Adianti Framework

AdiantiApplicationConfig

Application config

Methods

public static function load($config)
Load configuration from array

public static function get()
Export configuration

AdiantiApplicationLoader

Application loader

Methods

public static function autoload($class)

AdiantiAutocompleteService

Autocomplete backend

Methods

public static function onSearch($param = null)
Search by the given word inside a model

AdiantiClassMap

Class map

Methods

public static function getMap()

public static function getAllowedClasses()
Return classes allowed to be directly executed

public static function getInternalClasses()
Return internal classes

public static function getAliases()
Aliases for backward compatibility

AdiantiCoreApplication

Basic structure to run a web application

Methods

public static function run($debug = FALSE)

public static function setRouter(Callable $callback)
Set router callback

public static function getRouter()
Get router callback

public static function executeMethod($class, $method = NULL, $parameters = NULL)
Execute a specific method of a class with parameters

$class : class name
$method : method name
$parameters : array of parameters

public static function gotoPage($class, $method = NULL, $parameters = NULL, $callback = NULL)
Goto a page

$class : class name
$method : method name
$parameters : array of parameters

public static function loadPage($class, $method = NULL, $parameters = NULL)
Load a page

$class : class name
$method : method name
$parameters : array of parameters

public static function postData($formName, $class, $method = NULL, $parameters = NULL)
Post data

$class : class name
$method : method name
$parameters : array of parameters

public static function buildHttpQuery($class, $method = NULL, $parameters = NULL)
Build HTTP Query

$class : class name
$method : method name
$parameters : array of parameters

public static function reload()
Reload application

public static function registerPage($page)
Register URL

$page : URL to be registered

public static function errorHandler($errno, $errstr, $errfile, $errline)
Handle Catchable Errors

AdiantiCoreLoader

Framework class autoloader

Methods

public static function loadClassMap()
Load the class map

public static function setClassPath($class, $path)
Define the class path

$class : Class name
$path : Class path

public static function autoload($className)
Core autloader

$className : Class name

public static function legacyAutoload($class)

autoloader

$class : classname

public static function globalScope($class)
make a class global

AdiantiCoreTranslator

Framework translation class for internal messages

Methods

public static function getInstance()
Returns the singleton instance

public static function setLanguage($lang)
Define the target language

$lang : Target language index

public static function getLanguage()
Returns the target language

public static function translate($word, $param1 = NULL, $param2 = NULL, $param3 = NULL, $param4 = NULL)
Translate a word to the target language

$word : Word to be translated

AdiantiFileSaveTrait

Methods
public function saveFile($input_name, $target_path, $object)
Save file

$input_name : input field name
$target_path : target file path
$object : Active Record

AdiantiMasterDetailTrait

Methods
public function storeItems($model, $foreign_key, $master_object, $detail_id, Callable $transformer = null)
Store an item from details session into database

$model : Model class name
$foreign_key : Detail foreign key name
$master_object : Master object
$detail_id : Detail key in session
$transformer : Function to be applied over the objects

public function loadItems($model, $foreign_key, $master_object, $detail_id, Callable $transformer = null)
Load items for detail into session

$model : Model class name
$foreign_key : Detail foreign key name
$master_object : Master object
$detail_id : Detail key in session
$transformer : Function to be applied over the objects

AdiantiMultiFileSaveTrait

Methods
public function saveFiles($input_name, $foreign_key, $file_field, $target_path, $object, $modelFiles)

Save files

$input_name : input field name
$foreign_key :
$field_field :
$target_path :
$object :
$modelFiles :

AdiantiMultiSearchService

MultiSearch backend

Methods

public static function onSearch($param = null)
Search by the given word inside a model

AdiantiPDFDesigner extends FPDF

FPDF Adapter that parses XML files from Adianti Framework

Methods

public function __construct($orientation = 'P', $format = 'a4', $unit = 'pt')
Constructor method

$orientation : Page orientation
$format : Page format

public function fromXml($filename)
Load designed elements from XML

$filename : XML file location

public function loadElements($elements)
Load Elements

$elements : Elements (shapes) to load

public function gotoAnchorXY($anchor_name)
Put the cursor at the anchor XY position

$anchor_name : Anchor name

public function gotoAnchorX($anchor_name)
Put the cursor at the anchor X position

$anchor_name : Anchor name

public function gotoAnchorY($anchor_name)
Put the cursor at the anchor Y position

$anchor_name : Anchor name

public function writeAtAnchor($anchor_name, $text)
Write at the anchor position

$anchor_name : Anchor name
$text : Text to write

public function replace($mark, $text)
Replace a piece of {text}

$mark : piece to be replaced
$text : new content

public function generate()
Generate one PDF page with the parsed elements

public function ellipse($x, $y, $rx, $ry, $style = 'D')
Draws an ellipse

$x : X
$y : Y
$rx : X Ray
$ry : Y Ray
$style : Line Style

public function writeHTML($x, $y, $html)
Write HTML

$x : X
$y : Y
$html : HTML

public function openTag($tag,$attr, $x)
Open Html TAG

$tag : Tag
$attr : Tag attributes
$x : X position

public function closeTag($tag)
Close Html TAG

$tag : Tag

public function setStyle($tag,$enable)
Set Style

$tag : Tag
$enable : Enable

public function putLink($URL,$txt)
Put link

$URL :
$txt :

public function setLocale()
Change PDF locale

public function unsetLocale()
Back to the old locale

public function setFontColorRGB($color)
Changes the color

$color : Color in RGB

public function setFillColorRGB($color)
Changes the fill color

$color : Color in RGB

public function setDrawColorRGB($color)
Changes the draw color

$color : Color in RGB

public function save($output)
Saves the PDF

$output : Output path

AdiantiRecordService

Record rest service

Methods

public function load($param)
Find a Active Record and returns it

$param : HTTP parameter

public function delete($param)
Delete an Active Record object from the database

[$id] : HTTP parameter

public function store($param)
Store the objects into the database

$param : HTTP parameter

public function loadAll($param)
List the Active Records by the filter

$param : HTTP parameter

public function deleteAll($param)
Delete the Active Records by the filter

$param : HTTP parameter

AdiantiStandardControlTrait

Methods
public function setDatabase($database)
method setDatabase() Define the database

public function setActiveRecord($activeRecord)
method setActiveRecord() Define wich Active Record class will be used

AdiantiStandardFormListTrait

Methods
public function setLimit($limit)
method setLimit() Define the record limit

public function setDefaultOrder($order, $direction = 'asc')
Define the default order

$order : The order field
$directiont : the order direction (asc, desc)

public function setCriteria($criteria)
method setCriteria() Define the criteria

public function setTransformer($callback)
Define a callback method to transform objects before load them into datagrid

public function onReload($param = NULL)
method onReload() Load the datagrid with the database objects

public function onSave()
method onSave() Executed whenever the user clicks at the save button

public function onDelete($param)
method onDelete() executed whenever the user clicks at the delete button Ask if the user really wants to delete the record

public function Delete($param)
method Delete() Delete a record

public function onClear($param)
Clear form

public function onEdit($param)
method onEdit() Executed whenever the user clicks at the edit button da datagrid

public function show()
Shows the page

public function setDatabase($database)
method setDatabase() Define the database

public function setActiveRecord($activeRecord)
method setActiveRecord() Define wich Active Record class will be used

AdiantiStandardFormTrait

Methods

public function onSave()
method onSave() Executed whenever the user clicks at the save button

public function onClear($param)
Clear form

public function onEdit($param)
method onEdit() Executed whenever the user clicks at the edit button da datagrid
$param : An array containing the GET ($_GET) parameters

public function setDatabase($database)
method setDatabase() Define the database

public function setActiveRecord($activeRecord)
method setActiveRecord() Define wich Active Record class will be used

AdiantiStandardListTrait

Methods

public function setLimit($limit)
method setLimit() Define the record limit

public function enableTotalRow()
Enable total row

public function setDefaultOrder($order, $direction = 'asc')
Define the default order
$order : The order field
$directiont : the order direction (asc, desc)

public function setFilterField($filterField)
method setFilterField() Define wich field will be used for filtering PS: Just for Backwards compatibility

public function setOperator($operator)
method setOperator() Define the filtering operator PS: Just for Backwards compatibility

public function addFilterField($filterField, $operator = 'like', $formFilter = NULL, $filterTransformer = NULL)
method addFilterField() Add a field that will be used for filtering
$filterField : Field name
$operator : Comparison operator

public function setCriteria($criteria)
method setCriteria() Define the criteria

public function setTransformer($callback)
Define a callback method to transform objects before load them into datagrid

public function onInlineEdit($param)
Inline record editing

$param : Array containing: key: object ID value field name: object attribute to be updated value: new attribute content

public function onSearch()
Register the filter in the session

public function onReload($param = NULL)
method onReload() Load the datagrid with the database objects

public function onDelete($param)
Ask before deletion

public function Delete($param)
Delete a record

public function onDeleteCollection($param)
Ask before delete record collection

public function deleteCollection($param)
method deleteCollection() Delete many records

public function show()
method show() Shows the page

public function setDatabase($database)
method setDatabase() Define the database

public function setActiveRecord($activeRecord)
method setActiveRecord() Define wich Active Record class will be used

AdiantiUIBuilder extends TPanel

Interface builder that takes a XML file save by Adianti Studio Designer and renders the form into the interface.

Methods

public function __construct($width, $height)
Class Constructor

$width : Panel width
$height : Panel height

public function getActions()
Return the found actions

public function parseFile($filename)
Parse XML form file

$filename : XML form file path

public function makeTLabel($properties)

public function makeTButton($properties)

public function makeTEntry($properties)

public function makeTSpinner($properties)

public function makeTSlider($properties)

public function makeTPassword($properties)

public function makeTDate($properties)

public function makeTFile($properties)

public function makeTColor($properties)

public function makeTSeekButton($properties)

public function makeTImage($properties)

public function makeTText($properties)

public function makeTCheckGroup($properties)

public function makeTDBCheckGroup($properties)

public function makeTRadioGroup($properties)

public function makeTDBRadioGroup($properties)

public function makeTCombo($properties)

public function makeTDBCombo($properties)

public function makeTSelect($properties)

public function makeTDBSelect($properties)

public function makeTSortList($properties)

public function makeTDBSortList($properties)

public function makeTMultiSearch($properties)

public function makeTDBMultiSearch($properties)

public function makeTNotebook($properties)

public function makeTFrame($properties)

public function makeTDataGrid($properties)

public function setController($object)
Defines the UI controller
$object : Controller Object

public function setForm($form)
Defines the Parent Form
$object : TForm

public function getFields()
Return the UI widgets (form fields)

public function getWidgets()
Return the parsed widgets

public function getWidget($name)
Return the widget by name
$name : Widget name

AdiantiUploaderService

File uploader listener

Methods

function show($param)

BootstrapDatagridWrapper

Bootstrap datagrid decorator for Adianti Framework

Methods

public function __construct(TDataGrid $datagrid)
Constructor method

function show()
Shows the DataGrid

public function enablePopover($title, $content)
Enable popover

$title : Title
$content : Content

public function makeScrollable()
Make the datagrid scrollable

public function setActionWidth($width)
Set the column action width

public function disableDefaultClick()
disable the default click action

function setHeight($height)
Define the Height

$height : An integer containing the height

public function addColumn(TDataGridColumn $object)
Add a Column to the DataGrid

$object : A TDataGridColumn object

public function getColumns()
Returns an array of TDataGridColumn

public function addAction(TDataGridAction $object)
Add an Action to the DataGrid

$object : A TDataGridAction object

public function addActionGroup(TDataGridActionGroup $object)
Add an Action Group to the DataGrid

