POSTECH

CSED 312:

Operating System Lab

Introduction

Autumn 2023

TA Contact Information

Name	Email	Phone	Office
박재준 (Jaejun Park)	jjpark17@postech.ac.kr	010-2043-7158	B4 115 Interaction Lab
강덕형 (Deokhyung Kang)	deokhk@postech.ac.kr	010-2126-4700	PIAI 322 NLP Lab
오민현 (Minhyeon Oh)	minhyeonoh@postech.ac.kr	010-6659-7300	RIST B4 4309 ML Lab
유동현 (Donghyun Yu)	donghyeonryu@postech.ac.kr	010-9520-9516	PIAI 423 System Software Lab

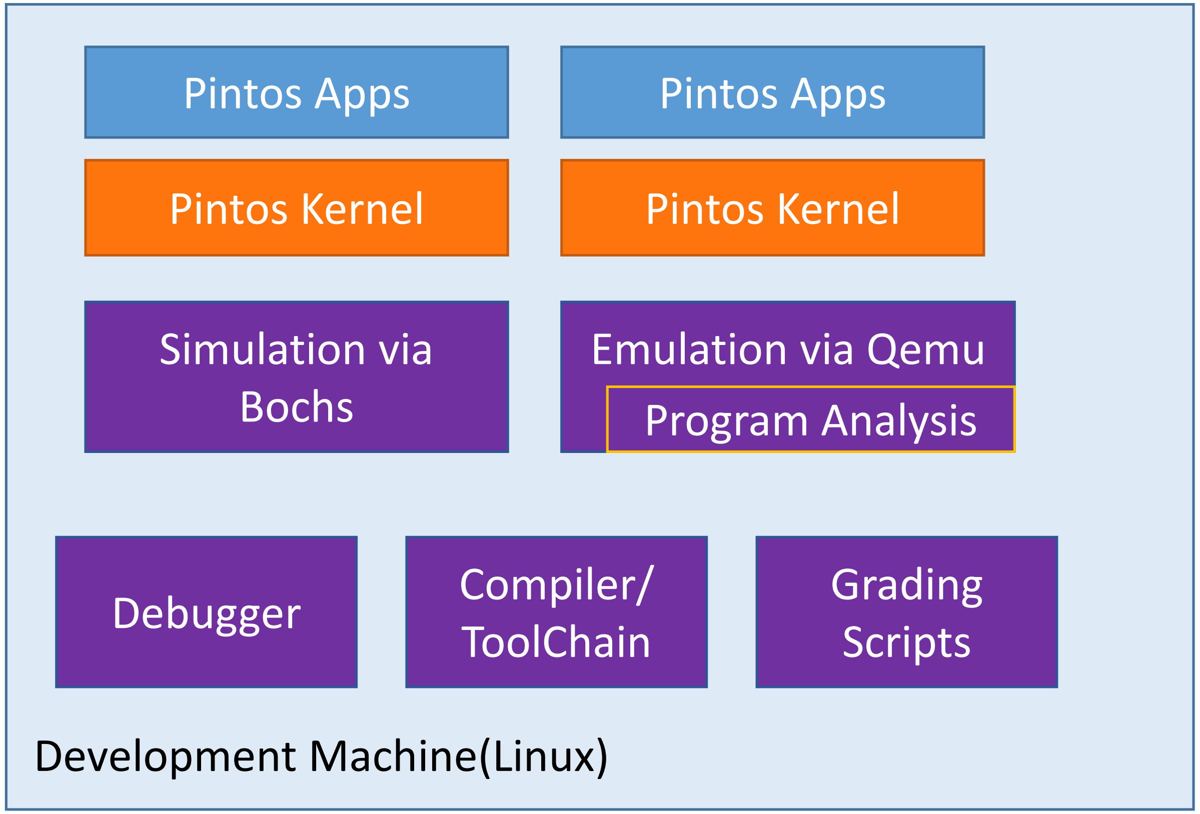
- Announcement: **PLMS**
- Q&A board: **PLMS Question Board**
- Contact TAs if you have any questions or problems

What is Pintos?

- **Pintos Project**
  - Simple operating system framework for the 80x86 architecture
  - Written by Ben Pfaff (**Stanford University**)
  - Introduces students to the principles of multi-programming, scheduling, virtual memory, and file systems.
  - Website : https://web.stanford.edu/class/cs140/projects/
- Pintos supports **kernel threads**, loading and running **user programs**, and a **file system** in a very simple way
- During project 1~3, you will **improve them**, and also add a **virtual memory** implementation

Reference : Paper(The Pintos Instructional Operating System Kernel)

Pintos Development Environment



Pintos Apps

Pintos Apps

Pintos Kernel

Pintos Kernel

Simulation via Bochs

Emulation via Qemu

Program Analysis

Debugger

Compiler/ ToolChain

Grading Scripts

Development Machine(Linux)

Project Overview

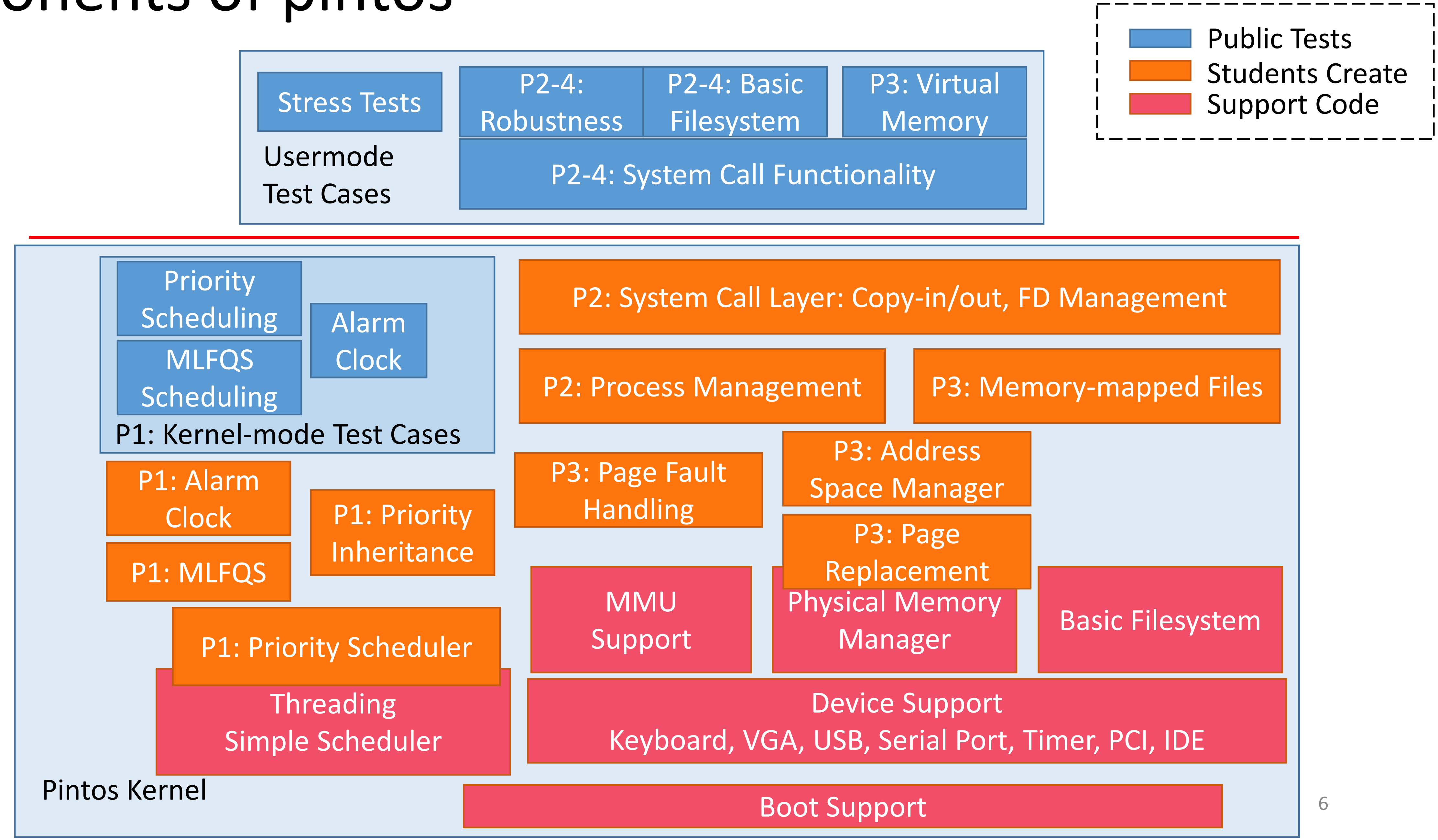
- **Project 1: Threads (TA: 오민현 / 유동현)**
  - Make a more efficient but well synchronized thread scheduler

- **Project 2: User Programs (TA: 강덕형 / 박재준)**
  - Enable programs to interact with the OS via system calls

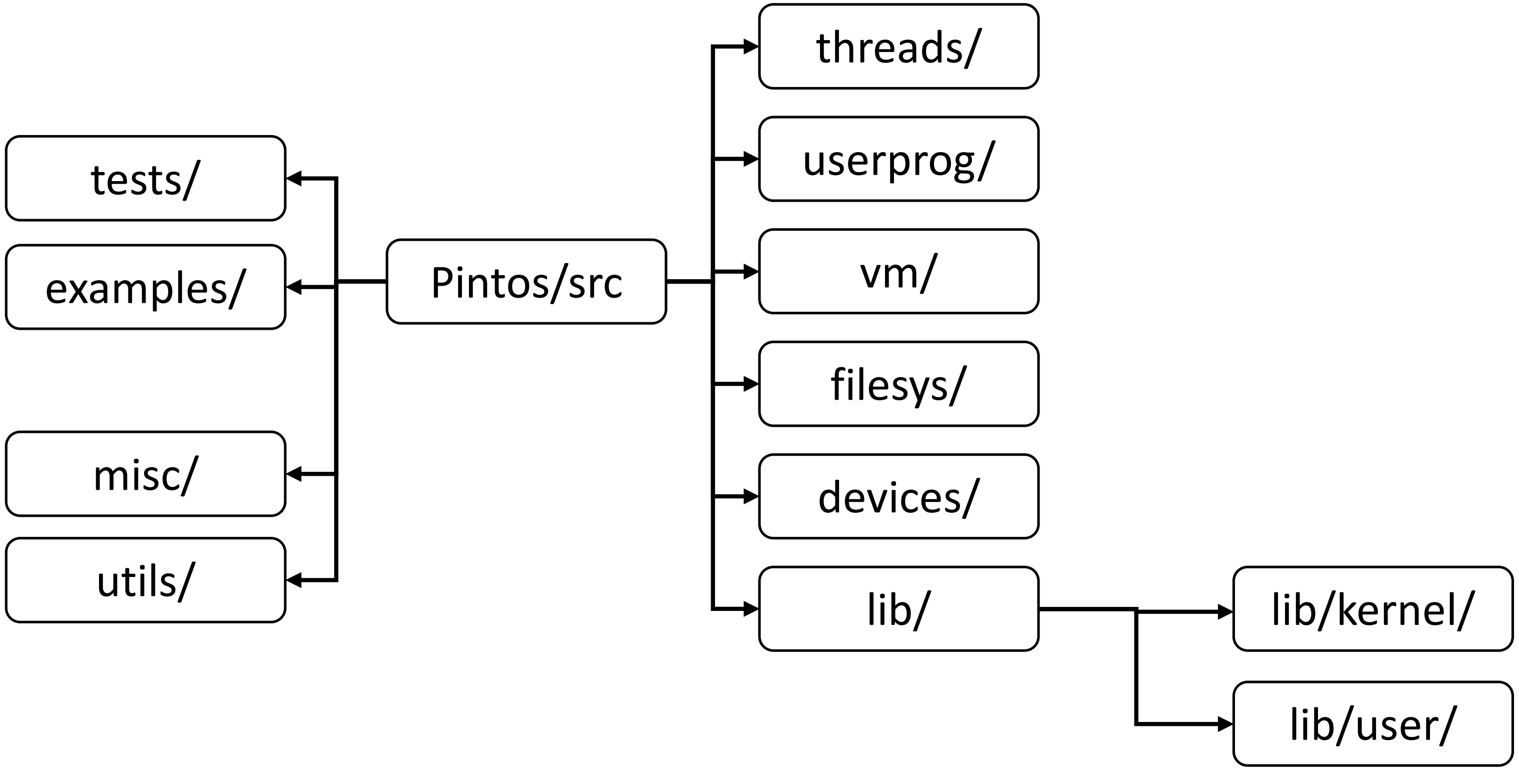
- **Project 3: Virtual Memory (TA: 박재준 / 오민현)**
  - Remove the limitation of the number and size of programs (due to the limited physical memory size of the machine)

Reference : Paper(The Pintos Instructional Operating System Kernel)

Components of pintos



Pintos Source Tree



Pintos Directory Structure

- threads/
  - Source code for the base kernel, which you will modify starting in project 1.
- userprog/
  - Source code for the user program loader, which you will modify starting with project 2.
- vm/
  - An almost empty directory. You will implement virtual memory here in project 3.
- filesys/
  - Source code for a basic file system. You will use file system in project 2.

Pintos Directory Structure(Cont.)

- devices/
  - Source code for I/O device interfacing: keyboard, timer, disk, etc.
  - You will modify the timer implementation in project 1
- lib/
  - An Implementation of a subset of the standard C library. The code in this directory is compiled into both the Pintos kernel and user programs
- lib/kernel/
  - Parts of the C library that are included only in the Pintos Kernel.
- lib/user/
  - Parts of the C library that are included only in Pintos user programs
- tests/
  - Test for each project
- examples/
  - Example user programs for use starting with project 2
- misc/, utils
  - These files may come in handy if you decide to try working with Pintos on your machine

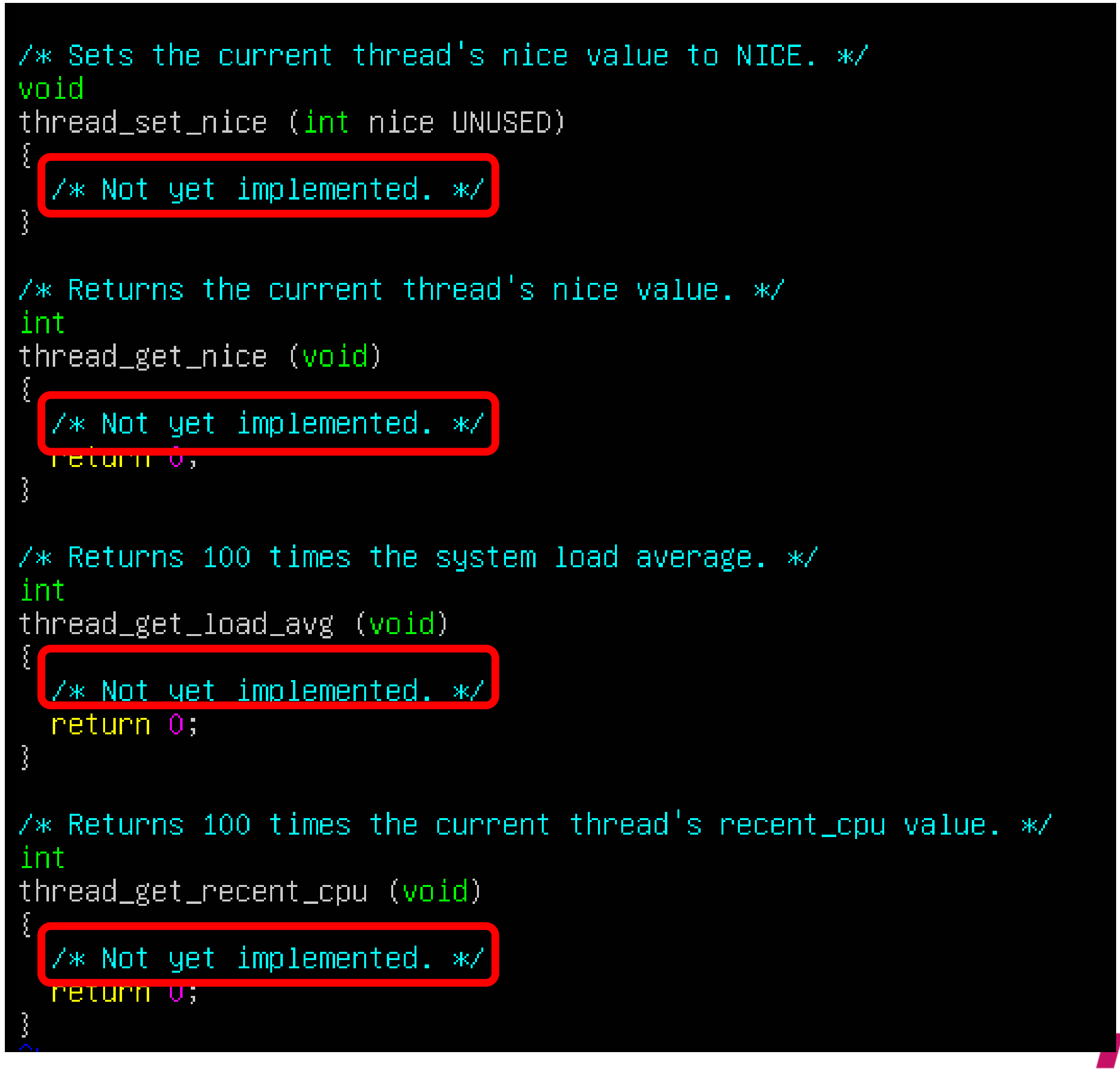
How to install Pintos?

