

# Desenvolvendo um pequeno módulo

Aqui, veremos como um pequeno módulo, contendo operações de cadastro, leitura, edição e deleção, pôde ser criado utilizando o MIOLO. Dada a realidade em que estamos, o módulo denominado GeLab (Gestão de Laboratórios) é implementado no e-Campus. Basicamente, refere-se à manutenção de dados de laboratórios, tendo como identificador apenas o seu nome.

Abaixo, temos os conteúdos dos arquivos necessários para o funcionamento do GeLab. Não se preocupe. Eles estão, também, anexos esse treinamento em [gelab.zip](#). Extraia-o, copie e cole-o na pasta modules do e-Campus. Execute o `scriptlaboratorio.sql` em sua base local do sistema e insira a linha abaixo no método `montarPainel(){}`  de `frmPerfil.class` do módulo `common`.

```
$cmPanel->AddUserAction('LAB_GESTAO', A_ACCESS, 'Gest&atilde;o de Laborat&oacute;rios', $ui->GetImage('gel
```

## Criação da tabela na base de dados

```
create table lab_laboratorio
(
    idlaboratorio    serial not null
        constraint lab_laboratorio_pk
            primary key,
    nome             varchar(255)          not null
);

comment on table lab_laboratorio is 'Tabela que armazena os dados do laboratorio';

comment on column lab_laboratorio.idlaboratorio is 'Chave primaria';

comment on column lab_laboratorio.nome is 'Nome do laboratorio';

alter table lab_laboratorio
    owner to ecampus;
```

```
create unique index lab_laboratorio_nome_uindex
on lab_laboratorio (nome);
```

```
drop table lab_laboratorio
```

# Classe de negócio

## Arquivo laboratorio.class:

```
<?php

/**
 * Classe que representa um laboratório
 * Class BusinessGelabLaboratorio
 */
class BusinessGelabLaboratorio extends MBusiness
{
    public $idLaboratorio;
    public $nome;

    /**
     * BusinessGelabLaboratorio constructor.
     * @param $data
     */
    public function __construct($data = null)
    {
        parent::__construct('ufjf', $data);
    }

    /**
     * Método que pesquisa o laboratório de acordo com seu Id
     * @param mixed $idLaboratorio
     * @return mixed
     */
    function getById($idLaboratorio)
    {
        $this->idLaboratorio = $idLaboratorio;
        $this->retrieve();
    }
}
```

```
        return $this;
    }

    /**
     * @return mixed
     */
    public function getIdLaboratorio()
    {
        return $this->idLaboratorio;
    }

    /**
     * @param mixed $idLaboratorio
     */
    public function setIdLaboratorio($idLaboratorio)
    {
        $this->idLaboratorio = $idLaboratorio;
    }

    /**
     * @return mixed
     */
    public function getNome()
    {
        return $this->nome;
    }

    /**
     * @param mixed $nome
     */
    public function setNome($nome)
    {
        $this->nome = $nome;
    }

    /**
     * Método save que salva um objeto no banco de dados
     * @access public
     */
    public function save()
```

```

{
    $this->saveLog();
}

/**
 * Método update que atualiza um objeto no banco de dados
 * @access public
 */
public function update()
{
    $this->updateLog();
}

/**
 * Método delete que apaga um objeto no banco de dados
 * @access public
 */
public function delete()
{
    $this->deleteLog();
}

/**
 * Método que retorna laboratórios a partir do nome
 * @param string $busca
 * @return mixed
 */
public function pesquisarLaboratorioPorNome($busca = "")
{
    $criteria = $this->getCriteria();
    $criteria->addColumnAttribute('idLaboratorio');
    $criteria->addColumnAttribute('nome');

    $criteria->addCriteria('nome', 'ilike', "%$busca%");

    $criteria->addOrderAttribute('nome', 'DESC');

    return $criteria->retrieveAsQuery();
}