$object : A TDataGridActionGroup object

public function getTotalColumns()
Returns the total columns

public function setGroupColumn($column, $mask)
Set the group column for break

function clear($preserveHeader = TRUE)
Clear the DataGrid contents

public function createModel($create_header = true)
Creates the DataGrid Structure

public function getHead()
Return thead

public function getBody()
Return tbody

public function insert($position, $content)
insert content

public function addItems($objects)
Add objects to the DataGrid

$objects : An array of Objects

public function addItem($object)
Add an object to the DataGrid

$object : An Active Record Object

public function getItems()
Return datagrid items

public function getRowIndex($attribute, $value)
Find the row index by object attribute

$attribute : Object attribute
$value : Object value

public function getRow($position)
Return the row by position

$position : Row position

public function getWidth()
Returns the DataGrid's width

function setPageNavigation($pageNavigation)
Assign a PageNavigation object

$pageNavigation : object

function getPageNavigation()
Return the assigned PageNavigation object

public function addQuickColumn($label, $name, $align = 'left', $size = 200, TAction $action = NULL, $param = NULL)
Add a column

$label : Field Label
$object : Field Object
$size : Field Size

public function addQuickAction($label, TDataGridAction $action, $field, $icon = NULL)
Add action to the datagrid

$label : Action Label
$action : TAction Object
$icon : Action Icon

BootstrapFormBuilder

Bootstrap form builder for Adianti Framework

Methods

public function __construct($name = 'my_form')
Constructor method

$name : form name

public function setFormTitle($title)
Add a form title

$title : Form title

public function setPadding($padding)
Set padding

$padding :

public function setCurrentPage($i)
Define the current page to be shown

$i : An integer representing the page number (start at 0)

public function setProperty($name, $value, $replace = TRUE)
Define a field property

$name : Property Name
$value : Property Value

public function setName($name)
Set form name

$name : Form name

public function getName()
Get form name

public function addField(AdiantiWidgetInterface $field)
Add form field

$field : Form field

public function delField(AdiantiWidgetInterface $field)
Del form field

$field : Form field

public function setFields($fields)
Set form fields

$fields : Array of Form fields

public function getField($name)
Return form field

$name : Field name

public function getFields()
Return form fields

public function clear($keepDefaults = FALSE)
Clear form

public function setData($object)
Set form data

$object : Data object

public function getData($class = 'StdClass')
Get form data

$class : Object type of return data

public function getActions()
Return form actions

public function validate()
Validate form data

public function appendPage($title)
Append a notebook page

$title : Tab title

public function addFields()
Add form fields

mixed : $fields,... Form fields

public function addContent()
Add a form content

mixed : $content,... Form content

public function validateInlineArguments($args, $method)
Validate argument type

$args : Array of arguments
$method : Generator method

public function addAction($label, TAction $action, $icon = 'fa:save')
Add a form action

$label : Button label
$action : Button action
$icon : Button icon

public function addHeaderAction($label, TAction $action, $icon = 'fa:save')
Add a form header action

$label : Button label
$action : Button action
$icon : Button icon

public function addButton($label, $action, $icon = 'fa:save')
Add a form button

$label : Button label
$action : JS Button action
$icon : Button icon

public function setColumnClasses($key, $classes)

public function show()
Render form

public static function wrapField($field, $display)
Create a field wrapper

public static function showField($form, $field)

public static function hideField($form, $field)

BootstrapFormWrapper

Bootstrap form decorator for Adianti Framework

Methods

public function __construct(TQuickForm $form, $class = 'form-horizontal')
Constructor method

public function getElement()
Return render element

public function setName($name)
Set form name

public function getName()

Get form name

public function addField(AdiantiWidgetInterface $field)
Add form field

public function delField(AdiantiWidgetInterface $field)
Del form field

public function setFields($fields)
Set form fields

public function getField($name)
Return form field

public function getFields()
Return form fields

public function clear()
Clear form

public function setData($object)
Set form data

public function getData($class = 'StdClass')
Get form data

public function validate()
Validate form data

public function show()
Shows the decorated form

public function getActionsContainer()
Returns the actions container

public function getTable()
Returns the inner table

public function setFieldsByRow($count)
Define the field quantity per row
$count : Field count

public function getFieldsByRow()
Return the fields by row count

public function getContainer()
Returns the form container

public function setFormTitle($title)
Add a form title
$title : Form title

public function getInputRows()
Returns the input groups

public function addQuickField($label, AdiantiWidgetInterface $object, $size = 200, TFieldValidator $validator = NULL, $label_size = NULL)
Add a form field
$label : Field Label
$object : Field Object
$size : Field Size

$validator : Field Validator

public function addQuickFields($label, $objects, $required = FALSE)
Add a form field

$label : Field Label
$objects : Array of Objects
$required : Boolean TRUE if required

public function addQuickAction($label, TAction $action, $icon = 'ico_save.png')
Add a form action

$label : Action Label
$action : TAction Object
$icon : Action Icon

public function addQuickButton($label, $action, $icon = 'ico_save.png')
Add a form button

$label : Action Label
$action : Javascript action
$icon : Action Icon

public function delActions()
Clear actions row

public function getActionButtons()
Return an array with action buttons

public function detachActionButtons()
Detach action buttons

public function addRow()
Add a row

public static function showField($form, $field)

public static function hideField($form, $field)

BootstrapNotebookWrapper

Bootstrap datagrid decorator for Adianti Framework

Methods

public function __construct(TNotebook $notebook)
Constructor method

public function setTabsDirection($direction, $divisions = null)
Set tabs direction

$direction : Tabs direction (left right)

public function show()
Show the notebook

public function setTabsVisibility($visible)
Define if the tabs will be visible or not

$visible : If the tabs will be visible

public function setTabsSensibility($sensibility)
Define the tabs click sensibility

$sensibility : If the tabs will be sensible to click

public function getId()
Returns the element ID

public function setSize($width, $height)
Set the notebook size

$width : *Notebook's width*
$height : *Notebook's height*

public function getSize()
Returns the frame size

public function setCurrentPage($i)
Define the current page to be shown

$i : *An integer representing the page number (start at 0)*

public function getCurrentPage()
Returns the current page

public function appendPage($title, $object)
Add a tab to the notebook

$title : *tab's title*
$object : *tab's content*

public function getPageCount()
Return the Page count

public function setTabAction(TAction $action)
Define the action for the Notebook tab

$action : *Action taken when the user clicks over Notebook tab (A TAction object)*

public function render()
Render the notebook

TAPCache
Adianti APC Record Cache

Methods

public static function enabled()
Returns if the service is active

public static function setValue($key, $value)
Store a variable in cache

$key : *Key*
$value : *Value*

public static function getValue($key)
Get a variable from cache

$key : *Key*

public static function delValue($key)
Delete a variable from cache

$key : *Key*

public static function clear()
Clear cache

TAction
Structure to encapsulate an action

Methods

public function __construct($action, $parameters = null)
Class Constructor

$action : Callback to be executed
$parameters : = array of parameters

public function toString()
Returns the action as a string

public function setParameter($param, $value)
Adds a parameter to the action

$param : = parameter name
$value : = parameter value

public function setParameters($parameters)
Set the parameters for the action

$parameters : = array of parameters

public function getParameter($param)
Returns a parameter

$param : = parameter name

public function getParameters()
Return the Action Parameters

public function getAction()
Returns the current calback

public function serialize($format_action = TRUE)
Converts the action into an URL

$format_action : = format action with document or javascript (ajax=no)

public function isStatic()
Returns if the action is static

TActionLink extends TTextDisplay

Action Link

Methods

public function __construct($value, TAction $action, $color = null, $size = null, $decoration = null, $icon = null)
Class Constructor

$value : text content
$action : TAction Object
$color : text color
$size : text size
$decoration : text decorations (b=bold, i=italic, u=underline)

TAdiantiCoreTranslator

Framework translation class for internal messages

Methods

public static function getInstance()
Returns the singleton instance

public static function setLanguage($lang)
Define the target language

$lang : Target language index

public static function getLanguage()
Returns the target language

public static function translate($word, $param1 = NULL, $param2 = NULL, $param3 = NULL, $param4 = NULL)
Translate a word to the target language

$word : Word to be translated

TAlert extends TElement

Alert

Methods

public function __construct($type, $message)
Class Constructor

$type : Type of the alert (success, info, warning, danger)
$message : Message to be shown

TBreadCrumb extends TElement

BreadCrumb

Methods

public function __construct()
Handle paths from a XML file

$xml_file : path for the file

public static function create($options, $home = true)
Static constructor

public function addHome()
Add the home icon

public function addItem($path, $last = FALSE)
Add an item

$path : Path to be shown
$last : If the item is the last one

public function select($path)
Mark one breadcrumb item as selected

public static function setHomeController($className)
Define the home controller

$class : Home controller class

TButton extends TField

Button Widget

Methods

public static function create($name, $callback, $label, $image)
Create a button with icon and action

public function addStyleClass($class)
Add CSS class

public function setAction(TAction $action, $label = NULL)
Define the action of the button

$action : TAction object

$label : Button's label

public function getAction()
Returns the buttona action

public function setImage($image)
Define the icon of the button

$image : image path

public function setLabel($label)
Define the label of the button

$label : button label

public function getLabel()
Returns the button label

public function setProperty($name, $value, $replace = TRUE)
Define a field property

$name : Property Name
$value : Property Value

public function getProperty($name)
Return field property

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public function show()
Show the widget at the screen

TCNPJValidator extends TFieldValidator

CNPJ validation (Valid only in Brazil)

Methods

public function validate($label, $value, $parameters = NULL)
Validate a given value

$label : Identifies the value to be validated in case of exception
$value : Value to be validated
$parameters : aditional parameters for validation

TCPFValidator extends TFieldValidator

CPF validation (Valid only in Brazil)

Methods

public function validate($label, $value, $parameters = NULL)
Validate a given value

$label : Identifies the value to be validated in case of exception
$value : Value to be validated
$parameters : aditional parameters for validation

TCalendar extends TElement

Calendar Widget

Methods

public function __construct()
Class Constructor

public function setSize($width, $height)
Define the calendar's size

$width : Window's width
$height : Window's height

public function setMonth($month)
Define the current month to display

$month : Month to display

public function setYear($year)
Define the current year to display

$year : Year to display

public function getMonth()
Return the current month

public function getYear()
Return the current year

public function setAction(TAction $action)
Define the action when click at some day

$action : TAction object

public function selectDays(array $days)
Select a collection of days

$days : Collection of days

public function show()
Show the calendar

TCheckButton extends TField

CheckButton widget

Methods

public function setIndexValue($index)
Define the index value for check button

public function show()
Shows the widget at the screen

TCheckGroup extends TField

A group of CheckButton's

Methods

public function __construct($name)
Class Constructor

$name : name of the field

public function checkAll()

Check all options

public function setLayout($dir)
Define the direction of the options
$direction : String (vertical, horizontal)

public function getLayout()
Get the direction (vertical or horizontal)

public function setBreakItems($breakItems)
Define after how much items, it will break

public function setUseButton()
Show as button

public function addItems($items)
Add items to the check group
$items : An indexed array containing the options

public function getItems()
Return the items

public function getButtons()
Return the option buttons

public function getLabels()
Return the option labels

public function setValueSeparator($sep)
Define the field's separator
$sep : A string containing the field's separator

public function setValue($value)
Define the field's value
$value : A string containing the field's value

public function getPostData()
Return the post data

public function setChangeAction(TAction $action)
Define the action to be executed when the user changes the combo
$action : TAction object

public function setChangeFunction($function)
Set change function

public static function enableField($form_name, $field)
Enable the field
$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field
$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
clear the field
$form_name : Form name
$field : Field name

public function show()

