



# ISSET INTERNET PROFESSIONALS

How to approach CI projects which  
go beyond the 'homepage'



## Who we are

- Jeroen van der Gulik
  - Lead Developer Isset
  - [jeroen@isset.nl](mailto:jeroen@isset.nl)
- Rommert van Til
  - Project Manager
  - [rommert@isset.nl](mailto:rommert@isset.nl)



## What we do

- Isset Internet Professionals targets companies with an ambition on the internet.
- We provide solutions for problems related to the web using internet technology.
- Isset only works with professionals who are up to date on current internet problems as well as solutions.
- We are purely functional and technical.



## What we are currently doing

- Building custom web applications / websites.
- Building connectors/ middleware applications.
- Optimizing and monitoring SEO performance on a technical level (not ga).
- Migrating a large custom build webshop.
- We are currently developing an in house application that will be make sense of SEO / SEA.



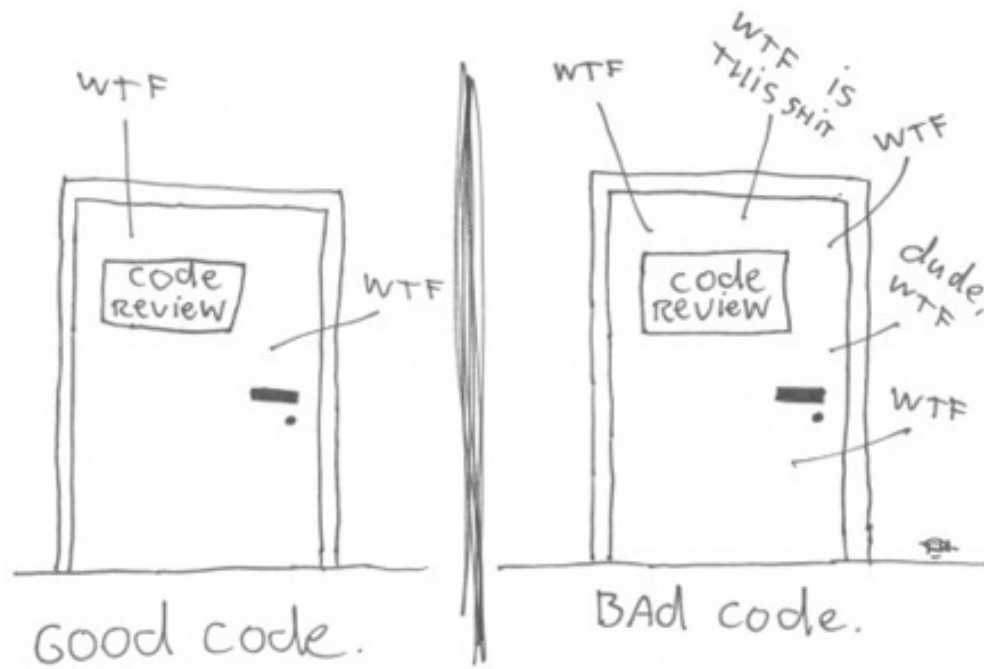


# Why we chose CodeIgniter

Avoiding the DreamWeaver effect.

# The Dream Weaver Effect

The ONLY VALID MEASUREMENT  
OF CODE QUALITY: WTFs/MINUTE



(c) 2008 Focus Shift



## General pros

- Lightweight
- Fast (!)
- Open Source
- Very good documentation
- Very good User Community
- Simple (KISS)
- MVC



## What we think is important

- All we need is routing + database abstraction.
- Easily extendible/ adjustable.
- Simple Caching mechanism.
- Has few 'conventions'.
- Has many 'add-ons'.
- Easy to add external classes/ libraries



