# Processus de choix d'un framework php et de son intégration pour des interfaces de gestion de BDDs en archéologie

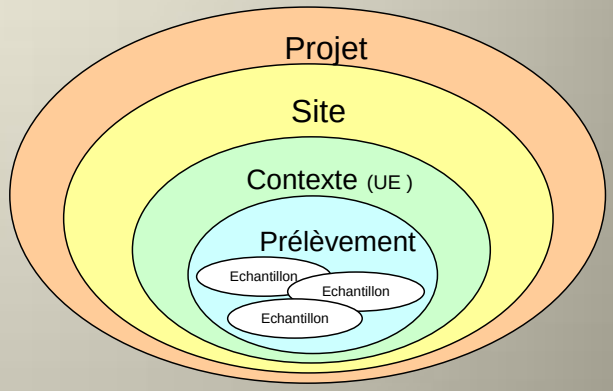

T8.GT5 Retour d'expériences sur les framework Web

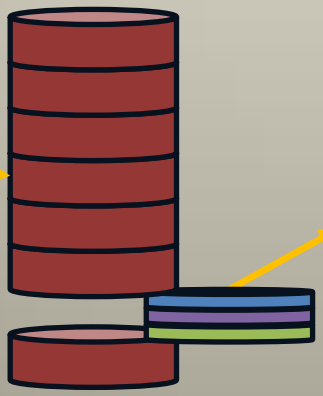
Jeudi 5 septembre 2013

jean-baptiste.barreau@univ-rennes1.fr (UMR6566)

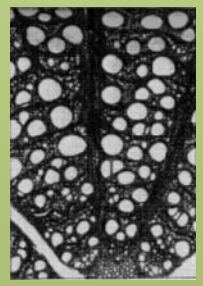
# Projet ABCData (Archéologie, Biodiversité, Chronologie Data)

Bibliothèque de données paléoenvironnementales

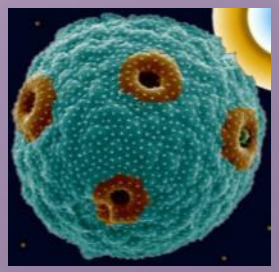


Prélèvement par carottage



Echantillon anthracologique



Echantillon palynologique

Echantillon malacologique

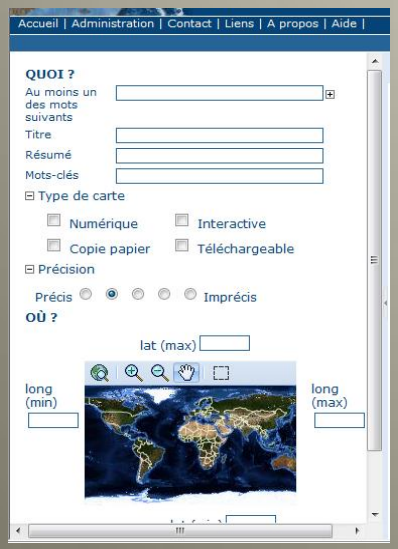
# Besoins et contraintes

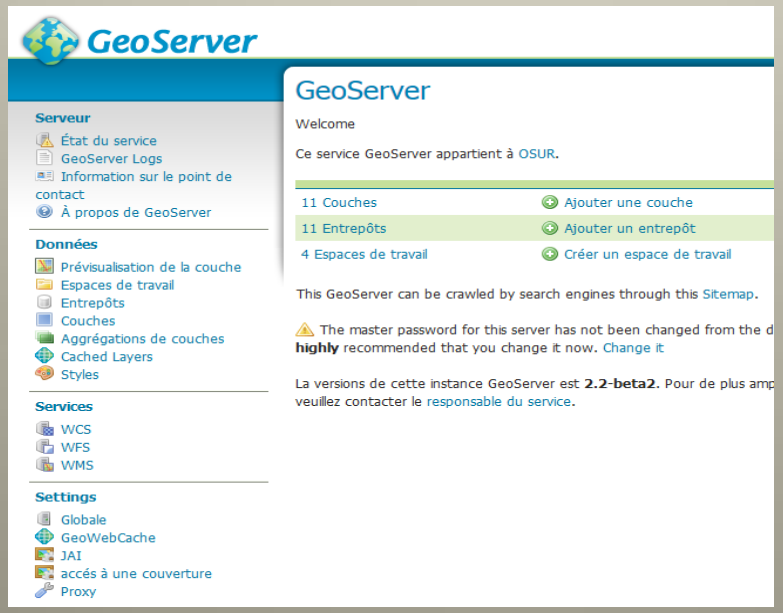
-Base de données multi-utilisateurs
-SGBD PostgreSQL / PHP
-Compatible avec la base BIOARCHEODAT (UMR7209): recensement des données archéozoologiques et archéobotaniques de l'Holocène, en France Métropolotaine.
-Générer des interfaces web simplement et rapidement
-Créer, Lire, Editer, Supprimer (CRUD)
-Tableaux de moyennes dimensions à entêtes fixes
-des cartes interactives:
    -icônes cliquables marquant les sites archéologiques
    -mise en valeur du site sélectionné
    -affichage des échantillons du site
-quelques relations many-to-one et many-to-many
-authentification sécurisée

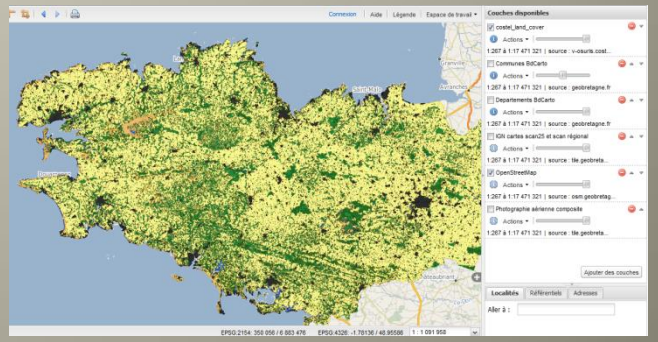
# OSURIS



UNIVERSITÉ RENNES 1

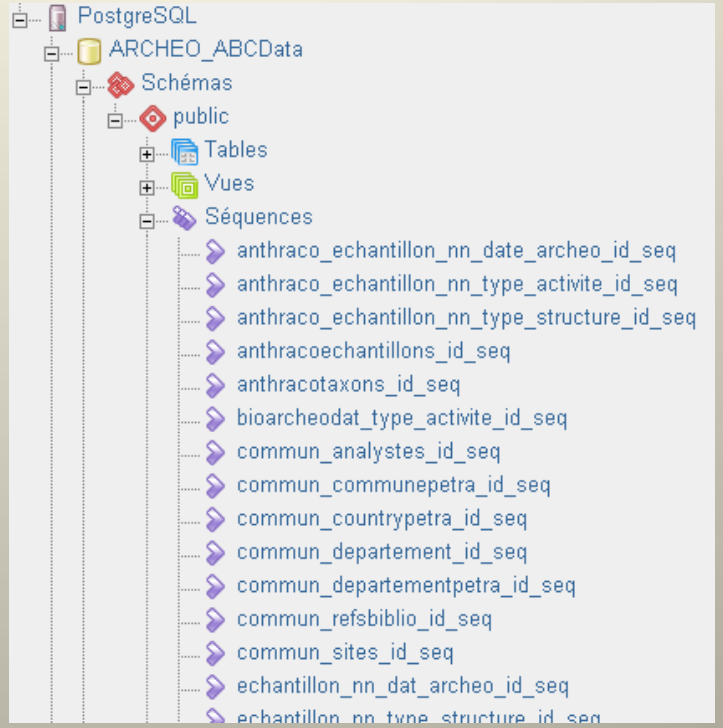
geovisu

# PostgreSql / PostGIS


-Système de gestion de base de données relationnel-objet
-Très stable
-Possibilités de programmation étendues
-Libre et opensource
-Nombreux types de données
-Communauté française très active