Shows the widget at the screen

TColor extends TEntry

Color Widget

Methods

public function __construct($name)
Class Constructor

$name : Name of the widget

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public function setChangeFunction($function)
Set change function

public function setChangeAction(TAction $action)
Define the action to be executed when the user changes the content

$action : TAction object

public function show()
Shows the widget at the screen

TCombo extends TField

ComboBox Widget

Methods

public function __construct($name)
Class Constructor

$name : widget's name

public function setBooleanMode()
Enable/disable boolean mode

public function setValue($value)
Define the field's value

$value : A string containing the field's value

public function getValue()
Returns the field's value

public function clear()
Clear combo

public function addItems($items)
Add items to the combo box

$items : An indexed array containing the combo options

public function getItems()
Return the combo items

public function enableSearch()
Enable search

public function getPostData()
Return the post data

public function setChangeAction(TAction $action)
Define the action to be executed when the user changes the combo

$action : TAction object

public function setChangeFunction($function)
Set change function

public static function reload($formname, $name, $items, $startEmpty = FALSE)
Reload combobox items after it is already shown

$formname : form name (used in gtk version)
$name : field name
$items : array with items
$startEmpty : if the combo will have an empty first item

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
Clear the field

$form_name : Form name
$field : Field name

public function setDefaultOption($option)
Define the combo default option value

$option : option value

public function renderItems()
Render items

public function show()
Shows the widget

TComboCombined extends TField

ComboBox Widget with an entry

Methods

public function __construct($name, $text_name)
Class Constructor

$name : widget's name
$text : widget's name

public function getTextName()
Returns the text widget's name

public function setTextName($name)
Define the text widget's name

$name : A string containing the text widget's name

public function addItems($items)
Add items to the combo box

$items : An indexed array containing the combo options

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
Clear the field

$form_name : Form name
$field : Field name

public function show()
Shows the widget

TConnection

Singleton manager for database connections

Methods

public static function open($database)
Opens a database connection

$database : Name of the database (an INI file).

public static function openArray($db)
Opens a database connection from array with db info

$db : Array with database info

public static function getDatabaseInfo($database)
Returns the database information as an array

$database : INI File

TCriteria extends TExpression

Provides an interface for filtering criteria definition

Methods

public function __construct()
Constructor Method

public static function create($simple_filters, $properties = null)
create criteria from array of filters

public function add(TExpression $expression, $operator = self::AND_OPERATOR)
Adds a new Expression to the Criteria

$expression : TExpression object
$operator : Logic Operator Constant

public function getPreparedVars()
Return the prepared vars

public function dump($prepared = FALSE)

Returns the final expression

$prepared : Return a prepared expression

public function setProperty($property, $value)
Define a Criteria property

$property : Name of the property (limit, offset, order, direction)
$value : Value for the property

public function resetProperties()
reset criteria properties

public function setProperties($properties)
Set properties form array

$properties : array of properties

public function getProperty($property)
Return a Criteria property

$property : Name of the property (LIMIT, OFFSET, ORDER)

TDBCheckGroup extends TCheckGroup

Database CheckBox Widget

Methods

public function __construct($name, $database, $model, $key, $value, $ordercolumn = NULL, TCriteria $criteria = NULL)
Class Constructor

$name : widget's name
$database : database name
$model : model class name
$key : table field to be used as key in the combo
$value : table field to be listed in the combo
$ordercolumn : column to order the fields (optional)
$criteria : criteria (TCriteria object) to filter the model (optional)

TDBCombo extends TCombo

Database ComboBox Widget

Methods

public function __construct($name, $database, $model, $key, $value, $ordercolumn = NULL, TCriteria $criteria = NULL)
Class Constructor

$name : widget's name
$database : database name
$model : model class name
$key : table field to be used as key in the combo
$value : table field to be listed in the combo
$ordercolumn : column to order the fields (optional)
$criteria : criteria (TCriteria object) to filter the model (optional)

public static function reloadFromModel($formname, $field, $database, $model, $key, $value, $ordercolumn = NULL, $criteria = NULL, $startEmpty = FALSE)
Reload combo from model data

$formname : form name
$field : field name
$database : database name
$model : model class name
$key : table field to be used as key in the combo
$value : table field to be listed in the combo
$ordercolumn : column to order the fields (optional)

$criteria : criteria (TCriteria object) to filter the model (optional)
$startEmpty : if the combo will have an empty first item

TDBEntry extends TEntry

Database Entry Widget

Methods

public function __construct($name, $database, $model, $value, $orderColumn = NULL, TCriteria $criteria = NULL)
Class Constructor

$name : widget's name
$database : database name
$model : model class name
$value : table field to be listed in the combo
$ordercolumn : column to order the fields (optional)
$criteria : criteria (TCriteria object) to filter the model (optional)

public function setMinLength($length)
Define the minimum length for search

public function setOperator($operator)
Define the search operator

$operator : Search operator

public function show()
Shows the widget

TDBMultiSearch extends TMultiSearch

Database Multisearch Widget

Methods

public function __construct($name, $database, $model, $key, $value, $orderColumn = NULL, TCriteria $criteria = NULL)
Class Constructor

$name : widget's name
$database : database name
$model : model class name
$key : table field to be used as key in the combo
$value : table field to be listed in the combo
$ordercolumn : column to order the fields (optional)
$criteria : criteria (TCriteria object) to filter the model (optional)

public function setService($service)
Define the search service

$service : Search service

public function disableIdSearch()
Disable search by id

public function setOperator($operator)
Define the search operator

$operator : Search operator

public function setMask($mask)
Define the display mask

$mask : Show mask

public function setValue($values)
Define the field's value

$values : An array the field's values

public function getPostData()
Return the post data

public function show()
Shows the widget

TDBRadioGroup extends TRadioGroup

Database Radio Widget

Methods

public function __construct($name, $database, $model, $key, $value, $ordercolumn = NULL, TCriteria $criteria = NULL)
Class Constructor

$name : widget's name
$database : database name
$model : model class name
$key : table field to be used as key in the combo
$value : table field to be listed in the combo
$ordercolumn : column to order the fields (optional)
$criteria : criteria (TCriteria object) to filter the model (optional)

TDBSeekButton extends TSeekButton

Abstract Record Lookup Widget: Creates a lookup field used to search values from associated entities

Methods

public function __construct($name, $database, $form, $model, $display_field, $receive_key, $receive_display_field, TCriteria $criteria = NULL, $operator = 'like')
Class Constructor

$name : name of the form field
$database : name of the database connection
$form : name of the parent form
$model : name of the Active Record to be searched
$display_field : name of the field to be searched and shown
$receive_key : name of the form field to receive the primary key
$receive_display_field : name of the form field to receive the "display field"

TDBSelect extends TSelect

Database Select Widget

Methods

public function __construct($name, $database, $model, $key, $value, $ordercolumn = NULL, TCriteria $criteria = NULL)
Class Constructor

$name : widget's name
$database : database name
$model : model class name
$key : table field to be used as key in the combo
$value : table field to be listed in the combo
$ordercolumn : column to order the fields (optional)
$criteria : criteria (TCriteria object) to filter the model (optional)

TDBSortList extends TSortList

Database Sortlist Widget

Methods

public function __construct($name, $database, $model, $key, $value, $ordercolumn = NULL, TCriteria $criteria = NULL)
Class Constructor

$name : widget's name
$database : database name
$model : model class name
$key : table field to be used as key in the combo
$value : table field to be listed in the combo
$ordercolumn : column to order the fields (optional)
$criteria : criteria (TCriteria object) to filter the model (optional)

TDBUniqueSearch extends TDBMultiSearch

DBUnique Search Widget

Methods

public function __construct($name, $database, $model, $key, $value, $orderColumn = NULL, TCriteria $criteria = NULL)
Class Constructor

$name : Widget's name

public function setValue($value)
Define the field's value

$value : Current value

public function getPostData()
Return the post data

TDataGrid extends TTable

DataGrid Widget: Allows to create datagrids with rows, columns and actions

Methods

public function __construct()
Class Constructor

public function enablePopover($title, $content)
Enable popover

$title : Title
$content : Content

public function makeScrollable()
Make the datagrid scrollable

public function setActionWidth($width)
Set the column action width

public function disableDefaultClick()
disable the default click action

function setHeight($height)
Define the Height

$height : An integer containing the height

public function addColumn(TDataGridColumn $object)
Add a Column to the DataGrid

$object : A TDataGridColumn object

public function getColumns()
Returns an array of TDataGridColumn

public function addAction(TDataGridAction $object)
Add an Action to the DataGrid
$object : A TDataGridAction object

public function addActionGroup(TDataGridActionGroup $object)
Add an Action Group to the DataGrid
$object : A TDataGridActionGroup object

public function getTotalColumns()
Returns the total columns

public function setGroupColumn($column, $mask)
Set the group column for break

function clear($preserveHeader = TRUE)
Clear the DataGrid contents

public function createModel($create_header = true)
Creates the DataGrid Structure

public function getHead()
Return thead

public function getBody()
Return tbody

public function insert($position, $content)
insert content

public function addItems($objects)
Add objects to the DataGrid
$objects : An array of Objects

public function addItem($object)
Add an object to the DataGrid
$object : An Active Record Object

public function getItems()
Return datagrid items

public function getRowIndex($attribute, $value)
Find the row index by object attribute
$attribute : Object attribute
$value : Object value

public function getRow($position)
Return the row by position
$position : Row position

public function getWidth()
Returns the DataGrid's width

function show()
Shows the DataGrid

function setPageNavigation($pageNavigation)
Assign a PageNavigation object
$pageNavigation : object

function getPageNavigation()
Return the assigned PageNavigation object

TDataGridAction extends TAction

Represents an action inside a datagrid

Methods

public function setImage($image)
Define an icon for the action
$image : The Image path

public function getImage()
Returns the icon of the action

public function setLabel($label)
define the label for the action
$label : A string containing a text label

public function getLabel()
Returns the text label for the action

public function setField($field)
Define wich Active Record's property will be passed along with the action
$field : Active Record's property

public function setFields($fields)
Define wich Active Record's properties will be passed along with the action
$field : Active Record's property

public function getField()
Returns the Active Record's property that will be passed along with the action

public function getFields()
Returns the Active Record's properties that will be passed along with the action

public function setButtonClass($buttonClass)
define the buttonClass for the action
$buttonClass : A string containing the button css class

public function getButtonClass()
Returns the buttonClass

public function setUseButton($useButton)
define if the action will use a regular button
$useButton : A boolean

public function getUseButton()
Returns if the action will use a regular button

public function setDisplayCondition(/*Callable*/ $displayCondition)
Define a callback that must be valid to show the action
Callback : $displayCondition Action display condition

public function getDisplayCondition()
Returns the action display condition

public function serialize($format_action = TRUE)
Converts the action into an URL
$format_action : = format action with document or javascript (ajax=no)

TDataGridActionGroup

Represents a group of Actions for datagrids

Methods

public function __construct($label, $icon = NULL)
Constructor

$label : Action Group label
$icon : Action Group icon

public function getLabel()
Returns the Action Group label

public function getIcon()
Returns the Action Group icon

public function addAction(TAction $action)
Add an action to the actions group

$action : TAction object

public function addSeparator()
Add a separator

public function addHeader($header)
Add a header

$header : Options header

public function getActions()
Returns the actions

public function getHeaders()
Returns the headers

public function getSeparators()
Returns the separators

TDataGridColumn

Representes a DataGrid column

Methods

public function __construct($name, $label, $align, $width = NULL)
Class Constructor

