J-Spring

16 april 2008 Spant! - Bussum

Did We Spoil the End User? Building Personalization into JavaServer Faces.

Lucas Jellema

AMIS

http://technology.amis.nl/blog

Overview

- Demonstration of “FootieFans” application
- Discussion of Trends in Application Development and Deployment
- Run-time Customization of JSF Applications
  - Dynamic Menu, Configurable Text, Application Settings, …
- The SaaS (single instance) deployment
- Personalization of the application

With (in order of appearance)

JSF PhaseListener

AJAX

Faces-Config.xml

JSF Converter

Partial Page Rendering

EL Expressions

SQL

Personalization

SaaS!

Implementation MyHistory

- PhaseListener on Before Render Response
  - If the page to render is one to add to the history, then call …
- HistoryController bean (session scope) that manages the collection of navigation events
  - Display Label, Entity Type and Primary Key
  - bean calls …
- BusinessService to record event in database for this user
    - Also remove events for deleted records!


.nl
jug
Google
salesforce.com
experience success.
amazon.com
Linked in
facebook
Hyves.nl
Microsoft
ORACLE
AMIS
Windows Installer

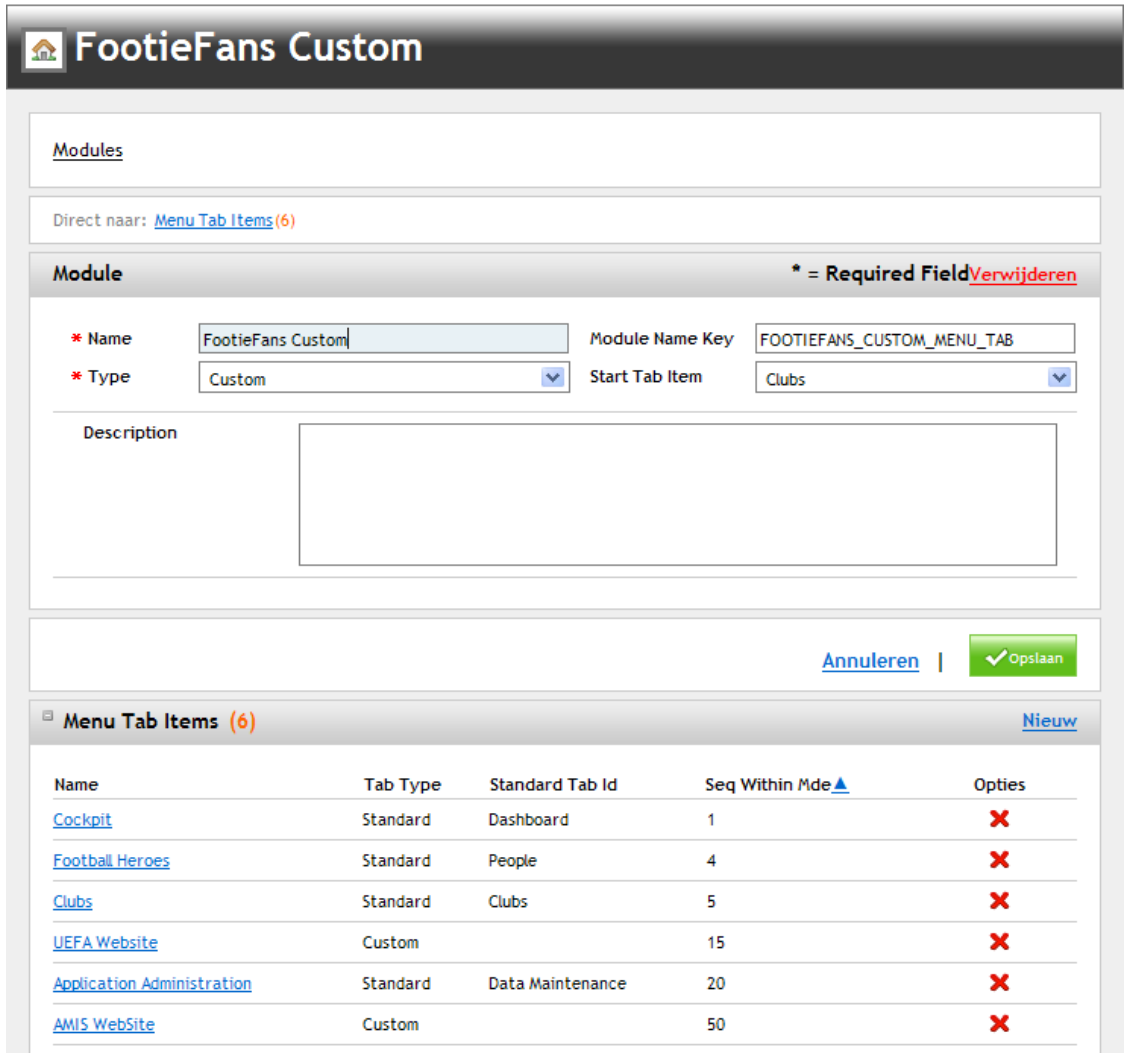
Trends in Business Applications

- Web based deployment, SaaS model (fka ASP):
  - single instance with multiple customers
- Web 2.0, AJAX, RIA, 'internet savvy' users
  - Social Networks
- Personalization features: My….
  - My favorites, my history, my preferences
- Standard applications – reduced custom development
  - However: customization (configuration)