```

```
}
```

## Mapeamento classe x base de dados

### Arquivo laboratorio.xml:

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<map>
  <moduleName>gelab</moduleName>
  <className>laboratorio</className>
  <tableName>lab_laboratorio</tableName>
  <databaseName>ufjf</databaseName>
  <attribute>
    <attributeName>idLaboratorio</attributeName>
    <columnName>idlaboratorio</columnName>
    <idgenerator>lab_laboratorio_idlaboratorio_seq</idgenerator>
    <key>primary</key>
  </attribute>
  <attribute>
    <attributeName>nome</attributeName>
    <columnName>nome</columnName>
  </attribute>
</map>
```

## Formulários

### Arquivo frmLaboratorioFind.class:

```
<?php

/**
 * Classe que implementa a lógica de renderização na pesquisa de laboratórios
 * Class frmLaboratorioFind
 */
class frmLaboratorioFind extends MForm
{
    private $acao; // Ação atual (por exemplo, 'find' para pesquisa)
```

```
// Construtor da classe, inicializa variáveis e configura o formulário
function __construct($acao)
{
    $this->acao = $acao; // Define a ação passada como parâmetro

    // Texto de cabeçalho do formulário
    $texto = 'Laboratório';

    parent::__construct($texto); // Chama o construtor da classe pai

    $this->eventHandler(); // Chama o manipulador de eventos
}

// Cria os campos do formulário
function createFields()
{
    if ($this->acao == 'find') { // Se a ação for 'find', cria o grid de laboratórios
        $this->getGridLaboratorio();
    }
}

// Cria o grid de laboratórios
private function getGridLaboratorio()
{
    global $MIOLO; //Define o uso de uma variável global instanciada fora do escopo do método
    $module = $MIOLO->getCurrentModule(); // Obtém o módulo atual

    $pesquisarButton = new MButton('btnPesquisar', 'Pesquisar'); // Botão de pesquisa
    $novoLaboratorio = new MButton('btnNovo', 'Cadastrar', $MIOLO->getActionURL($module,
'main:laboratorio:dados')); // Botão para cadastrar novo laboratório

    $ui = $MIOLO->getUI(); // Obtém a interface do usuário
    $grid = $ui->getGrid($module, 'gridLaboratorio', null, 'laboratorio'); // Cria o grid de laboratórios

    $bntVoltar = new MButtonBack('bntVoltar', 'Voltar'); // Botão de voltar

    // Define os controles do formulário
    $fields = array(
        array(
```

```

        new MTextField('txtProcurarLaboratorio', '', 'Pesquisar Laborat&oacute;rio', '50'), // Campo de texto
        para pesquisar laborat&otilde;rio
        $pesquisarButton,
        $novoLaboratorio,
    ),
    $grid, // Adiciona o grid ao formul&eacute;rio
    $bntVoltar // Adiciona o bot&atilde;o de voltar ao formul&eacute;rio
);

$this->setFields($fields); // Define os campos do formul&eacute;rio
$this->defaultButton = false; // Define que n&ao; h&aa; bot&atilde;o padr&ao;
}

// Fun&ccedil;~o chamada ao clicar no bot&atilde;o 'Voltar'
function bntVoltar_click()
{
    global $MIOLO; //Define o uso de uma vari&acirc;vel global instanciada fora do escopo do m&eacute;todo
    $module = $MIOLO->getCurrentModule(); // Obt&eacute;m o m&odulo; atual
    // Redireciona para a URL principal do m&odulo;
    $this->page->redirect($this->manager->GetActionURL($module, 'main'));
}
}

```

### Arquivo frmLaboratorio.class:

```

<?php

/**
 * Classe que implementa a l&ogica de renderiza&ccedil;~o no gerenciamento de laborat&otilde;rios
 * Class frmLaboratorio
 */
class frmLaboratorio extends MForm
{
    private $objLaboratorio; // Objeto para gerenciar as opera&ccedil;~es do laborat&otilde;rio
    private $idLaboratorio; // ID do laborat&otilde;rio atual
    private $acao; // A&ccedil;~o atual (cadastrar, editar, visualizar, apagar)
}

```

