

Revista Digital

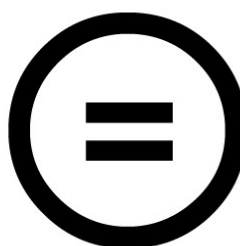
ATIX

**Regreso
con Fuerza**



 **Libre**
atix
Fundación

Distribuido bajo:



2013 - Bolivia



<http://revista.atixlibre.org>

Twitter: @atixlibre

Facebook: facebook.com/Atix.Libre



Dirección y Coordinación General

Esteban Saavedra López (esteban.saavedra@atixlibre.org)

Diseño y Maquetación

Jenny Saavedra López (jenny.saavedra@atixlibre.org)

Esteban Saavedra López (esteban.saavedra@atixlibre.org)

Revisiones

Esteban Saavedra López

Jenny Saavedra López

Noticias

Jenny Saavedra López

Autores Frecuentes

Esteban Saavedra López

Ernesto Rico Smith

Martín Márquez

Rafael Rendón

Herramientas

La edición de esta revista fue realizada de forma integra haciendo uso de Software Libre





atlix

**El que lo
Intenta**

**El que lo
Sabe**

**El que lo
Puede**

**El que lo
Logra**

Regreso con Fuerza

Después de un tiempo sin publicar nuevos números de nuestra revista, debido a varios factores que dificultaron nuestro emprendimiento; dificultades que en este momento no vale la pena recordarlas, pero si aprender de ellas, nuestro equipo decidió pasar página y encarar un regreso, mentalizados con toda la fuerza necesaria que nos ayude a continuar el objetivo que hace unos años nos fijamos.

Se dice que un buen luchador no es aquel que pocas veces o nunca cayó, sino aquel que cayó muchas veces e igual número de veces supo levantarse.

Regreso con fuerza, un título que representa mucho para nosotros, ya que todos los miembros de nuestro equipo trabajamos duro para dedicarle un poco de tiempo a este proyecto, que si bien no es remunerado, pero nos llena de satisfacción de poder proveer a nuestros lectores un medio de difusión, aprendizaje y colaboración en todo lo referente al software, conocimiento, tecnología y cultura libre.

En este primer número del 2013, estrenamos un nuevo diseño y presentación, pero no todo queda ahí, ya que en los próximos números iremos mostrando algunas nuevas secciones con diversos contenidos.

Asimismo con este primer número del 2013, queremos dar continuidad a nuestro trabajo que desde que iniciamos con el proyecto de la revista siempre lo hicimos con la mejor intensidad y la mayor voluntad.

Bienvenidos a nuestro vigésimo número



Esteban Saavedra López
Presidente Fundación AtixLibre

{CONTENIDO}

7 Play Web Framework

14 Trabajando con Binary Large Object (BLOB)
en PostgreSQL con MonoDevelop

23 Luz, Cámara y Orientémoslo al Móvil

31 Herramientas de un Desarrollador Linux
tmux

38 Tratamiento de Textos en Perl Primeros
pasos rumbo al Textmining

48 Exceptions
Clases de errores incorporados

61 Willay news

64 Arte Libre

66 Información de contacto

Play Web Framework

En el mundo de desarrollo de software, el uso de los frameworks se ha convertido en una herramienta poderosa, eficiente y eficaz al momento de afrontar proyectos de desarrollo de software, por su fácil aprendizaje, rápido desarrollo y estructura robusta.



Introducción

Actualmente el mundo del desarrollo de software está muy orientado hacia la construcción de entornos web, haciendo que aplicaciones en esta categoría sean cada vez más requeridas, de forma que su desarrollo sea simple, en menor tiempo y con calidad incremental a precios razonables.

El crecimiento exponencial de plataformas de desarrollo web como Rails se debe principalmente a su capacidad de elaborar aplicaciones en docenas de horas con muy poco esfuerzo. Pero este crecimiento ha demostrado que Rails tiene limitaciones inherentes a su plataforma y arquitectura con respecto a dos temas muy importantes: rendimiento y escalabilidad.

Estos aspectos son puntos fuertes en otras plataformas tales como .Net y Java, pero éstas tienen también 'incomodidades' debido a la naturaleza de estos lenguajes (estáticamente tipados y compilables), tales como la cantidad de esfuerzo requerido para obtener resultados y la falta de un esquema de desarrollo integrado con la visión y consistencia que Rails tiene.

Para reducir esta brecha entre plataformas eficientes y escalables, múltiples proyectos libres han optado por usar una aproximación mixta, usar las fortalezas de la plataforma Java y todos los beneficios de los principios que Rails ha aplicado desde sus inicios, tales como Convención sobre Configuración y DRY.