-Extension de PostgreSQL: manipulation d'informations géographiques sous forme de géométries (points, lignes, polygones)
-Conforme aux standards établis par l'Open Geospatial Consortium
-Libre
-GDAL (raster), OGR (vecteur), PROJ.4 (projections), GEOS (géométrie)
-Fonctions de base SIG (rééchantillonnage, rognage, intersection, union, projection, ...)
-Fonctions algébriques
-…

# phpPgAdmin


Pour générer des identifiants uniques pour les lignes d'une table (eq champ auto-incrémenté)

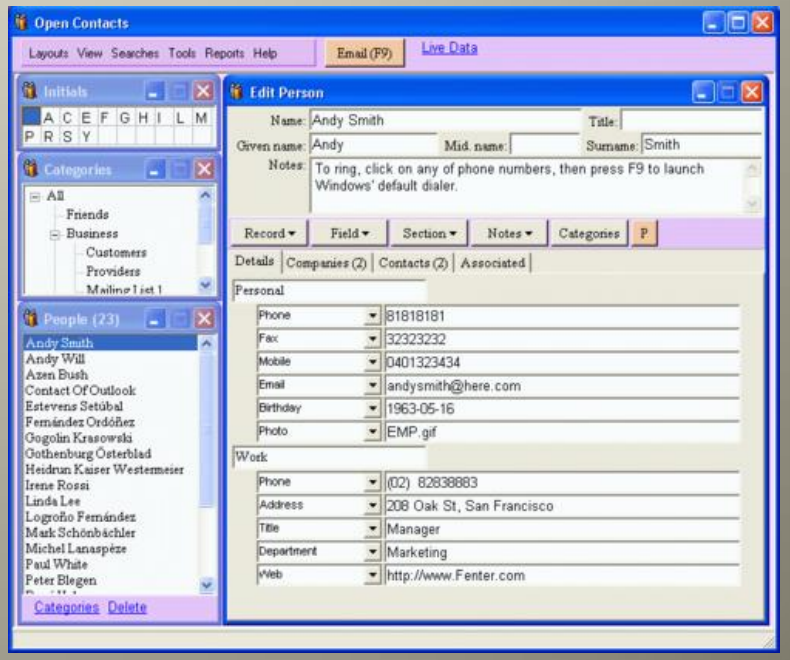
Colonnes | Index? | Contraintes? | Triggers? | Règles? | Info | Droits? | Importer

| Colonne | Type | NOT NULL | Défaut | Contraintes | Actions | Commentaire |
|---|---|---|---|---|---|---|
| id | integer | NOT NULL | nextval('anthracoechantillons_id_seq'::regclass) | | Parcourir Modifier Supprimer | |
| nom | character(32) | | | | Parcourir Modifier Supprimer | |
| id_site | integer | | | | Parcourir Modifier Supprimer | |
| precision_structure | text | | | | Parcourir Modifier Supprimer | |
| precision_type_activite | character(255) | | | | Parcourir Modifier Supprimer | |
| c_bp | character(32) | | | | Parcourir Modifier Supprimer | |
| c_cal_bc | character(32) | | | | Parcourir Modifier Supprimer | |
| id_analyste | integer | | | | Parcourir Modifier Supprimer | |

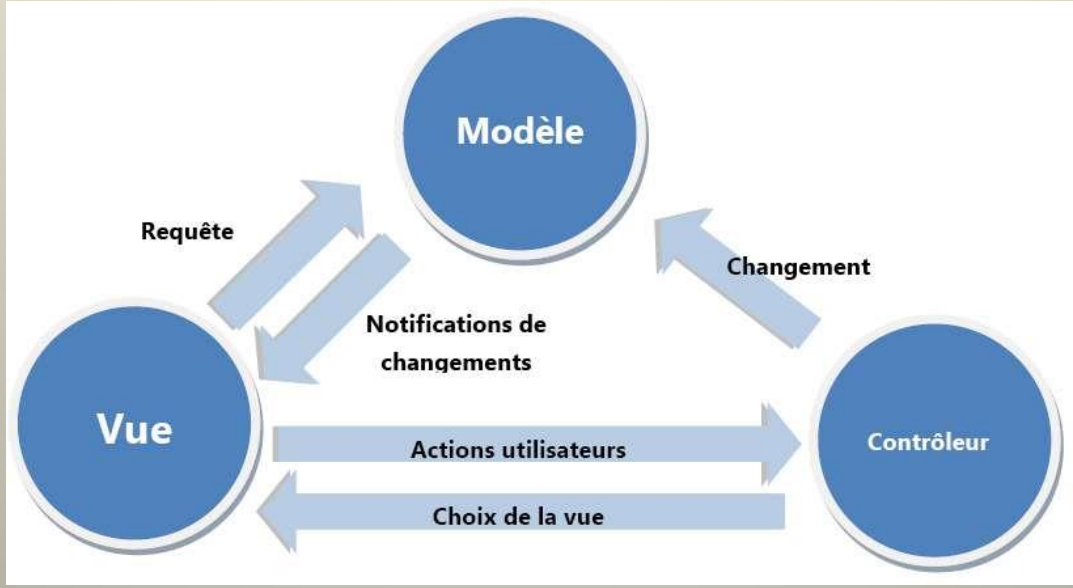
# CRUD

CRUD (Create, Read, Update, Delete) désigne les quatre opérations de base pour la persistance des données.

Exemple d'un carnet d'adresse:
-l'élément le plus simple est un contact
-l'application doit permettre:
-Créer un contact
-Lire un contact
-Mettre à jour un contact
-Supprimer un contact



# Architecture MVC




Répondre aux besoins des applications interactives en structurant l'application :

• Modèle (modèle de données): Décrit les données manipulées par l'application, interagit avec la base de données.

• Vue (présentation, interface utilisateur): Présente les résultats renvoyés par le modèle et reçoit toute action de l'utilisateur.

• Contrôleur (logique de contrôle, gestion des événements, synchronisation): Analyse la requête du client, appelle le modèle adéquat et renvoie la vue correspondant à la demande.