- Using VMware Player
    - TAs already installed Pintos and its emulator
    - Download VMware Player
        - https://www.vmware.com/kr/products/workstation-player/workstation-player-evaluation.html
            - Recommend that you download VMWare Workstation 16 Player
        - https://www.dropbox.com/s/urwvw0od0utb3oh/Ubuntu.zip?dl=0
        - Unzip the file and open Ubuntu.vmdk file with VMware Player
    - You just download VM image and enjoy projects
        - https://github.com/postech-csed312-2019/pintos
        - Username / password of VM Image: pintos / pintos
        - Pintos source codes are in /home/pintos/pintos/src
- Using Server
    - TAs will set up a server for your projects
    - You can connect to the development server and enjoy projects
    - The information about the server will be noticed through PLMS
    - Accounts will be assigned after making teams
    - You should submit the codes on the server.

How to Modify/Build/Test Pintos?

- **Modification**
  - You can add or modify any .c and .h files in source codes
  - But FAQ section in Pintos project website gives hints for it
    - http://www.stanford.edu/class/cs140/projects/pintos/pintos.html
  - Pintos Manual
    - https://web.stanford.edu/class/cs140/projects/pintos/pintos.pdf
- **Build**
  - Go to "pintos/src/threads" (for project 1), and enter “make” to build
  - Go to "pintos/src/utils", and enter “make” to build
  - “make clean”: remove current build results
- **Test**
  - "pintos run (program)": run (program) in pintos, e.g. pintos run alarm-multiple
  - "make check": run all tests which used in actual grading
    - "make check" should be commanded in "build" directory such as "pintos/src/threads/build"

What to modify (Example)

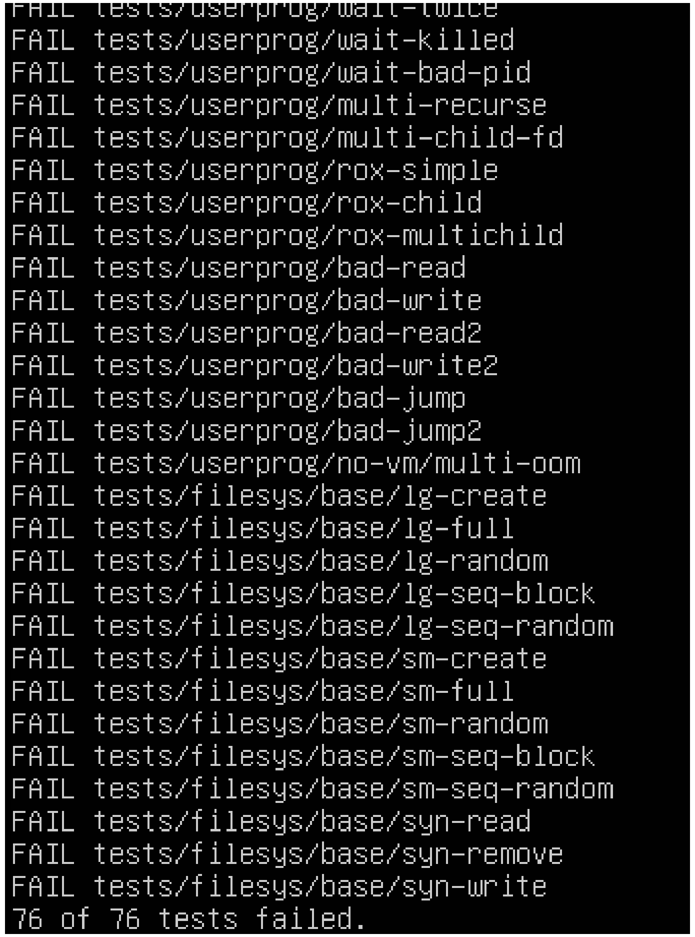


```
/* Sets the current thread's nice value to NICE. */
void
thread_set_nice (int nice UNUSED)
{
  /* Not yet implemented. */
}

/* Returns the current thread's nice value. */
int
thread_get_nice (void)
{
  /* Not yet implemented. */
  return 0;
}

/* Returns 100 times the system load average. */
int
thread_get_load_avg (void)
{
  /* Not yet implemented. */
  return 0;
}

/* Returns 100 times the current thread's recent_cpu value. */
int
thread_get_recent_cpu (void)
{
  /* Not yet implemented. */
  return 0;
}
```

Example result of Test

- **make check**
  - This will build and run each test and print a “pass” or “fail” message for each one.
  - When a test fails, make check also prints some details of the reason for failure

- **make grade**
  - You can get the actual grade that you will get.



```
FAIL tests/userprog/wait-twice
FAIL tests/userprog/wait-killed
FAIL tests/userprog/wait-bad-pid
FAIL tests/userprog/multi-recurse
FAIL tests/userprog/multi-child-fd
FAIL tests/userprog/rox-simple
FAIL tests/userprog/rox-child
FAIL tests/userprog/rox-multichild
FAIL tests/userprog/bad-read
FAIL tests/userprog/bad-write
FAIL tests/userprog/bad-read2
FAIL tests/userprog/bad-write2
FAIL tests/userprog/bad-jump
FAIL tests/userprog/bad-jump2
FAIL tests/userprog/no-vm/multi-oom
FAIL tests/filesys/base/lg-create
FAIL tests/filesys/base/lg-full
FAIL tests/filesys/base/lg-random
FAIL tests/filesys/base/lg-seq-block
FAIL tests/filesys/base/lg-seq-random
FAIL tests/filesys/base/sm-create
FAIL tests/filesys/base/sm-full
FAIL tests/filesys/base/sm-random
FAIL tests/filesys/base/sm-seq-block
FAIL tests/filesys/base/sm-seq-random
FAIL tests/filesys/base/syn-read
FAIL tests/filesys/base/syn-remove
FAIL tests/filesys/base/syn-write
76 of 76 tests failed.
```

Schedule for Each Project

- **1st week**
  - Team Making / Project description from TA
- **3rd week**
  - Submit the design report until the due date (on PLMS)
  - TAs will give scores to the design report with some feedback within several days
- **5th week**
  - Submit the source codes (on the server) and final report (on PLMS)
  - Demo (on site)
  - Next project description and quiz

Contents of Design & Final Report

- **Design report**
  - Brief problem description
  - Current implementation in source codes and how it works
    - You should analyze all related source codes
  - How to solve
    - Data structure & algorithms

- **Final report**
  - The contents of the design report
  - Which function or data structure did you modified or added?
  - Discussion (what you've learned, …)

Submitting project implementation

- Submit your implementation to the project server
  - Detailed instruction will be released soon
- **You should submit .git file along with your implementation!**
  - Students are highly encouraged to make meaningful commit messages

Marking checkpoints are always good for you guys!

Whole Schedule (1)

✓ : There will be a lab session

Sun	Mon	Tue	Wed	Thu	Fri	Sat	
	9/4	5	6	7	8	9	Week 1
		Lab Intro (Pintos)					
10	11	12	13	14	15	16	Week 2
	Team-making due	✓ Project 1 Description					
17	18	19	20	21	22	23	Week 3
24	25	26	27	28	29	30	Week 4
		Project 1 Design Report due		Happy Chuseok!			
10/1	2	3	4	5	6	7	Week 5
8	9	10	11	12	13	14	Week 6
		✓ Project 1 Code, Final Report due Project 1 Quiz, Demo Project 2 Description					
15	16	17	18	19	20	21	Week 7
		Project 2 Design Report due					
22	23	24	25	26	27	28	Week 8 (Midterm Exam Period)
		OS Midterm Exam					

Whole Schedule (2)

Sun	Mon	Tue	Wed	Thu	Fri	Sat	
29	30	31	11/1	2	3	4	Week 9
5	6	7	8	9	10	11	Week 10
		✓ Project 2 Code, Final Report due Project 2 Quiz, Demo Project 3 Description					
12	13	14	15	16	17	18	Week 11
19	20	21	22	23	24	25	Week 12
		Project 3 Design Report due					
26	27	28	29	30	12/1	2	Week 13
3	4	5	6	7	8	9	Week 14
		✓ Project 3 Code, Final Report due Project 3 Quiz, Demo					
10	11	12	13	14	15	16	Week 15
17	18	19	20	21	22	23	Week 16 (Final Exam Period)
		OS Final Exam					

Whole Schedule (3)

Summary

Project 1) 4 weeks (9/12 – 10/10)

- Please try to make up your team and start the project early.
- Simultaneously think of solution for project 2 as soon as possible.

Project 2) 4 weeks (10/10 – 11/7)

- including the midterm exam period.

Project 3) 4 weeks (11/12 – 12/10)

Announcements

- **Make a team (2 students – One-person team is not allowed)**
  - **Team making board: https://url.kr/k2iqfd**
  - Write down the team information until **9/11 (MON) 11:59 pm**

    - If you are not in a team till then, we will make up the team arbitrarily and notice it on **9/12**.
- **You MUST write your own source code!**
  - TAs have many solutions (accumulated during a few years)
  - We will ask your source code line by line in the demo sessions.

Demo session

- We will have a demo session after each project.
- Before the demo session, we will have a short project quiz
  - **The quiz starts at 6:20PM**
- Students will be asked to answer 2 questions regarding the implementation, respectively (4 questions in total, for a team)

[Demo session]

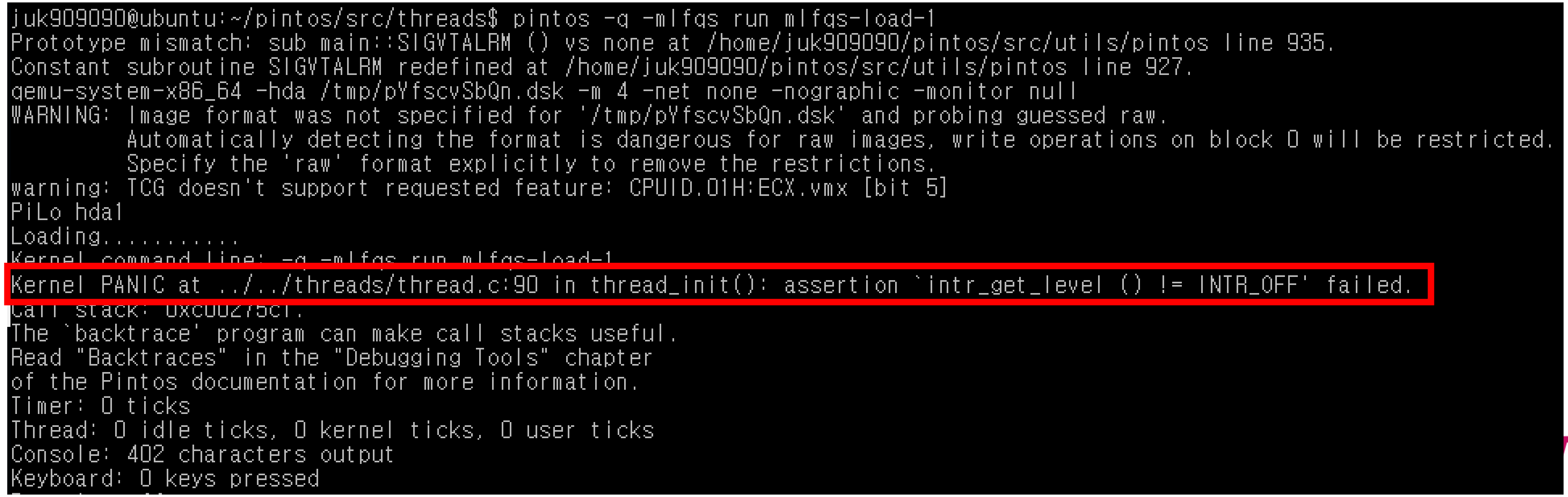
- When: 7:30 PM ~ 11:00 PM
- Where: 컴퓨터공학과 학생휴게실

Debugging Tools

https://www.scs.stanford.edu/10wi-cs140/pintos/pintos_10.html

Kernel panic

- Most things you will see in this project
  - Assertion
  - Not implemented
  - Wrong implementation
  - …
  - Whatever you imagine, you will see beyond your imagination



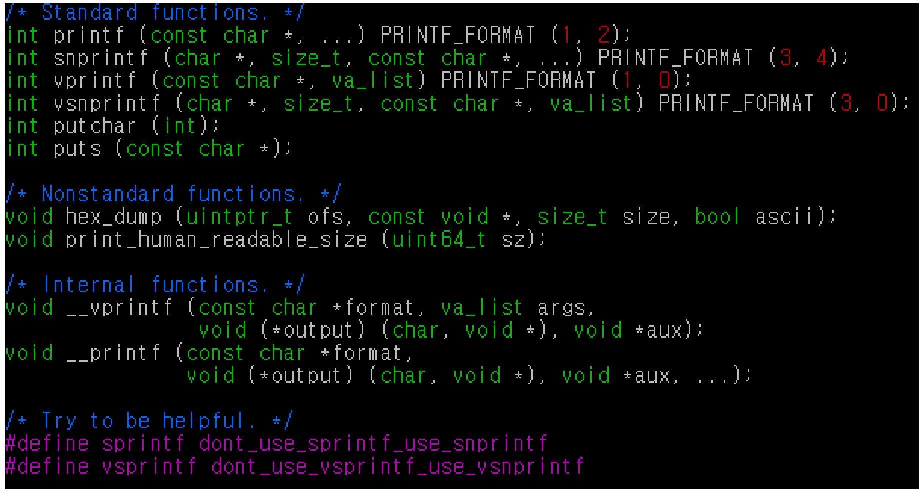
```
juk909090@ubuntu:~/pintos/src/threads$ pintos -q -mlfqs run mlfqs-load-1
Prototype mismatch: sub main::SIGVTALRM () vs none at /home/juk909090/pintos/src/utils/pintos line 935.
Constant subroutine SIGVTALRM redefined at /home/juk909090/pintos/src/utils/pintos line 927.
qemu-system-x86_64 -hda /tmp/pYfscvSbQn.dsk -m 4 -net none -nographic -monitor null
WARNING: Image format was not specified for '/tmp/pYfscvSbQn.dsk' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
warning: TCG doesn't support requested feature: CPUID.01H:ECX.vmx [bit 5]
PiLo hda1
Loading...........
Kernel command line: -q -mlfqs run mlfqs-load-1
Kernel PANIC at ../../threads/thread.c:90 in thread_init(): assertion `intr_get_level () != INTR_OFF' failed.
Call stack: 0xc00275c1.
The `backtrace' program can make call stacks useful.
Read "Backtraces" in the "Debugging Tools" chapter
of the Pintos documentation for more information.
Timer: 0 ticks
Thread: 0 idle ticks, 0 kernel ticks, 0 user ticks
Console: 402 characters output
Keyboard: 0 keys pressed
```

Pintos debugging tools

- printf()
- ASSERT
- Backtraces
- GDB(GNU Project Debugger)

printf function (1/3)

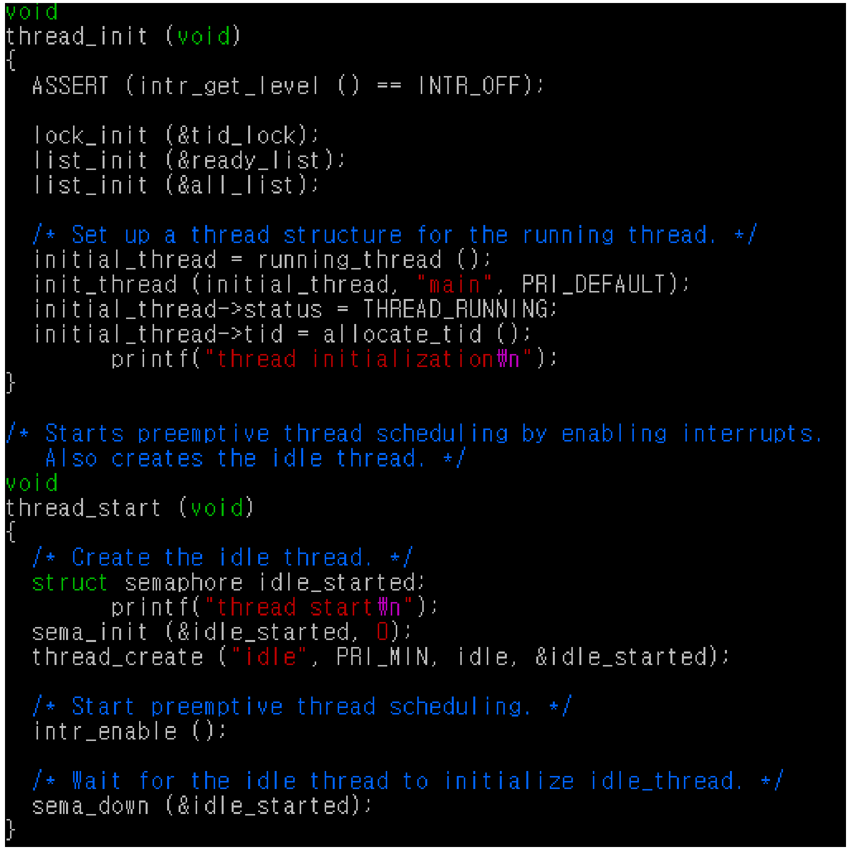
- Very simple, but powerful debugging tool
- Defined in src/lib/stdio.h
- Same output format with original printf