```

/**
 * Construtor da classe
 * @param string $acao A ação a ser executada
 */
function __construct($acao)
{
    $this->acao = $acao;

    // Obtém o valor do campo 'item' do formulário, que é o ID do laboratório
    $this->idLaboratorio = $this->getFormValue('item');

    // Define o texto do título do formulário com base na ação
    $texto = 'Cadastrar Laboratório';
    if ($this->idLaboratorio != NULL) {
        $texto = 'Editar Laboratório';
    }
    if ($this->acao == 'view') {
        $texto = 'Visualizar Laboratório';
    }
    if ($this->acao == 'del') {
        $texto = 'Apagar Laboratório';
    }

    // Chama o construtor da classe pai (MForm) com o texto do título
    parent::__construct($texto);

    // Chama o manipulador de eventos
    $this->eventHandler();
}

/**
 * Cria os campos do formulário
 */
function createFields()
{
    global $MIOLO; //Define o uso de uma variável global instanciada fora do escopo do método

    // Obtém o objeto de negócios do laboratório

```



```

$this->objLaboratorio = $MIOLO->getBusiness('gelab', 'laboratorio', $this->idLaboratorio);

// Obtém o módulo atual
$module = $MIOLO->getCurrentModule();

// Cria a descrição da tela
$content = array(new MLabel ('<p>Esta tela refere-se ao cadastro de Laborat&oacute;rios.</p>'));

// Define os campos do formulário
$fields = array(
    !$this->idLaboratorio ? (new MThemeBox ('Descri&ccedil;&atilde;o', $content)) : null,
    new MSpacer('5px'),
    new MTextField('nome', $this->objLaboratorio->getNome(), 'Nome', 100),
    new MSpacer('15px'),
);

// Define os campos no formulário
$this->setFields($fields);

// Define os validadores dos campos
$validators = array(
    new MRequiredValidator('nome'),
);
$this->SetValidators($validators);

// Define os botões do formulário
$buttons = array(
    new MButton('btnConfirmarSalvar', 'Salvar Laborat&oacute;rio', '', '', 'fa-floppy-o'),
    new MButtonBack('btnVoltar', 'Voltar', $MIOLO->GetActionURL($module, 'main:laboratorio:find')),
);
$this->SetButtons($buttons);

// Define as ações baseadas na ação atual
$this->acoes();
}

/**
 * Método que solicita ao usuário a confirmação do cadastro de um laboratório
 */
public function btnConfirmarSalvar_click()

```

```

{
    global $MIOLO; //Define o uso de uma variável global instanciada fora do escopo do método
    // Obtém o módulo atual
    $module = $MIOLO->getCurrentModule();

    // if(!$this->GetFieldValue('nome')){
    // $MIOLO->Error("Nome não informado!",
    // $MIOLO->getActionURL($module, 'main:laboratorio:dados'));
    //}

    // Obtém os dados do formulário
    $data = $this->getData();
    $this->idLaboratorio = $this->getFormValue('item');

    // Define as URLs para as ações de sim e não na confirmação
    $sim = $MIOLO->getActionURL($module, "", $this->idLaboratorio, array('event' => 'salvarLaboratorio',
'nome' => $data->nome));
    $nao = $MIOLO->getActionURL($module, 'main:laboratorio:find');

    // Define a mensagem de confirmação
    $mensagem = "Voc&ecirc; confirma o cadastro?";
    if ($this->idLaboratorio) {
        $mensagem = "Confirma a altera&ccedil;&atilde;o do laborat&oacute;rio?";
    }

    // Solicita a confirmação do usuário
    $MIOLO->question($mensagem, $sim, $nao);
}