$name : = Name of the column in the database
$label : = Text label that will be shown in the header
$align : = Column align (left, center, right)
$width : = Column Width (pixels)

public function setProperty($name, $value)
Define a field property

$name : Property Name
$value : Property Value

public function getProperty($name)
Return a field property

$name : Property Name

public function getProperties()
Return field properties

public function getName()
Returns the database column's name

public function getLabel()
Returns the column's label

public function setLabel($label)
Set the column's label
$label : column label

public function getAlign()
Returns the column's align

public function getWidth()
Returns the column's width

public function setAction(TAction $action)
Define the action to be executed when the user clicks over the column header
$action : A TAction object

public function getAction()
Returns the action defined by set_action() method

public function setEditAction(TDataGridAction $editaction)
Define the action to be executed when the user clicks do edit the column
$action : A TDataGridAction object

public function getEditAction()
Returns the action defined by setEditAction() method

public function setTransformer(Callable $callback)
Define a callback function to be applyed over the column's data
$callback : A function name of a method of an object

public function getTransformer()
Returns the callback defined by the setTransformer()

public function setTotalFunction(Callable $callback)
Define a callback function to totalize column
$callback : A function name of a method of an object

public function getTotalFunction()
Returns the callback defined by the setTotalFunction()

TDate extends TEntry

DatePicker Widget

Methods

public function __construct($name)
Class Constructor
$name : Name of the widget

public function setValue($value)
Store the value inside the object

public function getPostData()
Return the post data

public static function convertToMask($value, $fromMask, $toMask)
Convert from one mask to another
$value : original date
$fromMask : source mask

$toMask : target mask

public function setMask($mask)
Define the field's mask

$mask : Mask for the field (dd-mm-yyyy)

public function setDatabaseMask($mask)
Set the mask to be used to colect the data

public function setOption($option, $value)
Set extra datepicker options (ex: autoclose, startDate, daysOfWeekDisabled, datesDisabled)

public static function date2us($date)
Shortcut to convert a date to format yyyy-mm-dd

$date : = date in format dd/mm/yyyy

public static function date2br($date)
Shortcut to convert a date to format dd/mm/yyyy

$date : = date in format yyyy-mm-dd

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public function show()
Shows the widget at the screen

TDateTime extends TEntry

DatTimePicker Widget

Methods

public function __construct($name)
Class Constructor

$name : Name of the widget

public function setValue($value)
Store the value inside the object

public function getPostData()
Return the post data

public static function convertToMask($value, $fromMask, $toMask)
Convert from one mask to another

$value : original date
$fromMask : source mask
$toMask : target mask

public function setMask($mask)
Define the field's mask

$mask : Mask for the field (dd-mm-yyyy)

public function setDatabaseMask($mask)
Set the mask to be used to colect the data

public function setOption($option, $value)
Set extra datepicker options (ex: autoclose, startDate, daysOfWeekDisabled, datesDisabled)

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public function show()
Shows the widget at the screen

TDropDown extends TElement

TDropDown Widget

Methods

public function __construct($label, $icon = NULL, $use_caret = TRUE, $title = '', $height = null)
Class Constructor

$title : Dropdown title
$icon : Dropdown icon

public function setPullSide($side)
Define the pull side

public function setButtonSize($size)
Define the button size

public function setButtonClass($class)
Define the button class

public function getButton()
Returns the dropdown button

public function addAction($title, $action, $icon = NULL)
Add an action

$title : Title
$action : Action (TAction or string Javascript action)
$icon : Icon

public function addHeader($header)
Add a header

$header : Options Header

public function addSeparator()
Add a separator

TElement

Base class for all HTML Elements

Methods

public function __construct($tagname)
Class Constructor

$tagname : tag name

public static function tag($tagname, $value, $attributes = NULL)
Create an element

$tagname : Element name
$value : Element value
$attributes : Element attributes

public function setName($tagname)
Change the element name

$tagname : Element name

protected function setIsWrapped($bool)
Define if the element is wrapped inside another one

@bool : Boolean TRUE if is wrapped

public function getIsWrapped()
Return if the element is wrapped inside another one

public function getProperties()
Return element properties

public function add($child)
Add an child element

$child : Any object that implements the show() method

public function insert($position, $child)
Insert an child element

$position : Element position
$child : Any object that implements the show() method

public function setUseLineBreaks($linebreaks)
Set the use of linebreaks

$linebreaks : boolean

public function setUseSingleQuotes($singlequotes)
Set the use of single quotes

$singlequotes : boolean

public function del($object)
Del an child element

$child : Any object that implements the show() method

public function getChildren()
get children

public function get($position)
Get an child element

$position : Element position

public function open()
Opens the tag

public function show()
Shows the tag

public function close()
Closes the tag

public function getContents()
Returns the element content as a string

public function clearChildren()
Clear element children

TEmailValidator extends TFieldValidator

Email validation

Methods

public function validate($label, $value, $parameters = NULL)
Validate a given value

$label : Identifies the value to be validated in case of exception
$value : Value to be validated
$parameters : aditional parameters for validation

TEntry extends TField

Entry Widget

Methods

public function __construct($name)
Class Constructor

$name : name of the field

public function setInputType($type)
Define input type

public function setMask($mask)
Define the field's mask

$mask : A mask for input data

public function setNumericMask($decimals, $decimalsSeparator, $thousandSeparator, $replaceOnPost = FALSE)
Define the field's numeric mask (available just in web)

$decimals : Sets the number of decimal points.
$decimalsSeparator : Sets the separator for the decimal point.
$thousandSeparator : Sets the thousands separator.

public function setValue($value)
Define the field's value

$value : A string containing the field's value

public function getPostData()
Return the post data

public function setMaxLength($length)
Define max length

$length : Max length

function setCompletion($options)
Define options for completion

$options : array of options for completion

function setExitAction(TAction $action)
Define the action to be executed when the user leaves the form field

$action : TAction object

public function setExitFunction($function)
Define the javascript function to be executed when the user leaves the form field

$function : Javascript function

public function forceLowerCase()
Force lower case

public function forceUpperCase()
Force upper case

public function show()
Shows the widget at the screen

TExceptionView
Exception visualizer

Methods

function __construct(Exception $e)
Constructor method

TExpander extends TElement
Expander Widget

Methods

public function __construct($label = '')
Class Constructor

$value : text label

public function setCaretSide($caret_side)
Set caret side

public function setPullSide($side)
Define the pull side

public function setButtonProperty($property, $value)
Define a button property

$property : Property name (Ex: style)
$value : Property value

public function setProperty($property, $value)
Define a container property

$property : Property name (Ex: style)
$value : Property value

public function add($content)
Add content to the expander

$content : Any Object that implements show() method

public function show()
Shows the expander

TExpression
Base class for TCriteria and TFilter (composite pattern implementation)

Methods

abstract public function dump();

TField
Base class to construct all the widgets

Methods

public function __construct($name)
Class Constructor

$name : name of the field

public function setLabel($label)
Define the field's label

$label : A string containing the field's label

public function getLabel()
Returns the field's label

public function setName($name)
Define the field's name

$name : A string containing the field's name

public function getName()
Returns the field's name

public function setId($id)
Define the field's id

$id : A string containing the field's id

public function getId()
Returns the field's id

public function setValue($value)
Define the field's value

$value : A string containing the field's value

public function getValue()
Returns the field's value

public function getFormName()
Return the name of the form to wich the field is attached

public function setTip($tip)
Define the field's tooltip

$name : A string containing the field's tooltip

public function getPostData()
Return the post data

public function setEditable($editable)
Define if the field is editable

$editable : A boolean

public function getEditable()
Returns if the field is editable

public function setProperty($name, $value, $replace = TRUE)
Define a field property

$name : Property Name
$value : Property Value

public function getProperty($name)
Return a field property

$name : Property Name
$value : Property Value

public function setSize($width, $height = NULL)
Define the Field's width

$width : Field's width in pixels

public function getSize()

Returns the field size

public function addValidation($label, TFieldValidator $validator, $parameters = NULL)
Add a field validator

$label : Field name
$validator : TFieldValidator object
$parameters : Aditional parameters

public function getValidations()
Returns field validations

public function isRequired()
Returns if the field is required

public function validate()
Validate a field

public function getContents()
Returns the element content as a string

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
Clear the field

$form_name : Form name
$field : Field name

TFieldList extends TTable

Create a field list

Methods

public function __construct()
Class Constructor

public function enableSorting()
Enable sorting

public function setSortAction(TAction $action)
Define the action to be executed when the user sort rows

$action : TAction object

public function setRemoveFunction($action)
Set the remove javascript action

public function setCloneFunction($action)
Set the clone javascript action

public function addField($label, AdiantiWidgetInterface $field)
Add a field

$label : Field Label
$object : Field Object

public function addHeader()
Add table header

public function addDetail($item)
Add detail row
$item : Data object

public function addCloneAction()
Add clone action

public function show()

TFieldValidator
TFieldValidator abstract validation class

Methods

abstract public function validate($label, $value, $parameters = NULL);
Validate a given value
$label : Identifies the value to be validated in case of exception
$value : Value to be validated
$parameters : aditional parameters for validation

TFile extends TField
FileChooser widget

Methods

public function __construct($name)
Constructor method
$name : input name

public function setDisplayMode($mode)
Define the display mode {file}

public function setService($service)
Define the service class for response

public function setAllowedExtensions($extensions)
Define the allowed extensions

public function enableFileHandling()
Define to file handling

public function setPlaceHolder(TElement $widget)
Set place holder

public function setSize($width, $height = NULL)
Set field size

public function setHeight($height)
Set field height

public function getPostData()
Return the post data

public function setValue($value)
Set field value

public function show()
Show the widget at the screen

function setCompleteAction(TAction $action)
Define the action to be executed when the user leaves the form field
$action : TAction object

public static function enableField($form_name, $field)
Enable the field
$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field
$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
Clear the field
$form_name : Form name
$field : Field name

TFilter extends TExpression
Provides an interface to define filters to be used inside a criteria

Methods

public function __construct($variable, $operator, $value, $value2 = NULL)
Class Constructor
$variable : = variable
$operator : = operator (>,
$value : = value to be compared
$value2 : = second value to be compared (between)

public function getPreparedVars()
Return the prepared vars

public function dump($prepared = FALSE)
Return the filter as a string expression

TForm
Wrapper class to deal with forms

Methods

public function __construct($name = 'my_form')
Class Constructor
$name : Form Name

public function setProperty($name, $value, $replace = TRUE)
Define a form property
$name : Property Name
$value : Property Value

public static function getFormByName($name)
Returns the form object by its name

public function setName($name)
Define the form name
$name : A string containing the form name

public function getName()
Returns the form name

public static function sendData($form_name, $object, $aggregate = FALSE, $fireEvents = TRUE)
Send data for a form located in the parent window

$form_name : Form Name
$object : An Object containing the form data

public function setEditable($bool)
Define if the form will be editable

$bool : A Boolean

public function addField(AdiantiWidgetInterface $field)
Add a Form Field

$field : Object

public function delField(AdiantiWidgetInterface $field)
Remove a form field

$field : Object

public function delFields()
Remove all form fields

public function setFields($fields)
Define wich are the form fields

$fields : An array containing a collection of TField objects

public function getField($name)
Returns a form field by its name

$name : A string containing the field's name

public function getFields()
Returns an array with the form fields

public function clear($keepDefaults = FALSE)
clear the form Data

public function setData($object)
Define the data of the form

$object : An Active Record object

public function getData($class = 'StdClass')
Returns the form POST data as an object

$class : A string containing the class for the returning object

public function getValues($class = 'StdClass', $withOptions = false)
Returns the form start values as an object

$class : A string containing the class for the returning object

public function validate()
Validate form

public function add($object)
Add a container to the form (usually a table or panel)