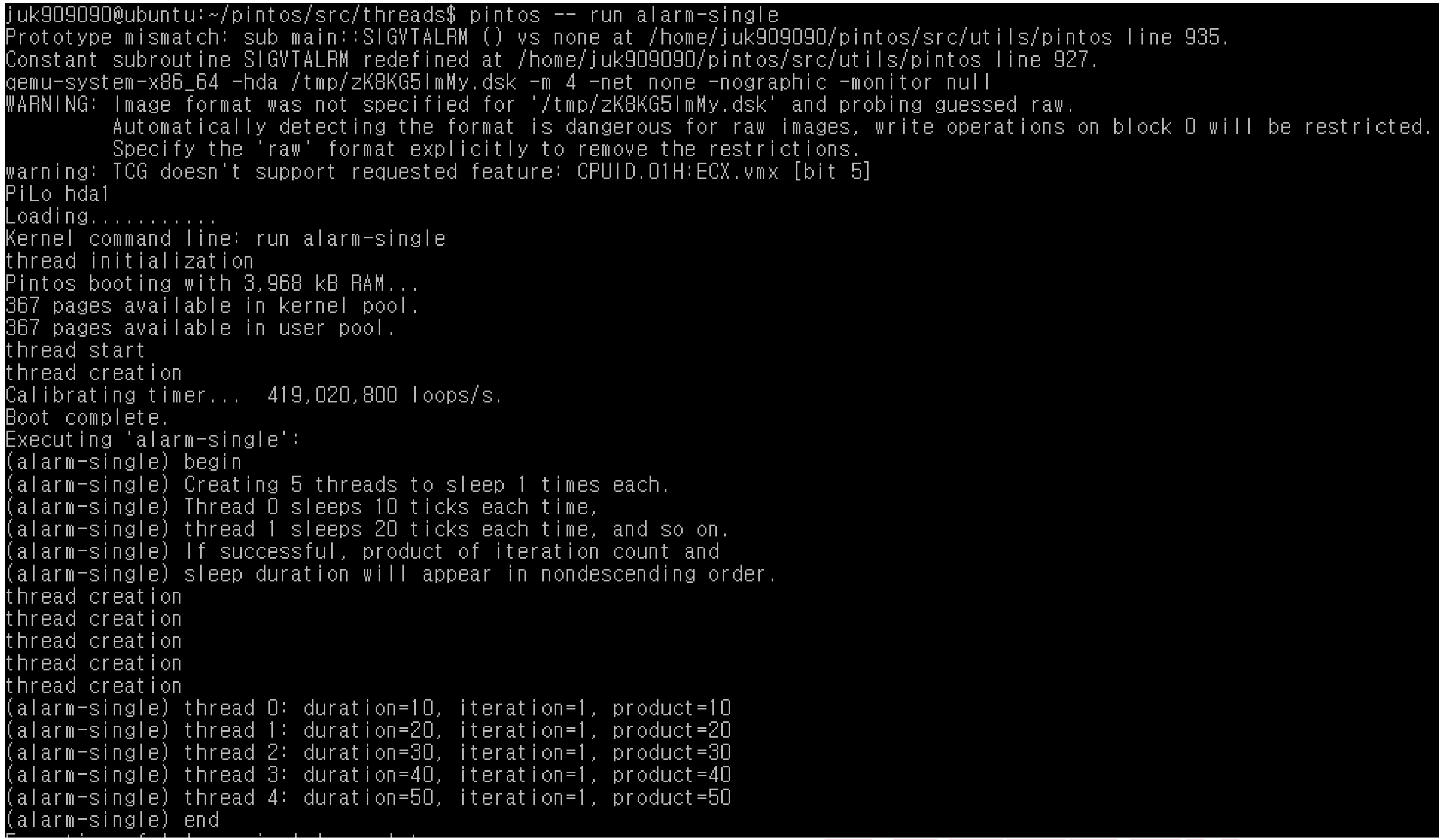
```
/* Standard functions. */
int printf (const char *, ...) PRINTF_FORMAT (1, 2);
int snprintf (char *, size_t, const char *, ...) PRINTF_FORMAT (3, 4);
int vprintf (const char *, va_list) PRINTF_FORMAT (1, 0);
int vsnprintf (char *, size_t, const char *, va_list) PRINTF_FORMAT (3, 0);
int putchar (int);
int puts (const char *);

/* Nonstandard functions. */
void hex_dump (uintptr_t ofs, const void *, size_t size, bool ascii);
void print_human_readable_size (uint64_t sz);

/* Internal functions. */
void __vprintf (const char *format, va_list args,
                void (*output) (char, void *), void *aux);
void __printf (const char *format,
               void (*output) (char, void *), void *aux, ...);

/* Try to be helpful. */
#define sprintf dont_use_sprintf_use_snprintf
#define vsprintf dont_use_vsprintf_use_vsnprintf
```

printf function (2/3)

1. To trace the control flow of your program



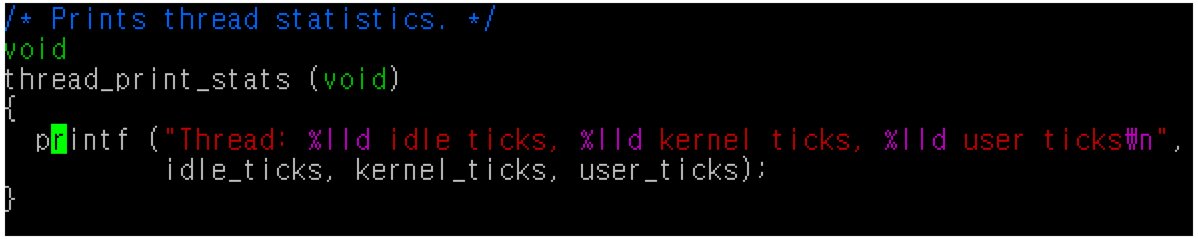
```
void
thread_init (void)
{
  ASSERT (intr_get_level () == INTR_OFF);

  lock_init (&tid_lock);
  list_init (&ready_list);
  list_init (&all_list);

  /* Set up a thread structure for the running thread. */
  initial_thread = running_thread ();
  init_thread (initial_thread, "main", PRI_DEFAULT);
  initial_thread->status = THREAD_RUNNING;
  initial_thread->tid = allocate_tid ();
        printf("thread initialization₩n");
}

/* Starts preemptive thread scheduling by enabling interrupts.
   Also creates the idle thread. */
void
thread_start (void)
{
  /* Create the idle thread. */
  struct semaphore idle_started;
        printf("thread start₩n");
  sema_init (&idle_started, 0);
  thread_create ("idle", PRI_MIN, idle, &idle_started);

  /* Start preemptive thread scheduling. */
  intr_enable ();

  /* Wait for the idle thread to initialize idle_thread. */
  sema_down (&idle_started);
}
```



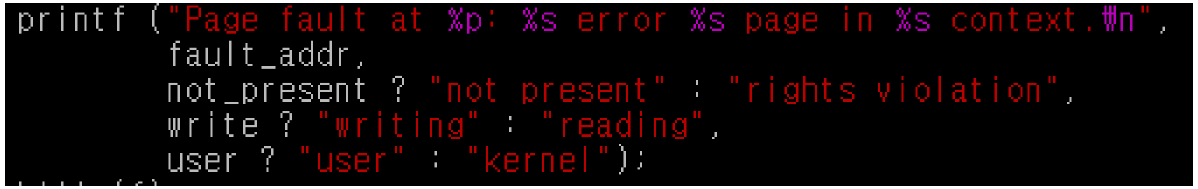
```
juk909090@ubuntu:~/pintos/src/threads$ pintos -- run alarm-single
Prototype mismatch: sub main::SIGVTALRM () vs none at /home/juk909090/pintos/src/utils/pintos line 935.
Constant subroutine SIGVTALRM redefined at /home/juk909090/pintos/src/utils/pintos line 927.
qemu-system-x86_64 -hda /tmp/zK8KG5lmMy.dsk -m 4 -net none -nographic -monitor null
WARNING: Image format was not specified for '/tmp/zK8KG5lmMy.dsk' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
warning: TCG doesn't support requested feature: CPUID.01H:ECX.vmx [bit 5]
PiLo hda1
Loading...........
Kernel command line: run alarm-single
thread initialization
Pintos booting with 3,968 kB RAM...
367 pages available in kernel pool.
367 pages available in user pool.
thread start
thread creation
Calibrating timer...  419,020,800 loops/s.
Boot complete.
Executing 'alarm-single':
(alarm-single) begin
(alarm-single) Creating 5 threads to sleep 1 times each.
(alarm-single) Thread 0 sleeps 10 ticks each time,
(alarm-single) thread 1 sleeps 20 ticks each time, and so on.
(alarm-single) If successful, product of iteration count and
(alarm-single) sleep duration will appear in nondescending order.
thread creation
thread creation
thread creation
thread creation
thread creation
(alarm-single) thread 0: duration=10, iteration=1, product=10
(alarm-single) thread 1: duration=20, iteration=1, product=20
(alarm-single) thread 2: duration=30, iteration=1, product=30
(alarm-single) thread 3: duration=40, iteration=1, product=40
(alarm-single) thread 4: duration=50, iteration=1, product=50
(alarm-single) end
```

printf function (3/3)

2. To trace value or status of your program



```
/* Prints thread statistics. */
void
thread_print_stats (void)
{
  printf ("Thread: %lld idle ticks, %lld kernel ticks, %lld user ticks₩n",
          idle_ticks, kernel_ticks, user_ticks);
}
```

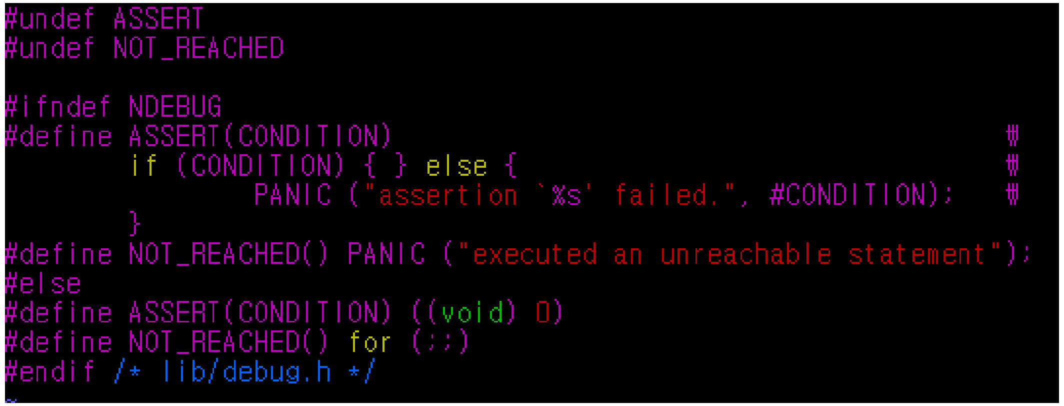


```
printf ("Page fault at %p: %s error %s page in %s context.₩n",
        fault_addr,
        not_present ? "not present" : "rights violation",
        write ? "writing" : "reading",
        user ? "user" : "kernel");
```

ASSERT (1/2)

- Simple debugging tool

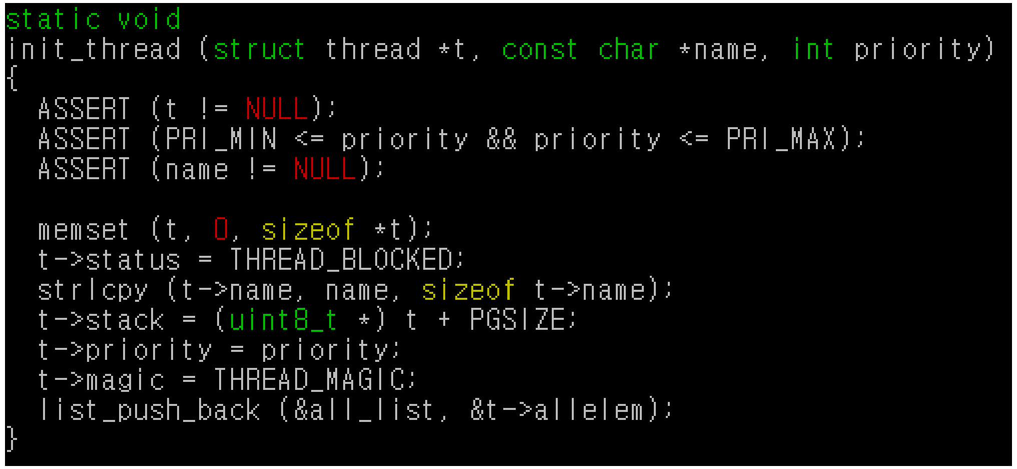
- Defined in src/lib/debug.h



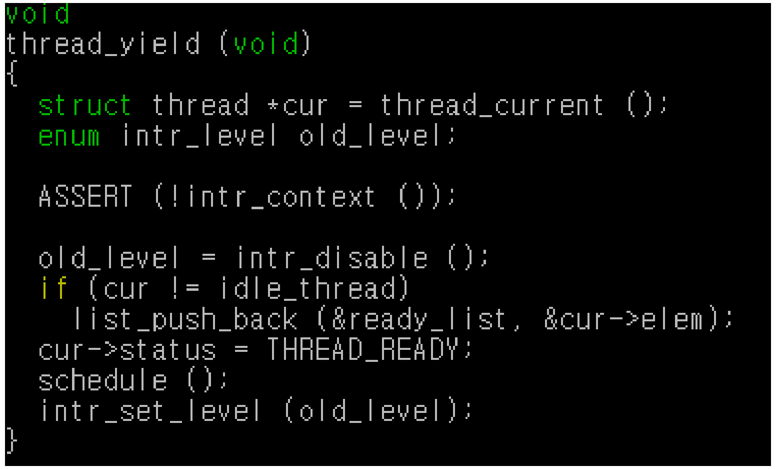
```
#undef ASSERT
#undef NOT_REACHED

#ifndef NDEBUG
#define ASSERT(CONDITION)                                         ₩
        if (CONDITION) { } else {                                 ₩
                PANIC ("assertion `%s' failed.", #CONDITION);     ₩
        }
#define NOT_REACHED() PANIC ("executed an unreachable statement");
#else
#define ASSERT(CONDITION) ((void) 0)
#define NOT_REACHED() for (;;)
#endif /* lib/debug.h */
```

ASSERT (2/2)

1. To convince yourself that everything goes well



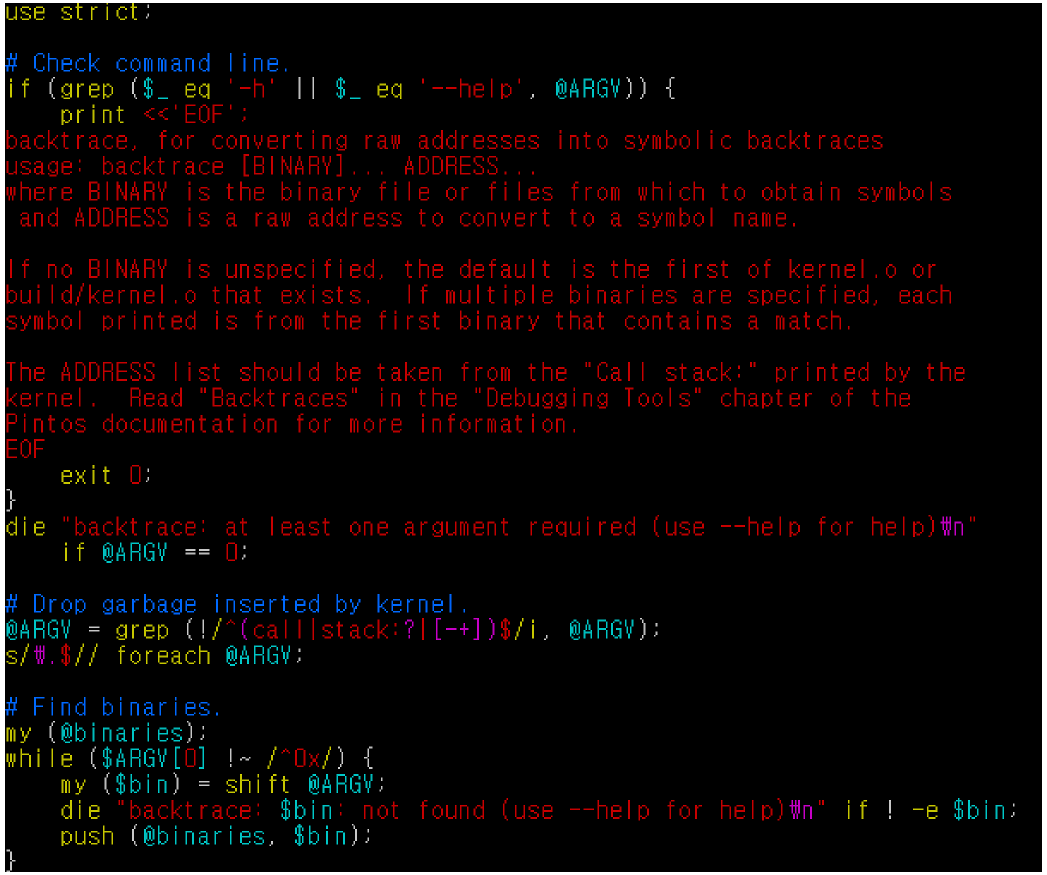
```
static void
init_thread (struct thread *t, const char *name, int priority)
{
  ASSERT (t != NULL);
  ASSERT (PRI_MIN <= priority && priority <= PRI_MAX);
  ASSERT (name != NULL);

  memset (t, 0, sizeof *t);
  t->status = THREAD_BLOCKED;
  strlcpy (t->name, name, sizeof t->name);
  t->stack = (uint8_t *) t + PGSIZE;
  t->priority = priority;
  t->magic = THREAD_MAGIC;
  list_push_back (&all_list, &t->allelem);
}
```



```
void
thread_yield (void)
{
  struct thread *cur = thread_current ();
  enum intr_level old_level;

  ASSERT (!intr_context ());

  old_level = intr_disable ();
  if (cur != idle_thread)
    list_push_back (&ready_list, &cur->elem);
  cur->status = THREAD_READY;
  schedule ();
  intr_set_level (old_level);
}
```

Backtraces (1/2)