# Choix d'un framework PHP

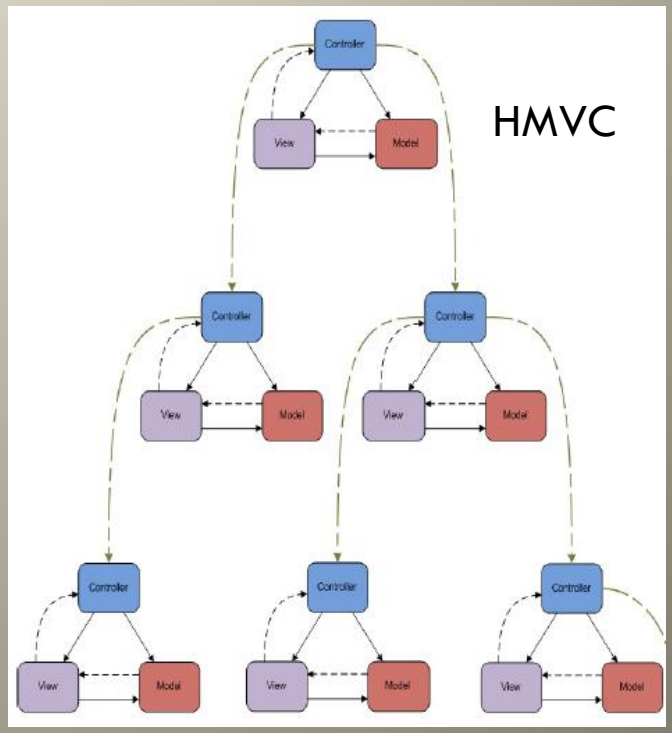
| Nom | PostgreSQL? | CRUD? | MVC? |
|---|---|---|---|
| Agavi | oui | non | oui |
| CodeIgniter | oui | non | oui |
| **FuelPHP** | **PHP Data Objects (PDO)** | **oui** | **oui** |
| **Symfony** | **oui** | **oui** | **oui** |
| Zend Framework | oui | non | oui |
| Kohana | oui | non | oui |
| Jelix | oui | non | oui |
| KumbiaPHP | oui | non | oui |
| Qcodo | oui | non | oui |
| Akelos PHP | oui | non | oui |


>

Symfony

# FuelPHP

-se définit comme une version enrichie de CodeIgniter
-utilise l'extension-interface PHP Data Objects (PDO)
-respecte l'architecture HMVC (Hierarchical MVC)
-framework relativement jeune (2010)
-dispose de son ORM oil
-très flexible
-documentation très complète
-moins riche que Symfony mais plus simple



# Configuration des paramètres de connexion

*db.php*

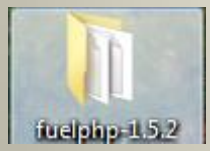
```php
/**
 * The development database settings. These get merged with the global settings.
 */
return array(
    'default' => array(
        'connection'     => array(
            'dsn'            => 'pgsql:host=localhost;dbname=ARCHEO_ABCData',
            'username'       => [redacted],
            'password'       => [redacted],
            'persistent'     => false,
            'compress'       => false,
        ),
        'identifier'   => '"',
        'table_prefix'   => '',
        'charset'        => 'utf8',
        'enable_cache'   => true,
        'profiling'      => false,
    ),
);
```

# Model_Crud vs Orm\Model

-Effectue simplement les opérations CRUD
-Pour des opérations simples sur les tables
-Très rapide à mettre en place
-Génération de code par Scaffold
-Utilisation: *class Model_... extends \Model_Crud*

-Pas de gestion des relations, pour ça: ORM (object-relational mapping)
-Model_Crud et ORM utilisent une syntaxe semblable: possibilités de migrations…

# Création des fichiers du CRUD



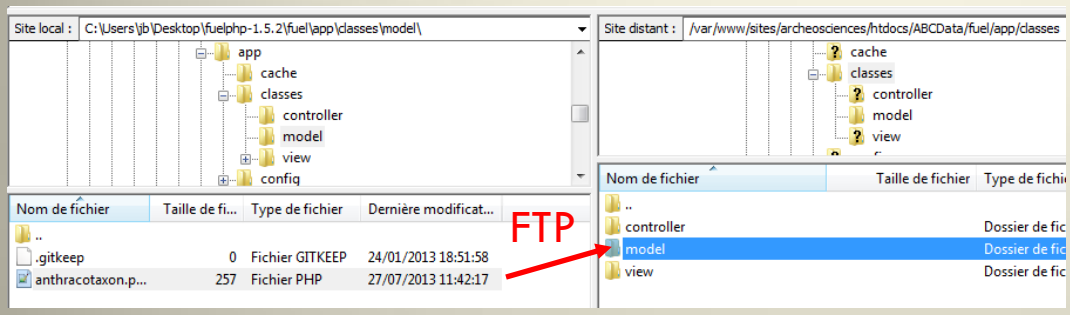

| | | | |
|---|---|---|---|
| docs | 08/02/2013 13:07 | Dossier de fichiers | |
| fuel | 24/01/2013 17:51 | Dossier de fichiers | |
| public | 24/01/2013 17:51 | Dossier de fichiers | |
| .gitignore | 24/01/2013 17:51 | Fichier GITIGNORE | 1 Ko |
| .gitmodules | 08/02/2013 12:58 | Fichier GITMODU... | 1 Ko |
| build.xml | 24/01/2013 17:51 | Document XML | 3 Ko |
| CHANGELOG.md | 08/02/2013 12:58 | Fichier MD | 40 Ko |
| CONTRIBUTING.md | 24/01/2013 17:51 | Fichier MD | 7 Ko |
| oil | 24/01/2013 17:51 | Fichier | 2 Ko |
| README.md | 24/01/2013 17:51 | Fichier MD | 2 Ko |
| TESTING.md | 24/01/2013 17:51 | Fichier MD | 4 Ko |

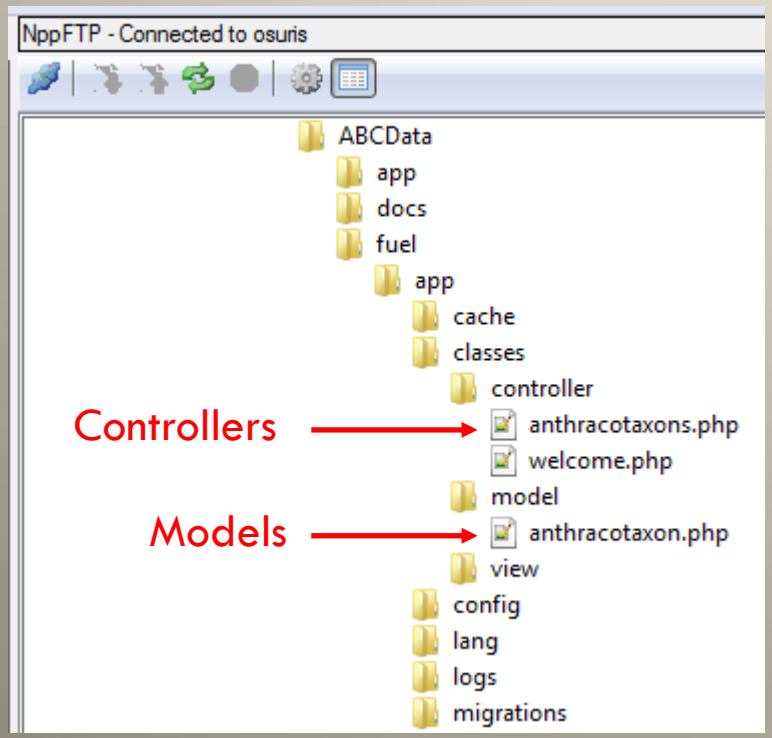
Colonnes | Index? | Contraintes? | Triggers? | Règles? | Info

