

# r3 - Written in stone!

- Maktoob.com Inc.
- Outsourcing Team
- Presented By:
- **Anas K. Al-Far**

# What is the hell r3?

- r3 is a Yahoo! project.
- Used for customization.
- Used for localization.
- r3 helps you manage an application's view layer using templates and simple dictionaries.

# How does r3 works?

- It uses search path to get all needed templates to generate the final output.
- Starts with local components and end with global contents (shared).
- These components (templates) joined together using system of includes.

# Why Go r3?

- Do we provide interfaces for more than one country or language?
- Do we have distinct shared products, items, services or components?
- Do we support a number of different brands for a product at the same time?
- Do we perform some costly process at runtime (like RSS feeds)?
- Do we support different interfaces for different platforms and clients?
- Do we have versioning in our project?

# r3 objective!

- r3 uses inheritance and composition to reduce or eliminate duplication.
- To control the components so that a target can share layout as much as possible but define its own customisations where necessary.
- r3 aims to allow a similar strategy for templates and translations.
- Manage sharing and customisation when by combine templates to build targets.

# r3 Installation

- Requires PHP 5 with support for the PDO.
- Support for either MySQL, SQLite or both.
- PEAR to perform the installation.
- Get stickleback, which is the plugin engine that powers r3's command line interface from:

```
wget http://sourceforge.net/projects/stickleback
```

```
wget http://sourceforge.net/projects/rthree
```

```
pear install -f stickleback-1.0.0.tgz
```

```
pear install -f --alldeps r3-1.0.0.tgz
```

# Setting UP r3

```
#Create a working area
```

```
r3 setup setuphome -f /var/www/r3home
```

```
#define an enviroment variable (set global path)
```

```
export R3HOME=/var/www/r3home
```

```
#Change DB connection in the config file
```

```
vim r3.conf.xml
```

```
#Installing r3 DB
```

```
r3 setup installdb
```

# Dimensions and Domains

- **Dimensions:** `Product, intl, target`

- **Elements:** `cookery, us, index.php`

`wine, ca, upload.php`

- **Locations:** `cars/jp/quiz.php`

`mail/fr/about.php`

`gen_prod/gen_intl/index.php`

- **Search paths:** sets of locations.

- **Domains:** `templates, translations and pages`

# Dimensions and Domains

- User-defined element in a page dimension always has an implicit parent, called generic. search path will look like this:

`cookery/us/about.php`

`cookery/ca/generic`

# Dimensions and Domains

- For other dimensions, it's up to the user to define parental relationships.

`cookery/fr/about.php`

`cookery/fr/generic`

`cookery/all_intls/about.php`

`cookery/all_intls/generic`

`all_prods/fr/about.php`

`all_prods/fr/generic`

`all_prods/all_intls/about.php`

`all_prods/all_intls/generic`

# Products Dimension List

- cookery
- wine
- cars

# Intl Dimension List

- us
- ca
- fr
- jp

# Example 1

container--+-header

```
(Page) | |
      | |
      | | +--core branding
      | |
      | | +--site branding
      |
      |
      | +--navigation
      |
      |
      | +--content
```

# Manage Relations Rules

- "Cookery & Wine" shares the site style.
- "ca & fr" shares the language.
- "ca & us" shares much of its culture.
- "cars & cookery" have little in common.
- "ca & jp" have little in common.

# Sites Results

- Combining these two axes gives us  $3 * 4 = 12$  web sites to build:
  - cookery/us
  - cookery/ca
  - cookery/fr
  - cookery/jp
  - wine/us
  - wine/ca
  - wine/fr
  - wine/jp
  - cars/us
  - cars/ca
  - cars/fr
  - cars/jp

# Createing Dimensions

- These elements will become catch-alls for shared domain objects (Main Elements):

```
r3 dimension product create generic_product
```

```
r3 dimension intl create generic_intl
```

# Creating Dimensions

- Creating specialized dimensions:

```
r3 dimension product create cookery -p generic_product
```

```
r3 dimension product create wine -p cookery
```

```
r3 dimension product create cars -p generic_product
```

```
r3 dimension intl create us -p generic_intl
```

```
r3 dimension intl create ca -p us
```

```
r3 dimension intl create fr -p generic_intl
```

```
r3 dimension intl create jp -p generic_intl
```

# Dimensions Map

generic\_product--+-cookery--+-wine

|

+--cars

generic\_intl--+-us--+-ca

|

+--fr

|

+--jp

# Changing Inheritance Rules

- Change the inheritance relationship between fr & ca for the translation domain only:

```
r3 dimension intl parent ca set fr -d translation
```

- We also don't want any inheritance from generic\_intl in the translation domain:

```
r3 dimension intl parent fr unset -f -d translation
```

```
r3 dimension intl parent us unset -f -d translation
```

```
r3 dimension intl parent jp unset -f -d translation
```

