

ARC::FIO Reference Manual

Generated by Doxygen 1.3.5

Mon Jan 10 22:59:06 2005

Contents

1	ARC::FIO Class Index	1
1.1	ARC::FIO Class List	1
2	ARC::FIO Class Documentation	3
2.1	FIOClient Class Reference	3
2.2	FIOMessage Class Reference	4
2.3	FIOObject Class Reference	6
2.4	FIORegistration Class Reference	9
2.5	FIOService Class Reference	10
2.6	FIOServiceRemote Class Reference	12
2.7	HTTP_FIO Class Reference	13
2.8	XMLNode Class Reference	14

Chapter 1

ARC::FIO Class Index

1.1 ARC::FIO Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

FIOClient (Interface to remote service)	3
FIOMessage (Message sent between services for notification)	4
FIOObject	6
FIORegistration (Registration request)	9
FIOService (Kind of main class. Represents local service itself)	10
FIOServiceRemote (Interface to remote service)	12
HTTP_FIO (FIO service as seen by connected client (interface to FIOService class))	13
XMLNode	14

Chapter 2

ARC::FIO Class Documentation

2.1 FIOClient Class Reference

Interface to remote service.

```
#include <client.h>
```

Public Member Functions

- **FIOClient** (pFIOServiceRemote *s*)
- **FIOClient** ([FIOServiceRemote](#) &*s*)
- bool [registr](#) ([FIOService](#) &*s*)
- bool [notify](#) (std::list< [FIOMessage](#) > &*msgs*)

2.1.1 Detailed Description

Interface to remote service.

2.1.2 Member Function Documentation

2.1.2.1 bool FIOClient::notify (std::list< [FIOMessage](#) > & *msgs*)

Send notification to remote service.

2.1.2.2 bool FIOClient::registr ([FIOService](#) & *s*)

Register at remote service.

The documentation for this class was generated from the following file:

- [client.h](#)

2.2 FIOMessage Class Reference

Message sent between services for notification.

```
#include <message.h>
```

Public Member Functions

- **FIOMessage** ([FIOObject::action_t](#) *t*)
- **FIOMessage** (const [FIOObject](#) &o, [FIOObject::action_t](#) *t*)
- void [serialize](#) (std::string &msg)
- void [operator=](#) ([FIOObject::action_t](#) *t*)
- bool [operator==](#) (const std::string &id) const
- bool [operator==](#) (const [FIOObject](#) &o) const
- bool [operator==](#) (const_pFIOObject *p*) const
- [FIOObject::action_t](#) [type](#) (void) const
- const_pFIOObject [object](#) (void) const

Static Public Member Functions

- [FIOMessage](#) * [deserialize](#) (const std::string &msg)

2.2.1 Detailed Description

Message sent between services for notification.

2.2.2 Member Function Documentation

2.2.2.1 [FIOMessage](#)* [FIOMessage::deserialize](#) (const std::string & *msg*) [static]

Process received message. Created message have to be deleted.

2.2.2.2 const_pFIOObject [FIOMessage::object](#) (void) const [inline]

Get object associated with message.

2.2.2.3 void [FIOMessage::operator=](#) ([FIOObject::action_t](#) *t*) [inline]

Modify type of message.

2.2.2.4 bool [FIOMessage::operator==](#) (const_pFIOObject *p*) const [inline]

check if message refers to same object

2.2.2.5 bool [FIOMessage::operator==](#) (const [FIOObject](#) & *o*) const [inline]

Check if message refers to same object.

2.2.2.6 `bool FIOMessage::operator==(const std::string & id) const` [inline]

Check if message refers to object with same ID.

2.2.2.7 `void FIOMessage::serialize(std::string & msg)`

Convert message into form suitable for sending through channel.

2.2.2.8 `FIOObject::action_t FIOMessage::type(void) const` [inline]

Get type of message.

The documentation for this class was generated from the following file:

- message.h

2.3 FIOObject Class Reference

```
#include <object.h>
```

Public Types

- enum [type_t](#) { **undefined, reference, ordinary** }
- enum [action_t](#) { **unknown, created, updated, deleted** }

Public Member Functions

- [FIOObject](#) (const std::string &id, [type_t](#) t=undefined)
- [FIOObject](#) ([FIOService](#) &service, const std::string &id, [type_t](#) t=undefined)
- const std::string & [id](#) (void) const
- [type_t](#) [type](#) (void) const
- [FIOService](#) * [service](#) (void)
- virtual [operator bool](#) (void)
- virtual bool [operator!](#) (void)
- virtual void [serialize](#) (std::string &msg) const
- virtual void [serialize](#) (std::string &msg, [FIOMessage](#) &m) const
- virtual void [notification](#) ([FIOMessage](#) &msg, pFIOServiceRemote rsrv)
- virtual void [maintain](#) (void)