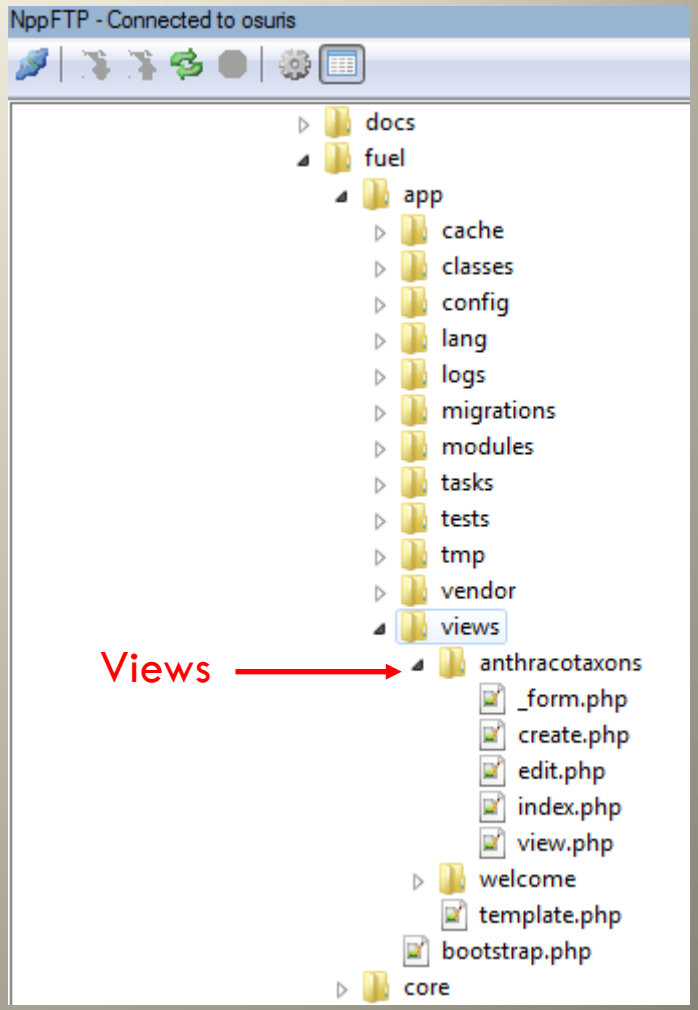
| Colonne | Type | NOT NULL | Défaut | Contraintes | Actions | Commentaire |
|---|---|---|---|---|---|---|
| id | integer | NOT NULL | nextval('taxons_id_seq'::regclass) | | Parcourir Modifier Supprimer | |
| nom | character(32) | | | | Parcourir Modifier Supprimer | |

```
C:\Users\jb\Desktop\fuelphp-1.5.2>php oil generate scaffold/crud anthracotaxons
nom:character(32)
        Creating migration: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\migration
s/002_create_anthracotaxons.php
        Creating model: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\classes/model
/anthracotaxon.php
        Creating controller: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\classes/
controller/anthracotaxons.php
        Creating view: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\views/anthraco
taxons/index.php
        Creating view: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\views/anthraco
taxons/view.php
        Creating view: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\views/anthraco
taxons/create.php
        Creating view: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\views/anthraco
taxons/edit.php
        Creating view: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\views/anthraco
taxons/_form.php
```