## What we discovered

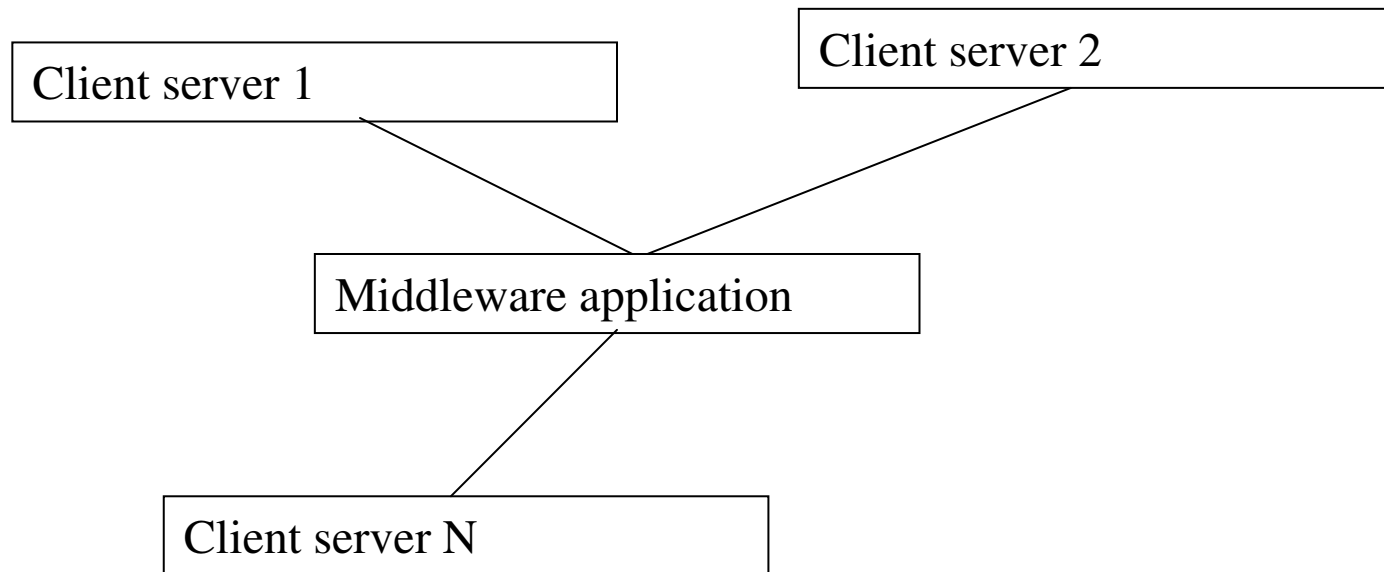
- Application framework, not a website framework.
- Forces a team to work in the same structure.
- CI is fast so you can write ‘sloppy’ code.
- CI plays nice (mostly) with Zend which is a good thing
- CodeIgniter is relaxed.