$object : Any Object that implements the show() method

public function pack()
Pack a container to the form (usually a table or panel)

mixed : $object, ...Any Object that implements the show() method

public function getChild()
Returns the child object

public function show()
Shows the form at the screen

TFormSeparator extends TElement
Form separator

Methods

public function __construct($text)
Class Constructor

$text : Separator title

public function setFontSize($size)
Set font size

$size : font size

public function setFontColor($color)
Set font color

$color : font color

public function setSeparatorColor($color)
Set separator color

$color : separator color

TFrame extends TElement
Frame Widget: creates a bordered area with a title located at its top-left corner

Methods

public function __construct($width = NULL, $height = NULL)
Class Constructor

$value : text label

public function getSize()
Returns the frame size

public function setLegend($legend)
Set Legend

$legend : frame legend

public function getLegend()
Returns the inner legend

TFullCalendar extends TElement
FullCalendar Widget

Methods

public function __construct($current_date = NULL, $default_view = 'month')
Class Constructor

$current_date : Current date of calendar
$default_view : Default view (month, agendaWeek, agendaDay)

public function setTimeRange($min_time, $max_time)
Define the time range

public function enableDays($days)
Enable these days

public function setCurrentDate($date)
Set the current date of calendar

$date : Current date of calendar

public function setCurrentView($view)
Set the current view of calendar

$view : Current view of calendar (month, agendaWeek, agendaDay)

public function setReloadAction(TAction $action)
Define the reload action

$action : reload action

public function setEventClickAction(TAction $action)
Define the event click action

$action : event click action

public function setDayClickAction(TAction $action)
Define the day click action

$action : day click action

public function setEventUpdateAction(TAction $action)
Define the event update action

$action : event updaet action

public function enablePopover($title, $content)
Enable popover

$title : Title
$content : Content

public function addEvent($id, $title, $start, $end = NULL, $url = NULL, $color = NULL, $object = NULL)
Add an event

$id : Event id
$title : Event title
$start : Event start time
$end : Event end time
$url : Event url
$color : Event color

public function show()
Show the callendar and execute required scripts

THBox extends TElement

Horizontal Box

Methods

public function __construct()
Class Constructor

public function add($child, $style = 'display:inline-table;')
Add an child element

$child : Any object that implements the show() method

public function addRowSet()
Add a new row with many cells

$cells : Each argument is a row cell

public static function pack()
Static method for pack content

$cells : Each argument is a cell

THidden extends TField

Hidden field

Methods

public function show()
Show the widget at the screen

THtmlEditor extends TField

Html Editor

Methods

public function __construct($name)
Class Constructor

$name : Widet's name

public function setSize($width, $height = NULL)
Define the widget's size

$width : Widget's width
$height : Widget's height

public function getSize()
Returns the size

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
Clear the field

$form_name : Form name
$field : Field name

public function show()
Show the widget

THtmlRenderer

Html Renderer

Methods

public function __construct($path)
Constructor method

$path : HTML resource path

public function enableTranslation()
Enable translation inside template

public function enableSection($sectionName, $replacements = NULL, $repeat = FALSE)
Enable a HTML section to show

$sectionName : Section name
$replacements : Array of replacements for this section

$repeat : Define if the section is repeatable

public function show()
Show the HTML and the enabled sections

public static function recursiveKeyArraySearch($needle,$haystack)
Static search in memory structure

public function getContents()
Returns the HTML content as a string

THyperLink extends TTextDisplay

File Link

Methods

public function __construct($value, $location, $color = null, $size = null, $decoration = null, $icon = null)
Class Constructor

$value : text content
$location : link location
$color : text color
$size : text size
$decoration : text decorations (b=bold, i=italic, u=underline)

TIcon extends TEntry

Color Widget

Methods

public function __construct($name)
Class Constructor

$name : Name of the widget

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public function setChangeFunction($function)
Set change function

public function show()
Shows the widget at the screen

TImage extends TElement

Image Widget

Methods

public function __construct($source)
Class Constructor

$source : Image path, of bs:bs-glyphicon, fa:font-awesome

TInputDialog
Input Dialog

Methods

public function __construct($title_msg, AdiantiFormInterface $form, TAction $action = NULL, $caption = '')
Class Constructor

$title_msg : Dialog Title
$form : Dialog form body
$action : Action to be processed when closing the dialog
$caption : Button caption

TJQueryDialog extends TElement
JQuery dialog container

Methods

public function __construct()
Class Constructor

$name : Name of the widget

public function setUseOKButton($bool)
Define if will use OK Button

$bool : boolean

public function setTitle($title)
Define the dialog title

$title : title

public function setModal($bool)
Turn on/off modal

$modal : Boolean

public function setResizable($bool)
Turn on/off resizeable

$bool : Boolean

public function setDraggable($bool)
Turn on/off draggable

$bool : Boolean

public function getId()
Returns the element ID

public function setSize($width, $height)
Define the dialog size

$width : width
$height : height

public function setPosition($left, $top)
Define the dialog position

$left : left
$top : top

public function addAction($label, $action)
Add a JS button to the dialog

$label : button label
$action : JS action

public function setStackOrder($order)
Define the stack order (zIndex)

$order : Stack order

public function show()
Shows the widget at the screen

public function close()
Closes the dialog

public static function closeAll()
Close all TJQueryDialog

TLabel extends TField

Label Widget

Methods

public function __construct($value, $color = null, $size = null, $decoration = null)
Class Constructor

$value : text label

public function setFontSize($size)
Define the font size

$size : Font size in pixels

public function setFontStyle($decoration)
Define the style

$decoration : text decorations (b=bold, i=italic, u=underline)

public function setFontFace($font)
Define the font face

$font : Font Family Name

public function setFontColor($color)
Define the font color

$color : Font Color

function add($content)
Add a content inside the label

$content :

public function getValue()
Get value

public function show()
Shows the widget at the screen

TLogger

Provides an abstract interface to register LOG files

Methods

public function __construct($filename = NULL)
Class Constructor

$filename : path for LOG file

abstract function write($message);
Write abstract method Must be declared in child classes

TLoggerHTML extends TLogger

Register LOG in HTML files

Methods

public function write($message)
Writes an message in the LOG file
$message : Message to be written

TLoggerSTD extends TLogger

Register LOG in Standard Output

Methods

public function write($message)
Writes an message in the LOG file
$message : Message to be written

TLoggerTXT extends TLogger

Register LOG in TXT files

Methods

public function write($message)
Writes an message in the LOG file
$message : Message to be written

TLoggerXML extends TLogger

Register LOG in HTML files

Methods

public function write($message)
Writes an message in the LOG file
$message : Message to be written

TMaxLengthValidator extends TFieldValidator

Maximum length validation

Methods

public function validate($label, $value, $parameters = NULL)
Validate a given value
$label : Identifies the value to be validated in case of exception
$value : Value to be validated
$parameters : aditional parameters for validation (length)

TMaxValueValidator extends TFieldValidator

Maximum value validation

Methods

public function validate($label, $value, $parameters = NULL)
Validate a given value

$label : Identifies the value to be validated in case of exception
$value : Value to be validated
$parameters : aditional parameters for validation (max value)

TMenu extends TElement

Menu Widget

Methods

public function __construct($xml, $permission_callback = NULL, $menu_level = 1, $menu_class = 'dropdown-menu', $item_class = '', $link_class = 'dropdown-toggle')
Class Constructor

$xml : SimpleXMLElement parsed from XML Menu

public function addMenuItem(TMenuItem $menuitem)
Add a MenuItem

$menuitem : A TMenuItem Object

public function getMenuItems()
Return the menu items

public function parse($xml, $permission_callback = NULL)
Parse a XMLElement reading menu entries

$xml : A SimpleXMLElement Object
$permission_callback : check permission callback

public function show()
Shows the widget at the screen

TMenuBar extends TElement

Menubar Widget

Methods

public function __construct()

public static function newFromXML($xml_file, $permission_callback = NULL, $bar_class = 'nav navbar-nav', $menu_class = 'dropdown-menu', $item_class = '')
Build a MenuBar from a XML file

$xml_file : path for the file
$permission_callback : check permission callback

public function show()
Show

TMenuItem extends TElement

MenuItem Widget

Methods

public function __construct($label, $action, $image = NULL, $level = 0)
Class constructor

$label : The menu label
$action : The menu action
$image : The menu image

public function setLinkClass($class)
Set link class

public function setMenu(TMenu $menu)
Define the submenu for the item

$menu : A TMenu object

public function show()
Shows the widget at the screen

TMenuParser
Menu Parser

Methods

public function __construct($xml_file)
Parse a menu XML file

$xml_file : file path

public function getIndexedPrograms()
Return an indexed array of programs

public function getPath($controller)
Return the controller path

public function appendPage($module, $label, $action, $icon)
append page

TMessage
Message Dialog

Methods

public function __construct($type, $message, TAction $action = NULL, $title_msg = '')
Class Constructor

$type : Type of the message (info, error)
$message : Message to be shown
$action : Action to be processed when closing the dialog
$title_msg : Dialog Title

TMinLengthValidator extends TFieldValidator
Minimum length validation

Methods

public function validate($label, $value, $parameters = NULL)
Validate a given value

$label : Identifies the value to be validated in case of exception
$value : Value to be validated
$parameters : aditional parameters for validation (length)

TMinValueValidator extends TFieldValidator
Minimum value validation

Methods

public function validate($label, $value, $parameters = NULL)
Validate a given value

$label : Identifies the value to be validated in case of exception

$value : Value to be validated
$parameters : aditional parameters for validation (min value)

TMultiEntry extends TSelect

Multi Entry Widget

Methods

public function __construct($name)
Class Constructor

$name : Widget's name

public function setSize($width, $height = NULL)
Define the widget's size

$width : Widget's width
$height : Widget's height

public function setMaxSize($maxsize)
Define the maximum number of items that can be selected

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
Clear the field

$form_name : Form name
$field : Field name

protected function renderItems($with_titles = true)
Render items

public function show()
Shows the widget

TMultiField extends TField

MultiField Widget: Takes a group of input fields and gives them the possibility to register many occurrences

Methods

public function __construct($name)
Class Constructor

$name : Name of the widget

public function setOrientation($orientation)
Define form orientation

$orientation : (vertical, horizontal)

public function addField($name, $text, TField $object, $size, $mandatory = FALSE)
Add a field to the MultiField

$name : Widget's name
$text : Widget's label
$object : Widget
$size : Widget's size

$mandatory : Mandatory field

public function setClass($class)
Define the class for the Active Records returned by this component

$class : Class Name

public function getClass()
Returns the class defined by the setClass() method

public function setValue($objects)
Define the MultiField content

$objects : A Collection of Active Records

public function getPostData()
Return the post data

public function setHeight($height)
Define the MultiField height

$height : Height in pixels

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
Clear the field

$form_name : Form name
$field : Field name

public function show()
Show the widget at the screen

TMultiFile extends TField

FileChooser widget

Methods

public function __construct($name)
Constructor method

$name : input name

public function setService($service)
Define the service class for response

public function setAllowedExtensions($extensions)
Define the allowed extensions

public function enableFileHandling()
Define to file handling

public function setSize($width, $height = NULL)
Set field size

public function setHeight($height)
Set field height

public function getPostData()
Return the post data

public function setValue($value)
Set field value

public function show()
Show the widget at the screen

function setCompleteAction(TAction $action)
Define the action to be executed when the user leaves the form field
$action : TAction object

public static function enableField($form_name, $field)
Enable the field
$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field
$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
Clear the field
$form_name : Form name
$field : Field name

TMultiSearch extends TSelect
Multi Search Widget

Methods

public function __construct($name)
Class Constructor
$name : Widget's name

public function disableMultiple()
Disable multiple selection

public function disableClear()
Disable clear

public function disableSearch()
Disable search

public function setSize($width, $height = NULL)
Define the widget's size
$width : Widget's width
$height : Widget's height

public function getSize()
Returns the size

public function setMinLength($length)
Define the minimum length for search

public function setMaxSize($maxsize)
Define the maximum number of items that can be selected

public function setValueSeparator($sep)

