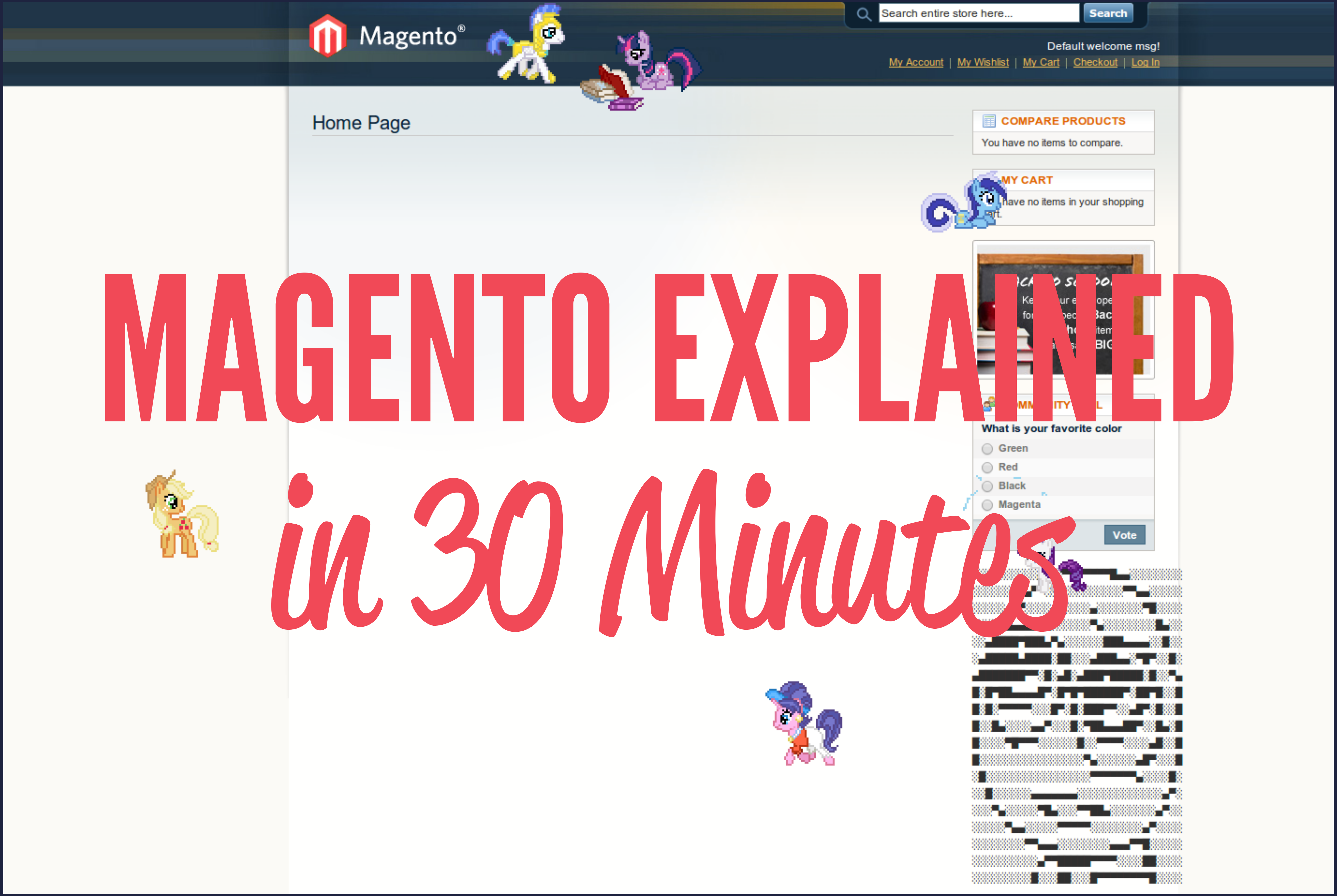
MUC Mage UNCONFERENCE

MAGENTO PEOPLE
A LOT OF TIME
YOUR TOPICS
YOUR TALKS

BERLIN 2015

#mageuc15
mageunconference.org

MAGENTO EXPLAINED in 30 Minutes



Magento®
Search entire store here...
Search
My Account | My Wishlist | My Cart | Checkout | Log In
Vote

Default welcome msg!

Home Page

COMPARE PRODUCTS

You have no items to compare.

MY CART

You have no items in your shopping cart.

What is your favorite color

- Green
- Red
- Black
- Magenta

SOMETHING TO SAY?

ON TWITTER PLEASE TAG #MHRH

BY AND WITH RIBIAN BLECHTZEL

RICO NEITZEL *and* FABIAN BLECHSCHMIDT

TODAY PRESENTED BY FABIAN BLECHSCHMIDT

SLIDES ARE ONLINE

Slides are here:
http://theslidesareonline.de/

CONTENT:

A LOT WHAT A MAGENTO DEVELOPER SHOULD KNOW

INDEX.PHP

DEVMODE

MULTISITE

CODEPOOLS

NAMESPACES

SELF WRITTEN MODULES

CONFIG.XML

FACTORYNAMES

ROUTINGWORKFLOW

FRONTNAME

CONTROLLER

ACTION

LAYOUTWORKFLOW

LOCAL.XML

LAYOUTXML

BLOCKS

TEMPLATES

CACHING

LAYOUTHANDLES

PAGETEMPLATES

FALLBACK

OBSERVER

SYSTEM CONFIGURATION

SYSTEM.XML

ACL

DEFAULTS

SETUP SCRIPTS

DATA SCRIPTS

HELPER

REWRITES

MODELS

RESOURCEMODELS

COLLECTIONS

EAV

SQL-ABSTRACTION

FORWARDS

REDIRECTS

EXCEPTIONS

OH DAMN 💩
WE'LL NEVER FINISH

AND GO!

INDEX.PHP

- Apache Configuration (never allowed)
- .htaccess changes (not often allowed)
- index.php hacks (Standard)

„Avoid Changes in index.php.“

– Ribian Blechtzel

DEVMODE

ACTIVATE ERROR REPORTING

more and start showing errors in frontend,
the show stopper for missing classes and methods

`ini_set('display_errors', 1);` uncomment

"LESS" TRANSLATIONS

Block and Translations need to be in the same module

It is only translated what is in the translate CSV in the Module (`app/locale/de_DE/My_Module.csv`),

Translation may need the Standard-Data-Helper `MyCompany_MyModule_Helper_Data`

AVOID BLOCK OVERWRITES

When two blocks have the same name…

```
# …/core/Mage/Core/Model/Layout.php:450
…
} elseif (isset($this->_blocks[$name]) && Mage::getIsDeveloperMode()) {
    //Mage::throwException(Mage::helper('core')->__('Block with name "%s" already exists', $name));
}
…
```

but the code is commented, so here happens nothing. Not even in DevMode!

