

1 Target consumers of this standard

The target consumer of this standard is a desktop application. Not all desktop services nor a Internet or enterprise service but a desktop application.

Desktop applications like browsers, office suites, E-mail clients and instant messengers if they store personal configuration data sometimes also called the preferences of the user.

This standard is about personal preferences that influence the behaviour of desktop applications.

2 Technology dependencies of this standard

For a desktop developer, the only dependency for using this standard is the D-BUS standard and the service implementation that will implement it.

This standard doesn't add any other dependency. An implementation that respects this standard, can be used by any consumer of this standard.

3 Serve and consume using D-BUS

This standard defines a client versus service protocol. The service delivers the information and that performs requests coming from the client.

The client consumes the service. D-BUS is the transport layer technology between client and service.

The desktop application is the client. The implementation of this standard is the service.

4 Keys and values

This standard defines the key-names to be used. Some information that is configuration data can be shared between overlapping desktop applications. To avoid duplication of settings it's important to agree on a key-scheme.

Huge TODO (a lot work, and very hard to get people to agree)

/apps/office/font as string

/apps/networking/mail/accounts as list

/apps/networking/www/proxy as list

/apps/networking/application_name/specific as integer

5 The protocol

5.1 The types

The VALUE STRUCT

UINT32	the type	
STRING	the STRING data	or
INT32	the integer data	or
BOOLEAN	the boolean data	or
DOUBLE	the double data	or
ARRAY of STRUCT	the list data	or
ARRAY of STRUCT	the struct data	

One instance of the list data ARRAY of STRUCT

STRUCT	the value data
--------	----------------

One instance of the struct data ARRAY of STRUCT

STRING	the member-name
STRUCT	the value data

The METAINFO STRUCT

ARRAY OF STRING	comments about the key
ARRAY OF STRING	applications using the key
STRING	owner of the key
STRING	description of the key
UINT32	last time accessed
UINT32	last time modified
UINT32	time of creation

The SCHEMA STRUCT

The schema enforces both the client and the service to use a specific type for a specific key. A schema can enforce the list or struct type on the key but can't enforce the type of the value instances in such a list or a struct. This is done for simplification.

UINT32	type of the key
VALUE STRUCT	default value
MEAINFO STRUCT	default meta info
BOOLEAN	whether or not the key is writable

The KEYEVENT STRUCT

A keyevent struct is the information about an event that happened on one single key.

UINT32	type of the event
STRING	the affected key

The EVENT STRUCT

An event struct contains the information about an event that happened on one or more keys. It's an array of keyevent structs. An event happens when a group of value setting, metainfo setting and removals of keys happened. The amount of actions in a group can be one or more.

ARRAY of KEYEVENT STRUCT keyevents for the affected keys

5.2 Getting values

Gets the value of a key. In case no value is available, the default from the schema of the key is returned.

In case there is also no schema an error is returned. (TODO: Which error)

org.freedesktop.configuration.GetValues

org.freedesktop.configuration.GetValue

As methods:

ARRAY of VALUE STRUCT	GetValues	(STRING root)
VALUE STRUCT	GetValue	(STRING key)

5.3 Setting values

This action sets (overwrites or creates) the value of a key. The notify flag is to be set to false in case this is a member of a group of actions.

In case the schema of the key says the key is read-only, an error is returned. (TODO: Which error)

SetValue (STRING key, VALUE STRUCT value, BOOLEAN notify)

5.4 Getting metainfo

Gets the metainfo of a key. In case no metainfo is available, the default from the schema of the key is returned.

In case there is also no schema an error is returned. (TODO: Which error)

org.freedesktop.configuration.GetMetaInfo

As a method:

METAINFO STRUCT	GetMetaInfo	(STRING key)
-----------------	-------------	--------------

5.5 Setting metainfo

This action sets (overwrites or creates) the metainfo of a key. The notify flag is to be set to false in case this is a member of a group of actions.

In case the schema of the key says the key is read-only, an error is returned. (TODO: Which error)

org.freedesktop.configuration.SetMetaInfo

As a method:

SetMetaInfo (STRING key, METAINFO STRUCT metainfo, BOOLEAN notify)

5.6 Getting schemas

Gets the schema of a key.

In case there is also no schema and no value an error is returned.

In case there's a value but no schema a schema is generated from the value. (TODO: yes/no?)

org.freedesktop.configuration.GetSchema

As a method:

```
SCHEMA STRUCT          GetSchema          (STRING key)
```

5.7 Subscribing to notification of changes

A client will only receive a KeysChanged event about a specific key if it's subscribed to events about that key. These actions subscribe and unsubscribe a client.

If subscribing on a non-existing, an error is returned. (TODO: Which error)

If unsubscribing from a key on which the client isn't subscribed, nothing happens.

If unsubscribing from a non-existing key, nothing happens.

org.freedesktop.configuration.SubscribeOnKey

org.freedesktop.configuration.UnSubscribeFromKey

As methods:

```
SubscribeOnKey          (STRING key)  
UnSubscribeFromKey      (STRING key)
```

5.8 Receiving notification of changes

This signal happens (on the client) if the client is subscribed to a key that was changed. (TODO: Is this already possible with D-BUS? Only clients that are interested should be signalled).

org.freedesktop.configuration.KeysChanged

This is a signal:

```
KeysChanged            (EVENT STRUCT event)
```

5.9 Notifying the other clients about a group of changes

This action is only to be used when the developer has done SetMetaInfo, SetValue or RemoveKeys actions with the notify flag set to false.

In which case at the end of this group of actions the developer should let the other clients know what happened.

If the developer decides not to do this, the service implementation does it automatically after one minute.

The EVENT STRUCT instance contains an array of KEYEVENT STRUCT instances which must be created by the developer. The service will forward the EVENT STRUCT instance to clients that are subscribed on one of the keys in it.

org.freedesktop.configuration.NotifyAboutChanges

As a method:

```
NotifyAboutChanges      (EVENT STRUCT event)
```

5.10 Removing

This action removes all keys starting from the root. The notify flag is to be set to false in case this is a member of a group of actions.

org.freedesktop.configuration.RemoveKeys

As a method:

```
RemoveKeys          (STRING root, BOOLEAN notify)
```