
Subject: [PATCH][DOCUMENTATION] Minimal controller code for a quick start
Posted by [Pavel Emelianov](#) on Thu, 07 Feb 2008 15:37:17 GMT

[View Forum Message](#) <> [Reply to Message](#)

The Documentation/cgroups.txt file contains the info on how to write some controller for cgroups subsystem, but even with this, one need to write quite a lot of code before developing the core (or copy-n-paste it from some other place).

I propose to put this minimal controller into Documentation directory to let people copy-n-paste a) from a known place and b) a small piece of code.

Besides, many people learn better reading an example rather than/along with a document.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
diff --git a/Documentation/cgroups.txt b/Documentation/cgroups.txt
index 42d7c4c..66068dc 100644
--- a/Documentation/cgroups.txt
+++ b/Documentation/cgroups.txt
@@ -531,6 +531,9 @@ and root cgroup. Currently this will only involve movement between
the default hierarchy (which never has sub-cgroups) and a hierarchy
that is being created/destroyed (and hence has no sub-cgroups).
```

+For a quick start you may want to look at the
+Documentation/controllers/example.c file.

+

4. Questions

=====

```
diff --git a/Documentation/controllers/example.c b/Documentation/controllers/example.c
new file mode 100644
index 0000000..4a73c77
--- /dev/null
+++ b/Documentation/controllers/example.c
@@ -0,0 +1,134 @@
+/*
+ * Documentation/controllers/example.c - A simple controller
+ *
+ * Copy and make s/foo/$SUBSYS_NAME/g in it to get a minimal
+ * working code. Don't forget to add a SUBSYS(foo) line in the
+ * include/linux/cgroup_subsys.h file.
+ *
+ */
```

```

+
+ #include <linux/cgroup.h>
+
+ /*
+  * the foo main structure - it is used to store any info, that
+  * is required from the group of tasks
+  */
+
+ struct foo_cgroup {
+ /*
+  * put your fields here
+  */
+
+ struct cgroup_subsys_state css;
+
+ /*
+  * ... or/and here
+  */
+ };
+
+ /*
+  * helpers to get the foo_cgroup from a task and a control group
+  */
+
+ static inline struct foo_cgroup *foo_from_css(struct cgroup_subsys_state *css)
+ {
+ return container_of(css, struct foo_cgroup, css);
+ }
+
+ static inline struct foo_cgroup *foo_from_cgroup(struct cgroup *cg)
+ {
+ return foo_from_css(cgroup_subsys_state(cg, foo_subsys_id));
+ }
+
+ static inline struct foo_cgroup *foo_from_task(struct task_struct *p)
+ {
+ return foo_from_css(task_subsys_state(p, foo_subsys_id));
+ }
+
+ /*
+  * foo files
+  */
+
+ static ssize_t foo_bar_read(struct cgroup *cg, struct cftype *cft,
+ struct file *file, char __user *userbuf,
+ size_t nbytes, loff_t *ppos)
+ {
+ struct foo_cgroup *foo;

```

```

+
+ foo = foo_from_cgroup(cg);
+
+ /*
+  * produce some output
+  */
+
+ return nbytes;
+}
+
+static ssize_t foo_bar_write(struct cgroup *cg, struct cftype *cft,
+ struct file *file, const char __user *userbuf,
+ size_t nbytes, loff_t *ppos)
+{
+ struct foo_cgroup *foo;
+
+ foo = foo_from_cgroup(cg);
+
+ /*
+  * read and tune the foo
+  */
+
+ return nbytes;
+}
+
+static struct cftype foo_files[] = {
+ {
+ .name = "bar",
+ .read = foo_bar_read,
+ .write = foo_bar_write,
+ },
+};
+
+/*
+ * foo subsystem basic callbacks
+ */
+
+static struct cgroup_subsys_state *foo_create(struct cgroup_subsys *cs,
+ struct cgroup *cg)
+{
+ struct foo_cgroup *foo;
+
+ foo = kmalloc(sizeof(struct foo_cgroup), GFP_KERNEL);
+ if (foo == NULL)
+ return NULL;
+
+ /*
+  * initialize your fields

```

```

+ */
+
+ return &foo->css;
+}
+
+static void foo_destroy(struct cgroup_subsys *cs, struct cgroup *cg)
+{
+ struct foo_cgroup *foo;
+
+ foo = foo_from_cgroup(cg);
+
+ /*
+  * clean your fields
+  */
+
+ kfree(foo);
+}
+
+static int foo_populate(struct cgroup_subsys *cs, struct cgroup *cg)
+{
+ return cgroup_add_files(cg, cs, foo_files, ARRAY_SIZE(foo_files));
+}
+
+struct cgroup_subsys foo_subsys = {
+ .name = "foo",
+ .subsys_id = foo_subsys_id,
+ .create = foo_create,
+ .destroy = foo_destroy,
+ .populate = foo_populate,
+};

```

Containers mailing list
Containers@lists.linux-foundation.org
<https://lists.linux-foundation.org/mailman/listinfo/containers>