DEVMODE ON/OFF

VHOST CONFIGURATION OR .HTACCESS

```
SetEnv MAGE_IS_DEVELOPER_MODE
```

no value needed, just set it

DEVMODE ON/OFF

Ugliest but most often used solution:

INDEX.PHP:

```
if (true || isset($_SERVER['MAGE_IS_DEVELOPER_MODE'])) {
```

START DIFFERENT WEBSITES/STORES DIRECTLY

VHOST CONFIGURATION / .HTACCESS

```
SetEnv MAGE_RUN_CODE {storecode}
```

or

```
SetEnvIf Host www\.domain2\.com MAGE_RUN_CODE={storecode}
```

or

```
SetEnv MAGE_RUN_CODE {websitecode}
SetEnv MAGE_RUN_TYPE website
```

works well and non-destructiv, because:

```
# index.php:82
$mageRunCode = isset($_SERVER['MAGE_RUN_CODE']) ? _SERVER['MAGE_RUN_CODE'] : '';
$mageRunType = isset($_SERVER['MAGE_RUN_TYPE']) ? _SERVER['MAGE_RUN_TYPE'] : 'store';
```

INDEX.PHP

And here again

Ugliest but most often used solution

```
switch($_SERVER['SERVER_NAME']):
    case 'domain1.de':
    case 'www.domain1.de':
        $_SERVER['MAGE_RUN_CODE'] = '{storeviewcode}';
        break;
    case 'domain2.de':
    case 'www.domain2.de':
        $_SERVER['MAGE_RUN_CODE'] = '{websitecode}';
        $_SERVER['MAGE_RUN_TYPE'] = 'website';
        break;
endswitch;

...
```

SELF WRITTEN MODULES IN MAGENTO

where else?

MODULES ARE GROUPED IN NAMESPACES

NAMESPACES

are only directories

no magic

exists to structure

everything from one module sticks together

to have less conflicts

same module name, different namespaces

FOR YOUR MODULE, YOU NEED THREE THINGS:

1. A good idea
2. a declaration file (/app/etc/modules/)
3. a configuration file (./etc/config.xml)

DECLARE YOUR OWN MODULE

Create a file in `app/etc/modules/`

{Namespace}_{Modul}.xml

Advice: Both parts should start with a capital letter!

CONTENT OF {NAMESPACE}_{MODUL}.XML

```
<?xml version="1.0"?>
<config>
    <modules>
        <{Namespace}_{Modul}>
            <active>{true|false}</active>
            <codePool>{local|community}</codePool>
            <depends>
                <Mage_Catalog />
                <Mage_Customer />
            </depends>
        </{Namespace}_{Modul}>
    </modules>
</config>
```

CLASSICAL MISTAKE

`<codePool>` is spelled with a capital P!
(see later: `<frontName>`)

WHERE TO PUT THE CODE?

The whole module code is placed in `app/code/{codePool}/{Namespace}/{Modul}/`

INACTIV

While checking whether a module is activ or not,
Magento expects the string 'true'

so `<active>1</active>`
or `<active>TRUE</active>`
is for magento a "NO, therefore inactive"

`# …/core/Mage/Core/Model/Config.php:807`

MODULJUNKIES

If you need other modules loaded before:

```
<depends>
   <Mage_Catalog />
</depends>
```

CONFIGURATION IS NEEDED

CONFIG.XML

We need the config.xml in the module directory to configure the whole module. Here all PHP classes are configured and files "linked"

explained example of a small but complete config.xml

```
<?xml version="1.0"?>
<config>
    <!-- Versionin: important for install scripts (setup and data) -->
    <modules>
        <Namespace_Modul>
            <version>0.1.0</version>
        </Namespace_Modul>
    </modules>

    <!-- Nodes fir the global configuration -->
    <global>
        <models>
            <!-- this node is called class-group and is later used in Mage::getModel() -->
            <namespace_modul>
                <!-- the class-group 'namespace_modul' is solved by magento to 'Namespace_Modul_Model' -->
                <class>Namespace_Modul_Model</class>
                <resourceModel>namespace_modul_resource</resourceModel>
            </namespace_modul>
            <!-- this node is called class-group too and is later used indirectly by Mage::getResourceModel()
            <namespace_modul_resource>
                <class>Namespace_Modul_Model_Resource</class>
                <entities>
                    <!-- this node is called entity and is used to find table names -->
                    <developer>
                        <!-- the table name itself -->
                        <table>developer</table>
                    </developer>
                </entities>
            </namespace_modul_resource>
        </models>
        <blocks>
            <!-- this node is again a class-group and is used by Mage::getBlock -->
            <namespace_modul>
                <!-- the class-group 'namespace_modul' is solved by Magento to 'Namespace_Modul_Block' -->
                <class>Namespace_Modul_Block</class>
            </namespace_modul>
        </blocks>
        <helpers>
            <!-- hey, a class-group. This is used in Mage::helper() -->
            <namespace_modul>
                <!-- and the solution this time is: 'Namespace_Modul_Helper' -->
                <class>Namespace_Modul_Helper</class>
            </namespace_modul>
        </helpers>
        <resources>
            <!-- this node defines the name of the sub directory name in ./sql und ./data for the install scripts -->
            <namespace_modul_setup>
                <setup>
                    <module>Namespace_Modul</module>
                </setup>
            </namespace_modul_setup>
        </resources>
    </global>
</config>
```

SUBNODES IN CONFIG.XML

MODULES

```
<modules>
    <{Namespace_Modul}>
        <version>
```

GLOBAL /1

```
<models>
    <{namespace_modul}>
        <class>
        <resourceModel>
    <{namespace_modul}_resource>
        <deprecatedNode>
        <entities>
            <{entity}>
                <table>
        <rewrite>
            <{model}>
```

GLOBAL /2

```
<helpers>
    <{namespace_modul}>
        <class>
        <rewrite>
            <{helper}>

<blocks>
    <{namespace_modul}>
        <class>
        <rewrite>
            <{block}>
```

GLOBAL /3

```
<index>
    <indexer>
        <{indexer_code}>
            <model>
            <depends>
                <{indexer_code}/>

<cache>
    <types>
        <{cache_code}>
            <label>
            <description>
            <tags>
```

GLOBAL /4

```
<resources>
    <{namespace_modul}_setup>
        <connection>
            <use>
        <setup>
            <module>
            <class>

<template>
    <email>
        <{email_template_code}>
            <label>
            <file>
            <type>

<page>
    <layouts>
        <{page_template_code}>
            <label>
            <template>
            <layout_handle>
            <is_default>
```

GLOBAL /5

```
<events>
    <{event_name}>
        <observers>
            <{namespace_modul}>
                <type>
                    <model>
                    <class>
                    <method>
```

GLOBAL /6

```
<sales>
    <quote>
        <item>
            <product_attributes>
                <sku/>
                <type_id/>
                <name/>
                <status/>
                <visibility/>
                <price/>
                <weight/>
                <url_path/>
                <url_key/>
                …
```

INSTALL

```
<events>
    <{event_name}>
        <observers>
            <{namespace_modul}>
                <type>
                    <model>
                    <class>
                    <method>

<translate>
    <modules>
        <{Namespace_Modul}>
            <file>
                <default>
                <others>

<layout>
    <updates>
        <{namespace_modul}>
            <file>
```

ADMIN

```
<routers>
    <{routername}>
        <use>
        <args>
            <module>
            <frontName>
            <modules>
                <{Namespace_Modul}>
```