Define the field's separator

$sep : A string containing the field's separator

public function setValue($value)
Define the field's value

$value : A string containing the field's value

public function getPostData()
Return the post data

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
Clear the field

$form_name : Form name
$field : Field name

public function show()
Shows the widget

TNotebook extends TElement

Notebook

Methods

public function __construct($width = null, $height = null)
Class Constructor

$width : Notebook's width
$height : Notebook's height

public function setTabsVisibility($visible)
Define if the tabs will be visible or not

$visible : If the tabs will be visible

public function setTabsSensibility($sensibility)
Define the tabs click sensibility

$sensibility : If the tabs will be sensible to click

public function getId()
Returns the element ID

public function setSize($width, $height)
Set the notebook size

$width : Notebook's width
$height : Notebook's height

public function getSize()
Returns the frame size

public function setCurrentPage($i)
Define the current page to be shown

$i : An integer representing the page number (start at 0)

public function getCurrentPage()
Returns the current page

public function appendPage($title, $object)
Add a tab to the notebook

$title : tab's title
$object : tab's content

public function getPageCount()
Return the Page count

public function setTabAction(TAction $action)
Define the action for the Notebook tab

$action : Action taken when the user clicks over Notebook tab (A TAction object)

public function render()
Render the notebook

public function show()
Show the notebook

TNumeric extends TEntry
Numeric Widget

Methods

public function __construct($name, $decimals, $decimalsSeparator, $thousandSeparator, $replaceOnPost = true)

TNumericValidator extends TFieldValidator
Numeric validation

Methods

public function validate($label, $value, $parameters = NULL)
Validate a given value

$label : Identifies the value to be validated in case of exception
$value : Value to be validated
$parameters : aditional parameters for validation (min value)

TPDFDesigner
FPDF Adapter that parses XML files from Adianti Framework

Methods

public function __construct($orientation = 'P', $format = 'a4', $unit = 'pt')
Constructor method

$orientation : Page orientation
$format : Page format

public function fromXml($filename)
Load designed elements from XML

$filename : XML file location

public function loadElements($elements)
Load Elements

$elements : Elements (shapes) to load

public function gotoAnchorXY($anchor_name)

Put the cursor at the anchor XY position

$anchor_name : Anchor name

public function gotoAnchorX($anchor_name)
Put the cursor at the anchor X position

$anchor_name : Anchor name

public function gotoAnchorY($anchor_name)
Put the cursor at the anchor Y position

$anchor_name : Anchor name

public function writeAtAnchor($anchor_name, $text)
Write at the anchor position

$anchor_name : Anchor name
$text : Text to write

public function replace($mark, $text)
Replace a piece of {text}

$mark : piece to be replaced
$text : new content

public function generate()
Generate one PDF page with the parsed elements

public function ellipse($x, $y, $rx, $ry, $style = 'D')
Draws an ellipse

$x : X
$y : Y
$rx : X Ray
$ry : Y Ray
$style : Line Style

public function writeHTML($x, $y, $html)
Write HTML

$x : X
$y : Y
$html : HTML

public function openTag($tag,$attr, $x)
Open Html TAG

$tag : Tag
$attr : Tag attributes
$x : X position

public function closeTag($tag)
Close Html TAG

$tag : Tag

public function setStyle($tag,$enable)
Set Style

$tag : Tag
$enable : Enable

public function putLink($URL,$txt)
Put link

$URL :
$txt :

public function setLocale()
Change PDF locale

public function unsetLocale()
Back to the old locale

public function setFontColorRGB($color)
Changes the color
$color : Color in RGB

public function setFillColorRGB($color)
Changes the fill color
$color : Color in RGB

public function setDrawColorRGB($color)
Changes the draw color
$color : Color in RGB

public function save($output)
Saves the PDF
$output : Output path

TPage extends TElement

Page Controller Pattern: used as container for all elements inside a page and also as a page controller

Methods

public function __construct()
Class Constructor

public function run()
Interprets an action based at the URL parameters

public static function include_js($js)
Include a specific JavaScript function to this page
$js : JavaScript location

public static function include_css($css)
Include a specific Cascading Stylesheet to this page
$css : Cascading Stylesheet

public static function register_css($cssname, $csscode)
Register a specific Cascading Stylesheet to this page
$cssname : Cascading Stylesheet Name
$csscode : Cascading Stylesheet Code

public static function openFile($file)
Open a File Dialog
$file : File Name

public static function isMobile()
Discover if the browser is mobile device

public function show()
Decide wich action to take and show the page

TPageNavigation

Page Navigation provides navigation for a datagrid

Methods

public function __construct()

public function hide()
Hide

public function setLimit($limit)
Set the Amount of displayed records

$limit : An integer

public function getLimit()
Returns the limit of records

public function setWidth($width)
Define the PageNavigation's width

$width : PageNavigation's width

public function setCount($count)
Define the total count of records

$count : An integer (the total count of records)

public function getCount()
Return the total count of records

public function setPage($page)
Define the current page

$page : An integer (the current page)

public function getPage()
Returns the current page

public function setFirstPage($first_page)
Define the first page

$page : An integer (the first page)

public function setOrder($order)
Define the ordering

$order : A string containint the column name

public function setDirection($direction)
Define the ordering

$direction : asc, desc

public function setProperties($properties)
Set the page navigation properties

$properties : array of properties

public function setAction($action)
Define the PageNavigation action

$action : TAction object (fired when the user navigates)

public function show()
Show the PageNavigation widget

TPanel extends TElement

Panel Container: Allows to organize the widgets using fixed (absolute) positions

Methods

public function __construct($width, $height)
Class Constructor

$width : Panel's width
$height : Panel's height

public function setSize($width, $height)
Set the panel's size

$width : Panel width

$height : Panel height

public function getSize()
Returns the frame size

public function put($widget, $col, $row)
Put a widget inside the panel

$widget : = widget to be shown
$col : = column in pixels.
$row : = row in pixels.

public function show()
Show the widget

TPanelGroup extends TElement

Bootstrap native panel for Adianti Framework

Methods

public function __construct($title = NULL, $background = NULL)
Constructor method

$title : Panel Title
$footer : Panel Footer

public static function pack($title, $element, $footer = null)
Static creator for panels

$title : Panel title
$element : Panel content

public function add($content)
Add the panel content

public function getHeader()
Return panel header

public function getBody()
Return panel body

public function getFooter()
Return panel footer

public function addFooter($footer)
Add footer

TPassword extends TField

Password Widget

Methods

function setExitAction(TAction $action)
Define the action to be executed when the user leaves the form field

$action : TAction object

public function setExitFunction($function)
Define the javascript function to be executed when the user leaves the form field

$function : Javascript function

public function show()
Show the widget at the screen

TProgressBar extends TElement

TProgressBar

Methods

public function __construct()

public function setMask($mask)
set mask for progress bar value Ex: "{value}%"

public function setClass($class)
set style class

public function setValue($value)
Set the value of progress bar

public function show()
Shows the widget at the screen

TQuestion

Question Dialog

Methods

public function __construct($message, TAction $action_yes = NULL, TAction $action_no = NULL, $title_msg = '')
Class Constructor

$message : A string containint the question
$action_yes : Action taken for YES response
$action_no : Action taken for NO response
$title_msg : Dialog Title

TQuickForm extends TForm

Create quick forms for input data with a standard container for elements

Methods

public function __construct($name = 'my_form')
Class Constructor

$name : Form Name

public function getActionsContainer()
Returns the actions container

public function getTable()
Returns the inner table

public function setFieldsByRow($count)
Define the field quantity per row

$count : Field count

public function getFieldsByRow()
Return the fields by row count

public function getContainer()
Returns the form container

public function setFormTitle($title)
Add a form title

$title : Form title

public function getInputRows()
Returns the input groups

public function addQuickField($label, AdiantiWidgetInterface $object, $size = 200, TFieldValidator $validator = NULL, $label_size = NULL)
Add a form field

$label : Field Label
$object : Field Object
$size : Field Size
$validator : Field Validator

public function addQuickFields($label, $objects, $required = FALSE)
Add a form field

$label : Field Label
$objects : Array of Objects
$required : Boolean TRUE if required

public function addQuickAction($label, TAction $action, $icon = 'ico_save.png')
Add a form action

$label : Action Label
$action : TAction Object
$icon : Action Icon

public function addQuickButton($label, $action, $icon = 'ico_save.png')
Add a form button

$label : Action Label
$action : Javascript action
$icon : Action Icon

public function delActions()
Clear actions row

public function getActionButtons()
Return an array with action buttons

public function detachActionButtons()
Detach action buttons

public function addRow()
Add a row

public static function showField($form, $field)

public static function hideField($form, $field)

TQuickGrid extends TDataGrid

Create quick datagrids through its simple interface

Methods

public function addQuickColumn($label, $name, $align = 'left', $size = 200, TAction $action = NULL, $param = NULL)
Add a column

$label : Field Label
$object : Field Object
$size : Field Size

public function addQuickAction($label, TDataGridAction $action, $field, $icon = NULL)
Add action to the datagrid

$label : Action Label
$action : TAction Object
$icon : Action Icon

TQuickNotebookForm extends TQuickForm

Create quick forms with a notebook wrapper

Methods

public function __construct($name = 'my_form')
Class Constructor

$name : Form Name

public function setNotebookWrapper($notebook)
Set the notebook wrapper

$notebook : Notebook wrapper

public function setFormTitle($title)
Add a form title

$title : Form title

public function appendPage($title, $container = NULL)
Append a notebook page

$title : Page title
$cotnainer : Page container

public function addQuickAction($label, TAction $action, $icon = 'fa:save')
Add a form action

$label : Action Label
$action : TAction Object
$icon : Action Icon

public function show()
Show the component

TRadioButton extends TField

RadioButton Widget

Methods

public function show()
Show the widget at the screen

TRadioGroup extends TField

A group of RadioButton's

Methods

public function __construct($name)
Class Constructor

$name : name of the field

public function setBooleanMode()
Enable/disable boolean mode

public function setValue($value)
Define the field's value

$value : A string containing the field's value

public function getValue()
Returns the field's value

public function getPostData()

Return the post data

public function setLayout($dir)
Define the direction of the options

$direction : String (vertical, horizontal)

public function getLayout()
Get the direction (vertical or horizontal)

public function setBreakItems($breakItems)
Define after how much items, it will break

public function setUseButton()
Show as button

public function addItems($items)
Add items to the radio group

$items : An indexed array containing the options

public function getItems()
Return the items

public function getButtons()
Return the option buttons

public function getLabels()
Return the option labels

public function setChangeAction(TAction $action)
Define the action to be executed when the user changes the combo

$action : TAction object

public function setChangeFunction($function)
Set change function

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
clear the field

$form_name : Form name
$field : Field name

public function show()
Show the widget at the screen

TRecord

Base class for Active Records

Methods

public function __construct($id = NULL, $callObjectLoad = TRUE)
Class Constructor Instantiates the Active Record