# Testing Results

- In order to test inheritance results for the 3 domains for our products use:

```
r3 dimension intl list
```

```
generic_intl
```

```
jp
```

```
fr
```

```
us
```

```
ca
```

# Testing Results

```
r3 dimension intl list -t template
```

```
generic_intl
```

```
    jp
```

```
    fr
```

```
    us
```

```
        ca
```

# Testing Results

r3 dimension intl list -t translation

jp

us

ca

fr

generic\_intl

# Testing Results

r3 dimension product list

generic\_product

cars

cookery

wine

# Testing Results

```
r3 dimension product list -t template
```

```
generic_product
```

```
cars
```

```
cookery
```

```
wine
```

# Testing Results

```
r3 dimension product list -t translation
```

```
generic_product
```

```
cars
```

```
cookery
```

```
wine
```

# Creating a Target

- This command accepts a full path to a page, dimensions must be already exists.

```
r3 target create  
generic_product/generic_intl/index.html
```

- This command give info about a page (note the search paths):

```
r3 target info  
generic_product/generic_intl/index.html
```

- Once we have a target, we can generate an output, r3 will simply pick up the content in the template file and reproduce it in the php file:

```
r3 generate -av
```

# Editing Template

- To edit a template you need to fire up your favourite editor on our only template:

```
vim
```

```
templates/generic_product/generic_intl/index.html/index.html.ros
```

- We'll see that it consists of that single comment:

```
<!-- autogenerated default template:  
index.html.ros -->
```

# Editing Template

- Notice those three include directives:

```
<html><head></head><body><table width="100%"  
  border="2"><tr><th colspan="2">Container:  
  <r3:trans>Container</r3:trans></th></tr>  
  
<tr><td colspan="2"><r3:include path="header.ros"  
  /></td>  
  
</tr><tr><td width="30%"><r3:include  
  path="navigation.ros" /></td><td  
  width="70%"><r3:include path="content.ros"  
  /></td></tr>  
  
</table></body></html>
```

# Editing Template

- Create and edit the include files (header):

```
vim
```

```
templates/generic_product/generic_intl/index.html  
1/header.ros
```

```
<table width="100%" border="2">
```

```
<tr><th colspan=2>Header:
```

```
<r3:trans>Header</r3:trans></th><tr>
```

```
<td><r3:include path="core_branding.ros" /></td>
```

```
<td><r3:include path="site_branding.ros" /></td>
```

```
</tr></table>
```

# Editing Template

- Create and edit the include files (navigation):

```
vim
```

```
templates/generic_product/generic_intl/index.html  
l/navigation.ros
```

```
Navigation: <r3:trans>Navigation</r3:trans>
```

# Editing Template

- Create and edit the include files (content):

```
vim
```

```
templates/generic_product/generic_intl/index.html  
1/content.ros
```

```
Content
```

# Editing Template

- Create and edit the include files (core\_branding):

```
vim
```

```
templates/generic_product/generic_intl/index.html  
1/core_branding.ros
```

```
Core Branding: <r3:trans>Core Branding</r3:trans>
```

# Editing Template

- Create and edit the include files (core\_branding):

```
vim
```

```
templates/generic_product/generic_intl/index.html  
l/site_branding.ros
```

```
Site Branding: <r3:trans>Site Branding</r3:trans>
```

# Generate Template

- After creating all includes files, we should generate them:

```
r3 generate -a
```

# Example 2

Product	Intl	Contains
	specific intl	content
specific product	generic_intl	site branding, navigation
	specific intl	(header)
generic_product	generic_intl	container, header, core_branding

# Specializing Templates

- Starting with the content, In order to specialize, we need to see all of the locations that will be searched for a particular target page:

```
r3 template searchpath cookery/us/index.html
```

- If we want to see where a particular template is:

```
r3 template searchpath cookery/us/index.html  
content.ros
```

# Specializing Templates

- We want to specialize content.ros all the way to the most specific location for this target:

```
r3 template specialize cookery/us/index.html  
content.ros 0
```

- Now we can edit the cookery/us/index.html version of content.ros and make it very specific to that product:

```
vim templates/cookery/us/index.html/content.ros
```

```
Content: Cookery in US
```

# Specializing Templates

- We can use the following script to specialize content.ros for all targets:

```
for product in cookery wine cars
do for intl in us ca fr jp
do r3 template specialize $product/$intl/index.html
  content.ros 0
done
done
```

# Specializing Templates

- Now we want to edit content.ros for all targets and write the new content:

```
for product in cookery wine cars
do for intl in us ca fr jp
do vim
    templates/$product/$intl/index.html/content.ros
done
done
```