/**
 * Método que persiste um laboratório
 * @throws EBusinessException
 */
function salvarLaboratorio()
{
    global $MIOLO; //Define o uso de uma variável global instanciada fora do escopo do método
    // Obtém os dados do formulário
    $data = $this->getData();
    $this->objLaboratorio->setData($data);
}

```

```

// Obtém o módulo atual
$module = $MIOLO->getCurrentModule();
$msg = '';

try {
    // Inicia uma transação e salva ou atualiza o laboratório
    if (!$this->idLaboratorio) {
        $this->objLaboratorio->beginTransaction();
        $this->objLaboratorio->save();
        $this->objLaboratorio->endTransaction();
        $msg = 'Laboratório criado com sucesso!';
    } else {
        $this->objLaboratorio->update();
        $msg = 'Laboratório alterado com sucesso!';
        $this->objLaboratorio->endTransaction();
    }
} catch (Exception $e) {
    // Em caso de erro, faz rollback na transação e exibe uma mensagem de erro
    $this->objLaboratorio->getTransaction()->rollback();
    $MIOLO->Error("Erro ao cadastrar o laboratório.<br>" . $e->getMessage(), $MIOLO-
>getActionURL($module, 'main:laboratorio:find'));
}

// Exibe uma mensagem de informação sobre o sucesso da operação
$MIOLO->information($msg, $MIOLO->GetActionURL('gelab', 'main:laboratorio:find'), '');
}

/**
 * Método que solicita ao usuário a confirmação da remoção de um laboratório
 */
public function confirmacaoRemocao()
{
    global $MIOLO; //Define o uso de uma variável global instanciada fora do escopo do método
    // Obtém o módulo atual
    $module = $MIOLO->getCurrentModule();

    // Obtém o valor do campo 'idLaboratorio' do formulário
    $this->idLaboratorio = $this->getFormValue('idLaboratorio');

```

```

// Obtém o objeto de negócios do laboratório
$this->objLaboratorio = $MIOLO->getBusiness($module, 'laboratorio', $this->idLaboratorio);

// Define as URLs para as ações de sim e não na confirmação
$sim = $MIOLO->getActionURL($module, 'main:laboratorio:excluir', $this->idLaboratorio, array('event' =>
'removeLaboratorio'));
$nao = $MIOLO->getActionURL($module, 'main:laboratorio:find');

// Solicita a confirmação do usuário
$MIOLO->question("Você confirma a exclusão?", $sim, $nao);
}

/**
 * Método que realiza a remoção do laboratório
 */
public function removeLaboratorio()
{

    global $MIOLO; //Define o uso de uma variável global instanciada fora do escopo do método
    // Obtém o módulo atual
    $module = $MIOLO->getCurrentModule();

    // Obtém o valor do campo 'item' do formulário
    $idLaboratorio = $this->getFormValue('item');

    // Obtém o objeto de negócios do laboratório e carrega os dados pelo ID
    $this->objLaboratorio = $MIOLO->getBusiness($module, 'laboratorio');
    $this->objLaboratorio->getById($idLaboratorio);

    try {
        // Inicia uma transação e remove o laboratório
        $this->objLaboratorio->beginTransaction();
        $this->objLaboratorio->delete();
        $this->objLaboratorio->endTransaction();

        // Exibe uma mensagem de sucesso
        $MIOLO->Information('Laboratório excluído com sucesso!', $MIOLO->getActionURL($module, 'main:laboratorio:find'));
    } catch (Exception $e) {

```

```

        // Em caso de erro, faz rollback na transação e exibe uma mensagem de erro
        $this->objLaboratorio->getTransaction()->rollback();

        $MIOLO->Error("Erro ao excluir o laborat&oacute;rio.<br>" . $e->getMessage(), $MIOLO->getActionURL($module, 'main:laboratorio:find'));
    }
}

/**
 * Método que controla a exibição de campos quando a tela é apenas para exibição de dados de um
laboratório
 */
function acoes()
{
    if ($this->acao == 'view') {
        // Define o campo 'nome' como somente leitura
        $this->SetFieldAttr('nome', 'readonly', true);