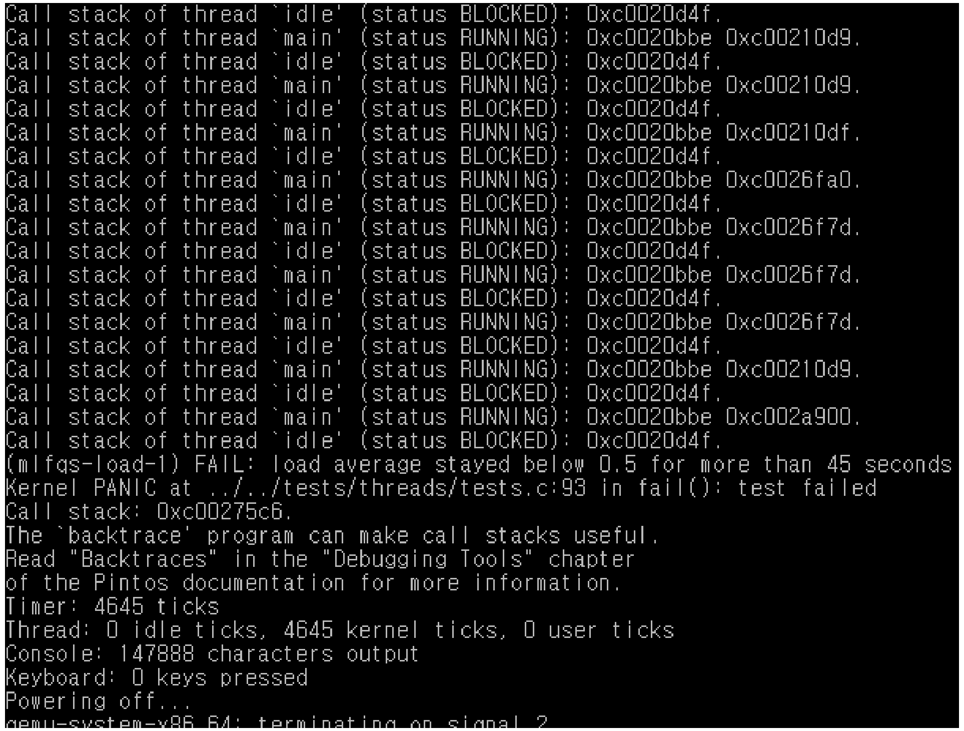
- To make call stack human readable
- Located in src/utils/bracktrace
- To interpret kernel binary file (*/build/kernel.o)
  - Ex) backtrace threads/build/kernel.o [0xc0000000…]



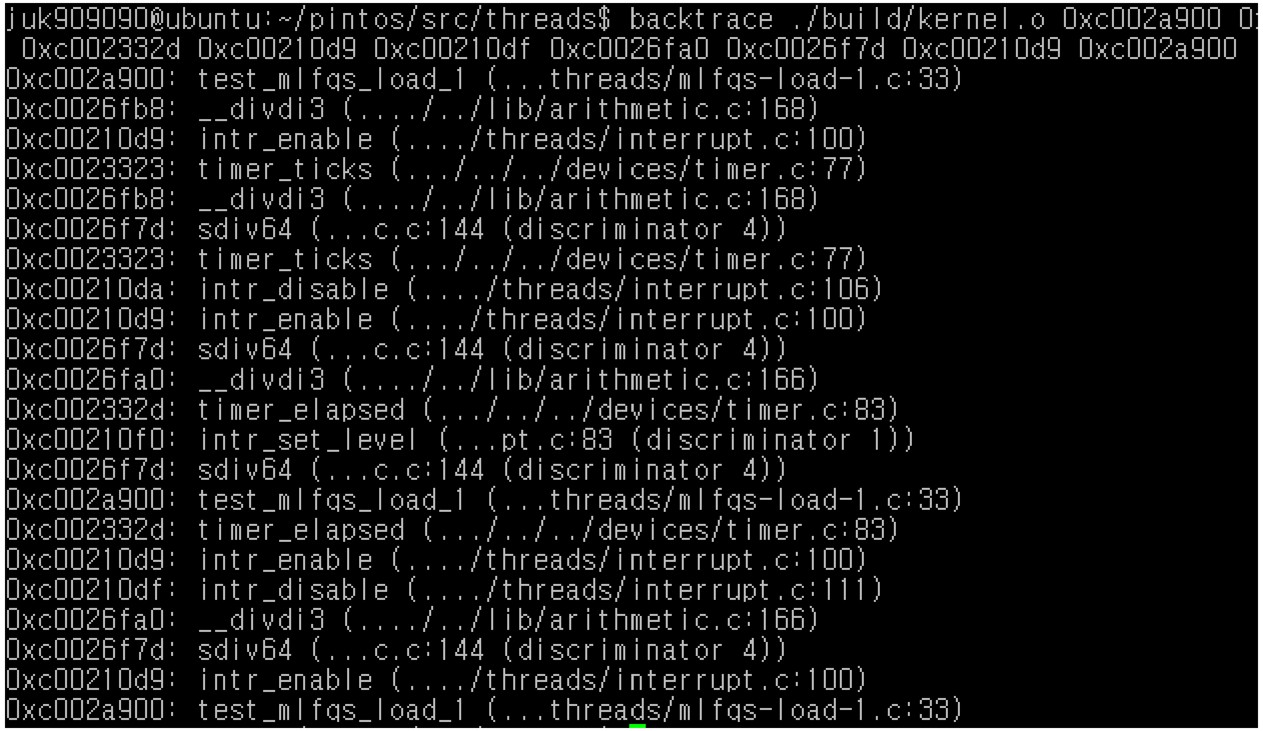
```
use strict;

# Check command line.
if (grep ($_ eq '-h' || $_ eq '--help', @ARGV)) {
    print <<'EOF';
backtrace, for converting raw addresses into symbolic backtraces
usage: backtrace [BINARY]... ADDRESS...
where BINARY is the binary file or files from which to obtain symbols
 and ADDRESS is a raw address to convert to a symbol name.

If no BINARY is unspecified, the default is the first of kernel.o or
build/kernel.o that exists.  If multiple binaries are specified, each
symbol printed is from the first binary that contains a match.

The ADDRESS list should be taken from the "Call stack:" printed by the
kernel.  Read "Backtraces" in the "Debugging Tools" chapter of the
Pintos documentation for more information.
EOF
    exit 0;
}
die "backtrace: at least one argument required (use --help for help)₩n"
    if @ARGV == 0;

# Drop garbage inserted by kernel.
@ARGV = grep (!/^(call|stack:?|[-+])$/i, @ARGV);
s/₩.$// foreach @ARGV;

# Find binaries.
my (@binaries);
while ($ARGV[0] !~ /^0x/) {
    my ($bin) = shift @ARGV;
    die "backtrace: $bin: not found (use --help for help)₩n" if ! -e $bin;
    push (@binaries, $bin);
}
```

Backtraces (2/2)

1. To trace the control flow of your program

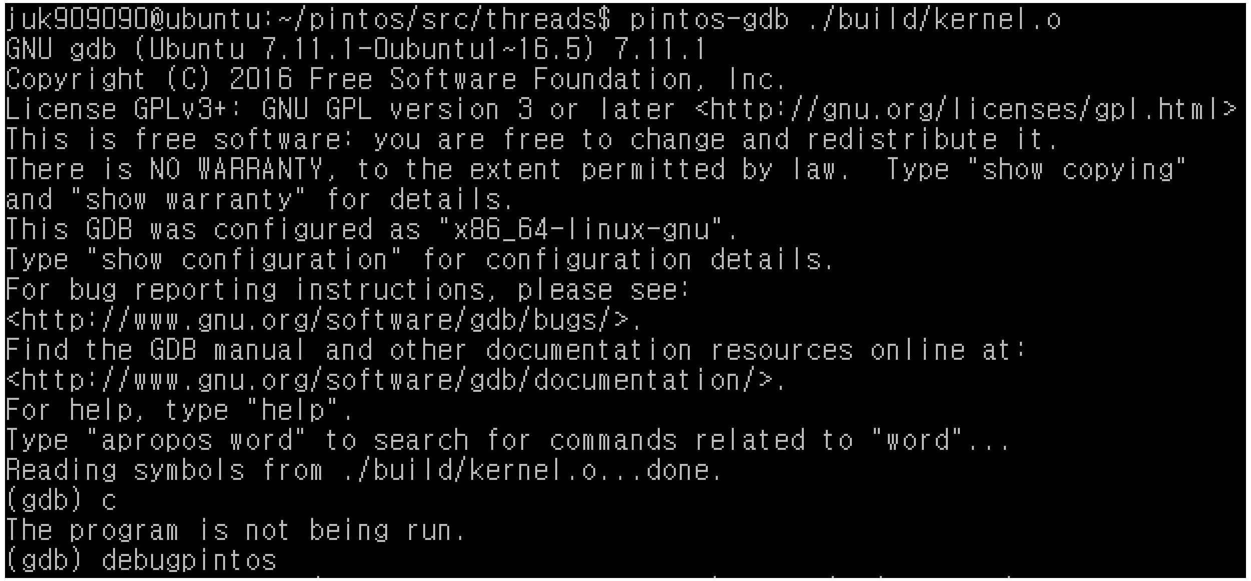


```
Call stack of thread `idle' (status BLOCKED): 0xc0020d4f.
Call stack of thread `main' (status RUNNING): 0xc0020bbe 0xc00210d9.
Call stack of thread `idle' (status BLOCKED): 0xc0020d4f.
Call stack of thread `main' (status RUNNING): 0xc0020bbe 0xc00210d9.
Call stack of thread `idle' (status BLOCKED): 0xc0020d4f.
Call stack of thread `main' (status RUNNING): 0xc0020bbe 0xc00210df.
Call stack of thread `idle' (status BLOCKED): 0xc0020d4f.
Call stack of thread `main' (status RUNNING): 0xc0020bbe 0xc0026fa0.
Call stack of thread `idle' (status BLOCKED): 0xc0020d4f.
Call stack of thread `main' (status RUNNING): 0xc0020bbe 0xc0026f7d.
Call stack of thread `idle' (status BLOCKED): 0xc0020d4f.
Call stack of thread `main' (status RUNNING): 0xc0020bbe 0xc0026f7d.
Call stack of thread `idle' (status BLOCKED): 0xc0020d4f.
Call stack of thread `main' (status RUNNING): 0xc0020bbe 0xc0026f7d.
Call stack of thread `idle' (status BLOCKED): 0xc0020d4f.
Call stack of thread `main' (status RUNNING): 0xc0020bbe 0xc00210d9.
Call stack of thread `idle' (status BLOCKED): 0xc0020d4f.
Call stack of thread `main' (status RUNNING): 0xc0020bbe 0xc002a900.
Call stack of thread `idle' (status BLOCKED): 0xc0020d4f.
(mlfqs-load-1) FAIL: load average stayed below 0.5 for more than 45 seconds
Kernel PANIC at ../../tests/threads/tests.c:93 in fail(): test failed
Call stack: 0xc00275c6.
The `backtrace' program can make call stacks useful.
Read "Backtraces" in the "Debugging Tools" chapter
of the Pintos documentation for more information.
Timer: 4645 ticks
Thread: 0 idle ticks, 4645 kernel ticks, 0 user ticks
Console: 147888 characters output
Keyboard: 0 keys pressed
Powering off...
qemu-system-x86_64: terminating on signal 2
```



```
juk909090@ubuntu:~/pintos/src/threads$ backtrace ./build/kernel.o 0xc002a900 0
 0xc002332d 0xc00210d9 0xc00210df 0xc0026fa0 0xc0026f7d 0xc00210d9 0xc002a900
0xc002a900: test_mlfqs_load_1 (...threads/mlfqs-load-1.c:33)
0xc0026fb8: __divdi3 (..../../lib/arithmetic.c:168)
0xc00210d9: intr_enable (..../threads/interrupt.c:100)
0xc0023323: timer_ticks (.../../../devices/timer.c:77)
0xc0026fb8: __divdi3 (..../../lib/arithmetic.c:168)
0xc0026f7d: sdiv64 (...c.c:144 (discriminator 4))
0xc0023323: timer_ticks (.../../../devices/timer.c:77)
0xc00210da: intr_disable (..../threads/interrupt.c:106)
0xc00210d9: intr_enable (..../threads/interrupt.c:100)
0xc0026f7d: sdiv64 (...c.c:144 (discriminator 4))
0xc0026fa0: __divdi3 (..../../lib/arithmetic.c:166)
0xc002332d: timer_elapsed (.../../../devices/timer.c:83)
0xc00210f0: intr_set_level (...pt.c:83 (discriminator 1))
0xc0026f7d: sdiv64 (...c.c:144 (discriminator 4))
0xc002a900: test_mlfqs_load_1 (...threads/mlfqs-load-1.c:33)
0xc002332d: timer_elapsed (.../../../devices/timer.c:83)
0xc00210d9: intr_enable (..../threads/interrupt.c:100)
0xc00210df: intr_disable (..../threads/interrupt.c:111)
0xc0026fa0: __divdi3 (..../../lib/arithmetic.c:166)
0xc0026f7d: sdiv64 (...c.c:144 (discriminator 4))
0xc00210d9: intr_enable (..../threads/interrupt.c:100)
0xc002a900: test_mlfqs_load_1 (...threads/mlfqs-load-1.c:33)
```

GDB (1/7)

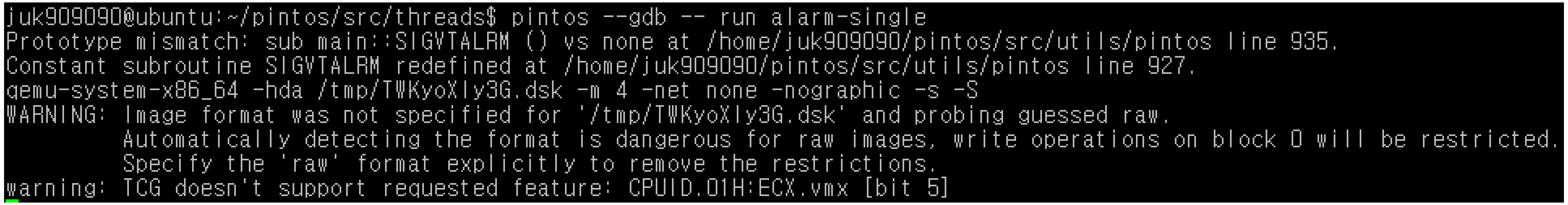


```
juk909090@ubuntu:~/pintos/src/threads$ pintos-gdb ./build/kernel.o
GNU gdb (Ubuntu 7.11.1-0ubuntu1~16.5) 7.11.1
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./build/kernel.o...done.
(gdb) c
The program is not being run.
(gdb) debugpintos
```

- GDB (GNU Project Debugger)

- Included in the GCC package

- Pintos supports GDB as an remote debugging tool
  - Run test with “--gdb” option
  - **pintos-gdb threads/build/kernel.o** (in other terminal)
  - debugpintos (in gdb)



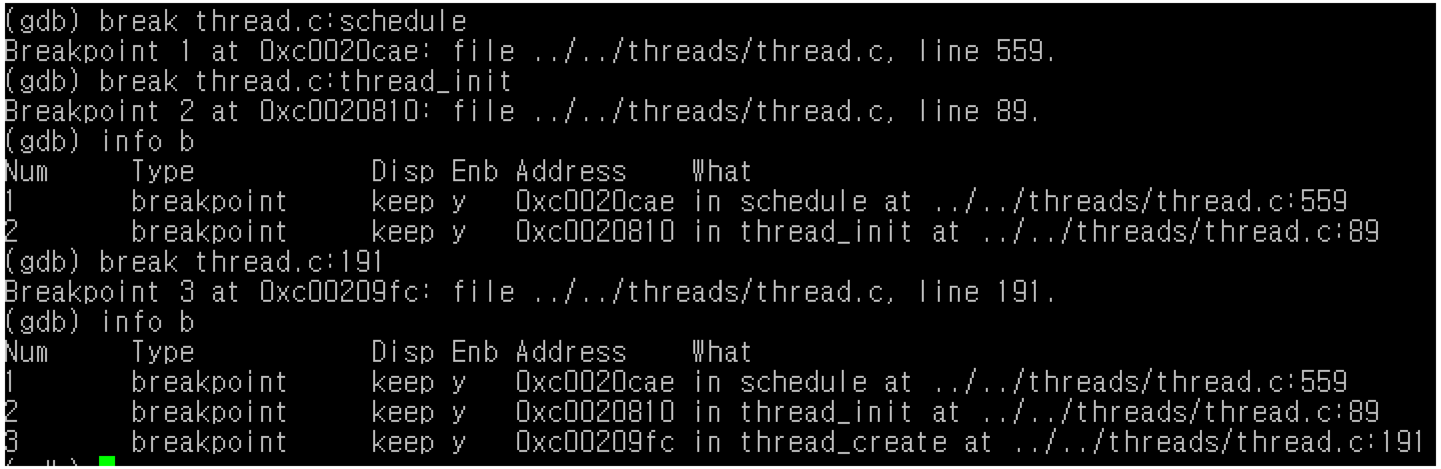
```
juk909090@ubuntu:~/pintos/src/threads$ pintos --gdb -- run alarm-single
Prototype mismatch: sub main::SIGVTALRM () vs none at /home/juk909090/pintos/src/utils/pintos line 935.
Constant subroutine SIGVTALRM redefined at /home/juk909090/pintos/src/utils/pintos line 927.
qemu-system-x86_64 -hda /tmp/TWKyoXly3G.dsk -m 4 -net none -nographic -s -S
WARNING: Image format was not specified for '/tmp/TWKyoXly3G.dsk' and probing guessed raw.
         Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
         Specify the 'raw' format explicitly to remove the restrictions.
warning: TCG doesn't support requested feature: CPUID.01H:ECX.vmx [bit 5]
```

GDB (2/7)

- GDB command – break (shorten b)
  - establish breakpoints of the program

  - break
  - break (function name)
  - break (line number)
  - break (file name):(function name)
  - break (file name):(line number)
  - …
  - info break

