

Q: How do you store and access elements in the Storage class?

A: Although Storage's instance variables permit you to access the data directly, this approach is discouraged. Instead, use the Storage object's methods (such as `addElement:` and `elementAt:`) to store and access the elements.

Following is a code snippet that creates an empty Storage object, adds new elements to it, and then references them. The type-specific code is `#defined` for clarity. Note that the elements need to be added and referenced as pointers.

```
#import <objc/Storage.h>
```

```
@interface SomeObject:Object
```

```
@end
```

```
@implementation SomeObject
```

```
#if 1
```

```
#define TYPE char *
```

```
#define PRINT(var1, var2) printf("first = %s second = %s\n", var1, var2);
```

```
#define VAL1 ("hello")
```

```
#define VAL2 ("world")
#else
#define TYPE int
#define PRINT(var1, var2) printf("first = %d second = %d\n", var1, var2);
#define VAL1 (-5)
#define VAL2 (32)
#endif
```

```
- appDidInit:sender
```

```
{
    Storage *store;
    TYPE a = VAL1;
    TYPE b = VAL2;
    TYPE *a1;
    TYPE *b1;

    store = [[Storage alloc] initWithCount:0 elementSize:sizeof(TYPE)
                                   description:@encode(TYPE)];
    [store addElement:(void *)&a];
    [store addElement:(void *)&b];
    PRINT(a, b);
}
```

```
        a1 = (TYPE *) [store elementAt:0];  
        b1 = (TYPE *) [store elementAt:1];  
        PRINT(*a1, *b1);  
  
    return self;  
}
```

QA504

Valid for 2.0, 3.0