Technical Track

고급 디버깅 기법

(박재호)

CWEB으로 즐기는 문학적 프로그래밍

(남수진)

Programming with Emacs

(신성국)

오픈소스로 모바일 프로그래밍 하기

(용영환)

KLDP 10주년 세미나(기술 트랙)
고급 디버깅 기법

2006년 9월 17일

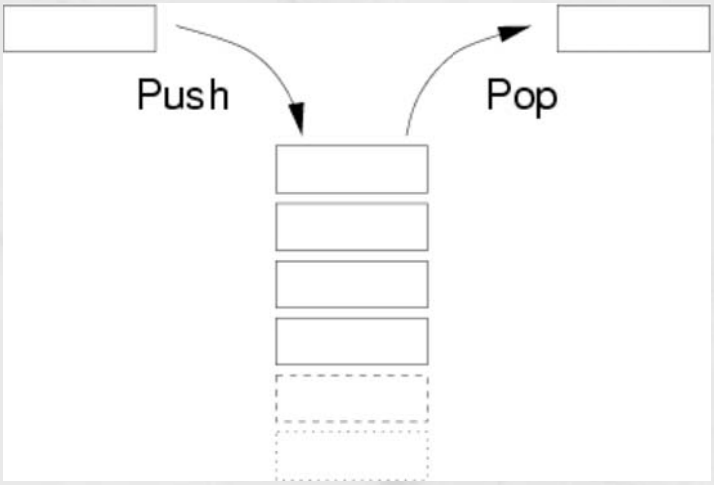
박재호

http://jhrogue.blogspot.com

목차

스택이란?

- 컴퓨터 과학 부문에서 가장 많이 사용하는 자료 구조
  - 후입선출(LIFO, Last In First Out)



함수와 프로시저 호출(1)

- 용어 정리
  - 서브루틴: 고수준 루틴에서 호출하는 루틴
    - 참고: 주로 어셈블리어에서 사용한다.
  - 함수: 이름을 통해 값을 반환하는 서브루틴
  - 프로시저: 값을 반환하지 않는 함수
  - 예
    - 함수: c = max(a, b);
    - 프로시저: printf(“hello, world!\n”);

함수와 프로시저 호출(2)

- 윈도우에서 함수 호출 관행
  - _cdecl: 마지막 인자부터 스택에 저장, 함수를 호출한 쪽에서 스택 정리
    - 일반적인 C API
  - _stdcall: 마지막 인자부터 스택에 저장, 함수가 호출된 쪽에서 스택 정리
    - 윈도우 C API
  - _fastcall:레지스터에 인수 두 개 저장, 함수가 호출된 쪽에서 스택 정리

함수와 프로시저 호출(3)

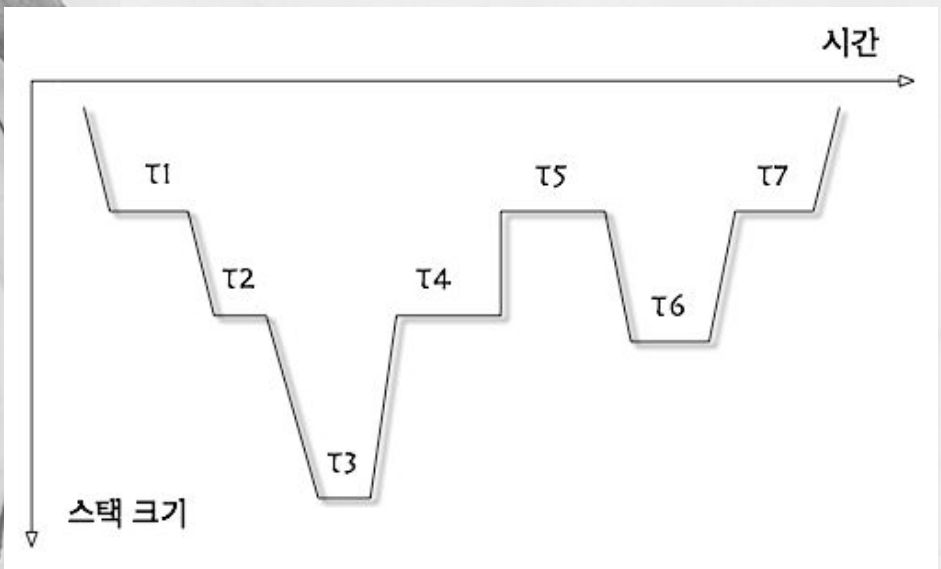
- C 함수 특성
  - 파스칼과 같은 엄격한 언어와는 달리 C에서는 함수와 프로시저의 구분이 없다.
  - 포트란과 같은 언어와는 달리 C에서 인수 전달은 call-by-value이며, call-by-reference를 사용하려면 반드시 포인터를 써야 한다. --> 스택 변수(자동 변수)의 한계점
    - 참고) 재귀 호출

함수와 프로시저 호출(4)

- 프로그램 층(hierarchy)
  - C 프로그램은 main에서 시작해서 함수로 제어 흐름을 전파한다. --> 호출 층(calling hierarchy)
  - 가장 말단 층은 라이브러리 함수나 시스템 호출이 된다.
    - 참고) 추상화 개념을 적용한 모듈화와 구분
  - 호출 층에 놓인 각 함수마다 각자 자기 지역 변수를 독립적으로 유지할 필요가 있다. --> 호출 스택 프레임 개념이 필요

함수와 프로시저 호출(5)

- 스택 프레임 변화




```
main() {
  ..        /* t1 */
  func1();
  ..        /* t5 */
  func2();
  ..        /* t7 */
} /* end of main */
func1() {
  ..        /* t2 */
  func2();
  ..        /* t4 */
} /* end of func1 */
func2() {
  ..        /* t3, t6 */
} /* end of func2 */
```

ABI(1)

- ABI란?
  - Application Binary Interface
  - 응용 프로그램과 운영체제, 응용 프로그램과 라이브러리 사이에 필요한 저 수준 인터페이스를 정의
    - 참고: ABI는 목적 파일과 관련이 있으므로 원시 코드 컴파일 과정에 개입하는 API(Application Programing Interface)와는 다르다.

ABI(2)

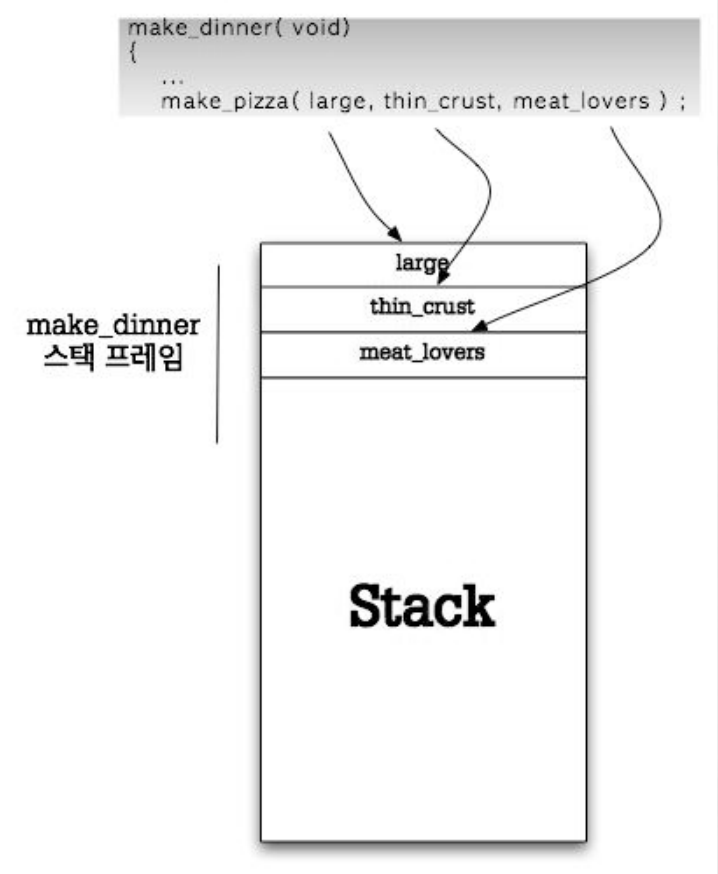
- ABI는
  - 호출 규약 명세를 정의한다:
    - 함수 인수 전달 방법과 반환값 전달 방법을 포함한다.
  - 아키텍처와 운영체제마다 다르다:
  - 다양한 컴파일러가 만들어낸 여러 목적 파일을 함께 링크시킬 수 있는 이유이다.
  - 최근에는 C++ 이름 규약(mangle)까지도 정의하고 있다.

ABI(3)

- x86 ABI
  - http://www.caldera.com/developers/devspecs/abi386-4.pdf
  - x86 ABI는 비교적 단순하다.
    - 규칙 1: 모든 인수를 스택에 푸시한다
    - 규칙 2: 반환값은 %eax 레지스터에 넣는다.
- x86_64 ABI
  - http://www.x86-64.org/documentation/abi.pdf
  - x86_64 ABI는 성능 개선을 위해 조금 복잡한 규칙을 사용한다.
    - 규칙 1: 첫 6개 인수를 왼쪽에서 오른쪽으로 %rdi, %rsi, %rdx, %rcx, %r8, %r9에 넣고 나머지를 스택에 푸시
    - 규칙 2: 반환값은 %eax 레지스터에 넣는다.

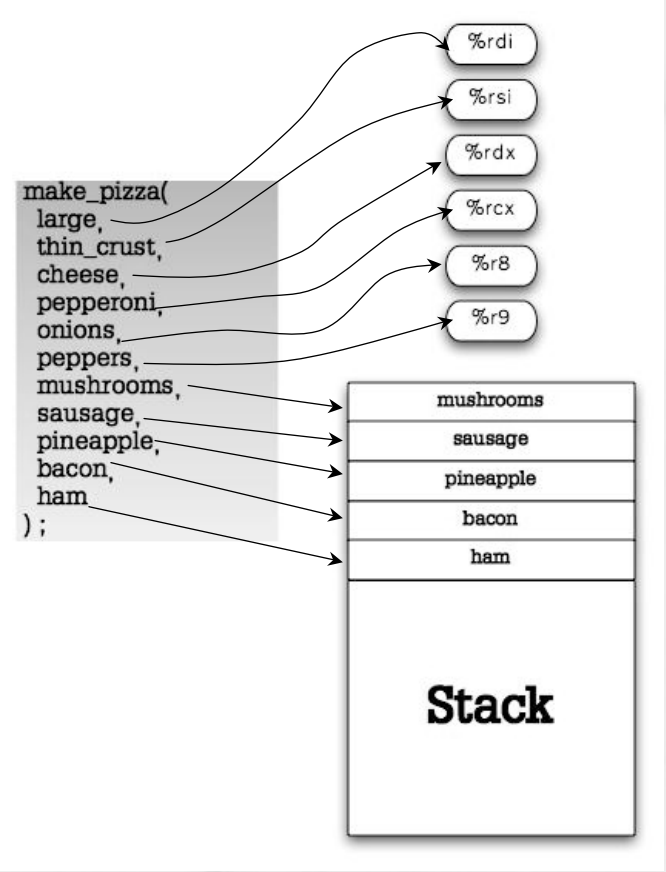
ABI(4)

x86 ABI



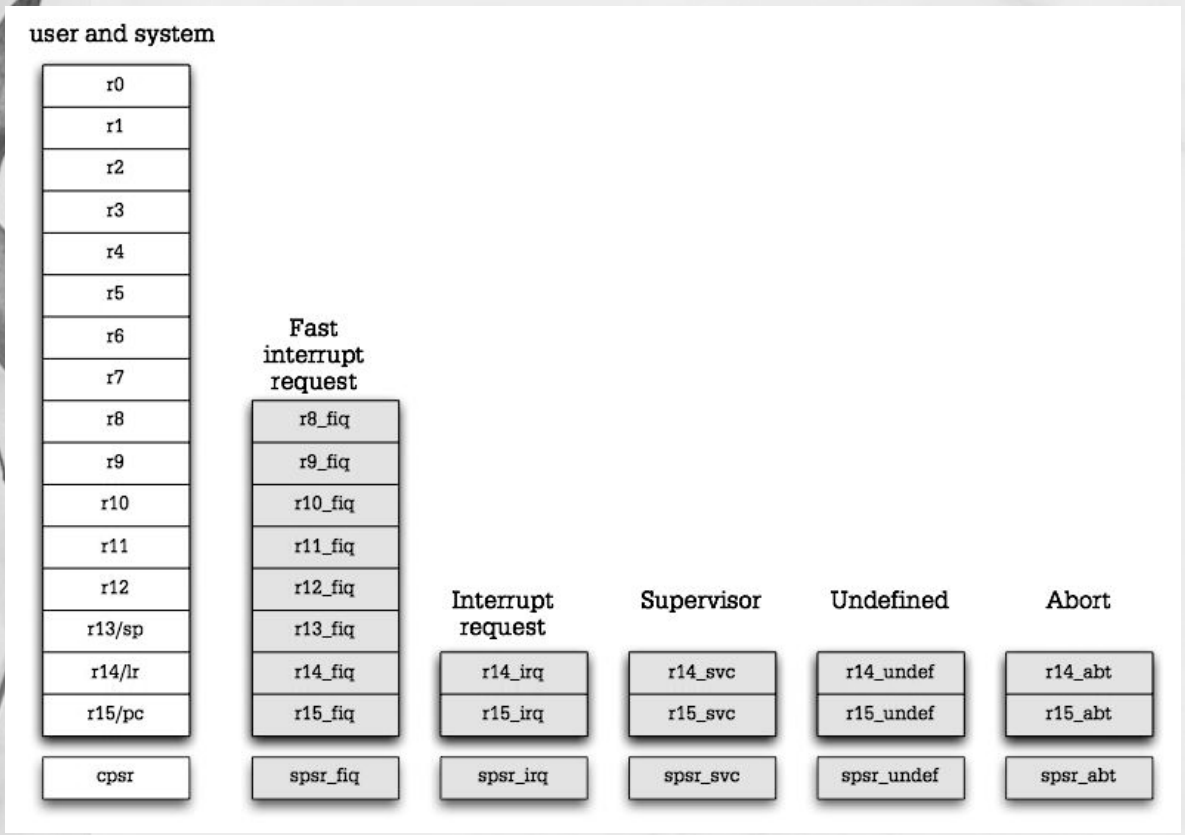
ABI(5)

x86_64 ABI

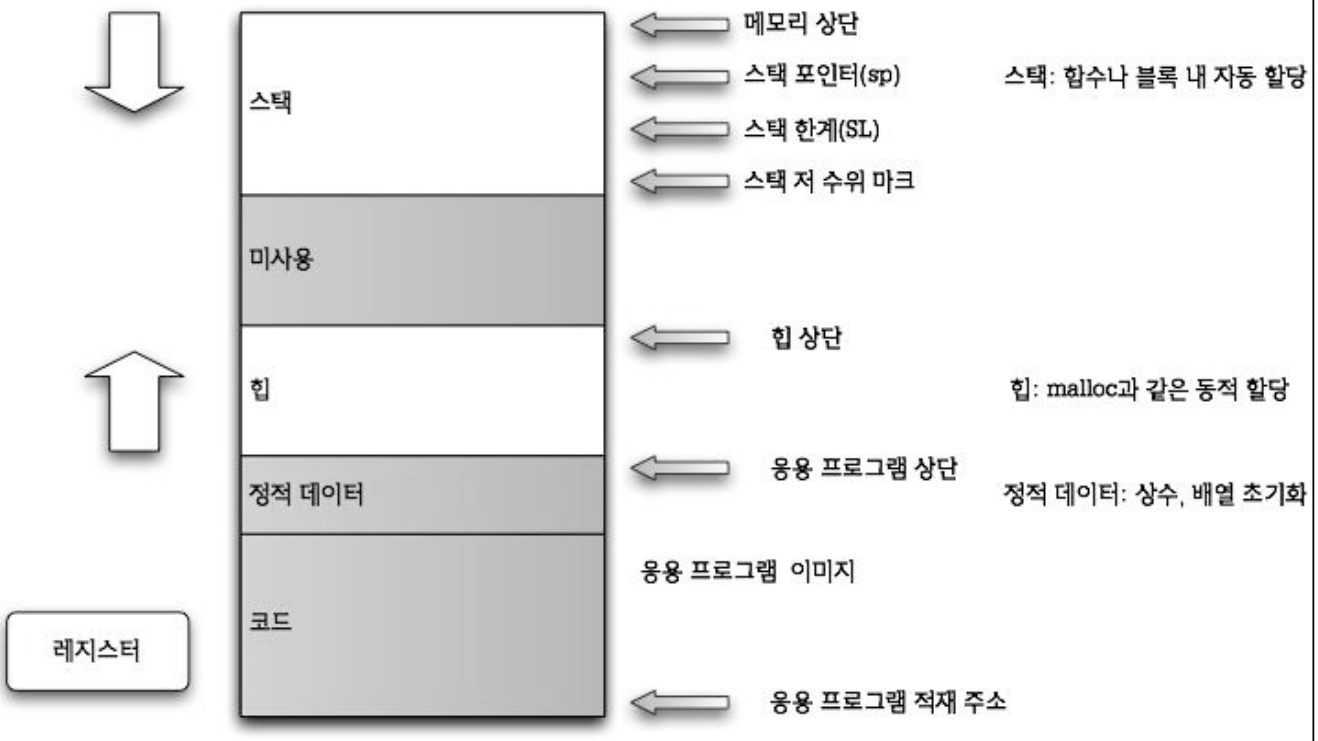


ABI(6)

- APCS
  - ARM Procedure Standard Call
    - 규칙 1: a1 ~ a4(caller-saved): 함수로 넘기는 인수를 담음, v1 ~ v5(callee-saved): 함수 내부에서 임의로 사용
    - 규칙 2: 부동 소수점일 경우 해당 레지스터에서 부동소수점 레지스터로 복사한다(하드웨어 지원).
    - 규칙 3: 인수가 네개 이하일 경우 레지스터만을 사용하며, 네개를 초과하는 인수는 스택에 역순으로 들어간다.
    - 규칙 4: 가변 길이 인수(va_arg) 역시 스택으로 처리한다.
    - 규칙 5: 워드를 초과하는 인수는 레지스터와 스택으로 나뉘어져 전달한다.
    - 규칙 6: 정수와 같은 결과값은 a1을 통해 반환하며, 복잡한 결과값은 a1에 결과값을 담은 주소를 넣고 반환한다.



레지스터	APCS 이름	APCS 임무
0	a1	인수1/정수 결과/임시 레지스터
1	a2	인수2/임시 레지스터
2	a3	인수3/임시 레지스터
3	a4	인수4/임시 레지스터
4	v1	레지스터 변수 1
5	v2	레지스터 변수 2
6	v3	레지스터 변수 3
7	v4	레지스터 변수 4
8	v5	레지스터 변수 5
9	sb/v6	정적 베이스/레지스터 변수 6
10	sl/v7	스택 한계/레지스터 변수 7
11	fp	프레임 포인터
12	ip	임시 레지스터
13	sp	현재 스택 프레임 아래 끝
14	lr	링크 주소/임시 레지스터
15	pc	프로그램 카운터


스택
미사용
힙
정적 데이터
코드
레지스터
메모리 상단
스택 포인터(sp)
스택 한계(SL)
스택 저 수위 마크
힙 상단
응용 프로그램 상단
응용 프로그램 적재 주소
스택: 함수나 블록 내 자동 할당
힙: malloc과 같은 동적 할당
정적 데이터: 상수, 배열 초기화
응용 프로그램 이미지

스택 프레임과 함수 호출 규약(1)

- GNU 스택 추적 명령
  - bt: backtrace(역추적)
  - bt full: backtrace full(지역 변수도표시)
  - frame [숫자]: 프레임 보기/이동
  - where: 현재 프레임 위치
  - where full: 현재 프레임 위치(지역 변수도 표시)
  - up/down [숫자]: 프레임 오르내리기
  - info frame: 프레임 세부 내역 출력

스택 프레임과 함수 호출 규약(2)

```
#include <stdio.h>


void bar(int c)
{
        int d;
        printf("bar: add(c) = %x₩n", &c);
        printf("bar: add(d) = %x₩n", &d);
        d = c;
        printf("d is %d₩n", d);
}
void foo(int a, int b)
{
        int c;
        printf("foo: add(a) = %x₩n", &a);
        printf("foo: add(b) = %x₩n", &b);
        printf("foo: add(c) = %x₩n", &c);
        c = a + b;
        printf("c is %d₩n", c);
        bar(c);
}
```

```
int main(int argc, char **argv)
{
        int a, b;
        printf("main: add(argc) = %x₩n", &argc);
        printf("main: add(argv) = %x₩n", &argv);
        printf("main: add(a) = %x₩n", &a);
        printf("main: add(b) = %x₩n", &b);
        a = 1;
        b = 2;
        foo(a, b);
}
```

스택 프레임과 함수 호출 규약(3)

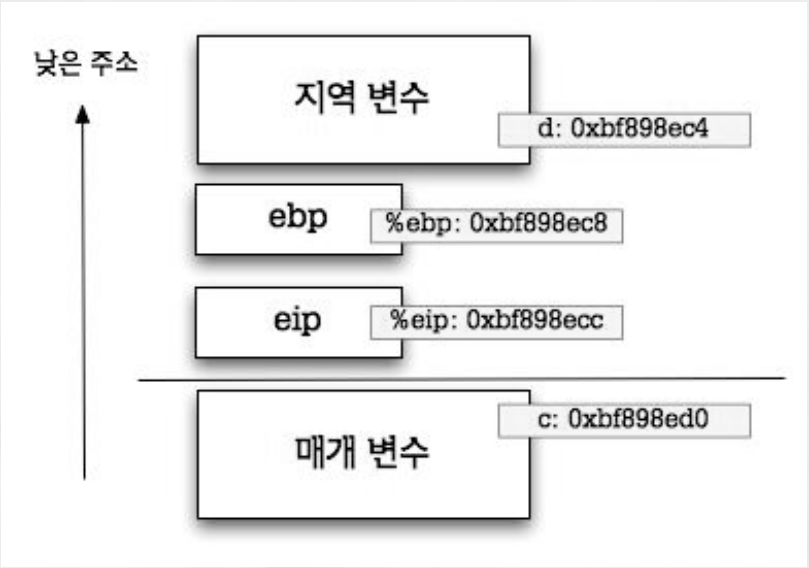
```
(gdb) b bar
Breakpoint 1 at 0x8048362: file stack.c, line 7.
(gdb) r
Starting program: /home/jaypark/tmp/stack
Reading symbols from shared object read from target memory...done.
Loaded system supplied DSO at 0x3b8000
main: add(argc) = bf898f10
main: add(argv) = bf898f14
main: add(a) = bf898f04
main: add(b) = bf898f00
foo: add(a) = bf898ef0
foo: add(b) = bf898ef4
foo: add(c) = bf898ee4
c is 3
Breakpoint 1, bar (c=3) at stack.c:7
7               printf("bar: add(c) = %x₩n", &c);
(gdb) n
bar: add(c) = bf898ed0
8               printf("bar: add(d) = %x₩n", &d);
(gdb) n
bar: add(d) = bf898ec4
10              d = c;
(gdb) n
12              printf("d is %d₩n", d);
```


스택 프레임과 함수 호출 규약(4)

```
(gdb) info frame
Stack level 0, frame at 0xbf898ed0:
 eip = 0x8048390 in bar (stack.c:12); saved eip 0x804840e
 called by frame at 0xbf898ef0
 source language c.
 Arglist at 0xbf898ec8, args: c=3
 Locals at 0xbf898ec8, Previous frame's sp is 0xbf898ed0
 Saved registers:
  ebp at 0xbf898ec8, eip at 0xbf898ecc
(gdb) up
#1  0x0804840e in foo (a=1, b=2) at stack.c:26
26              bar(c);
(gdb) info frame
Stack level 1, frame at 0xbf898ef0:
 eip = 0x804840e in foo (stack.c:26); saved eip 0x804848f
 called by frame at 0xbf898f10, caller of frame at 0xbf898ed0
 source language c.
 Arglist at 0xbf898ee8, args: a=1, b=2
 Locals at 0xbf898ee8, Previous frame's sp is 0xbf898ef0
 Saved registers:
  ebp at 0xbf898ee8, eip at 0xbf898eec
```

스택 프레임과 함수 호출 규약(5)