**What we will talk about**

# Why not Expression Engine?

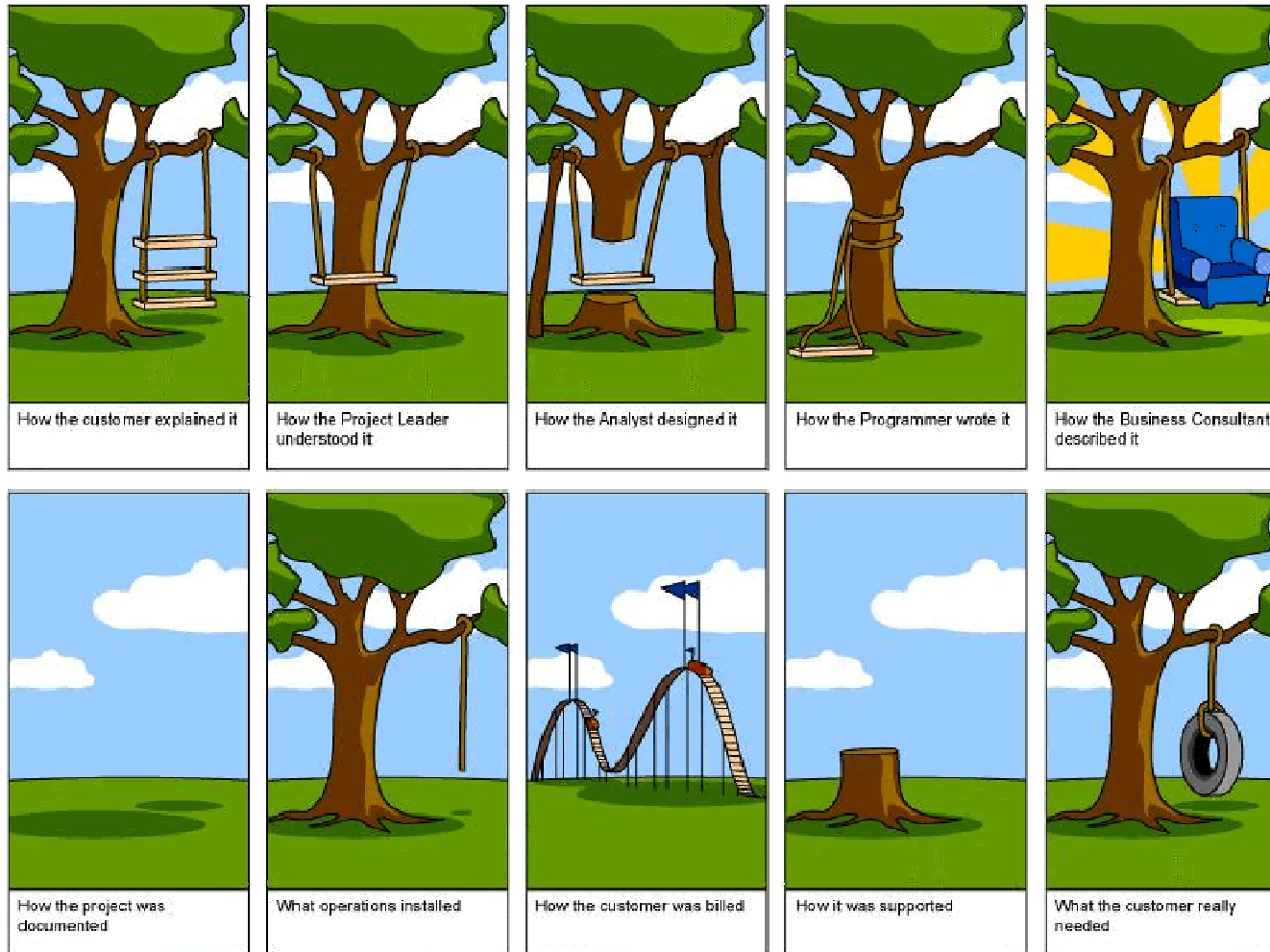


# Topics

- Making what the client needs
- Database Driven Development (DDD).
- How we design our projects
- Project file organization
- Source control management
- Extending Classes (Base Classes)
- DB routing
- Layout/ Templating
- ICF (Isset Content Framework)
- Logging
- Security (CLRF, AR)
- Making sense of SEO / SEM



# The Client



## Making the Client happy

- Force the client to participate.
- Three tier communication : developer(s), project manager, client(s)
- Make a 'design' document.
- Make sure both the client and the developer can understand the document.
- Use the project manager as mediator.
- Use feature list to determine if all requirements are made.
- Don't use vague terminology.
- Prevent scope creep.



# Database Driven Development

- Think about the data structure first.
- Generate template CRUD based on DB scheme





THIS CAT

IS GOING TO FUCK SOMEBODY UP

## How we design our projects

- Making a test environment
- New project
- Migration
- Middleware / connector / webservice



## Make a test environment

- Make sure that it matches the production environment.
- Replace all e-mail addresses and cell phone numbers with traceable test data.
- Don't use offensive test data for fun, just don't use it.



## New Project

- Prerequisite : no database, no legacy system.
- Database Driven Development.
- Plan meeting(s) to write the design document with the client.
- Use agile approach (small iterations so the Client can see progress and test).



# Migration

- Prerequisite: any project that has a database or legacy functionality.
- Should we migrate to CodeIgniter?
- What is the database scheme like? (DDD)
- What is the scope of the legacy functionality?
- How maintainable is the legacy code?
- Are there 3rd party components?
- What is the platform?
- Bottom line: start from scratch or migrate functionality over time.



## Migration Path

- Make legacy application work next to CodeIgniter.
- Be aware of sessions (authentication).
- Be aware of absolute paths.
- Be aware of User Generated Content.
- Avoid changing existing code; move functionality into CodeIgniter.



## Middleware / connector / webservice.

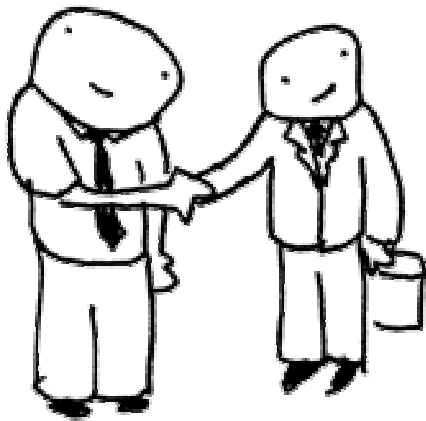
- Prerequisite; the application has to communicate with other applications.
- Demand (api) documentation up front.
- Log every transaction (request and response).
- Avoid connecting to third party databases directly.
- Clearly outline the responsibilities. (next slide)



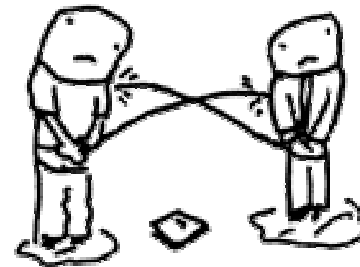
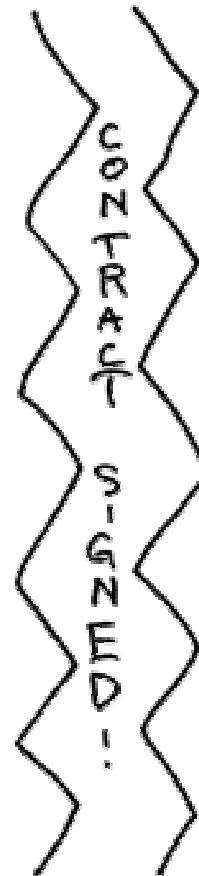
# Taking care of responsibilities

## HOW BUSINESS WORKS:

First, representatives from each business meet, discuss mutual interests, and forge an agreement.