        // Define a visibilidade dos botões
        $this->SetButtonAttr('btnVoltar', 'visible', true);
        $this->SetButtonAttr('btnPost', 'visible', false);
        $this->SetButtonAttr('btnConfirmarSalvar', 'visible', false);
    }
}
}

```

## Grid

### Arquivo gridLaboratorio.class

```

<?php

/**
 * Class gridLaboratorio
 */
class gridLaboratorio extends MDataGrid
{
    private $idLaboratorio; // ID do laboratório

    // Construtor da classe, inicializa variáveis e configura o grid de dados

```

```

public function __construct()
{
    global $MIOLO; //Define o uso de uma variável global instanciada fora do escopo do método
    $module = $MIOLO->getCurrentModule(); // Obtém o módulo atual

    $objLaboratorio = $MIOLO->getBusiness($module, 'laboratorio'); // Inicializa o objeto do laboratório

    $filter = MForm::getFormValue('txtProcurarLaboratorio'); // Obtém o valor do filtro do formulário

    $query = $objLaboratorio->pesquisarLaboratorioPorNome($filter); // Pesquisa laboratórios pelo nome

    $this->idLaboratorio = $query->getColumnNumber('idLaboratorio'); // Obtém o número da coluna
'idLaboratorio'

    // Define URLs para as ações de visualizar, editar e remover
    $href_view = $MIOLO->getActionURL('gelab', 'main:laboratorio:visualizar', "%{$this->idLaboratorio}%");
    $href_edit = $MIOLO->getActionURL($module, 'main:laboratorio:dados', "%{$this->idLaboratorio}%");
    $href_remove = $MIOLO->getActionURL($module, 'main:laboratorio:excluir', "",
        array('event' => 'confirmacaoRemocao', 'idLaboratorio' => "%{$this->idLaboratorio}%"));

    // Adiciona ícones de ação o grid de dados
    $this->addActionIcon('', 'select.gif', $href_view);
    $this->addActionUpdate($href_edit);
    $this->addActionDelete($href_remove);

    // Define as colunas no grid de dados
    $columns = array(
        new MDataGridColumn('nome', 'Nome', 'left', '', '50%'),
    );

    // Define a URL do grid de dados
    $hrefGrid = $MIOLO->getActionURL($module, $MIOLO->getCurrentAction());

    // Chama o construtor da classe pai
    parent::__construct($query, $columns, $hrefGrid, 50);

    // Define o título do grid de dados
    $this->SetTitle('Laborat&ocirc;rios');
}
}

```

# Handlers

## Arquivo handler.inc

```
<?php
class HandlerGeLab extends MHandler
{
    function init()
    {
        parent::init();

    }
}
```

## Arquivo main.inc:

```
<?php

// Obtém as permissões do gerenciador de módulos
$perms = $MIOLO->getPerms();

// Verifica se o usuário tem acesso ao módulo 'LAB_GESTAO' com a permissão 'A_ACCESS'.
// Se o acesso não for permitido, lança uma exceção ou realiza alguma ação conforme o contexto
$perms->checkAccess('LAB_GESTAO', A_ACCESS, true);

// Limpa o conteúdo do tema atual
$theme->ClearContent();

// Obtém a interface do usuário (UI) do gerenciador de módulos
$ui = $MIOLO->getUI();

// Adiciona uma opção ao navbar (barra de navegação) para 'Gestão de Laboratórios'
$navbar->AddOption('Gestão de Laboratórios', $MIOLO->getCurrentModule(), 'main');

// Obtém a URL de ação de fechar (não está claro o que exatamente está sendo fechado aqui)
```

```

$close = $MIOLO->getActionURL($MIOLO->getCurrentModule(), 'main');

