

סביבות פיתוח בלינוקס

שחר שמש

יועץ אינטרגצית תוכנה חופשית
חבר ועד עמותת "המקור"



מה בתוכנית

- כלי פיתוח:

- gcc, make, autoconf, autotmake

- gdb/ddd – אין מה להגיד

- ltrace, strace, syscalltrack

- valgrind

- User mode Linux

- סביבות עבודה

- Anjuta, KDevelop, IDLE

- כלים מסביבת Windows

- ctags

- CVS

- כלים נוספים

- Python, Perl, PHP, TCL/Tk

GCC (<http://gcc.gnu.org>)

- מודל הקומפילר הקלאסי
 - חלק קידמי
 - C
 - C++
 - Fortran
 - Java
 - Objective C
 - חלק אחורי
 - Intel
 - Sparc
 - Alpha
 - Strong ARM
 - Palm (Dragon ball)
 - ...

automake (<http://www.gnu.org/software/automake/>)

- התחליף הטקסטואלי לקבצי פרוייקט.
- גמישות גבוהה בייצירת ספריות, תוכנות, ושילוב שלהם.
- דורש שורות בודדות כדי לייצר פרוייקט.



autoconf (http://www.gnu.org/ software/autoconf)

- הכלי האולטימטיבי לייצירת פרויקטים חוצי פלטפורמה.
- מסוגל לבצע בדיקה לפני קומפילציה, ולהתאים את הקוד למערכת המסויימת.



*make (http://www.gnu.org/software/
make/)*

- כלי ניהול תלויות.
- לחם חוקו של כל סביבת פיתוח
– אפילו Visual Studio



ltrace, strace, syscalltrack

- מאפשר מעקב אחרי קריאות המערכת שתוכנית מבצעת.
 - strace – מאפשר לדעת לאילו system calls קוראים.
<http://sf.net/projects/strace>
 - ltrace – עוקב אחרי קריאות ספרייה.
<http://freshmeat.net/projects/ltrace/>
 - syscalltrack – תוכנה כלל מערכתית בסגנון strace.
[/http://syscalltrack.sf.net](http://syscalltrack.sf.net)
 - לצטט את מפתח syscalltrack:

“strace is good for knowing what a process did –
syscalltrack is good for knowing **which** process
did something.”
-
-

valgrind (<http://freshmeat.net/projects/valgrind/>)

- מריץ את התוכנה בתוך מכונה וירטואלית
- עוקב אחרי כל משתנה – מתי מאותחל, מתי משתמשים בו.
- יודע להתריע על שימוש במשתנה לא מאותחל
- – יודע להבדיל בין העתקה (לא שגיאה,) לבין שימוש (שגיאה)
- יודע למצוא חריגות זיכרון
- בסיום הריצה – מחפש בזיכרון התוכנה מצביעים לזכרון שלא שוחרר – אם לא מוצא, מתריע על איבוד זכרון.
- בימים אילו עובדים על תמיכה ב-Wine threading וב-Wine.

User Mode Linux (<http://user-mode-linux.sf.net>)

- מאפשר לבצע kernel modules כתוכנה רגילה.
 - הרצת debugger רגיל, breakpoints, וכו'
- מהווה thin virtual machine.



סביבות עבודה

- הכלים המסורתיים
 - vi/vim
 - GNU emacs/xemacs
 - ed
- הכלים המודרניים
 - Anjuta (<http://anjuta.sf.net>)
 - KDevelop (<http://www.kdevelop.org>)
 - IDLE (<http://www.python.org/idle/>)
 - Eclipse (<http://www.eclipse.org>)
- Wine
 - VisualStudio

ctags

- משפחה של כלים "מתחרים".
 - ctags
 - etags
 - LXR
 - ViewCVS
- מבצעים cross reference של כל המזהים בתוכנית
- באינטגרציה מלאה לכלים הסביבתיים

CVS, arch, SubVersion, Rational, BitKeeper

- ניהול תצורה.
- הכרחי לכל פרוייקט פיתוח, קטן ככל שיהיה.



כלים נוספים

- קיימים כלים רבים היכולים לעזור עם מטלות.
- כלי ה"scripting" הינם קיצור דרך חשוב.
- הרבה מטלות שמתרגמות לתוכנות C של 300 שורות יכולות להתבצע ע"י תוכנות Perl של 10 שורות.

לא יכוסה בהרצאה זו

- gprof – profiling
- sed, awk
- shell scripting



מספר מילים על התקנות

- התקנה לא גרפית – הכרח
- שימוש במנגנונים של ההפצה – רצוי
 - RPM –
 - DEB –
- אסור שהתקנה של תוכנה תכיל רכיבים מלבד התוכנה



למי שלא הספיק לרשום

את השקפים ניתן להוריד מ-

<http://shemesh.biz/lectures.html>