Static Public Member Functions

- [FIOObject](#) * [deserialize](#) (const std::string &msg)
- [FIOObject](#) * [deserialize](#) (const std::string &msg, [FIOMessage](#) &m)
- pFIOObject [notification](#) ([FIOService](#) &srv, [FIOMessage](#) &msg, pFIOServiceRemote rsrv)

Protected Member Functions

- virtual void [serialize](#) ([XMLNode](#) &x) const
- virtual void [serialize](#) ([XMLNode](#) &x, [FIOMessage](#) &m) const

Static Protected Member Functions

- [FIOObject](#) * [deserialize](#) (const [XMLNode](#) &x)
- [FIOObject](#) * [deserialize](#) (const [XMLNode](#) &x, [FIOMessage](#) &m)

2.3.1 Detailed Description

Class to represent both reference and ordinary objects with specific methods virtual. Real object reimplements one or another set of them.

2.3.2 Member Enumeration Documentation

2.3.2.1 enum `FIOObject::action_t`

State of object.

2.3.2.2 enum `FIOObject::type_t`

Kind of object.

2.3.3 Constructor & Destructor Documentation

2.3.3.1 `FIOObject::FIOObject (const std::string & id, type_t t = undefined)`

Create orphan object.

2.3.3.2 `FIOObject::FIOObject (FIOService & service, const std::string & id, type_t t = undefined)`

Create owned object.

2.3.4 Member Function Documentation

2.3.4.1 `FIOObject* FIOObject::deserialize (const std::string & msg, FIOMessage & m) [static]`

Process object recieved with message.

2.3.4.2 `FIOObject* FIOObject::deserialize (const std::string & msg) [static]`

Process received object.

2.3.4.3 `FIOObject* FIOObject::deserialize (const XMLNode & x, FIOMessage & m) [static, protected]`

Convenience method to convert XML tree into object.

2.3.4.4 `FIOObject* FIOObject::deserialize (const XMLNode & x) [static, protected]`

Convenience method to convert XML tree into object.

2.3.4.5 `const std::string& FIOObject::id (void) const [inline]`

Get unique id of object.

2.3.4.6 `virtual void FIOObject::maintain (void) [virtual]`

Called periodically by service to which this object belongs.

2.3.4.7 `pFIOObject FIOObject::notification (FIOService & srv, FIOMessage & msg, pFIOServiceRemote rsrv)` [static]

Process received notification about unknown object.

2.3.4.8 `virtual void FIOObject::notification (FIOMessage & msg, pFIOServiceRemote rsrv)` [virtual]

Process received notification about this object.

2.3.4.9 `virtual FIOObject::operator bool (void)` [virtual]

If object is valid.

2.3.4.10 `virtual bool FIOObject::operator! (void)` [virtual]

If object is invalid.

2.3.4.11 `virtual void FIOObject::serialize (std::string & msg, FIOMessage & m) const` [virtual]

Serialize object for including into message.

2.3.4.12 `virtual void FIOObject::serialize (std::string & msg) const` [virtual]

Turn object into format suitable to send over wire.

2.3.4.13 `virtual void FIOObject::serialize (XMLNode & x, FIOMessage & m) const` [protected, virtual]

Convenience method to convert object into XML tree Real object can reimplement it instead of one using string.

2.3.4.14 `virtual void FIOObject::serialize (XMLNode & x) const` [protected, virtual]

Convenience method to convert object into XML tree. Real object can reimplement it instead of one using string.

2.3.4.15 `FIOService* FIOObject::service (void)` [inline]

Get parent service.

2.3.4.16 `type_t FIOObject::type (void) const` [inline]

Get type of object.

The documentation for this class was generated from the following file:

- object.h

2.4 FIORegistration Class Reference

Registration request.

```
#include <message.h>
```

Public Member Functions

- **FIORegistration** ([FIOService](#) &s)
- [FIOService](#) & **service** (void)
- bool **operator==** (const [FIOService](#) &s) const

2.4.1 Detailed Description

Registration request.

The documentation for this class was generated from the following file:

- message.h

2.5 FIOService Class Reference

Kind of main class. Represents local service itself.

```
#include <service.h>
```

Public Member Functions

- **FIOService** (const string &url)
- **operator bool** (void) const
- bool **operator!** (void) const
- const string & **url** (void) const
- void **add** (pFIOObject o)
- void **add** (pFIOServiceRemote s)
- void **notify** (const FIOMessage &m)
- void **notification** (FIOMessage &msg, pFIOServiceRemote &srv)
- pFIOServiceRemote **service** (const std::string &id)
- void **registr** (pFIOServiceRemote s)
- pFIOObject **find** (std::string id)
- void **maintain** (void)

2.5.1 Detailed Description

Kind of main class. Represents local service itself.

