



Solid

Intégration avec le matériel sans utiliser d'aspirine

Kévin Ottens

26 Janvier 2008





- 1 Motivations
- 2 Architecture
- 3 Utiliser l'API



1 Motivations

2 Architecture

3 Utiliser l'API



Ce qui est disponible pour KDE3

- Détection du matériel: mediamanager, medianotifier...
- Gestion réseau: knetworkmanager, kwifimanager...
- Gestion de l'énergie: kpowersave, sebas' powermanager...

Pourquoi est-ce nocif?

- Couvre uniquement un sous ensemble des fonctionnalités
- Informations difficilement accessibles aux autres applications
- Lié à un système particulier



Permettre les améliorations



Nouvelles utilisations

- "Périphériques nomades"
- Bluetooth déjà présent
- Collaboration Matériel/Logiciel

Nouvelles plateformes

- Windows
- Mac OS X
- *BSD (le support de KDE3 n'était pas parfait)



Plusieurs domaines

- Détection du matériel, gestion du réseau et de l'énergie
- Chaque domaine correspond à un ensemble de classes
- Chaque domaine aura au moins un "policy agent"

Policy agents, kezaiko?

- Ils sont déjà parmi nous: knetworkmanager, mediamanager...
- Responsables de:
 - l'interaction avec l'utilisateur
 - d'appliquer ses réglages



Fiche d'identité

- Leader : hum... oui ok... j'avoue
- But : Améliorer l'interaction entre le matériel et les applications du bureau

Détails techniques

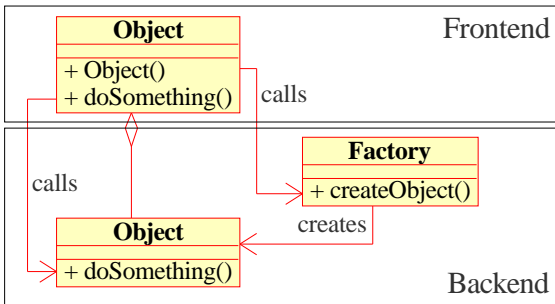
- Architecture à backends
- "Fake backend" pour l'écriture de tests unitaires
- Plusieurs domaines
 - Découverte du matériel
 - Gestion du réseau
 - Gestion de l'énergie
- API de haut niveau : faciliter la vie des développeurs



- 1 Motivations
- 2 Architecture
- 3 Utiliser l'API

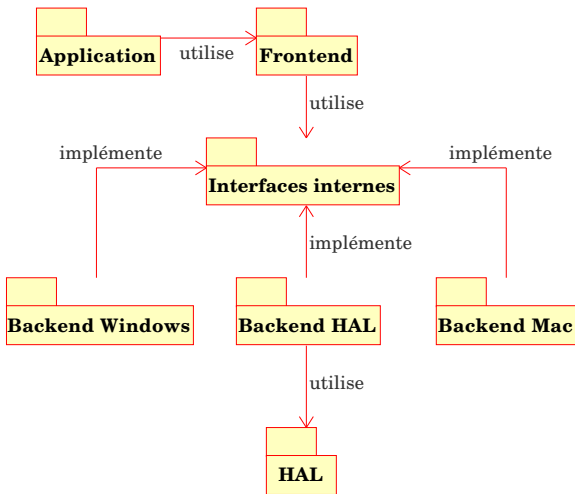


Séparation Frontend/Backend



Principe

- Compabilité binaire assurée dans le frontend
- Chaque objet du frontend pointe sur un objet du backend
- Une "factory" est utilisée pour créer les objets du backend





Satisfaire tous les besoins



libsolid (kdelibs)

- Couvre les besoins de 90% des applications
- Détection du matériel
- Tâches simples pour la gestion du réseau et de l'énergie

libsolidcontrol (kdebase)

- Couvre les besoins des 10% d'applications restantes
- API plus complexe et spécialisée
- Cible les applications faisant partie de la plateforme
- Couvre la gestion du réseau et de l'énergie
- Couvre le bluetooth

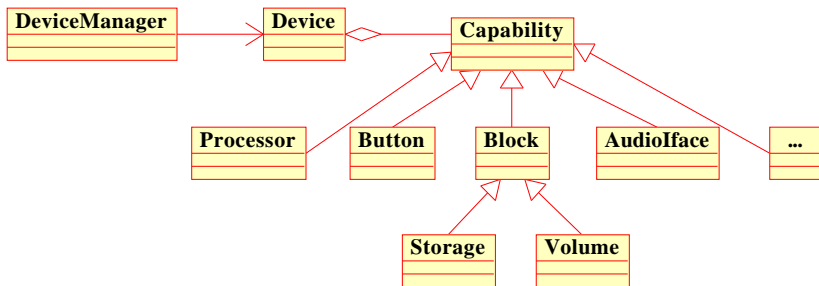


- 1 Motivations
- 2 Architecture
- 3 Utiliser l'API



Découverte du matériel (1/3)

Qu'est-ce qu'il y a de bon là dedans?





