Usage guide for ctags

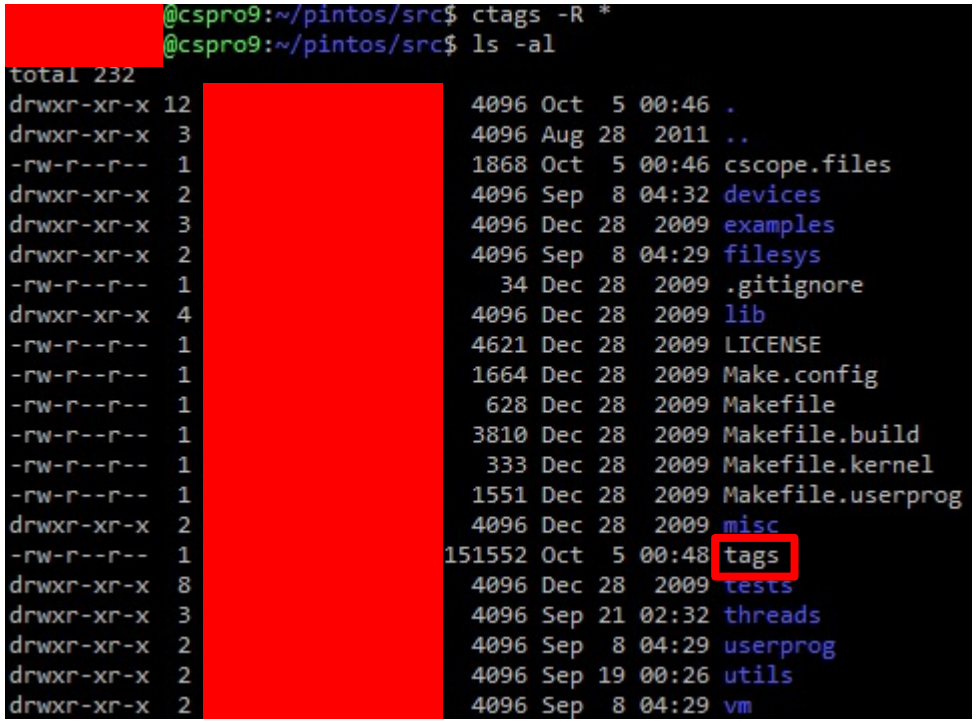
What is ctags?

- **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.**

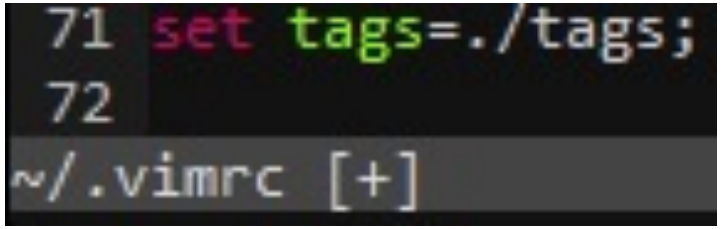
Generate tags file

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



Register tags file at .vimrc

- **Add line at .vimrc file**



- **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 }
```