함수 bar




```
main:
        pushl   %ebp # 베이스 포인터 저장
        movl    %esp, %ebp #스택 포인터 → 베이스 포
        subl    $8, %esp # a, b 자동 변수 영역 할당
        andl    $-16, %esp
        movl    $0, %eax
        subl    %eax, %esp
        subl    $8, %esp
        leal    8(%ebp), %eax
        pushl   %eax
        pushl   $.LC7
        call    printf
        addl    $16, %esp
        subl    $8, %esp
        leal    12(%ebp), %eax
        pushl   %eax
        pushl   $.LC8
        call    printf
        addl    $16, %esp
        subl    $8, %esp
        leal    -4(%ebp), %eax
        pushl   %eax
        pushl   $.LC9
        call    printf
        addl    $16, %esp
```

```
        subl    $8, %esp
        leal    -8(%ebp), %eax
        pushl   %eax
        pushl   $.LC10
        call    printf
        addl    $16, %esp
        movl    $1, -4(%ebp)
        movl    $2, -8(%ebp)
        subl    $8, %esp
        pushl   -8(%ebp)
        pushl   -4(%ebp)
        call    foo
        addl    $16, %esp # 스택 프레임 제거
        leave
        ret
```

```
movl %ebp, %esp
popl %ebp
```

```
int main(int argc, char **argv)
{
        int a, b;
        printf("main: add(argc) = %x₩n", &argc);
        printf("main: add(argv) = %x₩n", &argv);
        printf("main: add(a) = %x₩n", &a);
        printf("main: add(b) = %x₩n", &b);
        a = 1;
        b = 2;
        foo(a, b);
}
```

참고: 반환값이 있다면 %eax에…

GNU 확장을 사용한 역추적 기법

- 확장 API
  - backtrace: 역추적을 위한 핵심 API
  - backtrace_symbols: 심볼 주소(이름) 획득
- 컴파일러 플래그
  - -rdynamic: 심볼 주소를 이름으로 사상
    - -Wl,-export-dynamic
    - -Wl,-E
    - cf) 커널의 ksymoops

GNU 확장을 사용한 역추적 기법 (1)

```
#include <stdio.h>
#include <execinfo.h>

void print_gnu_backtrace( void )
{
        void *frame_addrs[16];
        char **frame_strings;
        size_t backtrace_size;
        int i;
        backtrace_size = backtrace( frame_addrs, 16 );
        frame_strings = backtrace_symbols( frame_addrs, backtrace_size );
        for ( i = 0; i < backtrace_size; i++ ) {
                printf("%d: [0x%x] %s₩n", i, frame_addrs[i], frame_strings[i]);
        }
        free( frame_strings );
}
```

GNU 확장을 사용한 역추적 기법 (2)

```
int foo( void )
{
        print_gnu_backtrace();
        return 0;
}
int bar( void )
{
        foo();
        return 0;
}
int boo( void )
{
        bar();
        return 0;
}
int baz( void )
{
        boo();
        return 0;
}
```

```
int main( void )
{
        baz();
        return 0;
}
```

```
[jaypark@compile tmp]$ ./gnu_backtrace
0: [0x804869c] ./gnu_backtrace(print_gnu_backtrace+0x14) [0x804869c]
1: [0x8048711] ./gnu_backtrace(foo+0xb) [0x8048711]
2: [0x8048723] ./gnu_backtrace(bar+0xb) [0x8048723]
3: [0x8048735] ./gnu_backtrace(boo+0xb) [0x8048735]
4: [0x8048747] ./gnu_backtrace(baz+0xb) [0x8048747]
5: [0x8048763] ./gnu_backtrace(main+0x15) [0x8048763]
6: [0x9dfde6] /lib/libc.so.6(__libc_start_main+0xc6) [0x9dfde6]
7: [0x80485fd] ./gnu_backtrace [0x80485fd]
```


libdl을 사용한 역추적 기법(1)

- 확장 API
  - dladdr: 특정 주소에 있는 심볼 정보를 얻는다
    - cf) GNU_SOURCE는 dladdr을 사용하기 위해 원시 코드 가장 처음에 정의한다.
- 컴파일러 플래그
  - -rdynamic: GNU 확장 참조
  - -ldl: dl API를 사용하기 위해 필수적으로 링크시켜야하는 라이브러리 지정

libdl을 사용한 역추적 기법(2)

```
#define _GNU_SOURCE
#include <stdio.h>
#include <dlfcn.h>
void **getEBP( int dummy )
{
        void **ebp = (void **)&dummy - 2;
        //printf( "₩n" );
        return( ebp );
}
void print_walk_backtrace( void )
{
        int dummy;
        int frame = 0;
        Dl_info dlip;
        void **ebp = getEBP( dummy );
        void **ret = NULL;
        printf( "Stack backtrace:₩n");
        while( *ebp ) {
                ret = ebp + 1;
                if (dladdr( *ret, &dlip ) == 0)
                        break;
                printf( " Frame %d: [ebp=0x%08x] [ret=0x%08x] %s₩n", frame++,
*ebp, *ret, dlip.dli_sname );
                ebp = (void**)(*ebp);
        }
}
```

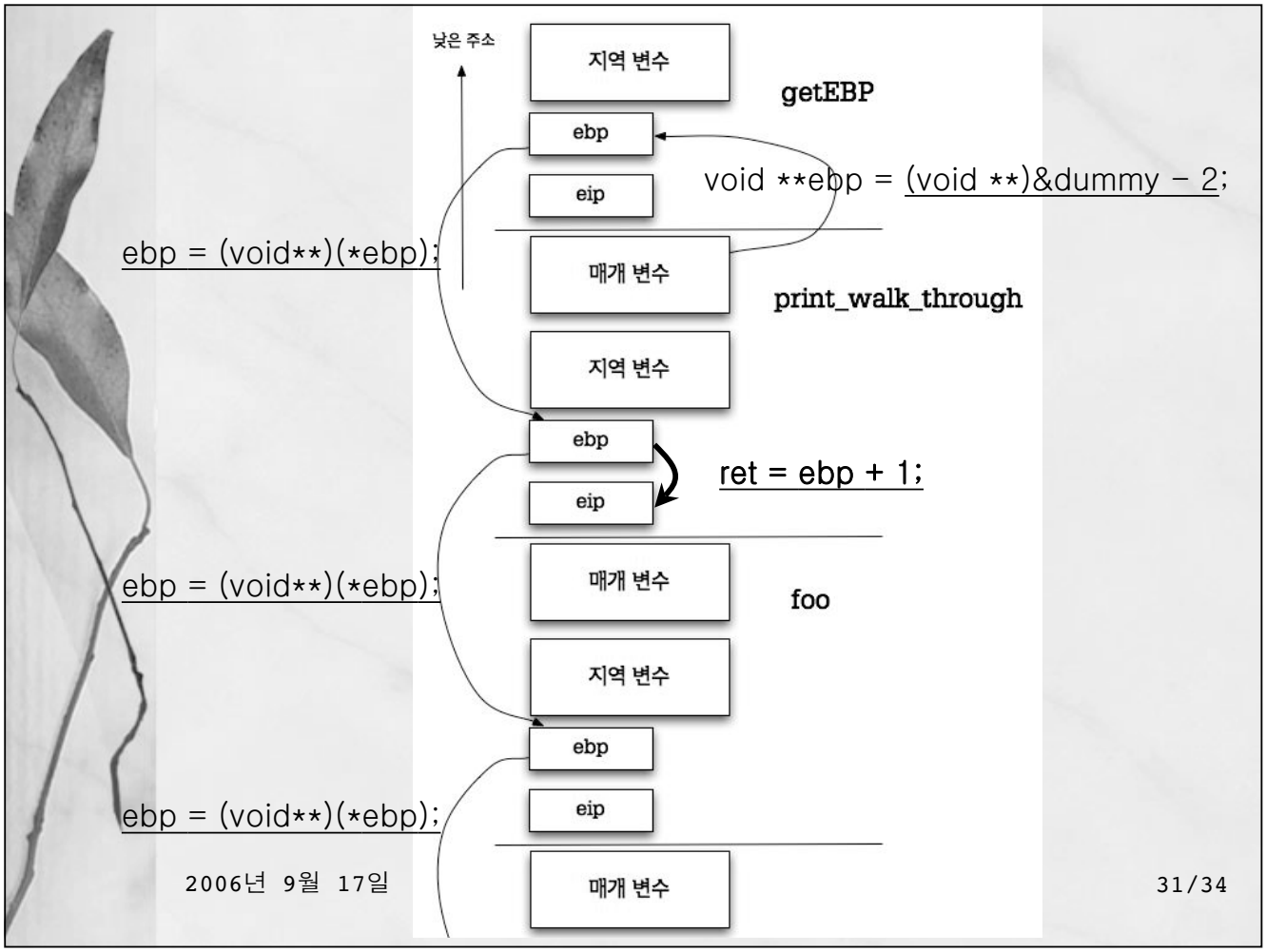
```
typedef struct {
  __const char *dli_fname;
  void *dli_fbase;
  __const char *dli_sname;
  void *dli_saddr;
} Dl_info;
```


libdl을 사용한 역추적 기법(3)

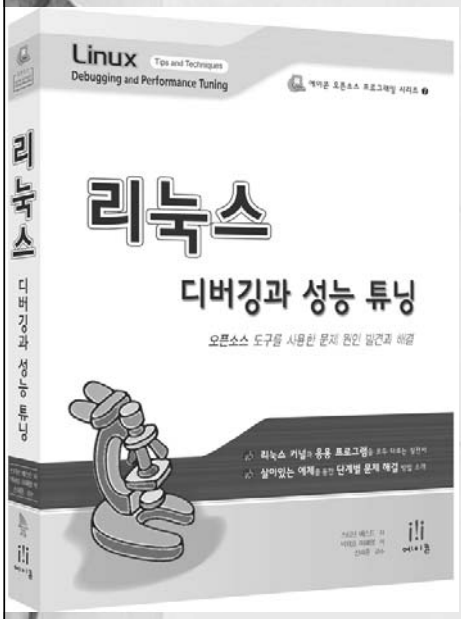
```
int foo( void )
{
        print_walk_backtrace();
        return 0;
}
int bar( void )
{
        foo();
        return 0;
}
int boo( void )
{
        bar();
        return 0;
}
int baz( void )
{
        boo();
        return 0;
}
```

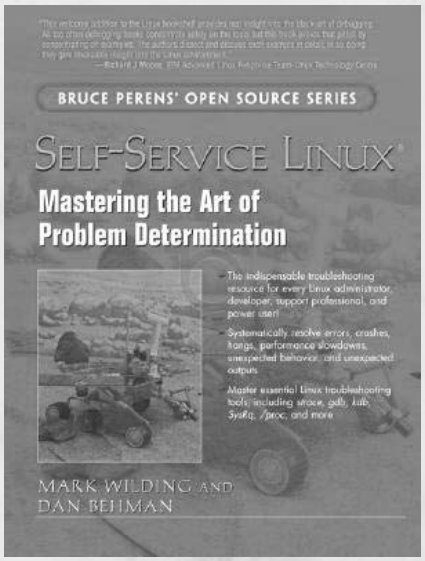
```
int main( void )
{
        baz();
        return 0;
}
```

```
[jaypark@compile tmp]$ ./bt
Stack backtrace:
 Frame 0: [ebp=0xbfa62ca8] [ret=0x080486a9] print_walk_backtrace
 Frame 1: [ebp=0xbfa62cb8] [ret=0x080486ef] foo
 Frame 2: [ebp=0xbfa62cc8] [ret=0x08048701] bar
 Frame 3: [ebp=0xbfa62cd8] [ret=0x08048713] boo
 Frame 4: [ebp=0xbfa62ce8] [ret=0x08048725] baz
 Frame 5: [ebp=0xbfa62cf8] [ret=0x08048741] main
 Frame 6: [ebp=0xbfa62d58] [ret=0x009dfde6] __libc_start_main
[jaypark@compile tmp]$
```


낮은 주소
지역 변수
getEBP
ebp
eip
void **ebp = (void **)&dummy – 2;
ebp = (void**)(*ebp);
매개 변수
print_walk_through
지역 변수
ebp
ret = ebp + 1;
eip
ebp = (void**)(*ebp);
매개 변수
foo
지역 변수
ebp
eip
ebp = (void**)(*ebp);
매개 변수

참고문헌


Linux
Tips and Techniques
Debugging and Performance Tuning
리눅스
디버깅과 성능 튜닝
오픈소스 도구를 사용한 문제 원인 발견과 해결


BRUCE PERENS' OPEN SOURCE SERIES
SELF-SERVICE LINUX
Mastering the Art of Problem Determination
MARK WILDING AND DAN BEHMAN


ARM System Developer's Guide
Designing and Optimizing System Software
SciTech

QnA

CWEB으로 즐기는 문학적 프로그래밍

TEX과 C 프로그래밍 언어와의 만남

남수진

한글 TEX 사용자 그룹

- 이 저작물은 상업적 목적으로 사용하실 수 없습니다.
- 이 저작물은 개인적 목적으로 수정, 배포하시는 것은 괜찮으나, 그럴 경우 반드시 출처를 밝혀야 합니다.

Part I

컴퓨터 프로그래밍

컴퓨터 프로그램은 읽기 쉬워야 한다

- 소프트웨어 개발하는데 드는 노력 중 프로그램 작성에 들어가는 노력은 고작 10%, 나머지 90%는 이미 작성된 코드의 유지보수, 디버깅, 문서화 작업에 들어간다.
- 타인이 작성한 컴퓨터 프로그램은 읽기가 힘들다. 심지어 자신이 작성한 프로그램도 시간이 지나면 읽기 어려워진다.
- 우리는 다른 사람이 작성한 소스 코드를 보고 프로그래밍을 배운다.
- 프로그래밍 언어도 사람들 사이에 사용되는 언어이기도 하다
- “모든 프로그램은 먼저 사람이 읽을 수 있어야 하고, 두번째로 기계가 읽을 수 있어야 한다.”

Computer Programming as an Art

- 1974년 Knuth의 ACM Turing Award 수상기념 강연 제목
- 좋은 프로그램은 쉽게 읽고 이해할 수 있는 것어야 한다.
- 프로그램을 효율성만 강조하여서 코드가 쓸데없이 복잡해져서 읽기 어렵게 되면, 디버그와 유지보수가 힘들어져서 결국 그 소프트웨어의 효율은 전체적으로 낮아진다.
- "Premature optimization is the root of all evil (or at least most of it) in programming."
- 소설가나 시인들이 독자들을 위해 문학 작품을 창작해 내듯이, 프로그램도 그것을 읽는 사람들을 위한 문학 작품(works of literature)이라 할 수 있다.

컴퓨터 프로그램은 실행하기 위해서 작성되는 것이 아니라, 읽기 위해서 작성되는 것이다.

Part II

문학적 프로그래밍

문학적 프로그래밍(Literate Programming)

- 프로그램을 작성할 때, 기본적인 마음가짐을 바꾸자.
  컴파일러가 쉽게 알아들을 수 있도록 프로그래밍 하기 보다는 사람이 쉽게 이해할 수 있도록 프로그래밍 하자. 결국은 이것이 프로그램의 효율을 높이는 것이다!
- 컴퓨터 프로그래밍의 정의 프로그래밍이란 컴퓨터가 해야 할 일들을 논리적이고 순차적으로 나열하는 것이 아니라, 컴퓨터가 해야할 일을 사람들에게 논리적이고 재미있게 설명하는 작업이다.
- 프로그래밍 언어의 문법과 규칙은 컴파일러에 적합한 것이어서 우리 사람들의 정서에는 맞지 않는다.
  따라서 프로그램은 사람이 작성하고 읽기 편하도록 사람에게 적합한 구조로 재배열되어야 한다.

문학적 프로그래밍(Literate Programming)

문학적 프로그래밍이란 사람들이 읽기 편하도록 프로그램을 재구성하여 자연스런 설명을 곁들인 프로그램 코드를 작성하는 것이다.

> I believe that the time is ripe for significantly better documentation of programs, and that we can best achieve this by considering programs to be *works of literature*. Hence, my title: "Literate Programming."
>
> Donald Knuth, Literate Programming

도대체 어떻게?

어떻게 하면 프로그램을 사람이 읽기 편하게 재배열 할 수 있을까?

프로그램의 구조적 특성을 이용하자.

프로그램의 구조는 "WEB" 이다

- 프로그램에는 수 많은 프로시져, 함수, 루틴들 있고 그 모든 것들은 프로그램의 논리적인 흐름에 따라 서로 서로를 호출하거나 이용한다.
- 소프트웨어 프로그램의 구조는 많은 작은 부분들이 내부적으로 서로 긴밀히 연결되어 있는 "웹(web)" 이라고 할 수 있다.
- 아무리 크고 복잡한 프로그램이라 할지라도, 그러한 프로그램을 구성하는 각각의 작은 부분들을 설명하고, 그 부분들이 서로 어떻게 연결되고, 어떠한 관련이 있는지를 설명하는 것이, 전체 프로그램을 보다 쉽게 이해할 수 있는 방법일 것이다.

프로그램의 구조는 "WEB" 이다

- 프로그램의 WEB의 특성을 이용해서 작성한 프로그램을 "WEB 프로그램" 이라고 부른다.
- 웹프로그램을 구성하는 작은 부분들을 "섹션(section)" 이라고 부르고, 그 섹션은 §1, §2 처럼 순차적으로 번호가 붙어 있다.
- 각각의 섹션은 그 자체로 독립적인 의미를 가질 수 있을 정도의 십여줄 내외의 소스 코드 크기가 적당하다.
- 커다란 소프트웨어 프로그램은 태생적으로 복잡할 수 밖에 없는 것이어서 그 프로그램의 미묘한 부분깐지 단번에 이해할 수 있는 방법은 존재하지 않는다.
- 하지만, 그 프로그램이 웹프로그램이라면, §1 부터 시작하여, 한 번에 섹션 하나씩 읽고 이해해 나감으로해서 그 프로그램의 전체적인 구조를 보다 쉽게 이해할 수 있다.

Part III

CWEB 프로그래밍

CWEB

- 가장 많이 사용되는 대표적인 문학적 프로그래밍 시스템.
- C 언어로 작성하는 문학적 프로그래밍 언어, 도구, 시스템.
- C 프로그램이란 사실은 섹션들이 유기적으로 결합 되어 있는 CWEB 프로그램에서 C 코드 만을 뽑아내어서 컴파일러가 이해하기 쉬운 구조로 재배열한 프로그램이다.
- CWEB 프로그램은 C 프로그램을 섹션으로 나눈 다음, 사람들이 이해하기 쉬운 구조로 설명을 곁들여 재배열한 프로그램이다.
- CWEB 프로그램과 C 프로그램은 근본적으로 동일한 것이고, 단지 그 배열 순서만이 틀릴 뿐이다.

CWEB의 섹션

- 웹프로그램의 섹션은 그 섹션의 목적과 하는 일을 설명하는 간단한 설명으로 시작하는데, 그 설명은 주어진 형식이 없이 자연스러운 인간의 언어로된 설명문이다.
- 그 자연스러운 설명 다음에는 그 설명에 해당하는 논리적이고 형식적인 프로그램 코드가 나온다.
- 그리고, 설명과 프로그램 코드 사이에는 짧고 간단한 하나 이상의 매크로 정의가 올 수 있다.
- 섹션의 구조
  - 형식이 없는 한글로 된 자유로운 설명: TEX 코드
  - 매크로 정의들: `#define`문
  - 논리적이고 정형화된 프로그램 코드: C 코드
- TEX으로 만들어지는 자유로운 설명과 그에 C 프로그램 코드를 통한 두 번의 설명으로 섹션을 보다 잘 이해할 수 있다.

CWEB의 문서화: TEX

- CWEB 프로그램에서 문서화는 TEX이 담당한다.
- TEX은 수식이 많이 사용되는 아름다운 책을 만들기 위한 시스템이다.

$$\prod_{j\geq 0}\left(\sum_{k\geq 0} a_{jk}z^k\right) = \sum_{n\geq 0} z^n \left(\sum_{\substack{k_0,k_1,\ldots\geq 0\\ k_0+k_1+\cdots=n}} a_{0k_0}a_{1k_1}\ldots\right).$$

- TEX은 워드프로세서가 아니라 조판 시스템으로 컴퓨터 프로그래밍 언어이기도 하다.
- TEX으로 문서를 작성한다고 하면, 십중팔구 LATEX을 뜻한다.

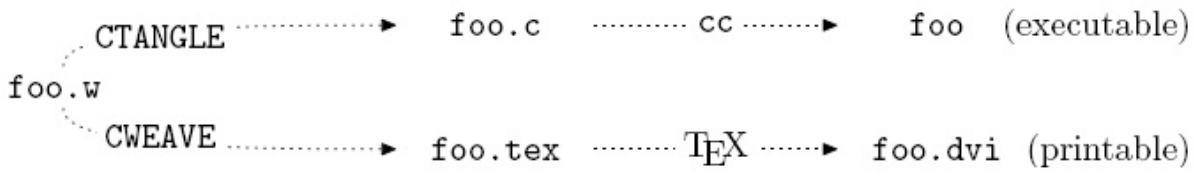
CWEB의 문서화

CWEB과 같은 문학적 프로그램에 워드 프로세서가 아닌 TEX과 같은 조판 언어(문서화 언어)가 필요한 이유는?

문학적 프로그램에서 문서화를 맡는 부분도 프로그래밍과 동일하게 소스코드 작성, 컴파일의 과정을 거쳐야 하기 때문이고, 이렇게 함으로써 프로그램과 문서를 하나의 파일에서 작성 할 수 있다.

CWEB 시스템 = CWEAVE + CTANGLE

문학적 프로그래밍이란 사람들이 읽기 편하도록 자연스런 설명과 프로그램 코드를 하나의 소스 파일에서 동시에 작성하는 것이다.



- CWEAVE: 웹파일로 부터 TEX 파일을 만들어 낸다. 소스 코드와 문서를 보기 좋게 잘 조합하여 직물을 짜내듯이(weave) 보기 좋은 문서를 만든다.
- CTANGLE: 웹파일의 각 섹션들에서 C 코드만 뽑아내어 C 파일을 만들어 낸다. 사람들의 기준에서 잘 정돈된(untangle) 코드를 뒤섞어서(tangle) 컴파일러가 이해 할 수 있는 코드를 생성한다.

CWEB의 장단점

장점

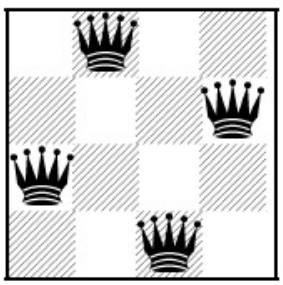
- 프로그래밍 작성하기와 읽기가 쉬워진다.
- 프로그램 소스 코드를 신경쓰지 않아도 된다.
- Emacs 내에서 gdb를 이용한 디버깅으로 쉽고 빠른 디버깅이 가능하다.

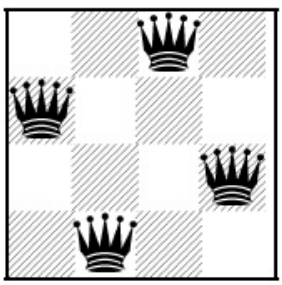
단점

- 프로그래밍 언어는 C, C++, Java만 사용할 수 있다.
- 고품질의 문서 생성해내기 때문에, 관련 명령어(control code)가 많다.
- plain TeX을 이용한다.

CWEB 프로그래밍 시연

N queens Problem





CWEB의 한글화

- TEX의 한글화로 가능해졌다.
- CWEB시스템의 소스 컴파일
- 한글 사용(UTF-8)을 위해서 `common.w` 소스를 수정해야 한다.
- 아래와 같은 내용을 포함하는 `comm-utf8.ch` 작성

```
@x
char *buffer_end=buffer+buf_size-2; /* end of |buffer| */
@y
char *buffer_end=buffer+long_buf_size-2; /* end of |buffer| */
@z
```

- CWEB의 문서화는 TEX이 담당한다.
  - `ucsplain.tex`: TEX에서 한글 사용
  - `cwbucsol.tex`: PDF 파일에서 한글 책갈피
  - `hangulcweb.tex`: CWEB의 매크로 파일인 `cwebmac.tex`에서 한글화가 필요한 매크로를 재정의

그밖의 문학적 프로그래밍 도구

- WEB: Pascal 언어를 이용. 최초의 웹프로그래밍 시스템
- noweb: LATEX + 모든 프로그래밍 언어. 매우 간단.
- Rambutan: TEX + Java 언어, CWEB와 동일.
- FWEB: Fortran 언어
- xmlLP
- ...