- Now we want to edit content.ros for all targets and write the new content:

```
r3 generate -a
```

# Specializing Product, with generic Intl

- Now we want to specialize `site_branding.ros` and `navigation.ros` to a specific product. First we check locations:

```
r3 template searchpath cookery/us/index.html  
site_branding.ros
```

- Now we specialize `site_branding.ros` to be specialized for `cookery` to all intls:

```
r3 template specialize cookery/us/index.html  
site_branding.ros 2
```

- Make sure that JP is updated:

```
r3 template searchpath cookery/jp/index.html  
site_branding.ros
```

# Specializing Product, with generic Intl

- **Edit site\_branding.ros for specializing:**

```
vim
```

```
templates/cookery/generic_intl/index.html/site_branding.ros
```

```
Cookery Site Branding: <r3:trans>Cookery Site  
Branding</r3:trans>
```

- **Generate new templates:**

```
r3 generate -a
```

# Specializing Product, with generic Intl

- Note that wine product inherits cookery.
- Now we should change cars product:

```
r3 template searchpath cars/us/index.html
  site_branding.ros
```

```
r3 template specialize cars/us/index.html
  site_branding.ros 2
```

- **Edit site\_branding.ros for specializing:**

```
vim
```

```
  templates/cars/generic_intl/index.html/site_brandi
  ng.ros
```

```
Cars Site Branding: <r3:trans>Cars Site
  Branding</r3:trans>
```

# Specializing Product, with generic Intl

- Generate new templates:

```
r3 generate -a
```

# Specializing Product, with generic Intl

- Now we want to specialize navigation.ros specific product. First we check locations:

```
r3 template searchpath cookery/us/index.html  
navigation.ros
```

- Now we specialize navigation.ros to be specialized for cookery to all intls:

```
r3 template specialize cookery/us/index.html  
navigation.ros 2
```

- Make sure that JP is updated:

```
r3 template searchpath cookery/jp/index.html  
navigation.ros
```

# Specializing Product, with generic Intl

- Edit navigation.ros for specializing:

```
vim
```

```
templates/cookery/generic_intl/index.html/navigation.ros
```

```
Cookery Brand Navigation: <r3:trans>Cookery  
Navigation Branding</r3:trans>
```

- **Generate new templates:**

```
r3 generate -a
```

# Specializing Product, with generic Intl

- Note that wine product inherits cookery.
- Now we should change cars product:

```
r3 template searchpath cars/us/index.html
  navigation.ros
```

```
r3 template specialize cars/us/index.html
  navigation.ros 2
```

- **Edit navigation.ros for specializing:**

```
vim
```

```
  templates/cars/generic_intl/index.html/navigation.
  ros
```

```
Cars Navigation Branding: <r3:trans>Cars Navigation
  Branding</r3:trans>
```

# Specializing Product, with generic Intl

- **Now we want to change wine product:**

```
r3 template searchpath wine/us/index.html  
navigation.ros
```

```
r3 template specialize wine/us/index.html  
navigation.ros 2
```

- **Edit navigation.ros for specializing:**

```
vim  
templates/wine/generic_intl/index.html/navigation.  
ros
```

```
Wine Navigation Branding: <r3:trans>Wine Navigation  
Branding</r3:trans>
```

# Specializing Product, with generic Intl

- Generate new templates:

```
r3 generate -a
```

# Specializing Intl

- We need to localize the header for JP to make it layed out in right to left order:

```
r3 template searchpath cars/jp/index.html header.ros
r3 template specialize cars/jp/index.html header.ros 4
vim templates/generic_product/jp/index.html/header.ros
```

- Editing that file, we change it to:

```
<table width="100%" border="2"><tr>
<th colspan=2>Japanese Header: <r3:trans>Japanese
Header</r3:trans></th><tr><td><r3:include
path="site_branding.ros" /></td><td><r3:include
path="core_branding.ros" /></td></tr></table>
```

# Specializing Intl

- Generate new translation:

```
r3 generate -a
```

# Translation

- To get translation list:

```
r3 translation list
```

- To perform translation from the r3 cli:

```
r3 translation set
```

- More common approach would be to save the translations for a particular site, or all sites, as an XLIFF file and to send that file to a third party translator:

```
r3 translation save all alltrans.xml
```

# Exporting and Importing Translations

- To get translation list:

```
r3 translation save inherited generic_product/fr  
fr.xml
```

- Edit the Translation xml File.
- Merge new translations into the dictionary:

```
r3 translation merge fr.xml
```

- More common approach would be to save the translations for a particular site, or all sites, as an XLIFF file and to send that file to a third party translator:

```
r3 translation save all alltrans.xml
```

# Specializing Intl

- Generate new templates:

```
r3 generate -a
```