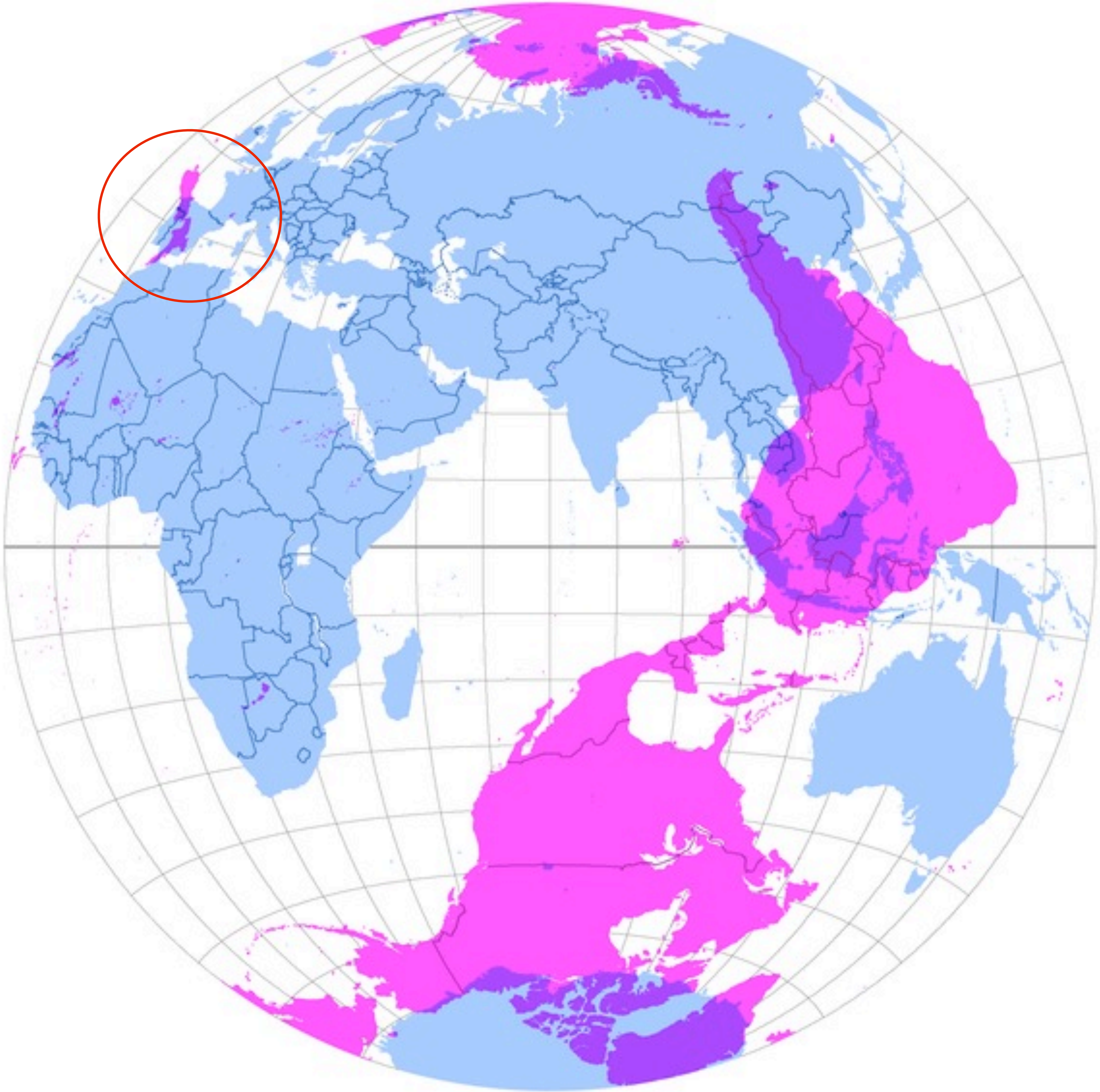
Magento Customization With Extensions

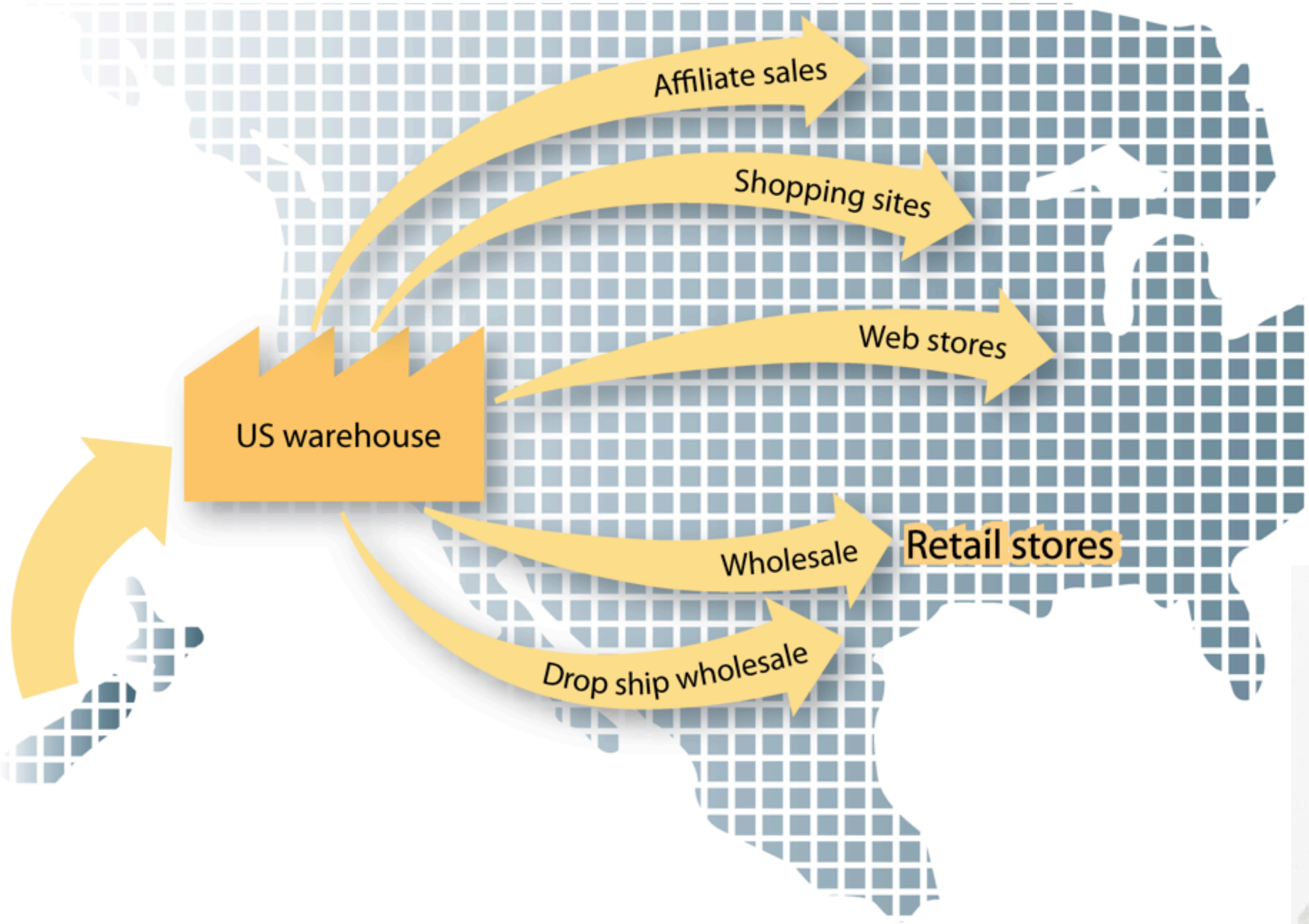
Ashley Schroder

aschroder.com

ashley.schroder@gmail.com



World Wide Access



Affiliate sales

Shopping sites

Web stores

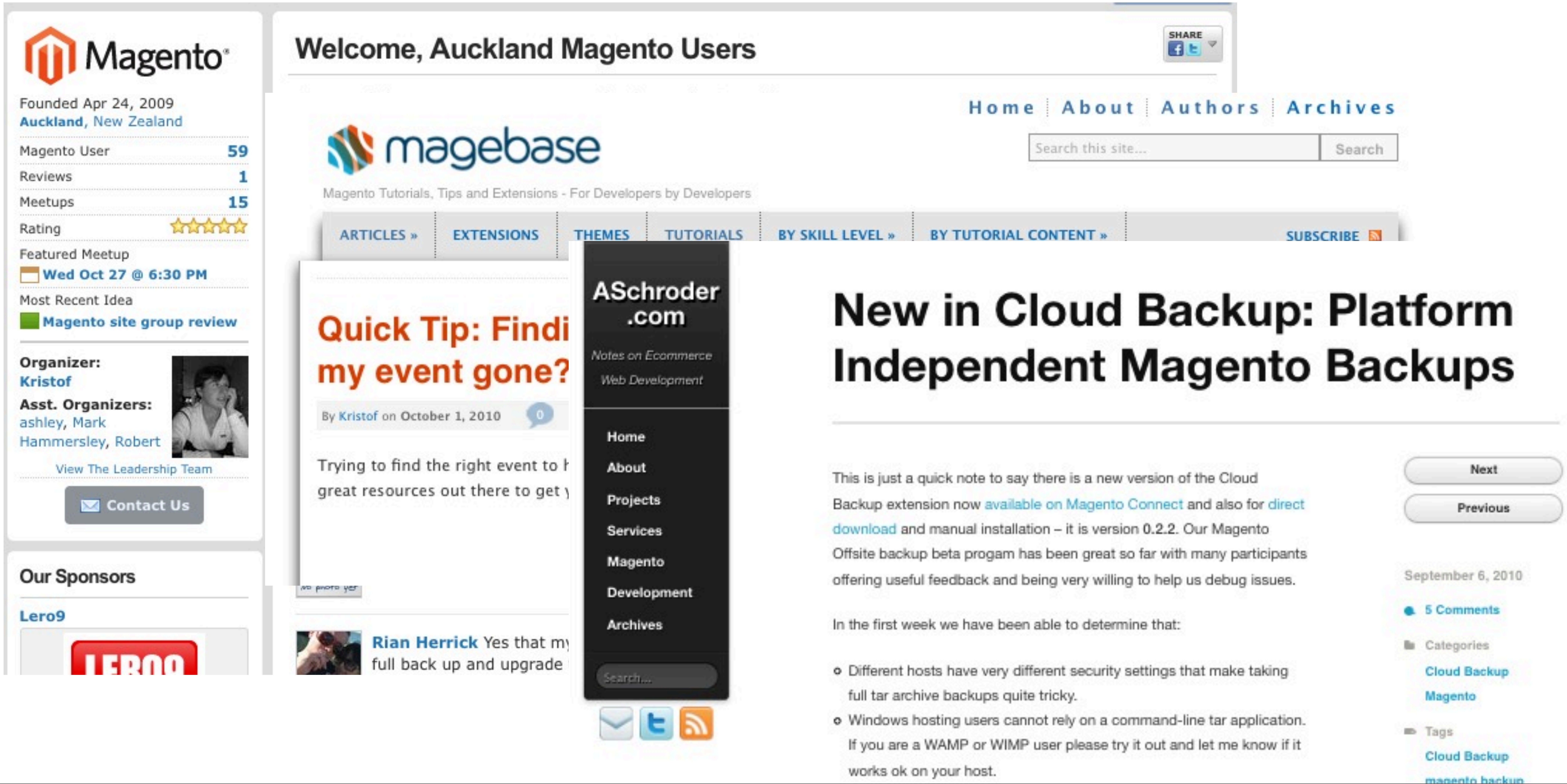
US warehouse

Wholesale

Retail stores

Drop ship wholesale

Magento In New Zealand



Magento

Founded Apr 24, 2009
Auckland, New Zealand

Magento User	59
Reviews	1
Meetups	15
Rating	★★★★★

Featured Meetup