Part IV

결론

컴퓨터 프로그래밍은 예술이다

- 세상에서 가장 즐거운 일 중 하나는 우리들이 작성한 컴퓨터 프로그램을 다른 사람들 혹은 여러분 자신이 읽고 기쁨을 얻는 것이다.
- 컴퓨터 프로그래밍은 음악, 미술, 문학과 같은 예술이다.
  - 축적된 지식을 세상에 적용해서 그렇고,
  - 기술과 독창력을 요구해서 그렇고,
  - 무엇보다도 아름다움을 만들어내기 때문에 그렇다.
- 어렴풋이나마 자신을 예술가로 인식하는 프로그래머는 자신이 하는 일을 즐길 것이며, 더욱 잘할 것이다.

Knuth's Interview: TUGboat(2005년)

Question: Please, tell our readers briefly what made you decide to start the project, which tools you used, and how many people you had at the core of the TEX team.

Answer: "The tools I used were home grown and became known as Literate Programming. I am enormously biased about Literate Programming, which is surely the greatest thing since sliced bread. I continue to use it to write programs almost every day, and it helps me get efficient, robust, maintainable code much more successfully than any other way I know. Of course, I realize that other people might find other approaches more to their liking; but wow, I love the tools I've got now. I couldn't have written difficult programs like the MMIX meta-simulator at all if I hadn't had Literate Programming; the task would have been too difficult."

참고문헌

- Knuth, *Literate Programming*, Center for the Study of Language and Information, 1992
- Knuth, *The CWEB System of Structured Documentation*, Reading, Massachusetts: Addison-Wesley, 1993
- Daniel Mall의 Literate Programming 웹사이트
  http://www.literateprogramming.com
- Chris Lee의 Literate Programming 간단한 소개 웹사이트
  http://vasc.ri.cmu.edu/old_help/Programming/Literate/literate.html
- 문학적 프로그래밍: http://faq.ktug.or.kr/faq/LiterateProgramming
- CWEB: http://faq.ktug.or.kr/faq/CWEB

Part V

KTUG: Korean TEX User Group

KTUG: 한글 TEX 사용자 그룹

- LATEX의 한글화와 보급 http://www.ktug.or.kr
- 2001년 이전
  - ChoF's TeX Archive : MiKTeX 2.1, HLaTeX 0.98/0.99.
  - 도은이네 집 : fpTeX, HLaTeX 0.98/0.99
- 2001 ∼ 2002 년간
  - KTUG : MiKTeX-KTUG 2.2 (by ChoF)
  - KTUG : HPack (by 홍석호)
  - 2002년은 가장 생산적인 해 중 하나였음. 옛한글, dvipdfm-cjk(DVIPDFMx), pdftex subfont patch(by ChoF)[1], ConTeXt 한글화 등등...
- 2003년
  - MiKTeX-KTUG 2.3
  - cvs를 이용한 HLaTeX의 설치가 권장된 적이 있음.
  - 은글꼴과 DVIPDFMx를 이용한 검색/추출 가능한 pdf 제작 기술 발전. 은글꼴이라는 GPL 트루타입 글꼴은 이 분야의 발전에 획기적인 전기였음.
  - 옛한글 관련 Omega/Lambd를 이용한 다양한 실험.

KTUG: 한글 TEX 사용자 그룹

- 2003 ∼ 2004년간
  - MiKTeX 2.4와 더불어 MiKTeX-KTUG 프로젝트 중단.
  - HPack for MiKTeX 2.4를 이용한 HLaTeX 설치 (홍석호)
  - DVIPDFMx의 MiKTeX 2.4를 위한 바이너리가 나오는 데 한참 걸려서 한동안 pdf 제작 기술이 실용화되지 못하였음. 이 때 CygWin teTeX에 대한 관심이 증가함.
  - pdf 제작 기술은 hangul-k(HANGULkStyle)로 발전함.
  - 한글 타이포그래피에 대한 관심이 증가하여 hlatex-interword 등이 제작됨.
  - 옛한글 및 고문헌 처리 시도들이 좋은 결과를 보이기 시작하고 DHHangul이 제작됨.
  - LaTeX2html에 대한 수요가 줄어들고 그 대신 TeX4ht의 한글화가 진행됨 (by synapse, HuidaeCho)
  - HanyangPuaTableProject가 진행됨.

KTUG: 한글 TEX 사용자 그룹

- 2005년
  - Hangul-ucs 패키지의 발전. DHUcs라는 이름으로 시작되어 나중에 상당한 규모의 프로젝트로 발전하였음.
  - HLaTeX 설치 지원은 MiKTeX + KTUG patch(dvipdfmx, ttf2pk, etc.) + HPack의 방법이 표준적인 방식으로 자리잡음. (이 기간이 상당히 길었음)
  - KTUGCollection2005 제작. MiKTeX + HLaTeX/DHUcs 등, 그 시기까지 발전한 거의 모든 한글 환경을 하나의 통합 설치 패키지로 만든 것이었음.
  - 한글 pdf 제작에 관하여 요구되던 거의 모든 문제점을 Hangul-ucs가 해결함.
  - Hangul-ucs의 발전과 더불어 memoir의 한글화가 이루어짐.

KTUG: 한글 TEX 사용자 그룹

- 2006년
  - ALee님에 의한 Hangul-ucs Debian 패키지가 나오고, WkPark 님에 의한 Fedora 설치 패키지를 시작으로 리눅스 쪽에서 Hangul-ucs가 광범위하게 받아들여짐.
  - Hangul-ucs까지 지원하는 HPack 1.2 (beta) 공개.
  - WinEdt의 정체 때문에 WinEdt은 HLaTeX 전용 에디터로 인식됨. Hangul-ucs 쪽에서는 Emacs나 EmEditor를 권장하게 됨.
  - KTUGCollection2006 제작. MiKTeX 대신 W32TeX/ko를 채택하고 최신 베타 버전의 pdftex을 탑재한 실험적인 TeX 환경이 선보임. 이 과정에서 ConTeXt, XeTeX의 한글 사용 가능성을 제시함.

KTUG Collection2006 CD

QnA

Perhaps we will even one day find Pulitzer prizes awarded to computer programs.

Donald Knuth, Literate Programming

감사합니다

Programming in the Emacs

신성국 <cinsky@gmail.com>

KLDP 10 Year Anniversary F/OSS Conference

Introducing Emacs

- Emacs is the extensible, customizable, self-documenting real-time display editor with built-in LISP interpreter.
- Supports various major modes for many file formats. (e.g. text, C, C++, java, python, lisp, sh, ada, Makefile, asm, etc)
- Supports multi-lingual environments with/without XIM.
- Large number of extensions which adds other functionality
- Rich documentation (core 608 pages, total more than 2400 pages)

Creative Commons Attribution—NonCommercial-NoDerivs 2.0 License

Why develop with Emacs?

- Can handle complicated text operations; search/replace, macros, abbreviations, smart in-dentation.
- Integrated compilation.
- Debugger supports.
- Documentation viewer(man, info)
- Shell support.
- Easy multiple buffer managements (frame/window)

KLDP 10 Year Anniversary F/OSS Conference

System Environment

- emacs 22.0.50
  - .emacs (http://www.cinsk.org/viewcvs/emacs-scripts/)
  - xcscope.el
  - aspell.el
  - css-mode.el
  - color-theme.el
  - nxml-mode.el
  - po-mode.el
  - psvn.el
  - gnuplot.el
  - htmlize.el

Creative Commons Attribution—NonCommercial-NoDerivs 2.0 License

Notation

- 일반 PC 에서 <Alt> 키가 meta 키의 역할임 .
- Sun, HHK 에서는 "<>" 키가 meta 키임 .

C-a	Control + <a>
M-[	Meta + <[>
C-M-r	Control + Meta + <r>
M-{	Meta + { 또는 Meta + Shift + <[>
C-x <RET> r	C-x 다음 <RET> 치고 <r>
M-x goto-line	M-x 다음 goto-line 치고 <RET>
C-q <TAB>	C-q 다음 <TAB>

KLDP 10 Year Anniversary F/OSS Conference

Editing Operations

- Contextual Navigation
- xcscope
- Smart Indentation (들여쓰기)
- set-selective-display
- cwarn-mode (C Warning Mode)
- Filling Paragraph
- Recursive Edit
- Keyboard Macro
- Replace with LISP code
- Spelling Check (ispell)
- Narrow/Widen Region

Creative Commons Attribution—NonCommercial-NoDerivs 2.0 License SOME RIGHTS RESERVED

Programming Tools

- Version Control
- Change Log
- Diff/Merge Tool
- Find/Grep Tool
- Compilation, gud
- Shell
- Dired (Directory Editor)
- ECB (Source Browser)
- Org-Mode (Organizer)
- Calc (Calculator)
- c-macro-expand

Contextual Navigation

- sexp/ 단락 / 함수 / 문장 / 페이지 단위로 scroll 가능
  - forward-sexp/backward-sexp
  - forward-paragraph/backward-paragraph
  - beginning-of-defun/end-of-defun
  - c-forward-subword/c-backward-subword
  - c-forward-subword/c-backward-subword
- 부분단어 단위로 이동 가능
  - (e.g print_msg, PrintMsg)
  - c-forward-subword/c-backward-subword
- 단어 /(..) 쌍 / 단락 / 함수 / 페이지 단위로 블럭 지정
  - 차례대로 M-@, C-M-@, M-h, C-M-h, C-x C-p
- Preprocessing directive 깊이에 따른 탐색
  - c-forward-conditional/c-backward-conditional
  - c-down-conditional/c-up-conditional

Emacs - /usr/local/src/paragui-1.1.8/src/core/pgnavigator.cpp

File Edit Options Buffers Tools C++ Cscope Help

```
        return my_currentWidget->Action(action);
}
^L
PG_Widget* PG_Navigator::Goto(PG_Widget* widget) {
        iterator i = find(begin(), end(), widget);

        if(i == end()) {
                return NULL;
        }

        Action(PG_Widget::ACT_DEACTIVATE);
        my_currentWidget = widget;
        Action(PG_Widget::ACT_ACTIVATE);

        return my_currentWidget;
}

PG_Widget* PG_Navigator::() {
        iterator i = begin();

        if(i == end()) {
                return NULL;
        }

        return Goto(*i);
}
^L
PG_Widget* PG_Navigator::GotoLast() {
        if(size() == 0) {
                return NULL;
        }
```

--:** pgnavigator.cpp 36% (78,25) (C++/l CWarn Abbrev)--[PG_Navigator::GotoFirs

KLDP 10 Year Anniversary F/OSS Conference

Emacs - /home/cinsk/src/tmp.c

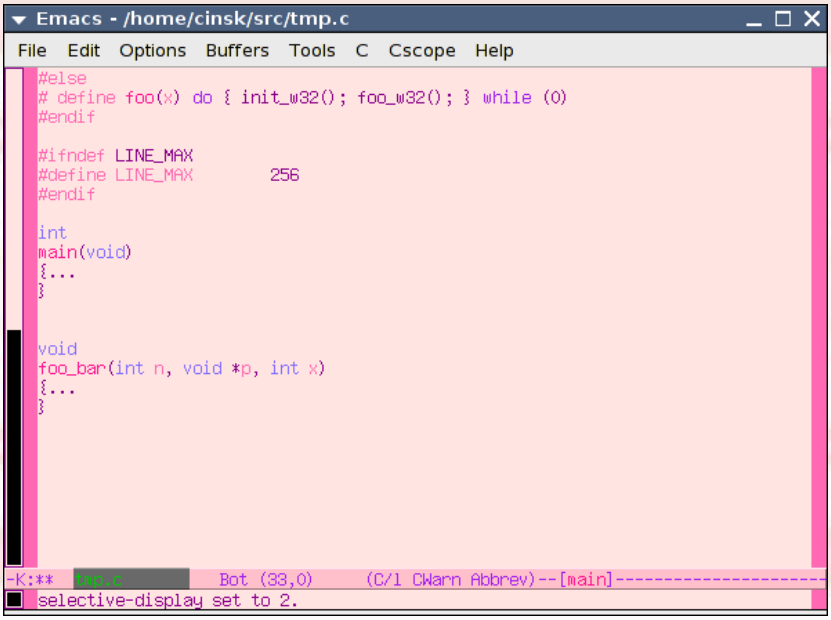
File Edit Options Buffers Tools C Cscope Help

```
int dist_xnorm[] = { 1, 2, 3, 7, 8, 10, 12 };
int dist_ynorm[] = { 0.12, 0.39, 0.61,
                     0.24, 0.39, 0.48, 0.61, };

#ifdef UNIX
# if defined(BSD)
#  define foo(x)         foo_bar(0, NULL, x)
# elif defined(MINIX)
#  define foo(x)         foo_minix(0, x)
# elif defined(SPARC)
#  define foo(x)         foo_sparc(x)
# endif
#else
# define foo(x) do { init_w32(); foo_w32(); } while (0)
#endif

#ifndef LINE_MAX
#define LINE_MAX        256
#endif

int
main(void)
```

-K:** tmp.c 11% (9,0) (C/l CWarn Abbrev)--[]----------

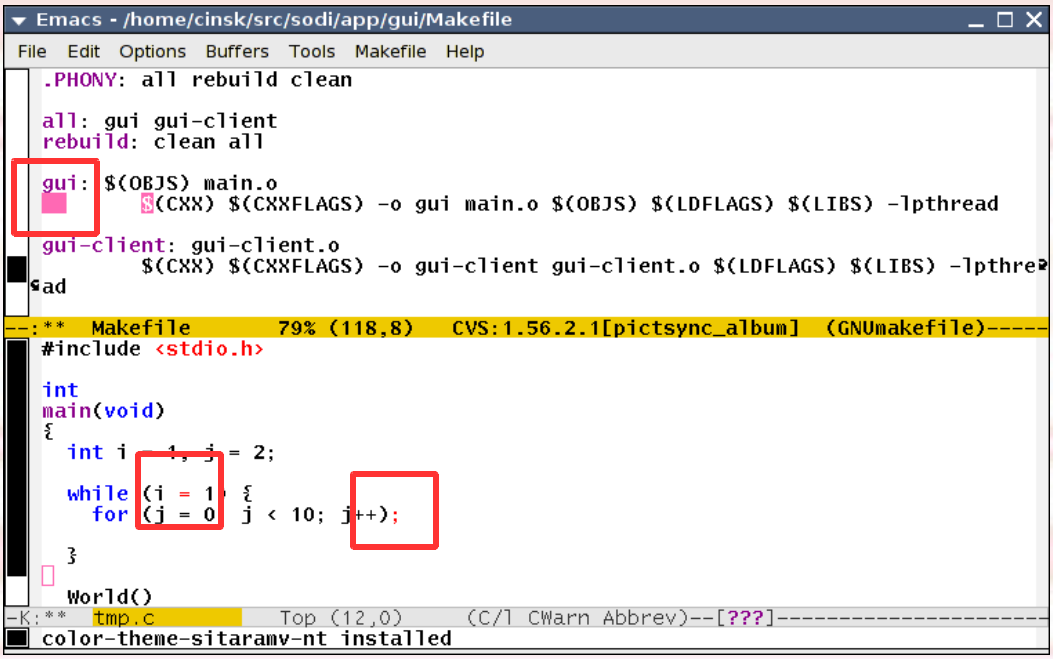
c-down-conditional-with-else

set-selective-display



KLDP 10 Year Anniversary F/OSS Conference

cwarn-mode / Makefile



Registers

- Emacs 는 여러가지 정보를 저장 / 불러올 수 있는 저장공간인 register 를 제공
  - 버퍼 위치
  - 텍스트 (블럭)
  - 텍스트 (사각형)
  - 윈도우 위치 / 크기
  - 프레임 위치 / 크기
  - 수치
  - 파일

Keyboard Macro

- To produce this list:

```
test1
test2
test3
...
```

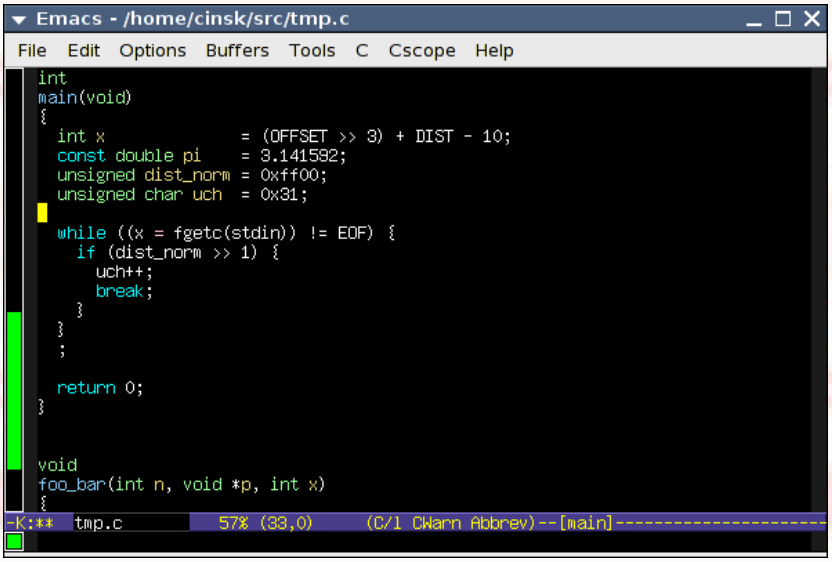
```
C-x r n q          ;  레지스터 q 에 1  저장
C-x (              ;  매크로 정의 시작
test C-x r i q     ;  레지스터 q  내용 붙이기
C-e                ;  커서 줄 끝으로 이동
C-x r + q          ;  레지스터 q  내용 증가
RET                ; newline
C-x )              ;  매크로 정의 끝
C-x e e e ...      ;  매크로 호출
```

Smart Indentation

- TAB indents the current line based on the language context, not inserting '\t' character.
- To insert real TAB char, press C-q tab
- M-x indent-region indents whole region(block) based on the context
- M-x align-regexp, M-x align**

KLDP 10 Year Anniversary F/OSS Conference

Smart Indentation (Demo)



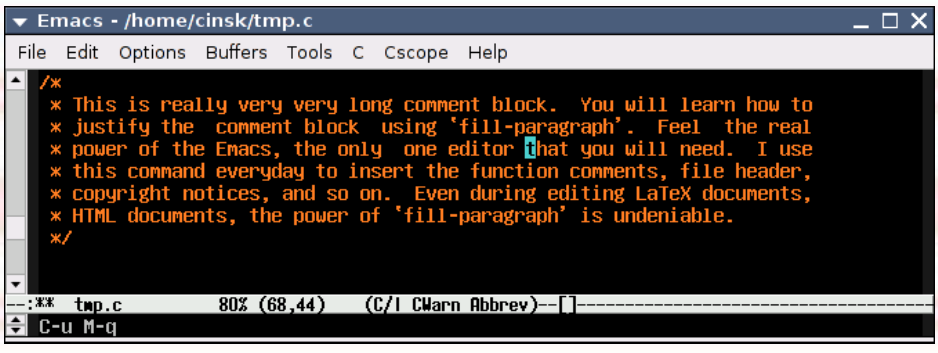
Creative Commons Attribution—NonCommercial-NoDerivs 2.0 License

Narrow/Widen Region

- 텍스트 연산을 현재 블럭에만 적용하고 싶다면 ?
- Restricting editing in this buffer to the current region.
- 특히 , 블럭 안에서만 여러 문자열을 바꿀 때 (AA->BB), 편리함 .

Filling Paragraph

- 현재 문맥에 맞게 단락을 채워주는 기능
- 블럭 단위의 주석을 달때 매우 편리함
- ‘M-q’ 또는 ‘ fill-paragraph’.
- ‘C-u M-q’ 를 쓰면 , (여분의 공백 문자를 추가 ,) 오른쪽으로 자리를 맞춰줌 .

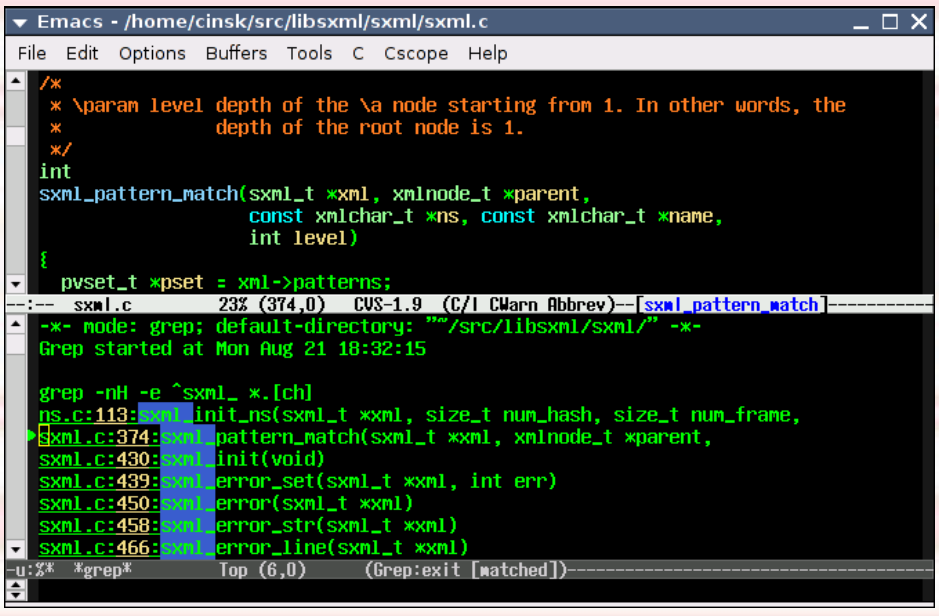


Grep #1

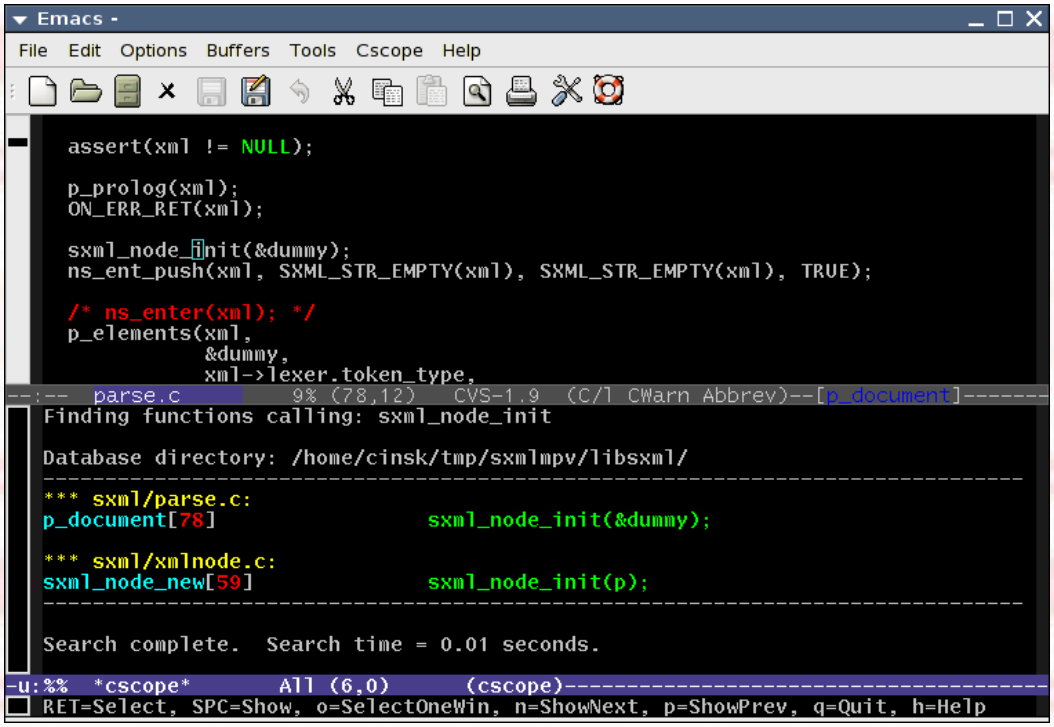
- Search for patterns using grep(1) then shows the search result with the `compilation' interface.
- “M-x grep” runs grep(1) asynchronously.
- “M-x find-grep” runs “find ... | xargs grep”.
- Use C-x ' to see the next search result.