ADMINHTML /1

```
<events>
    <{event_name}>
        <observers>
            <{namespace_modul}>
                <type>
                <model>
                <class>
                <method>

<translate>
    <modules>
        <{Namespace_Modul}>
            <file>
                <default>
                <others>
```

ADMINHTML /2

```
<layout>
    <updates>
        <{namespace_modul}>
            <file>

<global_search>
    <{entity}>
        <class>
```

FRONTEND /1

```
<category>
    <collection>
        <attributes>
            <{category_attributecode_to_load}>

<product>
    <collection>
        <attributes>
            <{product_attributecode_to_load}>

<routers>
    <{routername}>
        <use>
        <args>
            <module>
            <frontName>
            <modules>
                <{Namespace_Modul}>
```

FRONTEND /2

```
<events>
    <{event_name}>
        <observers>
            <{namespace_modul}>
                <type>
                    <model>
                    <class>
                    <method>

<translate>
    <modules>
        <{Namespace_Modul}>
            <file>
                <default>
                <others>

<layout>
    <updates>
        <{namespace_modul}>
            <file>
```

CRONTAB

```
<crontab>
    <jobs>
        <{cronjob_identifier}>
            <schedule>
                <cron_expr>
            <run>
                <model>
```

FACTORYNAMES

In Magento `new Klassenname()` doesn't exist

Instead Factories with FactoryNames via XML-Konfiguration are used

and yes, that is one reason, why magento is so damn complex

ONE EXAMPLE BASED ON A BLOCKS

catalog/product_view

is solved to

Mage_Catalog_Block_Product_View

and lives in the file

/app/code/core/Mage/Catalog/Block/Product/View.php

How the part BEFORE / is solved we'll explained later.
The part AFTER / is solved with this method:

```
str_replace(" ", DS, ucfirst(str_replace("_", " ", $secondTeil))).'.php'
```

IN NORMAL WORDS

1. replace all _ with spaces
2. change all words to first letter uppercase
3. replace all spaces with the constant DS
4. add .php at the end

DS means DirectorySeparator and depending on the Operating System contains / or \ ... \o/ YAY!

GROSSBUCHSTABENSALAT

my_module/grOSSbuChstaBenSaLAT
is translated to
My_Module_Block_GrOSSbuChstaBenSaLAT
and lives in the file
/app/code/local/My/Module/Block/GrOSSbuChstaBenSaLAT.php

ROUTING-WORKFLOW

FROM INDEX.PHP UP TO CONTROLLER

SELF WRITTEN FRONTNAME (ROUTE)

IN CONFIG.XML

```
<config>
    <frontend>
        <routers>
            <{namespace_module}>
                <use>standard</use>
                <args>
                    <module>{Namespace_Module}</module>
                    <frontName>{frontname}</frontName>
                </args>
            </{namespace_module}>
        </routers>
    </frontend>
</config>
```

ADVICE

The node frontName is spelled with a capital N!
(see also `<codePool>`)

So you can access in the frontend:

```
http://domain.tld/{frontname)/
```

Example:

```
http://domain.tld/ribian/
<frontName>ribian</frontName>
```

SELF WRITTEN CONTROLLER

Create in your own module the file IndexController.php
in the directory controllers
(Attention: lower case and plural!)

The classname contains parts of the directory path:
Namespace_Module_IndexControllers

This class have to extend

`Mage_Core_Controller_Front_Action`.

So you can use

`http://domain.tld/{frontname)/index/`
in frontend

THE SECOND PART OF THE URL

```
http://domain.tld/{frontname)/{controller}/
```

(here {controller}) is the start of the filename in the controllers directory: {Controller}Controller.php.

Attention: The first letter needs to be UPPERCASE, because magento loads the class via Autoloader

example:

```
http://domain.tld/ribian/blechtzel/

# is looking for
./controllers/BlechtzelController.php
```

SELF WRITTEN ACTION

```
http://domain.tld/{frontname)/{controller}/{action}/
```

Actions are **public methods** in the Controller class.

The name of this method is always made up of the URL part {action} and "Action"

```
http://domain.tld/ribian/blechtzel/mergeface

public function mergefaceAction()
```

Attention: **The first letter has to be lowercase!**

```
http://domain.tld/ribian/blechtzel/mergeface
```

ribian `<frontName>ribian</frontName>`

blechtzel `./controllers/BlechtzelController.php`

mergeface `public function mergefaceAction()`

FRONTNAME/CONTROLLER/ACTION SUMMARY

SELF WRITTEN ACTION

Layout handle takes the NODE name!

```xml
<routers>
    <companyModule_frontend>
        <use>standard</use>
        <args>
            <module>Company_Module</module>
            <frontName>frontend</frontName>
        </args>
    </companyModule_frontend>
</routers>
```

Route: frontend/<controllerName>/<actionName>
Layout-Handle: companyModule_frontend (NODE NAME!)/<controllerName>/<actionName>

ADDING YOUR OWN GET PARAMETER

GET parameters can be added via /key/value to the URL

```
http://domain.tld/ribian/blechtzel/mergeface/key/value
```

```
array("key" => "value");
```

ROUTING WORKFLOW EN DETAIL

Index.php calls Mage::run()

Mage::run() calls Mage_Core_Model_App::run()

Mage_Core_Model_App::run() loads Mage_Core_Controller_Varien_Front ('FrontController');

FrontController gathers all available Routes of the modules afterwards run() calls dispatch() on the FrontController dispatch() match()'es the requested URL against all Routes match() checks all available Controller classes match() checks for available Action methods in the found Controller classes

after that dispatch() is called
on the found Controller class

(which is implemented in Mage_Core_Controller_Varien_Action) dispatch() calls finally the Action method on the Controller class.

LAYOUT-WORKFLOW

FROM LAYOUT-XML TO HTML

LOCAL.XML

Just throw a local.xml in your own theme

```xml
<?xml version="1.0"?>
<layout>
    <default>
    </default>
</layout>
```

LOCAL.XML

So you can influence the layout with your own layout.xml
The local.xml is loaded automatically from magento AT THE END!

Advantage: The local.xml is not overwritten during an update!

