NAME
>   gdbus − Tool for working with D−Bus objects

SYNOPSIS
>   **gdbus** introspect [−−system | −−session | −−address *address*] −−dest *bus_name*
>   −−object−path */path/to/object* [−−xml] [−−recurse] [−−only−properties]
>
>   **gdbus** monitor [−−system | −−session | −−address *address*] −−dest *bus_name*
>   [−−object−path */path/to/object*]
>
>   **gdbus** call [−−system | −−session | −−address *address*] −−dest *bus_name* −−object−path */path/to/object*
>   −−method *org.project.InterfaceName.MethodName* [−−timeout *seconds*] ARG1 ARG2...
>
>   **gdbus** emit [−−system | −−session | −−address *address*] −−object−path */path/to/object*
>   −−signal *org.project.InterfaceName.SignalName* [−−dest *unique_bus_name*] ARG1 ARG2...
>
>   **gdbus** wait [−−system | −−session | −−address *address*] −−activate *bus_name* [−−timeout *seconds*]
>   *bus_name*
>
>   **gdbus** help

DESCRIPTION
>   **gdbus** is a simple tool for working with D−Bus objects.

COMMANDS
>   **introspect**
>>   Prints out interfaces and property values for a remote object. For this to work, the owner of the object
>>   needs to implement the org.freedesktop.DBus.Introspectable interface. If the **−−xml** option is used, the
>>   returned introspection XML is printed, otherwise a parsed pretty representation is printed. The
>>   **−−recurse** option can be used to introspect children (and their children and so on) and the
>>   **−−only−properties** option can be used to only print the interfaces with properties.
>
>   **monitor**
>>   Monitors one or all objects owned by the owner of *bus_name*.
>
>   **call**
>>   Invokes a method on a remote object. Each argument to pass to the method must be specified as a
>>   serialized **GVariant** except that strings do not need explicit quotes. The return values are printed out
>>   as serialized **GVariant** values.
>
>   **emit**
>>   Emits a signal. Each argument to include in the signal must be specified as a serialized **GVariant**
>>   except that strings do not need explicit quotes.
>
>   **wait**
>>   Waits until *bus_name* is owned by some process on the bus. If the **−−activate** is specified, that bus
>>   name will be auto−started first. It may be the same as the bus name being waited for, or different.
>
>   **help**
>>   Prints help and exit.

BASH COMPLETION
>   **gdbus** ships with a bash completion script to complete commands, destinations, bus names, object paths
>   and interface/method names.

EXAMPLES
>   This shows how to introspect an object − note that the value of each
>   property is displayed:

> $ gdbus introspect −−system \
>   −−dest org.freedesktop.NetworkManager \
>   −−object−path /org/freedesktop/NetworkManager/Devices/0
>   node /org/freedesktop/NetworkManager/Devices/0 {

```
interface org.freedesktop.DBus.Introspectable {
 methods:
   Introspect(out s data);
};
interface org.freedesktop.DBus.Properties {
 methods:
   Get(in  s interface,
      in  s propname,
      out v value);
   Set(in  s interface,
      in  s propname,
      in  v value);
   GetAll(in  s interface,
       out a{sv} props);
};
interface org.freedesktop.NetworkManager.Device.Wired {
 signals:
   PropertiesChanged(a{sv} arg_0);
 properties:
   readonly b Carrier = false;
   readonly u Speed = 0;
   readonly s HwAddress = '00:1D:72:88:BE:97';
};
interface org.freedesktop.NetworkManager.Device {
 methods:
   Disconnect();
 signals:
   StateChanged(u arg_0,
           u arg_1,
           u arg_2);
 properties:
   readonly u DeviceType = 1;
   readonly b Managed = true;
   readwrite o Ip6Config = '/';
   readwrite o Dhcp4Config = '/';
   readwrite o Ip4Config = '/';
   readonly u State = 2;
   readwrite u Ip4Address = 0;
   readonly u Capabilities = 3;
   readonly s Driver = 'e1000e';
   readwrite s Interface = 'eth0';
   readonly s Udi = '/sys/devices/pci0000:00/0000:00:19.0/net/eth0';
 };
};
```

