

Быстрое выделение памяти под множество однотипных объектов

Для @CocoaHeadsMsk

Александр Черный
<http://chernyy.ru/>

@mail.ru

Откуда ноги растут?



Два класса

```
// First

@interface First : NSObject
{
@public
    uint32 first;
}

@end

@implementation First
@end
```

```
// Second

@interface Second : First
{
@public
    uint32 second;
}

@end

@implementation Second
@end
```

Second → First → NSObject

isa — всему голова?

```
Second *second = [[Second alloc] init];  
second->first = 0x11111111;  
second->second = 0x22222222;  
  
NSUInteger secondObjectSize = class_getInstanceSize([second class]);  
NSData *secondObjectData = [NSData dataWithBytes:second length:secondObjectSize];  
  
NSLog(@"Second object data: %@", secondObjectData);
```

2012-09-28 13:32:57.062 ISAIIsFirst[17206:707] Second object data: <b0300000 11111111 22222222>

isa?



Поля NSObject

```
/****** Base class *****/  
  
NS_ROOT_CLASS  
@interface NSObject <NSObject> {  
    Class isa;  
}  
  
...  
+ (Class)superclass;  
+ (Class)class;  
...  
  
@end
```

1

```
typedef struct objc_class *Class;  
typedef struct objc_object {  
    Class isa;  
} *id;
```

2

```
struct objc_class {  
    Class isa;  
  
#if !__OBJC2__  
    Class super_class  
    const char *name  
    long version  
    long info  
    long instance_size  
    struct objc_ivar_list *ivars  
    struct objc_method_list **methodLists  
    struct objc_cache *cache  
    struct objc_protocol_list *protocols  
#endif  
  
} OBJC2_UNAVAILABLE;  
/* Use `Class` instead of `struct objc_class` */
```

3

Добавим эталон

```
Second *second = [[Second alloc] init];

second->first = 0x11111111;
second->second = 0x22222222;

NSUInteger secondObjectSize = class_getInstanceSize([second class]);
NSData *secondObjectData = [NSData dataWithBytes:second length:secondObjectSize];

NSLog(@"Second object data: %@", secondObjectData);
NSLog(@"Second superclass of superclass isa: %p", [[second superclass] superclass]);

NSObject *etalonObject = [[NSObject alloc] init];

NSUInteger etalonObjectSize = class_getInstanceSize([etalonObject class]);
NSData *etalonObjectData = [NSData dataWithBytes:etalonObject length:etalonObjectSize];

NSLog(@"Etalon object data: %@", etalonObjectData);
NSLog(@"Etalon object isa: %p", [etalonObject class]);

[second release];
[etalonObject release];
```


isa — ВСЕМУ ГОЛОВА

```
2012-09-28 13:32:57.062 ISAIIsFirst[17206:707] Second object data: <b0300000 11111111 22222222>  
2012-09-28 13:32:57.067 ISAIIsFirst[17206:707] Second superclass of superclass isa: 0xac4bad1c  
2012-09-28 13:32:57.068 ISAIIsFirst[17206:707] Etalon object data: <1cad4bac>  
2012-09-28 13:32:57.070 ISAIIsFirst[17206:707] Etalon object isa: 0xac4bad1c
```

[second class] — Second

[second superclass] — First


[[second superclass] superclass] — NSObject

[etalonObject class] — NSObject

Кроме isa у etalonObject нет данных

NSString

Старый знакомый
И полей нет



```
@interface NSString : NSObject <NSCopying, NSMutableCopying, NSSecureCoding>

/*
    NSString primitive (funnel) methods.
    A minimal subclass of NSString just needs to implement these,
    although we also recommend getCharacters:range:.
    See below for the other methods.
*/

- (NSUInteger)length;
- (unichar)characterAtIndex:(NSUInteger)index;

@end
```