Modules and Menus

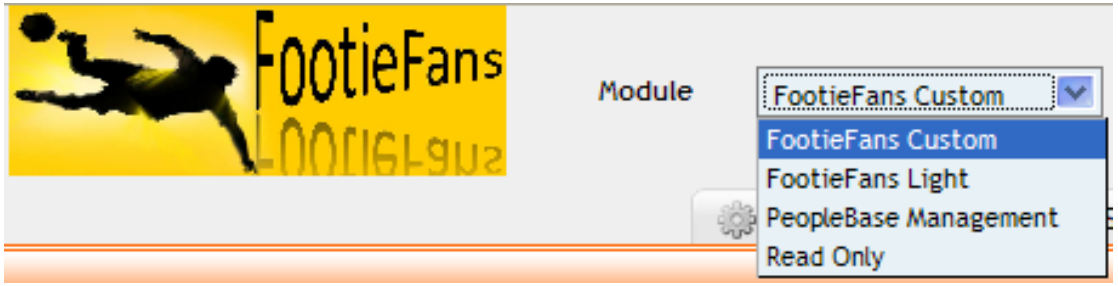
- Module & Menu management for FootieFans
- Application Manager can create “Modules”:
  - Custom combinations of Menu Tab Items
  - Combining Standard Application Tabs and Custom Tab Items (associated with URLs)
  - Modules can be granted to users

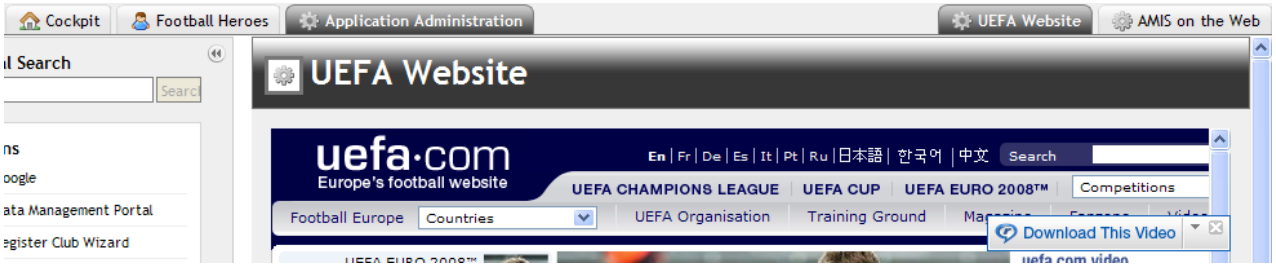
Module & Menu Management



Demonstration of Customization

- At run-time, the user picks one of the available Modules to work in a tailored setting

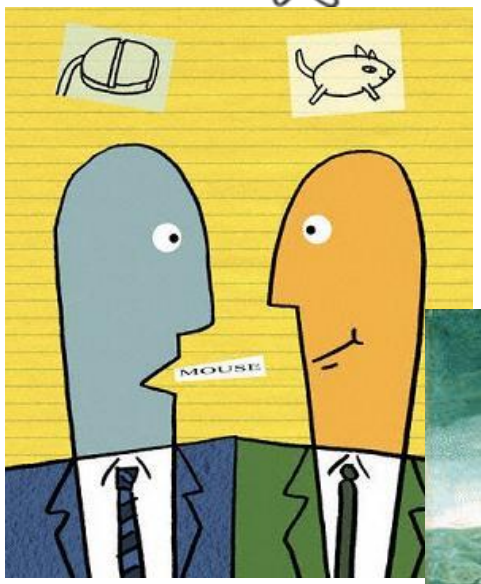




Dynamic Menu Implementation

- Database Tables for Modules & Menu Items
- Business Service reads *applicable* set
- Global Menu component (reusable) leverages ModuleManager bean
  - Bean retrieves Module list from service
  - Bean exposes Items for selected Module
  - Handles change of Module
    - refresh menu items and re-render the page
- JavaScript (document.write) renders HTML

Tower of Babel



On the fly Customizing Text

- Internationalization of all boilerplate text items is done via ResourceBundles
  - Locale specific property sets: en, fr_ca, nl_be
- The application contains hundreds of references like: label="#{nls['someKey']}"
  - That at runtime are resolved based on the current locale and the available bundles
- Typically, resource bundles are implemented in property files …. but they do not have to be!

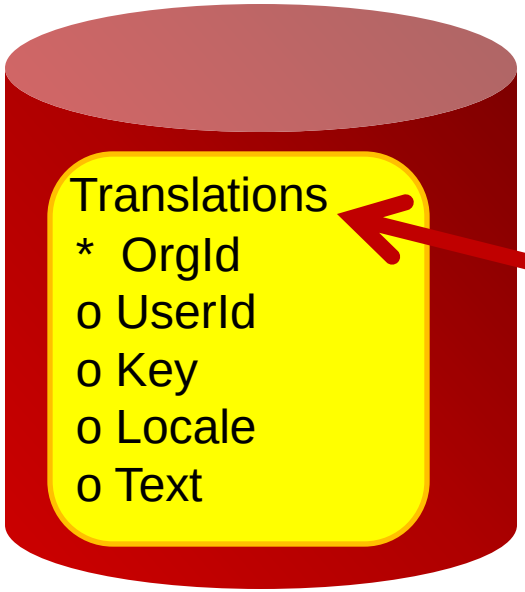
Databased ResourceBundles

- Allows for run-time management of Text
  - via the application's User Interface
  - text entries specific for an organization
- Custom class that extends ListResourceBundle
  - Overriding method Object [][] getContents()
  - Reading the contents from the database
- *Force refresh* "trick" to load changes from database into the in memory resource bundle

RDBMS backed ResourceBundles

Class DBResourceBundle
extends ListResourceBundle

```
Object[][] getContents()
Locale getLocale()
```



Business Service
(persistency layer)

```
Map getTranslationsForLocale()
```

Application ‘levels of reality’

Factory

Customer

FootieFans Inc.