ADD AND REMOVE CSS FILES

```
<!-- load block with the name 'header' -->
<reference name="head">
    <!-- call the method 'addItem' on the block -->
    <action method="addItem">
        <!-- pass the following four parameters to the method  -->
        <!-- define the HTML tag type and the root directory for the file -->
        <type>skin_css</type>
        <!-- relative path to the file -->
        <filename>{relative/path/in/theme/file.css}</filename>
        <!-- define html attributes if needed, if not, pass an empty node: <params/> -->
        <!-- when there is no <if> needed, you can skip the parameters -->
        <params>{attribute="value"}</params>
        <!-- add a conditional comment -->
        <if>{lt IE 7}</if>
    </action>
</reference>
```

ADD AND REMOVE JAVASCRIPT FILES

```
<reference name="head">
    <action method="addItem">
        <type>skin_js</type>
        <filename>{relative/path/in/theme/file.js}</filename>
        <params>{attribute="value"}</params>
        <if>{lt IE 7}</if>
    </action>
</reference>
```

ALLOWED VALUES FOR `<type>`

- skin_css – CSS file relative to the theme directory, theme CSS files
- skin_js – JavaScript files relative to the theme directory (js)
- js_css – CSS files relative to the root /js directory (CSS for js libs)
- js – JS files relative to the root /js directory (JS libs)

UNUSUAL VALUES FOR `<type>`

- rss – RSS-URL
- link_rel – <link> Url

SELF WRITTEN BLOCK

For your own block you need a config.xml

and a class file with one of the following classes extended:
Mage_Core_Block_Abstract
or Mage_Core_Block_Template

Think about setting the $template property

CONFIG.XML

```
<config>
    <global>
        <blocks>
            <!-- this node is called: class-group
            (and is later used in Mage::getBlock()) -->
            <{namespace_modul}>
                <!-- the class-group 'namespace_modul'
                is solved by Magento to 'Namespace_Modul_Block' -->
                <class>{Namespace_Modul}_Block</class>
            </{namespace_modul}>
        </blocks>
    </global>
</config>
```

```
<!-- Layout XML -->
<block type="namespace_modul/ribian" …>`

<!-- config.xml -->
<global>
    <blocks>
        <{namespace_modul}>
            <!-- NO LINE BREAK in  <class>!! -->
            <class>Namespace_Modul_Block</class>
```

then add the 2nd part of the factory name: _Ribian

```
Namespace_Modul_Block_Ribian
```

CLASS FILE

The class file for Namespace_Modul_Block_Ribian have, according to its name, to live in app/code/local/Namespace/Modul/Block/ and is named Ribian.php.

INHERITANCE

Most of the block classes extends, when they have a phtml file, the class Mage_Core_Block_Template.

It is NOT enough to extend Mage_Core_Block_Abstract.

PRESET TEMPLATE

Block with Template? Then preset the phtml file, in case the frontend dev forgets it.

Can be done in _construct()'or

```
protected function _construct()
{
    $this->setTemplate('{modul}/pfad/zur/datei.phtml');
    parent::_construct();
}
```

BLOCK CACHING

You need:

- Lifetime in seconds
- Key, to distinguish the variants
- CacheTag, to group them and bulk delete them

BLOCK CACHING

```php
// app/code/core/Mage/Core/Block/Abstract.php:904 - shortened

final public function toHtml()
{
    $html = $this->_loadCache();
    if ($html === false) {
        $this->_beforeToHtml();
        $html = $this->_toHtml();
        $this->_saveCache($html);
    }
    $html = $this->_afterToHtml($html);
```

SURPRISING FINDING

is only called, when not cached:

```
$this->_beforeToHtml();
```

is called every time, even cached:

```
$html = $this->_afterToHtml($html);
```

BLOCK CACHING

```php
# …/core/Mage/Core/Block/Abstract.php:1362

public function getCacheLifetime()
{
    if (!$this->hasData('cache_lifetime')) {
        return null;
    }
    return $this->getData('cache_lifetime');
}
```

overwrite `getCacheLifetime()` **or** `$this->setCacheLifetime(120)`

BLOCK CACHING

```
# …/core/Mage/Core/Block/Abstract.php:1289

public function getCacheKey()
{
    if ($this->hasData('cache_key')) {
        return $this->getData('cache_key');
    }
    $key = $this->getCacheKeyInfo();
    $key = array_values($key);  // ignore array keys
    $key = implode('|', $key);
    $key = sha1($key);
    return $key;
}
```

BLOCK CACHING

```
protected function getCacheKeyInfo() {}
```

overwrite or

```
$this->setCacheKey(
    $storeId.
    $customerId.
    $productId
);
```

BLOCK CACHING

```
# …/core/Mage/Core/Block/Abstract.php:1311

public function getCacheTags()
{
    if (!$this->hasData('cache_tags')) {
        $tags = array();
    } else {
        $tags = $this->getData('cache_tags');
    }
    $tags[] = self::CACHE_GROUP;
    return $tags;
}
```

BLOCK-CACHING

```
$this->setCacheTags(array(
    Mage_Catalog_Model_Product::CACHE_TAG,
    'STORE_'.$storeId.'-PRODUCTS',
));
```

One cache entry can have **multiple** cache tags.

SELF WRITTEN LAYOUT.XML

SELF WRITTEN LAYOUT.XML

CONFIG.XML

```
<frontend>
    <layout>
        <updates>
            <namespace_modul>
                <file>namespace/modul.xml</file>
```

lives in /app/design/frontend/base/default/layout/

PHTANDARD LAYOUT-HANDLE

PHTEVEN?
IS THAT YOU?...

PHTANDARD LAYOUT-HANDLE

The most important Layout-Handle which don't belong to a page:

- <default> (all pages)
- <customer_logged_(in|out)> (customer status)
- <catalog_category_default> (category w/I LN)
- <catalog_category_layered> (category with LN)
- <catalog_category_view> (all categories)
- <catalog_product_view> (product detail page)

SPECIAL LAYOUT-HANDLE

<PRODUCT_TYPE_(simple|configurable|grouped|virtual|downloadable|bundle{|giftcard})

SELF WRITTEN LAYOUT HANDLE

Easiest: add one with an observer

```
public function addMyOwnLayoutHandle(){
    Mage::app()->getLayout()->addHandle('my_own_handle');
}
```

than you can use in layout XML

```
<layout>
    <my_own_handle>
```

SELF WRITTEN PAGE TEMPLATE

LET'S CALL IT 4COLUMNS.PHTML

config.xml for page template and fourcolumns.xml

```
<frontend>
    <layout>
        <updates>
            <file>fourcolumns.xml</file>
<global>
    <page>
        <layouts>
            <four_columns module="page" translate="label">
                <label>4 columns</label>
                <template>page/4columns.phtml</template>
                <layout_handle>page_four_columns</layout_handle>
            </four_columns>
```

create fourcolumns.xml

```
<layout>
    <page_four_columns>
        <reference name="root">
            <block type="core/text_list" name="four" />
            <action method="setTemplate">
                <file>page/4columns.phtml</file>
```

create 4columns.phtml

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="<?php echo $this->getLang() ?>" lang="<?php echo $this->getLang() ?>">
<head>
<?php echo $this->getChildHtml('head') ?>
</head>
<body<?php echo $this->getBodyClass()?' class="'.$this->getBodyClass().'"':'' ?>>
<?php echo $this->getChildHtml('after_body_start') ?>
<div class="wrapper">
    <?php echo $this->getChildHtml('global_notices') ?>
    <div class="page">
        <?php echo $this->getChildHtml('header') ?>
        <div class="main-container col3-layout">
            <div class="main">
                <?php echo $this->getChildHtml('breadcrumbs') ?>
                <div class="col-wrapper">
                    <div class="col-main">
                        <?php echo $this->getChildHtml('global_messages') ?>
                        <?php echo $this->getChildHtml('content') ?>
                    </div>
                    <div class="col-left sidebar"><?php echo $this->getChildHtml('left') ?></div>
                </div>
                <div class="col-right sidebar"><?php echo $this->getChildHtml('right') ?></div>
            </div>
            <div class="col-right sidebar"><?php echo $this->getChildHtml('four') ?></div>
        </div>
        <?php echo $this->getChildHtml('footer') ?>
        <?php echo $this->getChildHtml('global_cookie_notice') ?>
        <?php echo $this->getChildHtml('before_body_end') ?>
    </div>
</div>
<?php echo $this->getAbsoluteFooter() ?>
</body>
</html>
```

SELF WRITTEN BLOCK TEMPLATE

Templates are PHP+HTML files (.phtml)

SELF WRITTEN BLOCK TEMPLATE

Define the template inside of block class

