Task Class

Task is a base class for multi-part, asynchronous tasks. **Task** provides two main benefits to its subclasses:

- **Task** objects let you break a task into smaller components and handle them separately. Each **Task** object can have child **Task** objects that handle a smaller part of a job. (These children can in turn have their own **Task** children). **Task** objects handle cleanup of their children when they complete or are abandoned. Therefore, management of a task is relatively simple because only the top level **Task** object need be controlled.
- **Task** objects are multithreaded state machines, and good for handling very asynchronous tasks, such as sending and receiving XMPP stanzas. In fact, libjingle defines a specialized task type, **XmppTask**, for sending and receiving XMPP messages.

All tasks are managed by the **TaskRunner** class. A **Task** object should either be the child of an object that implements **TaskRunner**, or it should implement **TaskRunner** itself. To start a task, call its **Start** method, after calling any required methods to set required values. Example task-based objects include **SessionSendTask**, which sends an XMPP message and waits for a reply, and **Receiver**, which listens to all incoming XMPP stanzas.

A **Task** implementation overrides the base **ProcessStart** method, which returns a value indicating the status of the process. When the task returns STATE_DONE, it is complete, and the **TaskRunner** object will delete it from the queue. See the code comments in task.h for more information. Example **Task** objects include **Receiver** and **SessionSendTask**. Example **TaskRunner** objects include **XmppPump**.

The basic state values that a **Task** object can return are defined by the following nameless enum in task.h

```
enum {
  STATE_BLOCKED = -1,
  STATE_INIT = 0,
  STATE_START = 1,
  STATE_DONE = 2,
  STATE_ERROR = 3,
  STATE_RESPONSE = 4,
  STATE_NEXT = 5, // Subclasses which need more states start here and higher
  };
```

Additional state values can be defined by overriding classes.

XmppTask is a specialized subclass that handles incoming XMPP stanzas. The task manager for all **XmppTask** objects is **XmppClient**. When **XmppClient** receives a new XMPP stanza from the network, it queries each of its child **XmppTask** objects to see which one can handle it. Child tasks are queried in order of the priority hard-coded in their constructor code. As soon as one of the child tasks indicates that it will handle the stanza (by returning true for **HandleStanza**) **XmppClient** will stop querying the other children.

Syntax

```
class Task
```

Signals

SignalTimeout

Sent when the task times out. Sets the state to STATE_DONE.

Attributes: public

Declaration file: talk/base/task.h

Last updated March 23, 2012.