Wed Oct 27 @ 6:30 PM

Most Recent Idea
🟩 Magento site group review

Organizer:
Kristof

Asst. Organizers:
ashley, Mark Hammersley, Robert

View The Leadership Team

Contact Us

Our Sponsors

Lero9

Welcome, Auckland Magento Users

magebase

Magento Tutorials, Tips and Extensions - For Developers by Developers

Quick Tip: Findi
my event gone?

By Kristof on October 1, 2010

Trying to find the right event to h
great resources out there to get y

Rian Herrick Yes that my
full back up and upgrade

ASchroder.com

Notes on Ecommerce
Web Development

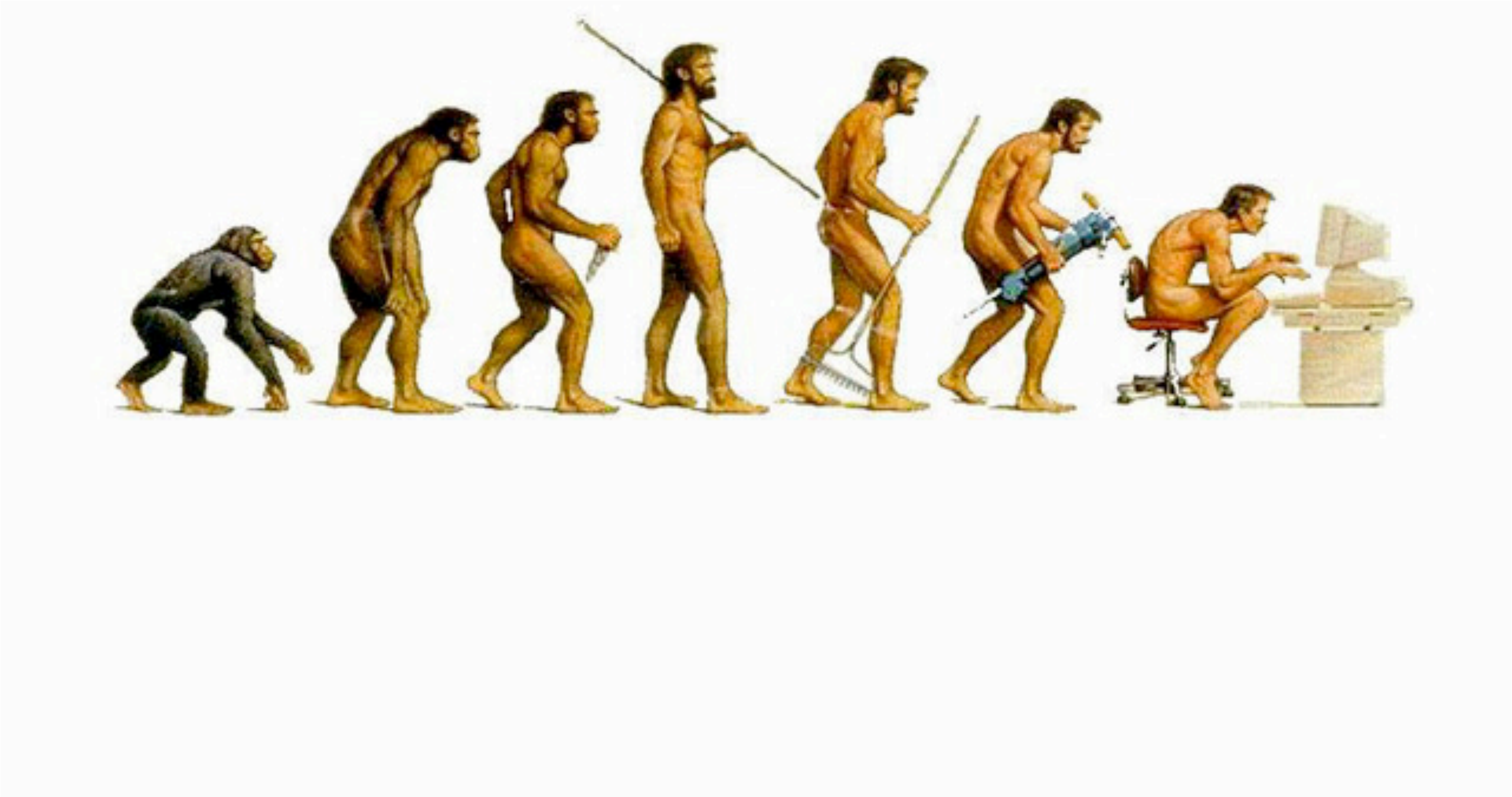
New in Cloud Backup: Platform Independent Magento Backups

This is just a quick note to say there is a new version of the Cloud Backup extension now available on Magento Connect and also for direct download and manual installation – it is version 0.2.2. Our Magento Offsite backup beta progam has been great so far with many participants offering useful feedback and being very willing to help us debug issues.

