encodeWithCoder:

initWithCoder:

NSObject autorelease

class
conformsToProtocol:

retain
retainCount
self
zone

init
+ new
dealloc

Identifying classes+ class
+ superclass

Testing class functionality+ instancesRespondToSelector:

Testing protocol conformance+ conformsToProtocol:

Obtaining method information methodForSelector:
+ instanceMethodForSelector:
methodSignatureForSelector:

Describing objects+ description
description

Posing+ poseAsClass:

Error handling doesNotRecognizeSelector:

Forwarding messages forwardInvocation:

Dynamic loading+ load

Archiving awakeAfterUsingCoder:
classForArchiver
classForCoder
replacementObjectForArchiver
replacementObjectForCoder:
+ setVersion:
+ version

init, + new

init

class (NSObject protocol)

conformsToProtocol:

description

init, class (NSObject protocol)

methodForSelector:

respondsToSelector:, forwardInvocation:

init, + alloc, + allocWithZone:

superclass

If a replacement takes place, the implementation of awakeAfterUsingCoder: is responsible for rele

initWithCoder:(NSCoding protocol)

(Class)classForArchiver

Identifies the class to be used during archiving. NSObject's implementation returns the object retu
.

(Class)classForCoder

Identifies the class to be used during coding. An NSObject returns its own class by default.

class (NSObject protocol)

(void)dealloc

Deallocates the memory occupied by the receiver. Subsequent messages to the object will generate
a message was sent to a freed object (provided that the freed memory hasn't been reused yet).

You never send a dealloc message directly. Instead, an object's dealloc method is invoked indirec
method. See the introduction to the Foundation Kit for more details on the use of these methods.

Subclasses must implement their own versions of dealloc to allow the deallocation of any additiona
by the objectÐsuch as dynamically allocated storage for data, or other objects that are tightly coup
and are of no use without it. After performing the class-specific deallocation, the subclass method
superclass versions of dealloc through a message to super:

release (NSObject protocol), autorelease (NSObject protocol)

(NSString *)description

Returns a string object that represents the contents of the receiver. The debugger's print-object co
method to produce a textual description of an object.

NSObject's implementation of this method simply prints the name of the receiver's class and the h
id.

(void)doesNotRecognizeSelector:(SEL)aSelector

forwardInvocation:

(void)forwardInvocation:(NSInvocation *)anInvocation

Implemented by subclasses to forward messages to other objects. When an object is sent a messag
corresponding method, the run-time system gives the receiver an opportunity to delegate the messa
It does this by creating an NSInvocation object representing the message and sending the receiver
message containing this NSInvocation as the argument. The receiver's forwardInvocation: method
forward the message to another object. (If the delegated receiver can't respond to the message eith
chance to forward it.)

The forwardInvocation: message thus allows an object to establish relationships with other objects
messages, act on its behalf. The forwarding object is, in a sense, able to ªinheritº some of the char
it forwards the message to.

A forwardInvocation: message is generated only if the message encoded in anInvocation isn't impl
receiving object's class or by any of the classes it inherits from.

An implementation of the forwardInvocation: method has two tasks:

·To locate an object that can respond to the message encoded in anInvocation. This need not be th
messages.

·To dispatch the message to that object.

In the simple case, in which an object forwards messages to just one destination (such as the hypot
variable in the example below), a forwardInvocation: method could be as simple as this:

doesNotRecognizeSelector:

(unsigned int)hash

Returns an unsigned integer that can be used as a table address in a hash table structure. For NSOb
value based on the object's id. If two objects are equal (as determined by the isEqual: method), the

An init message is generally coupled with an alloc or allocWithZone: message in the same line of c

(IMP)methodForSelector:(SEL)aSelector

Locates and returns the address of the receiver's implementation of the aSelector method, so that it function. If the receiver is an instance, aSelector should refer to an instance method if the receiver refer to a class method.

aSelector must be a valid, non-NULL selector. If in doubt, use the respondsToSelector: method to the selector to methodForSelector:.

IMP is defined as a pointer to a function that returns an id and takes a variable number of argumen two ªhiddenº argumentsÐ self and _cmdÐthat are passed to every method implementation):

(NSMethodSignature *)methodSignatureForSelector:(SEL)aSelector

Returns an NSMethodSignature object that contains a description of the aSelector method, or nil if can't be found. When the receiver is an instance, aSelector should be an instance method when the should be a class method. This method is mostly used in the implementation of protocols.

(id)replacementObjectForArchiver:(NSArchiver *)anArchiver

Allows an object to substitute another object for itself during archiving. NSObject's implementatio returned by replacementObjectForCoder:.

replacementObjectForCoder:(NSCoder *)encoder