- 跳轉到博客目錄頁面[在線閱讀] [本地] *點擊此查看html網頁格式　pdf格式*

這篇文章比我在這裏分享的任何代碼和創業項目都重要，其中的發現關係到每一個人的方方面面。哲學比科學和技術更重要！哲學是人生，科學和技術只是喫飯而已！

心智是可以被操控的！心智是可以被操控的！心智是可以被操控的！你所不知道的5G/6G微波腦機接口技術！

點擊下面鏈接訪問

- 無眠月照無情門 . 失去自由的歌手 [點擊此前往github在線閱讀] 本地模式 html網頁版　pdf版本
- 心学新解 : https://github.com/brianwchh/worldofheart 本地模式 html網頁版　pdf版本

有必要學RUST嗎

RUST語言似乎用了過多的compiler自動生成代碼。語法爺相對比較複雜，是否真的如傳聞所說的能取代C？ 其垃圾內存“回收”是否完備？有些動態內存，似乎又還要程序員自己處理！* vs reference, *是不安全的訪問，會不會出現該變量被註銷了，而又不知的情況？尤其是在多線程的情況下。

拿來做app或許是個選項，用來做操作系統和寫驅動，感覺還是要底層一點的語言，C是寫操作系統的首選，寫操作系統，偏底層的語言更好，每個代碼都自己掌握，不需要編譯器過多干預和插入一些自己無法預知和把控的代碼。

然而作爲app開發語言，又似乎比Go複雜太多，畢竟一個垃圾回收的線程也不是太豪資源，現在的處理器都很強勁了。

所以想要面面俱到，註定會什麼都不如。寫操作系統不如C，寫app不如Go易學和設計簡單。RUST在處理heap和stack的靈活上，似乎到了C的程度。app有bug不那麼可怕，但操作系統有bug就不好了。

多學一門語言，也無妨。且用項目來對比看看。

學完C/Ç++，學其他語言會很簡單。學完rust，我們會對如何設計語言有更深的理解。