

# c<sup>1</sup>/4h - D-Bus intro

Merovius

September 19, 2013

# Was ist D-Bus?

Die folgenden Folien bauen aufeinander auf. Man kann jede Schicht ohne die darüber liegenden nutzen.

- Konvertiere - in portabler Weise - Daten in wohldefinierte, umkehrbarer Weise in Binärdaten

# Serialisierungs Format

- Konvertiere - in portabler Weise - Daten in wohldefinierte, umkehrbarer Weise in Binärdaten
- Basic-types verschiedene ints, bool, double, string,

# Serialisierungs Format

- Konvertiere - in portabler Weise - Daten in wohldefinierte, umkehrbarer Weise in Binärdaten
- Basic-types verschiedene ints, bool, double, string, object-path,

# Serialisierungs Format

- Konvertiere - in portabler Weise - Daten in wohldefinierte, umkehrbarer Weise in Binärdaten
- Basic-types verschiedene ints, bool, double, string, object-path, signature

# Serialisierungs Format

- Konvertiere - in portabler Weise - Daten in wohldefinierte, umkehrbarer Weise in Binärdaten
- Basic-types verschiedene ints, bool, double, string, object-path, signature
- Container types:

- Konvertiere - in portabler Weise - Daten in wohldefinierte, umkehrbarer Weise in Binärdaten
- Basic-types verschiedene ints, bool, double, string, object-path, signature
- Container types:
  - array



- Konvertiere - in portabler Weise - Daten in wohldefinierte, umkehrbarer Weise in Binärdaten
- Basic-types verschiedene ints, bool, double, string, object-path, signature
- Container types:
  - array
  - struct

- Konvertiere - in portabler Weise - Daten in wohldefinierte, umkehrbarer Weise in Binärdaten
- Basic-types verschiedene ints, bool, double, string, object-path, signature
- Container types:
  - array
  - struct
  - dict

- Konvertiere - in portabler Weise - Daten in wohldefinierte, umkehrbarer Weise in Binärdaten
- Basic-types verschiedene ints, bool, double, string, object-path, signature
- Container types:
  - array
  - struct
  - dict
- Polymorpher type: variant

- Konvertiere - in portabler Weise - Daten in wohldefinierte, umkehrbarer Weise in Binärdaten
- Basic-types verschiedene ints, bool, double, string, object-path, signature
- Container types:
  - array
  - struct
  - dict
- Polymorpher type: variant
- Awesomer type: unix\_fd

- 1-on-1 Kommunikation

- 1-on-1 Kommunikation
- Asynchrones Design

- 1-on-1 Kommunikation
- Asynchrones Design
- Authentifikation built-in

- 1-on-1 Kommunikation
- Asynchrones Design
- Authentifikation built-in
- Adressiert über Namen die wohlbekannt sind, oder on-the-fly erstellt werden



- 1-on-1 Kommunikation
- Asynchrones Design
- Authentifikation built-in
- Adressiert über Namen die wohlbekannt sind, oder on-the-fly erstellt werden
- Basierend auf „messages“

- 1-on-1 Kommunikation
- Asynchrones Design
- Authentifikation built-in
- Adressiert über Namen die wohlbekannt sind, oder on-the-fly erstellt werden
- Basierend auf „messages“
- Message-types:

- 1-on-1 Kommunikation
- Asynchrones Design
- Authentifikation built-in
- Addressiert über Namen die wohlbekannt sind, oder on-the-fly erstellt werden
- Basierend auf „messages“
- Message-types:
  - ① Method call - „Führ mal aus!“

- 1-on-1 Kommunikation
- Asynchrones Design
- Authentifikation built-in
- Adressiert über Namen die wohlbekannt sind, oder on-the-fly erstellt werden
- Basierend auf „messages“
- Message-types:
  - 1 Method call - „Führ mal aus!“
  - 2 Method return - „Fertig, hier ist das Ergebnis!“

- 1-on-1 Kommunikation
- Asynchrones Design
- Authentifikation built-in
- Adressiert über Namen die wohlbekannt sind, oder on-the-fly erstellt werden
- Basierend auf „messages“
- Message-types:
  - 1 Method call - „Führ mal aus!“
  - 2 Method return - „Fertig, hier ist das Ergebnis!“
  - 3 Error - „Oops, da ist was schief gelaufen“

- 1-on-1 Kommunikation
- Asynchrones Design
- Authentifikation built-in
- Adressiert über Namen die wohlbekannt sind, oder on-the-fly erstellt werden
- Basierend auf „messages“
- Message-types:
  - 1 Method call - „Führ mal aus!“
  - 2 Method return - „Fertig, hier ist das Ergebnis!“
  - 3 Error - „Oops, da ist was schief gelaufen“
  - 4 Signal - „Hey, hier ist was passiert!“

- 1-on-1 Kommunikation
- Asynchrones Design
- Authentifikation built-in
- Adressiert über Namen die wohlbekannt sind, oder on-the-fly erstellt werden
- Basierend auf „messages“
- Message-types:
  - 1 Method call - „Führ mal aus!“
  - 2 Method return - „Fertig, hier ist das Ergebnis!“
  - 3 Error - „Oops, da ist was schief gelaufen“
  - 4 Signal - „Hey, hier ist was passiert!“
- Interfaces

- 1-on-1 Kommunikation
- Asynchrones Design
- Authentifikation built-in
- Adressiert über Namen die wohlbekannt sind, oder on-the-fly erstellt werden
- Basierend auf „messages“
- Message-types:
  - 1 Method call - „Führ mal aus!“
  - 2 Method return - „Fertig, hier ist das Ergebnis!“
  - 3 Error - „Oops, da ist was schief gelaufen“
  - 4 Signal - „Hey, hier ist was passiert!“
- Interfaces
- Spezielle Interfaces für Properties, Introspection. . .



- Besonderer Bus-Prozess, implementiert Bus-interface

- Besonderer Bus-Prozess, implementiert Bus-interface
- 1-on-1 Verbindung zum Bus-Prozess

- Besonderer Bus-Prozess, implementiert Bus-interface
- 1-on-1 Verbindung zum Bus-Prozess
- Bus-Prozess routet messages zu anderen Prozessen

- Besonderer Bus-Prozess, implementiert Bus-interface
- 1-on-1 Verbindung zum Bus-Prozess
- Bus-Prozess routet messages zu anderen Prozessen
- Basierend auf registrierten Namen

- Besonderer Bus-Prozess, implementiert Bus-interface
- 1-on-1 Verbindung zum Bus-Prozess
- Bus-Prozess routet messages zu anderen Prozessen
- Basierend auf registrierten Namen
- Default: Session- und SystemBus (über env bzw. spezifizierte Pfade erreichbar)

- Besonderer Bus-Prozess, implementiert Bus-interface
- 1-on-1 Verbindung zum Bus-Prozess
- Bus-Prozess routet messages zu anderen Prozessen
- Basierend auf registrierten Namen
- Default: Session- und SystemBus (über env bzw. spezifizierte Pfade erreichbar)
- Eigene Busprozesse könnten erstellt werden

- Kapselt Serialisierung, RPC und Kommunikation mit dem Bus

- Kapselt Serialisierung, RPC und Kommunikation mit dem Bus
- $\Rightarrow$  Kann sowohl für 1-on-1 IPC benutzt werden, als auch zur Kommunikation mit dem Bus



# Demo