In the first week we have been able to determine that:

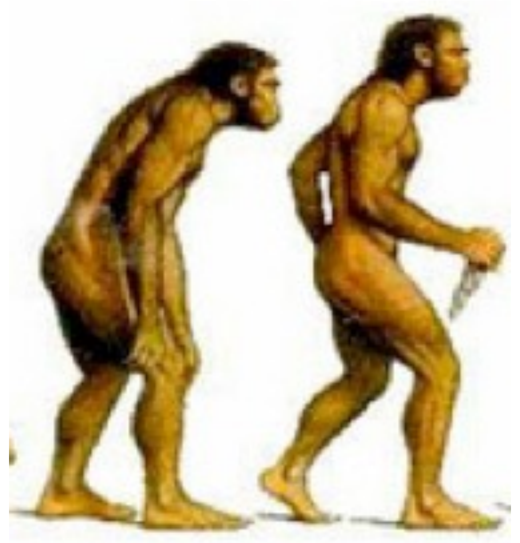
- Different hosts have very different security settings that make taking full tar archive backups quite tricky.
- Windows hosting users cannot rely on a command-line tar application. If you are a WAMP or WIMP user please try it out and let me know if it works ok on your host.

Evolving Magento developers

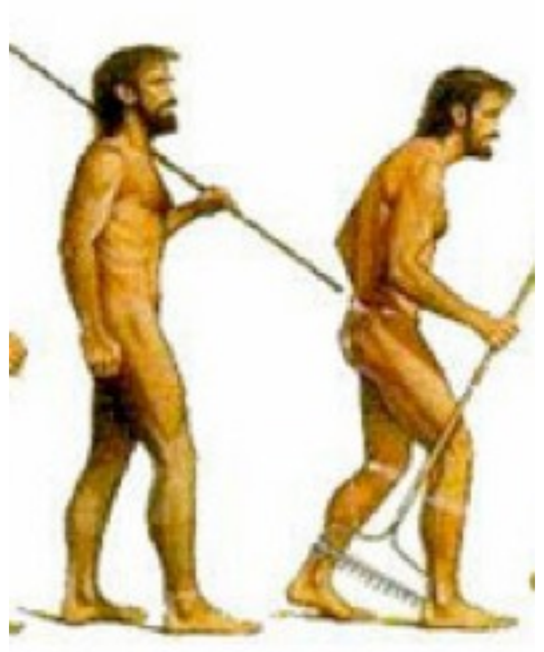


Starting with Magento

Difficult to understand, hard to make changes



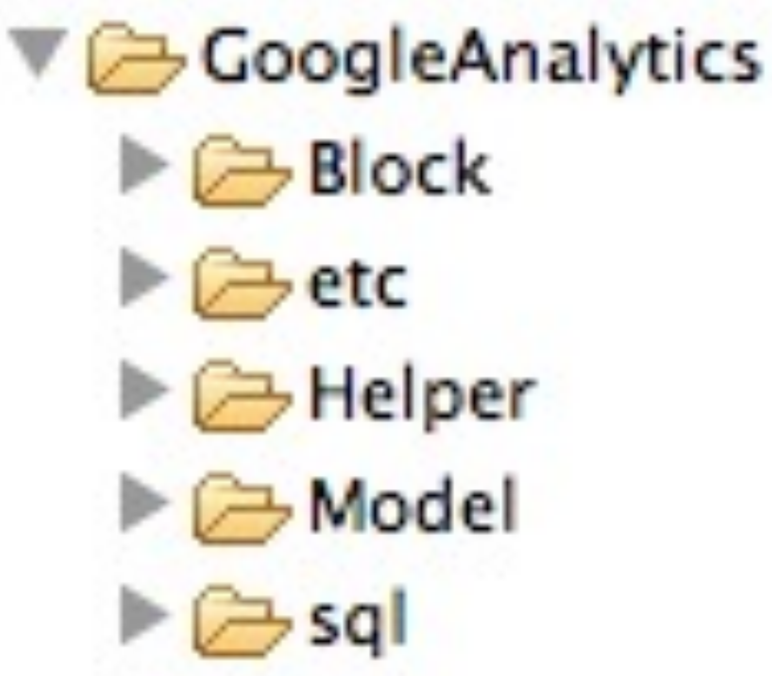
Things take a long time, quick core hacks