End User

Mark

Linda

Customer

IT HistoryBase

End User

Jessica

Application Rollout process…

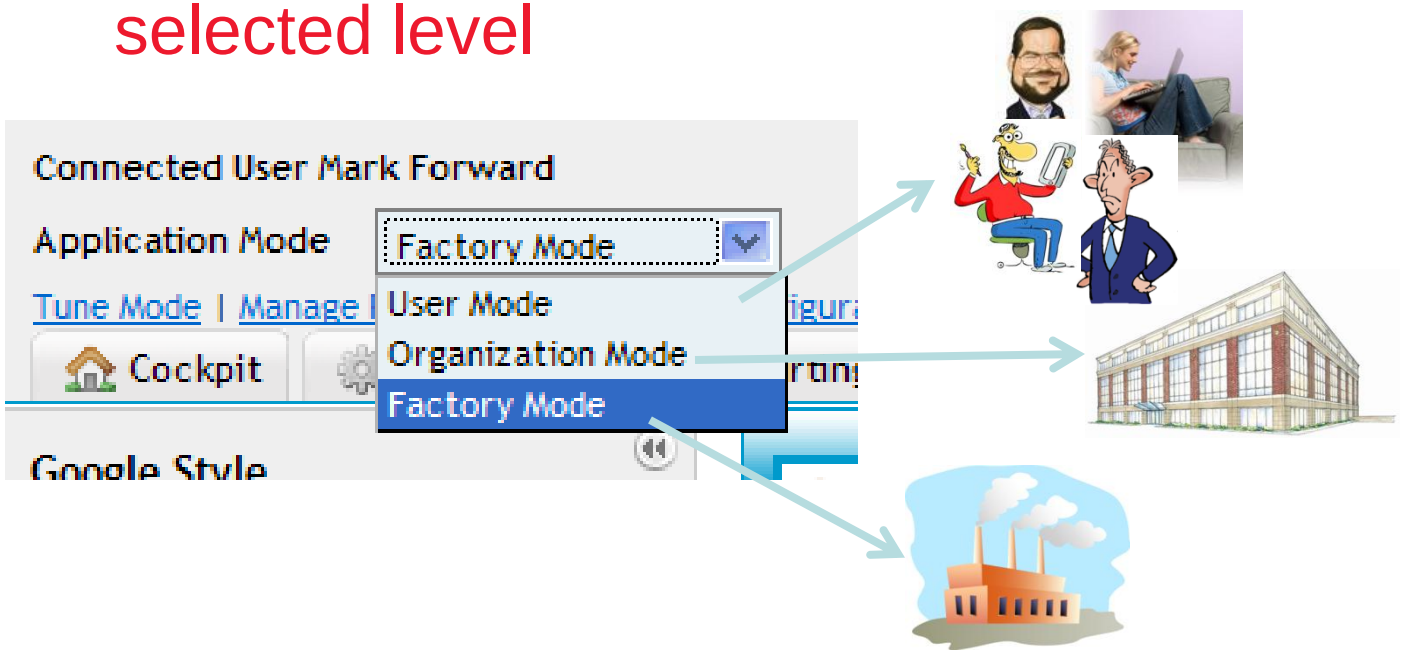
- Development in the Factory
- Production-time refining

- Customer (organization) specific implementation
  - By SaaS Helpdesk, on site consultant, Application Admin

- End User (dynamic, run-time) Personalization

Application Mode Switcher

- Application Managers can switch between Application Modes
  - To view and manipulate the application at the selected level



Application Modes explained

- Application Modes can be used to review the various levels of customization

- Factory level = as shipped (the original standard application - PeopleBase)
- Organization level = as customized during implementation and/or by application administrator – FootieFans)
  - Also contains Factory Level where not overridden

- User level = all personal settings and options applied by an individual user
  - Also Factory and Organization level

Application Settings

- Some Application Behavior is developed as *dynamically configurable*
  - Next to Data itself, NLS and Modules/Menus
  - Preferences (Application Settings)
  - Such as Locale, Use Pictures for People, Units for Height and Weight, Default Value, Validation parameters, Date Format etc.
- Settings can be configured at Factory (default), Organization and User level

Application Settings Data Model

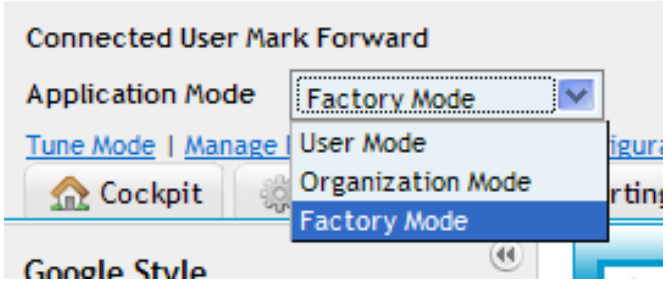
Application Settings
* Name
o Factory Value
o Allowable Values
o Description
o Data Type