// Cria um painel de ação (MActionPanel) com o ID 'pnaGelab', título 'Painel', sem descrição (''), e URL de fechar
determinada anteriormente
$panel = new MActionPanel('pnaGelab', 'Painel', '', $close, null);

// Adiciona uma ação de usuário ao painel com permissão 'LAB_GESTAO' e 'A_ACCESS', exibindo o nome
'Laboratório' e uma imagem padrão (provavelmente um ícone)
$panel->AddUserAction('LAB_GESTAO', A_ACCESS, 'Laborat&oacute;rio', $ui->GetImage('', 'default.png'),
$MIOLO->getCurrentModule(), 'main:laboratorio:find');

// Obtém o próximo manipulador de ação do contexto atual
$nextHandler = $context->shiftAction();

// Invoca o manipulador de ação obtido, passando o módulo atual e o manipulador de ação
$handled = $MIOLO->invokeHandler($MIOLO->getCurrentModule(), $nextHandler);

// Se o manipulador de ação não foi tratado (handled), insere o painel no conteúdo do tema
if (!$handled) {
    $theme->insertContent($panel);
}

// Inclui o arquivo main_menu.inc do diretório especificado pelo gerenciador de módulos
include_once($this->manager->GetConf('home.modules') . '/main_menu.inc');

```

### Arquivo laboratorio.inc:

```

<?php

// Obtém as permissões do gerenciador de módulos
$perms = $MIOLO->getPerms();

// Verifica se o usuário tem acesso ao módulo 'LAB_GESTAO' com a permissão 'A_ACCESS'.
// Se o acesso não for permitido, lança uma exceção com a mensagem de erro
if (!$perms->CheckAccess('LAB_GESTAO', A_ACCESS, FALSE)) {
    $MIOLO->exception("Voc&ecirc; n&atilde;o tem acesso a esse conte&uacute;do!");
}

// Limpa o conteúdo do tema atual
$theme->ClearContent();

```



```

// Obtém a interface do usuário (UI) do gerenciador de módulos
$ui = $MIOLO->GetUI();

// Adiciona uma opção ao navbar (barra de navegação) para 'Laboratório'
$navbar->addOption('Laboratório', $MIOLO->getCurrentModule(), 'laboratorio:find');

// Obtém a URL de ação de fechar (não está claro o que exatamente está sendo fechado aqui)
$close = $MIOLO->GetActionURL($MIOLO->getCurrentModule(), 'main');

// Cria um painel de ação (MActionPanel) com o ID 'panel', título 'Laboratório', sem descrição (''), e URL de fechar
determinada anteriormente
$panel = new MActionPanel('panel', 'Laboratório', '', $close);

// Define o tamanho do controle no painel (provavelmente não especificado aqui)
$panel->setControlSize("", "");

// Adiciona uma ação de usuário ao painel com permissão 'LAB_GESTAO' e 'A_ACCESS', exibindo o nome
'Gerenciar Laboratório'
// A imagem associada é obtida pela função getImage do UI
$panel->addUserAction('LAB_GESTAO', A_ACCESS, 'Gerenciar Laboratório',
    $ui->getImage($MIOLO->getCurrentModule(), 'default.png'), $MIOLO->getCurrentModule(),
    'main:laboratorio:find');

// Obtém a próxima ação do contexto atual
$action = $context->ShiftAction();

// Invoca o manipulador de ação do módulo atual com base na ação obtida
$handled = $MIOLO->InvokeHandler($MIOLO->getCurrentModule(), "laboratorio/$action");

// Se a ação não foi tratada (handled), insere o painel no conteúdo do tema
if (!$handled) {
    $theme->insertContent($panel);
}

```

## Arquivo find.inc:

```

<?php

$perms = $MIOLO->getPerms(); // Obtém as permissões do usuário

```

```
// Verifica se o usuário tem acesso ao módulo de gestão de laboratórios
if (!$perms->CheckAccess('LAB_GESTAO', A_ACCESS, FALSE)) {
    // Se não tiver acesso, lança uma exceção com a mensagem apropriada
    $MIOLO->exception("Você não tem acesso a este conteúdo!");
}

