Upgrade management

Upgrade management

- Introduction
- Addon upgrades
- Soffid upgrades

Introduction

Soffid is concerned about component versioning and how to upgrade them.

A upgrade system is designed in order to easiest Soffid core components upgrade, as well as Soffid addons addition, suppression or upgrades. Soffid also defines the interface to use service beans. This service beans are use by both addon services and user interface.

> See Soffid upgrades and Addon upgrades

Addon upgrades

Soffid addons should be able to manage data schema changes. Soffid gives addons a standard procedure to perform data and schema upgrades. The data upgrade process is performed in three steps:

1. The addons to upgrade are deployed into the soffid console using the Plugins screen.

2. On console boot, the data schema is updated. To perform the schema update, a Soffid developed schema updater is used. This schema updater will read the core-ddl.xml and any plugin-ddl.xml file to upgrade at once the core Soffid database objects as well as any addon database object. (See Data schema descriptor)

3 The application bootstrap process is executed. At this step, any spring bean implementing ApplicationBootService will be invoked. This been is responsible for completing data upgrade. Addons can also define a bean implementing ApplicationBootService for this purpose.

Soffid upgrades

Soffid upgrades are managed using the same mechanisms as addon upgrades.

Data upgrade

The data upgrade process is performed in three steps:

1. The console is upgraded using soffid installer.

2. On console boot, the data schema is updated. To perform the schema update, a Soffid developed schema updater is used. This schema updater will read the core-ddl.xml and any plugin-ddl.xml file to upgrade at once the core Soffid database objects as well as any addon database object.

3. The application bootstrap process is executed. At this step, any spring bean implementing ApplicationBootService will be invoked. This been is responsible for completing data upgrade. Addons can also define a bean implementing ApplicationBootService for this purpose.

Service beans

The new Soffid version could have new service beans. The addons should use a prefix in order to avoid name collisions. So, as long as exists a "accountService" bean, there could by any "addon-accountService" bean. This prefix should be used for any service bean or DAO bean.

Hibernate objects

It's possible to have name collisions on hibernate entity beans. In order to avoid it, the hibernate mapping files should always set the auto-import clause to "false".

User interface

Addons cannot replace any Soffid user interface component. In order to customize or modify user interface, addon contains XSL transformation files that are applied onto existing user interface components. This mechanisms guarantees that many addons can introduce changes on the same user interface component, and it also assures that user interface upgrades will proceed smoothly.