Class 05: Inter-Application Communication with D-Bus (1): The Basics

Held: Thursday, 21 February 2013

Summary: We begin our exploration of D-Bus.

Related Pages:

* EBoard.

Notes:

* We'll spend a few minutes talking about the homework.
* The next homework can be found at the end of the outline.
* Reminder: Summer research applications are due this Friday. Your first priority should be to get the division application in on time. My application can probably come in any time over the weekend. Interviews should happen by some time next week.
* Just in case you didn't see the notice: Career Chat with Board of Trustee Member Atul Gupta â88 Thursday, Feb. 21 at 12:10 p.m. in JRC 224A (Marketplace Dining Room).

Overview:

* A brief introduction to inter-application communication
* A brief introduction to D-Bus
* Playing with D-Bus
* Homework assignment

A Quick Overview of Inter-Application Communication

* It can be useful for applications and services to talk to each other.
* For example, one way to implement "Send this document" in a word processing program is to have it communicate with an email program.
* Similarly, one graphics application might query another about some global setting.
* The issue of having applications talk to each other is typically called "inter-application communication".
* Inter-application communication can be particularly useful for the kinds of application scripting we do,
  * Rather than having n application send commands and queries to another application, our scripts send commands and queries.
  * Once we've taught our language (or users) how to send commands and queries, they can do so in many situations.
* Unfortunately, there are few, if any, common infrastructures for inter-application communication. It

is often dealt with on an ad-hoc basis.
- *What characteristics would you expect an inter-application system to have?*

A Quick Overview of D-Bus

- Today, we'll look primarily at the D-Bus inter-application communication model.
  - ○ Next week, we'll write a server.
  - ○ Two weeks later, we'll write a client.
- Two "buses" for sending messages.
  - ○ System bus provides general system services
  - ○ Session bus provides user-specific or session-specific services.
- Object-oriented.
  - ○ A service provides a (dynamic) collection of objects.
  - ○ Each object has one or more interfaces (lists of capabilities).
  - ○ Each interface specifies methods (functions), properties (fields, variables), and signals (backwards messages).

Playing with D-Bus

- Unfortunately, we don't have many D-Bus-enabled applications running in MathLAN right now, so we'll have to do some ad-hoc approaches.
  - ○ We also don't have all the tools I'd like, so things may be slightly complicated.
- I've compiled the sample D-Bus server from
  `http://developer.gnome.org/gio/2.28/GDBusConnection.html`.
- That service is at
- We'll use `qdbus` and `gdbus` to communicate with the server.
- We use `qdbus` to see what's available
  `$ qdbus`
- We can use `gdbus introspect` to find out more about the sample server
  `$ gdbus introspect -e -d org.gtk.GDBus.TestServer -o`
  `/org/gtk/GDBus/TestObject`
  - ○ We may have to do a little work to find the object path.
- We can monitor messages witha
  `gdbus monitor org.gtk.GDBus.TestInterface`

Homework

1. Look at the sample D-Bus server (Example 2) at
`http://developer.gnome.org/gio/2.28/GDBusConnection.html`. I do not expect you to understand all of the code (or even much of it).

2. Pick some part of the code that you think you can understand and be ready to explain it. You might choose a chunk of code, a function, or even a high-level overview of the code. I'd prefer that you choose something that you think will challenge you.

3. Pick some part of the code that looks important but that you can't understand. Email me to let me know what it is. (Send email by Tuesday night.)