NSScanner

Inherits From:	NSObject
Conforms To:	NSCoding NSCopying NSObject
Declared In:	foundation/NSScanner.h

Creating an NSScanner

+ (id)**localizedScannerWithString:**(NSString *)*aString*	Creates and returns a scanner that scans *aString*. Invokes **initWithString:** and sets the locale to the user's default locale.
+ (id)**scannerWithString:**(NSString *)*aString*	Creates and returns a scanner that scans *aString*.
- (id)**initWithString:**(NSString *)*aString*	Initializes the receiver, a newly allocated scanner, to scan *aString*. Returns **self**.

Getting an NSScanner's String

- (NSString *)**string**	Returns the string object that the scanner was created with.

Configuring an NSScanner

- (BOOL)**caseSensitive**	Returns YES if the scanner distinguishes case, and NO otherwise. Scanners are by default *not* case sensitive.
- (NSCharacterSet *)**charactersToBeSkipped**	Returns a character set containing those characters that the scanner ignores when looking for an element. The default set is the whitespace and newline character set.
- (NSDictionary *)**locale**	Returns a dictionary object containing locale information. Returns **nil** if the locale dictionary has not been set.
- (unsigned)**scanLocation**	Returns the character index at which the scanner will begin its next scanning operation.
- (void)**setCaseSensitive:**(BOOL)*flag*	If *flag* is YES, the scanner considers case when scanning characters. If *flag* is NO, it ignores case distinctions. NSScanners are by default *not* case sensitive.
- (void)**setCharactersToBeSkipped:**(NSCharacterSet *)*aSet*	Sets the scanner to ignore characters from *aSet* when scanning its string.
- (void)**setLocale:**(NSDictionary *)*localeDictionary*	Sets the receiver's dictionary object containing locale information.
- (void)**setScanLocation:**(unsigned int)*anIndex*	Sets the location at which the next scan will begin to *anIndex*.

Scanning a String

- (BOOL)**scanCharactersFromSet:**(NSCharacterSet *)*aSet* **intoString:**(NSString **)*value*	Scans the string as long as characters from *aSet* are encountered, accumulating characters into an optional string that's returned by reference in *value*. If any characters are scanned, returns YES; otherwise returns NO.
- (BOOL)**scanDouble:**(double *)*value*	Scans a **double** into *value* if possible. Returns YES if a valid floating-point expression was scanned; NO otherwise. HUGE_VAL or -HUGE_VAL is put in *value* on overflow; 0.0 on underflow. Returns YES in overflow and underflow cases.
- (BOOL)**scanFloat:**(float *)*value*	Scans a **float** into *value* if possible. Returns YES if a valid floating-point expression was scanned; NO otherwise. HUGE_VAL or -HUGE_VAL is

	put in *value* on overflow; 0.0 on underflow. Returns YES in overflow and underflow cases.
- (BOOL)**scanInt:**(int *)*value*	Scans an **int** into *value* if possible. Returns YES if a valid integer expression was scanned; NO otherwise. INT_MAX or INT_MIN is put in *value* on overflow. Returns YES in overflow cases.
- (BOOL)**scanLongLong:**(long long *)*value*	Scans a **long long int** into *value* if possible. Returns YES if a valid integer expression was scanned; NO otherwise. LONG_LONG_MAX or LONG_LONG_MIN is put in *value* on overflow. Returns YES in overflow cases.
- (BOOL)**scanString:**(NSString *)*aString* **intoString:**(NSString **)*value*	Scans for *aString*, and if a match is found returns by reference in the optional *value* argument a string object equal to it. If *aString* matches the characters at the scan location, returns YES; otherwise returns NO.
- (BOOL)**scanUpToCharactersFromSet:**(NSCharacterSet *)*aSet* **intoString:**(NSString **)*value*	Scans the string until a character from *aSet* is encountered, accumulating characters encountered into a string that's returned by reference in the optional *value* argument. If any characters are scanned, returns YES; otherwise returns NO.
- (BOOL)**scanUpToString:**(NSString *)*aString* **intoString:**(NSString **)*value*	Scans the string until *aString* is encountered, accumulating characters encountered into a string that's returned by reference in the optional *value* argument. If any characters are scanned, returns YES; otherwise returns NO.
- (BOOL)**isAtEnd**	Returns YES if the scanner has exhausted all characters in its string; NO if there are characters left to scan.