Este artículo se centra en uno de los frameworks de código libre más prometedores de los últimos años llamado Play Web Framework, el cual tiene como objetivo los siguientes tópicos:

- ✓ Usar toda la potencia de Java, sus librerías y frameworks ampliamente usados en la industria.
- ✓ Aplicar los principios y buenas prácticas heredadas de Rails y Django: DRY, Convención sobre Configuración, etc.
- ✓ Permitir construir aplicaciones ligeras de forma diferente al estándar JEE5/6.
- ✓ Ser compatible con las plataformas de desarrollo web actuales (REST, SOA).
- ✓ Proveer un sistema **JITC**(Just In Time Compilation) para Java, el cual permite desarrollar y probar aplicaciones de forma rápida.
- ✓ Permitir definir ambiente de trabajo dentro de las aplicaciones: desarrollo, pruebas, integración.

Requisitos

Para el desarrollo de la aplicación se requieren los siguientes elementos:

- ✓ Java Development Kit 1.6.x (OpenJDK u Oracle JDK)
- ✓ Terminal de comandos, Bash(Unix), Command o PowerShell(Windows)
- ✓ Play 1.2.5
- ✓ Acceso a una terminal de comandos, Bash en Unix/Linux o Command en Windows.

Instalación

- ✓ Descargar Play 1.2.5 y descomprimirlo en una carpeta (por ejemplo `/opt/play12`).
- ✓ Agregar la carpeta descomprimida al PATH de su consola de comandos
 - ✓ **Unix:** Agregar la línea `export PATH=$PATH:/opt/play12` a su archivo `~/.bashrc`
 - ✓ **Windows:** Agregar el directorio a través de su utilidad de asignación de variables globales.

Primer ejemplo

Ahora demos un vistazo rápido a las capacidades de Play desarrollando una aplicación sencilla que nos muestre el uso completo de todo el stack de desarrollo del Play:

- ✓ Requerimos una aplicación que nos permita registrar tareas y marcarlas como completadas en un entorno web.

Primeramente definimos la estructura base de una aplicación vacía, para esto crearemos una aplicación genérica a través de la utilidad de línea de comandos de Play.

Desde la consola generamos el proyecto y accedemos a él:

```
$> play new Tareas
$> cd Tareas
$> play run
```

Accedemos a nuestra aplicación a través de la URL `http://localhost:9000` y veremos la siguiente página:

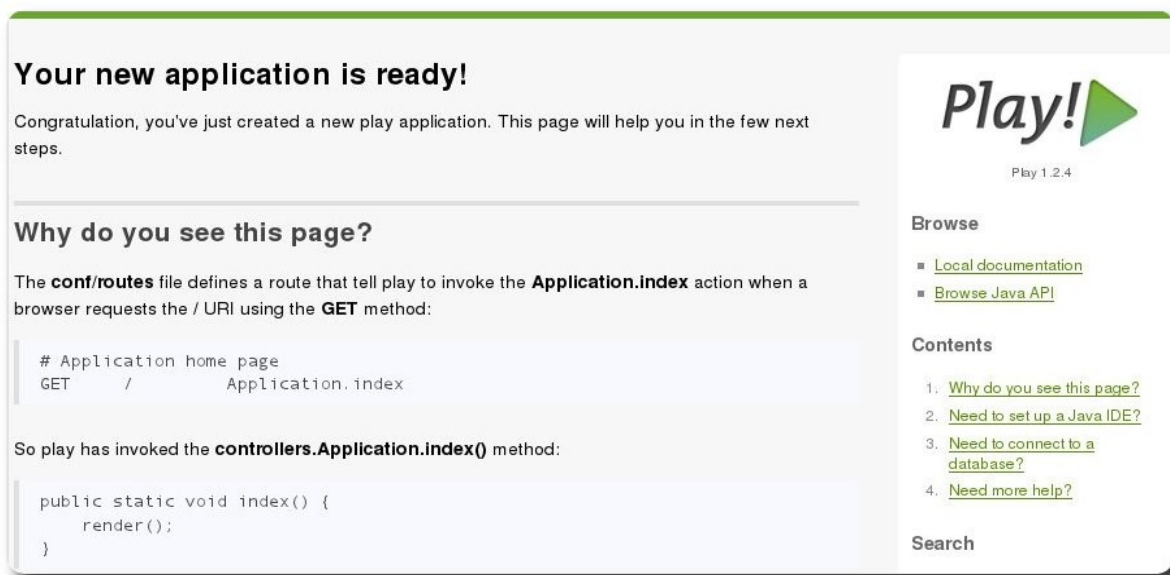


Figura 1.

Esta página inicial contiene toda la documentación local base para Play, incluyendo sus preguntas más frecuentes y guías de uso.