The **−−recurse** and **−−only−properties** options can be useful when wanting to inspect all objects owned by a particular process:

```
$ gdbus introspect −−system −−dest org.freedesktop.UPower −−object−path / −−recurse  −−only−properties
node / {
 node /org {
  node /org/freedesktop {
   node /org/freedesktop/UPower {
    interface org.freedesktop.UPower {
      properties:
```

```
            readonly b IsDocked = true;
            readonly b LidForceSleep = false;
            readonly b LidIsPresent = false;
            readonly b LidIsClosed = false;
            readonly b OnLowBattery = false;
            readonly b OnBattery = false;
            readonly b CanHibernate = true;
            readonly b CanSuspend = true;
            readonly s DaemonVersion = '0.9.10';
          };
        node /org/freedesktop/UPower/Policy {
        };
        node /org/freedesktop/UPower/Wakeups {
         interface org.freedesktop.UPower.Wakeups {
          properties:
            readonly b HasCapability = true;
         };
        };
      };
    };
  };
};
```

In a similar fashion, the **introspect** command can be used to learn details about the Notify method:

```
[...]
  interface org.freedesktop.Notifications {
   methods:
     GetServerInformation(out s return_name,
                   out s return_vendor,
                   out s return_version,
                   out s return_spec_version);
     GetCapabilities(out as return_caps);
     CloseNotification(in  u id);
     Notify(in  s app_name,
         in  u id,
         in  s icon,
         in  s summary,
         in  s body,
         in  as actions,
         in  a{sv} hints,
         in  i timeout,
         out u return_id);
  };
[...]
```

With this information, it's easy to use the **call** command to display a notification

```
$ gdbus call −−session \
        −−dest org.freedesktop.Notifications \
        −−object−path /org/freedesktop/Notifications \
        −−method org.freedesktop.Notifications.Notify \
        my_app_name \
        42 \
        gtk−dialog−info \
        "The Summary" \
```

            "Here's the body of the notification" \
            [] \
            {} \
            5000
    (uint32 12,)

    Monitoring all objects on a service:

    $ gdbus monitor −−system −−dest org.freedesktop.ConsoleKit
    Monitoring signals from all objects owned by org.freedesktop.ConsoleKit
    The name org.freedesktop.ConsoleKit is owned by :1.15
    /org/freedesktop/ConsoleKit/Session2: org.freedesktop.ConsoleKit.Session.ActiveChanged (false,)
    /org/freedesktop/ConsoleKit/Seat1: org.freedesktop.ConsoleKit.Seat.ActiveSessionChanged (",)
    /org/freedesktop/ConsoleKit/Session2: org.freedesktop.ConsoleKit.Session.ActiveChanged (true,)
    /org/freedesktop/ConsoleKit/Seat1: org.freedesktop.ConsoleKit.Seat.ActiveSessionChanged ('/org/freedesktop/ConsoleKi

    Monitoring a single object on a service:

    $ gdbus monitor −−system −−dest org.freedesktop.NetworkManager −−object−path /org/freedesktop/NetworkManager/Ac
    Monitoring signals on object /org/freedesktop/NetworkManager/AccessPoint/4141 owned by org.freedesktop.NetworkMa
    The name org.freedesktop.NetworkManager is owned by :1.5
    /org/freedesktop/NetworkManager/AccessPoint/4141: org.freedesktop.NetworkManager.AccessPoint.PropertiesChanged
    /org/freedesktop/NetworkManager/AccessPoint/4141: org.freedesktop.NetworkManager.AccessPoint.PropertiesChanged
    /org/freedesktop/NetworkManager/AccessPoint/4141: org.freedesktop.NetworkManager.AccessPoint.PropertiesChanged
    /org/freedesktop/NetworkManager/AccessPoint/4141: org.freedesktop.NetworkManager.AccessPoint.PropertiesChanged

    Emitting a signal:

    $ gdbus emit −−session −−object−path /foo −−signal org.bar.Foo "['foo', 'bar', 'baz']"

    Emitting a signal to a specific process:

    $ gdbus emit −−session −−object−path /bar −−signal org.bar.Bar someString −−dest :1.42

    Waiting for a well−known name to be owned on the bus; this will *not* auto−start the service:

    $ gdbus wait −−session org.bar.SomeName

    Auto−starting then waiting for a well−known name to be owned on the bus:

    $ gdbus wait −−session −−activate org.bar.SomeName

    Auto−starting a different service, then waiting for a well−known name to be owned on the bus. This is
    useful in situations where *SomeName* is not directly activatable:

    $ gdbus wait −−session −−activate org.bar.PrerequisiteName org.bar.SomeName

    Waiting for a well−known name and giving up after 30 seconds. By default, the timeout is disabled; or set
    −−timeout to 0 to disable it:

    $ gdbus wait −−session −−timeout 30 org.bar.SomeName

BUGS
    Please send bug reports to either the distribution bug tracker or the upstream bug tracker at
    https://gitlab.gnome.org/GNOME/glib/issues/new.

SEE ALSO
    dbus-send(1)