They draft a contract...



Then they try to piss all over each other without getting the contract wet.



# Project file organization

- Depends on scale and environment
- Learn about symlinks!
- Separate system from site root if possible
- Separate application from site root if possible
- Separate public (or assets)
- Use what works for you



# Source control management

- Just do it.
- Git/ SVN/ Mercurial/ Bazar: Use what works for you
- We chose SVN because of general acceptance
- Use sensible commit messages



## Extending Classes (base classes)

- Base Controllers will make your life easier
- For details: **Jamie Rumbelow**
- Use MY\_Controller
  - Use different base controllers according to the purpose of the (sub)application
  - Alternatively use modules



# MY\_Controller

```
1  <?php  if ( ! defined('BASEPATH'))
2          exit('No direct script access allowed');
3  /**
4   * Description of MY_Controller.php
5   *
6   * @author jeroen
7   */
8  class MY_Controller extends Controller {
9
10     function MY_Controller()
11     {
12         parent::Controller();
13     }
14 }
15
16 require_once(APPPATH .'libraries/Frontend_Controller'.EXT);
17 require_once(APPPATH .'libraries/Backend_Controller'.EXT);
18
19 /* End of file MY_Controller.php */
20 /* Location: ./system/application/libraries/MY_Controller.php */
```



```

7 class Frontend_Controller extends Controller {
8
9     private $title           = 'My Awesome Project';
10    private $keywords        = 'Awesome Project';
11    private $description      = 'This is an Awesome Project (TM)';
12    private $template        = 'template_view';
13    protected $lang_id;
14
15    function Frontend_Controller()
16    {
17        parent::Controller();
18        if ( ! $this->lang->lang() )
19        {
20            $this->guess_language();
21        }
22        else
23        {
24            $this->lang_id = $this->lang->lang();
25        }
26    }
27
28    protected function set_template($view) {}
29
30    protected function render($childdata = NULL)
31    {
32        if( ! empty($childdata) )
33        {
34            $data = array_merge($this->data, $childdata);
35        }
36        else
37        {
38            $data = $this->data;
39        }
40
41        $this->load->view($this->template, $data);
42    }
43

```



```
7 class Backend_Controller extends Controller {
8
9     protected $template = 'admin/template_view';
10
11     function Backend_Controller()
12     {
13         parent::Controller();
14         $this->load->library('auth');
15         if ($this->auth->is_loggedin() === FALSE)
16         {
17             redirect(site_url('admin/login'));
18         }
19     }
20
21     protected function render($childdata = NULL)
22     {
23         if(! empty($childdata))
24         {
25             $data = array_merge($this->data, $childdata);
26         }
27         else
28         {
29             $data = $this->data;
30         }
31
32         $this->load->view($this->template, $data);
33     }
34 }
```