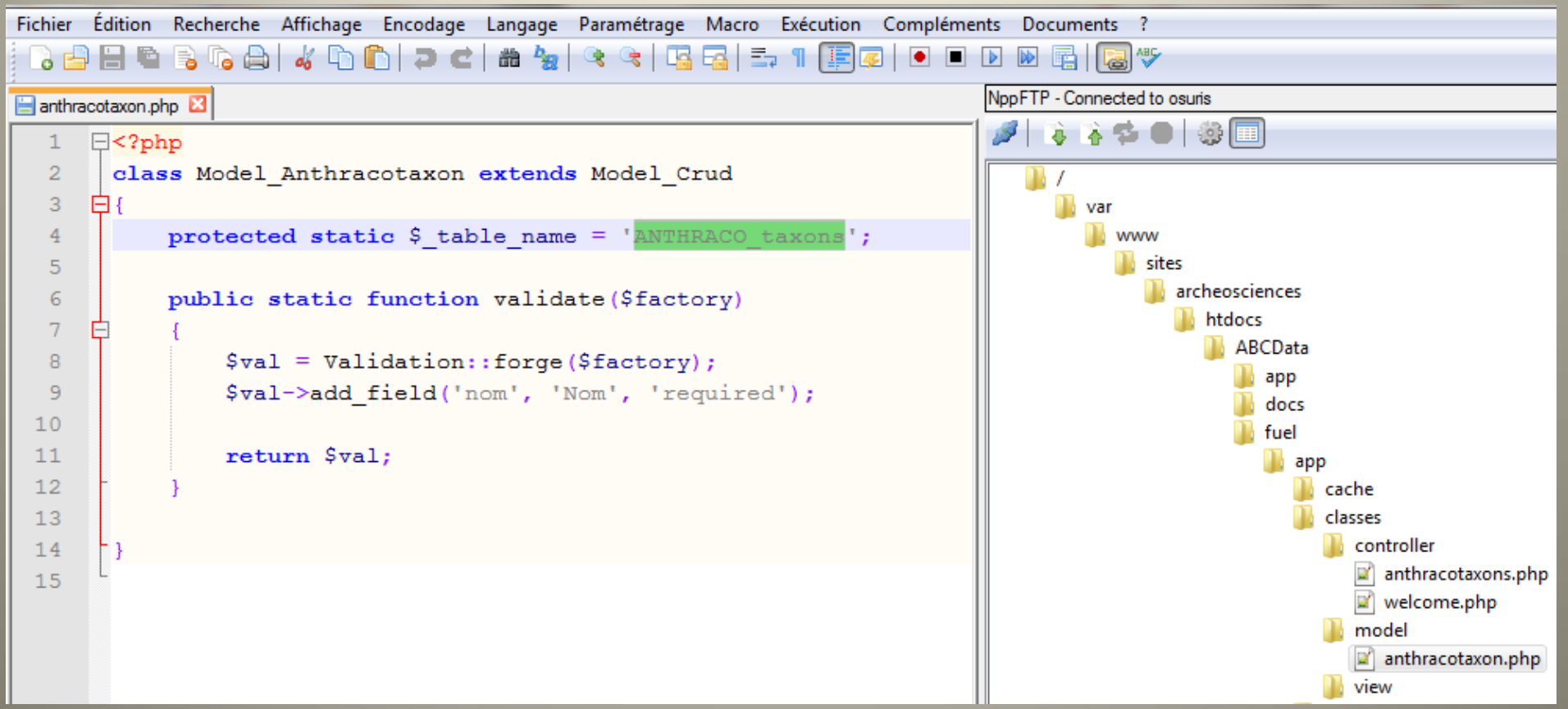
# Upload des fichiers du CRUD



# Modification du nom de la table dans le modèle



# …/index.php/anthracotaxons

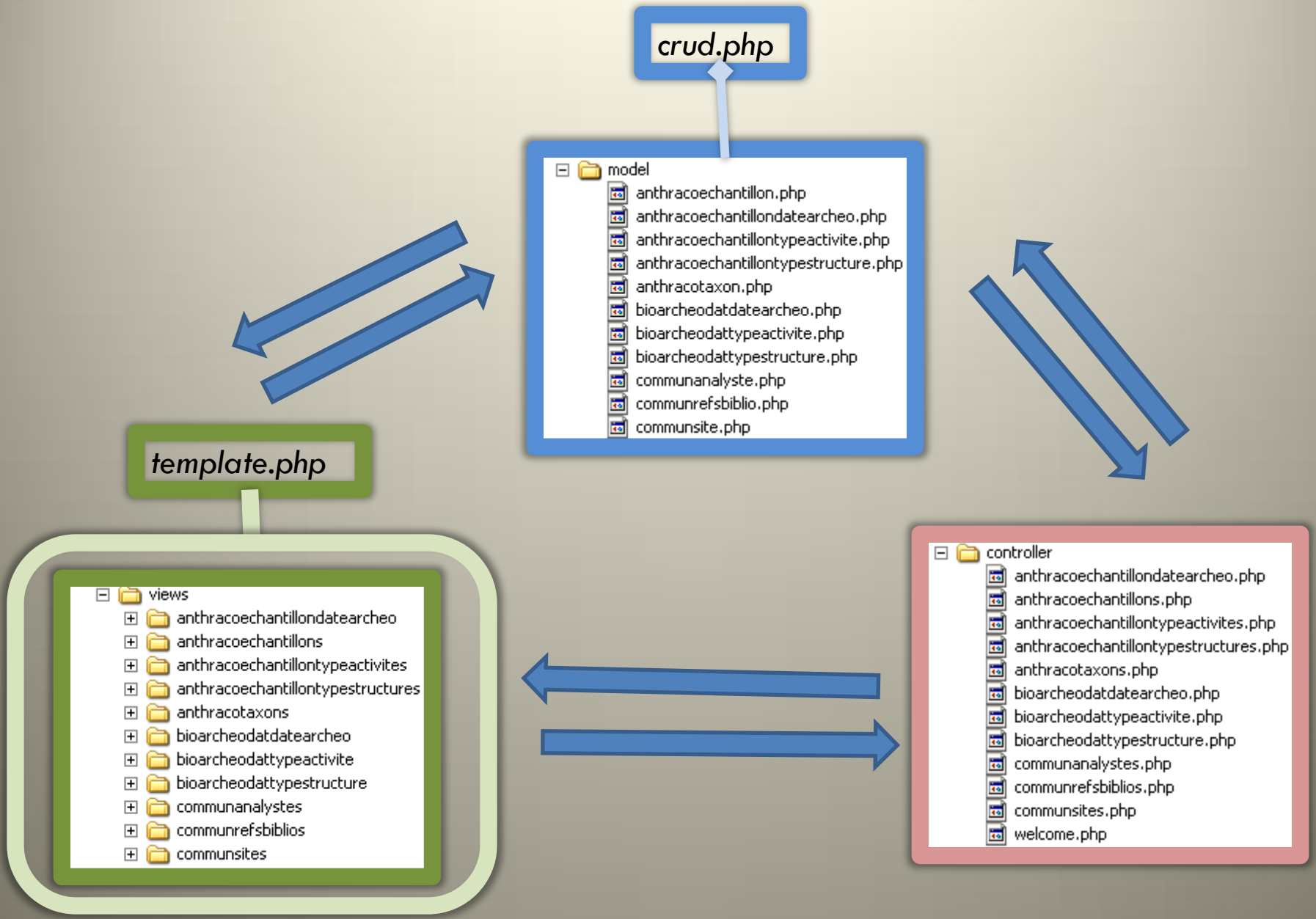
## Anthracotaxons

## Listing Anthracotaxons

| Nom | |
|---|---|
| Quercus sp. | View \| Edit \| Delete |
| Quercus/Castanea | View \| Edit \| Delete |
| Cytisus sp. | View \| Edit \| Delete |
| Ulex sp. | View \| Edit \| Delete |
| Fagus sylvatica | View \| Edit \| Delete |
| Juglans sp | View \| Edit \| Delete |
| Fraxinus sp. | View \| Edit \| Delete |
| Populus sp. | View \| Edit \| Delete |
| Salix sp. | View \| Edit \| Delete |
| Acer sp. | View \| Edit \| Delete |
| Alnus sp. | View \| Edit \| Delete |
| Castanea sativa | View \| Edit \| Delete |
| Ulmus sp. | View \| Edit \| Delete |
| Pinus sp. | View \| Edit \| Delete |

Merci de votre attention ;-)

# Organisation des fichiers



# Tableaux à entêtes fixes (1)

http://www.tablefixedheader.com/


*template.php*

```php
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title><?php echo $title; ?></title>
    <?php
    echo Asset::css('bootstrap.css');
    echo Asset::css('fixheadertable.css');
    echo Asset::css('jquery-ui-1.10.3.custom.css');
    echo Asset::js('https://ajax.googleapis.com/ajax/libs/jquery/1.5.2/jquery.min.js');
    echo Asset::js('jquery.fixheadertable.js');
```

*index.php*

```php
echo initFixHeaderTable(400);
?>
<table class="fixme">
        <thead>
        <tr>
            <th>Nom échantillon</th>
            <th>Site archéologique</th>
            <th>Dates archeo</th>
            <th>Types de structure</th>
```

*template.php*

```php
function initFixHeaderTable($_Height)
{
    $l_result = '<script type="text/javascript">
                $(document).ready(function() {
                    $(".fixme").fixheadertable({
                        caption : "",
                        resizeCol: true,
                        whiteSpace: "normal",
                        sortable: false,
                        height  : ' . $_Height . '
                    });
                });
                </script>';
    return $l_result;
}
```