Using `app/code/local` overrides

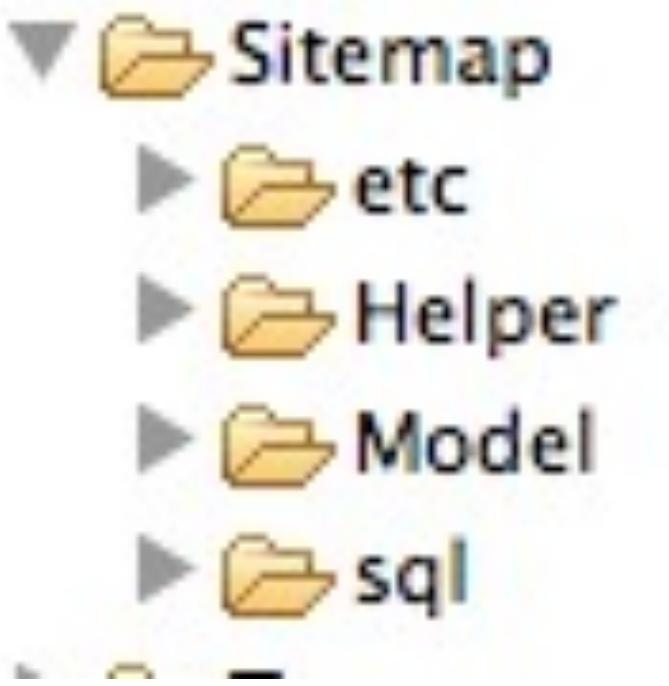
Using Extensions

Google Analytics



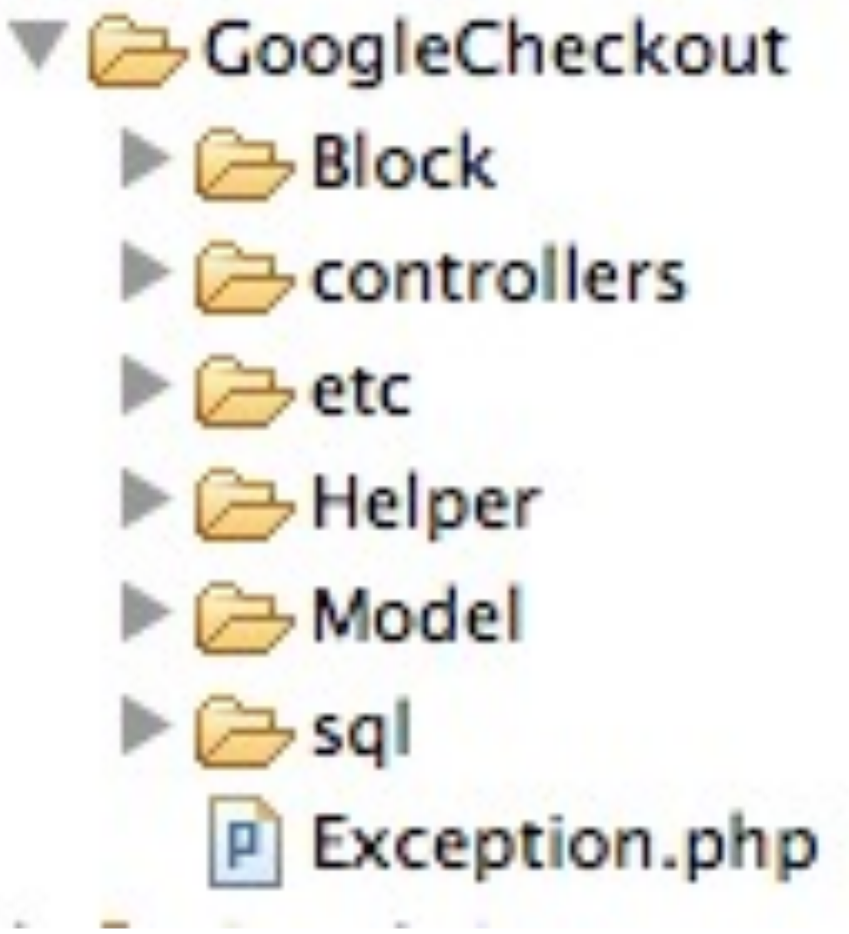
- GoogleAnalytics
  - Block
  - etc
  - Helper
  - Model
  - sql

Sitemap



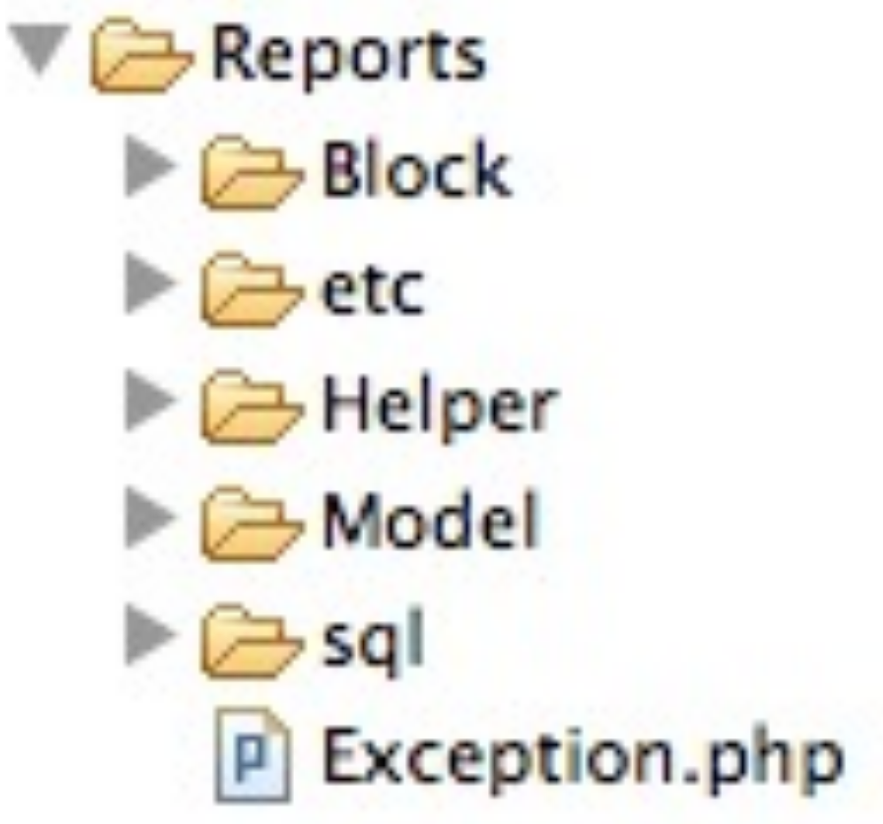
- Sitemap
  - etc
  - Helper
  - Model
  - sql