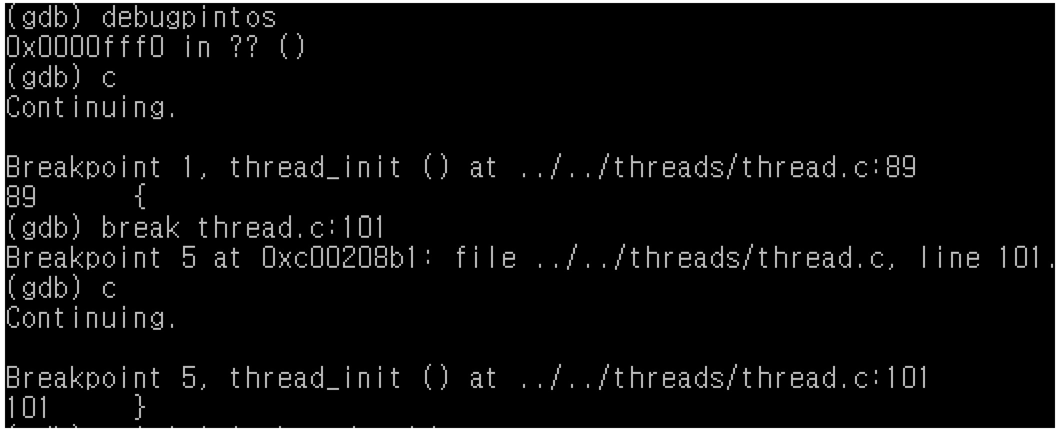
  - disable br (num)
  - enable br (num)
  - …



```
(gdb) break thread.c:schedule
Breakpoint 1 at 0xc0020cae: file ../../threads/thread.c, line 559.
(gdb) break thread.c:thread_init
Breakpoint 2 at 0xc0020810: file ../../threads/thread.c, line 89.
(gdb) info b
Num     Type           Disp Enb Address    What
1       breakpoint     keep y   0xc0020cae in schedule at ../../threads/thread.c:559
2       breakpoint     keep y   0xc0020810 in thread_init at ../../threads/thread.c:89
(gdb) break thread.c:191
Breakpoint 3 at 0xc00209fc: file ../../threads/thread.c, line 191.
(gdb) info b
Num     Type           Disp Enb Address    What
1       breakpoint     keep y   0xc0020cae in schedule at ../../threads/thread.c:559
2       breakpoint     keep y   0xc0020810 in thread_init at ../../threads/thread.c:89
3       breakpoint     keep y   0xc00209fc in thread_create at ../../threads/thread.c:191
```

GDB (3/7)

- GDB command – continue (shorten c)
  - execute the program until breakpoints



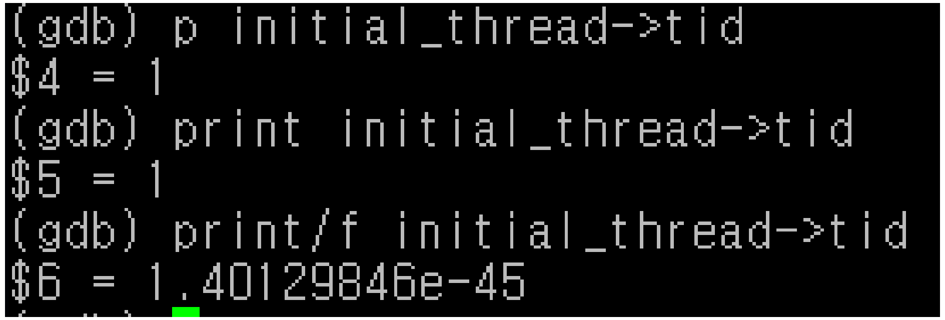
```
(gdb) debugpintos
0x0000fff0 in ?? ()
(gdb) c
Continuing.

Breakpoint 1, thread_init () at ../../threads/thread.c:89
89        {
(gdb) break thread.c:101
Breakpoint 5 at 0xc00208b1: file ../../threads/thread.c, line 101.
(gdb) c
Continuing.

Breakpoint 5, thread_init () at ../../threads/thread.c:101
101       }
```

GDB (4/7)

- GDB command – step (shorten s)
  - execute the program line by line

GDB (5/7)

- GDB command – print (shorten p)
  - print a variable
  - print/format variable
    - x : hex
    - d : dec
    - u : unsigned dec
    - t : binary
    - f : floating point
    - …

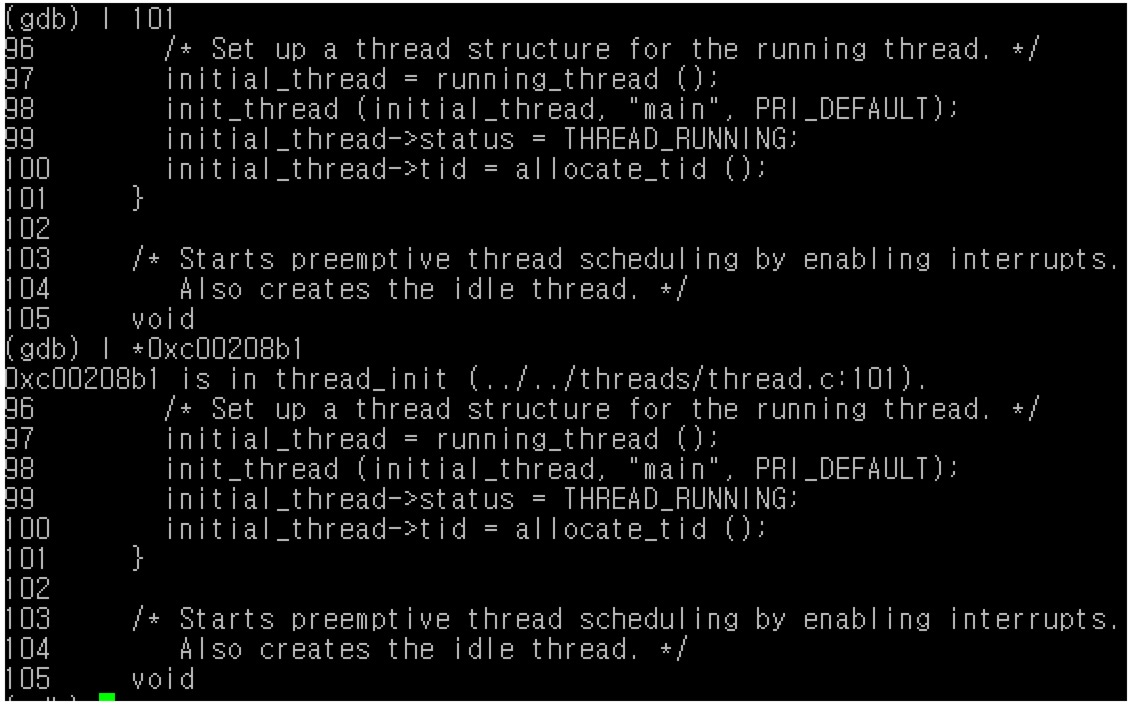


```
(gdb) p initial_thread->tid
$4 = 1
(gdb) print initial_thread->tid
$5 = 1
(gdb) print/f initial_thread->tid
$6 = 1.40129846e-45
```

GDB (6/7)

- GDB command – list (shorten l)
  - show the partial code of the program

  - list (function name)
  - list (line number)
  - list (file name):(function name)
  - list (file name):(line number)
  - list *(address)



```
(gdb) l 101
96          /* Set up a thread structure for the running thread. */
97          initial_thread = running_thread ();
98          init_thread (initial_thread, "main", PRI_DEFAULT);
99          initial_thread->status = THREAD_RUNNING;
100         initial_thread->tid = allocate_tid ();
101       }
102
103       /* Starts preemptive thread scheduling by enabling interrupts.
104          Also creates the idle thread. */
105       void
(gdb) l *0xc00208b1
0xc00208b1 is in thread_init (../../threads/thread.c:101).
96          /* Set up a thread structure for the running thread. */
97          initial_thread = running_thread ();
98          init_thread (initial_thread, "main", PRI_DEFAULT);
99          initial_thread->status = THREAD_RUNNING;
100         initial_thread->tid = allocate_tid ();
101       }
102
103       /* Starts preemptive thread scheduling by enabling interrupts.
104          Also creates the idle thread. */
105       void
```

GDB (7/7)

- References
  - https://sourceware.org/gdb/onlinedocs/gdb/index.html#SEC_Contents
  - https://kldp.org/node/71806
  - https://kldp.org/node/87778

Thank you

Autumn 2023

Supplementary Utilities

Supplementary Utilities

- **Git**
  - **Version control system** for tracking changes in computer files
  - Useful for Team Project

- Git GUI Client (e.g. gitk, git-gui, Sourcetree, GitKraken)
  - Clear representation of git history
  - for Git beginner

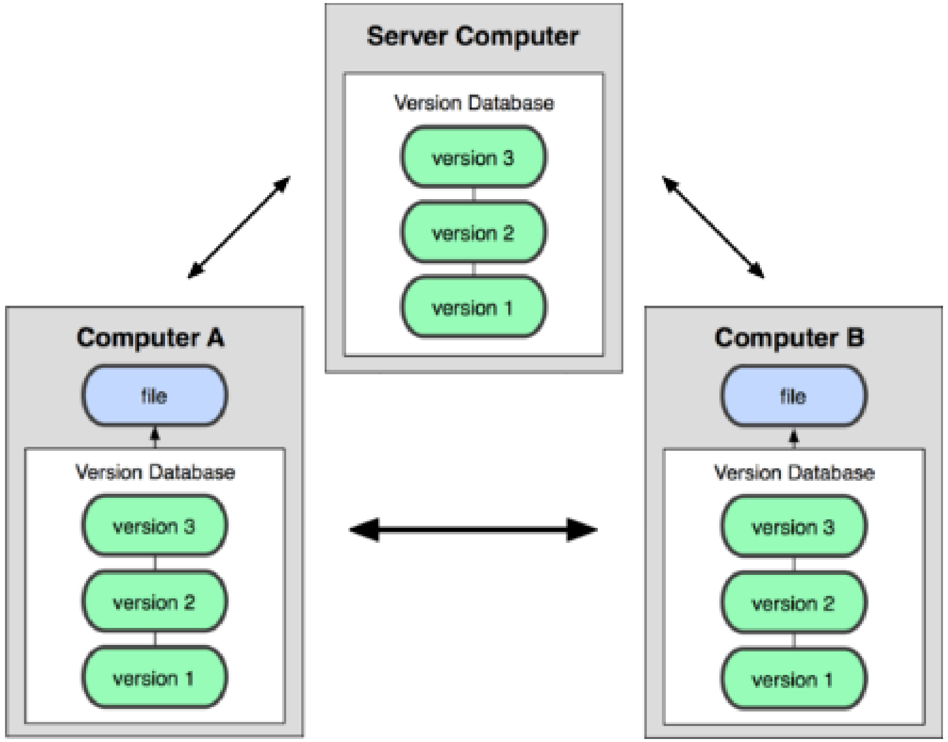
- **Ctags**
  - tool that will sift through your code, indexing methods, variables, and other identifiers, storing the index in a **tags** file
  - makes it much easier to navigate a larger project such as Pintos

Git : Distributed VCS (Version control system)

- 2 main components
  - Server & Local repository

- Server
  - remote repository storing general version of project

- Local repository (Local computer)
  - For local users
  - initially copy from the Server
  - modify & commit changes of files
  - push changes to server

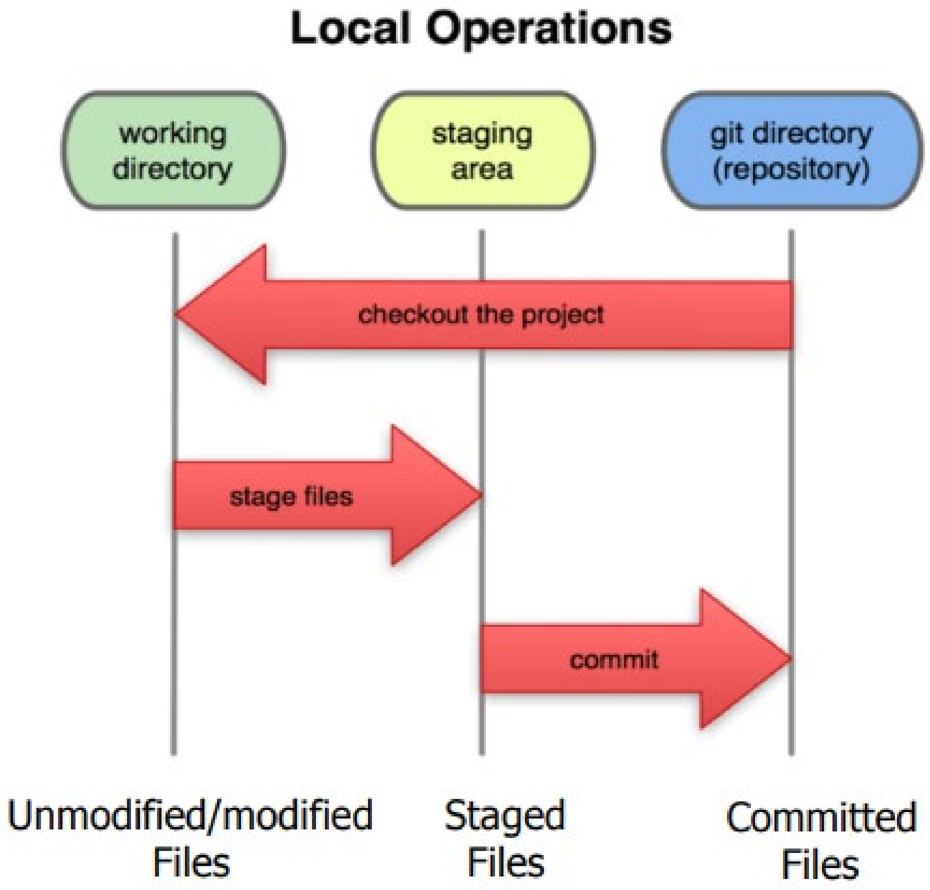


Git : Local areas

- Files in the local repository can be in 3 areas

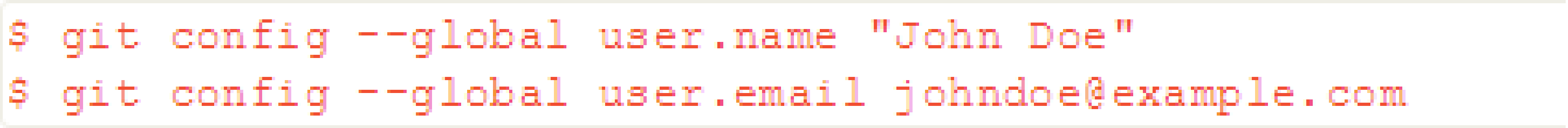
- Staged files are ready to be committed

- Basic workflow
  - Modify files in your working directory
  - Stage files, adding snapshots of them to your staging area
  - Commit, which takes the files in the staging area and stores that snapshot permanently to your git directory



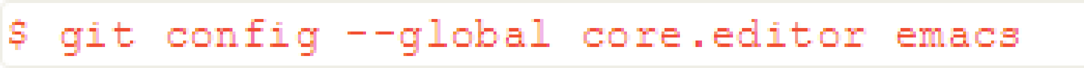
Git : initial git config

- Set the name and email for Git to use when you commit



```
$ git config --global user.name "John Doe"
$ git config --global user.email johndoe@example.com
```

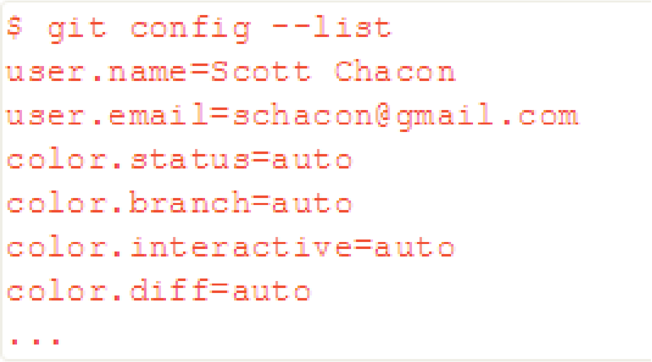
- Set the editor that is used for Git



```
$ git config --global core.editor emacs
```

  - Default : vi(vim)

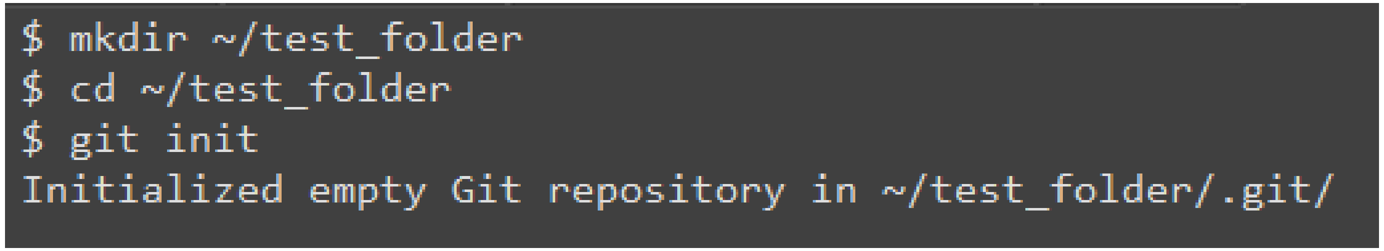
- Identify configuration



```
$ git config --list
user.name=Scott Chacon
user.email=schacon@gmail.com
color.status=auto
color.branch=auto
color.interactive=auto
color.diff=auto
...
```