# Tableaux à entêtes fixes (2)

```php
function createIconsHTMLCode($_Modelname,$_IconSize)
{
    $icons = Html::anchor($_Modelname . '/view/tag_id', Asset::img('view.png', array('width' => $_IconSize))) .
    $icons .= Html::anchor($_Modelname . '/edit/tag_id', Asset::img('edit.png', array('width' => $_IconSize)))
    $icons .= Html::anchor($_Modelname . '/delete/tag_id', Asset::img('delete.png', array('width' => $_IconSize
    return $icons;
}
```

*template.php*

```php
<?php if ($anthracoechantillons):
$icons = createIconsHTMLCode('anthracoechantillons','24');
echo initFixHeaderTable(400);
?>
```

*index.php*

```php
<td align="center">
    <?php echo str_replace("tag_id", $anthracoechantillon->id, $icons); ?>
</td>
```

*index.php*

| Nom échantillon | Site archéologique | Dates archeo | Types de structure | Types d'activité | C bp | C cal bc | Analyste | Référence | |
|---|---|---|---|---|---|---|---|---|---|
| maur_tp.146_data | La Rochette | -Bronze final IIa<br>-Bronze final IIb<br>-Bronze final IIc<br>-Bronze final IIIa<br>-Bronze final IIIb | -Trou de poteau (précision: entrée habitat fortifié) | -Construction<br>-Autre bois d'œuvre (précision: activité domestique - construction - autre bois d'œuvre ) | | | MARCOUX Nancy | Tinévez et al. (2011), RAO N °28, p 71-148 | |
| maur_tp.154_data | La Rochette | -Bronze final IIa<br>-Bronze final IIb<br>-Bronze final IIc<br>-Bronze final IIIa<br>-Bronze final IIIb | -Trou de poteau (précision: entrée habitat fortifié) | -Activité domestique<br>-Construction<br>-Autre bois d'œuvre (précision: activité domestique construction autre bois d'œuvre ) | 2860 ± 40 | 1191-913 | MARCOUX Nancy | Tinévez et al. (2011), RAO N °28, p 71-148 | |

# Cartes (1)

Méthode possible pour gérer l'affichage de cartes sans accès à un serveur de données géographiques « clé en main »: OpenLayers + Leaflet + Proj4js

*carte_open_layer.js*

```
function init()
{
    la_carte = new OpenLayers.Map('div_ma_carte');
    layer_de_base =  new OpenLayers.Layer.OSM();
    AutoSizeAnchored = OpenLayers.Class(OpenLayers.Popup.Anchored, {'autoSize': true});
    couche_markers = new OpenLayers.Layer.Markers("Markers");

    la_carte.addLayer(layer_de_base);

    var un_point=new OpenLayers.LonLat(200273.000,2356052.000);
    un_point=un_point.transform(
                new OpenLayers.Projection("EPSG:27572"),
                new OpenLayers.Projection("EPSG:900913")
              );

    var zoom=8;

    la_carte.setCenter(un_point,zoom);
    la_carte.addControl(new OpenLayers.Control.MousePosition());

    /**************************************/
    la_carte.addLayer(couche_markers);
    var longlat, popupClass, popupContentHTML;
    for (x in l_points.rows)
    {
        longlat = new OpenLayers.LonLat(l_points.rows[x].lat,l_points.rows[x].lon).transform( new OpenLayers.Projection("EPSG:
        popupClass = AutoSizeAnchored;
        popupContentHTML = l_points.rows[x].description;
        addMarker(longlat, popupClass, popupContentHTML,null,l_points.rows[x].iconpath,l_points.rows[x].iconwidth,l_points.row
    }
}
```

***Proj4js**: librairie JavaScript pour transformer des coordonnées, Lambert II => WGS84 Web Mercator.*

```
<body onload="init();">
```

*template.php*

# Cartes (2)

*anthracoechantillons.php*

```php
public function action_map($id = null)
{
    $data['communsites'] = Model_Communsite::find(array('order_by' => array('id' => 'asc')));
    $SymbolPathArray = array_fill(0, count($data['communsites']), 'AnthracTag');
    $l_Title = "Carte des échantillons anthracologiques";
    if($id)
    {
        $data['anthracoechantillons'] = Model_Anthracoechantillon::find( array( 'where' => array( 'id_site' => $id ), '
        $SymbolPathArray[$id-1] = "AnthracSelectTag";
        $l_Title = "Échantillons anthracologiques du site \" " . $data['getAllSite'][$id] . "\"";
    }

    $data['initMap'] = Model_Communsite::initMap($data['communsites'],'x_lambert','y_lambert',$SymbolPathArray,'nom_sit
    $this->template->title = $l_Title;
    $this->template->content = View::forge('anthracoechantillons/map', $data);
}
```

*crud.php*

```php
public static function initMap $Array,$X_param,$Y_param,$_SymbolPathArray,$Name)
{
    $l_result = "";
    $l_result = '<script type="text/javascript">var l_points={"rows" : [';
    $i = 0;
    foreach ($Array as $l_value):
        $l_result .= '{';
        $l_result .= '"lat" : "' . $l_value->$X_param . '",';
        $l_result .= '"lon" : "' . $l_value->$Y_param . '",';
        $l_result .= '"title" : "titre",';
        $l_result .= '"iconpath" : "' . $_SymbolPathArray[$i] . '",';
        $l_result .= '"description" : "' . utf8_encode(trim($l_value->$Name) . "<br>tag_icons " . $l_value->id ) . '",';
        $l_result .= '"iconwidth" : "24",';
        $l_result .= '"iconheight" : "24",';
        $l_result .= '"iconOffset" : "0,-24"';
        $l_result .= '},';
        $i++;
    endforeach;
    $l_result .= ']};</script>';
    return $l_result;
}
```

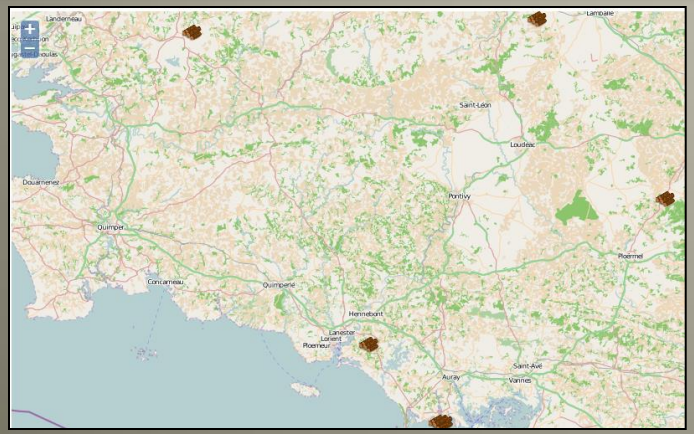
# Cartes (3)