Google Checkout



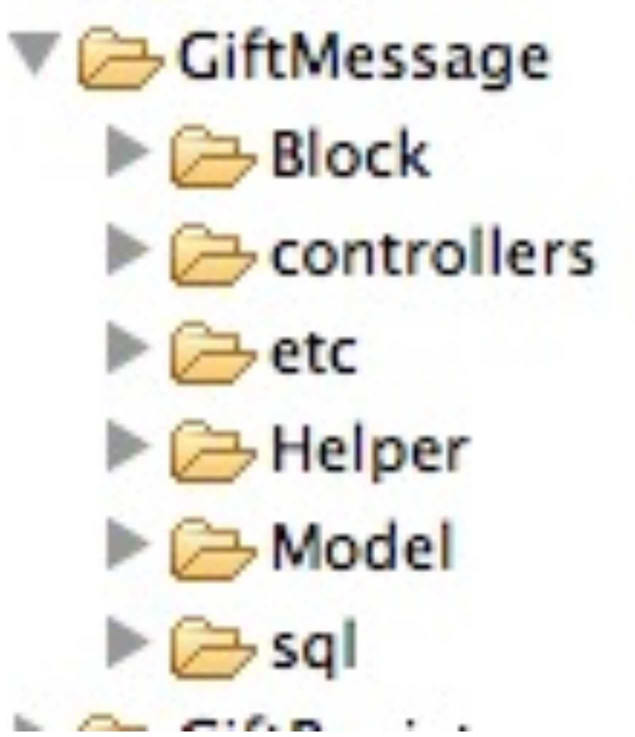
- GoogleCheckout
  - Block
  - controllers
  - etc
  - Helper
  - Model
  - sql
  - Exception.php

Reports



- Reports
  - Block
  - etc
  - Helper
  - Model
  - sql
  - Exception.php

Gift Message



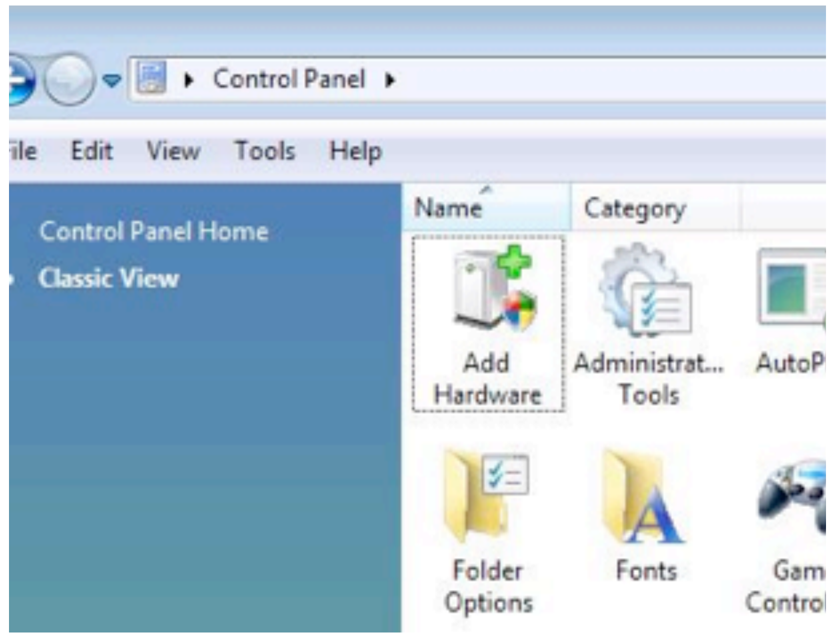
- GiftMessage
  - Block
  - controllers
  - etc
  - Helper
  - Model
  - sql

Customization Pain Points

- Upgrade heartache
- Related code gets spread out and duplicated
- Difficult to *engineer*

Why Extensions?

- Extensions are self-contained blocks of functionality, they can go places



- You can add configurability to your extensions

- You can *engineer* your extensions

Extension Do Cool Things

- Scheduled processes using Cron
- Schema changes using SQL setup scripts
- Hook into events for pinpoint customizing

Processing like clockwork

- Works like the Unix cron daemon, but let’s you work purely in Magento
- Use it for scheduling tasks in your extensions
- Warehouse inventory and Fulfillment tracking

Scheduled Processing

```
<crontab>
     <jobs>
        <fbashipping_inventorysynch>
            <schedule><cron_expr>*/5 * * * *</cron_expr></schedule>
            <run><model>fbashipping/inventory::cron</model></run>
        </fbashipping_inventorysynch>
        <fbashipping_tracking>
            <schedule><cron_expr>0 */6 * * *</cron_expr></schedule>
            <run><model>fbashipping/tracking::cron</model></run>
        </fbashipping_tracking>
     </jobs>
  </crontab>
```

Processing like clockwork

```
public function cron() {

  // We need to load the Adminhtml events
  Mage::app()->loadAreaPart('adminhtml', Mage_Core_Model_App_Area::PART_EVENTS);

  // Only actually do anything if tracking is enabled
  if (Mage::helper('fbashipping')->isTrackingEnabled()) {
    $this->submitRequest();
  }
}
```

BUT: Don't forget to allow cron functionality to be disabled, and make sure you enable admin events if your cron job is doing admin things.