Customized Settings
* OrgId
o Value
o UserId
o Comment

Date Format	dd-mm-yyyy
Use Photo	Y
Metric/ Imperial	m
…	

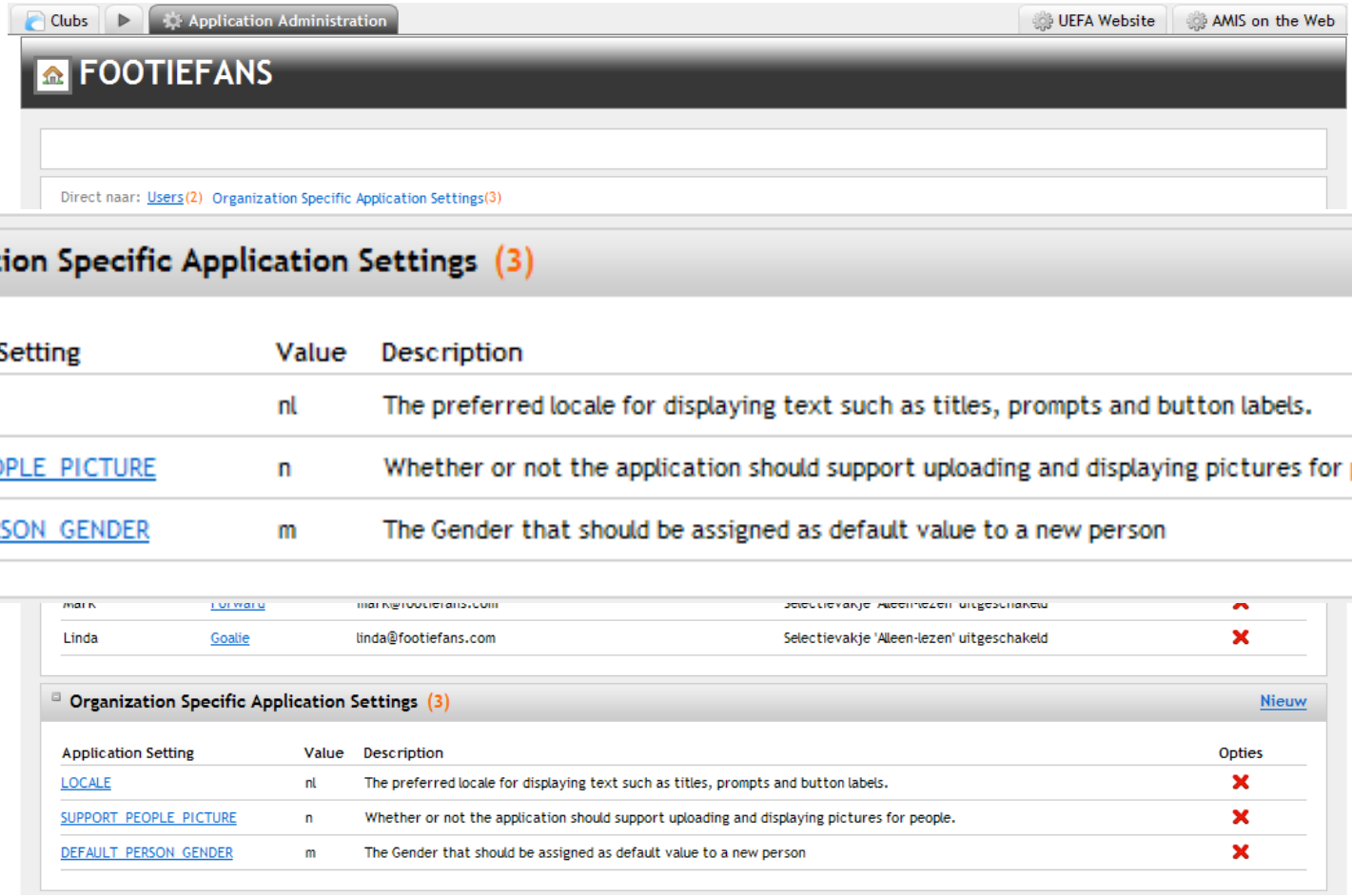
	mm/dd/yyyy
	dd-mm-yy
	N
	Y
	N
	i

Determining application settings



- Query all Application Settings & Factory Values
  - Store them in the map user.settings
- If Application Mode is *user* or *organisation*
  - Query Org-level Customizations and apply to the same map (factory values are overridden)
- If Application Mode is *user*
  - Query User-level Customizations and apply to same map (overriding factory & org values)