## DB Routing / MY\_Router

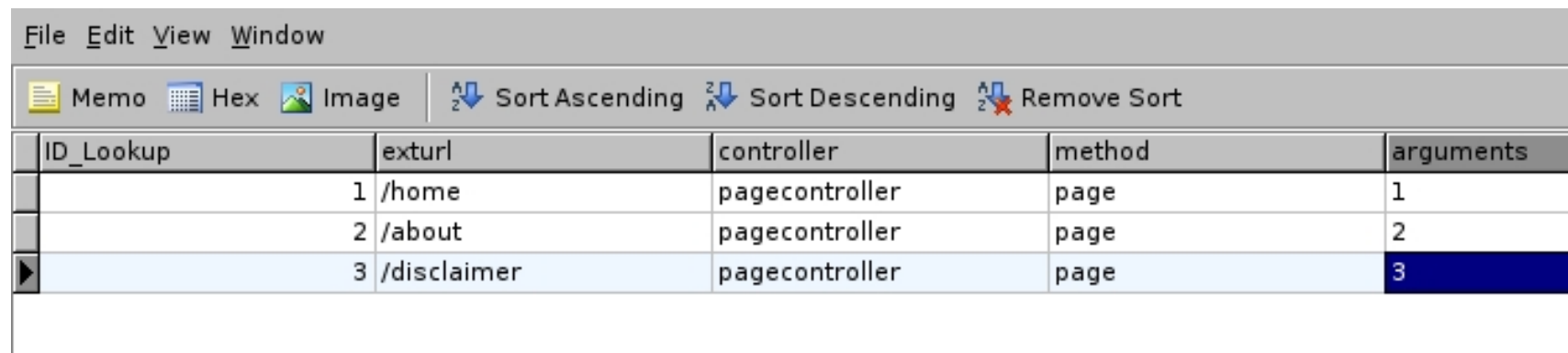
- How to match database records with controllers that do not exist.
- If a controller exist, the controller will be executed. Else it will do a db lookup.
- Make a table that has the external url eg <http://mysite.tld/home> with an internal controller e.g. `pagecontroller/page/1`
- Make exceptions for things like news, blog items etc. in the router file
- For heavy traffic websites, make a pre-system cache hook.



```
98 require(APP_PATH.'config/database'.EXT);
99
100 $mysqli = new mysqli(
101     $db['default']['hostname'],
102     $db['default']['username'],
103     $db['default']['password'],
104     $db['default']['database']);
105
106 if (mysqli_connect_errno())
107 {
108     log_message('debug', "Could not connect to Database."
109         . mysqli_connect_error());
110     show_404();
111 }
112
113 global $URI;
114
115 $sql = "SELECT controller, method, arguments
116     FROM tbllookup WHERE strExtId = ? LIMIT 1";
117 $stmt = $mysqli->prepare($sql);
118 $stmt->bind_param('s', $slug);
119 $slug = $URI->uri_string;
120 $stmt->execute();
121 $stmt->bind_result($controller, $method, $arguments);
122 if($stmt->fetch())
123 {
124     $segments = array($controller, $method, $arguments);
125     return($segments);
126 }
127 show_404($segments[0]);
```



# Example Lookup Table



The screenshot shows a web application interface with a menu bar (File, Edit, View, Window) and a toolbar containing icons for Memo, Hex, Image, Sort Ascending, Sort Descending, and Remove Sort. Below the toolbar is a table with the following data:

ID_Lookup	exturl	controller	method	arguments
1	/home	pagecontroller	page	1
2	/about	pagecontroller	page	2
3	/disclaimer	pagecontroller	page	3



```
3 class Pagecontroller extends Frontend_Controller {
4
5     function Pagecontroller()
6     {
7         parent::Frontend_Controller();
8     }
9
10    function page($page_id = NULL)
11    {
12        if (is_null($page_id) OR ! is_natural($page_id))
13        {
14            $page_id = 1;
15        }
16
17        $this->load->model('page_model','page');
18        $partial = $this->page->get_content($page_id);
19        $data['content'] = $this->load->view('partial/content_view',$partial,TRUE);
20        $this->render($data);
21    }
22 }
23 /* End of file page.php */
24 /* Location: ./system/application/controllers/pagecontroller.php */
```



# Exceptions

```
43 $route['default_controller'] = "page/pageproperties/1";
44 $route['(\w{2})/nieuws/(:any)'] = "nieuws/news_detail/$2";
45 $route['(\w{2})/news/(:any)'] = "nieuws/news_detail/$2";
46 $route[$url[1].'/news'] = "nieuws";
47 $route['scaffolding_trigger'] = "";
48 $route['admin'] = "admin/adminhome";
49
50 // URI like '/en/about' -> use controller 'about'
51 $route['^nl/(.+)$'] = "$1";
52 $route['^en/(.+)$'] = "$1";
53
54 // '/en' and '/fr' URIs -> use default controller
55 $route['^nl$'] = $route['default_controller'];
56 $route['^en$'] = "page/pageproperties/18";
57
```