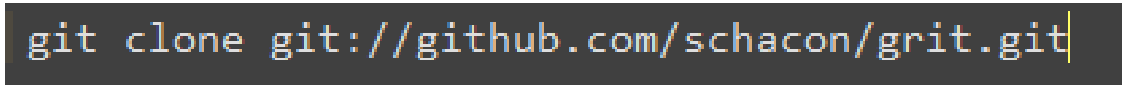
Git : Creating a repository

1. To create a new local Git repo in your current directory
   - git init



```
$ mkdir ~/test_folder
$ cd ~/test_folder
$ git init
Initialized empty Git repository in ~/test_folder/.git/
```

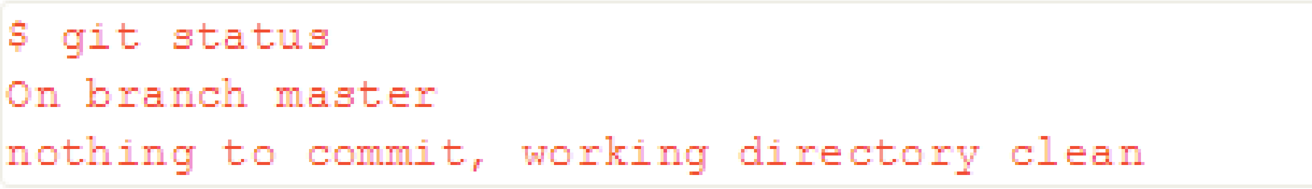
2. To clone a remote repo to your current directory
   - git clone [url]



```
git clone git://github.com/schacon/grit.git
```

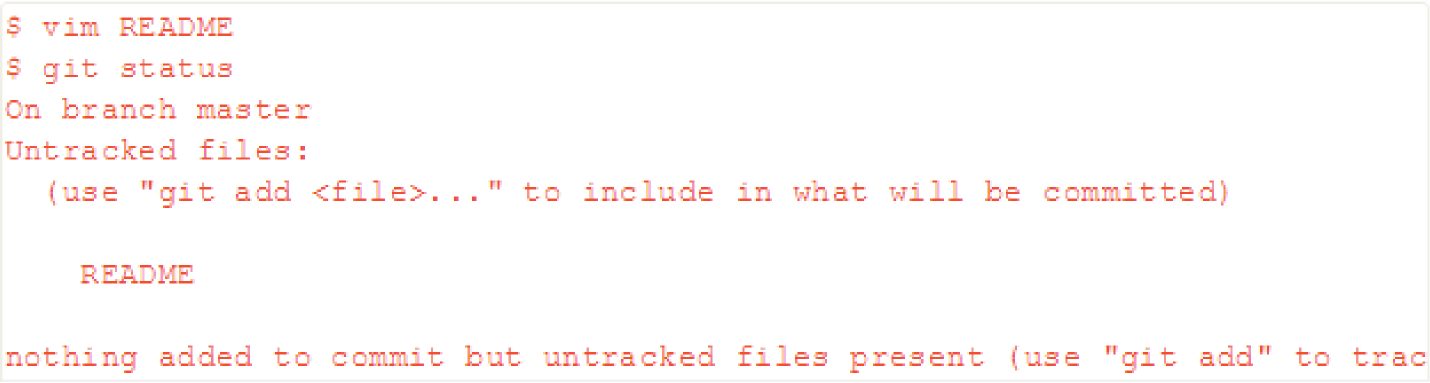
Git : status

- To view the status of your files in the working directory and staging area



```
$ git status
On branch master
nothing to commit, working directory clean
```

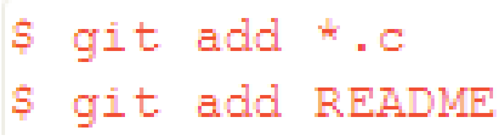
- If you make a file in your working directory, this new one is an untracked file



```
$ vim README
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    README

nothing added to commit but untracked files present (use "git add" to trac
```

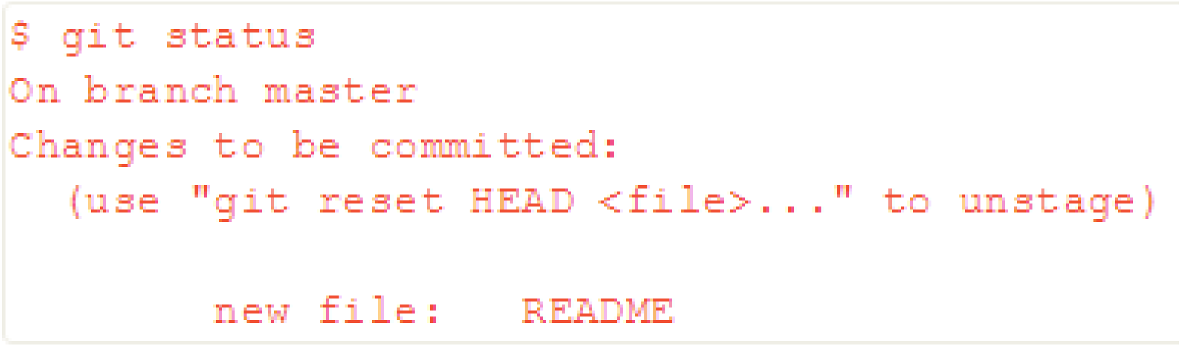
Git : add

- To add file contents to the staging area

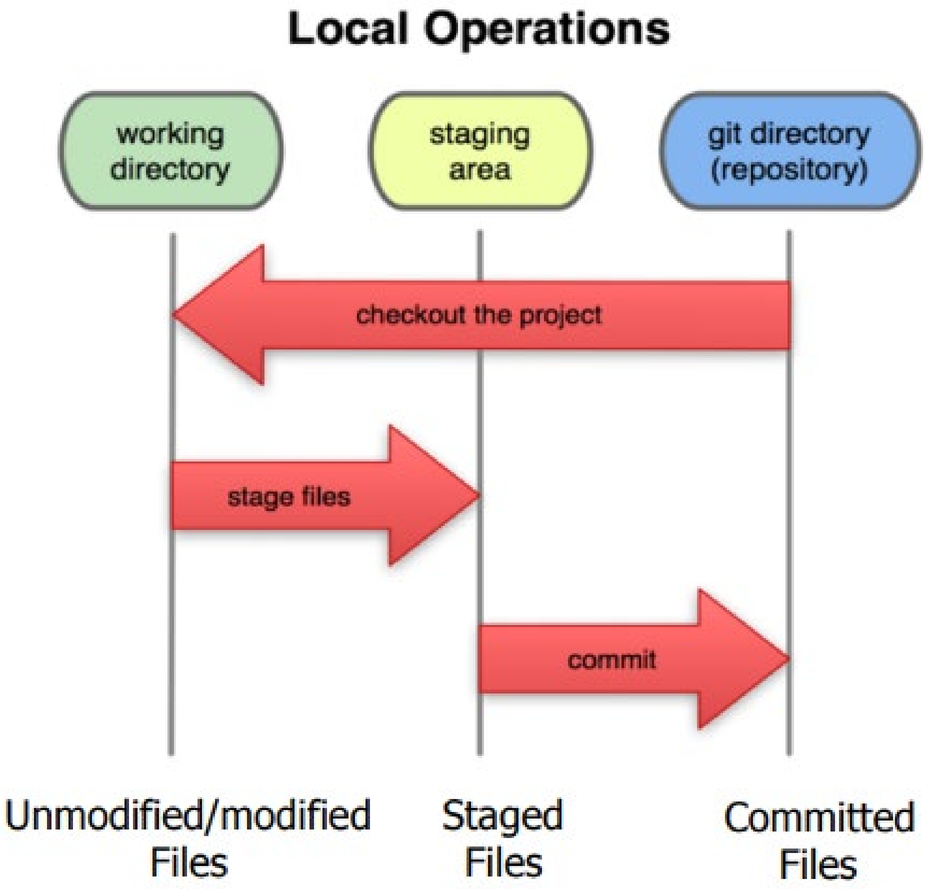


```
$ git add *.c
$ git add README
```

(status command shows the following results)



```
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   README
```

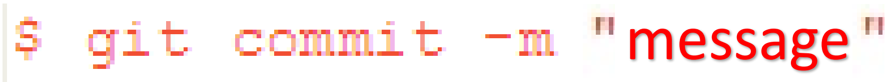


Git : commit

- To record a snapshot of the staging area

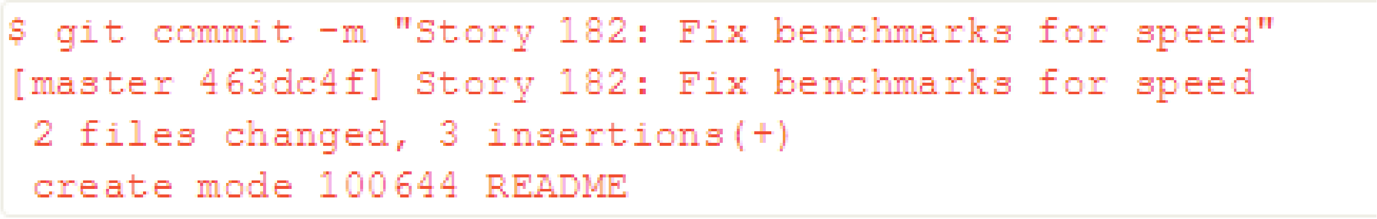
```
$ git commit
```

Or

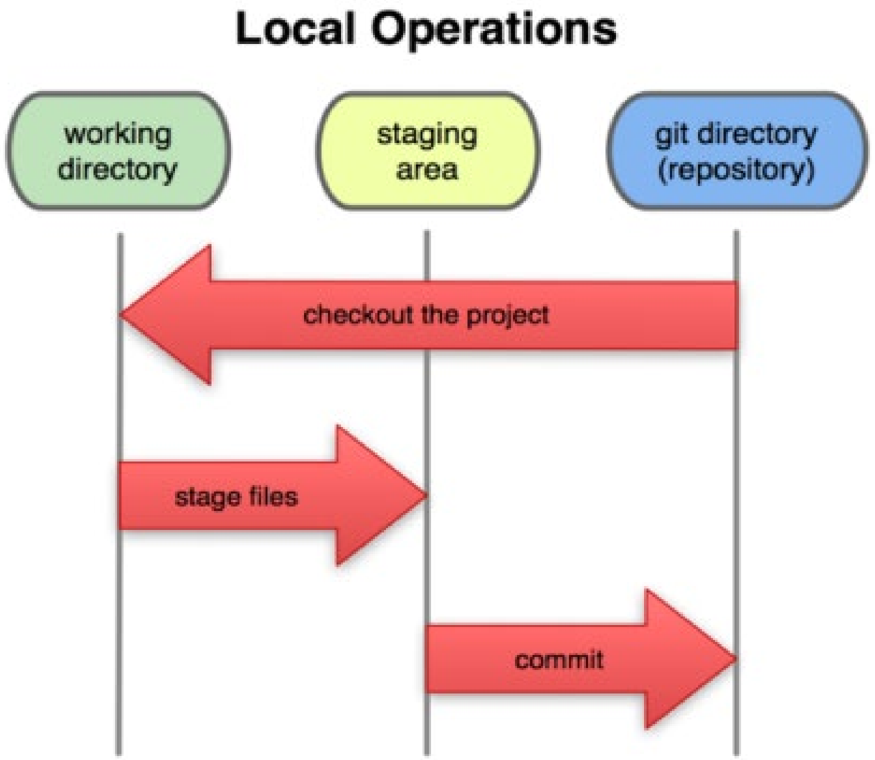


```
$ git commit -m "message"
```

- Example



```
$ git commit -m "Story 182: Fix benchmarks for speed"
[master 463dc4f] Story 182: Fix benchmarks for speed
 2 files changed, 3 insertions(+)
 create mode 100644 README
```



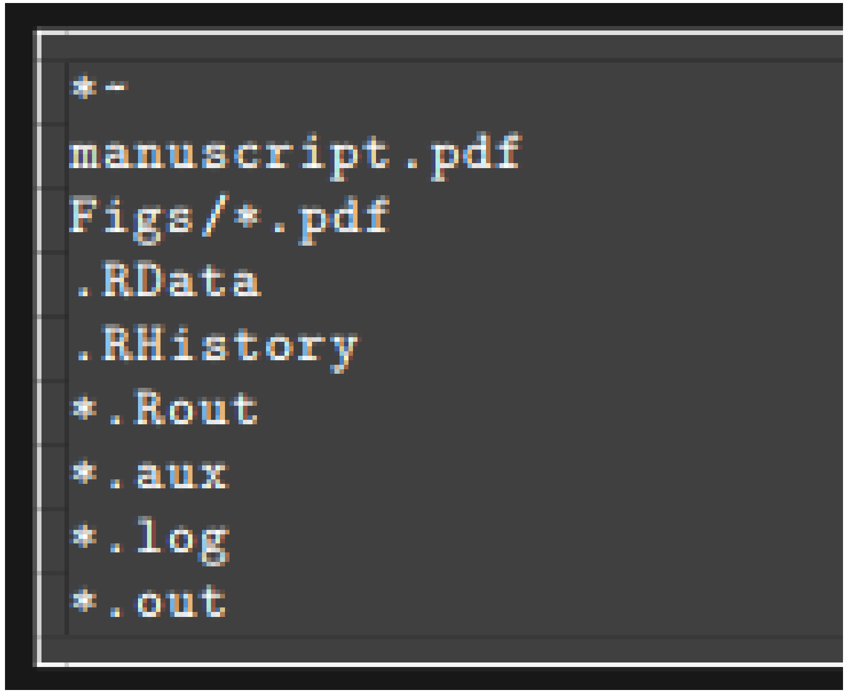
Unmodified/modified Files

Staged Files

Committed Files

Git : .gitignore file

- To indicate files to be ignored
  - For example, no need to track *.o & *.a file…
    => *.[oa]
  - Some pattern:
    - # : comment
    - ! : not ignore that file

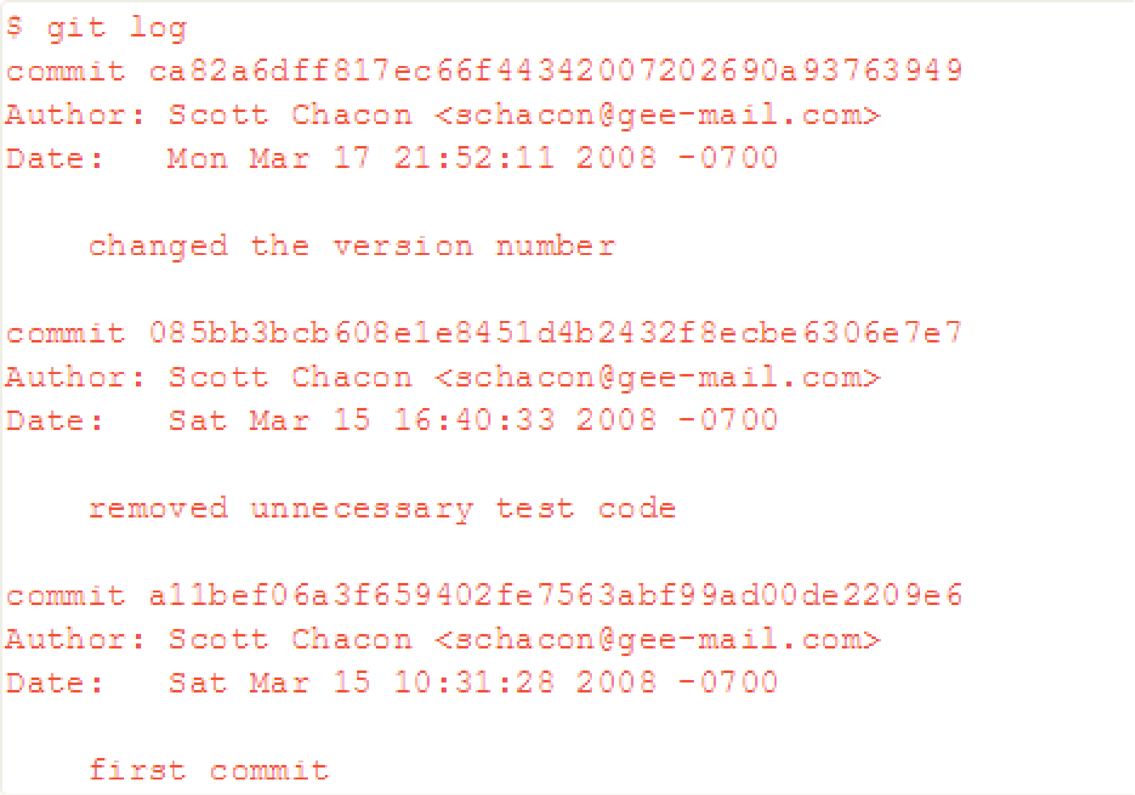


```
*~
manuscript.pdf
Figs/*.pdf
.RData
.RHistory
*.Rout
*.aux
*.log
*.out
```

Git: log

- To see a commit history

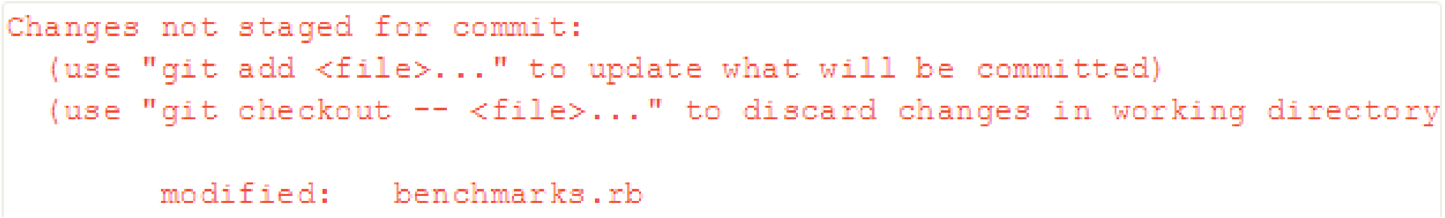
- Some options
  - -p : show the diff results
  - -2 : last 2 results