2.5.2 Member Function Documentation

2.5.2.1 void FIOService::add (pFIOServiceRemote s)

Add new service to be notified about new objects.

2.5.2.2 void FIOService::add (pFIOObject o)

Add new object to be handled.

2.5.2.3 pFIOObject FIOService::find (std::string id)

Find object by id (NULL if not found).

2.5.2.4 void FIOService::maintain (void)

Called to run internal tasks periodically.

2.5.2.5 void FIOService::notification (FIOMessage & msg, pFIOServiceRemote & srv)

Process recieved notification.

2.5.2.6 void FIOService::notify (const FIOMessage & m)

Inform registered services about changes (calls corresponding methods of hadled objects)

2.5.2.7 void FIOService::registr (pFIOServiceRemote s)

Register itself service to remote service.

2.5.2.8 pFIOServiceRemote FIOService::service (const std::string & id)

Find service in list by id or create a new one.

The documentation for this class was generated from the following file:

- service.h

2.6 FIOServiceRemote Class Reference

Interface to remote service.

```
#include <service.h>
```

Public Member Functions

- **FIOServiceRemote** (const std::string &url, const std::string &id)
- const std::string & **url** (void)
- const std::string & **id** (void)
- void **notify** (const [FIOMessage](#) &m)
- void **registr** ([FIOService](#) &s)
- void **maintain** (void)

2.6.1 Detailed Description

Interface to remote service.

2.6.2 Member Function Documentation

2.6.2.1 void FIOServiceRemote::maintain (void)

Called by parent service to run internal tasks periodically.

2.6.2.2 void FIOServiceRemote::notify (const [FIOMessage](#) & m)

Put messages into a queue (could also send it immediately).

2.6.2.3 void FIOServiceRemote::registr ([FIOService](#) & s)

Register s to this service.

The documentation for this class was generated from the following file:

- service.h

2.7 HTTP_FIO Class Reference

FIO service as seen by connected client (interface to [FIOService](#) class).

```
#include <flo.h>
```

Public Member Functions

- **HTTP_FIO** ([HTTPS_Connector](#) *c, [FIOService](#) &srv)
- virtual [HTTP_Error](#) **get** (const char *uri, int &keep_alive)
- virtual [HTTP_Error](#) **put** (const char *uri, int &keep_alive)
- virtual void **soap_methods** (void)

Friends

- int [nf__notification](#) (struct soap *, int, char **, struct [nf__response](#) &)
- int [nf__registration](#) (struct soap *, char *, char *, char *, struct [nf__response](#) &)

2.7.1 Detailed Description

FIO service as seen by connected client (interface to [FIOService](#) class).

2.7.2 Friends And Related Function Documentation

2.7.2.1 int [nf__notification](#) (struct soap *, int, char **, struct [nf__response](#) &) [friend]

Process notification from sent by other service.

2.7.2.2 int [nf__registration](#) (struct soap *, char *, char *, char *, struct [nf__response](#) &) [friend]

Process registration request from another service.

The documentation for this class was generated from the following file:

- flo.h

2.8 XMLNode Class Reference

```
#include <xmlwrap.h>
```

Public Member Functions

- [XMLNode](#) (void)
- [XMLNode](#) (const std::string &name)
- [XMLNode](#) (const [XMLNode](#) &node)
- [XMLNode](#) ([XMLNode](#) &parent, const std::string &name)
- [operator bool](#) (void) const
- [bool operator!](#) (void) const
- void [operator=](#) (const [XMLNode](#) &node)
- void [addNode](#) (const [XMLNode](#) &node)
- void [linkNode](#) ([XMLNode](#) &node)
- [XMLNode * getNode](#) (int n) const
- [XMLNode * getNode](#) (const std::string &name) const
- [XMLNode * getNode](#) (const std::string &name, int n) const
- int [nodes](#) (void) const
- std::string [name](#) (void) const
- void [name](#) (const std::string &s)
- std::string [content](#) (void) const
- void [content](#) (const std::string &s)
- std::string [get](#) (void) const
- void [set](#) (const std::string &s)

2.8.1 Detailed Description

Wrapper for sophisticated XML management API. Provides very simple, but also very simple to use functionality. There is no separate XML document. Everything is node. This implementation uses libxml2.

2.8.2 Constructor & Destructor Documentation

2.8.2.1 [XMLNode::XMLNode](#) (void) [inline]

Empty tree.

2.8.2.2 [XMLNode::XMLNode](#) (const std::string & *name*)

Top named node.

