iOS

Objective-C Primer
Xcode provides everything developers need to create great applications for Mac, iPhone, and iPad. Xcode 4 has been streamlined to help you write better apps. It has unified user interface design, coding, testing, and debugging all within a single window. The Xcode IDE analyzes the details of your project to identify mistakes in both syntax and logic, it can even help fix your code for you.

Xcode runs on OS X Lion and includes the Xcode IDE, Instruments, iOS Simulator, the latest Mac OS X and iOS SDKs, and hundreds of powerful features:

More by Apple

- OS X Lion
- Final Cut Pro
- Pages
- Logic Pro
iOS Dev Center

```c
#include <stdio.h>

int main(int argc, const char * argv[]) {
    printf("Hello, World!\n");
    return 0;
}
```
statements

printf("Hello, World!\n");
variables

int n;
Primitive Data Types

char
double
float
int
long
unsigned int
...
...
printf

%s
%d
%lu
%lld
%f
...

Boolean Expressions

!  >  >=  ==  <=  <  &&  ||
Conditions

if  else
Loops
for (initialization; condition; increment) {
    statements
}

while (condition) {
  statements
}

do { 
  statements
} while (condition);
Casting
Pointers

c char *
d double *
f float *
i int *
l long *
...

struct
enum
Arrays

[]
Memory Management

malloc    free
Objective-C
Data Types

BOOL
id
nil
...

...
Foundation Data Types

NSInteger
NSPoint
NSRect
NSSize
NSSUInteger
...
...
@interface Foo: NSObject {
    // instance variables
}
// declarations of methods
@end
@implementation Foo

// definitions of methods
@end
Instance Variables

@protected
@private
@public
Class Methods

+alloc;
Messages

Student *student = [Student alloc];
Instance Methods

- (int)age;
- (void)setAge:(int)age;
- (void)init;
- (void)initWithName:(NSString *)name andAge:(int)age;
[student age];

[student setAge:20];

[student init];

[student initWithName:@"Alice" andAge:20];
Selectors

alloc

age

setAge:

init

initWithName:andAge:
@property

assign

copy

strong

weak

atomic

nonatomic

readonly

readwrite
@synthesize
Collections

NSArray
NSMUTABLEARRAY

NSDICTIOnary
NSMUTABLEDICTIONARY

NSSet
NSMUTABLESET

...
Fast Enumeration

for (id foo in bar) {
  // do something with foo
}

Categories

@interface Foo (Bar)
-(void)baz;
@end
@interface Student <NSCopying> {
}

...

@end
Protocols

@implementation Student
...
-(id)copyWithZone:(NSZone *)zone
{
    Student *s = [Student allocWithZone:zone];
    [s initWithName:_name andAge:_age];
    return s;
}
...
@end
@try {
    // try something here
}
@catch (NSEException *e) {
    // handle exception here
}
@finally {
    // do something here
}
NSError

NSError *e = nil;
if ([foo bar:baz error:&e] == nil)
{
    // handle error
}
to be continued...