```
$block->setTemplate('meinTemplate.phtml');
```

or in layout XML

```
<block type="core/template" name="…" template="folder/filename.phtml" />
<!-- oder -->
<reference name="blockname">
    <action method="setTemplate">
        <val>folder/filename.phtml</val>
    </action>
</reference>
```

THEME FALLBACK

1. *your* theme for file type in *your* package
2. *your* default theme in *your* package
3. default theme in *your* package
4. default theme in base package

THEME FALLBACK

You want to have a custom fallback?

.../core/Mage/Core/Model/Design/Package.php:392

have fun ...

IMPLEMENT OBSERVERS

FIND THE RIGHT EVENT

MODELS

Before and after every ->load()

```
# …/core/Mage/Core/Model/Abstract.php:255
Mage::dispatchEvent($this->_eventPrefix.'_load_before', $params);

# …/core/Mage/Core/Model/Abstract.php:267
Mage::dispatchEvent($this->_eventPrefix.'_load_after', $this->_getEventData());
```

Before and after every ->save()

```
# …/core/Mage/Core/Model/Abstract.php:391
Mage::dispatchEvent($this->_eventPrefix.'_save_before', $this->_getEventData());

# …/core/Mage/Core/Model/Abstract.php:466
Mage::dispatchEvent($this->_eventPrefix.'_save_after', $this->_getEventData());
```

Before, during and after every ->delete()

```
# …/core/Mage/Core/Model/Abstract.php:501
Mage::dispatchEvent($this->_eventPrefix.'_delete_before', $this->_getEventData());

// INSIDE the database transaction!
# …/core/Mage/Core/Model/Abstract.php:529
Mage::dispatchEvent($this->_eventPrefix.'_delete_after', $this->_getEventData());

// after the transaction
# …/core/Mage/Core/Model/Abstract.php:541
Mage::dispatchEvent($this->_eventPrefix.'_delete_commit_after', $this->_getEventData());
```

BLOCKS

before and after _prepareLayout()

```
# …/core/Mage/Core/Block/Abstract.php:237
Mage::dispatchEvent('core_block_abstract_prepare_layout_before',
array('block' => $this));

$this->_prepareLayout();

Mage::dispatchEvent('core_block_abstract_prepare_layout_after',
        array('block' => $this));
```

At the beginning of `_toHtml()`

```
# …/core/Mage/Core/Block/Abstract.php:850
Mage::dispatchEvent('core_block_abstract_to_html_before',
array('block' => $this));
```

At the end of `_toHtml()`

```
# …/core/Mage/Core/Block/Abstract.php:886
Mage::dispatchEvent('core_block_abstract_to_html_after',
array('block' => $this, 'transport' => self::$_transportObject));
```

Both are fired although with active block caching!

EXAMPLES

```
# …/core/Mage/Checkout/Model/Cart.php:296
Mage::dispatchEvent('checkout_cart_product_add_after',
array('quote_item' => $result, 'product' => $product));

# …/core/Mage/Checkout/Model/Cart.php:396
Mage::dispatchEvent('checkout_cart_update_items_before',
array('cart'=>$this, 'info'=>$data));

// Change the price ofa quote item -> change product
Mage::dispatchEvent('sales_quote_item_set_product', array(
    'product' => $product,
    'quote_item'=>$this
));
```

SEARCH, SEARCH, SEARCH

There is an event for everything, nearly
(except sending mails)

CONFIG.XML

```
<events>
    <sales_order_save_before>
        <observers>
            <namespace_modul>
                <type>singleton</type>
                <class>Namespace_Modul_Model_Observer</class>
                <method>salesOrderSaveBefore</method>
```

`<type>`

- singleton
- model or object
- disabled

```
Mage::getModel(…)
Mage::getSingleton(…)
```

`<class>`

Parameters for `getModel/getSingleton`
Best pratice: `My_Class` instead of `namespace/model`

CHANGE EVENT

- disable old observer: `disabled`
- implement your new one

`<method>`

it's the methodname - *no really!*

OBSERVER.PHP

```php
public function salesOrderSaveBefore(Varien_Event_Observer $observer)
{
    $order = $observer->getOrder();
    $order->setOldState($order->getState());
}
```

SYSTEM CONFIGURATION

Provides fields and select boxes, etc. inside of System / Configuration

SYSTEM.XML

```xml
<?xml version="1.0"?>
<config>
  <tabs />
  <sections>
    <groups>
      <fields />
    </groups>
  </sections>
</config>
```

TABS

Tabs are on the left side

```
  <tabs>
    <tabname translate="label" module="classgroup">
      <label>Label for tab</label>
      <sort_order>100</sort_order>
```

^module defines the used helper for translation!

# Sections

```xml
  <sections>
    <sectionname translate="label" module="classgroup">
      <tab>tabname</tab>
      <label>Label for section</label>
      <sort_order>10</sort_order>
      <show_in_default>1</show_in_default>
      <show_in_website>1</show_in_website>
      <show_in_store>1</show_in_store>
      <groups />
```

sections are part of the tabs and are also on the left

GROUPS

```
<sections>
  <sectionname translate="label" module="classgroup">
    <groups>
      <groupname translate="label" module="classgroup">
        <label>Label for group</label>
        <sort_order>10</sort_order>
        <show_in_default>1</show_in_default>
        <show_in_website>1</show_in_website>
        <show_in_store>1</show_in_store>
        <fields />
```

groups are the expandable parts on the right

FIELDS

```
<sections>
  <sectioname translate="label" module="classgroup">
    <groups>
      <groupname translate="label comment" module="classgroup">
        <fields>
          <fieldname translate="label">
          <label>Label for field</label>
          <comment><![CDATA[text for the comment]]></comment>
          <frontend_type>text</frontend_type>
          <source_model>adminhtml/system_config_source_yesno</source_model>
          <sort_order>10</sort_order>
          <show_in_default>1</show_in_default>
          <show_in_website>1</show_in_website>
          <show_in_store>1</show_in_store>
```

are the fields and selects on the right

OFTEN USED FRONTEND TYPES

TEXT, TEXTAREA, PASSWORD, SELECT, MULTISELECT

Attention: If you use select or multiselect you need a Source Model, which delivers the data for the dropdown

Want more? Have fun: /lib/Varien/Data/Form/Element

The node names of section / group / field are merged to the XML path in system configuration!

ADMIN ACL - ADMINHTML.XML

ACCESS CONTROL LIST

without logout/login there is a 404,
because the acl is saved in session
and is not up to date

- or just use this cool snippet (acl refresh):