2.8.2.3 [XMLNode::XMLNode](#) (const [XMLNode](#) & *node*)

Copy constructor (copies whole subtree).

2.8.2.4 [XMLNode::XMLNode](#) ([XMLNode](#) & *parent*, const std::string & *name*)

Named node will be attached to parent.

2.8.3 Member Function Documentation

2.8.3.1 void XMLNode::addNode (const XMLNode & node)

Attach copy of supplied node as a child.

2.8.3.2 void XMLNode::content (const std::string & s)

Set content of this node.

2.8.3.3 std::string XMLNode::content (void) const

Get content of node (including all children).

2.8.3.4 std::string XMLNode::get (void) const

Get XML subtree in printable format.

2.8.3.5 XMLNode* XMLNode::getNode (const std::string & name, int n) const

Get pointer to child node by name and order.

2.8.3.6 XMLNode* XMLNode::getNode (const std::string & name) const

Get pointer to child node by name (first found returned).

2.8.3.7 XMLNode* XMLNode::getNode (int n) const

Get pointer to child node by order number.

2.8.3.8 void XMLNode::linkNode (XMLNode & node)

Attach supplied node as a child (supplied node must be top node).

2.8.3.9 void XMLNode::name (const std::string & s)

Set name of this node.

2.8.3.10 std::string XMLNode::name (void) const

Get name of this node.

2.8.3.11 int XMLNode::nodes (void) const

Get number of child nodes.

2.8.3.12 XMLNode::operator bool (void) const [inline]

If node is not empty.

2.8.3.13 bool XMLNode::operator! (void) const [inline]

If node is empty.

2.8.3.14 void XMLNode::operator= (const XMLNode & node)

Replace this node with a copy of supplied one. This node becomes a top one.

2.8.3.15 void XMLNode::set (const std::string & s)

Set new XML tree. Node becomes top node.

The documentation for this class was generated from the following file:

- xmlwrap.h

Index

- action_t
 - FIOObject, 7
- add
 - FIOService, 10
- addNode
 - XMLNode, 15
- content
 - XMLNode, 15
- deserialize
 - FIOMessage, 4
 - FIOObject, 7
- find
 - FIOService, 10
- FIOClient, 3
- FIOClient
 - notify, 3
 - registr, 3
- FIOMessage, 4
- FIOMessage
 - deserialize, 4
 - object, 4
 - operator=, 4
 - operator==, 4
 - serialize, 5
 - type, 5
- FIOObject, 6
- FIOObject, 7
- FIOObject
 - action_t, 7
 - deserialize, 7
 - FIOObject, 7
 - id, 7
 - maintain, 7
 - notification, 7, 8
 - operator bool, 8
 - operator!, 8
 - serialize, 8
 - service, 8
 - type, 8
 - type_t, 7
- FIORegistration, 9
- FIOService, 10
- FIOService
 - add, 10
 - find, 10
 - maintain, 10
 - notification, 10
 - notify, 10
 - registr, 11
 - service, 11
- FIOServiceRemote, 12
- FIOServiceRemote
 - maintain, 12
 - notify, 12
 - registr, 12
- get
 - XMLNode, 15
- getNode
 - XMLNode, 15
- HTTP_FIO, 13
- HTTP_FIO
 - nf__notification, 13
 - nf__registration, 13
- id
 - FIOObject, 7
- linkNode
 - XMLNode, 15
- maintain
 - FIOObject, 7
 - FIOService, 10
 - FIOServiceRemote, 12
- name
 - XMLNode, 15
- nf__notification
 - HTTP_FIO, 13
- nf__registration
 - HTTP_FIO, 13
- nodes
 - XMLNode, 15
- notification
 - FIOObject, 7, 8
 - FIOService, 10

notify
 FIOClient, 3
 FIOService, 10
 FIOServiceRemote, 12

object
 FIOMessage, 4

operator bool
 FIOObject, 8
 XMLNode, 15

operator!
 FIOObject, 8
 XMLNode, 16

operator=
 FIOMessage, 4
 XMLNode, 16

operator==
 FIOMessage, 4

registr
 FIOClient, 3
 FIOService, 11
 FIOServiceRemote, 12

serialize
 FIOMessage, 5
 FIOObject, 8

service
 FIOObject, 8
 FIOService, 11

set
 XMLNode, 16

type
 FIOMessage, 5
 FIOObject, 8

type_t
 FIOObject, 7

XMLNode, 14
 addNode, 15
 content, 15
 get, 15
 getNode, 15
 linkNode, 15
 name, 15
 nodes, 15
 operator bool, 15
 operator!, 16
 operator=, 16
 set, 16
 XMLNode, 14