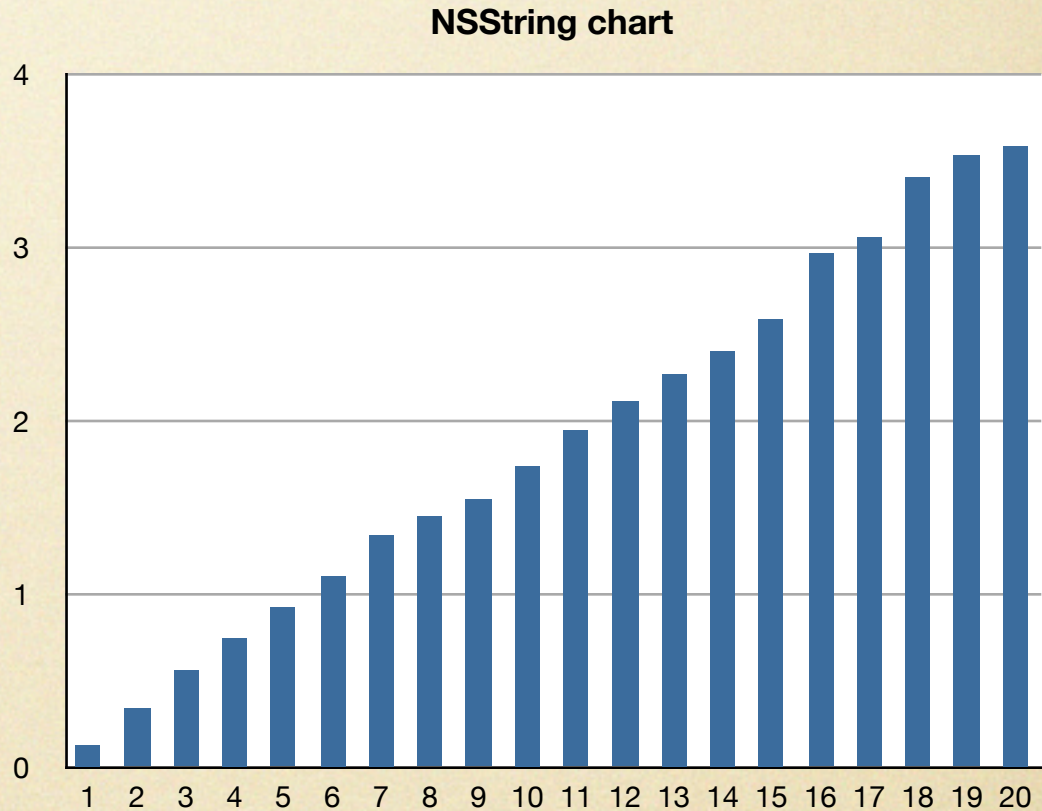

NSArray

```
/** ***** Immutable Array ***** */  
@interface NSArray : NSObject <NSCopying, NSMutableCopying, NSSecureCoding, NSFastEnumeration>  
- (NSUInteger)count;  
- (id)objectAtIndex:(NSUInteger)index;  
  
@end
```


Код

NSString

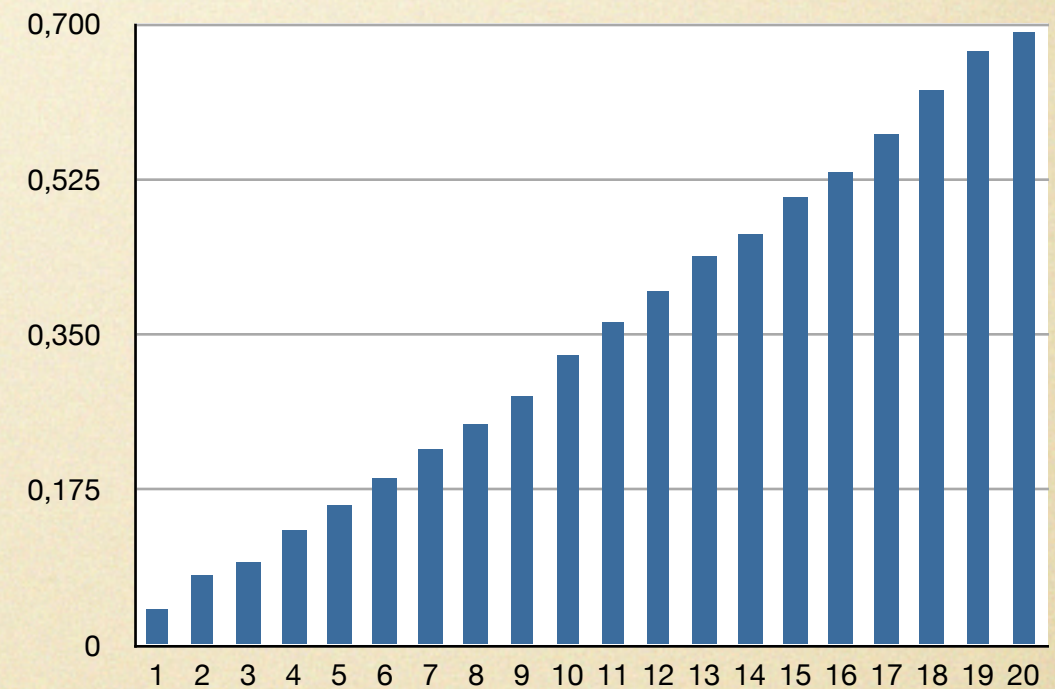
Experiment №	Time	Memory
1	0,131251	18276352
2	0,341883	18739200
3	0,559789	18739200
4	0,748177	18739200
5	0,923315	18739200
6	1,106459	18739200
7	1,341331	18739200
8	1,449332	18739200
9	1,547059	18739200
10	1,740061	18739200
11	1,943999	18739200
12	2,112459	18739200
13	2,270719	18739200
14	2,403548	18739200
15	2,587277	18739200
16	2,965954	18739200
17	3,056996	18739200
18	3,404000	18739200
19	3,530126	18739200
20	3,587230	18739200



ASCIIString

Experiment №	Time	Memory
1	0,040730	7626752
2	0,078444	8085504
3	0,093218	8085504
4	0,129534	8085504
5	0,157785	8085504
6	0,187519	8085504
7	0,220596	8085504
8	0,248450	8085504
9	0,280563	8085504
10	0,326714	8085504
11	0,363912	8085504
12	0,398687	8085504
13	0,437514	8085504
14	0,462569	8085504
15	0,503980	8085504
16	0,532753	8085504
17	0,575265	8085504
18	0,624677	8085504
19	0,669445	8085504
20	0,690897	8085504

ASCIIString chart



Еще несколько слов

- О выделении памяти
- Об автоматической оптимизации
- О PascalString
- Об объемах данных
- О посылке сообщения
- О вреде курения

That's all, folks!

```
NSDeveloper *alexChernyy = [NSDeveloper  
developerWithMail:@"alex@chernyy.ru"  
twitter:@"@alexchernyy"];
```