[$id] : Optional Object ID, if passed, load this object

public static function create($data)
Create a new TRecord and returns the instance
$data : indexed array

public function getCacheControl()
Returns the cache control

protected function getEntity()
Returns the name of database entity

public function getPrimaryKey()
Returns the the name of the primary key for that Active Record

public function mergeObject(TRecord $object)
Fill the Active Record properties from another Active Record
$object : An Active Record

public function fromArray($data)
Fill the Active Record properties from an indexed array
$data : An indexed array containing the object properties

public function toArray()
Return the Active Record properties as an indexed array

public function toJson()
Return the Active Record properties as a json string

public function render($pattern, $cast = null)
Render variables inside brackets

public function evaluate($pattern)
Evaluate variables inside brackets

public function addAttribute($attribute)
Register an persisted attribute

public function getAttributes()
Return the persisted attributes

public function store()
Store the objects into the database

public function exists($id)
Tests if an ID exists
$id : The object ID

public function reload()
ReLoad an Active Record Object from the database

public function load($id)
Load an Active Record Object from the database
$id : The object ID

public function delete($id = NULL)
Delete an Active Record object from the database
[$id] : The Object ID

public function getFirstID()
Returns the FIRST Object ID from database

public function getLastID()
Returns the LAST Object ID from database

public static function getObjects($criteria = NULL, $callObjectLoad = TRUE)
Method getObjects

$criteria : Optional criteria
$callObjectLoad : If load() method from Active Records must be called to load object parts

public static function countObjects($criteria = NULL)
Method countObjects

$criteria : Optional criteria

public function loadComposite($composite_class, $foreign_key, $id = NULL, $order = NULL)
Load composite objects (parts in composition relationship)

$composite_class : Active Record Class for composite objects
$foreign_key : Foreign key in composite objects
$id : Primary key of parent object

public function hasMany($composite_class, $foreign_key = NULL, $primary_key = NULL, $order = NULL)
Load composite objects. Shortcut for loadComposite

$composite_class : Active Record Class for composite objects
$foreign_key : Foreign key in composite objects
$primary_key : Primary key of parent object

public function filterMany($composite_class, $foreign_key = NULL, $primary_key = NULL, $order = NULL)
Create a criteria to load composite objects

$composite_class : Active Record Class for composite objects
$foreign_key : Foreign key in composite objects
$primary_key : Primary key of parent object

public function deleteComposite($composite_class, $foreign_key, $id, $callObjectLoad = FALSE)
Delete composite objects (parts in composition relationship)

$composite_class : Active Record Class for composite objects
$foreign_key : Foreign key in composite objects
$id : Primary key of parent object

public function saveComposite($composite_class, $foreign_key, $id, $objects, $callObjectLoad = FALSE)
Save composite objects (parts in composition relationship)

$composite_class : Active Record Class for composite objects
$foreign_key : Foreign key in composite objects
$id : Primary key of parent object
$objects : Array of Active Records to be saved

public function loadAggregate($aggregate_class, $join_class, $foreign_key_parent, $foreign_key_child, $id = NULL)
Load aggregated objects (parts in aggregation relationship)

$aggregate_class : Active Record Class for aggregated objects
$join_class : Active Record Join Class (Parent / Aggregated)
$foreign_key_parent : Foreign key in Join Class to parent object
$foreign_key_child : Foreign key in Join Class to child object
$id : Primary key of parent object

public function belongsToMany($aggregate_class, $join_class = NULL, $foreign_key_parent = NULL, $foreign_key_child = NULL)
Load aggregated objects. Shortcut to loadAggregate

$aggregate_class : Active Record Class for aggregated objects
$join_class : Active Record Join Class (Parent / Aggregated)
$foreign_key_parent : Foreign key in Join Class to parent object
$foreign_key_child : Foreign key in Join Class to child object
$id : Primary key of parent object

public function saveAggregate($join_class, $foreign_key_parent, $foreign_key_child, $id, $objects)
Save aggregated objects (parts in aggregation relationship)

$join_class : Active Record Join Class (Parent / Aggregated)

$foreign_key_parent : Foreign key in Join Class to parent object
$foreign_key_child : Foreign key in Join Class to child object
$id : Primary key of parent object
$objects : Array of Active Records to be saved

public static function find($id)
Find a Active Record and returns it

public static function all()
Returns all objects

public function save()
Save the object

public static function getIndexedArray($indexColumn, $valueColumn, $criteria = NULL)
Creates an indexed array

public static function where($variable, $operator, $value, $logicOperator = TExpression::AND_OPERATOR)
Creates a Repository with filter

public static function orWhere($variable, $operator, $value)
Creates a Repository with OR filter

public static function orderBy($order, $direction = 'asc')
Creates an ordered repository

$order : = Order column
$direction : = Order direction (asc, desc)

TRepository

Implements the Repository Pattern to deal with collections of Active Records

Methods

public function __construct($class)
Class Constructor

$class : = Active Record class name

protected function getEntity()
Returns the name of database entity

public function where($variable, $operator, $value, $logicOperator = TExpression::AND_OPERATOR)
Add a run time criteria using fluent interfaces

$variable : = variable
$operator : = comparison operator (>,
$value : = value to be compared
$logicOperator : = logical operator (TExpression::AND_OPERATOR, TExpression::OR_OPERATOR)

public function set($column, $value)
Assign values to the database columns

$column : = column name
$value : = column value

public function orWhere($variable, $operator, $value)
Add a run time OR criteria using fluent interfaces

$variable : = variable
$operator : = comparison operator (>,
$value : = value to be compared

public function orderBy($order, $direction = 'asc')
Define the ordering for criteria using fluent interfaces

$order : = Order column

$direction : = Order direction (asc, desc)

public function take($limit)
Define the LIMIT criteria using fluent interfaces
$limit : = Limit

public function skip($offset)
Define the OFFSET criteria using fluent interfaces
$offset : = Offset

public function load(TCriteria $criteria = NULL, $callObjectLoad = TRUE)
Load a collection of objects from database using a criteria
$criteria : An TCriteria object, specifiyng the filters
$callObjectLoad : If load() method from Active Records must be called to load object parts

public function getIndexedArray($indexColumn, $valueColumn, $criteria = NULL)
Return a indexed array

public function update($setValues = NULL, TCriteria $criteria = NULL)
Update values in the repository

public function delete(TCriteria $criteria = NULL, $callObjectLoad = FALSE)
Delete a collection of Active Records from database
$criteria : An TCriteria object, specifiyng the filters

public function count(TCriteria $criteria = NULL)
Return the amount of objects that satisfy a given criteria
$criteria : An TCriteria object, specifiyng the filters

public function get(TCriteria $criteria = NULL, $callObjectLoad = TRUE)

TRequiredValidator extends TFieldValidator

Required field validation

Methods

public function validate($label, $value, $parameters = NULL)
Validate a given value
$label : Identifies the value to be validated in case of exception
$value : Value to be validated
$parameters : aditional parameters for validation

TScript

Base class for scripts

Methods

public static function create($code, $show = TRUE)
Create a script
$code : source code

TScroll extends TElement

Scrolled Window: Allows to add another containers inside, creating scrollbars when its content is bigger than its visual area

Methods

public function __construct()

Class Constructor

public function setSize($width, $height)
Set the scroll size

$width : *Panel's width*
$height : *Panel's height*

public function setMargin($margin)
Set the scrolling margin

$margin : *Margin*

public function setTransparency($bool)
compability reasons

public function show()
Shows the tag

TSeekButton extends TEntry

Record Lookup Widget: Creates a lookup field used to search values from associated entities

Methods

public function __construct($name, $icon = NULL)
Class Constructor

$name : *name of the field*

public static function createButton($name, $icon)
Create seek button object

public function setUseOutEvent($bool)
Define it the out event will be fired

public function setAction(TAction $action)
Define the action for the SeekButton

$action : *Action taken when the user clicks over the Seek Button (A TAction object)*

public function setAuxiliar($object)
Define an auxiliar field

$object : *any TField object*

public static function enableField($form_name, $field)
Enable the field

$form_name : *Form name*
$field : *Field name*

public static function disableField($form_name, $field)
Disable the field

$form_name : *Form name*
$field : *Field name*

public function show()
Show the widget

TSelect extends TField

Select Widget

Methods

public function __construct($name)
Class Constructor

$name : widget's name

public function disableMultiple()
Disable multiple selection

public function setDefaultOption($option)

public function addItems($items)
Add items to the select

$items : An indexed array containing the combo options

public function getItems()
Return the items

public function setSize($width, $height = NULL)
Define the Field's width

$width : Field's width in pixels
$height : Field's height in pixels

public function setValueSeparator($sep)
Define the field's separator

$sep : A string containing the field's separator

public function setValue($value)
Define the field's value

$value : A string containing the field's value

public function getPostData()
Return the post data

public function setChangeAction(TAction $action)
Define the action to be executed when the user changes the combo

$action : TAction object

public function setChangeFunction($function)
Set change function

public static function reload($formname, $name, $items, $startEmpty = FALSE)
Reload combobox items after it is already shown

$formname : form name (used in gtk version)
$name : field name
$items : array with items
$startEmpty : ...

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public static function clearField($form_name, $field)
Clear the field

$form_name : Form name
$field : Field name

protected function renderItems($with_titles = true)
Render items

public function show()

Shows the widget

TSession

Session Data Handler

Methods

public function __construct(SessionHandlerInterface $handler = NULL, $path = NULL)
Class Constructor

public static function enabled()
Returns if the service is active

public static function setValue($var, $value)
Define the value for a variable

$var : Variable Name
$value : Variable Value

public static function getValue($var)
Returns the value for a variable

$var : Variable Name

public static function delValue($var)
Clear the value for a variable

$var : Variable Name

public static function regenerate()
Regenerate id

public static function clear()
Clear session

public static function freeSession()
Destroy the session data Backward compatibility

TSlider extends TField

Slider Widget

Methods

public function __construct($name)
Class Constructor

$name : Name of the widget

public function setRange($min, $max, $step)
Define the field's range

$min : Minimal value
$max : Maximal value
$step : Step value

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public function show()
Shows the widget at the screen

public function setValue($value)
Set the value

TSortList extends TField

A Sortable list

Methods

public function __construct($name)
Class Constructor

$name : widget's name

public function setOrientation($orientation)
Define orientation

$orienatation : (horizontal, vertical)

public function setLimit($limit)
Define limit

public function setItemIcon(TImage $icon)
Define the item icon

$image : Item icon

public function setSize($width, $height = NULL)
Define the list size

public function setValue($value)
Define the field's value

$value : An array the field's values

public function connectTo(TSortList $list)
Connect to another list

$list : Another TSortList

public function addItems($items)
Add items to the sort list

$items : An indexed array containing the options

public function getItems()
Return the sort items

public function getPostData()
Return the post data

public function setChangeAction(TAction $action)
Define the action to be executed when the user changes the combo

$action : TAction object

public function setChangeFunction($function)
Set change function

public static function enableField($form_name, $field)
Enable the field

public static function disableField($form_name, $field)
Disable the field

public static function clearField($form_name, $field)
Clear the field

public function show()
Shows the widget at the screen

TSourceCode
SourceCode View

Methods

public function loadFile($file)
Load a PHP file

$file : Path to the PHP file

public function loadString($content)
Load from string

public function show()
Show the highlighted source code

TSpinner extends TField
Spinner Widget (also known as spin button)

Methods

public function __construct($name)
Class Constructor

$name : Name of the widget

public function setRange($min, $max, $step)
Define the field's range

$min : Minimal value
$max : Maximal value
$step : Step value

function setExitAction(TAction $action)
Define the action to be executed when the user leaves the form field

$action : TAction object

public static function enableField($form_name, $field)
Enable the field

$form_name : Form name
$field : Field name

public static function disableField($form_name, $field)
Disable the field

$form_name : Form name
$field : Field name

public function setExitFunction($function)
Set exit function

public function show()
Shows the widget at the screen

public function setValue($value)
Set the value

TSqlDelete extends TSqlStatement

Provides an Interface to create DELETE statements

Methods

public function getInstruction($prepared = FALSE)
Returns a string containing the DELETE plain statement
$prepared : Return a prepared Statement

