Chapter 4

Data Type, Operator &
Control Statement

Topics

- Basic Data Type
- Operator
- Control Statement
- Try, catch, Finally
- NSArray, NSMutableArray
- NSDictionary

Basic Data Type

Type	Meaning
int	integer value : that is , a value that contains no decimal point; guaranteed to contain at least 32 bits of accuracy
float	Floating-point value; that is , a value that can contain decimal places; guaranteed to contain as least six digits of precision
double	Extended accuracy floating-point value; guaranteed to contain at least 10 digits of precision
char	Single character value

Type **id**

The **id** data type is used to store an object of any type. In a sense , it is a generic object type (or weak type).

- Example:
  - id number;
  - -(id) newObject : (int) type

Some of Operator(1)

Operator	Description
*	Multiplication
/	Division
%	Modulo (5 % 3 = 2)
<=	Less than or equal
>=	Greater than or equal
&&	Logical AND
||	Logical OR
+=	Assignment + (a = a + b, a +=b)
-=	Assignment - (a = a - b, a -=b)
/=	Assignment / (a = a/b, a /=b)
!=	Inequality
==	Logical comparison

The if statement

Syntax:

```
if (expression)
        program statement;
```

Example :

```
if (n < 0) {
        m += 2;
        NSLog(@" n is negative number");
}
```

The if-else statement

Syntax:

```
if (expression)
        program statement;
else
        program statement;
```

Example :

```
if (n < 0)
        m += 1;
else
        p += 1;
```

The switch statement

Syntax :

```
switch (expression) {
        case value1:
                program statement;
                break;
        case value2:
                program statement;
                break;
        default;
                program statement;
                break;
}
```

The for statement

Syntax :

```
for (init-expression; loop_condition ; loop_expression)
    program statement;
```

Example :

```
for (int i = 1; i <=10 ; i++)
    x +=1;
```

The for-each statement

Syntax :

```
for (<each-type> <instance> in <collection> )
        program statement;
```

Example :

```
for (NSString * s in myArray)
        NSLog(@"%@", s);
```

The while statement

Syntax:

```
while (expression)
        program statement;
```

Example :

```
while  ( x < 5 ) {
        NSLog(@"I am in loop");
        x +=1;
}
```

The do statement

Syntax:

```
do
        program statement;
while (expression);
```

Example :

```
do {
        NSLog(@”I am in loop”);
        x += 1;
} while (x < 5);
```

Try-catch-finally

Syntax:

```
@try {
        program statement;
}
@catch (NSException *ex) {
        program statement;
}
@finally {
        program statement;
}
```

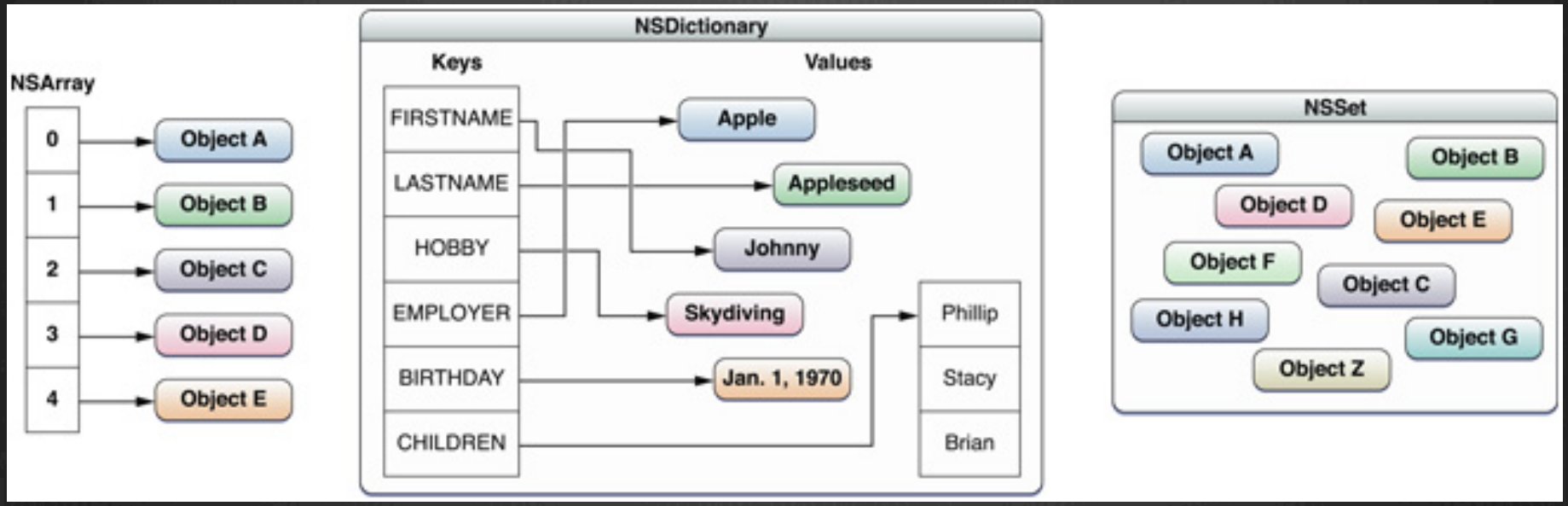
Basic Array

```
int x[10], i;

for (i = 0 ; i< 10 ; i++)
   x[i] = i;

for (i = 9 ; i >=0 ; i--)
   NSLog(@" %d", x[i]);
```

