DATA STORAGE IN OBJECTIVE-C

Data storage and its retrieval is one the most important in any program. In Objective-C, we generally won't depend on structures like linked list since it makes the work complex. Instead, we use the collections like NSArray, NSSet, NSDictionary and its mutable forms.

NSArray & NSMutableArray

NSArray is used to hold an immutable array of objects and the NSMutableArray is used to hold an mutable array of objects.

Mutablility helps to change the array in runtime a preallocated array, but if we use NSArray, we only replace the existing array and cannot change the contents of the existing array.

Important methods of NSArray are as follows

- alloc/initWithObjects: Used to initialize an array with objects.
- objectAtIndex: Returns the object at specific index.
- count: Returns the number of objects

NSMutableArray is inherited from NSArray and hence all instance methods of NSArray is available in NSMutableArray

Important methods of NSMutableArray are as follows:

- removeAllObjects: Empties the array.
- addObject: Inserts a given object at the end of the array.
- removeObjectAtIndex: This is used to remove objectAt a specific index
- exchangeObjectAtIndex:withObjectAtIndex: Exchanges the objects in the array at given indices.
- replaceObjectAtIndex:withObject: Replaces the object at index with an Object.

We have to remember that the above list is just the frequently used methods and we can jump into respective classes in our XCode to know more methods in these classes. A simple example is shown below.

```
#import <Foundation/Foundation.h>

int main()
{
   NSAutoreleasePool * pool = [[NSAutoreleasePool alloc] init];
   NSArray *array = [[NSArray alloc]
   initWithObjects:@"string1", @"string2",@"string3",nil];
   NSString *string1 = [array objectAtIndex:0];
   NSLog(@"The object in array at Index 0 is %@",string1);
   NSMutableArray *mutableArray = [[NSMutableArray alloc]init];
   [mutableArray addObject: @"string"];
   string1 = [mutableArray objectAtIndex:0];
   NSLog(@"The object in mutableArray at Index 0 is %@",string1);
   [pool drain];
   return 0;
}
```

Now when we compile and run the program, we will get the following result.

```
2013-09-29 02:33:23.195 demo[3487] The object in array at Index 0 is string1
2013-09-29 02:33:23.196 demo[3487] The object in mutableArray at Index 0 is string
```

In the above program, we have seen a simple differentiation between NSMutableArray and NSArray where we can insert a string after allocation in mutable array.

NSDictionary & NSMutableDictionary

NSDictionary is used to hold an immutable dictionary of objects and the NSMutableDictionary is used to hold an mutable dictionary of objects.

Important methods of NSDictionary are as follows

- alloc/initWithObjectsAndKeys: Initializes a newly allocated dictionary with entries constructed from the specified set of values and keys.
- valueForKey: Returns the value associated with a given key.
- count: Returns the number of entries in the dictionary.

NSMutableDictionary is inherited from NSDictionary and hence all instance methods of NSDictionary are available in NSMutableDictionary

Important methods of NSMutableDictionary are as follows:

- removeAllObjects: Empties the dictionary of its entries.
- removeObjectForKey: Removes a given key and its associated value from the dictionary.
- setValue:forKey: Adds a given key-value pair to the dictionary.

A simple example for dictionary is shown below:

```
#import <Foundation/Foundation.h>

int main()
{
   NSAutoreleasePool * pool = [[NSAutoreleasePool alloc] init];
   NSDictionary *dictionary = [[NSDictionary alloc] initWithObjectsAndKeys:
   @"string1",@"key1", @"string2",@"key2",@"string3",@"key3",nil];
   NSString *string1 = [dictionary objectForKey:@"key1"];
   NSLog(@"The object for key, key1 in dictionary is %@",string1);
   NSMutableDictionary *mutableDictionary = [[NSMutableDictionary alloc]init];
   [mutableDictionary setValue:@"string" forKey:@"key1"];
   string1 = [mutableDictionary objectForKey:@"key1"];
   NSLog(@"The object for key, key1 in mutableDictionary is %@",string1);
   [pool drain];
   return 0;
}
```

Now when we compile and run the program, we will get the following result.

```
2013-09-29 02:34:50.528 demo[9135] The object for key, key1 in dictionary is string1
2013-09-29 02:34:50.528 demo[9135] The object for key, key1 in mutableDictionary is
string
```

NSSet & NSMutableSet

NSSet is used to hold an immutable set of distinct objects and the NSMutableDictionary is used to hold an mutable set of distinct objects.

Important methods of NSSet are as follows:

- alloc/initWithObjects: Initializes a newly allocated set with members taken from the specified list of objects.
- allObjects - Returns an array containing the set's members or an empty array if the set has no members.
- count: Returns the number of members in the set.

NSMutableSet is inherited from NSSet and hence all instance methods of NSSet are available in NSMutableSet.

Important methods of NSMutableSet are as follows:

- removeAllObjects: Empties the set of all of its members.
- addObject: Adds a given object to the set, if it is not already a member.
- removeObject: Removes a given object from the set.

A simple example for sets is shown below:

```
#import <Foundation/Foundation.h>

int main()
{
   NSAutoreleasePool * pool = [[NSAutoreleasePool alloc] init];
   NSSet *set = [[NSSet alloc]
   initWithObjects:@"string1", @"string2",@"string3",nil];
   NSArray *setArray = [set allObjects];
   NSLog(@"The objects in set are %@",setArray);
   NSMutableSet *mutableSet = [[NSMutableSet alloc]init];
   [mutableSet addObject:@"string1"];
   setArray = [mutableSet allObjects];
   NSLog(@"The objects in mutableSet are %@",setArray);
   [pool drain];
   return 0;
}
```

Now when we compile and run the program, we will get the following result.

```
2013-09-29 02:35:40.221 demo[12341] The objects in set are (string3, string2, string1)
2013-09-29 02:35:40.222 demo[12341] The objects in mutableSet are (string1)
```