$ui = $MIOLO->GetUI(); // Obtém a interface do usuário
$form = $ui->GetForm($MIOLO->getCurrentModule(), 'frmLaboratorioFind', 'find', 'laboratorio'); // Obtém o formulário 'frmLaboratorioFind'
$form->setTitle("Gerenciar Laboratórios"); // Define o título do formulário

$theme->InsertContent($form); // Insere o formulário no tema
```

## Arquivo dados.inc:

```
<?php

// Obtém as permissões do gerenciador de módulos
$perms = $MIOLO->getPerms();

// Verifica se o usuário tem acesso ao módulo 'LAB_GESTAO' com a permissão 'A_ACCESS'.
// Se o acesso não for permitido, lança uma exceção com a mensagem de erro
if (!$perms->CheckAccess('LAB_GESTAO', A_ACCESS, FALSE)) {
    $MIOLO->exception("Você não tem acesso a este conteúdo!");
}

// Limpa o conteúdo do tema atual
$theme->ClearContent();

// Obtém a interface do usuário (UI) do gerenciador de módulos
$ui = $MIOLO->GetUI();

// Obtém o valor do campo 'item' do formulário (se existir), que parece ser o ID do laboratório
$idLaboratorio = Form::getFormValue('item');

// Determina o nome e a ação com base na existência do ID do laboratório
if ($idLaboratorio) {
    $nome = 'Editar Laboratorio';
    $acao = 'upd'; // Ação de atualizar
```

```

} else {
    $nome = 'Cadastrar Laboratorio';
    $acao = 'ins'; // Ação de inserir
}

// Adiciona opções ao navbar (barra de navegação)
//$navbar->AddOption('Gerenciar Laborat&ocirc;rio', $MIOLO->getCurrentModule(), 'main:laboratorio:find');
$navbar->AddOption($nome, $MIOLO->getCurrentModule(), 'main:laboratorio:dados', $idLaboratorio);

// Obtém o formulário 'frmLaboratorio' do módulo atual, especificando a ação determinada ('ins' ou 'upd') e o
nome do módulo 'laboratorio'
$form = $ui->GetForm($MIOLO->getCurrentModule(), 'frmLaboratorio', $acao, 'laboratorio');

// Insere o formulário obtido no conteúdo do tema
$theme->InsertContent($form);

```

### Arquivo visualizar.inc:

```

<?php

$perms = $MIOLO->getPerms(); // Obtém as permissões do usuário

// Verifica se o usuário tem acesso ao módulo de gestão de laboratórios
if (!$perms->CheckAccess('LAB_GESTAO', A_ACCESS, FALSE)) {
    // Se não tiver acesso, lança uma exceção com a mensagem apropriada
    $MIOLO->exception("Voc&ecirc; n&atilde;o tem acesso a esse conte&uacute;do!");
}

$theme->ClearContent(); // Limpa o conteúdo atual do tema
$ui = $MIOLO->GetUI(); // Obtém a interface do usuário
$idLaboratorio = Form::getFormValue('item'); // Obtém o valor do item (id do laboratório) do formulário

$nome = 'Visualizar Laborat&ocirc;rio'; // Define o nome do laboratório para visualização

// Adiciona uma opção ao navbar para visualizar o laboratório específico
$navbar->AddOption($nome, $MIOLO->getCurrentModule(), 'main:laboratorio:visualizar', $idLaboratorio);

// Obtém o formulário 'frmLaboratorio' na ação de visualização (view) do laboratório

```

```
$form = $ui->GetForm($MIOLO->getCurrentModule(), 'frmLaboratorio', 'view', 'laboratorio');
```

```
$theme->InsertContent($form); // Insere o formulário no tema
```

### Arquivo **excluir.inc**:

```
<?php
```

```
// Obtém as permissões do gerenciador de módulos
```

```
$perms = $MIOLO->getPerms();
```

```
// Verifica se o usuário tem acesso ao módulo 'LAB_GESTAO' com a permissão 'A_ACCESS'.
```

```
// Se o acesso não for permitido, lança uma exceção com a mensagem de erro
```

```
if (!$perms->CheckAccess('LAB_GESTAO', A_ACCESS, FALSE)) {
```

```
    $MIOLO->exception("Você não tem acesso a este conteúdo!");
```

```
}
```

```
// Limpa o conteúdo do tema atual
```

```
$theme->ClearContent();
```

```
// Obtém a interface do usuário (UI) do gerenciador de módulos
```

```
$ui = $MIOLO->GetUI();
```

```
// Obtém o formulário 'frmLaboratorio' do módulo atual, especificando a ação 'del' e o nome do módulo
```

```
'laboratorio'
```

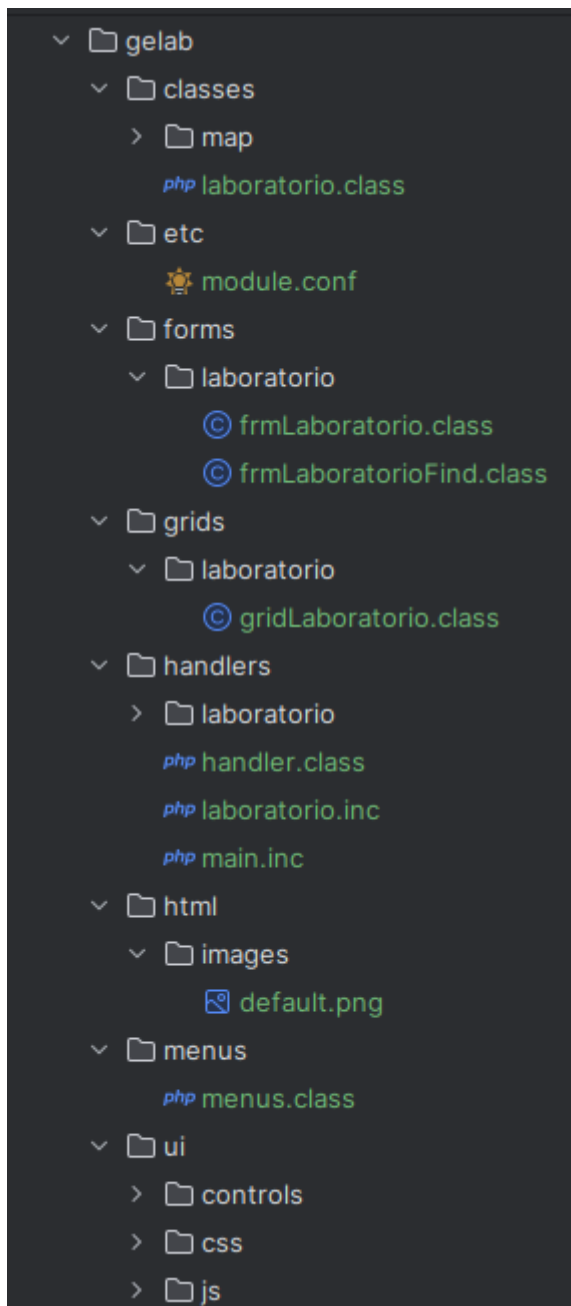
```
$form = $ui->GetForm($MIOLO->getCurrentModule(), 'frmLaboratorio', 'del', 'laboratorio');
```

```
// Insere o formulário obtido no conteúdo do tema
```

```
$theme->InsertContent($form);
```

## Estrutura final

Ao final, a estrutura do módulo é apresentada abaixo.



## Estrutura do módulo Gestão de Laboratórios

Revision #15

Created 18 July 2024 17:49:19 by Clayton Brás de Souza Lopes

Updated 19 July 2024 17:31:37 by Clayton Brás de Souza Lopes