Character Arrays

```
char word[] = {'H','e','l','l','o'};
int i;

for (i=0; i < 6 ; ++i)
   NSLog(@"%c", word[i]);
}
```

Collections



NSArray

- An NSArray object manages an immutable array. After you have create the array, you cannot add, remove, or replace objects.
- You can, however, modify individual elements themselves (if they support modification)
- Example

```
NSArray *array;
array = [NSArray arrayWithObjects:@"One", @"Two", @"Three", nil];

for (int i = 0 ; i < [array count] ; i++)
    NSLog(@"index %d has %@", i, [array objectAtIndex:i]);
```

NSMutableArray

- An NSMutableArray object manages a mutable array, which allows the addition and deletion of entries at any time, automatically allocating memory as needed.

```
NSMutableArray *items = [[NSMutableArray alloc] init];
[items addObject:@"One"];
[items addObject:@"Two"];
[items addObject:@"Three"];
[items insertObject:@"Zero" atIndex:0];

for (int i = 0; i < [items count]; i++)
    NSLog(@"%@", [items objectAtIndex:i]);
```

Copy NSArray to NSMutableArray

```
NSArray * myArray = [NSArray arrayWithObjects:@"One",
                                              @"Two",
                                              @"Three",
                                              nil];
NSMutableArray *myMutableArray = [myArray mutableCopy];

for (NSString *s in myMutableArray)
     NSLog(@"%@", s);
```

NSArray Modern Syntax

- Xcode 5 (iOS 7 / OS X 10.9 SDK) only
- Convert exist code from menu “Edit > Refactor > Convert to Modern Objective-C Syntax”
- Traditional Syntax

```
NSArray * array = [NSArray arrayWithObjects:
                                @"one", @"two", nil];
[array replaceObjectIndex:0 withObject:””];
```

- Modern Syntax

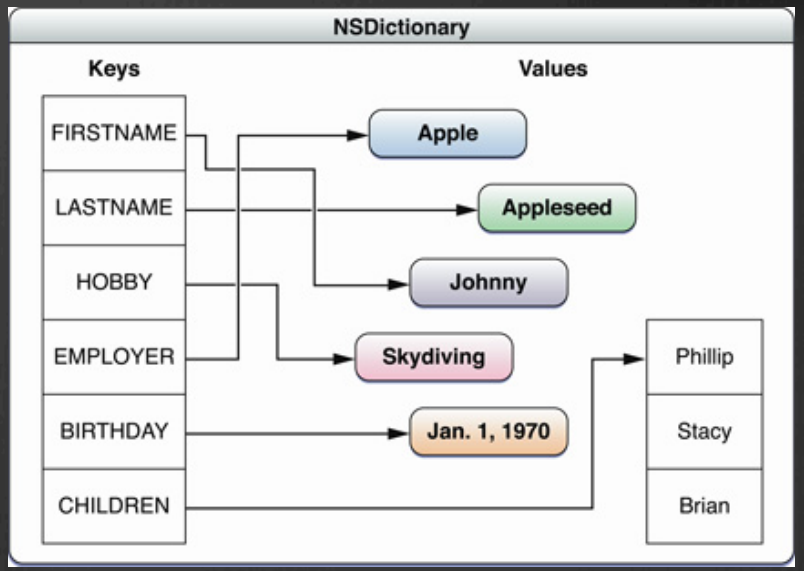
```
NSArrar * array = @[@“one”, @“two”];
array[0] = @””;
```

Most used NSMutableArray methods

- addObject:
- insertObject: atIndex:
- removeLastObject
- removeObjectAtIndex:
- replaceObjectAtIndex: withObject:

NSDictionary

- Dictionaries manage pairs of keys and values.
- A key-value pair within a dictionary is called an entry. Each entry consists of one object that represents the key, and a second object which is that key's value.
- Within a dictionary, the keys are unique.
- A key can be any object that adopts the `NSCopying` protocol and implements the hash and `isEqual:` methods.



NSDictionary

```
NSDictionary *myDictionary = [NSDictionary dictionaryWithObjectsAndKeys:
                              @"Harry Potter", @"2",
                              @"Ron Wisley", @"1",
                              @"Tom Riddle", @"3",
                              @"Voldermort", @"0",
                              nil ];

for (int i = 0; i < 4; i++)
    NSLog(@"%@", [myDictionary objectForKey:
                      [NSString stringWithFormat:@"%i", i]]);
```

NSMutableDictionary

- NSMutableDictionary object manages a mutable dictionary, which allows the addition and deletion of entries at any time, automatically allocating memory as needed.
- You should use a mutable dictionary if the dictionary changes incrementally, or is very large—as large collections take more time to initialize.
- Most used methods
  - dictionaryWithObjectsAndKeys:
  - removeObjectForKey:
  - setObject: forKey:
  - removeObjectForKey:

NSDictionary Modern Syntax

- Traditional Syntax

```
NSDictionary * dictionary =
    [NSDictionary dictionaryWithObjectsAndKey:
            @“one”, @”1”, @“two”, @”2”, nil];
```

- Modern Syntax

```
NSDictionary * dictionary =
    @{ @”1”: @”one”,
       @”2”: @”two”
    };
```

NSDictionary Modern Syntax

- Traditional Syntax

```
NSString * tmp = [dictionary objectForKey:1];
[dictionary setObject:@"" forKey:@"1"];
```

- Modern Syntax

```
NSString * tmp = dictionary[@1];
dictionary[@"1"] = @"";
```