```
$session = $adminuser = Mage::getSingleton('admin/session');
/* @var $adminuser Mage_Admin_Model_User */
$adminuser = $session->getUser();
$adminuser->setReloadAclFlag(true);
$session->refreshAcl();
```

ADMINHTML.XML

```
<acl>
  <resources>
   <admin>
    <children>
     <system>
      <children>
       <config>
        <children>
         <sectionname translate="title" module="classgroup">
          <title>Title of ACL</title>
```

So you can control the access on the sections of system configuration

SYSTEM CONFIGURATION

NOT IN THE DATABASE

In config.xml is enough room for everything,
for the system configuration too.

It is super, when you habe system configuration
in XML instead of Backend

Versioning, Reproducable, Reliable

DEFAULT

```
<default>
   <general>
       <locale>
           <firstday>1</firstday>
       </locale>
   </general>
</default>
```

Attention: When there is a value in `core_config_data`, magento uses the one from database!

DEFAULTS FOR

WEBSITES

The same which works for `<default>` can be set for different websites

Just put the configuration in this path:

```
<websites>
    <{website_code}>
```

DEFAULTS FOR

STORES

And again, you can store configuration for different stores

Just put it in

```
<stores>
    <{store_code}>
```

SELF WRITTEN SETUP UND DATA SCRIPTS

living in the directory `./sql` or `./data`

CONFIG.XML

```
<resources>
    <namespace_modul_setup>
        <setup>
            <class>Mage_Catalog_Model_Resource_Setup</class>
            <module>Namespace_Modul</module>
        </setup>
    </namespace_modul_setup>
</resource>
```

Attention: When you create this XML, the scripts runs, if there is no script nothing happens. But the version is raised in `core_resource` and the next time, WITH a script magento thinks it already ran.

VERSION NUMBER

Mgento decides based on versionnumber of the module and the database, whether installscripts have to run

Naming convention for the files:

- install-<version>.php
- upgrade-<version_von>-<version_nach>.php

DATA SCRIPTS

The same for data scripts.

Naming convention:

- data-install-<version>.php
- data-upgrade-<version_von>-<version_nach>.php

HELPER

The default helper's name is always: Data

This name is part of the filename.

CONFIG.XML

```
<global>
    <helpers>
        <namespace_modul>
            <class>Namespace_Modul_Helper</class>
```

INHERIT FROM MAGE_CORE_HELPER_ABSTRACT

```
class Namespace_Modul_Helper_Data
    extends Mage_Core_Helper_Abstract
{

    public function pleaseHelpMe(){
        return true;
    }

}
```

WHAT ARE HELPER FOR?

only for helper methods like:

- createCamelCase()
- escapeHtml()

REWRITES

REWRITES

simple und relatively "pain-free"
change functions or
add new features.

REWRITES

CLASSES YOU CAN'T REWRITE

- `Mage`
- `Mage_Core_Model_App`
- `Mage_Core_Model_Config`
- loaded via `new Mage_*`

```php
// \Mage::run
self::$_app    = new Mage_Core_Model_App();
self::$_config = new Mage_Core_Model_Config($options);
```

MODEL

Proxy for ORM layer and business logic container

Means

it forwards saving, deleting, changing and
may change or edit the data before.

CONFIG.XML

```
<global>
    <models>
        <namespace_modul>
            <class>Namespace_Modul_Model</class>
```

EXTENDS MAGE_CORE_MODEL_ABSTRACT

```
class Namespace_Modul_Model_Hotel
    extends Mage_Core_Model_Abstract
{

    protected function _construct(){
        $this->_init('namespace_model/hotel');
    }

}
```

RESOURCE MODEL

ORM layer for entities from the database

Means

Offers features like saving, creating and deleting for entities from the database.

Eaiser

You can save something to the database and delete.

CONFIG.XML

```
<global>
    <models>
        <namespace_modul>
            <class>Namespace_Modul_Model</class>
            <resourceModel>namespace_modul_resource</resourceModel>
        <namespace_modul>
        <namespace_modul_resource>
            <class>Namespace_Modul_Model_Resource</class>
```

EXTENDS MAGE_CORE_MODEL_RESOURCE_DB_ABSTRACT

```
class Namespace_Modul_Model_Resource_Hotel
    extends Mage_Core_Model_Resource_Db_Abstract
{

    protected function _construct(){
        $this->_init('namespace_model/hotel', 'id');
    }

}
```

These 2nd parameter is the primary key!

RESOURCE COLLECTION AKA COLLECTION

Is a container,
which loads multiple models at once
from the database

CONFIG.XML

is not needed, is built automagically.

EXTENDS MAGE_CORE_MODEL_RESOURCE_DB_COLLECTION_ABSTRACT

```
class Namespace_Modul_Model_Resource_Hotel_Collection
    extends Mage_Core_Model_Resource_Db_Collection_Abstract
{

    protected function _construct(){
        $this->_init('namespace_model/hotel');
    }

}
```

EAV VS. NORMAL COLLECTIONS

EAV collections don't load automatically all attributes.

EAV

ENTITITY ATTRIBUTE VALUE

EAV saves attribute vales in different tables.

So it is highly flexible for saving data

But super slow while saving and loading data

```
- entity
  +- entity-datetime
  +- entity-decimal
  +- entity-int
  +- entity-text
  +- entity-varchar
```

Models without EAV save all data in one row in one table.

So you can say all attributes are static.

EAV based models have static attributes too.

These are columns in entity-table.

SQL ABSTRACTION

SQL ABSTRACTION

FILTER METHODS

```
$collection->addFieldToFilter($field, array($operator => $maybe_filter_value);
$collection->addAttributerToFilter($field, array($operator => $maybe_filter_value);
```

WHERE OPERANDS

```
Magento-Option  |    SQL
--------------------------
eq              |    =
neq             |    !=
like            |    LIKE
nlike           |    NOT LIKE
in              |    IN
nin             |    NOT IN
is              |    IS
notnull         |    IS NOT NULL
null            |    IS NULL
moreq           |    >=
gt              |    >
lt              |    <
gteq            |    >=
lteq            |    <=
```

WHERE COMES THE PRICE FROM?

WHERE COMES THE PRICE FROM?

- on category pages
- on product detail pages

CATEGORY PAGE

```
CREATE TABLE `catalog_product_index_price` (
  `entity_id` int(10) unsigned NOT NULL COMMENT 'Entity ID',
  `customer_group_id` smallint(5) unsigned NOT NULL COMMENT 'Customer Group ID',
  `website_id` smallint(5) unsigned NOT NULL COMMENT 'Website ID',
  `tax_class_id` smallint(5) unsigned DEFAULT '0' COMMENT 'Tax Class ID',
  `price` decimal(12,4) DEFAULT NULL COMMENT 'Price',
  `final_price` decimal(12,4) DEFAULT NULL COMMENT 'Final Price',
  `min_price` decimal(12,4) DEFAULT NULL COMMENT 'Min Price',
  `max_price` decimal(12,4) DEFAULT NULL COMMENT 'Max Price',
  `tier_price` decimal(12,4) DEFAULT NULL COMMENT 'Tier Price',
  ...
```

PRODUCT DETAIL PAGE

