Objective-C Variables:

* Variable is nothing but a name given to a storage area that our programs can manipulate.
* Each variable in Objective-C has a specific type, which determines the size and layout of the variable's memory; the range of values that can be stored within that memory; and the set of operations that can be applied to the variable.
* The name of a variable can be composed of letters, digits, and the underscore character.
* It must begin with either a letter or an underscore.
* Upper and lowercase letters are distinct because Objective-C is case sensitive.
* The following are basic variable types:

|  |  |
| --- | --- |
| **Type** | **Description** |
| char | Typically a single octet (one byte). This is an integer type. |
| int | The most natural size of integer for the machine. |
| float | A single-precision floating point value. |
| double | A double-precision floating point value |
| void | Represents the absence of type. |

* Objective-C programming language also allows defining various other types of variables, like Enumeration, Pointer, Array, Structure, Union, etc.

**Variable Definition in Objective-C:**

* A variable definition means to tell the compiler where and how much to create the storage for the variable.
* A variable definition specifies a data type and contains a list of one or more variables of that type as follows:

**Syntax**: type variable list;

* Here, type must be a valid Objective-C data type including char, int, float, double, bool or any userdefined object, etc., and variable list may consist of one or more identifier names separated by commas.
* Some valid declarations are shown below:

int i, j, k;

char c, ch;

float f, salary;

double d;

* The line int i, j, k; both declares and defines the variables i, j and k; which instructs the compiler to create variables named i, j and k of type int.
* Variables can be initialized (assigned an initial value) in their declaration. The initializer consists of an equal sign followed by a constant expression as follows:

**Syntax**: type variable name = value;

**Examples:**

//defining & initializing Boolean type data type

BOOL checkBooleanOrNot = YES;

//defing and initializing the char val of letter m

char definingCharacter = 'm';

//Defining & initializing Single pricision val to variable

float aFloatingPointVal = 73.956;

//Double pricession floating point

double storingDoubleVal = -21.847;

//Defining & initializing an intiger val to anIntegerTypeOfVal variable

int anIntegerTypeOfVal = -1234678353;

//defining & storing an long type of val

long longTypeOfVal = -46476432131709874;

//Defing a short type of val & storing it into storingShortTypeVal Variable

short storingShortTypeVal = -1453;

Objective C DATA TYPES:

* In the Objective-C programming language, data types refer to declaring variables or functions of different types.
* The type of a variable determines how much space it occupies in storage and how the bit pattern stored is interpreted.

The types in Objective-C can be classified as follows:

**Integer Types**

Following table gives the details about standard integer types with its storage sizes and value ranges:

|  |  |  |
| --- | --- | --- |
| **Type** | **Storage size** | **Value range** |
| char | 1 byte | -128 to 127 or 0 to 255 |
| unsigned char | 1 byte | 0 to 255 |
| signed char | 1 byte | -128 to 127 |
| Int | 4 bytes | 32,768 to 32,767 or -2,147,483,648 to 2,147,483,647 |
| unsigned int | 4 bytes | 0 to 65,535 or 0 to 4,294,967,295 |
| short | 2 bytes | -32,768 to 32,767 |
| unsigned short | 2 bytes | 0 to 65,535 |
| long | 4 bytes | -2,147,483,648 to 2,147,483,647 |
| unsigned long | 4 bytes | 0 to 4,294,967,295 |
| long long | 8 bytes | A double long |
| BOOL | 1 byte | Boolean (signed char YES or NO) |

**Floating-Point Types:**

Following table gives you details about standard float-point types with storage sizes and value ranges and their precision:

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Storage size** | **Value range** | **Precision** |
| float | 4 byte | 1.2E-38 to 3.4E+38 | 6 decimal places |
| double | 8 byte | 2.3E-308 to 1.7E+308 | 15 decimal places |
| long double | 10 byte | 3.4E-4932 to 1.1E+4932 | 19 decimal places |

**Basic Data Types**

Some of the more common data types we use in Objective-C include:

* **int** – An integer value, i.e. a whole number (no decimals) that includes zero and negative numbers.
* **float** – A floating point value that includes as many decimal places as it can hold. Because the decimal place can change, or float, it’s important to know that these values may technically be imprecise. When precise decimals are needed, like for currency, we should use the NSDecimalNumber data type.
* **BOOL** – Short for “boolean”, this is a 1-bit “true” or “false” value that can only be in one of those states. The C language (and hence, Objective-C) treat 0 as “false” and 1 as “true”. As such, the following keywords can be used to represent true/false values: YES/NO, TRUE/FALSE, true/false, 1,0.
* **char** – A single character, such as the letter A or the symbol “#”. Note that lowercase and uppercase characters are different, so “a” and “A” are two different characters.
* **NSString** – String data is a bunch of characters strung together to make text, like a banner strung up at a party.
* **NSNumber** – This class is a lightweight “wrapper” class that gives object-oriented features to the primitive number types mentioned above (among others).

**Defining Constants in Objective-C**

There are the following two simple ways in Objective-C to define constants:

* Using #define preprocessor
* Using const keyword

**Objective-C The #define Preprocessor**

Following is the syntax to use #defines preprocessor to define a constant:

Syntax:

#define identifier value

Let's look at the following example:

Objective-C Example Source Code:

#import <Foundation/Foundation.h>

#define LENGTH 5

#define WIDTH 10

#define NEWLINE '\n'

int main()

{

int area;

area = LENGTH \* WIDTH;

NSLog(@"value of area : %d", area);

NSLog(@"%c", NEWLINE);

return 0;

}