NAME

g_attach, **g_detach** - attach/detach GEOM consumers to/from providers

SYNOPSIS

#include <geom/geom.h>

int
g_attach(*struct g_consumer *cp*, *struct g_provider *pp*);

void
g_detach(*struct g_consumer *cp*);

DESCRIPTION

The **g_attach**() function attaches given consumer *cp* to given provider *pp*, thus establishing a communication channel between the consumer and the provider that allows to change access counts and perform I/O operations.

The **g_detach**() function detaches given consumer *cp* from its corresponding provider, tearing down the communication channel between them.

RESTRICTIONS/CONDITIONS

g_attach():

The consumer must not be attached to a provider.

The operation must not create a topology loop.

The topology lock has to be held.

g_detach():

The consumer has to be attached.

The access count has to be 0.

There must be no active requests.

The topology lock has to be held.

RETURN VALUES

The **g_attach**() function returns 0 if successful; otherwise an error code is returned.

EXAMPLES

Create a consumer, attach it to a given provider, gain read access and clean up.

```
void
some_function(struct g_geom *mygeom, struct g_provider *pp)
{
        struct g_consumer *cp;

        g_topology_assert();

        /* Create new consumer on ’mygeom’ geom. */
        cp = g_new_consumer(mygeom);
        /* Attach newly created consumer to given provider. */
        if (g_attach(cp, pp) != 0) {
                g_destroy_consumer(cp);
                return;
        }
        /* Open provider for reading through our consumer. */
        if (g_access(cp, 1, 0, 0) != 0) {
                g_detach(cp);
                g_destroy_consumer(cp);
                return;
        }

        g_topology_unlock();
        /*
         * Read data from provider.
         */
        g_topology_lock();

        /* Disconnect from provider (release access count). */
        g_access(cp, -1, 0, 0);
        /* Detach from provider. */
        g_detach(cp);
        /* Destroy consumer. */
        g_destroy_consumer(cp);
}
```

ERRORS

Possible errors:

[ELOOP]	The operation creates a topology loop.
[ENXIO]	Provider got orphaned.

SEE ALSO

geom(4), DECLARE_GEOM_CLASS(9), g_access(9), g_bio(9), g_consumer(9), g_data(9), g_event(9), g_geom(9), g_provider(9), g_provider_by_name(9), g_wither_geom(9)

AUTHORS

This manual page was written by Pawel Jakub Dawidek <*pjd@FreeBSD.org*>.