```
// Mage_Catalog_Model_Product_Type_Price::getFinalPrice

1. $product->getPrice()
    // Price attribute
2. $this->_applyTierPrice($product, $qty, $finalPrice);
    // tier prices
3. $this->_applySpecialPrice($product, $finalPrice);
    // special price (from/until date)
4. Mage::dispatchEvent('catalog_product_get_final_price',
        array('product'=>$product, 'qty' => $qty));
     // change the price via event if needed
     4a. Mage_CatalogRule_Model_Observer::processFrontFinalPrice
         // catalog price rules
5. $this->_applyOptionsPrice($product, $qty, $finalPrice);
   // price of options are added
6. return max(0, $finalPrice);
   // price is at least 0
```

REDIRECTIONS

- via Exception
- via Request

VIA EXCEPTION

```
    // heavily shortened
    public function dispatch($action) {
        try {
            $this->preDispatch();
            if ($this->getRequest()->isDispatched()) {
                if (!$this->getFlag('', self::FLAG_NO_DISPATCH)) {
                    $this->$actionMethodName();
                    $this->postDispatch();
                }
            }
        }
        catch (Mage_Core_Controller_Varien_Exception $e) {
            // ...
        }
    }

# class Mage_Core_Controller_Varien_Exception extends Exception
```

BEST PRACTICE

`try ... catch` tests for `Mage_Core_Exception`, not `Exception`!

otherwise `Mage_Core_Controller_Varien_Exception` would be catched by mistake.

VIA EXCEPTION - 2

```
catch (Mage_Core_Controller_Varien_Exception $e) {
    list($method, $parameters) = $e->getResultCallback();
    switch ($method) {
        case Mage_Core_Controller_Varien_Exception::RESULT_REDIRECT:
            list($path, $arguments) = $parameters;
            $this->_redirect($path, $arguments);
            break;
        case Mage_Core_Controller_Varien_Exception::RESULT_FORWARD:
            list($action, $controller, $module, $params) = $parameters;
            $this->_forward($action, $controller, $module, $params);
            break;
        default:
            $actionMethodName = $this->getActionMethodName($method);
            $this->getRequest()->setActionName($method);
            $this->$actionMethodName($method);
            break;
    }
}
```

FORWARD IN PRACTICE

```
$exception = new Mage_Core_Controller_Varien_Exception();
// $exception->prepareForward(
//     $actionName, $controllerName, $moduleName, $params
// );
$exception->prepareForward(
    'view', 'category', 'catalog', array('id' => $id)
);
throw $exception;
```

REDIRECT IN PRACTICE

Bug in Mage_Core_Controller_Varien_Exception::prepareRedirect()

-> extend Exception
-> override Method

```php
public function prepareRedirect($path, $arguments = array())
{
    $this->_resultCallback = self::RESULT_REDIRECT;
    $this->_resultCallbackParams($path, $arguments); // ORIGINAL
    $this->_resultCallbackParams = array($path, $arguments);// CORRECT
    return $this;
}
```

REDIRECT IN PRACTICE - 2

```
$exception = new My_Extended_Exception_Because_Of_The_Bug();
// $exception->prepareRedirect($path, $arguments);
$exception->prepareRedirect(
    'catalog/category/view', array('id' => $id)
);
throw $exception;
```

NAME RESOLUTION
FOR BLOCKS, MODELS, HELPER

EXAMPLE OF A MODEL

```
$product = Mage::getModel('catalog/product');
```

getModel() 👀

MODEL

`Mage::getModel()`

```
public static function getModel($modelClass = '', $arguments = array())
{
    return self::getConfig()->getModelInstance($modelClass, $arguments);
}
```

getModelInstance() 👀

MODEL

Mage_Core_Model_Config::getModelInstance()

```
public function getModelInstance($modelClass='', $constructArguments=array())
{
    $className = $this->getModelClassName($modelClass);
    if (class_exists($className)) {
        $obj = new $className($constructArguments);
        return $obj;
    } else {
        return false;
    }
}
```

getModelClassName() 👀

Mage_Core_Model_Config::getModelClassName()

```php
public function getModelClassName($modelClass)
{
    $modelClass = trim($modelClass);
    if (strpos($modelClass, '/')===false) {
        return $modelClass;
    }
    return $this->getGroupedClassName('model', $modelClass);
}
```

getGroupedClassName() 👀

Mage_Core_Model_Config::getGroupedClassName() -1

```
public function getGroupedClassName($groupType, $classId, $groupRootNode=null)
{
    if (empty($groupRootNode)) {
        $groupRootNode = 'global/'.$groupType.'s';
    }

    $classArr = explode('/', trim($classId));
    $group = $classArr[0];
    $class = !empty($classArr[1]) ? $classArr[1] : null;

    if (isset($this->_classNameCache[$groupRootNode][$group][$class])) {
        return $this->_classNameCache[$groupRootNode][$group][$class];
    }

    $config = $this->_xml->global->{$groupType.'s'}->{$group};
}
```

getGroupedClassName() 👀

Mage_Core_Model_Config::getGroupedClassName() - 2

```
// First - check maybe the entity class was rewritten
$className = null;
if (isset($config->rewrite->$class)) {
    $className = (string)$config->rewrite->$class;
} else {
    if ($config->deprecatedNode) {
        $deprecatedNode = $config->deprecatedNode;
        $configOld = $this->_xml->global->{$groupType.'s'}->$deprecatedNode;
        if (isset($configOld->rewrite->$class)) {
            $className = (string) $configOld->rewrite->$class;
        }
    }
}
```

getGroupedClassName() 👀

Mage_Core_Model_Config::getGroupedClassName() - 3

```
// Second - if entity is not rewritten then use class prefix to form class name
if (empty($className)) {
    if (!empty($config)) {
        $className = $config->getClassName();
    }
    if (empty($className)) {
        $className = 'mage_'.$group.'_'.$groupType;
    }
    if (!empty($class)) {
        $className .= '_'.$class;
    }
    $className = uc_words($className);
}

$this->_classNameCache[$groupRootNode][$group][$class] = $className;
return $className;
```

BONUSTRACK

URBAN LEGEND

The CronJob triggers no ModulUpdates

```
// cron.php
Mage::app(); //doesn't call applyAllUpdates()
```

unlike here:

```
// index.php
Mage::run(); //calls applyAllUpdates()
```

HIDDENTRACK

WAT!?

Blocks in LayoutXML have a

NAME (*name="..."*)

and an **ALIAS** (*as="..."*)

fine

Unset a Block (cut out from the Layout)

<action method="_unsetChild_"> using Alias

fine too

Insert a Block (make it available in the Layout again)

<action method="_insert_"> using Name

WAT!?

NEWSLETTER MODULE

```
# …/core/Mage/Adminhtml/controllers/Newsletter/QueueController.php
# Line 173

// Todo: put it somewhere in config!
$countOfQueue  = 3;
$countOfSubscritions = 20;
```

WAT!?

We declare an Array

```
# …/core/Mage/Core/Controller/Varien/Exception.php
# Line 37

protected $_resultCallbackParams = array();
```

fine

Later the Array will be casted ... to a Method

