存储器

# RAM&ROM

RAM（random access memory)：运行内存，不能掉电存储。CPU可以直接访问。

ROM（read-only memory)：存储性内存，可以掉电存储，例如内存卡、Flash。 由于RAM类型不具备掉电存储能力（即一掉电数据消失），所以app程序一般存放于ROM中。RAM的访问速度要远高于ROM，价格也要高。

# APP启动

App程序启动，系统会把开启的那个App程序从ROM里面拷贝到内存（RAM），然后从内存里面执行代码。

# 内存分区

1. 栈区(stack)
2. 堆区(heap)
3. 全局/静态去(bss)
4. 常量区(data)
5. 代码区(code)

## 栈

1. FIFO先进先出
2. 栈又称堆栈, 是用户存放程序临时创建的局部变量,也就是说我们函数括弧“{}” 中定义的变量(但不包括static声明的变量,static意味着在数据段中存放变量)
3. 指针都存在栈区，用于指向分配在堆区的内存的地址。

## 堆

1. 堆是用于存放进程运行中被动态分配的内存段,它的大小并不固定,可动态扩张或缩减。当进程调用malloc等函数分配内存时,新分配的内存就被动态添加到堆上(堆被扩张); 当利用free等函数释放内存时,被释放的内存从堆中被剔除(堆被缩减)
2. 堆向高地址扩展的数据结构，是不连续的内存区域。程序员负责在何时释放内存（如用free或delete），在iOS的ARC程序中，系统自动管理计数器，计数器为0的时候，在当次的runloop结束后，释放掉内存。堆中的所有东西都是匿名的，这样不能按名字访问，而只能通过指针访问。
3. 对于堆来讲,频繁的new/delete势必会造成内存空间的不连续性，从而造成大量的碎片 ,使程序效率降低。

## 全局/静态区(bss)

1. bss段通常是指用来存放程序中未初始化的全局变量和静态变量的一块内存区域。
2. 通常来说如果不初始化全局变量和静态变量，编译器也会对它们进行一个隐式初始化（直接赋值就是显示初始化），赋给它们一个缺省值，是我们这里所说的未初始化。
3. bss段在程序执行之前会清0，所以未初始化的全局变量（静态变量）已经是0了。所以这种情况还是存放在BSS段，一旦初始化就会从BSS段中回收掉，转存到data段(数据段)中。
4. bss区-Block Started by Symbol（未初始化数据段）：并不给该段的数据分配空间，仅仅是记录了数据所需空间的大小。

## 文字/常量区(data)

1. 数据段分为只读数据段(常量区)和读写数据段
2. 通常是指用来存放程序中已经初始化的全局变量和静态变量的一块内存区域。数据段属于静态内存分配,可以分为只读数据段和读写数据段。字符串常量等,是放在只读数据段中，结束程序时才会被收回。

## 代码区(code)

通常是指用来存放程序执行代码的一块内存区域。这部分区域的大小在程序运行前就已经确定,并且内存区域通常属于只读, 某些架构也允许代码段为可写,即允许修改程序。在代码段中,也有可能包含一些只读的常数变量,例如字符串常量等,这些常量放在只读数据段(data segment)中，也有叫做常量区的说法。

图示

描述已自动生成

图形用户界面, 应用程序

描述已自动生成