

# Usage guide for ctags

# What is ctags?

---

- **Unix command that makes database of programming sources's tag (ex. Global variable, function definition, macro declaration)**
- **We can recognize where function and variable are using tags file.**

# Generate tags file

- **Command** `ctags -R *` on `pintos/src` directory.
  - Then `ctags` create tags file.

```
@cspro9:~/pintos/src$ ctags -R *
@cspro9:~/pintos/src$ ls -al
total 232
drwxr-xr-x 12 4096 Oct  5 00:46 .
drwxr-xr-x  3 4096 Aug 28 2011 ..
-rw-r--r--  1 1868 Oct  5 00:46 cscope.files
drwxr-xr-x  2 4096 Sep  8 04:32 devices
drwxr-xr-x  3 4096 Dec 28 2009 examples
drwxr-xr-x  2 4096 Sep  8 04:29 filesys
-rw-r--r--  1  34 Dec 28 2009 .gitignore
drwxr-xr-x  4 4096 Dec 28 2009 lib
-rw-r--r--  1 4621 Dec 28 2009 LICENSE
-rw-r--r--  1 1664 Dec 28 2009 Make.config
-rw-r--r--  1  628 Dec 28 2009 Makefile
-rw-r--r--  1 3810 Dec 28 2009 Makefile.build
-rw-r--r--  1  333 Dec 28 2009 Makefile.kernel
-rw-r--r--  1 1551 Dec 28 2009 Makefile.userprog
drwxr-xr-x  2 4096 Dec 28 2009 misc
-rw-r--r--  1 151552 Oct  5 00:48 tags
drwxr-xr-x  8 4096 Dec 28 2009 tests
drwxr-xr-x  3 4096 Sep 21 02:32 threads
drwxr-xr-x  2 4096 Sep  8 04:29 userprog
drwxr-xr-x  2 4096 Sep 19 00:26 utils
drwxr-xr-x  2 4096 Sep  8 04:29 vm
```

# Register tags file at .vimrc

---

- Add line at .vimrc file

```
71 set tags=./tags;  
72  
~/.vimrc [+]
```

- Then when launch vim, it finds tags file.

# Shortcuts when using ctags on vim

---

- **Ctrl + ]**
  - Go to declaration point of function or variable.
- **Ctrl + t**
  - Go back previous tags or code.
- **Google it to see more commands or shortcuts.**

# Example

```
103 /* Starts preemptive thread scheduling by enabling interrupts.
104    Also creates the idle thread. */
105 void
106 thread_start (void)
107 {
108     /* Create the idle thread. */
109     struct semaphore start_idle;
110     sema_init (&start_idle, 0);
111     thread_create ("idle", PRI_MIN, idle, &start_idle);
112
113     /* Start preemptive thread scheduling. */
114     intr_enable ();
115
116     /* Wait for the idle thread to initialize idle_thread. */
117     sema_down (&start_idle);
118 }
```

Ctrl + ]



```
165 tid_t
166 thread_create (const char *name, int priority,
167               thread_func *function, void *aux)
168 {
169     struct thread *t;
170     struct kernel_thread_frame *kf;
171     struct switch_entry_frame *ef;
172     struct switch_threads_frame *sf;
173     tid_t tid;
174     enum intr_level old_level;
175
176     ASSERT (function != NULL);
177
178     /* Allocate thread. */
179     t = palloc_get_page (PAL_ZERO);
180     if (t == NULL)
181         return TID_ERROR;
182
183     /* Initialize thread. */
184     init_thread (t, name, priority);
185     tid = t->tid = allocate_tid ();
```

# Example

```
165 tid_t
166 thread_create (const char *name, int priority,
167               thread_func *function, void *aux)
168 {
169     struct thread *t;
170     struct kernel_thread_frame *kf;
171     struct switch_entry_frame *ef;
172     struct switch_threads_frame *sf;
173     tid_t tid;
174     enum intr_level old_level;
175
176     ASSERT (function != NULL);
177
178     /* Allocate thread. */
179     t = palloc_get_page (PAL_ZERO);
180     if (t == NULL)
181         return TID_ERROR;
182
183     /* Initialize thread. */
184     init_thread (t, name, priority);
185     tid = t->tid = allocate_tid ();
```

Ctrl + t



```
103 /* Starts preemptive thread scheduling by enabling interrupts.
104    Also creates the idle thread. */
105 void
106 thread_start (void)
107 {
108     /* Create the idle thread. */
109     struct semaphore start_idle;
110     sema_init (&start_idle, 0);
111     thread_create ("idle", PRI_MIN, idle, &start_idle);
112
113     /* Start preemptive thread scheduling. */
114     intr_enable ();
115
116     /* Wait for the idle thread to initialize idle_thread. */
117     sema_down (&start_idle);
118 }
```