Rust

Memory Thinking

Dmitry Vostokov
Software Diagnostics Services
Memory Thinking for Rust

Slides with Descriptions and Source Code Illustrations

Dmitry Vostokov
Software Diagnostics Services

OpenTask
Memory Thinking for Rust: Slides with Descriptions and Source Code Illustrations

Published by OpenTask, Republic of Ireland

Copyright © 2024 by OpenTask

Copyright © 2024 by Dmitry Vostokov

Copyright © 2024 by Software Diagnostics Services

Copyright © 2024 by Dublin School of Security

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, transmitted in any form or by any means, or used for training artificial intelligence systems without the prior written permission of the publisher.

OpenTask books are available through booksellers and distributors worldwide. For further information or comments, send requests to press@opentask.com.

Product and company names mentioned in this book may be trademarks of their owners.

A CIP catalog record for this book is available from the British Library.


Revision 1.00 (June 2024)
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table of Contents</td>
<td>3</td>
</tr>
<tr>
<td>Preface</td>
<td>15</td>
</tr>
<tr>
<td>About the Author</td>
<td>16</td>
</tr>
<tr>
<td>Introduction</td>
<td>17</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>18</td>
</tr>
<tr>
<td>Training Goals</td>
<td>19</td>
</tr>
<tr>
<td>Warning</td>
<td>20</td>
</tr>
<tr>
<td>Training Principles</td>
<td>21</td>
</tr>
<tr>
<td>Schedule</td>
<td>22</td>
</tr>
<tr>
<td>Training Idea</td>
<td>23</td>
</tr>
<tr>
<td>General Rust Aspects</td>
<td>24</td>
</tr>
<tr>
<td>What We Do Not Cover</td>
<td>26</td>
</tr>
<tr>
<td>Linux Rust Aspects</td>
<td>27</td>
</tr>
<tr>
<td>Windows Rust Aspects</td>
<td>28</td>
</tr>
<tr>
<td>Why Rust?</td>
<td>29</td>
</tr>
<tr>
<td>My Genealogy of Rust</td>
<td>31</td>
</tr>
<tr>
<td>Rust Mastery Process</td>
<td>32</td>
</tr>
<tr>
<td>Thought Process</td>
<td>33</td>
</tr>
</tbody>
</table>
# Philosophy of Unsafe Pointers

- **A General Pointer Concept**
- **Pointer**
- **Pointer Dereference**
- **Many to One**
- **Many to One Dereference**
- **Invalid Pointer**
- **Invalid Pointer Dereference**
- **Wild (Dangling) Pointer**
- **Pointer to Pointer**
- **Pointer to Pointer Dereference**
- **Naming Pointers and Entities**
- **Names as Pointer Content**
- **Pointers as Entities**

## Unsafe Rust Code Examples

- **Unsafe Pointer**
- **Unsafe Pointer Dereference**
- **Unsafe Many to One**
- **Unsafe Many to One Dereference**
Memory Dereference Layout 161
Names as Addresses 162
Addresses and Entities 163
Addresses and Structures 164
Pointers to Structures 165
Arrays 166
Arrays and Pointers to Arrays 167
Fat Pointers 168
Array Slices 169
String Literals (UTF-8) 173
Byte Strings 176
Vectors 178
Vector Slices 179
Strings 183
String Slices 184
C-Strings 187
C-String Slices 188
Basic Types 191
Bytes, References, and Pointers 192
u32, Pointers, and References

Little-Endian System

u64 and Pointers

Size

Alignment

Entity Conversion

Conversion through Pointers

Safe Conversion

Forcing

Tuples

Structs

Tuple-like Structs

Newtypes

Named-field Structs

Reference/Pointer to Struct

Ref/Ptr to Struct Dereference

Dereference with Replacement

Many Ref/Ptr to One Struct

Many to One Dereference
Struct Function Fields

Associated Functions

Pointers to Associated Functions

Type-associated Functions

Trait Functions

Trait Objects

vtable Memory Layout

Trait Object Memory Layout

Boxed Trait Object Layout

Struct Constructors

Struct Destructor

Struct Clone

Struct Copy

Parameters by Value

Parameters by Ref/Ptr

self

Closures and Captures

Closure Struct

A64 Closure Struct Example
Captures (Borrowing) 325
Captures (Borrowing) x64 Linux 326
Captures (Move) 328
Captures (Move) x64 Linux 329
Pinning 331
Use Cases 332
Rust Books 338