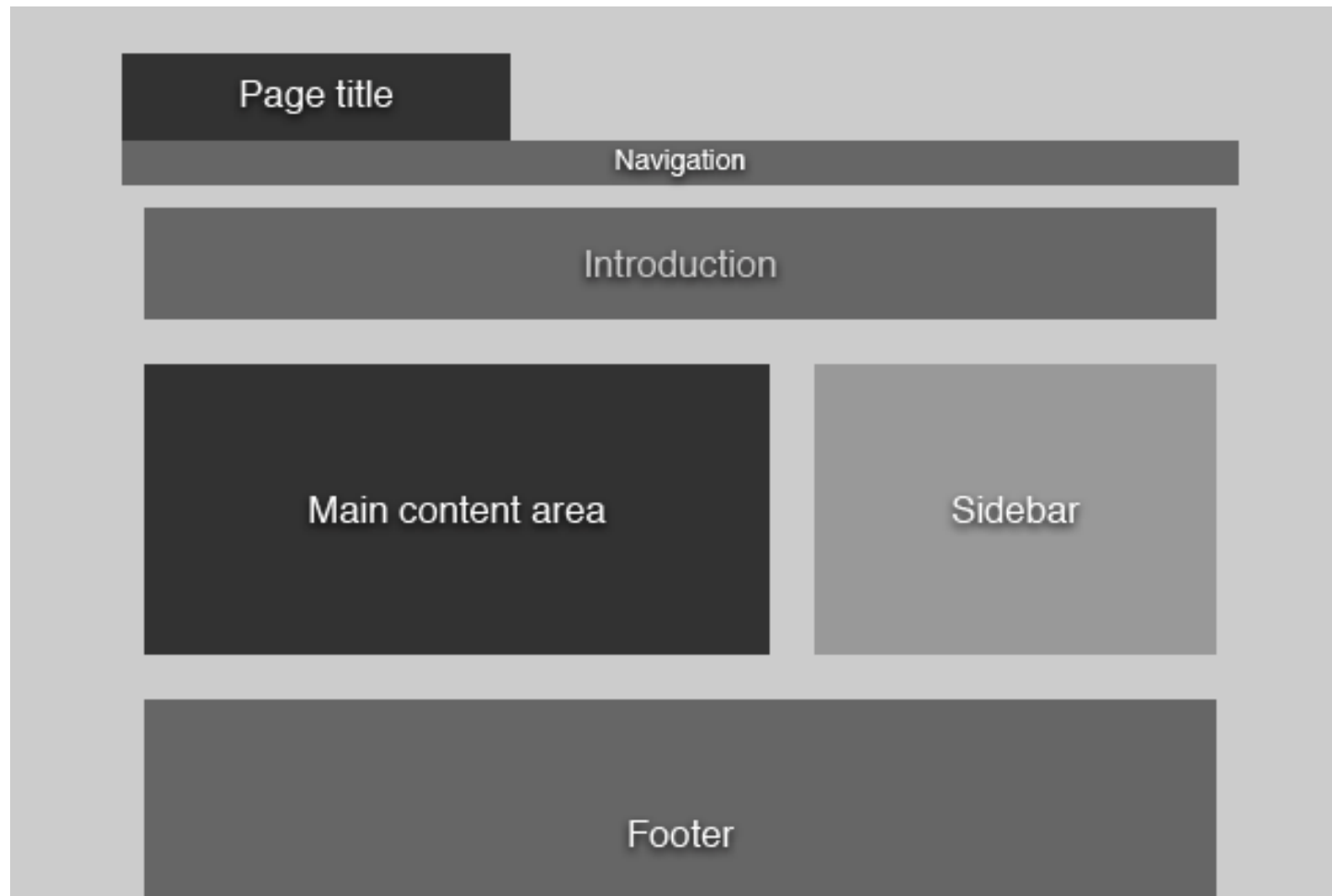


# Tackling the layout / template issue

- Default template
  - Overwrite when needed with variations
- Little template partials like navigation, sidebars etc.
- Every partial can be associated with model(s) and/or view(s).
- INCMS uses db records to define the relation between pages and partials.
- Complete separation of logic en layout (mvc)
  - Making a new layout for a site is easy.



# Template sections



## Issues with 'standard' solutions

- 80% was easy (click 'n go)
- 15% was hard (modify base code/ write workarounds or custom modules/plugins)
- 5% was not doable (restrictions by standard solution)
- Standard solutions that did work were either expensive or far too complex for the end user
- We hope that Ellis Labs will provide the 'ultimate' solution with Expression Engine 2.0

