

《約耳趣談軟體》

图书基本信息

书名：《約耳趣談軟體》

13位ISBN编号：9789866348341

10位ISBN编号：9866348342

出版时间：2010-4-22

出版社：悅知

作者：Joel Spolsky

页数：361

译者：梅普華

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu000.com

《約耳趣談軟體》

內容概要

出版至今，全球已有超過30種語言譯本，全書45篇文章，廣受程式開發人員的喜愛，是一本集結關於軟體開發、人才遴選及培訓、創業經營，以及企業管理的實錄。

如果你正深陷於公司決策的迂腐、管理高層的無能、固執的做事信仰等無限迴圈中，那麼，約耳將是引領你走出暗黑的明燈。

歷任過軟體開發人員、管理者、經營者的約耳，所經手的專案比你吃的鹽還多；從開發人員關注的程式問題、硬體分享、產品管理，甚至是經濟學探討，無一不談，廣泛涉略只為你全盤掌握軟體專案的整體面貌。

網友狂推！不可錯過的章節：

約耳測試 邁向高品質程式碼的12個步驟：檢視團隊的分數，找到成為高品質團隊的方法。

五個世界：一再提醒你軟體開發具備各種不同的領域或世界，而不同的領域適用不同的規則，正視此規則才能在專案中悠遊自得。

面試人員教戰守則：人是軟體專案的重要部分，對的人讓專案進行更順利。此篇是面試官的精神食糧，更是徵試者的求生寶典。

抽象滲漏法則：揭開TCP的神祕本質，探討將一切抽象化的風險。

別讓架構太空人嚇到你：別讓自己也成為只會用抽象的名詞來解決事物的太空人。

大麥克對上原味主廚：別讓規則手冊取代了才能，因為規則手冊永遠無法適應新時代。

邊開火邊移動：作戰理論也能成為開發專案的密招？只要每天前進一點，成功的專案就在不遠之處。

書籍目錄

Part 1 位元與位元組：程式設計的實踐

- CH01 | 選擇一種語言
- CH02 | 回歸簡單原則
- CH03 | 約耳測試：邁向高品質程式碼的12步驟
- CH04 | 每位開發人員至少且絕對要會的Unicode及字元集必備知識
- CH05 | 無痛的功能規格1：何必麻煩？
- CH06 | 無痛的功能規格2：規格是什麼？
- CH07 | 無痛的功能規格3：但是...該怎麼做？
- CH08 | 無痛的功能規格4：提示
- CH09 | 無痛的軟體時程
- CH10 | 每日編譯是你的好朋友
- CH11 | 絕不妥協的抓蟲行動
- CH12 | 五個世界
- CH13 | 紙上原型製作
- CH14 | 別讓架構太空人嚇到你
- CH15 | 邊開火邊移動
- CH16 | 工匠技藝
- CH17 | 電腦科學中三個錯誤的想法
- CH18 | 雙元文化主義
- CH19 | 由用戶端自動取得當機回報，一切全自動！

Part 2 開發人員管理

- CH20 | 面試人員教戰守則
- CH21 | 激勵是有害的
- CH22 | 不用測試人員的五大（錯誤）藉口
- CH23 | 人的工作切換有害無益
- CH24 | 你絕對不應該做的事之一
- CH25 | 揭露冰山的秘密
- CH26 | 抽象滲漏法則
- CH27 | 程式設計領域的帕麥爾斯頓勳爵
- CH28 | 測量

Part 3 如果你是約耳：既定主題的隨機思考

- CH29 | Rick Chapman在尋找愚蠢
- CH30 | 這個國家的狗做什麼工作？
- CH31 | 小員工也能做大事
- CH32 | 兩則故事
- CH33 | 大麥克對上原味主廚
- CH34 | 沒有事情像表面看起來那麼簡單
- CH35 | 為非我發明症辯護
- CH36 | 策略書之一：Ben and Jerry模式與Amazon模式
- CH37 | 策略書之二：雞生蛋蛋生雞問題
- CH38 | 策略書之三：讓我換回去！
- CH39 | 策略書之四：腫脹軟體與80/20迷思
- CH40 | 策略書之五：開放源碼的經濟學
- CH41 | 墨菲定律發威的一週
- CH42 | 微軟如何輸掉API戰爭

Part 4 對於.NET，有點多的的評論

- CH43 | 微軟瘋了

CH44 | 我們的.NET策略

CH45 | 先生，可以賞我一個連結程式嗎？

Part 5 附錄：很棒的問題

精彩短评

1、繁体翻译版

http://local.joelonsoftware.com/wiki/The_Joel_on_Software_Translation_Project:%E5%AF%A6%E9%AB%94%E6%9B%B8

2、這本書大部分的內容都是作者好幾年前寫的（差不多都是千禧年那個年代），雖然中譯本是今年才出版，而且網路上可以找到整本書的中文翻譯版，不過，那都不重要，我之所以買這本書是因為作者舉例子的功力實在是太高強了。

章节试读

1、《約耳趣談軟體》的笔记-總結

激勵是有害的,主要是說考績制度對程式設計師是不通的

工作切換有害無益,讓程式設計師只專注一件事情

絕對不能把程式碼重寫(這裡不是指重構)

冰山一角理論,冰山有90%是在水面下,大部分的軟體,那些漂亮的使用者介面通常只占10%的工作,而背後90%的程式設是看不到的,如果再考慮一半時間都在抓蟲,那使用者介面就只剩5%,如果只計算介面中的視覺部分,那客戶真正看到的,只有1%.....,這並不是秘密,真正的秘密是非程式人員根本不知道這件事.....

