

# How C++ Works

Ryan Baker

January 15, 2025

## Contents

<b>1</b>	<b>The Build Process</b>	<b>2</b>
1.1	The Preprocessor . . . . .	2
1.1.1	Text Replacement <code>#define</code> . . . . .	2
1.1.2	Conditional Compilation <code>#if</code> , <code>#ifdef</code> . . . . .	2
1.1.3	File Inclusion <code>#include</code> . . . . .	2
1.2	The Compiler . . . . .	2
1.2.1	Compilation Errors . . . . .	2
1.3	The Linker . . . . .	2
1.3.1	Linker Errors . . . . .	2
<b>2</b>	<b>Memory and Pointers</b>	<b>2</b>
2.1	Introduction to Memory . . . . .	2
2.2	Pointers . . . . .	2
2.2.1	NULL Pointers . . . . .	2
2.2.2	Pointer Arithmetic . . . . .	2
2.2.3	Pointers to Pointers . . . . .	2
2.3	References . . . . .	2
<b>3</b>	<b>Memory Segments</b>	<b>2</b>
3.1	Text Segment . . . . .	2
3.2	Static Memory . . . . .	2
3.2.1	<code>static</code> Keyword . . . . .	2
3.2.2	Initialized vs. Uninitialized Static Data . . . . .	2
3.3	Heap Segment . . . . .	2
3.3.1	Operators <code>new</code> and <code>delete</code> . . . . .	2
3.3.2	Memory Leaks . . . . .	2
3.4	Stack Segment . . . . .	2
3.4.1	Stack Pointer . . . . .	2

# 1 The Build Process

## 1.1 The Preprocessor

### 1.1.1 Text Replacement `#define`

### 1.1.2 Conditional Compilation `#if`, `#ifdef`

### 1.1.3 File Inclusion `#include`

## 1.2 The Compiler

### 1.2.1 Compilation Errors

## 1.3 The Linker

### 1.3.1 Linker Errors

# 2 Memory and Pointers

## 2.1 Introduction to Memory

## 2.2 Pointers

### 2.2.1 NULL Pointers

### 2.2.2 Pointer Arithmetic

### 2.2.3 Pointers to Pointers

## 2.3 References

# 3 Memory Segments

## 3.1 Text Segment

## 3.2 Static Memory

### 3.2.1 `static` Keyword

### 3.2.2 Initialized vs. Uninitialized Static Data

## 3.3 Heap Segment

### 3.3.1 Operators `new` and `delete`

### 3.3.2 Memory Leaks

## 3.4 Stack Segment

### 3.4.1 Stack Pointer