KLDP 10 Year Anniversary F/OSS Conference

Grep #2



xcscope.el

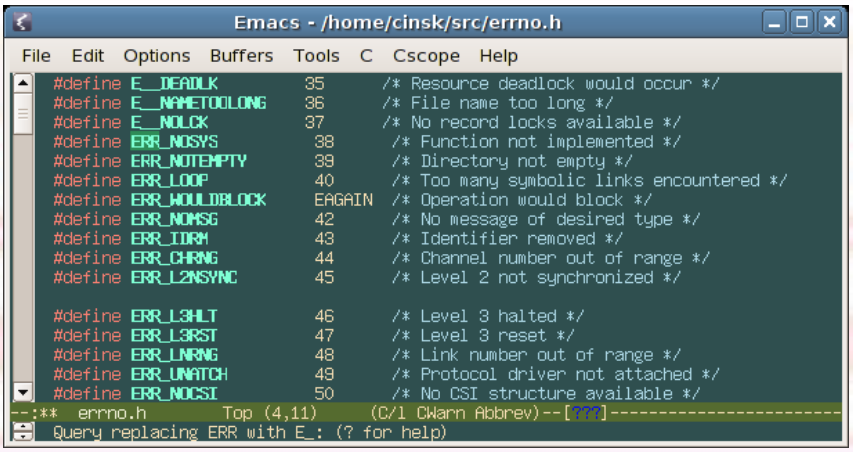


Recursive Edit #1

- 현재 버퍼에서 문자열 AAA 가 20 개 존재함 .
- 문맥에 따라 15 개를 BBB 로 바꾸고 있었다 . (M-x query-replace)
- 5 개를 바꾼 후 , 버퍼에서 CCC 를 발견 , 이 문자열도 BBB 로 바꿔야 할 상황이 발생 .
- Recursive edit 를 쓰면 , 현재 치환 명령을 중단하지 않고 CCC 를 BBB 로 바꿀 수 있다 .
- C-r 을 써서 recursive edit 시작 , C-M-c 를 써서 recursive edit 중단 .

Recursive Edit #2

- "ERR" 을 "E_" 로 바꾸던 도중 "EER" 을 발견 , recursive edit 로 "EER" 을 "ERR" 로 바꾸고 , 처음 치환 작업 계속 진행 .



Replace

- 단순 문자열 치환
- Regexp 문자열 치환
- 단순 문자열 하나씩 묻고 바꾸기
- Regexp 문자열 하나씩 묻고 바꾸기
- Regexp 새 문자열에 \,lisp 꼴로 LISP 코드를 직접 실행 가능 – 이 경우에도 \&, \1, \2, \3 의 표현이 LISP 코드에서 실행 가능

```
M-x replace-string <RET> STR <RET> NEW-STR <RET>
M-x replace-regexp <RET> REGEXP <RET> NEW-STR <RET>
M-x query-replace <RET> STR <RET> NEW-STR <RET>
M-x query-replace-regexp <RET> REGEXP <RET> NEW-STR
```

Replace with LISP code #1

```
x = y;
y = x;
x = x;
y = y;
```

```
y = x;
x = y;
y = y;
x = x;
```

M-x replace-regexp <RET> \(x\|y\) <RET>
\,(if \1 (string= "x") "y" "x") <RET>

KLDP 10 Year Anniversary F/OSS Conference

Replace with LISP code #2

```
We've nothing to say to
one another, devil.  You are
baatezu and I am tanar'ri.
There can be nothing but
death between us.
```

```
000 We've nothing to say to
001 one another, devil.  You are
002 baatezu and I am tanar'ri.
003 There can be nothing but
004 death between us.
```

M-x replace-regexp <RET> ^.*$ <RET>
\,(format "%03d %s" \# \&) <RET>

Replace with LISP code #3

```
#define XML_ENONE    0
#define XML_EEOF     1
#define XML_ETOKEN   2
#define XML_EPARSE   3
#define XML_ERUN     0x10
#define XML_EDEPTH   17
#define XML_EFILE    18
#define XML_EARG     19
```

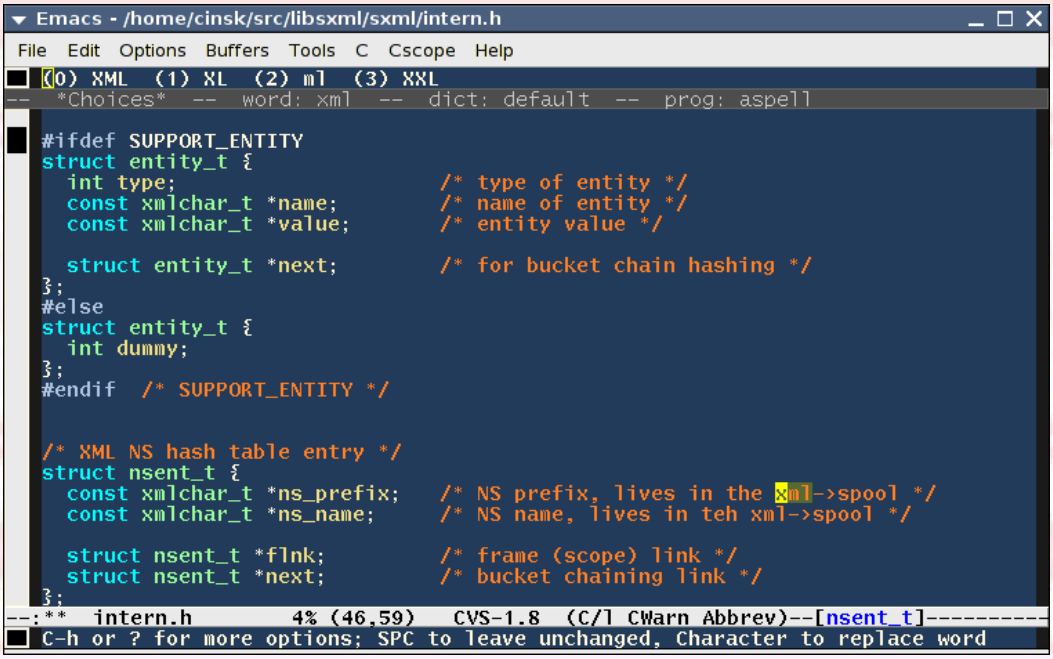
```
#define XML_ENONE    0
#define XML_EEOF     1
#define XML_ETOKEN   2
#define XML_EPARSE   3
#define XML_ERUNTIME 0x10
#define XML_EDEPTH   (XML_ERUN + 0x01)
#define XML_EFILE    (XML_ERUN + 0x02)
#define XML_EARG     (XML_ERUN + 0x03)
```

```
M-x replace-regexp <RET> \([0-9a-fx]+\)$ <RET>
\,(let ((num (string-to-int \1)))
    (if (> num 16)
        (format "(XML_ERUN + 0x%02x)" (- num #X10))
        \1)) <RET>
```


ispell

- Emacs 는 외부 프로그램 (ispell 또는 aspell) 의 도움을 받아 , 원하는 text 의 스펠링을 검사 / 수정해 준다 .
- 현재 버퍼 또는 블럭에 대해 스펠링을 검사할 수 있으며 (M-x ispell-buffer 또는 M-x ispell-region), 문자열과 주석 (comment) 부분에만 검사할 수도 있다 . (M-x ispell-comments-and-strings)

ispell (Demo)



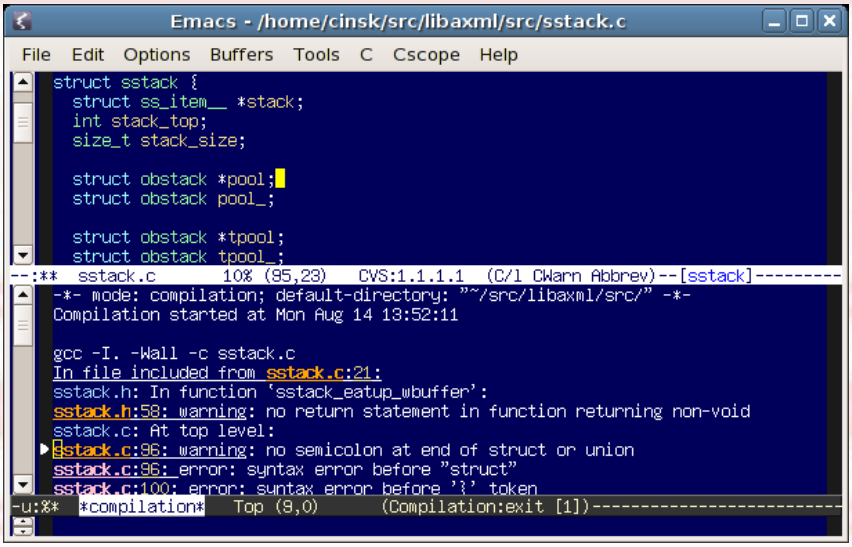
KLDP 10 Year Anniversary F/OSS Conference

Compilation

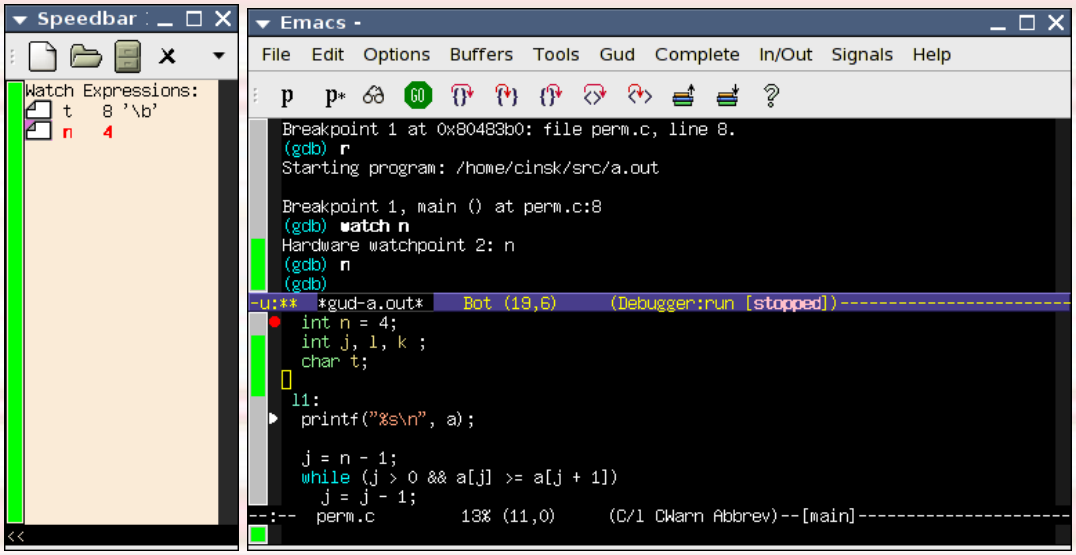
- Emacs can run compilers for noninteractive languages such as C and Fortran as inferior processes, feeding the error log into an Emacs buffer.
- Emacs 를 떠나지 않고 , 소스를 컴파일한 다음 , 에러가 발생한 곳의 위치를 바로 보여 줌 .
- GCC(g++,gcc,g77), Java, m4, shell, awk, perl, make 등 현존하는 거의 모든 언어 / 빌드툴 지원 ,

Creative Commons Attribution—NonCommercial-NoDerivs 2.0 License

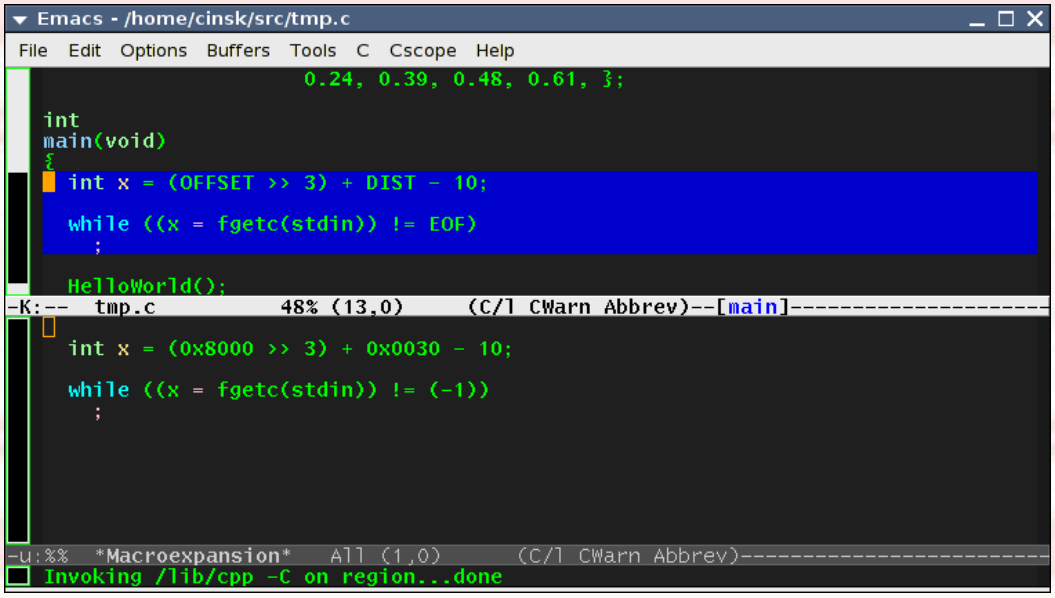
Compilation #2



gud (Debugger Mode)



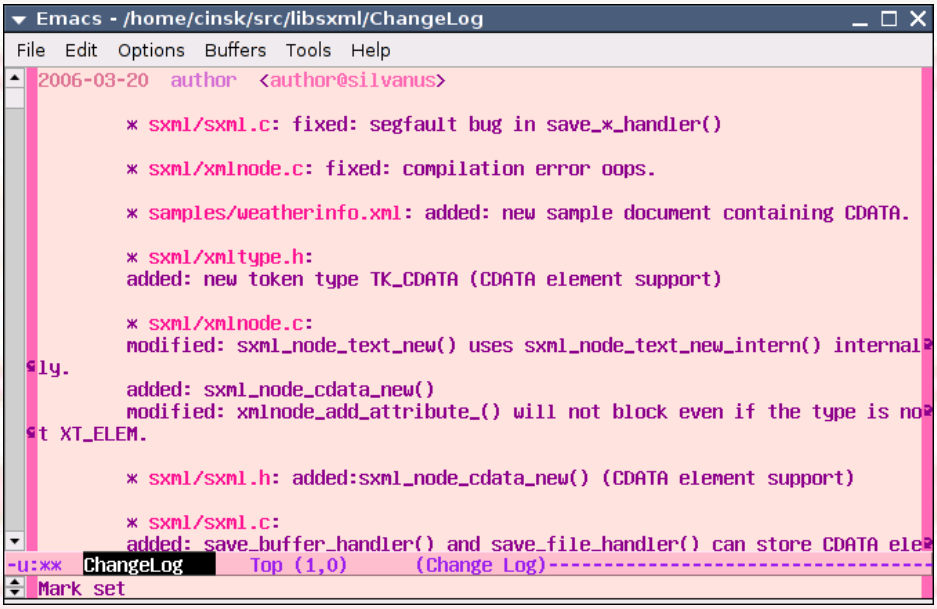
c-macro-expand



Change Log #1

- A change log file contains a chronological record of when and why you have changed a program, consisting of a sequence of entries describing individual changes.
- Normally it is kept in a file called ‘ChangeLog’ in the same directory as the file you are editing, or one of its parent directories.
- A single ‘ChangeLog’ file can record changes for all the files in its directory and all its subdirectories
- Emacs can add a new entry to the change log file, or create from the scratch for you.

Change Log #2



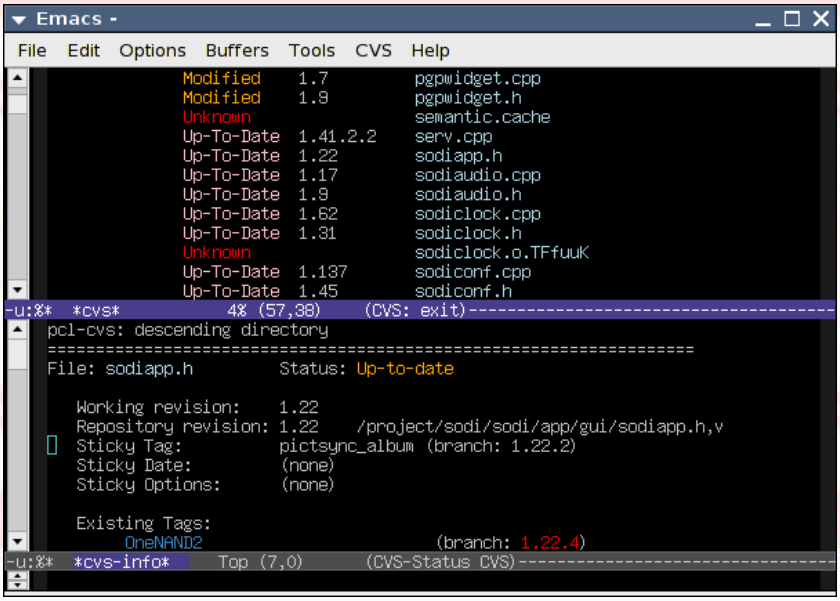
KLDP 10 Year Anniversary F/OSS Conference

Version Control

- CVS, GNU Arch, RCS, Meta-CVS, Subversion, SCCS 등을 지원 가능
- 대부분 명령은 'C-x v v' 또는 M-x vc-next-action 으로 수행 가능함 , 예를 들어 :
  - Readonly file -> cvs edit
  - Modified -> cvs commit
  - Old file -> cvs update
- M-x cvs-examine 을 통해 현재 상태를 점검
- 'd e' 로 diff 실행 , 'd E' 로 3-way diff (merge) 실행

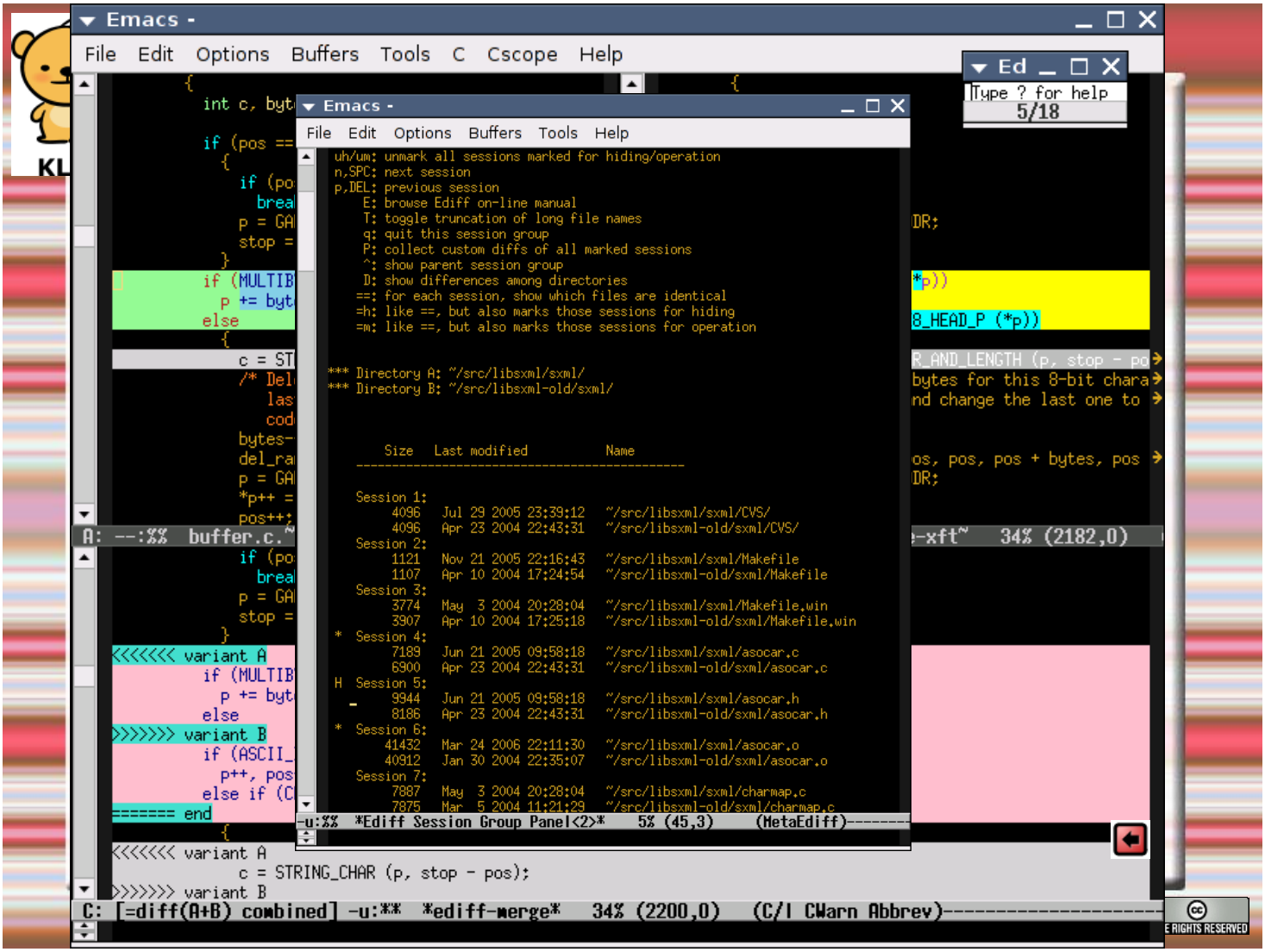
Creative Commons Attribution—NonCommercial-NoDerivs 2.0 License

PCL-CVS



ediff

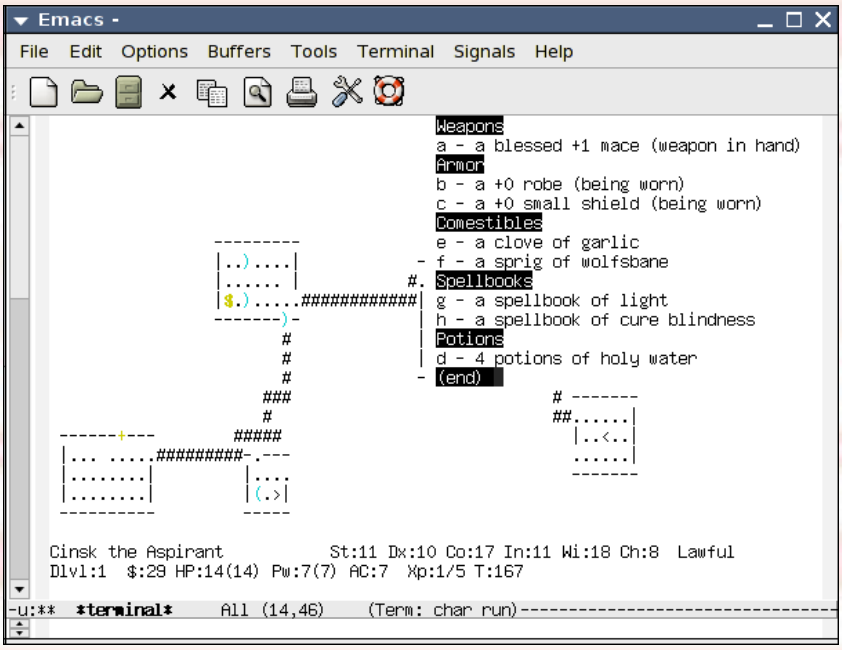
- 두 개 또는 세 개의 파일 / 버퍼 / 블럭의 차이점을 비교하거나 merge 하는 기능을 제공
- 두 개 또는 세 개의 디렉토리 비교 제공
- 패치를 적용하고 비교하는 기능 제공 (Multi-file patch 지원)
- Version Control System (e.g. CVS, Arch, RCS, Subversion, SCCS) 을 이해 , 다른 버전과 비교하는 기능 제공
- Remote, compressed file 지원 (e.g. FTP 로 압축 파일을 받아서 , 압축을 풀고 비교)



Shell #1

- 단순한 shell 명령 실행 (M-!)
- 현재 블럭을 shell 명령에 전달, 그 출력 결과를 가져옴 (M-|) 기존 블럭 치환은 (C-u M-|)
  - xxd 와 함께 쓰기 편함
- Interactive shell (dummy terminal) M-x shell
- Interactive shell (Full terminal emulation) M-x term