Configuring Application Settings

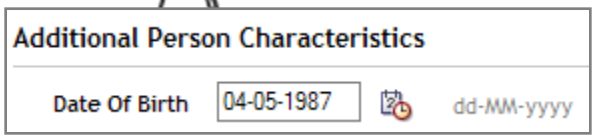


Integrating Application Settings during development

#{user.settings['NAME']}

Date Pattern Application Setting

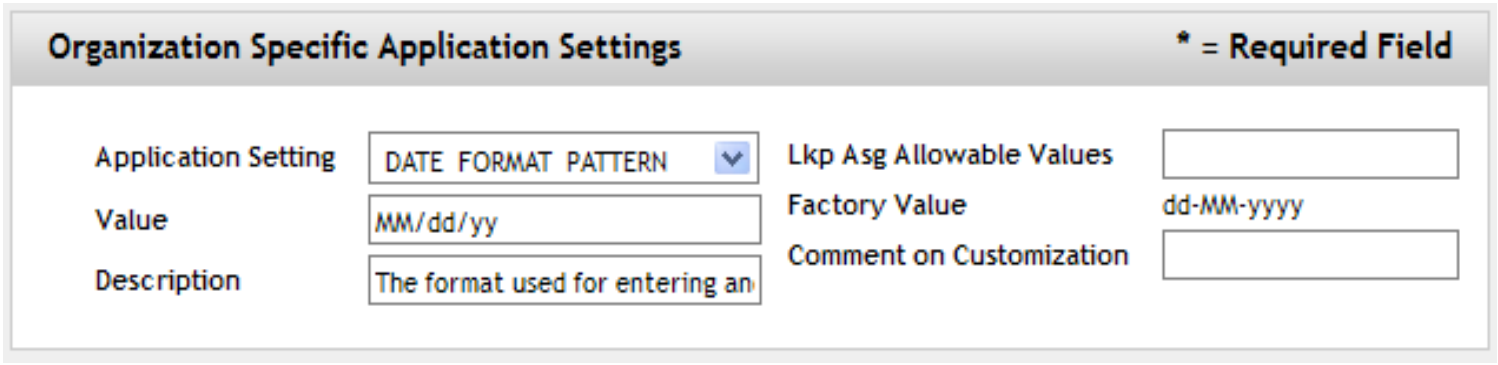
1



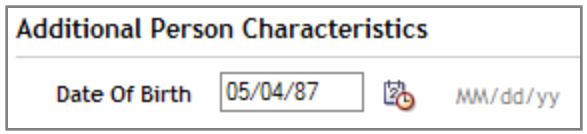
2

```
<af:panelGroup layout="horizontal">
  <af:selectInputDate id="PeopleDateOfBirth" value="#{bindings.PeopleDateOfBir
    <af:convertDateTime pattern="#{jhsUser.settings['DATE_FORMAT_PATTERN']}"/>
  </af:selectInputDate>
  <af:outputText value="#{jhsUser.settings['DATE_FORMAT_PATTERN']}" styleClass=
</af:panelGroup>
```

3

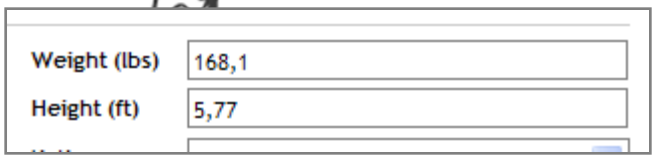


4



Height & Weight Converter

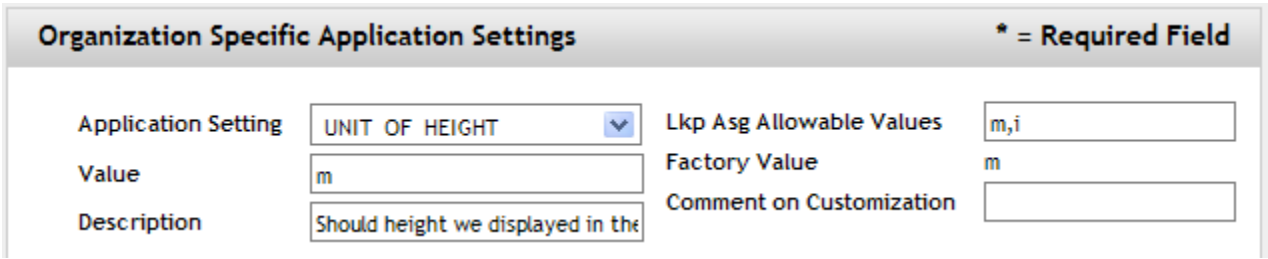
1



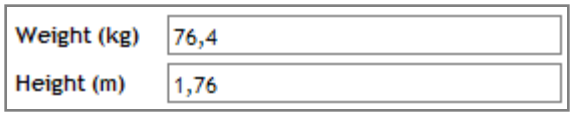
2

```
<af:inputText id="PeopleWeight" value="#{bindings.PeopleWeight.inputValue}"
        label="#{nls['PEOPLE_WEIGHT']} #{jhsUser.settings['UNIT_OF_WEIGHT']=='m'?' (kg)':' (lbs)'}">
    <f:converter converterId="WeightConverter"/>
</af:inputText>
<af:inputText id="PeopleHeight" value="#{bindings.PeopleHeight.inputValue}"
        label="#{nls['PEOPLE_HEIGHT']}  #{jhsUser.settings['UNIT_OF_HEIGHT']=='m'?' (m)':' (ft)'}" >
    <f:converter converterId="HeightConverter"/>
```

3



4



Weight Converter – lbs to kg

- Implement Converter interface
  - getAsObject() – from request (UI) to model
  - getAsString() – from model to User Interface
  - *Read Application Setting from user object to determine conversion factor*
- Register in faces-config.xml

```
<converter>
  <converter-id>WeightConverter</converter-id>
  <converter-class>nl.amis.jsf.WeightConverter</converter-class>
</converter>
```

- Associate Converter with InputText

```
<af:inputText id="PeopleWeight" value="#{binding
        label="#{nls['PEOPLE_WEIGHT']} #{jhsUser.
    <f:converter converterId="WeightConverter"/>
```

Convert using Application Setting

- Get Application Setting

```
private String getUnitOfWeight() {
    FacesContext ctx = FacesContext.getCurrentInstance();
    ValueBinding vb = ctx.getApplication().createValueBinding("#{jhsUser}");
    AppUser appUser = (AppUser)vb.getValue(ctx);
    return (String)appUser.getSettings().get("UNIT_OF_WEIGHT");
}
```

- Implement getAsString()

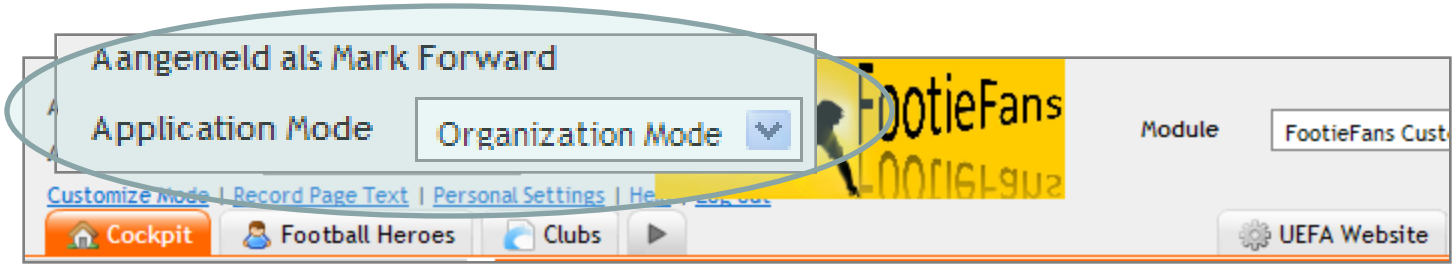
```
public String getAsString(FacesContext facesContext,
                          UIComponent uiComponent, Object object) {
    try {
        if (null == object || ((Number)object).compareTo(0)==0)
          return null;
        float floatValue = ((Number)object).floatValue();
         if ("i".equalsIgnoreCase(getUnitOfWeight())) {
           floatValue *=METRIC_TO_IMPERIAL_WEIGHT_CONVERSION_FACTOR;
         }
          return this.getNumberFormat().format(floatValue);
    } catch (Exception e) { throw new ConverterException(e);}
}
```