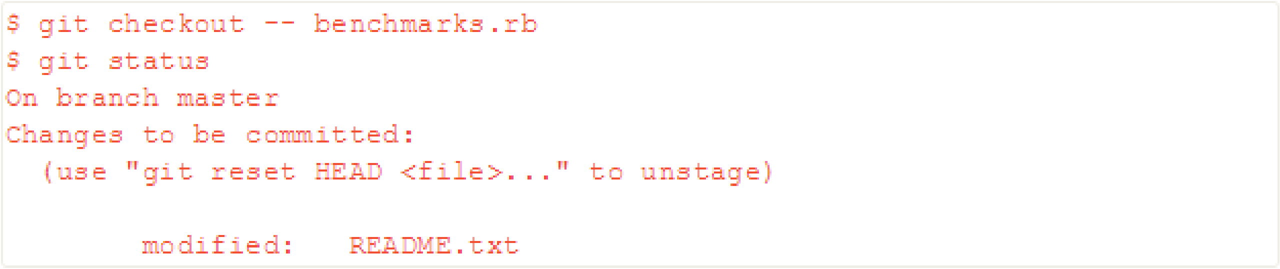
```
$ git log
commit ca82a6dff817ec66f44342007202690a93763949
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Mon Mar 17 21:52:11 2008 -0700

    changed the version number

commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Sat Mar 15 16:40:33 2008 -0700

    removed unnecessary test code

commit a11bef06a3f659402fe7563abf99ad00de2209e6
Author: Scott Chacon <schacon@gee-mail.com>
Date:   Sat Mar 15 10:31:28 2008 -0700

    first commit
```

Git : checkout

1. To discard changes in file



```
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory

        modified:   benchmarks.rb
```



```
$ git checkout -- benchmarks.rb
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   README.txt
```

2. To change current branch

- git checkout “branch name”

Git: pull and push

- To interact with the remote repository (server)

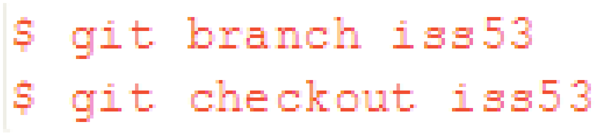
- **Pull** : From remote repo to your local repo
  - git pull : automatically merge your master branch with the remote master branch

- **Push** : your changes to remote branch
  - git push

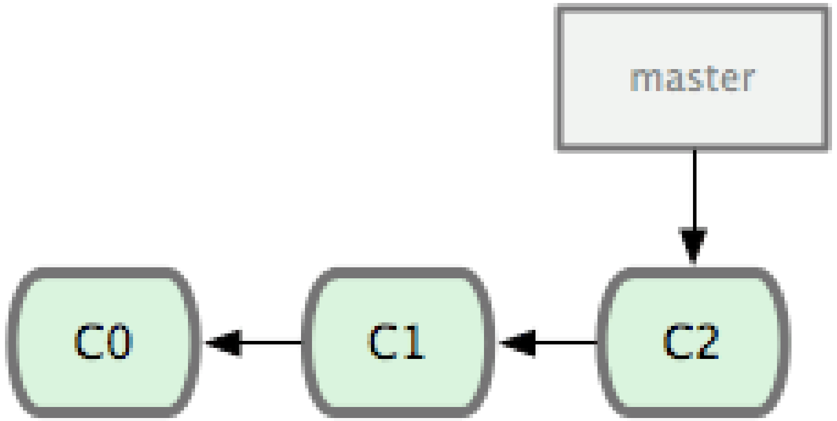
- If you and your partner are in same branch, you should pull before pushing to remote

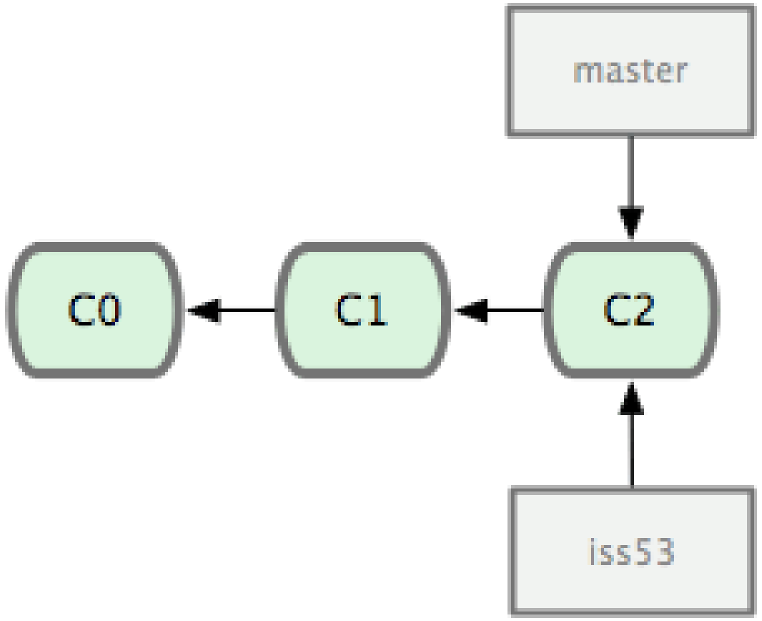
Git : branch (1/2)

- branch : make new branch
- checkout : change current branch

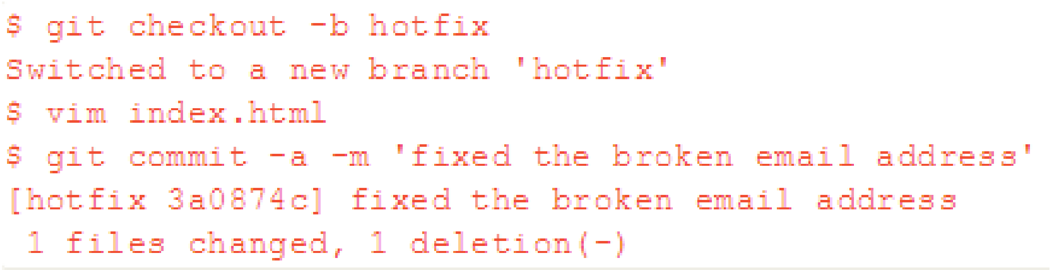


```
$ git branch iss53
$ git checkout iss53
```

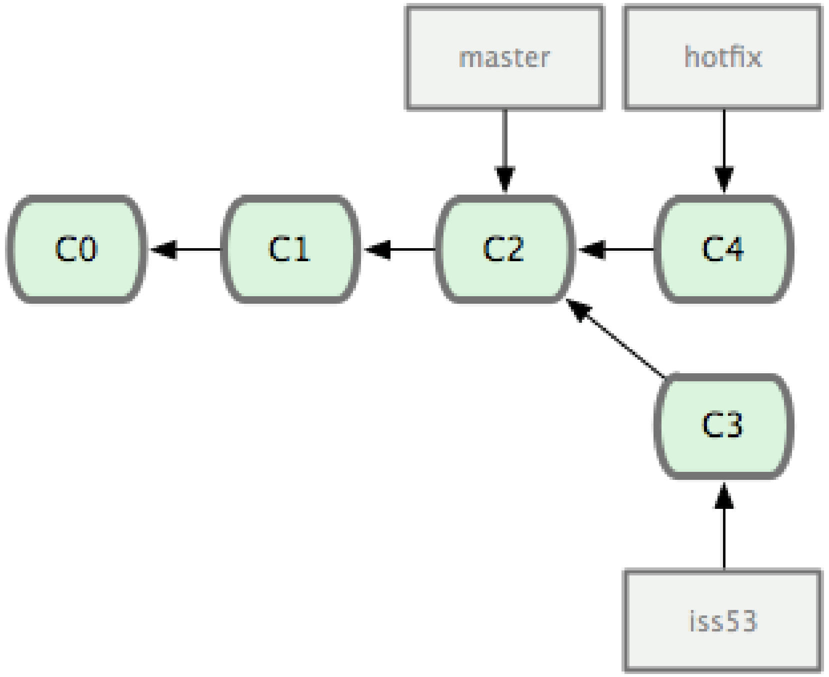




Git : branch (2/2)

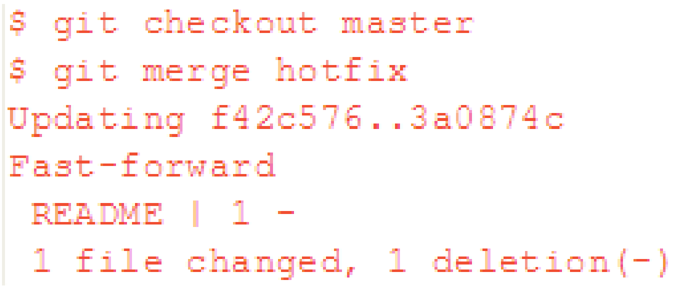


```
$ git checkout -b hotfix
Switched to a new branch 'hotfix'
$ vim index.html
$ git commit -a -m 'fixed the broken email address'
[hotfix 3a0874c] fixed the broken email address
 1 files changed, 1 deletion(-)
```

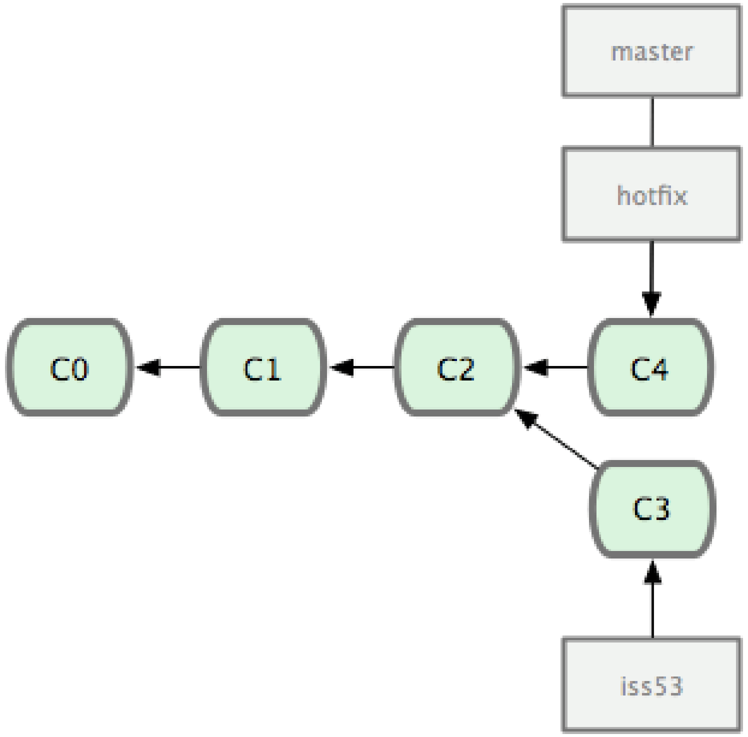


Git : merge (1/3)

- 1. Fast-forward : master branch just takes changes &
  master branch pointer just go forward

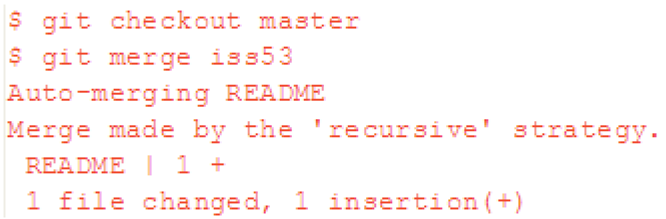


```
$ git checkout master
$ git merge hotfix
Updating f42c576..3a0874c
Fast-forward
 README | 1 -
 1 file changed, 1 deletion(-)
```

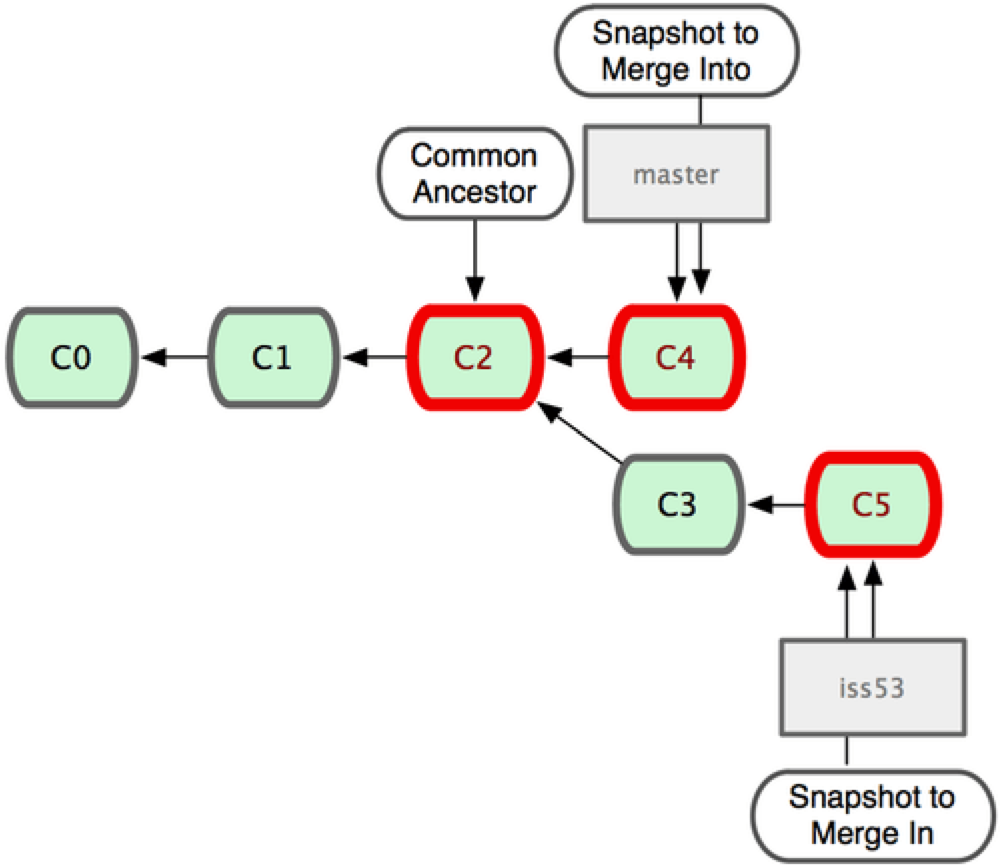


Git : merge (2/3)

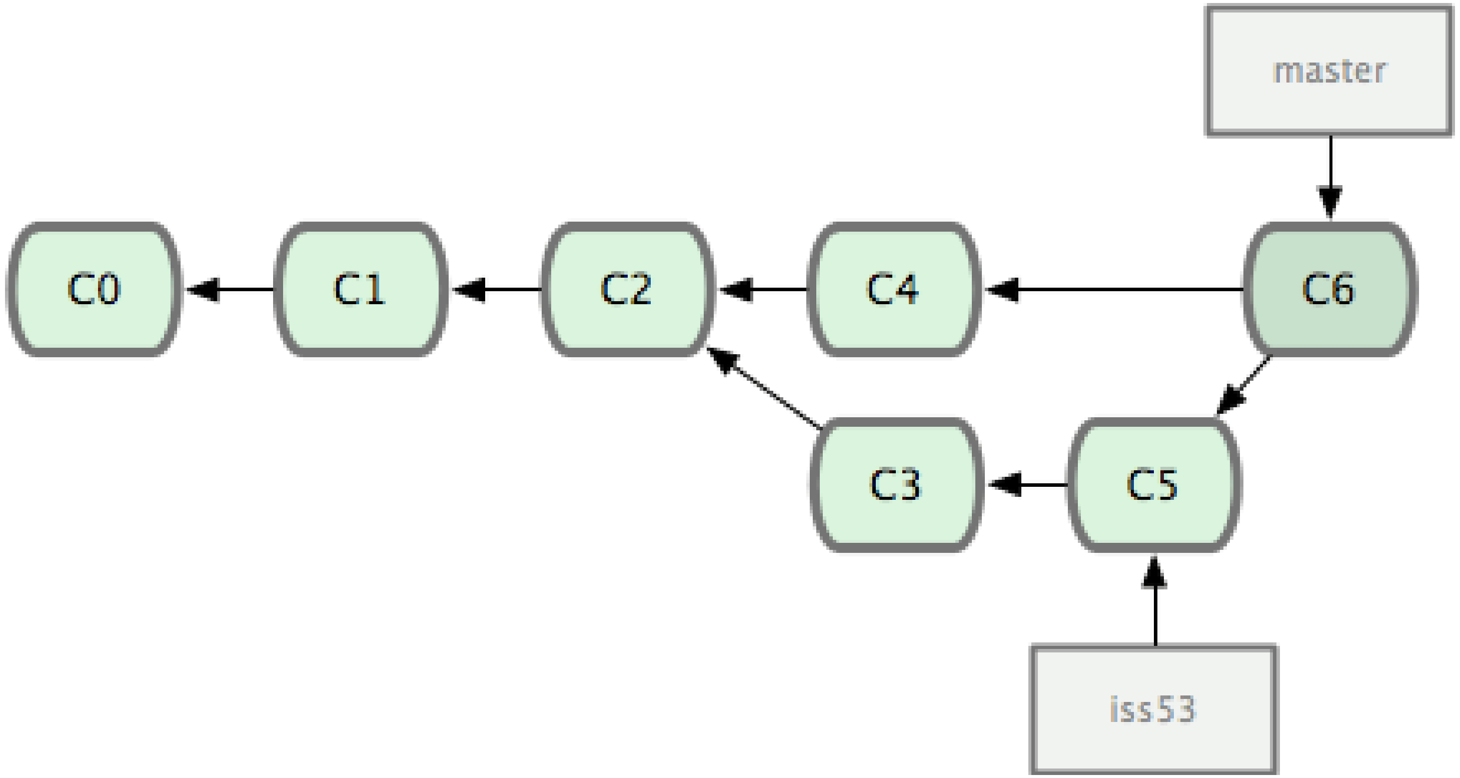
- 2. 3 way merge



```
$ git checkout master
$ git merge iss53
Auto-merging README
Merge made by the 'recursive' strategy.
 README | 1 +
 1 file changed, 1 insertion(+)
```

