaused property
    isStopped property
    duration property
    volume property
    loop property
```

SPTextField

```
        SPTextField *textField = [SPTextField
                                  textFieldWithWidth:145
                                  height:80
                                  text:@"Text"
                                  fontName:@"Helvetica"
                                  fontSize:64.0f
                                  color:0xff0000];
         textField.hAlign = SPHAlignCenter;  // horizontal alignment
         textField.vAlign = SPVAlignCenter; // vertical alignment
         textField.border = NO;
         [self addChild:textField];

        [SPTextField registerBitmapFontFromFile:@"myDigitalFont.fnt"];
```

Creating a UIView

```
[[UIView alloc] initWithFrame:[UIScreen mainScreen].bounds ];
```

Within your AppDelegate:

```
self.window.rootViewController = [[UIViewController alloc] init];
self.viewController.view=[[UIView alloc] initWithFrame:
                          [UIScreen mainScreen].bounds];
```

Essential Functions:

```
- (id)initWithFrame:(CGRect)frame{
    self = [super initWithFrame:frame];
    if (self) { /* Initialization code */ }
    return self;
}

[self setNeedsDisplay];

- (void)drawRect:(CGRect)rect{
}

-(void) touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event{
}

-(void) touchesEnded:(NSSet*)touches withEvent:(UIEvent*)event{
}
```

```
-(void) touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event{
}
```

Managing Views

```
-(void) touchesEnded:(NSSet*)touches withEvent:(UIEvent*)event{
    UITouch *T=[touches anyObject];
    if(T!=NULL){
        if([T locationInView:self].y<240)
        {
            mainViewClass *newView;
            CGRect bounds=[super bounds];
            newView=[[mainViewClass alloc]
                                initWithFrame:CGRectMake(
                                    bounds.origin.x+10 ,
                                    bounds.origin.y+10,
                                    bounds.size.width-20,
                                    bounds.size.height-20)
                                    ];
            [self addSubview:newView];
        }
        else{
            [self removeFromSuperview];
        }
    }
}
```

Number of Touches

```
UISwipeGestureRecognizer *SR=[[UISwipeGestureRecognizer alloc
     initWithTarget:self action:@selector(handleGesture:)];
[SR setDirection:UISwipeGestureRecognizerDirectionUp];
[self addGestureRecognizer:SR];
```

Simple Animations

```
animatable properties
 @property frame
 @property bounds
 @property center
 @property transform
 @property alpha
 @property backgroundColor
 @property contentStretch

[UIView
     animateWithDuration:0.1
     animations:^{self.alpha=0.0;}
     ];

[UIView animateWithDuration:1.0
     delay:3.0
     options:UIViewAnimationOptionCurveEaseInOut
     animations:^{self.alpha=0.0;}
     completion:NULL];

enum {
   UIViewAnimationOptionLayoutSubviews            = 1 <<  0,
   UIViewAnimationOptionAllowUserInteraction      = 1 <<  1,
   UIViewAnimationOptionBeginFromCurrentState     = 1 <<  2,
   UIViewAnimationOptionRepeat                    = 1 <<  3,
   UIViewAnimationOptionAutoreverse               = 1 <<  4,
   UIViewAnimationOptionOverrideInheritedDuration = 1 <<  5,
   UIViewAnimationOptionOverrideInheritedCurve    = 1 <<  6,
   UIViewAnimationOptionAllowAnimatedContent      = 1 <<  7,
   UIViewAnimationOptionShowHideTransitionViews   = 1 <<  8,

   UIViewAnimationOptionCurveEaseInOut            = 0 << 16,
   UIViewAnimationOptionCurveEaseIn               = 1 << 16,
   UIViewAnimationOptionCurveEaseOut              = 2 << 16,
   UIViewAnimationOptionCurveLinear               = 3 << 16,

   UIViewAnimationOptionTransitionNone            = 0 << 20,
   UIViewAnimationOptionTransitionFlipFromLeft    = 1 << 20,
   UIViewAnimationOptionTransitionFlipFromRight   = 2 << 20,
   UIViewAnimationOptionTransitionCurlUp          = 3 << 20,
   UIViewAnimationOptionTransitionCurlDown        = 4 << 20,
   UIViewAnimationOptionTransitionCrossDissolve   = 5 << 20,
   UIViewAnimationOptionTransitionFlipFromTop     = 6 << 20,
   UIViewAnimationOptionTransitionFlipFromBottom  = 7 << 20,
};
```