```
$this->_resultCallbackParams($path, $arguments);
```

WAT!?

PRODUCT IMPORT – IMAGES

- $productData['_media_lable']
  - lable instead of label

WAT!?

PRODUCT IMPORT – LOGGING

```
# …/core/Mage/ImportExport/Model/Abstract.php
# Line 89

$dirPath = Mage::getBaseDir('var') . DS . Mage_ImportExport_Model_Scheduled_Operation::LOG_DIRECTORY
```

The File `Scheduled/Operation.php` is missing; looks like it never existed!

I CAN'T EVEN ...

ADD ATTRIBUTES TO GROUPS

function addAttributeToGroup()

```
# …/core/Mage/Eav/Model/Entity/Setup.php
# Line 1081

…

$data = array(
    'entity_type_id'         => $entityType,
    'attribute_set_id'       => $setId,
    'attribute_group_id'     => $groupId,
    'attribute_id'           => $attributeId,
#   $data['sort_order']      =  $sortOrder; // declared in Line 1128
);

…

$this->getConnection()->insert(…, $data);
```

ADD ATTRIBUTES TO GROUPS

function addAttributeToSet()

```
# …/core/Mage/Eav/Model/Entity/Setup.php
# Line 1032

…

$data = array(
    'entity_type_id'          => $entityTypeId,
    'attribute_set_id'        => $setId,
    'attribute_group_id'      => $groupId,
    'attribute_id'            => $attributeId,
    'sort_order'              => $this->getAttributeSortOrder(…),
);

…

$this->_conn->insert(…, $data);
```

OH YEAH ... I MISSED THAT!

my bad!

FACTORY METHODS

MODELS

Mage::getModel(); Mage::getResourceModel();

SINGLETONS

Mage::getSingleton(); Mage::getResourceSingleton()

FACTORY METHODS

HELPER

`Mage::helper();` `Mage::getResourceHelper();`

WAT!?

I MEAN DOUBLEWAT!?

DIFFERENT TAGS FOR IDENTICAL FUNCTIONS

observer uses <class> and <method>

```
<catalog_wysiwyg>
    <class>catalog/observer</class>
    <method>catalogCheckIsUsingStaticUrlsAllowed</method>
</catalog_wysiwyg>
```

DIFFERENT TAGS FOR IDENTICAL FUNCTION

cron run uses <model> with group/model::method

looks like static, but is not

```
<run>
    <model>catalog/observer::reindexProductPrices</model>
</run>
```

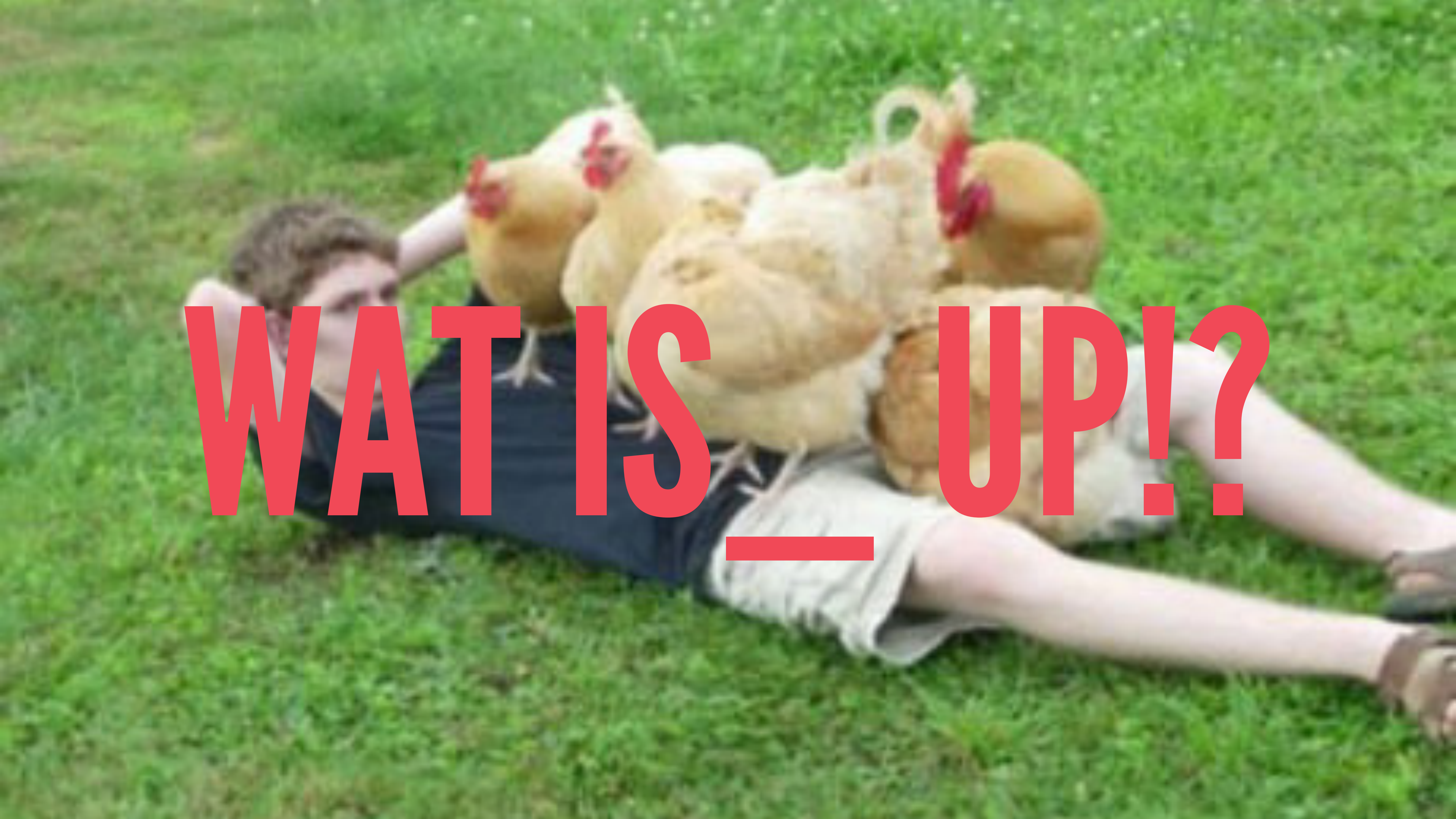
THAT'S ACTUALLY ...

WHAR THE HECK ARE THEY LOOKING AT???

ATTRIBUTES

CREATE AND MODIFY

addAtribute uses visible = ...
updateAttribute uses is_visible = ...



WAT IS _ UP!?

A STORE IS A STORE IS A STORE IS A STORE

in AdminPanel StoreView is called Store in PHP
in AdminPanel Store is called StoreGroup in PHP

STORE !== STORE !?!??!

WWWWT!?

wat
wat
wat

SOURCES

- forward vs redirect
- http://www.giantgeek.com/blog/?p=109
- http://www.javapractices.com/topic/TopicAction.do?Id=181
- Collection filtering (in German)
- http://www.webguys.de/magento/turchen-11-eine-collection-filtern/

K THX BAI