Elefant Cheat Sheet

github.com/jbroadw ay/elefant

Getting Started

These are the steps to get a basic Elefant installation up and running.

```
git clone http://github.com/jbroadway/elefant.git
cd elefant
vi conf/config.php
chmod -R 777 cache conf
php conf/createdb.php
```

Utilities

`php conf/make.php appname`	Create an app skeleton
`php conf/createdb.php`	Initialize an SQLite database
`phpunit tests/Form.php`	Run one of the unit test sets

Request Routing

`/`	→	`$conf['General']['default_handler']`
`/appname`	→	`apps/appname/handlers/index.php`
`/appname/handler`	→	`apps/appname/handlers/handler.php`
`/appname/handler/extra`	→	`apps/appname/handlers/handler/extra.php`

Cascade occurs in reverse order until a matching handler is found.

Folder Structure

`apps`	Your applications
`cache`	Cached templates
`conf`	Configuration files
`css`	Stylesheets
`index.php`	Front controller
`js`	Javascript files
`lang`	Translations
`layouts`	Design templates
`lib`	Core classes
`tests`	Unit tests

App Structure

`apps/appname/`

`conf`	Configuration files
`forms`	Form validations
`handlers`	Request handlers
`lib`	Custom classes
`models`	Data models
`views`	App templates

Global Objects

`$conf`	Array of conf/config.php
`$controller`	Controller object
`$db`	PDO object
`$i18n`	I18n object
`$page`	Page object
`$tpl`	Template object

Core Classes

Useful properties and methods of the core classes.

Controller

`->run($uri)`	Resolves and returns the output of a handler.
`->handle($handler)`	Returns the output of a handler.
`->route($uri)`	Resolves a handler from URI to filename.

Database functions

All functions except `db_error()` and `db_lastid()` accept `$sql` plus an array or variable number of values for replacement of placeholders, for example:

```
$res = db_execute('select * from mytable where id = ?', $id);
```

`db_execute()`	Execute a statement and return true/false.
`db_single()`	Fetch a single object.
`db_shift()`	Fetch the a single value from the first result returned.
`db_fetch_array()`	Fetch an array of all result objects.
`db_shift_array()`	Fetch an array of a single field.
`db_pairs()`	Fetch an associative array of tw o fields.
`db_lastid()`	Get the last inserted id value (int).
`db_error()`	Get the last error message, or false if no error.

Form

`$f = new Form($method, $rules)`	Constructor method.

`if ($f->submit()) {`
`  // handle form`
`}`
	Is the form submitted and valid?

`Form::verify_value($value, $type, $validator)`	Validate a data value, in or out of the context of a form.
`Form::merge_values($obj)`	Merge `$_GET` or `$_POST` values onto an object for pre-filling a re-rendered form.
`$failed = array()`	Fields that failed validation.
`$method = 'post'`	Required request method.
`$rules = array()`	Validation rules.
`$verify_referrer = true`	Whether to verify the referrer.
`$error = false`	The reason submit() failed to pass.

I18n

`i18n_get($original)`	Translate a string into the current user's language.
`i18n_getf($orig, $values)`	Like `i18n_get()` w ith `vsprintf()` post-processing.
`$language`	The current language.
`$locale`	The current locale.
`$charset = 'UTF-8'`	The current language's character set.
`$fullname`	The full name of the current language.
`$url_includes_lang`	Whether the URL includes the language, e.g., `/fr/index`.
`$new_request_uri`	The URL w ith the prefixed language removed.
`$negotiation`	The negotiation method used to determine the current language.
`$error`	Any error that occurs in the class.

Model

Usage:

`class Mytable extends Model {}`	Create model for `mytable` db table.

`$m = new Mytable(array(`
`  'name'=>'John',`
`  'age'=>30`
`));`
`$m->put();`
	Insert a new object.

`$m->id`	The ID from the inserted object.

`$m->age = 31;`
`$m->put();`
	Update the object.

`$m->remove()`	Delete the object.

`$res = $m->query()`
`  ->where('age > 30')`
`  ->order('name asc')`
`  ->fetch(20, 0);`
	Build a custom query and fetch the results. `$res` is an array of `Mytable` objects.

Page

`->render()`	Renders the page w ith its view and layout templates.
`$body = ''`	Body content.
`$head = ''`	Extra HTML headers.
`$layout = 'default'`	Design template.
`$template = ''`	View template.
`$title = ''`	Page title.

Template

`->render($template, $data)`	Render a template file.
`$charset = 'UTF-8'`	Template character set.
`{{ var }}`	In-template variable substitution.
`{{ var|filter_name }}`	Variable filtering.
`{" Text "}`	In-template translatable text.

`{% foreach foo %}`
`{% end %}`
	In-template looping.

`{% if foo %}`
`{% elseif bar %}`
`{% else %}`
`{% end %}`
	In-template conditional logic.

User

Extends `Model` w ith the follow ing:

`User::encrypt_pass($plain)`	Generates a random salt and encrypts a string using MD5.
`User::require_login()`	Requires a user to be logged in.
`User::is_valid()`	Check if the user is valid.
`User::logout($redirect_to)`	Log the current user out.