Cron Reliability

- It’s not *always* running!

```
$schedules = Mage::getModel('cron/schedule')->getCollection()
                    ->addFieldToFilter('status', Mage_Cron_Model_Schedule::STATUS_PENDING)
                    ->load();
```

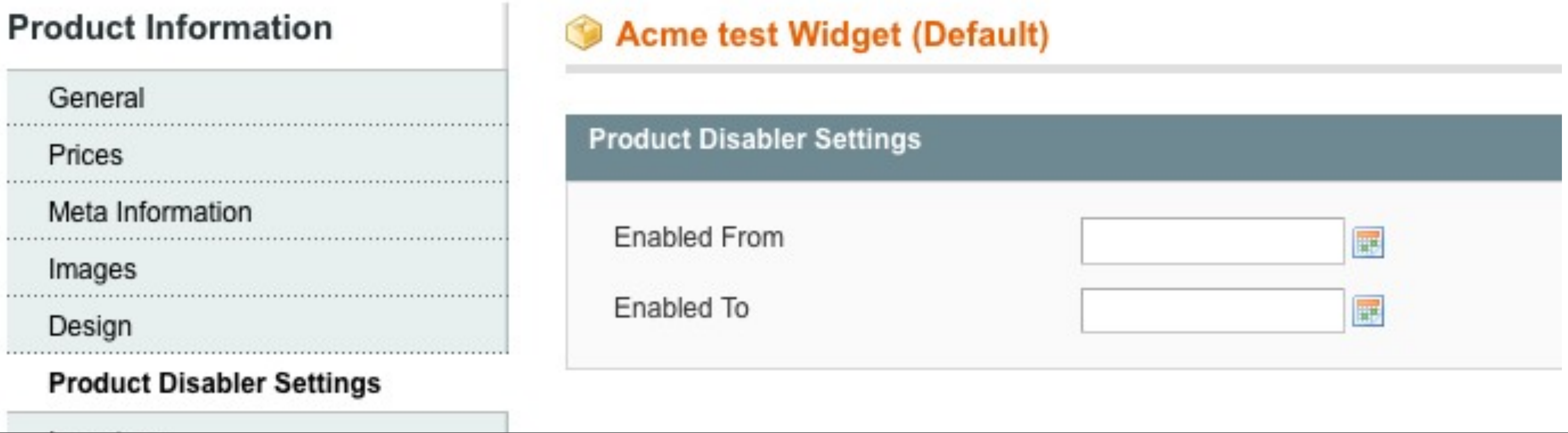
Use remote pinging tools as an easy setup, e.g webstore monitors

Schema Changes and EAV

Add/Edit attributes for core entities (thanks EAV).

```
$installer->startSetup();

$installer->addAttribute('catalog_product', 'disabled_from', array(
        'type'              => 'int',
      // snip...
        'apply_to'          => '',
        'is_configurable'   => false
    ));
```



Schema and EAV

- You can manipulate/create your own tables and entities.

Create Tables

```
$installer->run("
CREATE TABLE `{$this->getTable('smtppro_email_log')}` (
  `email_id` int(10) unsigned NOT NULL auto_increment,
  `log_at` timestamp NOT NULL default CURRENT_TIMESTAMP on update
CURRENT_TIMESTAMP,
  `to` varchar(255) NOT NULL default '',
  `template` varchar(255) NULL,
  `subject` varchar(255) NULL,
  `email_body` text,
  PRIMARY KEY  (`email_id`),
  KEY `log_at` (`log_at`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
");
```

Use them

```
$log = Mage::getModel('smtppro/email_log')
          ->setTo($to)
          ->setTemplate($template)
          ->setSubject($subject)
          ->setEmailBody(
            $isHtml?$email:nl2br($email))
          ->save();
```

Schema and EAV Gotchas

- Backward Compatibility
- Each script only runs once, logged to core_resource
- Keep an eye on your config.xml version number

Events

Explicit Events

```
 Mage::dispatchEvent
 ('admin_session_user_login_success',
             array('user'=>$user));
```

Implicit Events

```
protected function _beforeSave()
    {
        Mage::dispatchEvent('model_save_before', array('object'=>$this));
        Mage::dispatchEvent($this->_eventPrefix.'_save_before', array
($this->_eventObject=>$this));
        return $this;
    }
```