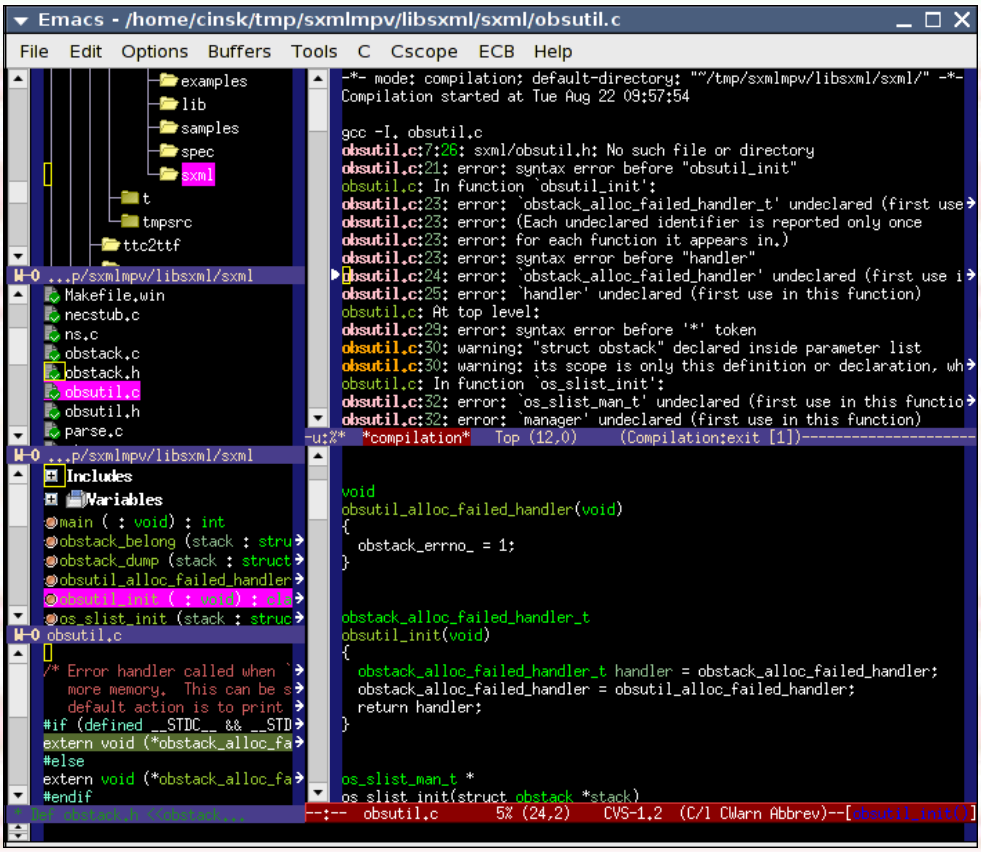
Shell #2



KLDP 10 Year Anniversary F/OSS Conference

Emacs Code Browser (ECB)

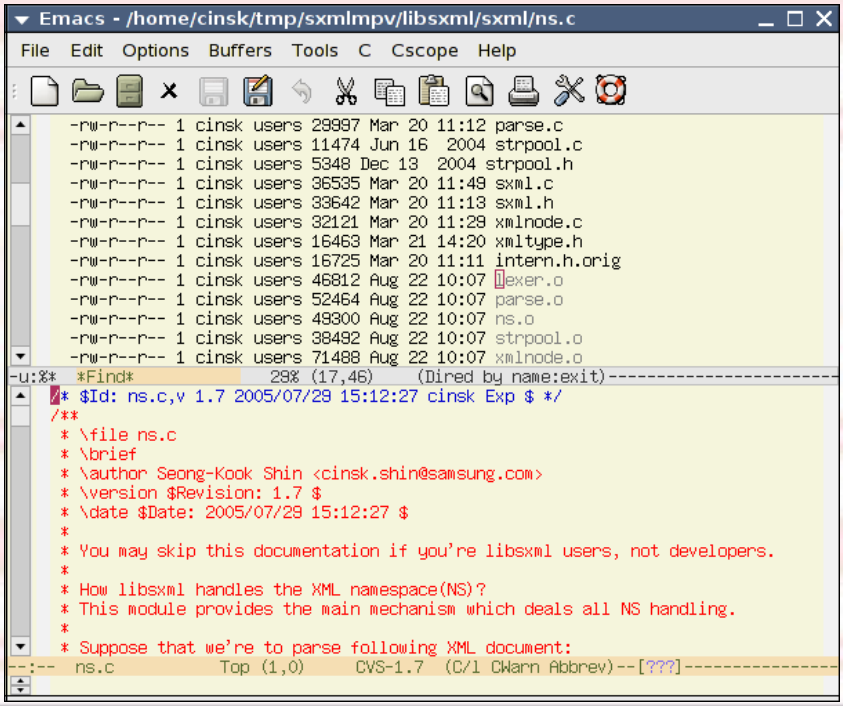
- Displays a number of informational windows that allow for easy source code navigation and overview.
  - A directory tree,
  - a list of source files in the current directory,
  - a list of functions/classes/methods/... in the current file,
  - a history of recently visited files,
  - the Speedbar and
  - output from compilation.



Dired

- 선택된 파일 이름을 모두 대문자로 바꾼다 .
- 선택된 파일에 원하는 shell 명령을 수행
- 파일 이름 Regexp 를 써서 파일을 선택
- 파일 내용 Regexp 를 써서 파일을 선택
- 선택된 파일 내용에서 Regexp 치환 실행
- 백업 / 임시 파일을 자동으로 숨김 / 선택
- Regexp 를 써서 파일 이름을 바꾸거나 새 이름으로 복사 또는 링크 생성 (예 : "^.*$" 를 "x-\&" 로)
- 파일 내용 Regexp 에 해당하는 파일만 리스팅
- "ls -l" style 텍스트 파일 import 기능

Dired (Demo)



KLDP 10 Year Anniversary F/OSS Conference

Calc

- Advanced calculator / mathmatical tool.
- Arbitrary precision integers / floating point numbers.
- rational/complex number
- lots of math functions plus bitwise/modulo op.
- financial/algebraic(e.g. symbolic calculus) function
- moving data to/from normal buffer
- graphics using GNUplot
- quick mode (수식 입력 후 , 결과를 버퍼에 추가)

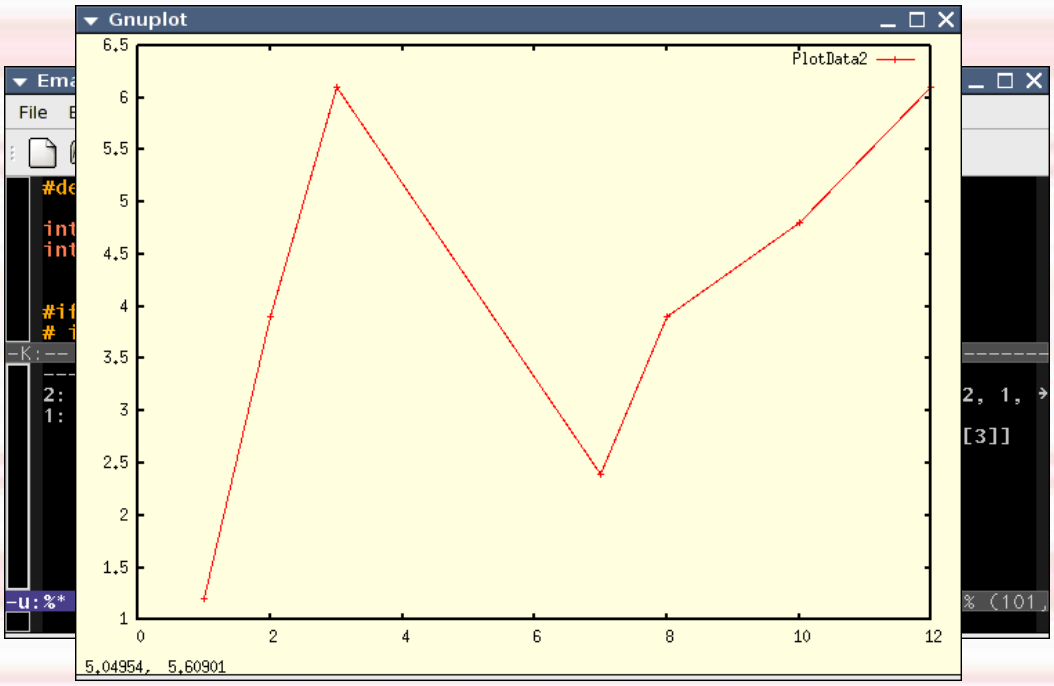
Creative Commons Attribution—NonCommercial-NoDerivs 2.0 License

Calc #2

- 0xDEADBEEF 가 10 진수로는 ?
- C 소스 파일의 int 배열의 분포를 그래프로 볼 수 있을까 ?
- 주어진 수치 값의 평균은 ?
- 현재 10 진수 값을 10bit shift 한 값은 ?

KLDP 10 Year Anniversary F/OSS Conference

Calc (Demo)



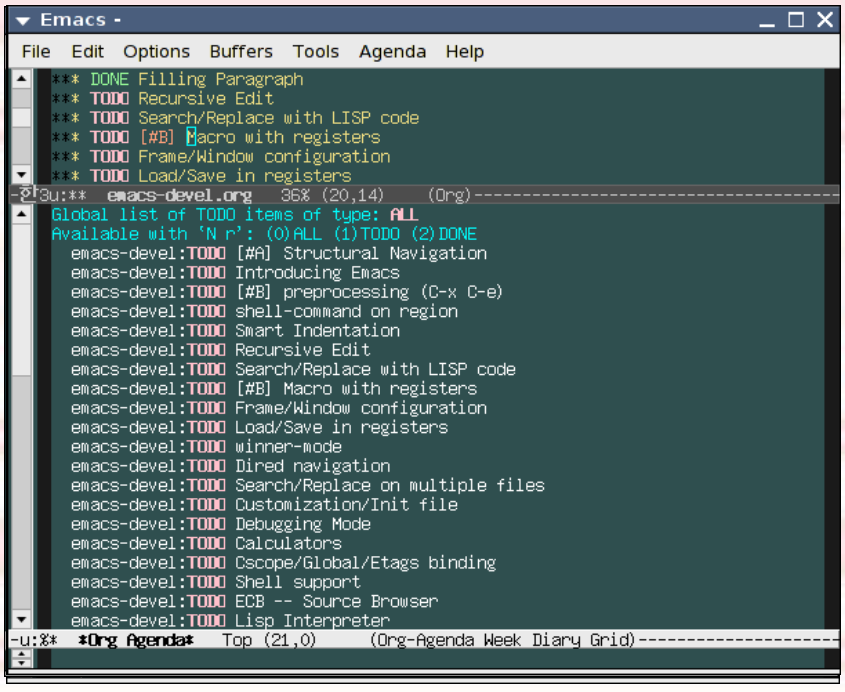
Creative Commons Attribution—NonCommercial-NoDerivs 2.0 License

Org-mode

- Org-mode 를 쓰면
  - 효과적으로 메모를 남기고 ,
  - TODO 리스트를 관리하고 ,
  - 프로젝트 스케쥴 관리가 가능함
- Wiki syntax 를 사용한 plain-text 포맷이기 때문에 , Emacs 가 아닌 곳에서도 부분 편집 가능 .
- 구조적 메모 , 하이퍼링크 , 스프레드시트 기능

KLDP 10 Year Anniversary F/OSS Conference

Org-Mode (Demo)



Creative Commons Attribution—NonCommercial-NoDerivs 2.0 License

오픈소스로 모바일 프로그래밍 하기

용영환 <xenonix@gmail.com>

Table of Contents

모바일 프로그래밍 시작하기

모바일 프로그래밍을 하기 위해서는 모바일 플랫폼과 이동통신사들의 특징을 알아야 한다. 모바일 플랫폼과 규격을 이해하고 모바일 응용프로그램을 작성 할 수 있도록 한다.

모바일 플랫폼이란

모바일 기기에서 응용프로그램이 실행될 수 있는 기초를 이루는 시스템을 의미한다. PC 에서 리눅스,윈도우즈 등이 있듯이 모바일 기기에서도 다양한 플랫폼이 있다.

국내 이동통신사 플랫폼 현황

이동통신사	*플랫폼*			
		사용언어	*개발사*	*비고*
SK 텔레콤	GVM	C/C++	Sinji Soft	Interpreter(VM)
	SKVM	Java	XCE	Interpreter(VM)
KTF	BREW	C/C++	Qualcomm	Binary
LG 텔레콤	KVM	Java	Sun Microsystems	Interpreter(VM)
3 개 통신사	WIPI	C/C++ , Java		Binary

WIPI 플랫폼

WIPI 의 필요성

이동통신사마다 서로 다른 플랫폼에서 응용프로그램을 구동하고 있기 때문에 응용프로그램 개발사들은 하나의 컨텐츠를 다수의 플랫폼에 맞춰 개발해야만 했다.

이동통신사의 차별화된 서비스를 충족하면서 상호 호환성을 고려한 플랫폼의 필요성.

표준 WIPI 규격

WIPI 란 모바일 응용프로그램의 기반 시스템에 대하여 정해놓은 규격으로 2005 년 1 월에 WIPI 2.0.1 규격이 한국통신기술협회 규격으로 채택되었다.

WIPI 플랫폼

표준 WIPI 규격을 구현해 놓은 기반 시스템을 말하며 현재 WIPI 2.0 규격의 플랫폼이 사용되고 있다.

현재의 WIPI 플랫폼

2005 년 4 월 1 일 정부가 국내 모든 휴대전화기에 WIPI 탑재를 의무화한 이후 WIPI 플랫폼을 탑재한 휴대전화기 보급대수가 1000 만대를 돌파하였다. 위피 플랫폼의 시장은 점차 확대되어 가고 있으며 더불어 위피 기반 응용프로그램 개발이 활성화 되고 있다.

WIPI SDK 설치하기

오픈소스 개발을 지원하고 있는 XCE 의 개발 환경에서 응용프로그램을 개발하는 방법을 알아본다.

XCE 웹사이트 : http://www.xce.co.kr

SDK 다운로드

XCE 의 개발자 지원 사이트인 http://www.developerzone.co.kr/ 에 회원가입을 한 후
SDK 중에 `SK-VM 에뮬레이터 2.0 & WIPI 에뮬레이터 2.0.2` 버전을 다운로드 한다.

SDK 설치

XCE 의 SDK 는 WIPI-Java SDK 라고 부른다. WIPI-JAVA SDK 는 Eclipse Plugin 을 제공하고 있다.

WIPI-Java SDK 구성물

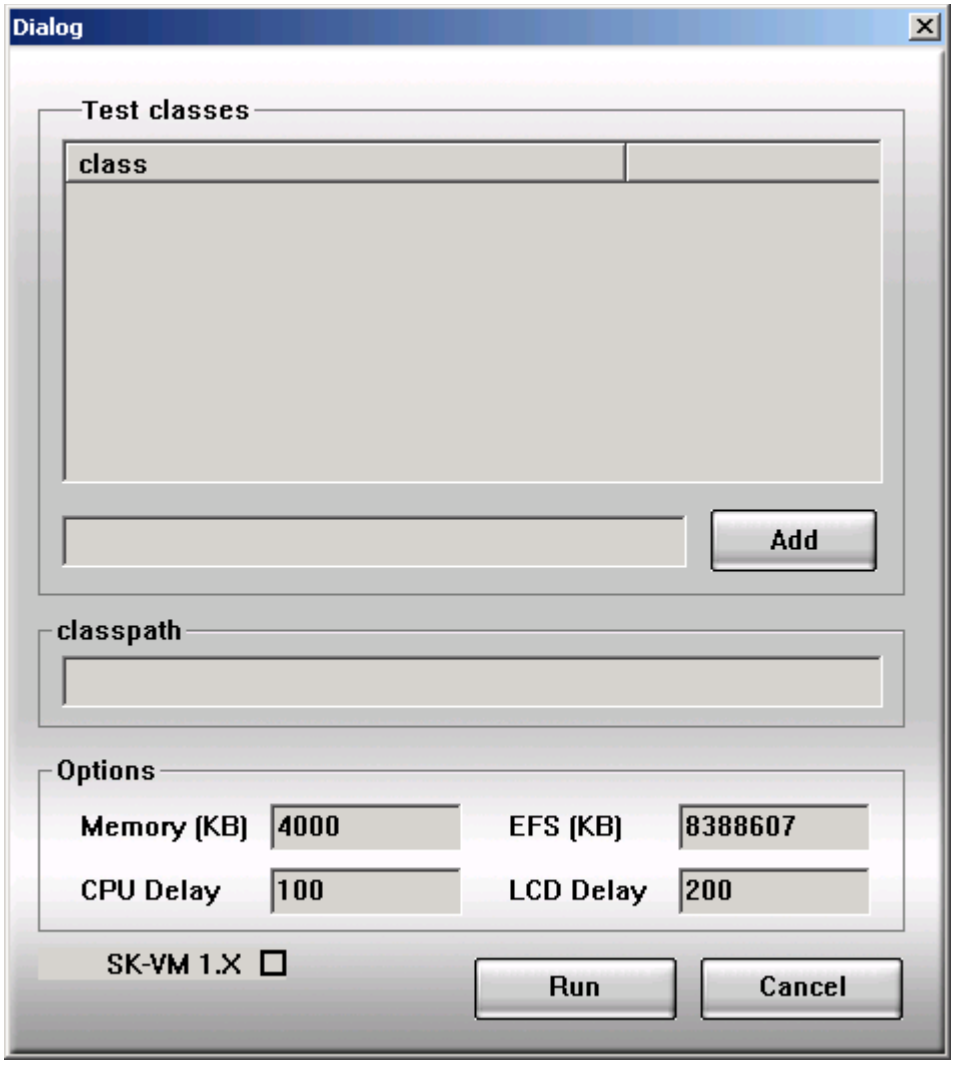
WIPI-Java 에뮬레이터, LBM PNG Mixer, LBM Viewer, MMF Player, PNG to LBM 변환기

에뮬레이터에서 예제 실행

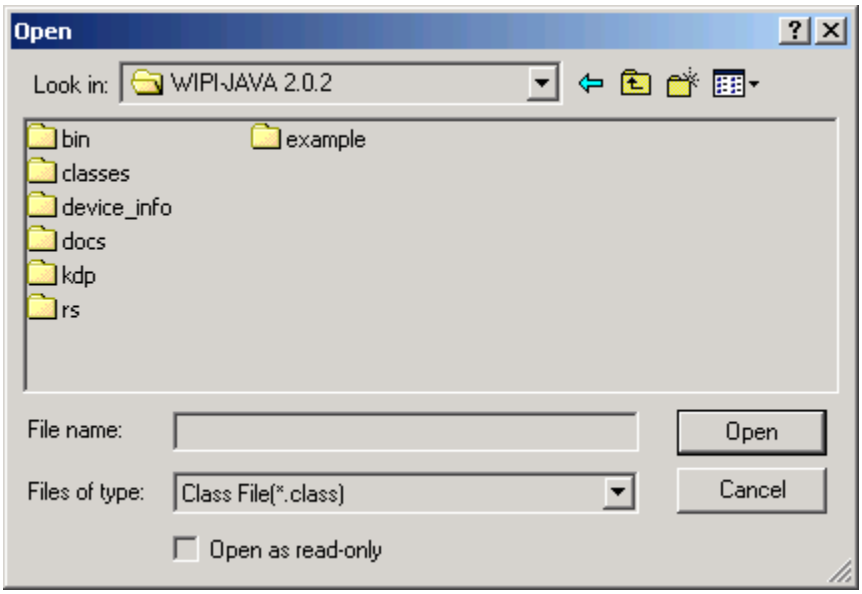
WIPI-Java를 설치하면 바로 실행 가능하도록 컴파일된 예제와 직접 분석해 볼 수 있는 소스 예제를 제공하고 있다. 예제는 WIPI-Java SDK가 설치되어있는 곳의 example 폴더에 담겨있다.

예제를 통해 에뮬레이터의 사용법을 알아본다.

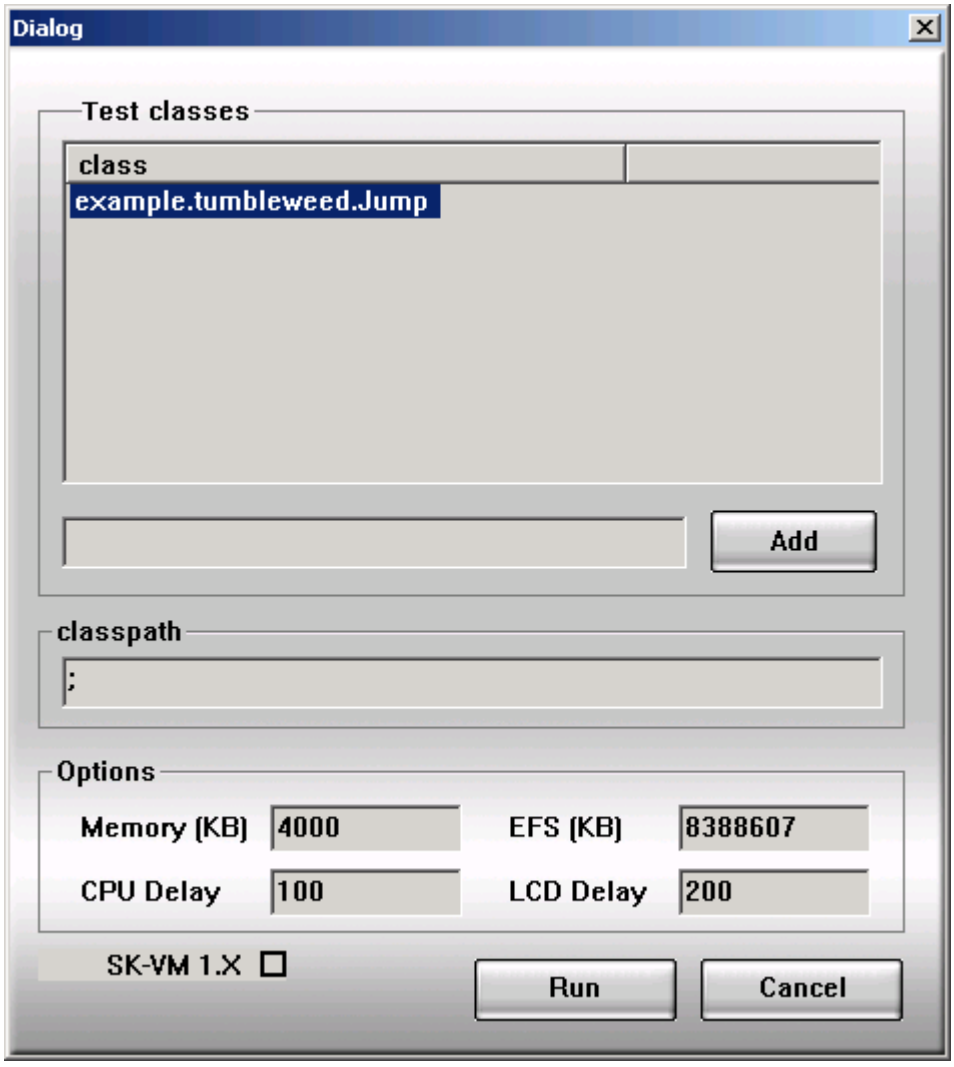
1. WIP-Java SDK를 실행한다.
2. File -> Open 을 클릭한다.