Application Personalization

Personalization Features

- My Navigation History
- My Favorites (quick links, bookmarks)
- My own custom properties (‘flex fields’)
- My flavor of Date Format, Units for Weight, Height, Currency, Default Values
- My lingo in titles, labels, prompts & messages
- Personalized Menu-structure
- … (my logo!)

The “Personal Touch”



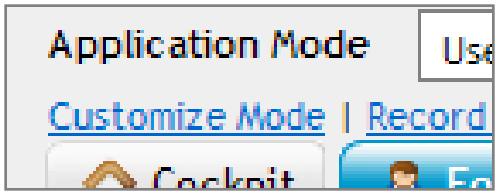
Chances are that a standard application is not perfect!

- Some fields are never used by some organizations (or even legally forbidden!)
- Other fields are sorely missed
- To allow optimal customization, an application ideally allows
  - Customization of standard items
  - Addition of custom items
    - Associated with data records
    - Integrated in the application's pages, indiscernible from 'regular' items

Adding the “Caps” flex item

1

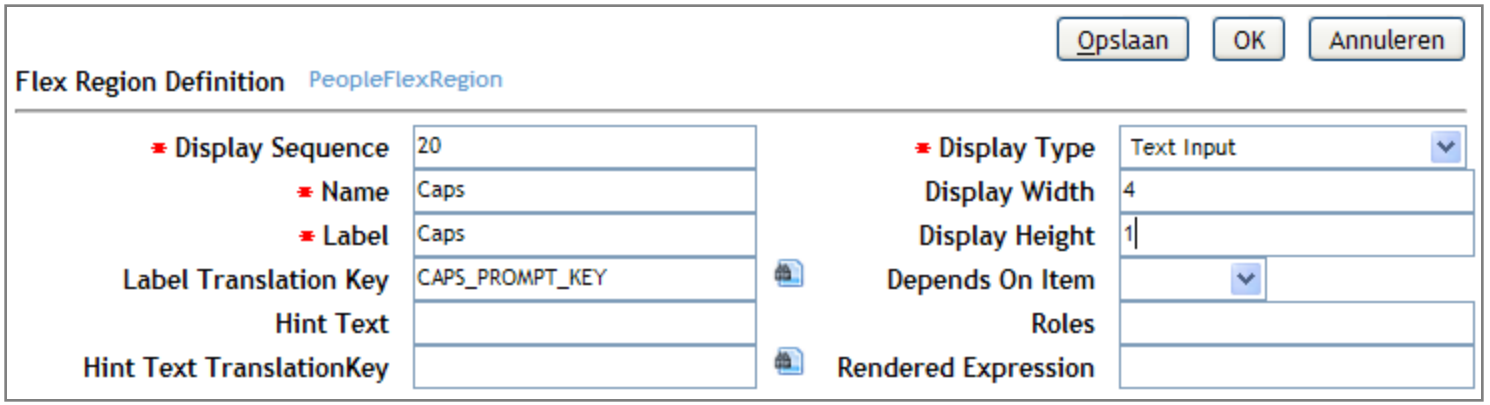
2



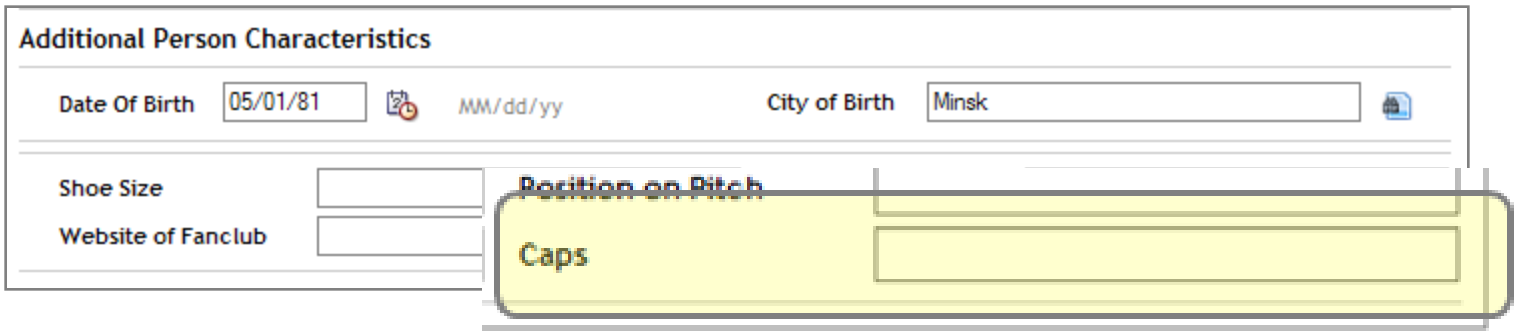
3

Adding “Caps” item (2)