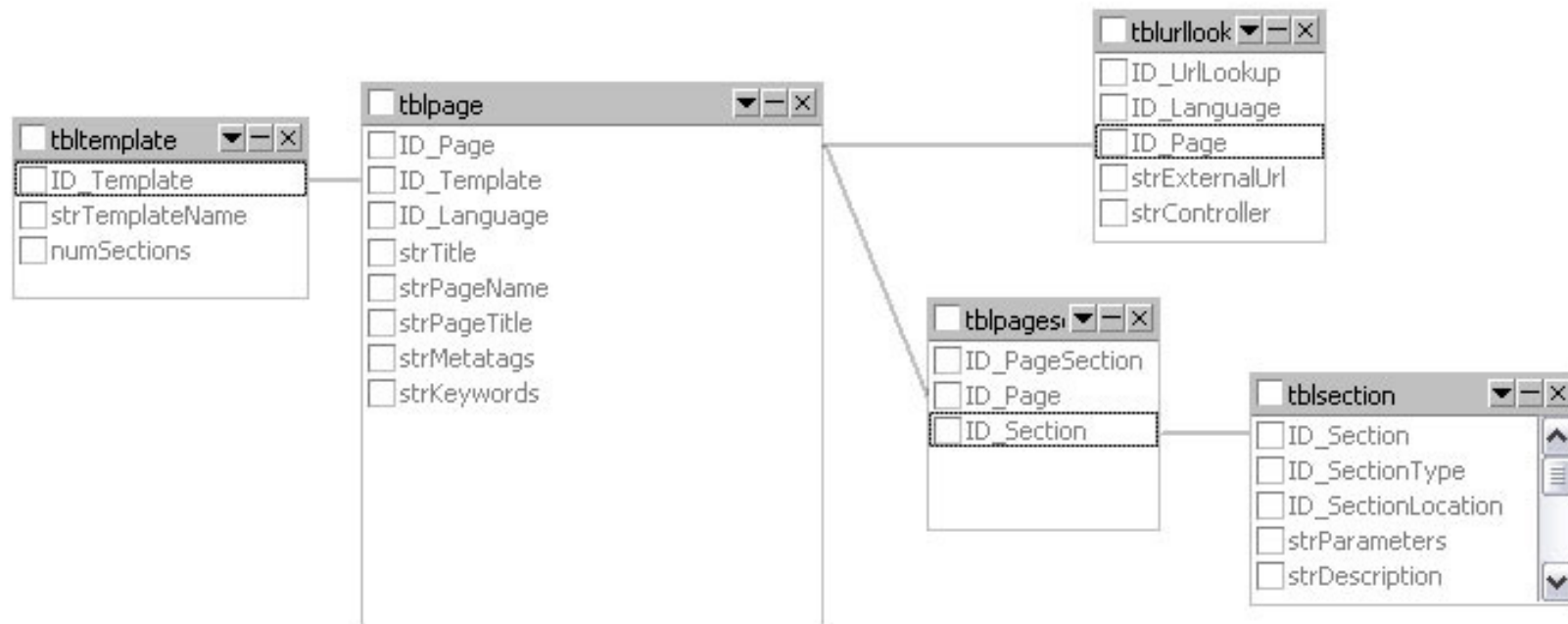


## ICF (Isset Content Framework)

- NOT a CMS
- Generating editable **content** sections.
- Using default template most of the time, specific templates when necessary.
- Best of both worlds: making basic content pages easy, making complex pages doable. (using standard CI framework).
- 80% of a website is just content.
- 20% of a website is custom made. No restrictions!
- Think custom ajaxified ‘shiny’ contact forms clients love to torture developers with.



# ICF database scheme



# Logging

- Especially important when making middleware.
- Helps when playing ‘the blame game’.
- Log incoming, outgoing requests.
- Log email / sms / whatever.
- Logging is cheap but provides valuable debug information.
- Logging 404 can help determine routing issues, SEO issues, hacking attempts.
- Logging executed queries can help find bottlenecks.
- Logging to a DB makes analyzing data easy.



## Example E-Mail Logging

- 'Build' the email to send
- Save it to a database
- Pass it to an email queue / cronjob
- Update the record when send
- Log if any errors occur



# Example Email Database Scheme

File Edit Window

New Save Save As Add Field Insert Field Delete Field Primary Key

Fields Indexes Foreign Keys Triggers Options Comment

Name	Type	Length	Decimals	Allow Null	
ID_Email	int	11	0	<input type="checkbox"/>	
blnSend	tinyint	1	0	<input type="checkbox"/>	
blnError	tinyint	1	0	<input checked="" type="checkbox"/>	
msgError	varchar	255	0	<input checked="" type="checkbox"/>	
from	varchar	100	0	<input type="checkbox"/>	
to	varchar	100	0	<input type="checkbox"/>	
subject	varchar	255	0	<input type="checkbox"/>	
message	text	0	0	<input type="checkbox"/>	
datSend	datetime	0	0	<input checked="" type="checkbox"/>	
datPlanned	datetime	0	0	<input checked="" type="checkbox"/>	

Default:

Comment:

Auto Increment

Unsigned

Zerofill

Number of Field: 10



```

class Emailqueue_model extends Model {

    function Emailqueue_model()
    {
        parent::Model();
    }

    function add($from, $to, $subject, $msg, $dateToSend)
    {
        $data = array ('from'=>$from,
                      'to'=>$to,
                      'subject'=>$subject,
                      'message'=>$msg,
                      'blnSend'=>0,
                      'datPlanned'=>$dateToSend);
        $this->db->insert('tblemail', $data);
    }

    function send()
    {
        $this->load->library('email');

        $this->db->where('blnSend',0);
        $list = $this->db->get('tblemail')->result();

        foreach ($list as $email)
        {
            $this->email->clear();

            $this->email->to($email->to);
            $this->email->from($email->from);
            $this->email->subject($email->subject);
            $this->email->message($email->message);

            if($this->email->send())
            {
                $this->db->set('blnSend',1);
                $this->db->set('datSend',date('Y-m-d'));
                $this->db->where('ID_Email', $email->ID_Email);
                $this->db->update('tblemail');
            }
            else
            {
                $this->db->set('blnSend',0);
                $this->db->set('blnError',1);
                $this->db->set('msgError',$this->email->print_debugger());
                $this->db->where('ID_Email', $email->ID_Email);
                $this->db->update('tblemail');
            }
        }
    }
}