約耳測試:

你有使用原始碼控制系統嗎?

SVN, CVS, Git, Mercurial...

你能用一個步驟建出所有結果嗎?

準備一個Script檔,只要執行這個腳本,就能一次搞定從最新原始碼快照到自動建立釋出產品的過程

你有進行每日編譯嗎?

提交原始碼到版本控制系統前,一定要編譯並且沒有出現錯誤,因為別人也想下班

你有沒有問題資料庫?

記錄已知的Bug清單,每筆Bug需記錄:

1.重現問題的完整步驟

2.應該看到的結果

3.實際看到的結果

4.被指派的負責人

你會先把問題都修好之後,才寫新的程式嗎?

愈晚修正問題,之後付出的代價成本愈高

你有一份最新的時程表嗎?

程式設計師討厭排時程,但牽扯到業務人員的決策規劃,擁有時程可以強迫自己決定要作哪些功能,並剔除不重要的功能,以避免過度膨脹

1.使用一些PMS工具

2.時程表簡單就好

3.每個功能應該包含多項任務(Task)

4.只有實際要寫該程式的程式設計人員,才能排出該項目的時程

5.要把任務分的很細(以小時為單位)

6.紀錄最初和目前的估計

7.每天更新已消耗時間

8.把休假時間算進去

9.把除錯時間算進去

10.把整合時間算進去

11.把緩衝時間算進去

12.絕對不讓經理縮短估計時間

你有寫規格嗎?例如: GDD TDD

又是一件程式設計師討厭的事情,設計初期的階段還看不出來,愈後期程式碼一多修正的代價就愈高,應貫徹沒有規格就不寫程式的原則

程式設計人員有沒有安靜的工作環境?

需要讓程式設計師進入沈浸狀態(in the zone),因為這時候是最能全神貫注,生產力最高的狀態,所以,要有安靜的環境!!!

你有沒有用市面上最好的工具?

你需要兩個以上的螢幕,以及編譯程式不會讓你抓狂的電腦配備

你有沒有測試人員?

省下測試人員的錢並不是真正的節省,因為你會付出更慘痛的代價,這裡本書第22章有非常有趣的見解
是否在面試時要求面試的對象試寫程式?

你會不請魔術師表演幾招就直接僱用他嗎,當然不會

是否進行過走廊使用性測試?

隨機找幾位使用者試用你的產品,他們可以幫你發現程式中95%應該注意到的使用性問題

約耳認為軟體開發有各種不同的領域,他把它分為五個世界:

熱縮封膜軟體(Shrinkwrap)

簡單來說就是一般常看到的軟體,商業軟體,共享軟體或開放源碼軟體等,特色就是使用者眾多,通常都有替代商品,所以開發者必須把東西做得更簡單易用才行

內部用軟體

這類軟體通常是公司針對特定狀況下能執行就可以了,因此開發起來較容易,但開發者較不容易從中得到成就

嵌入式軟體

這種軟體的特性是被放在硬體裡,且幾乎無法更新,因此品質要求會比平常高出許多

拋棄式軟體

為了產生其他東西而暫時創造的軟體,當達到目的後就不會用到了,例如格式轉檔軟體

遊戲軟體

遊戲開發的經濟學是打擊型,有些遊戲擊出安打,但更多的是三振,要賺大錢必須用很多的遊戲來平衡失敗的損失,和嵌入式軟體類似,版本修正代價大,所以品質要求高

紙上原型製作的重要性

別花太多時間想抽象的架構問題,邊開火邊移動!!!

寫程式並不是量產,也不是工匠技藝,它是一種設計

約耳認為面試人員應該問的問題:

簡介

最近專案的問題

尋找熱情

好的人選會仔細地把事情由各個層面解釋清楚

如果專案是由多個人一齊負責,尋找擔任領導角色的跡象

不可能的問題

程式問題

你有什麼問題嗎?

小員工也能做大事,這本書是關於軟體管理,不過有時候你並沒有那個權力去改變組織,如果你只是圖騰柱底層的一位程式設計人員,約耳給你一些建議:

去做就是了

沒有每日編譯伺服器嗎?自己架一台就對了

利用病毒性行銷的威力

團隊沒有在用版本管理系統?

自己先用吧,等問題發生之後,他們就會來找你了

建立一個卓越圈

找一些支持你的人一起改善吧

讓笨蛋無害

有時候會有天才花兩個星期寫出一點點程式,而且爛到不可思議,這時候你會想花15分鐘把這段程式重新寫過,請忍住,因為這是把這個白癡拖住幾個月的機會,這段期間他不會再造成其他地方傷害了

遠離干擾環境

提升自己的價值

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:www.tushu000.com