TSqlInsert extends TSqlStatement

Provides an Interface to create an INSERT statement

Methods

public function setRowData($column, $value)
Assign values to the database columns
$column : Name of the database column
$value : Value for the database column

public function setCriteria(TCriteria $criteria)
this method doesn't exist in this class context
$criteria : A TCriteria object, specifiyng the filters

public function getPreparedVars()
Return the prepared vars

public function getInstruction($prepared = FALSE)
Returns the INSERT plain statement
$prepared : Return a prepared Statement

TSqlSelect extends TSqlStatement

Provides an Interface to create SELECT statements

Methods

public function addColumn($column)
Add a column name to be returned
$column : A string containing a column name

public function getInstruction($prepared = FALSE)
Returns the SELECT statement as an string according to the database driver
$prepared : Return a prepared Statement

public function getStandardInstruction($prepared)
Returns the SELECT statement as an string for standard open source drivers
$prepared : Return a prepared Statement

public function getInterbaseInstruction($prepared)
Returns the SELECT statement as an string for standard open source drivers
$prepared : Return a prepared Statement

public function getSqlServerInstruction($prepared)
Returns the SELECT statement as an string for mssql/dblib drivers
$prepared : Return a prepared Statement

public function getOracleInstruction($prepared)
Returns the SELECT statement as an string for oci8 drivers

$prepared : Return a prepared Statement

TSqlStatement

Provides an abstract Interface to create a SQL statement

Methods

final public function setEntity($entity)
defines the database entity name
$entity : Name of the database entity

final public function getEntity()
Returns the database entity name

public function setCriteria(TCriteria $criteria)
Define a select criteria
$criteria : An TCriteria object, specifiyng the filters

protected function getRandomParameter()
Returns a random parameter

abstract function getInstruction();

TSqlUpdate extends TSqlStatement

Provides an Interface to create UPDATE statements

Methods

public function setRowData($column, $value)
Assign values to the database columns
$column : Name of the database column
$value : Value for the database column

public function getPreparedVars()
Return the prepared vars

public function getInstruction($prepared = FALSE)
Returns the UPDATE plain statement
$prepared : Return a prepared Statement

TStandardForm extends TPage

Standard page controller for forms

Methods

public function onSave()
method onSave() Executed whenever the user clicks at the save button

public function onClear($param)
Clear form

public function onEdit($param)
method onEdit() Executed whenever the user clicks at the edit button da datagrid
$param : An array containing the GET ($_GET) parameters

public function setDatabase($database)
method setDatabase() Define the database

public function setActiveRecord($activeRecord)
method setActiveRecord() Define wich Active Record class will be used

TStandardFormList extends TPage

Standard page controller for form/listings

Methods

public function setLimit($limit)
method setLimit() Define the record limit

public function setDefaultOrder($order, $direction = 'asc')
Define the default order

$order : The order field
$directiont : the order direction (asc, desc)

public function setCriteria($criteria)
method setCriteria() Define the criteria

public function setTransformer($callback)
Define a callback method to transform objects before load them into datagrid

public function onReload($param = NULL)
method onReload() Load the datagrid with the database objects

public function onSave()
method onSave() Executed whenever the user clicks at the save button

public function onDelete($param)
method onDelete() executed whenever the user clicks at the delete button Ask if the user really wants to delete the record

public function Delete($param)
method Delete() Delete a record

public function onClear($param)
Clear form

public function onEdit($param)
method onEdit() Executed whenever the user clicks at the edit button da datagrid

public function show()
Shows the page

public function setDatabase($database)
method setDatabase() Define the database

public function setActiveRecord($activeRecord)
method setActiveRecord() Define wich Active Record class will be used

TStandardList extends TPage

Standard page controller for listings

Methods

public function setLimit($limit)
method setLimit() Define the record limit

public function enableTotalRow()
Enable total row

public function setDefaultOrder($order, $direction = 'asc')
Define the default order

$order : The order field
$directiont : the order direction (asc, desc)

public function setFilterField($filterField)
method setFilterField() Define wich field will be used for filtering PS: Just for Backwards compatibility

public function setOperator($operator)
method setOperator() Define the filtering operator PS: Just for Backwards compatibility

public function addFilterField($filterField, $operator = 'like', $formFilter = NULL, $filterTransformer = NULL)
method addFilterField() Add a field that will be used for filtering

$filterField : Field name
$operator : Comparison operator

public function setCriteria($criteria)
method setCriteria() Define the criteria

public function setTransformer($callback)
Define a callback method to transform objects before load them into datagrid

public function onInlineEdit($param)
Inline record editing

$param : Array containing: key: object ID value field name: object attribute to be updated value: new attribute content

public function onSearch()
Register the filter in the session

public function onReload($param = NULL)
method onReload() Load the datagrid with the database objects

public function onDelete($param)
Ask before deletion

public function Delete($param)
Delete a record

public function onDeleteCollection($param)
Ask before delete record collection

public function deleteCollection($param)
method deleteCollection() Delete many records

public function show()
method show() Shows the page

public function setDatabase($database)
method setDatabase() Define the database

public function setActiveRecord($activeRecord)
method setActiveRecord() Define wich Active Record class will be used

TStandardSeek extends TWindow

Standard Page controller for Seek buttons

Methods

public function __construct()
Constructor Method Creates the page, the search form and the listing

public function onSearch()
Register the user filter in the section

public function onReload($param = NULL)
Load the datagrid with the active record objects

public function onSetup($param=NULL)
define the standars seek parameters

public static function onSelect($param)
Select the register by ID and return the information to the main form When using onblur signal, AJAX passes all needed parameters via GET instead of calling onSetup before.

TStyle

StyleSheet Manager

Methods

public function __construct($name)
Class Constructor
$mame : Name of the style

public function getName()
Returns the style name

public static function findStyle($object)
Find a style by its properties

public function hasContent()
Return if the style has any content

public function getContent()
Returns the style content

public function getInline()
Return the style inline code

public function show($inline = FALSE)
Show the style

TTable extends TElement

Creates a table layout, with rows and columns

Methods

public function __construct()
Class Constructor

public function addSection($type)
Add section

public function addRow()
Add a new row (TTableRow object) to the table

public function addRowSet()
Add a new row (TTableRow object) with many cells

$cells : Each argument is a row cell

public static function fromData($array_data, $table_properties = null, $header_properties = null, $body_properties = null)
Create a table from data array

$array_data : Array with raw data
$table_properties : Array of CSS properties for table
$header_properties : Array of CSS properties for header
$body_properties : Array of CSS properties for body

TTableCell extends TElement

TableCell: Represents a cell inside a table

Methods

public function __construct($value)
Class Constructor

$value : TableCell content

TTableRow extends TElement

TableRow: Represents a row inside a table

Methods

public function __construct()
Class Constructor

public function addCell($value)
Add a new cell (TTableCell) to the Table Row

$value : Cell Content

public function addMultiCell()
Add a multi-cell content to a table cell

$cells : Each argument is a row cell

public function clearChildren()
Clear any child elements

TText extends TField

Text Widget (also known as Memo)

Methods

public function __construct($name)
Class Constructor

$name : Widet's name

public function setSize($width, $height = NULL)
Define the widget's size

$width : Widget's width
$height : Widget's height

public function getSize()
Returns the size

function setExitAction(TAction $action)
Define the action to be executed when the user leaves the form field

$action : TAction object

public function setExitFunction($function)
Set exit function

public function show()
Show the widget

TTextDisplay extends TElement

Text Display

Methods

public function __construct($value, $color = null, $size = null, $decoration = null)
Class Constructor

$value : text content
$color : text color
$size : text size
$decoration : text decorations (b=bold, i=italic, u=underline)

TTransaction

Manage Database transactions

Methods

public static function open($database, $dbinfo = NULL)
Open a connection and Initiates a transaction

$database : Name of the database (an INI file).
$dbinfo : Optional array with database information

public static function get()
Returns the current active connection

public static function rollback()
Rollback all pending operations

public static function close()
Commit all the pending operations

public static function setLoggerFunction(Closure $logger)
Assign a Logger closure function

$logger : A Closure

public static function setLogger(AdiantiLoggerInterface $logger)
Assign a Logger strategy

$logger : A TLogger child object

public static function log($message)
Write a message in the LOG file, using the user strategy

$message : Message to be logged

public static function getDatabase()
Return the Database Name

public static function getDatabaseInfo()
Returns the Database Information

TTreeView extends TElement

TreeView

Methods

public function __construct()
Class Constructor

public function setTransformer($callback)
Set node transformer

public function setSize($width)
Set size
$size : width

public function setItemIcon($icon)
Set item icon
$icon : icon location

public function setItemAction($action)
Set item action
$action : icon action

public function collapse()
Collapse the Tree

public function expandTo($key)
Expand to Tree Node
$key : Node key

public function fromArray($array)
Fill treeview from an multi-dimensional array
multi-dimensional : array

public function show()
Shows the tag

TUIBuilder

Interface builder that takes a XML file save by Adianti Studio Designer and renders the form into the interface.

Methods

public function __construct($width, $height)
Class Constructor
$width : Panel width
$height : Panel height

public function getActions()
Return the found actions

public function parseFile($filename)
Parse XML form file
$filename : XML form file path

public function makeTLabel($properties)

public function makeTButton($properties)

public function makeTEntry($properties)

public function makeTSpinner($properties)

public function makeTSlider($properties)

public function makeTPassword($properties)

public function makeTDate($properties)

public function makeTFile($properties)

public function makeTColor($properties)

public function makeTSeekButton($properties)

public function makeTImage($properties)

public function makeTText($properties)

public function makeTCheckGroup($properties)

public function makeTDBCheckGroup($properties)

public function makeTRadioGroup($properties)

public function makeTDBRadioGroup($properties)

public function makeTCombo($properties)

public function makeTDBCombo($properties)

public function makeTSelect($properties)

public function makeTDBSelect($properties)

public function makeTSortList($properties)

public function makeTDBSortList($properties)

public function makeTMultiSearch($properties)

public function makeTDBMultiSearch($properties)

public function makeTNotebook($properties)

public function makeTFrame($properties)

public function makeTDataGrid($properties)

public function setController($object)
Defines the UI controller
$object : Controller Object

public function setForm($form)
Defines the Parent Form
$object : TForm

public function getFields()
Return the UI widgets (form fields)

public function getWidgets()
Return the parsed widgets

public function getWidget($name)
Return the widget by name
$name : Widget name

TUniqueSearch extends TMultiSearch

Unique Search Widget

Methods

public function __construct($name)
Class Constructor
$name : Widget's name

public function setValue($value)
Set value

public function getPostData()
Return the post data

TVBox extends TElement

Vertical Box

Methods

public function __construct()
Class Constructor

public function add($child)
Add an child element
$child : Any object that implements the show() method

public function addColSet()
Add a new col with many cells
$cells : Each argument is a row cell

public static function pack()
Static method for pack content
$cells : Each argument is a cell

TWindow extends TPage

Window Container (JQueryDialog wrapper)

Methods

public function __construct()

public static function create($title, $width, $height, $params = null)
Create a window

public function setStackOrder($order)
Define the stack order (zIndex)
$order : Stack order

public function setTitle($title)
Define the window's title
$title : Window's title

public function setModal($modal)
Turn on/off modal
$modal : Boolean

public function setSize($width, $height)
Define the window's size
$width : Window's width
$height : Window's height

public function setPosition($x, $y)
Define the top corner positions

$x : left coordinate
$y : top coordinate

public function setProperty($property, $value)
Define the Property value

$property : Property name
$value : Property value

public function add($content)
Add some content to the window

$content : Any object that implements the show() method

public static function closeWindow()
Close TJQueryDialog's

TXMLBreadCrumb extends TBreadCrumb

XMLBreadCrumb

Methods

public function __construct($xml_file, $controller)
Handle paths from a XML file

$xml_file : path for the file

public function getPath($controller)
Return the controller path