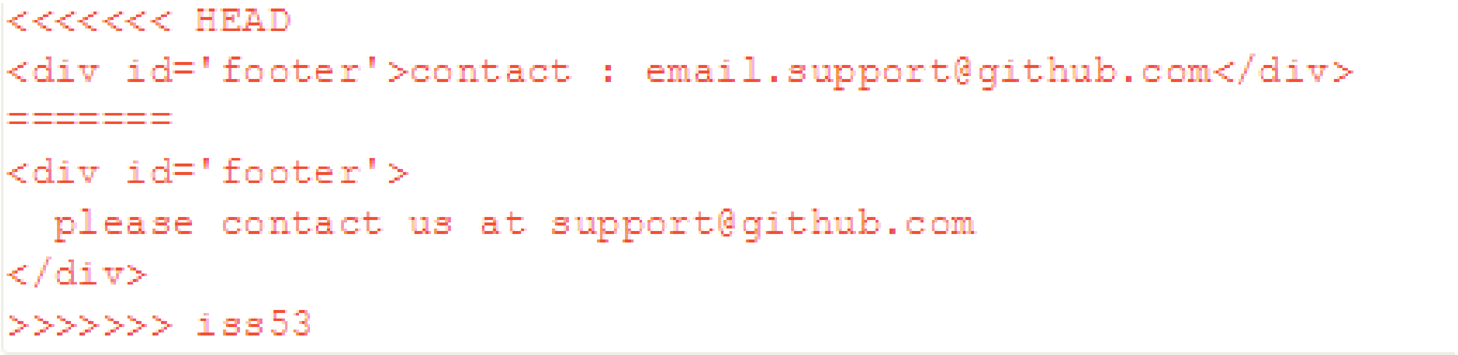


Git : merge (3/3)



Git : merge conflicts

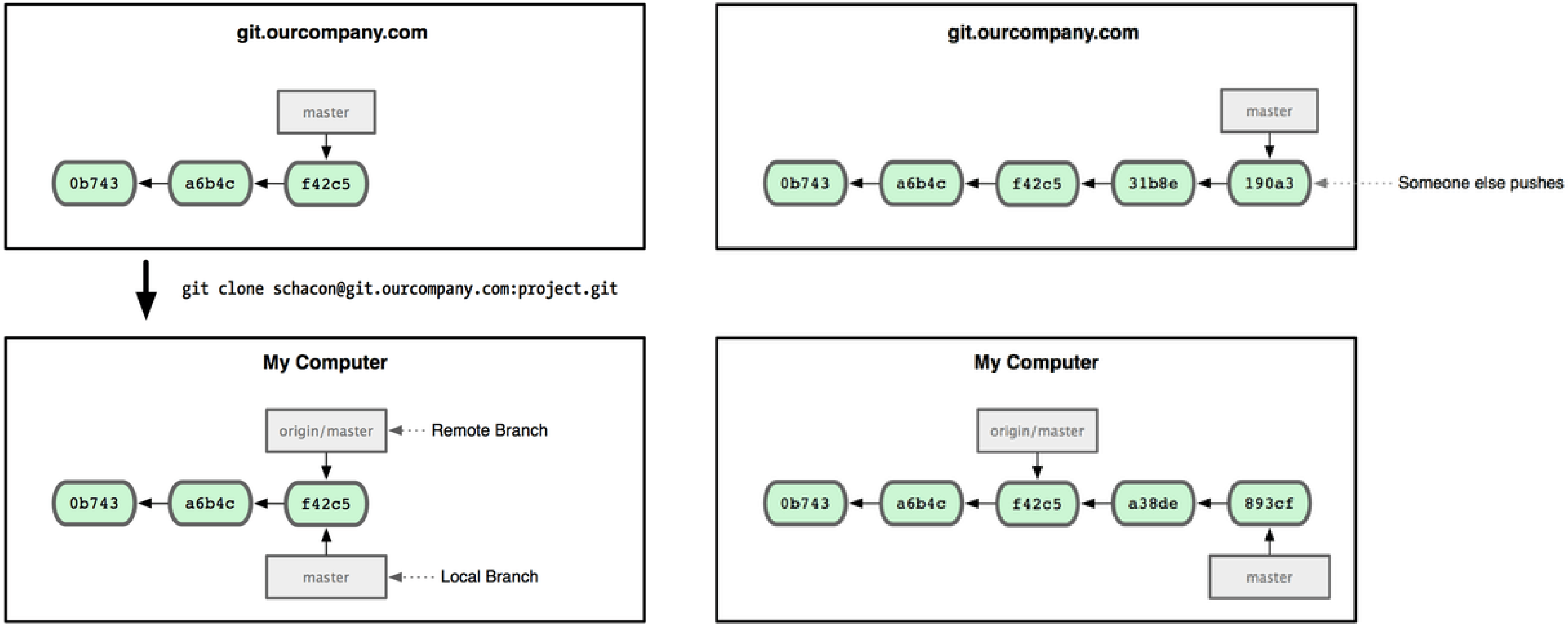
- The conflicting file will contain <<< and >>> sections to indicate where Git was unable to resolve a conflict



```
<<<<<<< HEAD
<div id='footer'>contact : email.support@github.com</div>
=======
<div id='footer'>
  please contact us at support@github.com
</div>
>>>>>>> iss53
```

- Find all such sections, and edit them to merge

Git : remote branch



Git : other basic operations

- git diff : show the diffrences between staging area files and working directory files
- git branch -d “branch name” : delete branch
- giv mv “file” “new file” : move file and that file is staged
- git rm “file” : delete file and that file is staged

Git reference

- Eng : https://git-scm.com/book/en/v2

- Kor : https://git-scm.com/book/ko/v2

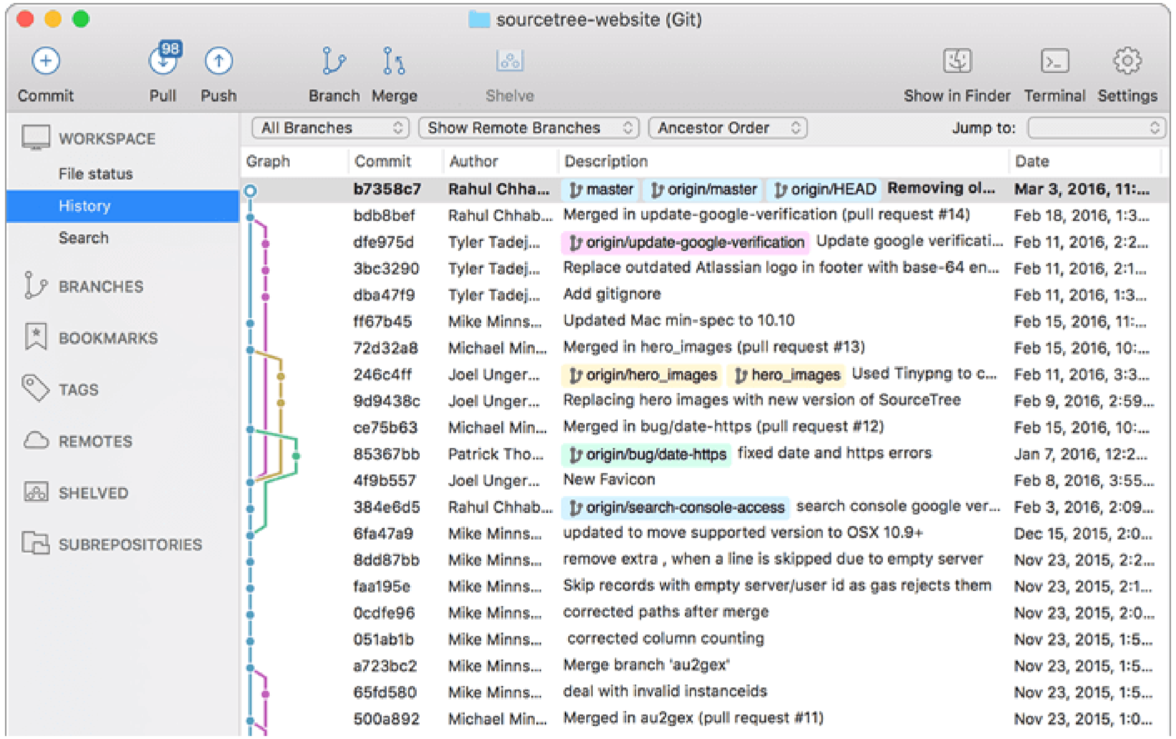
Git : Remote repo

- **GitHub** : Site for online storage of Git repositories
  - you can create a remote repo there
  - get free space for open source projects or pay for private projects

- BitBucket, GitLab etc…

Git GUI client :

- gitk, git gui
  - https://git-scm.com/book/en/v2/Appendix-A%3A-Git-in-Other-Environments-Graphical-Interfaces
- Sourcetree : for window, mac
- GitKraken : for linux



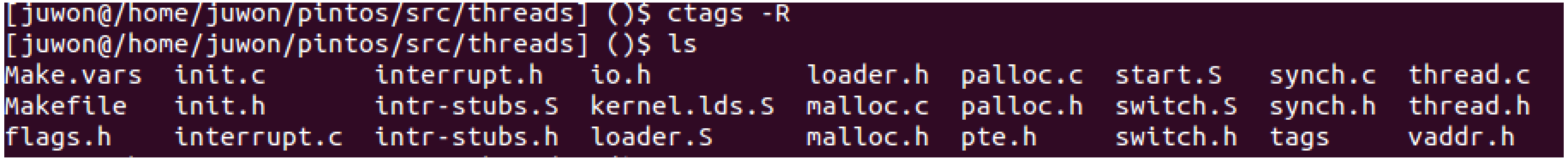
Graph	Commit	Author	Description	Date
	b7358c7	Rahul Chha...	master origin/master origin/HEAD Removing ol...	Mar 3, 2016, 11:...
	bdb8bef	Rahul Chhab...	Merged in update-google-verification (pull request #14)	Feb 18, 2016, 1:3...
	dfe975d	Tyler Tadej...	origin/update-google-verification Update google verificati...	Feb 11, 2016, 2:2...
	3bc3290	Tyler Tadej...	Replace outdated Atlassian logo in footer with base-64 en...	Feb 11, 2016, 2:1...
	dba47f9	Tyler Tadej...	Add gitignore	Feb 11, 2016, 1:3...
	ff67b45	Mike Minns...	Updated Mac min-spec to 10.10	Feb 15, 2016, 11:...
	72d32a8	Michael Min...	Merged in hero_images (pull request #13)	Feb 15, 2016, 10:...
	246c4ff	Joel Unger...	origin/hero_images hero_images Used Tinypng to c...	Feb 11, 2016, 3:3...
	9d9438c	Joel Unger...	Replacing hero images with new version of SourceTree	Feb 9, 2016, 2:59...
	ce75b63	Michael Min...	Merged in bug/date-https (pull request #12)	Feb 15, 2016, 10:...
	85367bb	Patrick Tho...	origin/bug/date-https fixed date and https errors	Jan 7, 2016, 12:2...
	4f9b557	Joel Unger...	New Favicon	Feb 8, 2016, 3:55...
	384e6d5	Rahul Chhab...	origin/search-console-access search console google ver...	Feb 3, 2016, 2:09...
	6fa47a9	Mike Minns...	updated to move supported version to OSX 10.9+	Dec 15, 2015, 2:0...
	8dd87bb	Mike Minns...	remove extra , when a line is skipped due to empty server	Nov 23, 2015, 2:2...
	faa195e	Mike Minns...	Skip records with empty server/user id as gas rejects them	Nov 23, 2015, 2:1...
	0cdfe96	Mike Minns...	corrected paths after merge	Nov 23, 2015, 2:0...
	051ab1b	Mike Minns...	corrected column counting	Nov 23, 2015, 1:5...
	a723bc2	Mike Minns...	Merge branch 'au2gex'	Nov 23, 2015, 1:5...
	65fd580	Mike Minns...	deal with invalid instanceids	Nov 23, 2015, 1:5...
	500a892	Michael Min...	Merged in au2gex (pull request #11)	Nov 23, 2015, 1:0...

Ctags

- To make tags file for programing source code (make index for variables…)
- You can directly jump to function definition or variable declaration
- How to install
  - Ubuntu : sudo apt-get install ctags

Ctags : make tags file

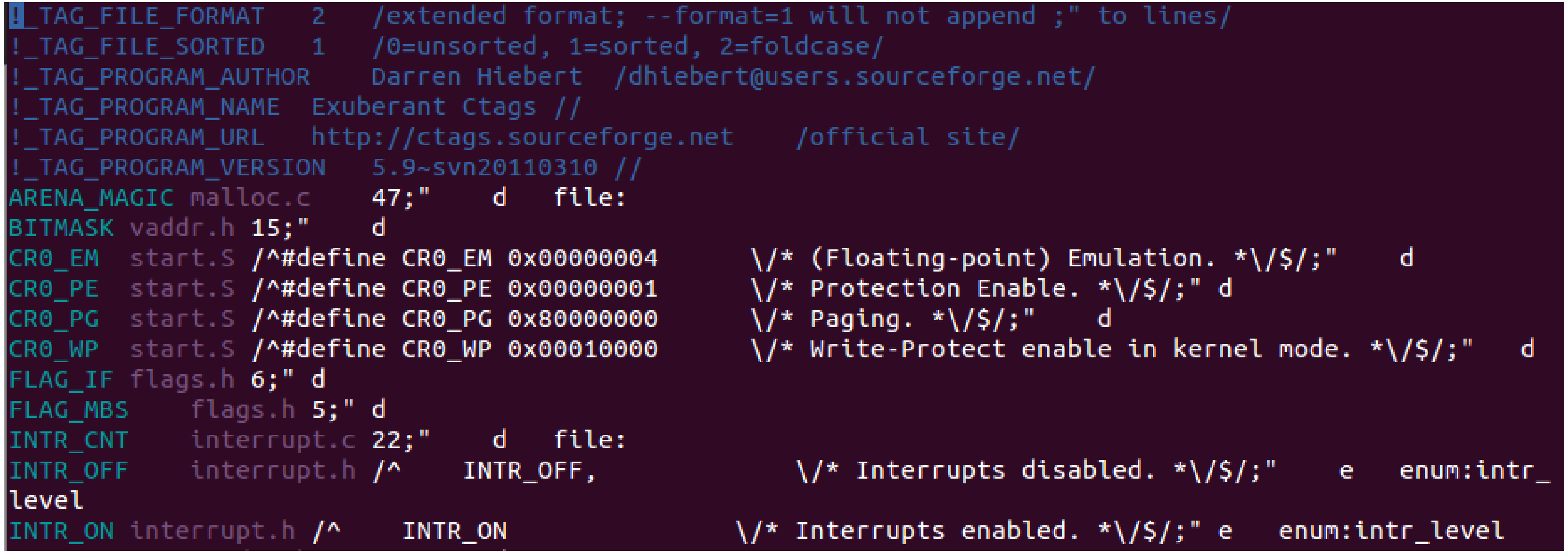
- ctags * : make tags in the current directory
- ctags -R : include the sub directory



```
[juwon@/home/juwon/pintos/src/threads] ()$ ctags -R
[juwon@/home/juwon/pintos/src/threads] ()$ ls
Make.vars   init.c       interrupt.h   io.h           loader.h   palloc.c   start.S    synch.c   thread.c
Makefile    init.h       intr-stubs.S  kernel.lds.S   malloc.c   palloc.h   switch.S   synch.h   thread.h
flags.h     interrupt.c  intr-stubs.h  loader.S       malloc.h   pte.h      switch.h   tags      vaddr.h
```

- :tj “tagname” : (in tags file) move to the tag
- :po : back to tags file

Ctags : Example tags file



```
!_TAG_FILE_FORMAT   2   /extended format; --format=1 will not append ;" to lines/
!_TAG_FILE_SORTED   1   /0=unsorted, 1=sorted, 2=foldcase/
!_TAG_PROGRAM_AUTHOR    Darren Hiebert  /dhiebert@users.sourceforge.net/
!_TAG_PROGRAM_NAME  Exuberant Ctags //
!_TAG_PROGRAM_URL   http://ctags.sourceforge.net    /official site/
!_TAG_PROGRAM_VERSION   5.9~svn20110310 //
ARENA_MAGIC malloc.c    47;"    d   file:
BITMASK vaddr.h 15;"    d
CR0_EM  start.S /^#define CR0_EM 0x00000004      \/* (Floating-point) Emulation. *\/$/;"    d
CR0_PE  start.S /^#define CR0_PE 0x00000001      \/* Protection Enable. *\/$/;" d
CR0_PG  start.S /^#define CR0_PG 0x80000000      \/* Paging. *\/$/;"    d
CR0_WP  start.S /^#define CR0_WP 0x00010000      \/* Write-Protect enable in kernel mode. *\/$/;"   d
FLAG_IF flags.h 6;" d
FLAG_MBS    flags.h 5;" d
INTR_CNT    interrupt.c 22;"    d   file:
INTR_OFF    interrupt.h /^    INTR_OFF,             \/* Interrupts disabled. *\/$/;"    e   enum:intr_
level
INTR_ON interrupt.h /^    INTR_ON               \/* Interrupts enabled. *\/$/;" e   enum:intr_level
```

Ctags : link with vim

- vi ~/.vimrc
- set tags=/home/pintos/pintos/src/threads/tags
- Ctrl +] : same as : tj
- Ctrl + t : same as : po

Ctags : Other operations

- :tnext : if multiple function name, jump to next function
- :tprevious
- :tn : jump to next tag
- :tp : jump to previous tag
- :tr, :tl : first, last tag

Ctags reference

- Reference
  - http://ctags.sourceforge.net/index.html