```



# Security

- Sanitize input, escape output.
- Be aware of XSS (CI provides some security).
- Be aware of Cross-Site Request Forgery (CSRF).
- Be aware of (MySQL) typecasting when using Active Record.



# Example 1 - PyroCMS

	Name	E-mail	Role	Joined	Last visit	Actions
<input type="checkbox"/>	Test user	<a href="mailto:jeroen@isset.nl">jeroen@isset.nl</a>	admin	Oct 19, 2009	Never	<a href="#">Edit</a>   <a href="#">Delete</a>
<input type="checkbox"/>	Demo User	<a href="mailto:demo@example.com">demo@example.com</a>	admin	Sep 09, 2008	Oct 20, 2009	<a href="#">Edit</a>   <a href="#">Delete</a>


<http://testsite.nd/admin/users/delete/2>



# Example 2 - SyntaxCMS

**List of Users** <http://demo.syntaxmonster.net/user/admin/delete/81>

[Get Help With This Page](#) [Manage Users](#) [Add a User](#)

Display Name	Email	Group	Actions
syntaxcms	demo@demo.com	Admin	 
jeroen	jeroen@isset.nl	Admin	 







## How to prevent CSRF

- Any destructive action should always be taken with a POST (update and delete).
- Cross Site POST is easy so gives no security.
- Use a token to verify that the send POST data is coming from the website.
- Michael Wales wrote extensively about it:  
<http://www.michaelwales.com/codeigniter/protecting-against-csrf-exploit>



```
function delete($user_id = NULL)
{
    $this->load->library('form_validation');

    $this->form_validation->set_rules('user_id', 'User ID', 'required|is_natural');

    if ($this->form_validation->run() == FALSE)
    {
        $data['user_id'] = $user_id;
        $data['key']     = $this->csrftoken();
        $data['value']   = $this->csrftoken();
        $this->session->set_flashdata('csrfkey', $data['key']);
        $this->session->set_flashdata('csrfvalue', $data['value']);
        $this->load->view('example_form', $data);
    }
    else
    {
        if ($this->input->post($this->session->flashdata('csrfkey')) != FALSE
            AND
            $this->input->post($this->session->flashdata('csrfkey')) ==
            $this->session->flashdata('csrfvalue'))
        {
            echo 'succes';
        }
        else
        {
            echo 'fail';
        }
    }
}

function csrftoken()
{
    $start = mt_rand(1,15);
    $length= mt_rand(3,16);
    $hash  = md5(mt_rand());
    $token = substr($hash, $start, $length);
    return $token;
}
```



```
<form action="/example/delete" method="post">
    <input type="hidden" name="<?=$key; ?>" value="<?=$value; ?>" />
    <input type="hidden" name="user_id" value="<?=$user_id; ?>" />
    <input type="submit" value="Are you sure?" />
</form>
```

```
<form action="/example/delete" method="post">
    <input type="hidden" name="982baa5f" value="a6b75893d9cc68" />
    <input type="hidden" name="user_id" value="1" />
    <input type="submit" value="Are you sure?" />
</form>
```



```
function delete()
{
    $email = $this->input->post('emial');

    $this->db->where("email", $email);
    $this->db->delete("tblemail");
    echo "SQL executed : " . $this->db->last_query();
    echo "<br />";
    echo "Affected Rows:". $this->db->affected_rows();
}
```



SQL executed : DELETE FROM `tblemail` WHERE `email` = 0  
Affected Rows:63



## Making sense of SEO / SEM efforts

- Keeping track of the position for a given landing page in combination with a search string
- Monitoring Google PageRank
- Measure cause and effect listing milestones in conversion / visitor timeline.
- Relate all search strings to conversions.
- Adding search strings to a campaign that are cheaper but also make a conversion
- Exclude search strings from a campaign that are expensive but never make a conversion.



## You can (in a few months)

- Visit [sedindex.com](http://sedindex.com) (next monday)
- Pre register
- Find more information
- We will get back to you in January 2010





Questions ?

Thanks for listening. We hope you enjoyed it