* **Name: Manar Abdelbaset Mohamed**
* **Track: Mobile Native Development**

Retain Cycle

**What is Retain Cycle?**

From what I read Retain Cycle Is Done in Auto garbage collecting Environments Like ARC (Automatic Reference Counting) maybe, As When the Class 1 Referring (Strong reference) or (Retain) the Class 2 and the Class 2 Referring (Retain) or (Strong reference) the Class 1, they continue to exist, as dealloc method can't be called in Class 1 as Class 2 refer to it and it's count won't reach zero, so the Class 2 dealloc method as well won't be called and release for this reference won't happen, so they keeping themselves alive causing leakage in Memory.

**Simple Example For the Concept I understood:**

@interface FirstClass : NSObject

@property SecondClass \* secondClass;

@end

@implementation FirstClass

-(void) dealloc{

NSLog(@"dealloc FirstClass ");

}

@end

@interface SecondClass : NSObject

@property FirstClass \* firstClass;

@end

@implementation SecondClass

-(void) dealloc{

NSLog(@"dealloc SecondClass ");

}

@end

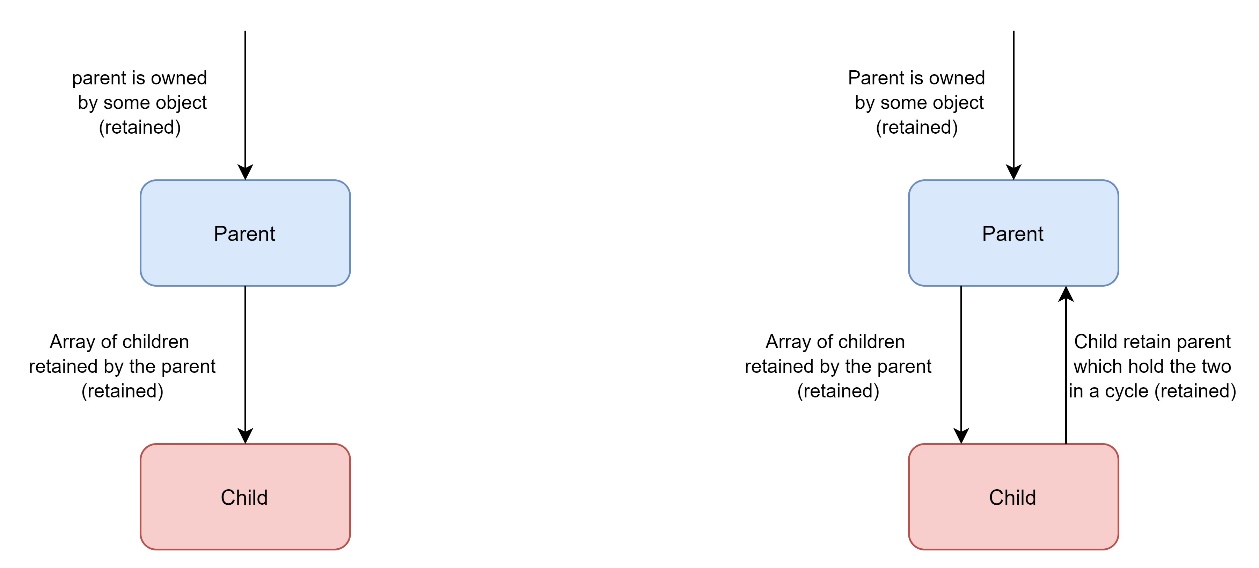
In Main something like that can be called:

* FirstClass \* firstClass = [[FirstClass alloc] init];
* SecondClass \* secondClass = [[SecondClass alloc] init];
* firstClass. secondClass = secondClass;
* department. firstClass = firstClass;

This is An Example Where retain cycle can take place.

If Parent object is retained by some class and the parent class has retained array of children. If the parent object is no longer needed and become nil, dealloc function will be called and memory will deallocated it leading to call release in all the children, in this case Retain Cycle won't happen and everything will go smooth with no proplem with releasing objects, this case I've drawn it in the bellow figure in the left diagram.

But if parent object retained by some object and parent itself retains array of children where each retain a reference to the parent, in this case the retain counter won't be zero ever and dealloc function won't be called as the counter Is increased due to children retains their parent, in this case SURE Retain Cycle also will happen as shown in the figure I've made the right diagram.



**Note I've read about it:**

**Difference between Strong and weak reference in Objective C:**

**A Weak reference is one that does NOT RETAIN (hold strong to) its target.** As in the previous Example, if the Class 1 (First Class) reference in the second class was weak reference, No Retain Cycle would happen as weak reference (pointer) is like a trigger for the containing object to know that this reference can be deallocated and become null (nil) at any time so don't retain it or hold to it. To make the previous example weak reference we should define the property in the Class 2 to be @property(weak)

**A Strong reference is one that does RETAIN its target**. As in previous Example it was strong reference as the default of property in ARC is strong reference so this leads for the retain cycle to happen here and FirstClass object won't be deallocated and dealloc won't be called as counter won't reach zero due to firstclass strong reference in Class 2.

**To Avoid this Cycle we should take care of the following:**

1. **Never** make a strong reference for 2 classes to each other as they would surely make retain cycle.
2. An object must **Never** retain its parent so The parent must set the parent pointer in the child to nil when the relationship is broken.
3. If a parent has array of children (retained), so children MUST NOT be retained to the parent or any of its ancestors (weak reference to it or no reference to parent).