3. Add 버튼을 클릭하여 WIPI-Java 가 설치되어 있는 폴더로 이동하면 example 폴더가 보일 것이다.



4. Example -> tumbleweed -> Jump.class 를 선택하여 Open 을 클릭한다.



5. Run 을 클릭하여 실행한다. 컨트롤러를 통해 움직임을 제어할 수 있다.

WIPI 관련 웹사이트

포럼

위피 표준화 포럼	http://wipi.or.kr
한국 무선인터넷 표준화 포러	http://www.kwisforum.org

개발사

XCE	http://www.xce.co.kr
아로마소프트	http://aromasoft.co.kr
신지소프트	http://sinjisoft.co.kr
Sun Microsystems	http://kr.sun.com/developers/j2me/tutorial.html

커뮤니티

XCE 개발자 커뮤니티	http://www.developerzone.co.kr
모바일자바	http:/mobilejava.co.kr

F/OSS Track

KDE 프로젝트 소개

(조성재)

모질라 프로젝트의 현재와 미래

(윤석찬)

오픈소스의 PMC와 커미터, Google Summer of Code

(이창신)

오픈소스 프로젝트에 참여하기

(장혜식)

KDE 프로젝트 소개

KDE 한국 번역 Coordinator
조성재

KDE 프로젝트 발생 배경

- FreeBSD, Linux 등의 공개된 유닉스 계열 OS를 일반 PC에서도 사용 가능하게 됨
- Windows와 Mac OS 같은 사용하기 쉬운 데스크탑 환경이 Unix 계열 OS에 없었음
- 초고속 인터넷 환경이 일반인들에게 보급되고 있었음

Kool Desktop Envrionment

- 네트워크 상에서 공개되어 개발되고 있는 데스크탑 환경
- 1994년 10월 14일 Matthias Ettrich 가 제안 – 데스크탑 환경의 통일성을 주장
- 패널, 파일관리자, 메일클라이언트, 쉬운 텍스트 에디터, 터미널, 이미지 뷰어, 창 관리자, 도움말 시스템, 시스템 도구, 게임, 그리고 응용프로그램 개발 환경

사용자 관점의 KDE

- 현대적인 외양을 갖춘 데스크탑 환경 제공
- 편리한 도움말 시스템
- 표준화된 메뉴, 도구모음, 단축키 등
- 대화상자식의 데스크탑 환경 설정
- 여러 나라의 언어를 지원
- 많은 수의 KDE 응용프로그램들

현재 KDE의 패키지 구성 분류(1)

- aRts: 실시간 아날로그 신서사이져와 사운드 데몬. KDE 4 에서는 Phonon 으로 제안된 시스템을 사용
- KDE-Libs : KDE 에 사용되는 다양한 런타임 라이브러리
- KDE-Base : 창 관리자, 데스크탑, 패널, 컨커러와 같은 기본 구성요소

현재 KDE의 패키지 구성 분류(2)

- KDE-Network : KNode, KNewsticker, Kppp 등 인터넷을 이용하는데 유용한 프로그램들
- KDE-Pim: KMail, KAddressbook, KOrganizer, KPilot 등 개인정보를 관리하는데 필요한 프로그램들
- KDE-Graphics: KDVI, KGhostview, KPaint, Kfax 와 같은 그래픽 관련 도구들

현재 KDE의 패키지 구성 분류(3)

- KDE-Multimedia: Noatun, KMidi, KSCD 등 멀티미디어 매체를 재생, 편집하기 위한 프로그램들
- KDE-Accessibility: 컴퓨터 사용에 있어, 신체적인 어려움이 있는 사람들을 위한 프로그램들
- KDE-Utilities: KEdit, KCalc, KHexEdit, KNotes 등 컴퓨터 사용에 편리한 프로그램들

현재 KDE의 패키지 구성 분류(4)

- KDE-Edu: 교육용 프로그램
- KDE-Games: 게임
- KDE-Toys: 데스크탑을 사용하는데 흥미를 갖출만한 프로그램
- KDE-Addons: 프로그램을 확장하기 위한 추가 프로그램들
- KDE-Artwork: 테마나 아이콘등의 패키지

현재 KDE의 패키지 구성 분류(5)

- KDE-Admin: 시스템 관리에 필요한 (GUI 방식의) 응용프로그램
- KDE-SDK: KDE 응용프로그램 개발을 간단하게 해주는 도구, 스크립트들
- KOffice: 통합된 오피스 환경
- KDevelop: C/C++ 통합 개발 환경(IDE)
- KDE-Bindings: 다양한 언어(Python, Ruby, Perl, Java...)와 데스크탑 환경 연결

현재 KDE의 패키지 구성 분류(6)

- KDEWebdev: 웹 개발 도구
- 그 외 비공식적인 가상 패키지
  - KDE-Extragear : KDE 프로그램과 연결되는 프로젝트이지만, 여러가지 사유로 공식 배포를 할 수 없는 프로그램
  - KDE-playground : KDE-Extragear와 비슷하지만, KDE 프로젝트와 관련되어 있으면서 KDE 프로젝트외의 프로젝트

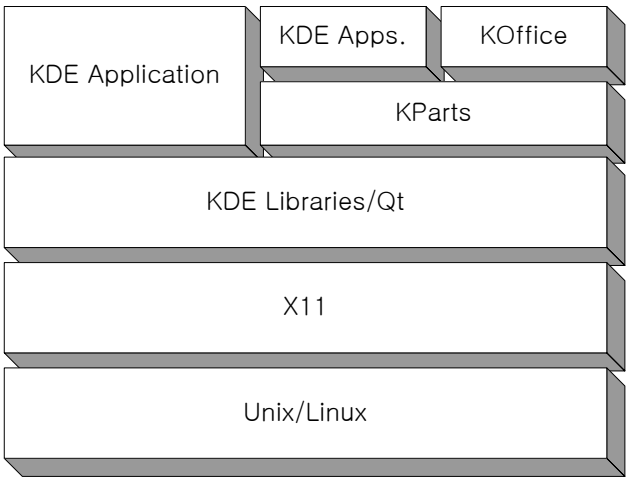
현재 KDE 프로젝트의 규모

- 400백만 줄의 코드
- 800 명 이상의 KDE 개발 공헌자
- 300 명 이상의 개인 번역자
- 12 개국, 17개 이상의 공식 미러
- 39 개국, 106개 이상의 공식 FTP 미러

K오피스의 구성

- KWord – 워드프로세서
- KSpread – 스프레드 시트
- KPresenter – 프레젠테이션 도구
- Kivio – Visio와 같은 도표 그림 제작 도구
- Karbon14 – 벡터 방식 드로잉 도구
- Krita – 스칼라 방식 이미지 제작 도구
- Kugar – 리포트 제작 프로그램
- KChart – 그래프, 차트 제작 도구
- KFormula – 수식 편집기
- Kexi – 데이터 관리 통합 환경

개발자 관점의 KDE



개발자 관점의 KDE (2)

- C++ 로 제작됨
- KDE 프로젝트 전반이 객체지향적 관점으로 제작
- KDE 프로젝트 제작을 위한 응용프로그램 개발 프레임 워크를 갖추고 있음
- 응용프로그램 Kapplication 객체를 사용하거나 KMainWindow를 이용하여 생성

Qt 라이브러리에 대한 의문

- KDE 개발에 한하여 비공개적으로 라이센스(Qt 라이센스)를 허용하였으나, 현재 Qt 4.x 버전에서는 GPL 을 따름
- Qt 라이브러리에 대한 사용에 대하여 KDE Free Qt Foundation 을 설립하여 현재와 향후 자유소프트웨어 제작에 대한 라이센스 지원을 획득

한국에서의 KDE 프로젝트

- 번역 프로젝트
  - http://kldp.net/projects/kdei18n/
  - http://l10n.kde.org/teams/infos.php?teamcode=ko
  - http://wiki.kldp.org/wiki.php/KDETranslationHowto
- 개발 프로젝트
- 사용자 그룹

Mozilla Project : Past and Future

Seokchan Yun

Mozilla Korean Project

http://www.mozilla.or.kr

Agenda

- History of Mozilla and Firefox
- Firefox 1.0 & 1.5
- Firefox 2.0 and 3.0
- Mozilla Community and Developement
- How to Join Mozilla Project

- Mini Session: Daum Open Strategy

Timeline

- April 1, 1998 – Communicator source code released and mozilla.org created
- November 10, 1998 – Move to Gecko announced
- September 25, 2000 – New Mozilla roadmap published
- November 14, 2000 – Netscape 6 released
- September, 2001 - relicensing begins, Mozilla to be relicensed under an MPL/GPL/LGPL tri-licence
- October 15, 2001 – Mozilla 1.0 Manifesto published
- June 5, 2002 – Mozilla 1.0 released
- August 28, 2002 – Netscape 7 ships without pop-up blocking
- September 23, 2002 - Phoenix 0.1 released
- July, 2003 - Mozilla Foundation is born. $2 million start-up support from AOL's Netscape division; Mitch Kapor pledges support, joins board
- February 9, 2004 - Firefox 0.8 released to millions of downloads
- February, 2004 - Mozilla Europe is founded to spur Mozilla's community, mindshare and marketshare in Europe
- August, 2004 - Mozilla Japan is founded on the heels of Mozilla Europe
- November 9, 2004 - Firefox 1.0 released
- December, 2004 - Mozilla rolls out email client, Thunderbird 1.0
- March, 2005 - Mozilla China is founded to tap into China's thriving Firefox community
- August, 2005 - Mozilla Foundation forms wholly-owned subsidiary, the Mozilla Corporation
- October, 2005 - Firefox surpasses 100 million downloads
- November, 2005 - Firefox 1.5 released
- January, 2006, Thunderbird 1.5 released
- June, 2006 – Firefox acquires 10% market share.
- August, 2006 – Firefox surpasses 200 million downloads
- October, 2006 – Firefox 2.0

Open Sourcing the Browser

Netscape, in a strategic move to try to compete with Microsoft's overwhelming IE push, decides to enlist the support of the Open Source community.

Source code to Communicator 5 released and the virtual organization, mozilla.org, set up to manage the open source project.

Early community rejects old Communicator 5 codebase and demands a fresh start with Gecko.

Communicator Transformed

- Netscape decides to move to Gecko and during the next year, a lot happens:
  - Gecko rendering engine becomes default
  - XPCOM and XPConnect developed
  - XPFE strategy announced and implemented
  - Mozilla achieves full CSS1 properties support
  - XPAT XML parser integrated
  - Necko replaces Netlib
  - XSLT, mathML, and more arrive in Mozilla codebase

New Mozilla Roadmap

On September 25, 2000, Brendan Eich posted a new roadmap for Mozilla outlining a strategy for Mozilla 1.0 independent of Netscape's release plans.

"Going forward, as netscape.com prepares to ship a commercial product based on Mozilla, we need a roadmap that prescribes fewer technical points and more planning and scheduling techniques."

"Mozilla needs performance, stability, and correctness. We are near the last ten percent of the 'Mozilla 1.0' project, where the going gets tough."

Netscape 6

On November 14, 2000, Netscape 6 was released and most reviews were negative. Netscape 6 suffered in terms of performance, stability, heavy-weight advertising, and even standards compliance.

"Netscape Navigator 6.0 to Fail Standards Compliance"

"Netscape 6: Does Anyone Care?"

"Netscape 6 Released to Mixed Reviews"

Mozilla 1.0 Manifesto

On October 15th, 2001, Brendan Eich published the Mozilla 1.0 Manifesto – developed by drivers@mozilla.org to define Mozilla 1.0:

- API compatibility commitments (for example, the frozen embedding APIs).
- Library version identification (so at least vendors can know when to require an upgrade, or to bundle a new rev to replace a downrev.dll or .so file).
- Enough modularity that important core modules (e.g., XPCOM) can stand alone.
- Stability -- acceptable MTBF, no serious dataloss bugs.
- Good performance and memory footprint.
- Better-than-any-competition standards compliance.
- Usability, correctness, polish (not too many non-crash bugs and misfeatures).

Mozilla 1.0 Released

"I have *no* complaints on (Mozilla 1.0's) standards-compliance," – Tim Bray, a founding member of the W3C's Technical Architecture Group"

"Mozilla 1.0 has gone gold, and from what we've seen, it's been worth the delay"

"The Mozilla project has quietly become a key building block in the open source infrastructure." – Tim O'Reilly

"Mozilla is the best free alternative to Microsoft IE and it's faster to boot." – Rex Baldazo, c|net

Netscape 7 Released

Netscape offers little that can't be had in the Mozilla suite, and in some cases, offers considerably less.

"Don't switch browsers" reads headline as reviewers learn that Netscape removed Mozilla's popular pop-up blocker for Netscape 7.

Reviewers weren't the only ones disappointed.

Birth of Firefox

During the same year that Mozilla 1.0 and Netscape 7 were released, development began on what would become the Firefox web browser. This splinter project was started by a small group of disgruntled Mozilla developers who were extremely disappointed in the Netscape browser and not satisfied with Mozilla's mediocre successes.

We believed in the technology that Mozilla had built in the last 4 years, but we knew that commercial organizations like Netscape and Microsoft did not care about making a better browser because it was not in their financial interest to do so.

- In 1999, the IE team was more than 1,000 people and it crushed Netscape removing any commercial incentive in web browsers and in 2001, after the release of IE 6, the IE team was disbanded.
- With no money to be made from the browser itself, the Netscape.com portal, which was still generating revenue, took over the development of the Netscape browser.

Phoenix 0.1 to Firefox 0.8

The first public version of the browser that would become Firefox happened on September 23, 2002 and was called Phoenix 0.1.

The amazing success of the early version, Phoenix 0.1, validated the new development model.

During the next year and a half, we released seven more updates and on February 9, 2004 Firefox 0.8 shipped. Firefox 0.8 was the first Firefox version that received a lot of press notoriety and even some early mainstream adoption.

By this point, Firefox was seeing stronger adoption than the Mozilla 1.x application suite.

Firefox 1.0

On November 9, 2004, Firefox 1.0 was released and downloaded by 10 million people in the first month demonstrating that an open source project, developed under our user-focused model, could achieve rapid adoption. More than 100,000,000 downloads in the first year!

"Security, Cool Features Of Firefox Web Browser Beat Microsoft's IE" - Walt Mossberg, Wall Street Journal

MAXIMUM PC - "Softy Awards 2004" - 1st Place "The créme of this year's software créme…. It is nothing short of a revelatory experience…. We fell in love with its open-ended nature and the access it gives us to an ever-expanding slew of interesting extensions."

CNET Editors' Choice, November 2004 "Mozilla Firefox is the dream Internet browser you've been looking for…. you should switch to Firefox--now."

Firefox 1.5

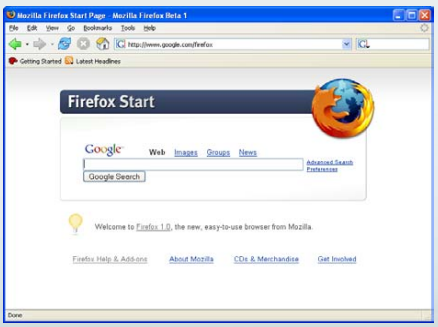
New Features

User Experience
- Tabbed browsing enhancements
- Performance
- Accessibility
- Usability improvements
  - User Preferences
  - RSS Discovery
- Improved pop-up blocking
- New search options - e.g., Answers.com

Open Standards
- Improving already best in class standards support
- <CANVAS> (think of it as "programmable <img>") and SVG support
- Enabling next generation web applications

Security and Privacy
- Automated Update
- Clear Private Data
- Architectural changes and development process improvements

Extensibility
- Extensions allow users to customize their browser to fit their needs
- Over 700 extensions.

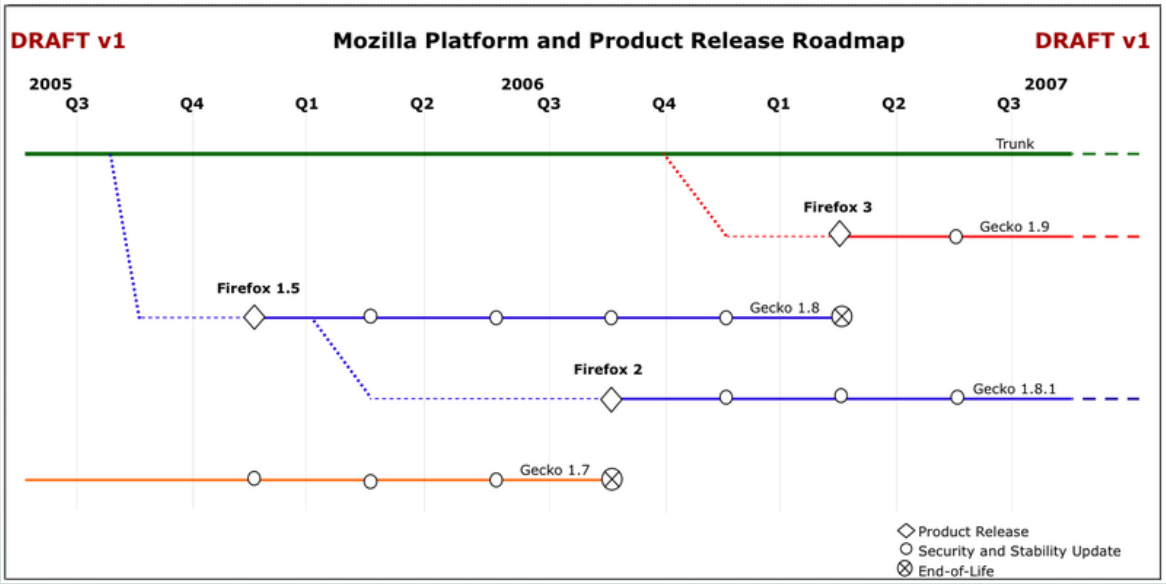


Beyond 1.5

We are moving to a new roadmap, to co-evolve Firefox with the Web.

- Since 1.0, we are seeing the introduction of very rich Web applications and services. Firefox is a strong platform for delivering the next generation of the Web.
- We're working with major Web partners, entrepreneurs and innovators who share our goal of making the web experience better.
- To do our part to help the Web grow, we're moving to a much more aggressive timetable for major releases.
- Where Mozilla has historically been on a 1-2 year release cycle, we're looking now to issue major releases every 6-9 months. Every other release will be a platform release that incorporates the latest Gecko engine.
- This will see us releasing Firefox Q3 2006, and Firefox 3 with the new Gecko engine in Q2 2007.

Mozilla Roadmap



Firefox 2

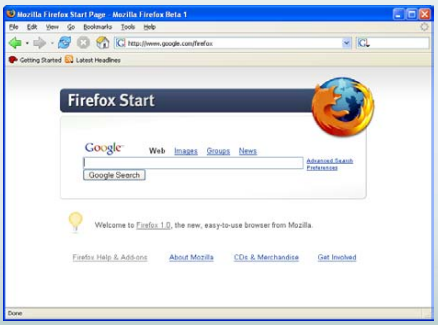
No Gecko update, Front-end features added and improved

User Experience
- Tabbed browsing UI enhancements
- New Theme
- Better support for previewing and subscribing RSS feeds
- Phishing Protection
- Enhanced search engine management

Benchmarking
- Resume data when crash and restart
- Inline spell checking
- New windows installer (NSIS)
- Enhanced Preferences

Open Standards
- JavaScript 1.7
- Support SVG:textPath
- Microsummaries
- Open Search
- DOM Stroage

Extensibility
- New add-on manager and update system.



Firefox 3

Gecko update, Graphic Experiences and Software Platform

- Various Patches for Gecko's bug and fuctions
- Completed Cairo Graphics
- Firefox on XULRunner
- Python for XUL
- Open Standards
  - Microformats, XBL2, Javascript2, SVG:Image
  - Cross-domain XMLHttpReauest and WHATWG things
- Places : New bookmark and Cache Storage with SQLite
- Switching to Cocoa Widget in Mac
- More wanted…
  - Mozilla Labs: http://labs.mozilla.com

Mozilla Products

- Products
  - Mozilla Firefox, Mozilla Thunderbird
- Projects
  - SeaMonkey, Sunbird, Minimo
- Others
  - Bugzilla,

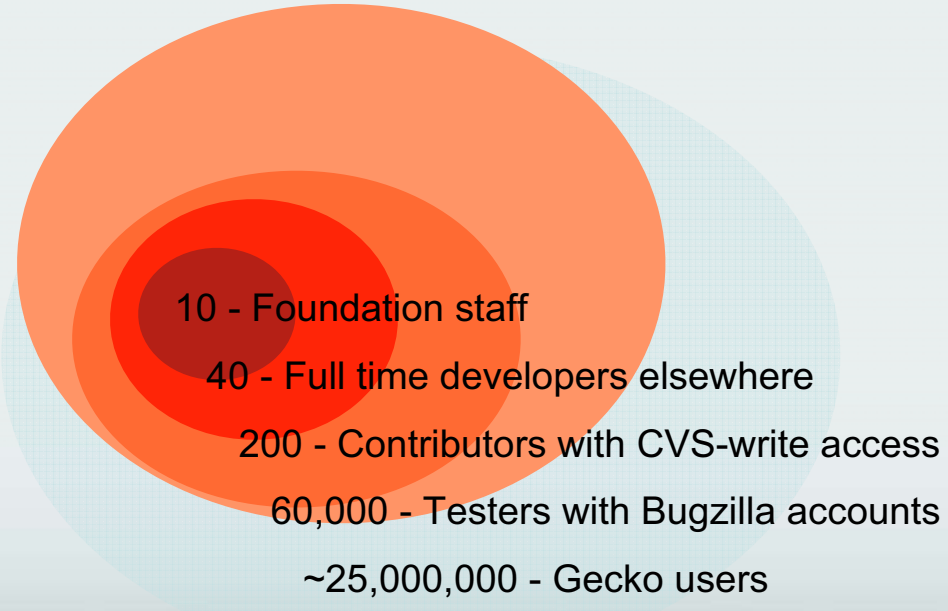
Development Tools

- Language: C++ (gcc, MSVC++, CWP, etc)
- CVS (Source repository)
- LXR (Source code viewer)
- Bonsai (Check-in viewer)
- Tinderbox (Build system)
- Bugzilla (Bug tracking and QA)

Development Products

- Release Products
  - Source Code
  - Binary and Installers: zip,exe/tar.gz/dmg,
  - RPM, Disk images, Ports…
- Release Cycle
  - Alpha(3), Beta(2), Release Candidates(3)
- Nightly Build
- L10N Build

Mozilla Community

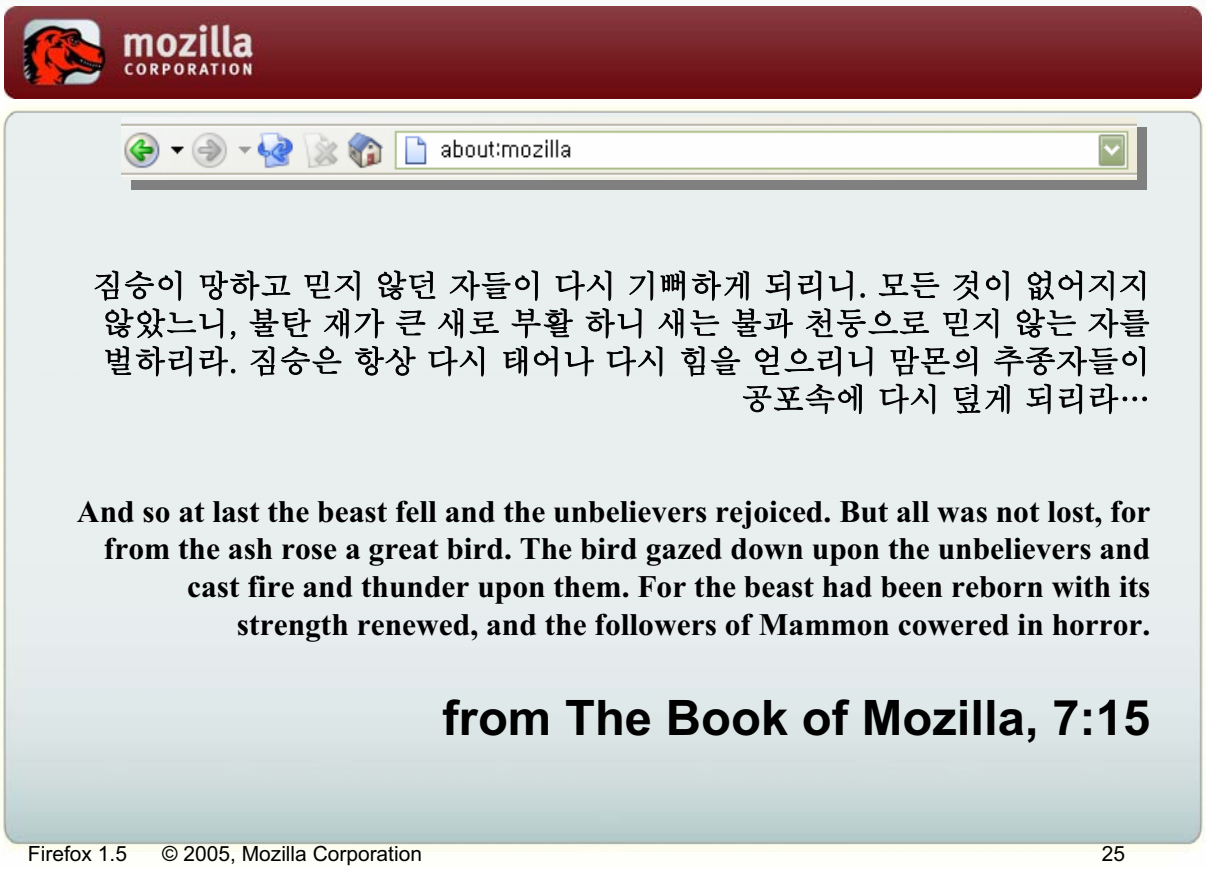


Partners, Customers, Deployments

- IBM, Sun - sponsors, deployment
- Oracle - calendar, certification, deployment
- AOL - Netscape 7.2
- Nokia - Minimo: Mozilla for cell phones
- Red Hat, Lindows - distribution partners
- Macromedia, Apple, Opera - plugin standard
- Fortune 100 customer - 50,000 Thunderbird seats
- GNOME, GTK - XUL-GTK alliance
- W3C - member

Contribution of Mozilla Project

- Joining local open source community
- Bug Reporting for good products
- Beta Testing especially i18n and l10n
- Localization for ko-KR locale
- Translation site management
- Code review, writing and testing
- Submit your code
- CVS committer
- Be major role member (if be, you give up your general job!)