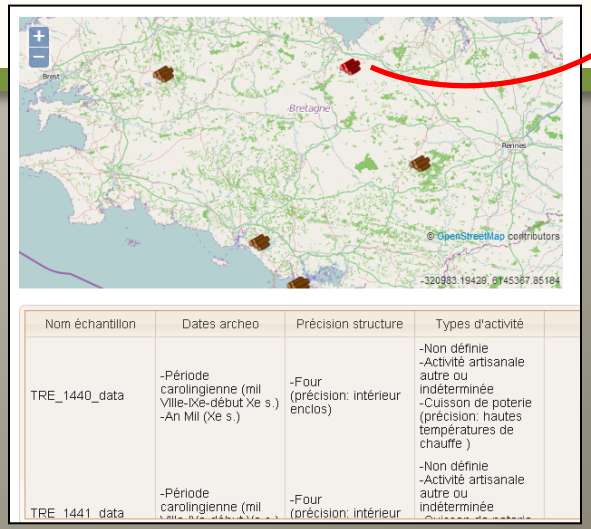
```php
$initMap = setLinksOnMap( $initMap, Html::anchor('anthracoechantillons/map/tag_id', 'Voir les échantillons' ) );
echo showMap($initMap,"600px","300px") . '<br>';
```

*map.php*

```php
function setEditIconsOnMap($_Map,$_EditIconsHTMLCode="",$_SrcIconSize="",$_MapIconSize="")
{
    return preg_replace('/tag_icons (\d+)/i',str_replace("tag_id",'${1}',str_replace(array("\"",$_SrcIconSize),array("'"
}

function setLinksOnMap($_Map,$_LinkHTMLCode)
{
    return preg_replace('/tag_icons (\d+)/i',str_replace("tag_id",'${1}',str_replace(array("\""),array("'"),$_LinkHTMLCo
}

function showMap($_initMap,$_Width,$_Height)
{
    $MapSymbolsPath = "http://archeosciences-abcdata.osuris.org/assets/img/map_symbols/";
    $TagSymbols = array("AnthracTag","AnthracSelectTag");
    $SymbolFiles = array( $MapSymbolsPath . "icon-anthraco.png", $MapSymbolsPath . "icon-anthraco-selected.png" );
    $l_result = str_replace ( $TagSymbols, $SymbolFiles, $_initMap );
    $l_result .= '<div style="width:' . $_Width . '; height:' . $_Height . '" id="div_ma_carte"></div>';
    return html_entity_decode($l_result);
}
```

*template.php*






| Nom échantillon | Dates archeo | Précision structure | Types d'activité |
|---|---|---|---|
| TRE_1440_data | -Période carolingienne (mil Ville-IXe-début Xe s.)<br>-An Mil (Xe s.) | -Four (précision: intérieur enclos) | -Non définie<br>-Activité artisanale autre ou indéterminée<br>-Cuisson de poterie (précision: hautes températures de chauffe ) |
| TRE_1441_data | -Période carolingienne (mil | -Four (précision: intérieur | -Non définie<br>-Activité artisanale autre ou indéterminée |

# Relations 1-n: index.php

crud.php | anthracoechantillons.php | index.php

```php
//retourn tableau complet pour relation n-1
public static function getAllTable($table1, $champ1, $champ2, $table2, $champ3)
{
    $modif1=$table1.'.'.$champ1;
    $modif2=$table2.'.'.$champ3;
    $query=DB::select($modif1, $champ2)->from($table1)->join($table2)->on($modif1, '=', $modif2)->execute()->as_array($champ1,$champ2);
    return $query;

}
```

*crud.php*

```php
<?php
class Controller_Anthracoechantillons extends Controller_Template
{

    public function action_index()
    {
        $data['getAllSite'] = Model_Anthracoechantillon::getAllTable( 'COMMUN_sites', 'id', 'nom_site', 'ANTHRACO echantillons', 'id site');
        $data['anthracoechantillons'] = Model_Anthracoechantillon::find_all();
        $this->template->title = "Anthraco";
        $this->template->content = View::forge('anthracoechantillons/index', $data);

    }
```

| id_site | |
|---|---|
| 1 | Site archéologique |
| 1 | |
| 1 | Champ du Château |

*anthracoechantillons.php*

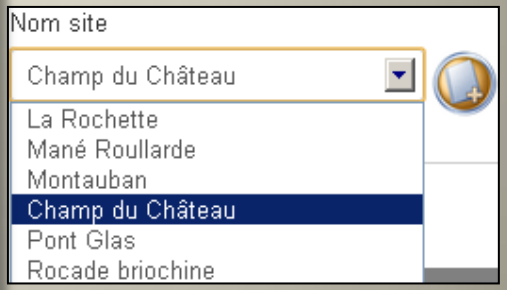
```php
<?php foreach ($anthracoechantillons as $anthracoechantillon): ?>        <tr>

        <td><?php echo $anthracoechantillon->nom; ?></td>
        <td>
        <?php if (($anthracoechantillon->id_site != null)) { print_r($getAllSite[$anthracoechantillon->id_site]); } else { echo $anthracoechantillon->id_site = null; }
        ?>
```

*index.php*

# Relations 1-n: form.php

*anthracoechantillon.php*

```php
<?php
class Model_Anthracoechantillon extends Model_Crud
{
    protected static $_table_name = 'ANTHRACO_echantillons';
    public static $_relation_params = Array
                                        (
                                            Array
                                                (
                                                    'table_name' => 'COMMUN_sites',
                                                    'id_field' => 'id_site',
                                                    'name_field' => 'nom_site',
                                                    'all_var' => 'getAllSite',
                                                    'one_var' => 'getOneSite',
                                                    'query_var' => 'EditSiteQuery'
                                                ),
```




*anthracoechantillons.php*

```php
public function action_edit($id = null)
{
    //Relations 1-n
    $data = Model_Anthracoechantillon::getRelationDatas($id,'id','query_var','ANTHRACO_echantillons',Model_Anthracoechantillon::$_relation_para
    $data = array_merge($data, Model_Anthracoechantillon::getRelationDatas($id,'id','one_var','ANTHRACO_echantillons',Model_Anthracoechantillon
```

*form.php*

```php
<div class="input">
    <?php
        $id = "";
        if(isset($getOneSite)) { $id = array_search($getOneSite, $EditSiteQuery); }
        echo Form::select('id_site', $id, $EditSiteQuery);
        echo " " . Html::anchor('communsites/create', Asset::img('add.png', array('width' => '35')));
    ?>
</div>
```

# Relations n-n : index.php

*crud.php*

```php
public static function getManyManyRelationArray($_ExternalTable,$_SelfIdField,$_ExternalIdField,$_ReturnNames=true)
{
    $l_cur_ech = -1;
    $l_Array = self::find(array('order_by' => array($_SelfIdField => 'asc')));
    if($l_Array)
    {
        foreach ( $l_Array as $value )
        {
            if( $l_cur_ech != $value[$_SelfIdField] )
            {
                $l_cur_ech = $value[$_SelfIdField];
                $l_result[$value[$_SelfIdField]] = array();
            }
            if($_ReturnNames)
            {
                $l_result[$value[$_SelfIdField]][] = $_ExternalTable[$value[$_ExternalIdField]];
            }
            else
            {
                $l_result[$value[$_SelfIdField]][] = $value[$_ExternalIdField];
            }
        }
    }
    return $l_result;
```

| id | echantillon | type_structure | Types de structure |
|---|---|---|---|
| 1 | 1 | 16 | -Fosse (étudiée dans un complexe) |
| 2 | 1 | 23 | -Ensemble de fosses/Trous (précision: calage/dalle posée à plat au fond) |
| 3 | 2 | 16 | |
| 4 | 2 | 23 | |
| 5 | 3 | 16 | |
| 6 | 3 | 23 | |

```php
function showManyToManyList($_ManyToManyArray)
{
    if(isset($_ManyToManyArray))
    {
        foreach ($_ManyToManyArray as $l_value):
            echo "-" . $l_value . "<br>";
        endforeach;
    }
```

