PHP ON COUCH

Introducción

Php On Couch es una librería en php para atacar bases de datos, en adelante bbdd, de CouchDB . Por el uso que le he dado he podido comprobar su estabilidad y ademas se actualiza regularmente, lo cual es de agradecer. Cabe decir también que esta bajo licencia GPL.

En este ejemplo veremos como conectar a una base de datos y extraer la información de un documento, obviamente esta librería ofrece muchísimas más funcionalidades, pero este post no pretende ser una explicación de cada una de ellas. Pretende mostrar una utilización práctica de la librería, en él se creará un gráfico con Google Chart Api a partir de datos guardados en CouchDB.

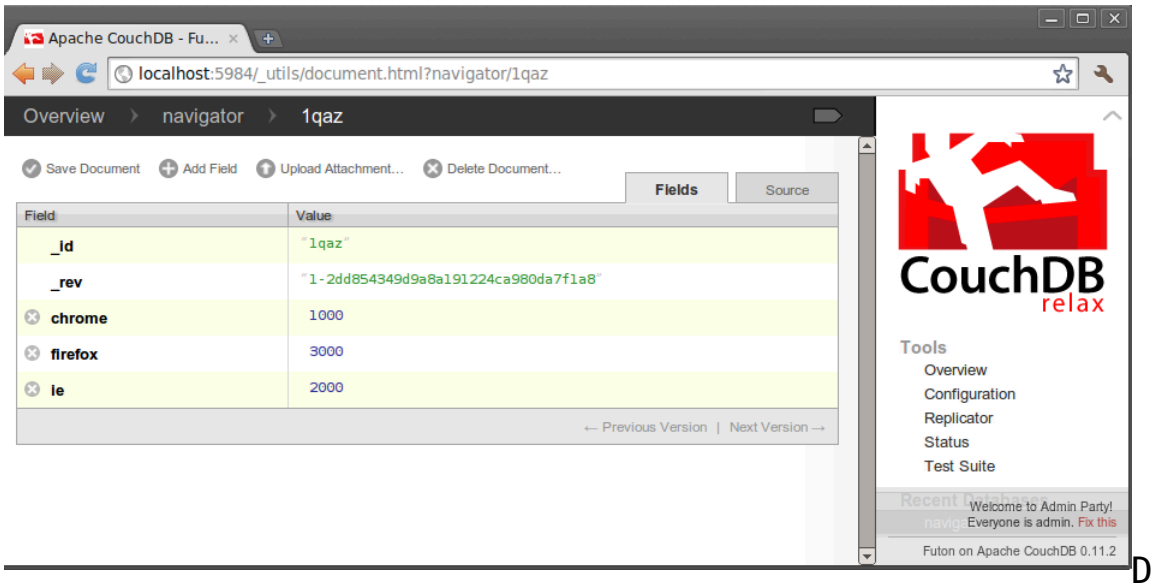
Estructura

Dispondremos de los siguiente ficheros:

- index.php: archivo principal.
- couch.php: clase que utilizaremos para acceder a CouchDB.
- config.php: archivo de configuración.

Crearemos una base de datos con las siguientes características:

- El nombre de la base de datos será navigator.
- El id del documento al que accederemos tendrá el siguiente id: 1qaz.
- El documento almacenara un valor para los siguientes tags: ie, firefox y chrome.



Debemos descargarnos la librería Php On CouchDB y estructurar el proyecto conforme la la carpeta lib, de dicha librería, se encuentre en el mismo nivel que los archivos PHP.

index.php

Este archivo incluirá la la clase couch.php, la instanciará y llamará al método chart para que este le retorne el gráfico.

```
/**
 * User: rmiguel
 * Date: 22/07/2015
 * Time: 20:00
 */

require_once "CouchWrapper.php";

print "

                    Couchdb Test

«;    $con    =    new    CouchWrapper();    print
$con->chart(«600×200″,»1qaz»);
```

config.php

Este archivo es el típico utilizado para parametrizar las WebApp que tienen acceso a bbdd, no comentare el archivo por su simpleza y los descriptivos nombres de las variables. Dado

que esto es un ejemplo, me permito la licencia de utilizar variables globales, no cabe decir que en una aplicación de uso está sería una mala implementación de uso para un archivo de configuración dada la información que contiene. Personalmente yo creo un xml que parseo con un método privado en la clase, el cual guarda el valor de los tags en variables privadas. De esta manera aseguramos que este método y las variables son usadas únicamente por la clase protegiéndolas dada la importancia de su valor.

```
/**
 * User: rmiguel
 * Date: 22/07/2015
 * Time: 20:00
 */
//Couch_db config
$COUCHDB_HOST = "127.0.0.1";
$COUCHDB_PORT = "5984";
$COUCHDB_TABLE = "navigator";
```

couch.php

Este archivo es el que contiene la clase _couch. Primero incluimos las dependencias (de la librería y el archivo de configuración) .