Mini Session

Daum Open Strategy

- Open Standard
- Open Source
- Open API

Open Standard

- 전략: 공개 표준을 이용하여 데이터(UCC) 플랫폼을 형성 함으로서 재 사용 및 글로벌 이식에 용이하게 한다.
  - 웹 표준을 기반한 UI 표준화 (XHTML/CSS)
  - 웹 어플리케이션 개발 강화 (Ajax)
  - RSS FEED 다변화 정책 추진 (RDF)
  - CROSS PLATFORM 기반 기술 계 개발 (Flash)
  - XML 데이터 플랫폼으로 이전 지속 (Semantic Web)

Open Source

- 전략: 오픈 소스를 회사 전반의 개발 플랫폼에 사용하여 핵심 기술 개발의 기반이 되도록 한다.
  - 사내 Open Source 기반 개발 플랫폼 (LAMJI Platform)
    - Linux + Apache + MySQL
    - Java (Tomcat + Struts + Spring)
    - IDE (Eclipse + SubVersion + Trac)
  - 사내 지식(KB)의 외부 전달 (Daum Tech Talk & Tech Note)
  - 사내 소스 외부 오픈 (mod_xss)
  - 외부 오픈 소스 프로젝트 지원 (Mozilla, KTUG, Tattertools)

Open API

- 전략: 데이터 플랫폼 공개를 통해 서드파티를 육성하고 이를 통한 비즈니스 플랫폼 진화를 목표로 한다.
  - Daum 오픈 API 1차 공개: http://dna.daum.net
    - 데이터 플랫폼의 외부 공개를 통한 3rd-Party 개발자 육성
    - 검색 API, 블로그 API, 디앤샵 API
  - 향후 3rd Party와 비지니스 관계 유지 주력
    - 여행 API를 통한 투어 서비스 재판매
    - 카페 API 및 블로그 API를 통한 커뮤니티 서비스 재판매
    - 디앤샵 및 온켓 API를 통한 전자 상거래 파트너쉽

- **D**evelopers **N**etworks and **A**ffiliates

오픈 소스에 대한 두 가지 이야기-커미터와 PMC, 그리고 SoC

NCSoft
이창신
(las)

소개

- 톰켓의 커미터가 되고픈 열정으로 시작
- 아파치 JaxMe 커미터로 마침내 꿈을 이룸
- 아파치 웹서비스 그룹 PMC 멤버
- 구글 SoC 2006 멘터로 참여

- 그나저나 진짜 소속은?

목차

- 1부 – las in Apache
  - 커미터의 꿈
  - 정치 무대
  - 아파치의 과거, 현재, 그리고 미래
- 2부 – las in SoC 2006
  - SoC는 어떻게 돌아가나
  - 학생과의 원격 협업
- 보너스 트랙

커미터는 어떻게 되는가?

- 공식적인 절차는 잘 알려져 있습니다만…
- 아파치 커미터 등극의 새로운 패턴들
  - 근사한 프로젝트를 다른 곳에서 시작한 다음 호평을 받아 아파치로 입성
    예) JaxMe, MyFaces
  - 회사의 제품을 기증하면서 초기 커미터로 잠입
    예) Beehive, XMLBeans

Project Management Committee

- 모든 최상위 수준(top-level) 프로젝트와 프로젝트 그룹(group)에는 PMC가 있어
  - 로드맵과 같은 계획이나
  - 릴리스등의 중요한 의사결정을 민주적인 투표로 진행합니다.
- PMC 멤버는 커미터중에서 선출되며 역시 기존 PMC 멤버의 투표로 결정됩니다.

아파치의 과거

- 일명 프로젝트 스카우팅은 적었고
- (거대) 기업의 참여가 두드러지지 않았고
  - 독특한 사례: IBM은 Axis 1.0 코드를 기증한 후 스스로 포킹(forking)을 뜨며 프로젝트를 이탈
- 전반적으로 관리에 대한 부담을 많이 느끼지 못했었습니다만…

아파치의 현재

- 인큐베이터에 산재한 후보 프로젝트들
  - 35개
  - 정식 프로젝트화에 엄격함
- 너무 많은 최상위 수준 프로젝트들
  - 34개 (그룹 포함)
  - projects.apache.org
- 프로젝트 리팩토링
  - 아파치의 건강에 대한 회의와 대응

아파치의 미래

- 아파치의 가치를 실현하고 전달
  - XML, SOA, Infra, Middleware, 검색…
- 다른 오픈 소스 프로젝트와의 유연한 제휴
  - LGPL
  - CDDL
- 커뮤니티로서의 역할
  - ApacheCon Asia

Google Summer of Code

- 학생들의 오픈 소스 프로젝트 참여를 장려하는 이벤트
- 여름방학동안 학생은 멘터(Mentor)의 지도아래 오픈 소스 프로젝트의 특정 제안을 구현하게 됩니다.
- 미화 총 5000불의 상금중 500불이 멘터측에 전달되며 오픈 소스에 간접적인 지원도 이루어 집니다.

SoC 2006 심사과정

- 당신이 만약 SoC를 지원하려는 학생이라면…
  - 마음에 드는 오픈 소스 재단과 제안 프로젝트를 검토
  - 지원하기 전에 충분한 교감 필요 (왜?)
- 당신이 만약 SoC를 제안하려는 멘터라면…
  - 방학이라는 한정된 기간동안 끝낼 수 있는 아이템을 설정
  - 후보 학생들과 충분한 사전교류 필요 (왜?)

SoC 2006 in Action

- 스리랑카 학생과 협업
- 위키로 보고서 작성
- 주기적(1주일)으로 진행상황 교류
- 부정기적인 문제 토론
  - 이메일과 채팅을 통해 논의
- 성공적인 중간 보고서 제출

SoC 2006 참여후기

- 모든 과정이 웹 애플리케이션으로 진행
  - 구글의 힘? ^^
- 엄청난 양의 질의응답
  - Greg를 비롯한 SoC 팀에게 감사
- 오픈 소스 단체와 학생들의 뜨거운 열기
  - 작년보다 양적·질적으로 모두 향상
- 그렇다면 한국은?

Bonus Track

- To be unveiled...

결론

- 개인의 오픈 소스에서 집단의 오픈 소스로
- 프로의 오픈 소스에서 학생의 오픈 소스로
- 지금 오픈 소스에 익숙한 학생들이 사회에 진출해서 오픈 소스에 어떤 기회를 줄지를 생각한다면, 우리에게 희망은 무한합니다.

오픈소스 프로젝트에 참여하기

▸ 장혜식 · FreeBSD, Python

오픈소스 프로젝트는 누구나 참여할 수 있도록 열려있다. 오랜 시간을 거치며 발전해온 오픈소스계의 여러가지 관례와 프로젝트 안에서 좀 더 의사소통 효율을 높이기 위한 여러 방법을 알면 좀 더 효과적이고 재미있게 오픈소스 프로젝트에 참가할 수 있다. 프로젝트에 기여하는 것은 코드를 작성하는 것 뿐만 아니라, 문서화, 번역, 재정적 기여 등 매우 다양한 방법이 있다. 내가 쓰는 뭔가를 좀 더 좋게 만들어 보자!

오픈소스 프로젝트에는 왜 참여하는가?

- ❖ 오픈소스 소프트웨어를 왜 쓰는가?
- ❖ 프로그램을 쓰다가 불편해서
- ❖ 좀 더 다양한 경험을 위해서
- ❖ 다양한 사람들과 친해지고 싶어서
- ❖ 회사에서 일로 하는 과정 중
- ❖ 유명해 지고 싶어서

오픈소스 프로젝트의 규모별 분류

- ❖ 재단이 있는 대형 프로젝트
- ❖ 개발자 리소스가 제공되는 중형 프로젝트
- ❖ 1인 프로젝트 또는 소형 프로젝트
- ❖ 기업에서 지원하는 프로젝트

프로젝트에 기여하는 여러 방법

- ❖ 소스코드 기여
- ❖ 문서화, 번역물
- ❖ 네트워크, 하드웨어 리소스
- ❖ 재정적 기여
- ❖ 기타 사회적 기여

작은 프로젝트에 참여하기!

- ❖ 패치 만들기 / 제출하기
- ❖ 설득하기
- ❖ 오픈소스에서의 메일 교환

중형 프로젝트에 참여하기

- ❖ 메일링 리스트에 참여하기
- ❖ 중형 프로젝트에서의 패치 작업
- ❖ 프로젝트 커뮤니티에서의 정치적 활동
- ❖ 참여가 목적일 때 관심사 찾기, 주목 받기
- ❖ 법적 문제 해결방법
- ❖ 버그를 보고하는 방법
- ❖ 관례 파악하기
- ❖ 중형 프로젝트에서 제공되는 개발자 리소스

대규모 프로젝트에 참여하기

- ❖ 망보기
- ❖ 대규모 프로젝트의 여러 특징
- ❖ 여러가지 트래커의 특징과 문화
- ❖ 친교적 활동과 개발자 그룹간 교류
- ❖ 의도적으로 참여하는 경우
- ❖ 패치를 홍보하는 여러가지 방법
- ❖ 대규모 프로젝트에서 제공되는 개발자 리소스
- ❖ 컨퍼런스 활동과 재단 활동

오픈소스 프로젝트들의 여러가지 관례

- ❖ 코딩 스타일
- ❖ 커뮤니케이션
- ❖ 빌드 구조, 의존성
- ❖ 법적인 문제와 사회적인 관례
- ❖ 다양성

❖개인 프로젝트 시작하기

❖소스트리 구성하기
❖프로젝트 초기 전략
❖홍보하기, 사람들의 관심을 끌어내기
❖성공적인 프로젝트를 위한 요소

단기적으로 참여할 수 있는 방법

❖구글 Summer of Code
❖상금 사냥꾼!
❖Junior Hackers' List

오픈소스는 언제 할 것인가?

❖쏟아지는 메일 관리하기
❖쏟아지는 패치 관리하기
❖메일링 리스트 유지
❖시간의 배분
❖관심사 유지하기
❖휴식과 오픈소스
❖동기 재생산

Business Track

제품개발시 고려해야 할 오픈소스 이슈

(권순선)

오픈소스와 회사 개발 프로세스

(최호진)

권순선 KSS@KLDP.ORG

Incorporating Free/Open Source Software Code into Proprietary Code

오픈소스 소프트웨어 코드를 상용 제품이나 서비스에 구현할 경우에 고려하여야 할 사항들에 대해서 논의합니다. 아래는 전체 강의에 대한 개요이며 각각의 주제에 대한 상세 내용은 강의를 통해서 전달됩니다.

본 강의에서 제공되는 내용들은 어디까지나 개인적인 의견이며 강사가 속한 단체나 회사의 의견을 대변하지 않습니다.

OPEN SOURCE DEFINITION

1. 자유로운 재배포
2. 소스 코드
3. 파생 저작물
4. 원시 코드 원형 유지
5. 개인 및 단체에 대한 차별 금지
6. 사용 분야에 대한 차별 금지
7. 라이센스 배포
8. 특정 제품에 대한 차별 금지
9. 다른 소프트웨어를 제한하는 사용 허가 금지
10. 라이센스 조항은 기술 중립적

Free Software와 Open Source Software는 실제로는 비슷하지만 엄밀한 의미에서는 Open Source가 좀더 범위가 넓습니다.

소프트웨어의 사용 범위와 방법에 대한 허가는 라이센스를 통해 이루어지는데 오픈소스도 마찬가지로 지켜야 할 의무사항들이 분명 존재합니다.

제품이나 서비스를 개발하는 기업에서 오픈소스 라이센스를 지켜야 하는 이유는 자사가 보유한 지적재산(IP)를 보호하기 위한 것 뿐만 아니라 타인의 IP를 침해하지 않기 위해서입니다.

그런 의미에서 오픈소스라고 하는 것도 모두 라이센스 조항이 정해져 있고 그 종류가 매우 많습니다. (GPL, LGPL, BSD, MPL, Apache, MIT, X...)

여러 다양한 오픈소스 라이센스를 문제없이 사용하기 위해 지켜야 할 주의 사항들은 크게 다음 5가지로 나뉠 수 있습니다.

- 소스코드 공개
- 사용 여부 명시
- 특허권 관련 처리
- Trademark
- 저작권 보호(comment 삭제 등)

오픈소스 라이센스는 매우 다양하기 때문에 가장 많이 사용되는 GPL과 LGPL을 중점적으로 살펴보겠습니다.

GPL은 현재 가장 많이 사용되는 오픈소스 라이센스이면서 가장 강력한 조건을 가지고 있습니다.

GPL에서는 전염성이 있어 GPL코드의 파생물은 모두 GPL이 되어야 하기 때문에 잘못할 경우 원하지 않는 코드가 GPL로 되어야 하는 경우가 생길 수 있습니다.

반면 LGPL은 라이브러리를 위한 GPL이라고 보시면 되고, GPL에 비해 LGPL 코드는 링크한다고 하더라도 링크한 자체 Application의 소스코드는 공개하지 않아도 된다는 것이 가장 큰 차이점입니다.

커널 모듈의 라이센스는 오랫동안 명확히 정해지지 않았는데 최근에는 주로 커널 모듈도 GPL이다 라는 식의 의견들이 많이 확대되는 추세입니다. 그렇다면 제품의 경쟁력과 차별성은 어떻게 확보할 수 있을까요?

GPL은 회피할 수 있을까요? GPL코드를 사용할 때 주의해야 할 내용들은 어떤 것이 있을까요?

감사합니다. 궁금한 점이 있으시면 질문해 주세요.

오픈 소스와 개발 프로세스

2006-09-17

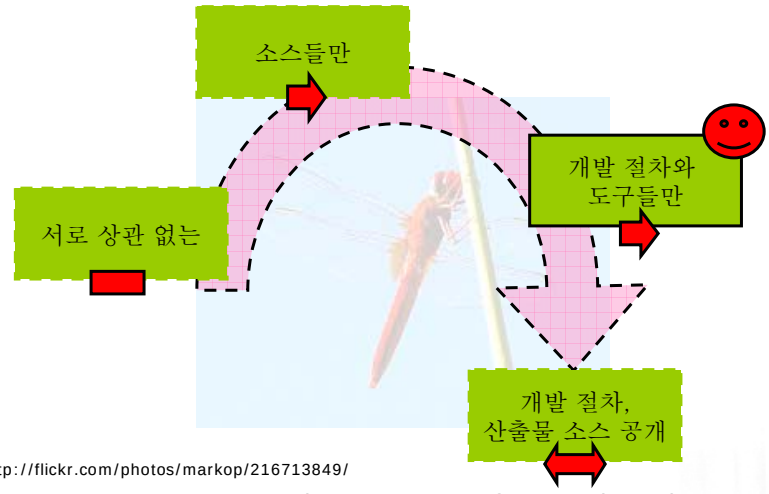
최 호 진
안철수 연구소

hojin.choi@gmail.com
http://coolengineer.com/

목 차

지금은
주변 이야기 중입니다

오픈 소스와 회사



http://flickr.com/photos/markop/216713849/

오픈의 핵심 1

- 소프트웨어 소스 열람 가능 (➔)
- 개발자와 직접 대화 가능 (➔➔)
- 코드에 기여 (➔)

- 열린 개발 프로세스
- 소프트웨어의 철학,방향을 쉽고 빨리 알 수 있음

오픈의 핵심 2 – 보기

- 소스 보기
- 아이디어 보기

- 개발 절차 보기
- 개발 도구 보기
- 실수한 기록(과거) 보기
- 버그 보기

- ~~-개발자- 얼굴- 보기~~ (???)

http://flickr.com/photos/archiprez/116985527/

개발의 단계들

http://www.cs.queensu.ca/Software-Engineering/tools.html

요구분석	S/W 설계	버전 관리
벤치 마킹	데이터 베이스 설계	산출문서 관리
다국어 지원	사용자 인터페이스	문서 자동화
테스트 계획	버그 트래킹	테스트 관리
리버스 엔지니어링	빌드 자동화	릴리즈 관리

밥,음료수,잠

각 단계에에 맞는 툴들이 많이 존재함

http://flickr.com/photos/jocularme/141535120/

도구 1: 실수의 최소화

- 스크립트 키드
  - 깊은 지식 없이 도구만 쓰는 해커
  - → 실수 없이 해킹(?)
- Makefile, 배치 파일, 쉘 스크립트
  - 일련의 빌드 명령을 순서대로 수행
  - → 반복되는 명령 수행
- 복잡한 명령을 도구로 수행
  - → 문제를 한 발짝 떨어져서 보게 함
- **명필은 붓을 가리지 않는다.**
  - → 명필은 어떠한 붓이든 이해한다.

(억지일지 모르지만…)

→ 훌륭한 개발자는 도구의 필요성을 이해한다.
→ 도구의 필요성을 이해하고 그 내부를 이해한다.

Open Source 툴로 도구를 원리를 알 수 있음

http://flickr.com/photos/68373439@N00/17731554/

도구 2: 제품과 공장 (자동화)

도구의 궁극의 목표는 자동화

지향점
(수작업 30%, 자동화 70%)

자동차?

자동차 공장?

가내 수공업으로
차를 만들 것이냐

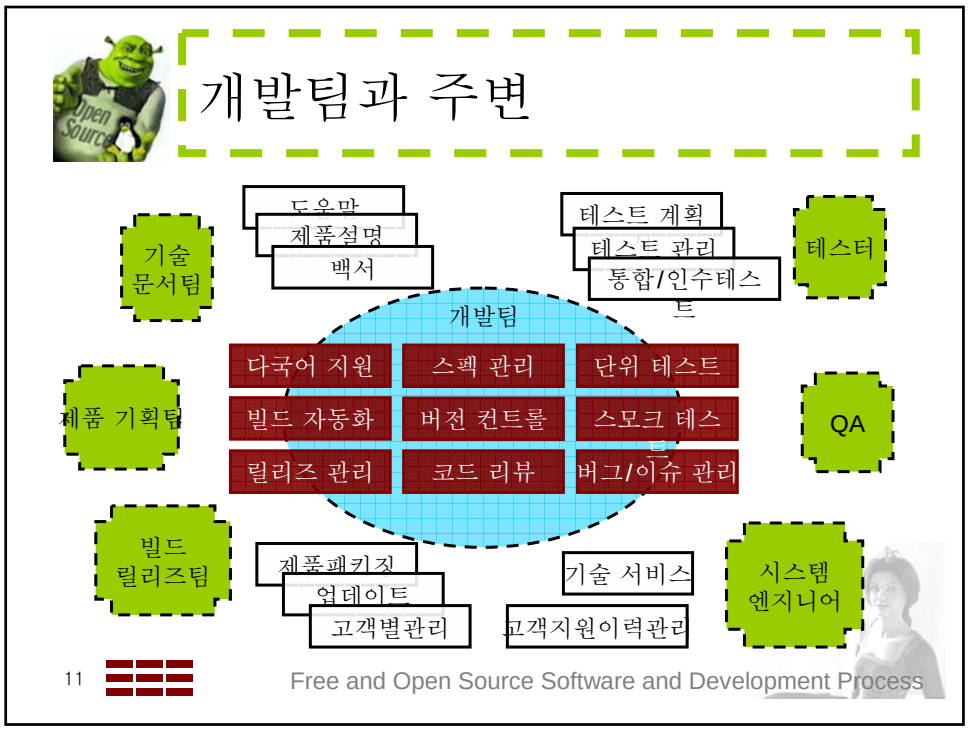
차 만드는 공장을
만들 것이냐

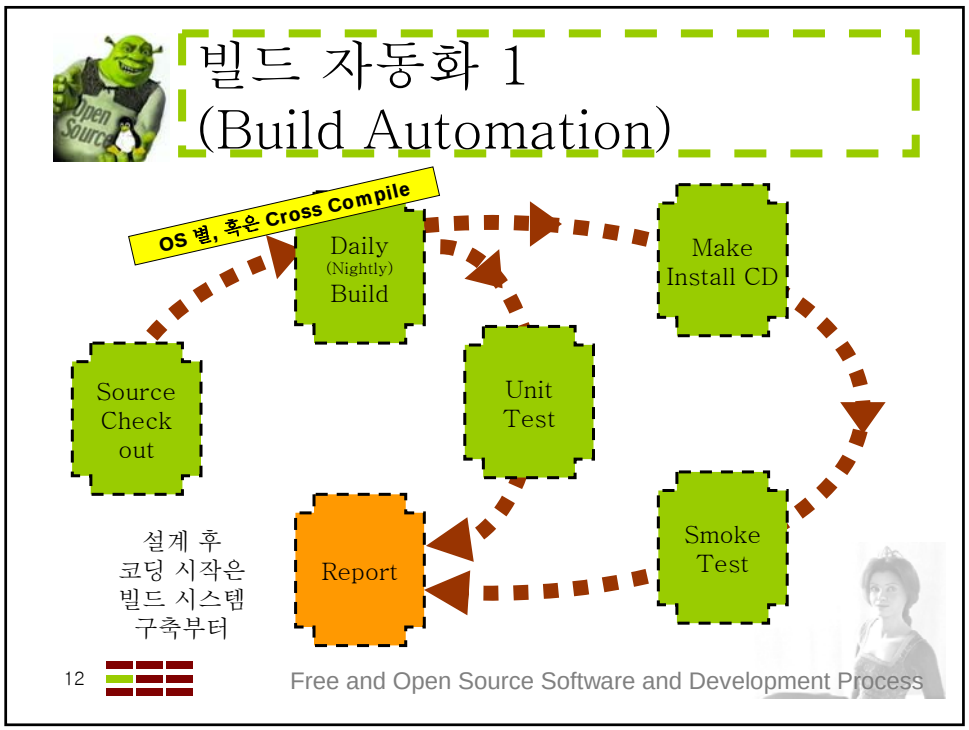
http://www.econotizie.it/ambiente/State%20of%20the%20world.htm

지금은
개발 중입니다

관리자의 관점보다 개발자의 관점에서

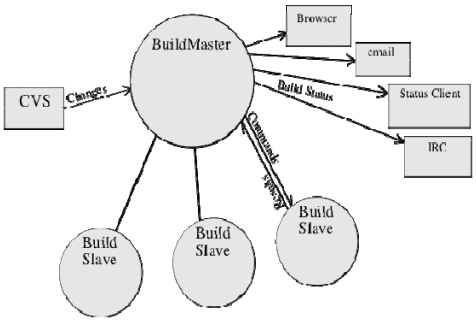
소프트웨어 단계를 살펴봅니다.