*template.php*

*anthracoechantillons.php*

```php
public function action_view($id = null)
{
    //Relations 1-n
    $data = Model_Anthracoechantillon::getRelationDatas($id,'id','one_var','ANTHRACO_echantillons',Model_Anthracoechantillon::
    $_relation_params);

    //Relations n-n
    $data['anthracoechantillontypeactivite'] = Model_Anthracoechantillontypeactivite::getManyManyRelationArray(Model_Anthracoechantillon
    ::getTabAssoc('BIOARCHEODAT_type_activite', 'id', 'description'),'ech_id','activite_id');
    $data['anthracoechantillondatearcheo'] = Model_Anthracoechantillondatearcheo::getManyManyRelationArray(Model_Anthracoechantillon::
    getTabAssoc('BIOARCHEODAT_dat_archeo', 'id', 'description'),'echantillon','dat_arch');
    $data['anthracoechantillontypestructure'] = Model_Anthracoechantillontypestructure::getManyManyRelationArray(
    Model_Anthracoechantillon::getTabAssoc('BIOARCHEODAT_type_structure', 'id', 'description'),'echantillon','type_structure');
```

```php
if (isset($anthracoechantillondatearcheo[$anthracoechantillon->id])) { showManyToManyList($anthracoechantillondatearcheo[
$anthracoechantillon->id]); }
?>
```

*index.php*

# Relations n-n : form.php

```php
public static function saveManyManyRelation($_selfId,$_SelfFieldId,$_ExternalFieldId)
{
    $l_result = true;
    if(Input::post($_ExternalFieldId))
    {
        foreach (Input::post($_ExternalFieldId) as $l_value):
            $l_Model = self::forge(array(
                $_SelfFieldId => $_selfId,
                $_ExternalFieldId => $l_value,
            ));
            $l_result = $l_result && $l_Model->save();
        endforeach;
    }
    return $l_result;
```

*crud.php*

Dates archeo

- Cerny
- Chambon
- Chasséen ancien
- Chasséen récent

```php
$data['bioarcheodat_date_archeo'] = Model_Anthracoechantillon::getTabAssoc('BIOARCHEODAT_dat_archeo', 'id', 'description'

if ($anthracoechantillon and $anthracoechantillon->save())
{
    $l_LastId = Model_Anthracoechantillon::getLastId();
    if( Model_Anthracoechantillontypeactivite::saveManyManyRelation($l_LastId,'ech_id','activite_id') &&
        Model_Anthracoechantillondatearcheo::saveManyManyRelation($l_LastId,'echantillon','dat_arch') &&
        Model_Anthracoechantillontypestructure::saveManyManyRelation($l_LastId,'echantillon','type_structure')
      )
    {
        Session::set_flash( 'success', 'Création de l\'échantillon anthracologique n°'. $l_LastId . ' réussie.' );
        Response::redirect('anthracoechantillons');
    }
```

*anthracoechantillons.php*

```php
<?php
$l_SelectedTypeDateArcheo = getManyToManySelectedList( isset($anthracoechantillondatearcheo) ?
$anthracoechantillondatearcheo : null, isset($anthracoechantillon) ? $anthracoechantillon->id : null );
echo Form::select('dat_arch[]', $l_SelectedTypeDateArcheo, $bioarcheodat_date_archeo, array('class' => 'span4',
'multiple'));
?>
```

*form.php*

# Validation de saisie

## Rules table

| Rule | Additional parameters | Description |
|---|---|---|
| **required** | (none) | The field must be set and have been given something other than **null**, **false** or empty string. |
| **required_with** | **$fieldname** | The field must be set if the field with the given **$fieldname** is set. |
| **match_value** | **$compare**, **$strict** = false | The field input must match **$compare**, will be done using == unless 2nd parameter is also given as true (then === is used). |
| **match_pattern** | **$pattern** | Will try to match the value against the given $pattern which must be a full PREG regex. |

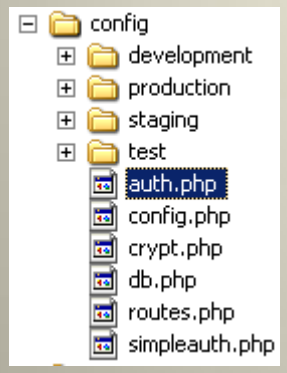
**Note:** you can **NOT** use the pipe symbol (|) in your pattern when you're using short syntax, as that symbol is used to split the rules in the string.

*anthracoechantillon.php*

```
public static function validate($factory)
{
    $val = Validation::forge($factory);
    $val->add_field('nom', 'Nom', 'required');
    $val->add_field('id_site', 'Id Site', 'required|valid_string[numeric]');
    //$val->add_field('precision_structure', 'Precision Structure', 'required');
    //$val->add_field('precision_type_activite', 'Precision Type Activite', 'required');
    //$val->add_field('c_bp', 'C Bp', 'required');
    //$val->add_field('c_cal_bc', 'C Cal Bc', 'required');
    $val->add_field('id_analyste', 'Nom Analyste', 'required|valid_string[numeric]');
    $val->add_field('id_reference', 'Reference', 'required|valid_string[numeric]');

    return $val;
}
```

Pour enlever la validation d'un champ (tests).

# Sécurisation (ex Simpleauth)




```php
'always_load' => array(
    'packages' => array(
        'auth',
    ),
),
```

config.php

```php
<?php

return array(
    // The drivers
    'driver' => array('Simpleauth'),

    // Set to true to allow multiple logins
    'verify_multiple_logins' => true,

    // Use your own salt for security reasons
    'salt' => 'Th1s=mYOWn_$@|+',
);
```

config.php

```php
public function action_login()
{
    $data = array();

    // If so, you pressed the submit button. Let's go over the steps.
    if (Input::post())
    {
        // Check the credentials. This assumes that you have the previous table created and
        // you have used the table definition and configuration as mentioned above.
        if (Auth::login())
        {
            // Credentials ok, go right in.
            Response::redirect('success_page');
        }
        else
        {
            // Oops, no soup for you. Try to login again. Set some values to
            // repopulate the username field and give some error text back to the view.
            $data['username']    = Input::post('username');
            $data['login_error'] = 'Wrong username/password combo. Try again';
        }
    }

    // Show the login form.
    echo View::forge('auth/login',$data);
}
```

users.php

# Merci de votre attention (2)