

ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITUTE OF TECHNOLOGY

CENTER OF INFORMATION TECHNOLOGY AND SCIENTIFIC COMPUTING

**Web Design and Development Assignment**

**Name:** - Kaleb Fekadu

ID:- ATR/6065/11

Section:- IT

January 2021

Table of Content

# Introduction

# Compiled Vs. Interpreted Programming Languages

In order to be compiled and executed programming languages use different approaches. One of the common ways to differentiate them is to split them into two compiled and interpreted languages.

The main purpose of Compilation and Interpretation is to transform human readable source codes into machine codes so that it can be executed.

#### Compiled Languages

A compiled language is a programming language that is implemented using compilers rather than interpreters. A compiler is a program that translates the source code written in a particular programming language into a machine code.

#### Interpreted Languages

An interpreted language is a programming language that is implemented using interpreters. The source code is not directly compiled into machine code rather it will be read and directly executed, line by line.

**Is JavaScript Compiled or interpreted language?**

According to sources in the internet JavaScript used to be an interpreted language. But in modern approaches the fact that determines JavaScript as an interpreted language is under a question mark. Why?

In modern time JavaScript runtime environment JavaScript engines no longer just interpret JavaScript, they compile it.

For this we can take V8 engine as an example. V8 is JavaScript engine that provides runtime environment in which JavaScript executes while browsing with chrome.

JavaScript is internally compiled by V8 with just-in-time (JIT) compilation to speed up the execution. JIT or just-in-time compilation is a hybrid between compilation and interpretation. They work by analyzing the byte-code and decide which part should be fully compiled, and then the code will be compiled into machine code.

To conclude, JavaScript code indeed gets compiled. It is closer to be compiled than Interpreted. It is compiled every time.

**The History of “typeof null”**

In JavaScript “typeof” is an operator which provides information about the type of data contained inside a variable. These type of data are Undefined, Boolean, Number, Object, Function, and Null. “typeof null” returns object since null is considered as an empty value for an object. This operator is remnant from the first version of JavaScript in which values were stored in 32-bit units, and small type tags stored in the lower bits were consisted. There were five them:

* 000: object. The data is a reference to an object.
* 1: int. The data is a 31 bit signed integer.
* 010: double. The data is a reference to a double floating point number.
* 100: string. The data is a reference to a string.
* 110: Boolean. The data is a Boolean.

The type tags were three bits in length having two additional bits for four types.

Two values were special:

* Undefined (JSVAL\_VOID) was the integer −230 (a number outside the integer range).
* Null (JSVAL\_NULL) was the machine code NULL pointer. Or: an object type tag plus a reference that is zero.

From these the type of operator examines the type tag of NULL and understands NULL as an object, as “typeof null” returns an object type.

When defining a variable that is meant to later hold an object, it is advisable to initialize the variable to null as opposed to anything else.