4

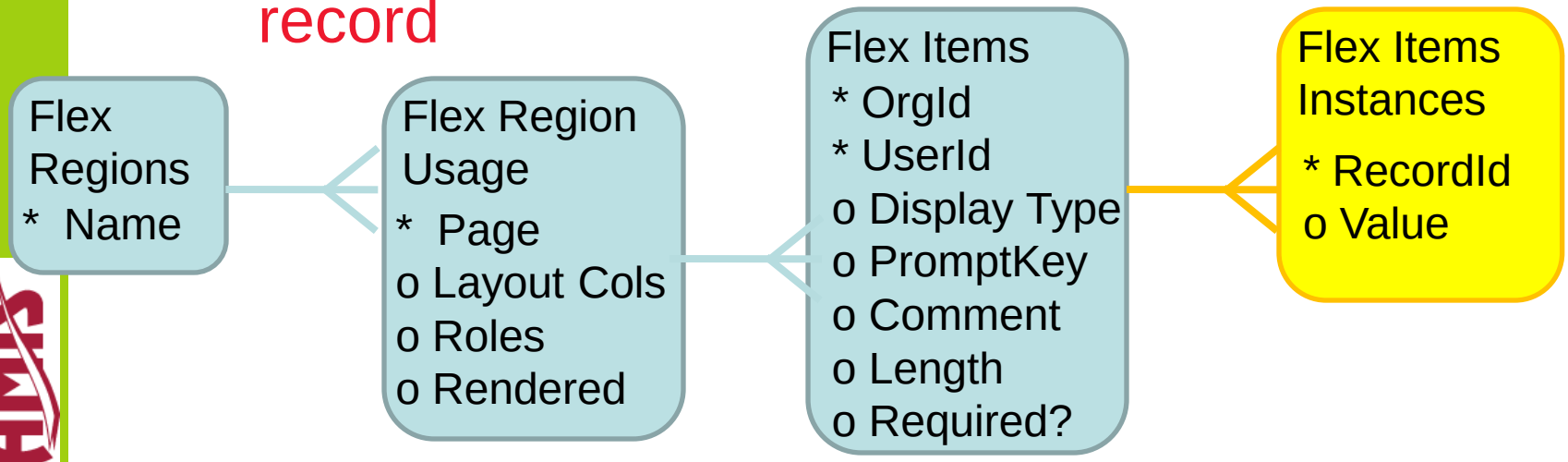


5



Implementation of Flex Items

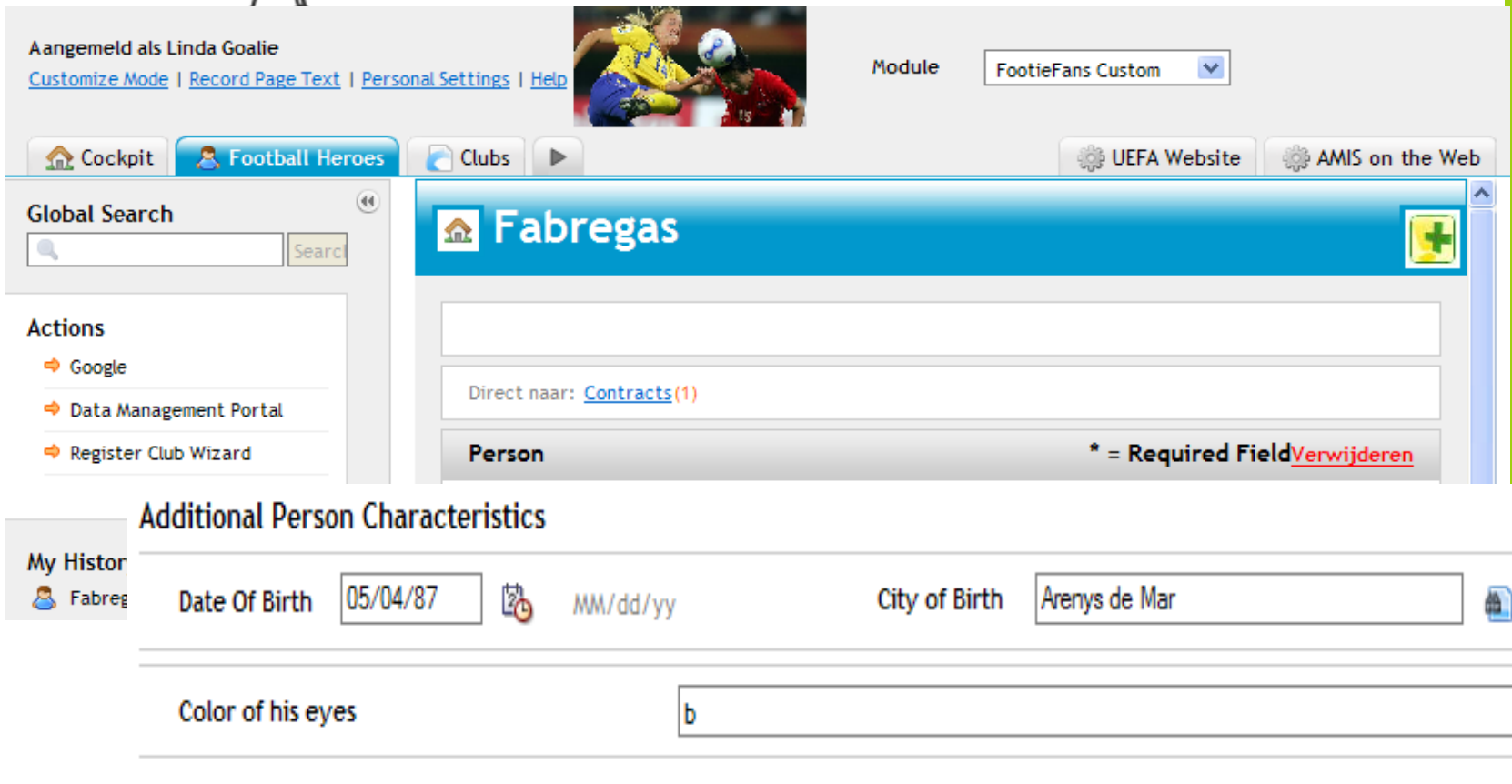
- Database:
  - Meta: tables for Flex Regions (containers for Flex Items to be page-linked) and Flex Items
  - Data: Flex Item Data Instances: based on Flex Item and associated with Primary Key of record



Implementation of Flex Items

- Every JSF Page that is flex item enabled:
  - **PanelGroup** per FlexRegionwith loop over FlexItems as
  - returned by the **FlexItemManager bean** for the page, region and Record Identifier
  - as retrieved from the **Meta Data Service**
  - Render proper component with data when available
    - inputText, selectOneChoice, inputFile,…
    - set attributes such as Rendered, Required, Label..