A continuación explicaremos la estructura de la classe:

Variables

- $client será donde se guardará la instancia cliente que ataca a la bbdd.
- $couchdsn será la url y el puerto (http://127.0.0.1:5894)
- $couch_db será la base de datos (navigator)

Métodos

- _couch -> es el constructor en él se realizará la conexión a la bbdd.
- get_doc($id) -> retornara una variable con la estructura

del documento correspondiente al id que se le pasa por parámetro.

- chart -> retornará una url del tipo pie chart que atacara a google chart api para crear un gráfico en función del id de documento y la resolución que se especifique.

```
/**
 * User: rmiguel
 * Date: 22/07/2015
 * Time: 20:00
 */
//-----------------------
require_once "lib/couch.php";
require_once "lib/couchClient.php";
require_once "lib/couchDocument.php";
require_once "config.php";
//-----------------------------------
class CouchWrapper {
    public $client;
    public $couch_dns;
    public $couch_db;
    /**
     * Constructor
     */
    function __construct(){
                                                $couch_dsn
=".$GLOBALS['COUCHDB_HOST'].":".$GLOBALS['COUCHDB_PORT'];
        $couch_db = $GLOBALS['COUCHDB_TABLE'];
        try{
                                    $this->client   =   new
couchClient($couch_dsn,$couch_db);
        } catch (Exception $e) {
                        echo  "Error:".$e->getMessage()."
(errcode=".$e->getCode().")\n";
        }
    }
    /**
     * @param $id
     * @return document values
     */
```

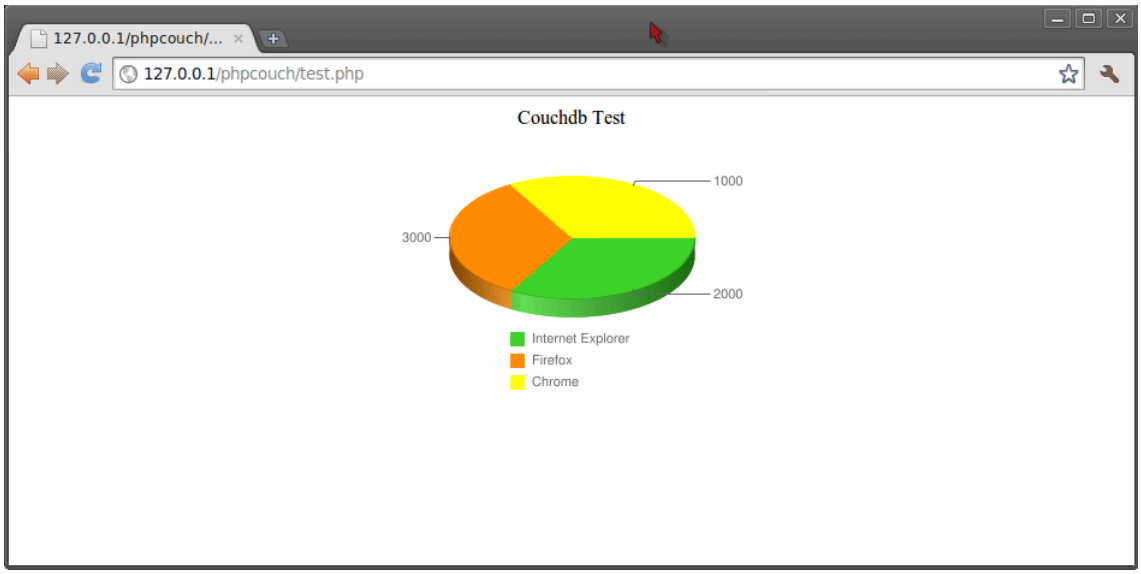
```
    function get_doc($id){
        try{
            $doc = $this->client->getDoc($id);
        } catch (Exception $e) {
            if ( $e->code() == 404 ) {
                echo "Document \"some_doc\" not found\n";
            } else {
                        echo "Something weird happened:
".$e->getMessage()." (errcode=".$e->getCode().")\n";
            }
            exit(1);
        }
        return $doc;
    }
    /**
     * @param $resolution
     * @param $id_document
     * @return chart image tag
     */
    function chart($resolution,$id_document){
        $doc = $this->get_doc($id_document);
        $green = $doc->ie;
        $orange = $doc->firefox;
        $yellow = $doc->chrome;
        $chart = "
```



```
«; return $chart; } }
```

Resultado

Una vez ejecutemos en nuestro servidor web index.php deberíamos obtener este resultado si todo ha funcionado correctamente:



Observaciones

Bien con este post doy por finalizada la breve introducción al mundo de las bbdd NoSql, espero y deseo que hayan sido de ayuda.

Posts relacionados

-> NoSql

-> NoSql – CouchDB

Código fuente en GitHub

PHPonCouch GitHub

Fuentes

- Wiki couchdb -> Getting_started_with_PHP
- Google Api Char

Ruben