**Para-C:** **C-based Coding designed to be simple and fast with new syntax and functionality**

LUNA KLATZER, HTBLA Leonding

LORENZ HOLZBAUER, HTBLA Leonding

Content

[1. Introduction 2](#_Toc71322691)

[2. Base structure 3](#_Toc71322692)

[2.1 File Structure 3](#_Toc71322693)

[2.1.1 Para-C modules and files 3](#_Toc71322694)

[2.1.2 Importing PARA-modules 3](#_Toc71322695)

[2.1.3 Importing C-modules 3](#_Toc71322696)

[2.2 Core Syntax 3](#_Toc71322697)

[2.3 Built-in Macros and defining Macros 3](#_Toc71322698)

[2.4 Structs and Type-def 3](#_Toc71322699)

[2.5 Exceptions 3](#_Toc71322700)

[3. Usage-examples of Para-C 3](#_Toc71322701)

[4. Para-C compiler and Compiling-Exceptions 3](#_Toc71322702)

[Footnotes 4](#_Toc71322703)

# Introduction

Para-C is a new programming language designed to integrate C and serve as a helper and simplifier to write better C code. C is a good language; it is flexible, fast, and easily expandable, which made it a great contester for the base of the modern higher-level programming languages and programs. Therefore, this new project has its core built around the C programming languages to use its flexibility and options to expand into the Object-Oriented Area using G-Object and introduce more complicated processes built on a simple structure. Still, unlike other languages built on top of C, like C#, C++, Java, etc., Para-C is intended to help writing code easier and faster. Including adding more features, like new built-in Macros and functions, OOP-structures, more straightforward array and malloc-handling, expanded data types and simplified functions. Still, unlike others to make an entirely new language with new syntax, systems and logic, it is built around to have just a simpler syntax and structure comparable to Python and TypeScript but can include directly code from C and use its speed in the compiled execution. So, programming in Para-C will be similar but still simpler and well looking due to the simplifications, additional structures and helper functions. Syntax-wise it will still lay onto C to avoid causing issues with more compiler code that would be required for a new syntax that can’t be easily integrated into the C-syntax. It is a programming language designed to help and provide features for writing in low-level areas. C is often the only option in such places (excluding Assembly). Additional features and help can significantly improve speed and functionality in less time than implementing it every time yourself.

Note: We intend Para-C not to be a widely „optimised“ or „production-ready“ programming language. It is solely a free-time project designed for learning and testing purposes, which we do not intend for anything other than that.

# Base structure

The structure of Para-C will closely lean to the C-Structure (C11-Standard) but still have its independent system apart from it. That means it will include its own: Structure for its Parac-modules and C-modules (*See Section 2.1 File Structure*), Import-structure, Built-in macros[[1]](#endnote-1), Built-in functions[[2]](#endnote-2) (which will partly replace the C-functions for easier handling) and OOP-Structure using GObject[[3]](#endnote-3) and system for integrating C-code and Macro definitions.

## 2.1 File Structure

The file structure in Para-C is similar to C. It will work mainly around the Para-C compiler folder with the executable and C-compiler materials and libraries that Para-C supports. (*See Section 3.1 Using C-Code inside Para-C)*

### 2.1.1 Para-C modules and files

### 2.1.2 Importing PARA-modules

### 2.1.3 Importing C-modules

## 2.2 Core Syntax

## 2.3 Built-in Macros and defining Macros

## 2.4 Structs and Type-def

## 2.5 Exceptions

# Usage-examples of Para-C

## 3.1 Using C-Code inside Para-C

### 3.1.1 Using Standard C-code

### 3.1.2 Using C-libraries

### 3.1.3 Restrictions of Para-C

# Para-C compiler and Compiling-Exceptions

Footnotes

1. List of Pre-defined macros in C: [[here]](https://gcc.gnu.org/onlinedocs/cpp/Predefined-Macros.html) [↑](#endnote-ref-1)
2. List of Built-in C-functions: [[here]](https://www.tutorialspoint.com/ansi_c/c_function_references.htm) [↑](#endnote-ref-2)
3. Introduction to GObject: [[here]](https://www.freedesktop.org/software/gstreamer-sdk/data/docs/latest/gobject/howto-gobject.html) [↑](#endnote-ref-3)