개발팀과 주변
기술 문서팀
도움말
제품설명
백서
테스트 계획
테스트 관리
통합/인수테스트
테스터
개발팀
다국어 지원
스펙 관리
단위 테스트
빌드 자동화
버전 컨트롤
스모크 테스
릴리즈 관리
코드 리뷰
버그/이슈 관리
제품 기획팀
QA
빌드 릴리즈팀
제품패키징
업데이트
고객별관리
기술 서비스
고객지원이력관리
시스템 엔지니어
11
Free and Open Source Software and Development Process


빌드 자동화 1
(Build Automation)
OS 별, 혹은 Cross Compile
Daily
(Nightly)
Build
Make
Install CD
Source
Check
out
Unit
Test
Smoke
Test
Report
설계 후
코딩 시작은
빌드 시스템
구축부터
12
Free and Open Source Software and Development Process

빌드 자동화 2 (Build Automation)

- http://tinderbox.mozilla.org/
  모질라 프로젝트에서 사용하는 빌드 로봇
- http://buildbot.sourceforge.net/
  Zope, Python, KDE 등의 프로젝트에서 채용하고 있는 분산 빌드/테스트 로봇



스모크 테스트 1 (Smoke Test)

◆ 주기적 혹은 커밋이 일어날 때, 빌드/테스트 결과를 메일로...

buildbot@mobsol.be	svn trunk r21118: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-18
buildbot@mobsol.be	svn trunk r21117: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-18
buildbot@mobsol.be	svn trunk r21114: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-18
buildbot@mobsol.be	svn trunk r21113: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-18
buildbot@mobsol.be	svn trunk r21113: FAIL (win32-xp VS2005)	2006-08-18
buildbot@mobsol.be	svn trunk r21112: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-18
buildbot@mobsol.be	svn trunk r21112: FAIL (win32-xp VS2005)	2006-08-18
buildbot@mobsol.be	svn trunk r21111: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-17
buildbot@mobsol.be	svn trunk r21111: PASS (win32-xp VS2005)	2006-08-17
buildbot@mobsol.be	svn trunk r21109: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-17
buildbot@mobsol.be	svn trunk r21109: PASS (win32-xp VS2005)	2006-08-17
buildbot@mobsol.be	svn trunk r21106: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-17
buildbot@mobsol.be	svn trunk r21106: PASS (win32-xp VS2005)	2006-08-17
buildbot@mobsol.be	svn trunk r21103: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-17
buildbot@mobsol.be	svn trunk r21106: PASS (x86-macosx-gnu shared ra_local fsfs)	2006-08-17
buildbot@mobsol.be	svn trunk r21105: PASS (win32-xp VS2005)	2006-08-17
buildbot@mobsol.be	svn trunk r21103: PASS (x86-macosx-gnu shared ra_local fsfs)	2006-08-17
buildbot@mobsol.be	svn trunk r21103: FAIL (win32-xp VS2005)	2006-08-17
buildbot@mobsol.be	svn trunk r21101: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-16
buildbot@mobsol.be	svn trunk r21101: PASS (win32-xp VS2005)	2006-08-16
buildbot@mobsol.be	svn trunk r21101: PASS (x86-macosx-gnu shared ra_local fsfs)	2006-08-16
buildbot@mobsol.be	svn trunk r21097: PASS (i686-debian-sarge1 shared gcc-3.3.5)	2006-08-16
buildbot@mobsol.be	svn trunk r21097: PASS (win32-xp VS2005)	2006-08-16

제목 svn trunk r21112: FAIL (win32-xp VS2005)

표준 보기 메시지 정보 데이터 출력

Date: Fri, 18 Aug 2006 09:00:39 +0200
From: buildbot@mobsol.be <buildbot@mobsol.be>
Subject: svn trunk r21112: FAIL (win32-xp VS2005)

Full details are available at:
http://www.mobsol.be/buildbot/win32-xp%2520VS2005/builds/398

Author list: cmpilato

Build Slave: djh-xp-vse2005

Subversion Buildbot
http://www.mobsol.be/buildbot/

http://subversion.tigris.org/servlets/SummarizeList?listName=svn-breakage

스모크 테스트 2 (Smoke Test)

◆ http://gcc.gnu.org/ml/gcc-testresults/

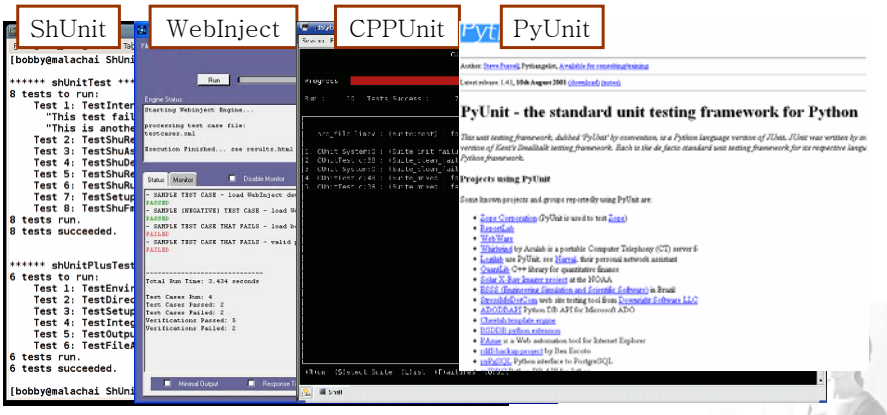
August 25, 2006

23:29	Results for 4.2.0 20060824 (experimental) testsuite on i686-pc-cygwin	David Billinghurst
22:59	Results for 4.2.0 20060825 (experimental) testsuite on x86_64-unknown-netbsd3.0	Krister Walfridsson
22:59	Results for 4.2.0 20060825 (experimental) testsuite on i386-unknown-netbsdelf3.0	Krister Walfridsson
22:52	Results for 4.1.2 20060825 (prerelease) testsuite on i686-pc-linux-gnu	CodeSourcery Automatic Testing
22:00	Results for 4.2.0 20060825 (experimental) testsuite on powerpc-apple-darwin8.5.0	regress
21:57	Results for 4.0.4 20060825 (prerelease) testsuite on s390-ibm-linux-gnu	Andreas Krebbel
21:04	Results for 4.2.0 20060825 (experimental) testsuite on hppa64-hp-hpux11.11	CodeSourcery Automatic Testing
20:10	Results for 4.0.4 20060825 (prerelease) testsuite on s390x-ibm-linux-gnu	Andreas Krebbel
19:15	Results for 4.2.0 20060825 (experimental) testsuite on hppa2.0w-hp-hpux11.11	CodeSourcery Automatic Testing
18:55	Results for 4.2.0 20060823 (experimental) testsuite on s390-ibm-linux-gnu (Debian)	Martin Michlmayr
18:54	Results for 4.2.0 20060823 (experimental) testsuite on mipsel-unknown-linux-gnu (Debian)	Martin Michlmayr
18:50	Results for 4.2.0 20060823 (experimental) testsuite on mips-unknown-linux-gnu (Debian)	Martin Michlmayr
17:20	Results for 4.2.0 20060825 (experimental) testsuite on arm-none-eabi	CodeSourcery Automatic Testing

단위 테스트 (Unit Test)

http://www.xprogramming.com/software.htm

ShUnit WebInject CPPUnit PyUnit



코딩규칙: Code Beutifier

- GNU Indent를 사용하는 프로젝트들
  http://www.google.com/search?q=.indent.pro
  (indent의 환경설정 파일)
- 커밋하기 전 코드 포맷을 맞춤
- 프로젝트에서 선택한 코딩 가이드라인의 일부를 자동화
- 수작업으로 확인
- 코드리뷰에 도움

•Postfix (Mail server)

```
[pynoos@shire ~/work/postfix-2.2-20050214]
$ ls -al
total 836
drwxr-xr-x  15 pynoos pynoos   4096 Sep  2 12:57 ./
drwxrwxr-x  74 pynoos pynoos   4096 Aug 28 15:23 ../
-rw-r--r--   1 pynoos pynoos   3147 Feb  2  2005 .indent.pro
-rw-r--r--   1 pynoos pynoos    418 Sep 21  2000 .printfck
-rw-r--r--   1 pynoos pynoos   5691 Oct 26  2004 AAAREADME
-rw-r--r--   1 pynoos pynoos   2952 Feb 12  2005 COMPATIBILITY
-rw-r--r--   1 pynoos pynoos   1922 Feb 23  2002 COPYRIGHT
-rw-r--r--   1 pynoos pynoos 362345 Feb 14  2005 HISTORY
lrwxrwxrwx   1 pynoos pynoos     20 Mar 25  2005 INSTALL -> READM
-rw-r--r--   1 pynoos pynoos  17819 Jan 19  2005 IPv6-ChangeLog
-r--r--r--   1 pynoos pynoos  11942 Jul 16  1999 LICENSE
-rw-rw-r--   1 pynoos pynoos   3016 Feb 15  2005 Makefile
-rw-r--r--   1 pynoos pynoos   2590 Feb  3  2005 Makefile.in
-rw-r--r--   1 pynoos pynoos    479 Jan 15  2002 Makefile.init
-rw-r--r--   1 pynoos pynoos   1193 Mar 14  2004 PORTING
drwxr-xr-x   2 pynoos pynoos   4096 Feb 15  2005 README_FILES/
-rw-r--r--   1 pynoos pynoos  21216 Feb 15  2005 RELEASE_NOTES
```


버전 관리 (Version Control)

http://en.wikipedia.org/wiki/Comparison_of_revision_control_software
http://better-scm.berlios.de/comparison/comparison.html

- 커밋 로그의 내용
  → 버그 번호, 파일별 내용, 검토자 리스트
- 1인 이상 검토후 커밋
- 수정된 파일들 중 연관된 파일끼리 모아서 커밋
- 커밋전에 Diff 하여 변경 내역을 최종 확인
- Merge를 통한 브랜치와 트렁크간의 병합
- 커밋과 동시에 메일로 통지
  예) python-checkins@python.org
  svn@subversion.tigris.org
  commits@tortoisesvn.tigris.org

버전 관리: Peer Review

- “하면 좋은 것”이 아니라 “해야 하는 것”
- 코드리뷰의 목적
  - → 조기에 문제 발견
  - → 코드 공유
  - → 문제 발생을 유연하게 대처
- 커밋하기 전 (추천)
- 커밋한 뒤 (리뷰가 제대로 되지 않음)

- 리뷰자를 커밋 메시지에 일정한 규칙으로 삽입
  - ➔ 버전 컨트롤러는 커밋 로그 규칙을 검사 후 수행

버전 관리: 예 1

- 수정한 것들 중 연관 있는 것만 모아 커밋

Jump to revision: 21101 Go

Author: sussman

Date: Thu Aug 17 03:07:14 2006 UTC *(11 days, 13 hours ago)*

Log Message:

```
Teach 'svn revert' to understand the --changelist option.

* subversion/svn/main.c
  (svn_cl__cmd_table):  allow 'revert' to take svn_cl__changelist_opt.

* subversion/svn/revert-cmd.c
  (svn_cl__revert):  if --changelist is given, discover changelist
                     paths and act on them as if user had typed them.
```

Changed paths:

Path
trunk/subversion/svn/main.c
trunk/subversion/svn/revert-cmd.c

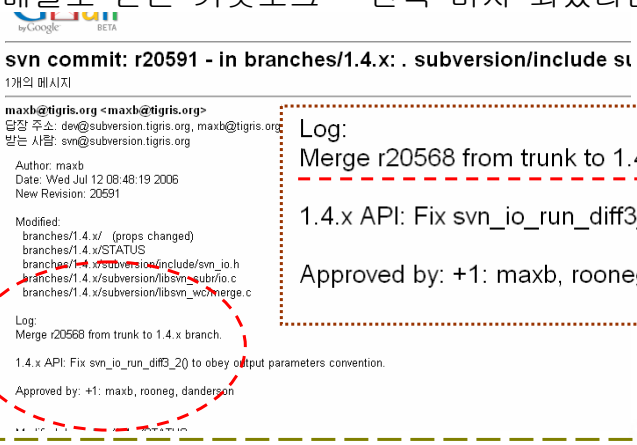
버전 관리: 예 2

◆ 메일로 받는 커밋로그 - 라인 별 Diff 와 함께

우린 Unified Diff에 익숙해질 필요가 있습니다.

버전 관리: 예 3

◆ 메일로 받는 커밋로그 - 간혹 머지 되었다는 로그



Log:
Merge r20568 from trunk to 1.4.x branch.

1.4.x API: Fix svn_io_run_diff3_2() to obey out

Approved by: +1: maxb, rooneg, danderson

버전 관리: 브랜치와 병합

- 브랜치는 언제 하나?
  → 버전이 올라갈 때 (즉, 릴리즈가까이)
  → 실험적인 패치를 진행할 때

- 병합(Merge)?
  → 브랜치된 어느 한 쪽이 수정되면 다른 쪽에 반영

바오밥나무 http://flickr.com/photos/kumasawa/215926212/

- 병합시 주의점
  → 브랜치간 비교
  → Id 와 같이 커밋된 다음 변경되는 것은 무시

코드 변경 (Diff/Patch)

- 기능 개선 ➔ 원본 + 패치파일 형태로 보관
- 패치 파일은 diff 결과로 만들어짐
- context diff, unified diff, ed-style … 포맷

- 패치는 대개 unified diff 를 선호 ➔ 익숙 필요
- diff 를 사용한다는 것 ➔ 행 단위 변경 관리
- annotate 기능을 이용한 변경관리
  ➔ 문제가 생겼을 경우
  해당 라인이 어떤 리비전에서 들어 왔는지 확인

변경 추적 (Annotate)

◆ 행 단위 저자와 리비전을 볼 수 있음

Annotation of /trunk/subversion/libsvn_fs/fs

Parent Directory | Revision Log

20 :	ghudson	9550	#include <string.h>
21 :			#include <apr.h>
22 :			#include <apr_hash.h>
23 :	ghudson	14066	#include <apr_thread_mutex.h>
24 :	ghudson	9550	
25 :	ghudson	9501	#include "svn_types.h"
26 :	rooneg	20996	#include "svn_dso.h"
27 :	ghudson	9550	#include "svn_version.h"
28 :	ghudson	9489	#include "svn_fs.h"
29 :	ghudson	9499	#include "svn_path.h"
30 :	sussman	13571	#include "svn_xml.h"
31 :	maxb	13761	#include "svn_pools.h"
32 :			#include "svn_string.h"
33 :	ghudson	9489	#include "svn_private_config.h"
34 :			
35 :	ghudson	9556	#include "fs-loader.h"

버그, 이슈 관리 1

◆ 버전 관리 시스템과 연동

커밋: 메시지안에 버그 번호 삽입

버그 노트: 로그에 커밋 리비전이 남음

◆ 버그 픽스 계획

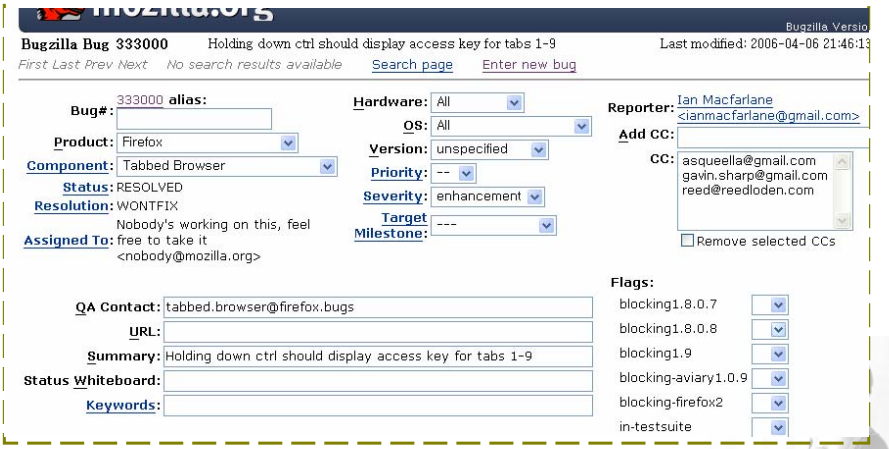
사용자들이 안심, 사용자들이 기다림, 중요성에 대한 공유

◆ 이슈 트래킹에 대한 공유

다음 버전에 들어갈 내용에 대한 정리

버그, 이슈 관리 2

◆ (https://bugzilla.mozilla.org/enter_bug.cgi?product=Firefox)



버그, 이슈 관리 3

◆ 재현 과정을 자세히 적을 것을 요구
(https://bugzilla.mozilla.org/enter_bug.cgi?product=Firefox)
A sentence which summarises the problem.
Please be descriptive and use lots of keywords.

Bad example: mail crashed
Good example:
crash if I close the mail window while checking for new POP mail

◆ 재현 과정을 정확히 적는 것처럼 해결 과정도
가능한 정보를 모두 기재

➔ Commit revision
➔ Fixed version
➔ Source 파일
➔ 근본적인 원인

http://en.wikipedia.org/wiki/Gecko

문서 자동 생성

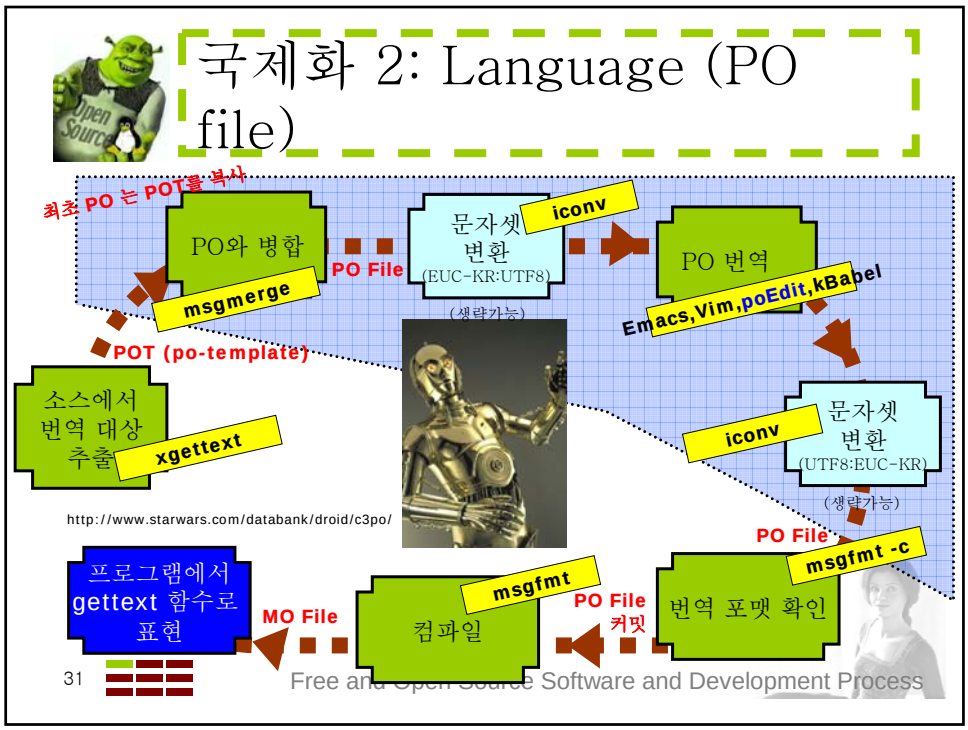
◆ 말들…

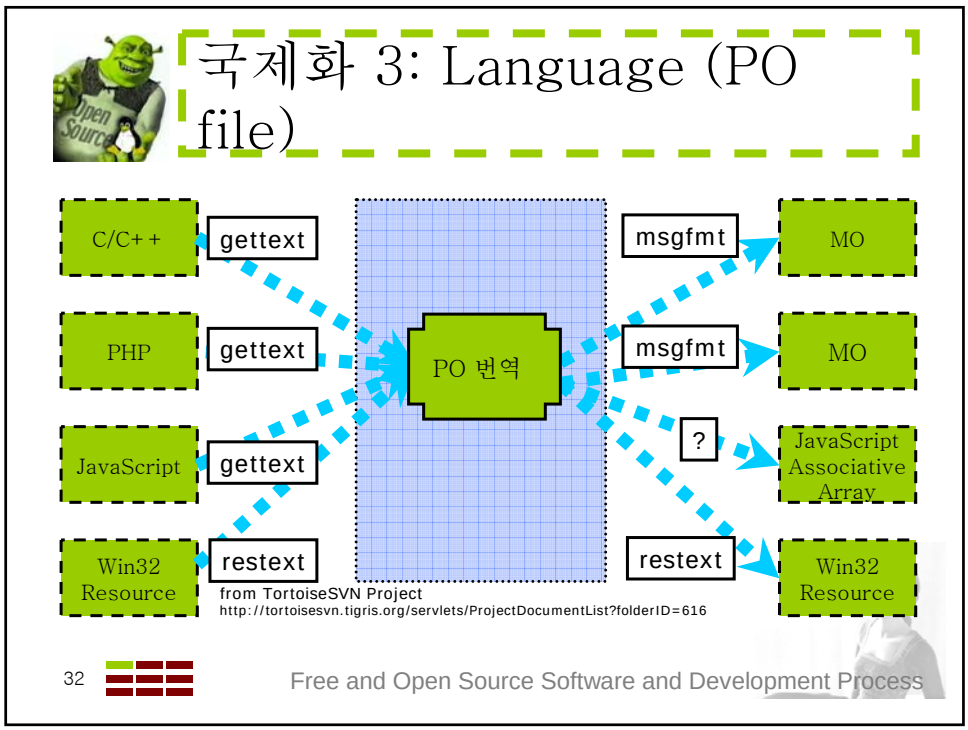
- 주석으로부터 문서를 생성
- 제출용? 관리용?
- 코드 분석용으로 사용
- 자동 생성되도록 문서를 만들 수는 없다.
- 주석 규칙을 통일하기 위한 속셈(?)
- 커다란 윤곽을 잡기 위한 것.
- 주석이 자세할 수만은 없다.
- 코드가 주석이 되어야 한다.
- Doxygen?

국제화 1: (Internationalization; I18n)

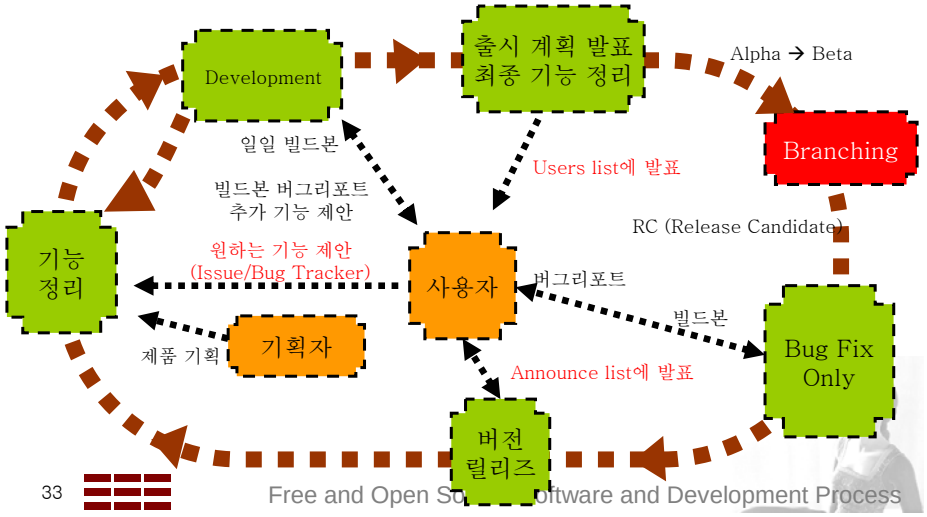
◆국제화의 여러 요소

- Language
- Date/time format
- Time zones
- Currency
- Images and colors
- Names and titles
- Social Security numbers and passports
- Telephone numbers, addresses postal codes
- Weights and measures
- Special support for East Asian languages
- Local customs,
- Local content
- Symbols
- Order of sorting
- Aesthetics
- Cultural values and social context
- Paper Size


국제화 2: Language (PO file)
최초 PO 는 POT를 복사
PO와 병합
msgmerge
PO File
문자셋 변환
(EUC-KR:UTF8)
(생략가능)
iconv
PO 번역
Emacs,Vim,poEdit,kBabel
POT (po-template)
소스에서 번역 대상 추출
xgettext
http://www.starwars.com/databank/droid/c3po/
iconv
문자셋 변환
(UTF8:EUC-KR)
(생략가능)
PO File
msgfmt -c
프로그램에서 gettext 함수로 표현
MO File
컴파일
msgfmt
PO File
커밋
번역 포맷 확인
31
Free and Open Source Software and Development Process


국제화 3: Language (PO file)
C/C++
gettext
PHP
gettext
JavaScript
gettext
Win32 Resource
restext
PO 번역
msgfmt
MO
msgfmt
MO
?
JavaScript Associative Array
restext
Win32 Resource
from TortoiseSVN Project
http://tortoisesvn.tigris.org/servlets/ProjectDocumentList?folderID=616
32
Free and Open Source Software and Development Process

출시관리 1 (Release Mngmnt.)



출시관리 2 (Release Mngmnt.)

- **릴리즈 이후 보관해야할 것**
- 릴리즈 바이너리의 MD5 값
- 디버그 심볼이 제거 되지 않은 바이너리
  (-g로 빌드후 strip 명령 이전 상태)
- 공유 라이브러리 의존성 리스트 (ldd)
- 바이너리의 심볼 리스트 (nm)
- 소스에 "Id"를 삽입하여 개발
- 바이너리만 가지고 버전을 알 수 있음

맺으며…

- 오픈 소스 소프트웨어의 운영 방식은 다양하다.
- 주로 이용되는 방식이 있다.
- 회사 개발에 이용할 수 있는 이점이 많다.
- 직접 참여하하면서 배우는 것이 가장 효과적이다.

- 더 많은 회사들이 오픈 소스에서 적절한 프로세스를 발견할 수 있기를…

감사합니다!

http://flickr.com/photos/good_day/159927879/

사용권 증서

- 본 문서는 아래 기술한 Creative Commons License에 따라 이용할 수 있습니다.

- **저작자표시**. 귀하는 원저작자를 표시하여야 합니다.
- **비영리**. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.
- **동일조건변경허락**. 귀하가 이 저작물을 개작, 변형 또는 가공했을 경우에는, 이 저작물과 동일한 이용허락조건하에서만 배포할 수 있습니다.
- http://creativecommons.org/licenses/by-nc-sa/2.0/kr/
- 여기에 사용된 그림들은 flickr에서 CC로 된 것들과 일부에 대해서는 공식 홈페이지에서 인용하였습니다.

creative commons
C O M M O N S D E E D