```
typedef NSUInteger UIViewAnimationOptions;
```

Transitions

```
CGContextRef myContext= UIGraphicsGetCurrentContext();
[UIView beginAnimations:nil context:myContext];
[UIView
          setAnimationTransition:UIViewAnimationTransitionCurlDown
          forView:self
          cache:YES];
[UIView setAnimationCurve:UIViewAnimationCurveEaseInOut];
[UIView setAnimationDuration:1.0];
[UIView commitAnimations];

typedef enum {
   UIViewAnimationTransitionNone,
   UIViewAnimationTransitionFlipFromLeft,
   UIViewAnimationTransitionFlipFromRight,
   UIViewAnimationTransitionCurlUp,
   UIViewAnimationTransitionCurlDown,
} UIViewAnimationTransition;

typedef enum {
   UIViewAnimationCurveEaseInOut,
   UIViewAnimationCurveEaseIn,
   UIViewAnimationCurveEaseOut,
   UIViewAnimationCurveLinear
} UIViewAnimationCurve;
```

UILabel - UIView with a text property

```
Accessing the Text Attributes
  text  property
  font  property
  textColor  property
  textAlignment  property
  lineBreakMode  property
  enabled  property
Sizing the Label's Text
  adjustsFontSizeToFitWidth  property
  baselineAdjustment  property
```

```
  minimumFontSize  property
  numberOfLines  property
Managing Highlight Values
  highlightedTextColor  property
  highlighted  property
Drawing a Shadow
  shadowColor  property
  shadowOffset  property
Drawing and Positioning Overrides
- textRectForBounds:limitedToNumberOfLines:
- drawTextInRect:
Setting and Getting Attributes
  userInteractionEnabled  property
```

UIButton

```
UIButton *B=[UIButton buttonWithType:UIButtonTypeRoundedRect];
B.frame=CGRectMake(0, 400, 320, 80);
[B setTitle:@"Button" forState:UIControlStateNormal];
[B addTarget:self action:@selector(buttonClick:)
          forControlEvents:UIControlEventTouchUpInside];
[self addSubview:B];

typedef enum {
   UIButtonTypeCustom = 0,
   UIButtonTypeRoundedRect,
   UIButtonTypeDetailDisclosure,
   UIButtonTypeInfoLight,
   UIButtonTypeInfoDark,
   UIButtonTypeContactAdd,
} UIButtonType;

[B
     setBackgroundImage:
          [UIImage imageNamed:@"buttonBackground.png"
     forState:
     UIControlStateNormal
];

[B
setImage:
[UIImage imageNamed:@"buttonBackground.png"
forState:
```

```
UIControlStateNormal
];
```

UITextView

```
UITextView *T=[[UITextView alloc]
          initWithFrame:CGRectMake(10, 10, 300, 300)];
[T setText:@"The Text You want to display"];
[T setEditable:NO];
[self addSubview:T];
```

UITextField

```
UITextField *textFieldA=[[UITextField alloc] initWithFrame:CGRectMake(10, 10,
300, 30)];
[textFieldA setBorderStyle:UITextBorderStyleRoundedRect];
[textFieldA setPlaceholder:@"text"];
textFieldA.delegate=self;    <UITextFieldDelegate>!!!
[self addSubview:textFieldA];

-(BOOL)textFieldShouldReturn:(UITextField *)textField{
    NSLog(@"%@",textField.text);
      [textField resignFirstResponder];
      return YES;
}
```

UIWebView

```
UIWebView *W=[[UIWebView alloc] initWithFrame:
          CGRectMake(10, 10, 300, 300)];

NSString *S=[NSString stringWithContentsOfURL:
          [NSURL URLWithString:@"http://www.google.com"]
          encoding:NSASCIIStringEncoding error:NULL];

[W loadHTMLString:S baseURL:
```

```
        [NSURL URLWithString:@"http://www.google.com"]];

[self addSubview:W];
```