Login as LINDA…

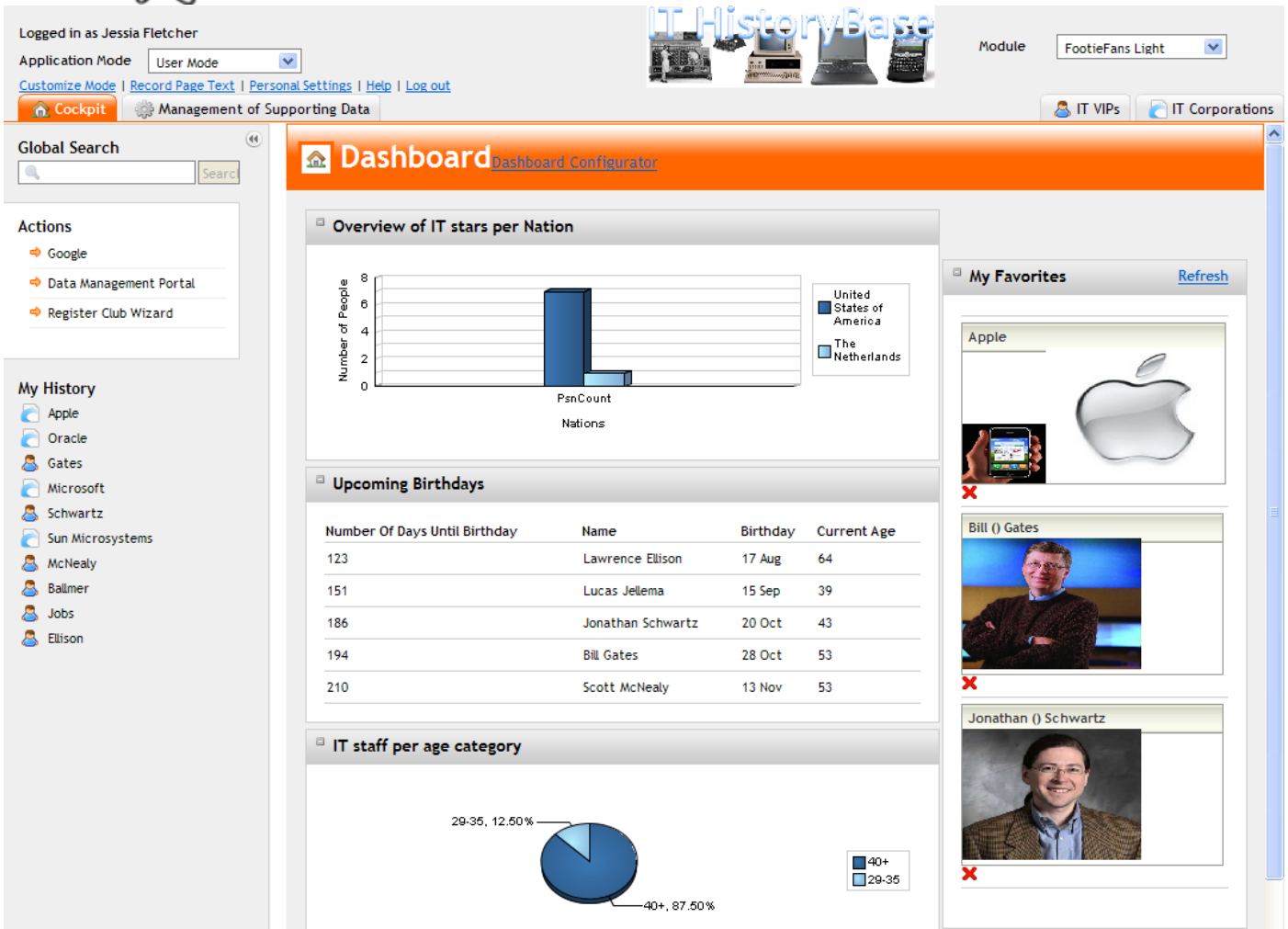


PeopleBase is *not* a Football App

- Organization
  IT HistoryBase has customized the PeopleBase application under FootieFans
  - For creating IT HistoryBase
- With the most important IT companies in the World and their top employees
  - Pictures of logos and HQ
    Photographs of IT hotshots

Demo

IT HistoryBase



Summary

- Trends in Application Development & Deployment and Administration
  - Web 2.0, RIA, Social Networking => *internet trends enter the enterprise*
  - Software as a Service
  - Customization at Organization Level
  - Personalization to fine-tune preferences
- JSF lends itself to run-time customization
  - Thanks to extensibility, programmatic control and EL expression bound attributes

Want to hear it again?

- JavaOne 2008 (5-9 May, San Francisco)
  - Birds of a Feather (BOF) session BOF-5224
- Anyone coming to work for AMIS is invited to come to JavaOne 2008
  - Or Oracle Open World (September, San Francisco)

http://technology.amis.nl/blog