Découverte du matériel (2/3)

Qu'est-ce qu'il y a de bon là dedans?



Principes

- Le système a des Devices, chacun ayant un identifiant unique
- Les Devices sont organisés dans une hiérarchie
- Chaque Device a des interfaces de différents type
- L'ensemble des DeviceInterfaces d'un Device décrit ce dont il est capable

Notifications

- `Solid::DeviceNotifier::deviceAdded(QString)`
- `Solid::DeviceNotifier::deviceRemoved(QString)`



Découverte du matériel (2/3)

Qu'est-ce qu'il y a de bon là dedans?



Trouver des devices

- `Solid::Device::allDevices()`
- `Solid::Device::devicesFromType()`
- `Solid::Device::devicesFromQuery()`

```
QList<Solid::Device> all  
    = Solid::Device::allDevices();
```



Découverte du matériel (2/3)

Qu'est-ce qu'il y a de bon là dedans?



Trouver des devices

- `Solid::Device::allDevices()`
- `Solid::Device::devicesFromType()`
- `Solid::Device::devicesFromQuery()`

```
QList<Solid::Device> processors
= Solid::Device::listFromType(
    Solid::DeviceInterface::Processor);
```




Découverte du matériel (2/3)

Qu'est-ce qu'il y a de bon là dedans?



Trouver des devices

- `Solid::Device::allDevices()`
- `Solid::Device::devicesFromType()`
- `Solid::Device::devicesFromQuery()`

```
QList<Solid::Device> usbDrives
    = Solid::Device::listFromQuery(
        "StorageDrive.bus == 'Usb'");
```



Découverte du matériel (2/3)

Qu'est-ce qu'il y a de bon là dedans?



Trouver des devices

- `Solid::Device::allDevices()`
- `Solid::Device::devicesFromType()`
- `Solid::Device::devicesFromQuery()`

Manipuler des devices

- `Solid::Device::is<T>()`
- `Solid::Device::as<T>()`

```
Solid::Device dev = ...  
if (dev.is<Solid::Processor>()) {  
    ...  
}
```



Découverte du matériel (2/3)

Qu'est-ce qu'il y a de bon là dedans?



Trouver des devices

- `Solid::Device::allDevices()`
- `Solid::Device::devicesFromType()`
- `Solid::Device::devicesFromQuery()`

Manipuler des devices

- `Solid::Device::is<T>()`
- `Solid::Device::as<T>()`

```
Solid::Device dev = ...
if (dev.is<Solid::Camera>()) {
    QVariant handle =
        dev.as<Solid::Camera>()->driverHandle();
    ...
}
```



Découverte du matériel (2/3)

Qu'est-ce qu'il y a de bon là dedans?



Trouver des devices

- `Solid::Device::allDevices()`
- `Solid::Device::devicesFromType()`
- `Solid::Device::devicesFromQuery()`

Manipuler des devices

- `Solid::Device::is<T>()`
- `Solid::Device::as<T>()`

Demo, "storage-plug" (94 lignes)



Solid::Networking

- `status()` / `statusChanged()`
- `shouldConnect()` / `shouldDisconnect()`

Solid::Powermanagement

- `appShouldConserveResources()`
- `requestSleep()`
- `begin/stopSuppressSleep()`



Solid::Networking

- `status()` / `statusChanged()`
- `shouldConnect()` / `shouldDisconnect()`

Solid::Powermanagement

- `appShouldConserveResources()`
- `requestSleep()`
- `begin/stopSuppressSleep()`



Solid::Networking

- `status()` / `statusChanged()`
- `shouldConnect()` / `shouldDisconnect()`

Solid::Powermanagement

- `appShouldConserveResources()`
- `requestSleep()`
- `begin/stopSuppressSleep()`



Solid::Networking

- `status()` / `statusChanged()`
- `shouldConnect()` / `shouldDisconnect()`

Solid::Powermanagement

- `appShouldConserveResources()`
- `requestSleep()`
- `begin/stopSuppressSleep()`



Solid::Networking

- `status()` / `statusChanged()`
- `shouldConnect()` / `shouldDisconnect()`

Solid::Powermanagement

- `appShouldConserveResources()`
- `requestSleep()`
- `begin/stopSuppressSleep()`



Questions ?

Kévin Ottens
ervin@kde.org