Events

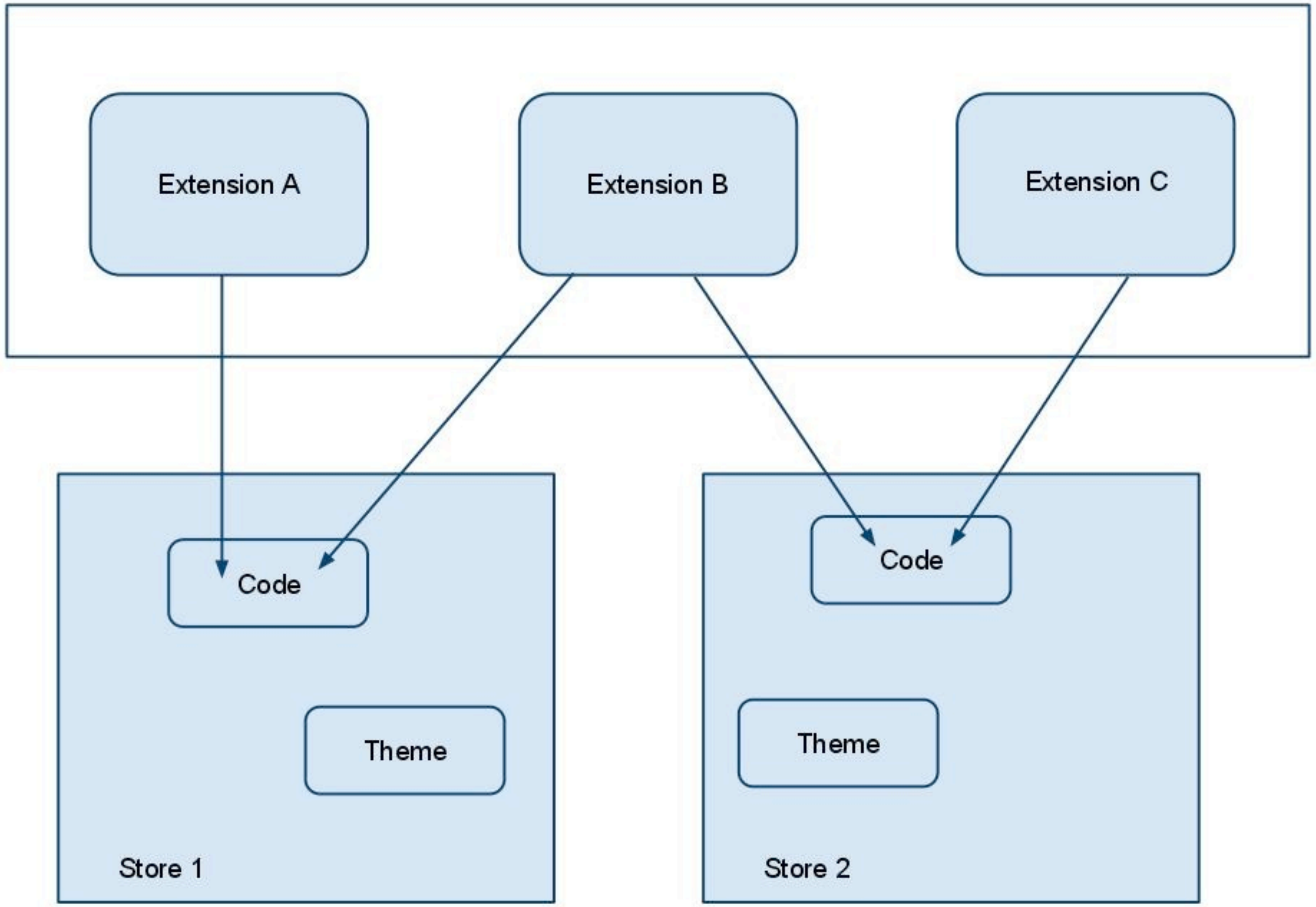
- Catch events using configuration in config.xml

```
<events>
  <sales_order_shipment_save_after>
    <observers>
      <customerservice_after_ship>
        <type>singleton</type>
        <class>customerservice/customer_observer</class>
        <method>shipment</method>
      </customerservice_after_ship>
    </observers>
  </sales_order_shipment_save_after>
</events>
```

Events

- You can fire events too e.g SMTP Pro email event
  - `Mage::dispatchEvent('smtppro_email_after_send'...`
- Allow customization without class overrides
- Help resolve extension conflicts - *we'll get to this soon*

Engineering Extensions



Version control & Releases

- Place extensions into a code repository
  - then either export them (or svn:externals) them into other projects
- Symlinks allow you to have a single extension installed in multiple stores.

Symlinking with modman

- modman is a great script made by Colin Mollenhour
- Create all symlinks with a recipe.
- All stores can use the same recipe

Dev and Prod versions

- Maintain a production version and a development version of each extension
- Push changes to production only when you are happy with the stability of development
- Test on multiple versions of Magento
- Update all stores running at the same time

Limitations of extensions

- Effort - takes longer to create
- Certain core class changes eg superclasses
- **Clashes**! Let’s look at this one in more detail...

Extension clashes

- How do they happen? override same class
- Bane of any extension developers life
- Real problem for Magento in general - with growth of extensions

What to do about clashes

- Detect them so the users know (self test)
- Publish a list of known clashes that users can search and see - before they install.

One way to resolve clashes

Create your own events fired in the override class

```
<events>
  <wwa_autologistics_register_order_button>
    <observers>
      <fbashipping_order_button_observer>
        <type>singleton</type>
        <class>fbashipping/observer</class>
        <method>get_order_button</method>
      </fbashipping_order_button_observer>
    </observers>
  </wwa_autologistics_register_order_button>
</events>
```

Then fire your event...

All of the extensions that need to add a button can listen

```
   //Fire event+add buttons, if we can ship, and we are enabled
   if ($this->_isAllowedAction('ship') &&
       $this->getOrder()->canShip() &&
       Mage::helper('autologistics')->isEnabled()) {

   Mage::dispatchEvent('wwa_autologistics_register_order_button',
     array('orderId' => $orderId,
         'block' => $this));

}
```

What do we do with all of these extensions...

So...

ASchroder.com

- Ashley Schroder
- @aschroder
- ashley.schroder@gmail.com
- visit New Zealand