Estructura del proyecto

Ingresando en la nueva carpeta de proyecto creada, tenemos los siguientes directorios:

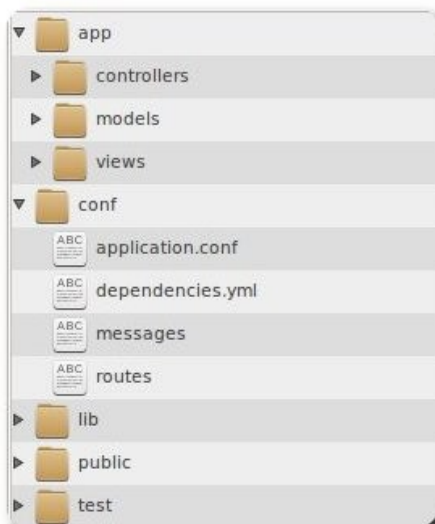


Figura 2. Estructura de Proyecto.

- ✓ **app:** Carpeta que aloja los principales componentes MVC de la aplicación.
- ✓ **controllers:** Componentes Java, controlando la interacción entre componentes visuales, redireccionando peticiones HTTP e interactuando como punto de acceso hacia los componentes del modelo.
- ✓ **models:** Componentes persistentes o no del dominio de la aplicación, por defecto Play usa el estándar JEE JPA 1.0 para la persistencia de datos.
- ✓ **views:** Plantillas HTML que permiten renderizar componentes obtenidos a través de los controladores. Play usa una sintaxis similar a Ruby para las plantillas de vistas.
- ✓ **conf:** Configuraciones de la aplicación, tales como conexiones a bases de datos, módulos adicionales, internacionalización, mapeo de rutas HTTP para interacciones REST, datos iniciales, etc.

- ✓ **lib:** Archivos librería adicionales (archivos JAR).
- ✓ **public:** Archivos Javascript, CSS e imágenes usadas en las páginas.
- ✓ **test:** Archivos de pruebas unitarias y pruebas de integración.

Todas las peticiones HTTP siguen el mismo ciclo de vida a través de los componentes MVC. Las interacciones se inician en las Vistas, las cuales envían señales al servidor, éste usa las Rutas configuradas para redireccionar todo a los respectivos Controladores, los cuales acceden a los elementos del Modelo, mismos que son renderizados (de ser necesario) por las vistas.

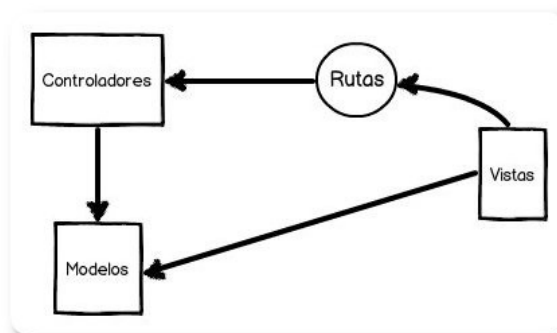


Figura 3. MVC

A partir de este punto cualquier modificación realizada al código y/o configuraciones es automáticamente aplicada al código al recargar la página. Esta es una de las características más trabajadas y útiles en Play ya que permite tener un entorno inmediato de desarrollo-resultado integrado con el framework.

Modelo de dominio

Como toda aplicación MVC empezamos definiendo el modelo, para nuestro caso usaremos una estructura sencilla, una clase persistente Tarea bajo la carpeta `app/models/Tarea.java`.

```
package models;

import play.*;
import play.db.jpa.*;
import play.data.validation.*;

import javax.persistence.*;
import java.util.*;

@Entity
public class Tarea extends Model{

    @Required
    public String titulo;
    public boolean completada;

    public Tarea(String titulo){
        this.titulo = titulo;
    }

    public String toString(){
        return id + " - " + titulo;
    }
}
```

Todos los componentes del modelo en Play extienden de la clase base `Model`, que permite definir clases que se mapean a tablas de forma automática. Adicionalmente tiene agregados de validación como el `@Required` usado para establecer un campo como requerido; nótese que el atributo ID es automáticamente definido en la clase base `Model`.

Para habilitar una base de datos de desarrollo configuramos Play para que la genere por nosotros en memoria, en el archivo `conf/application.conf` encuentre y descomente (quitar el carácter #):

```
db=mem
```

CRUD

Play nos permite generar estructuras CRUD (Create/Update/Delete) a través de uno de

Ahora acceder de nuevo a la dirección `http://localhost:9000` para verificar las operaciones CRUD, debería mostrar algo similar a la siguiente pantalla:

sus módulos de forma sencilla, para este cometido modificar el archivo `conf/dependencies.yml` para que quede de la siguiente forma:

```
require:
  - play
  - crud
```

Crear la siguiente clase en `/app/controllers/Tareas.java`:

```
package controllers;

import play.*;
import play.mvc.*;

import java.util.*;

import models.*;

public class Tareas extends CRUD{
}
```

Configurar las rutas de redirección de peticiones HTTP hacia componentes controladores en `conf/routes`:

Reemplazar

```
GET      /
Application.index
```

por

```
GET      /
Tareas.list
```

Finalmente instalar todas las dependencias cuando inicie la aplicación a través del comando:

```
$> play run --deps
```

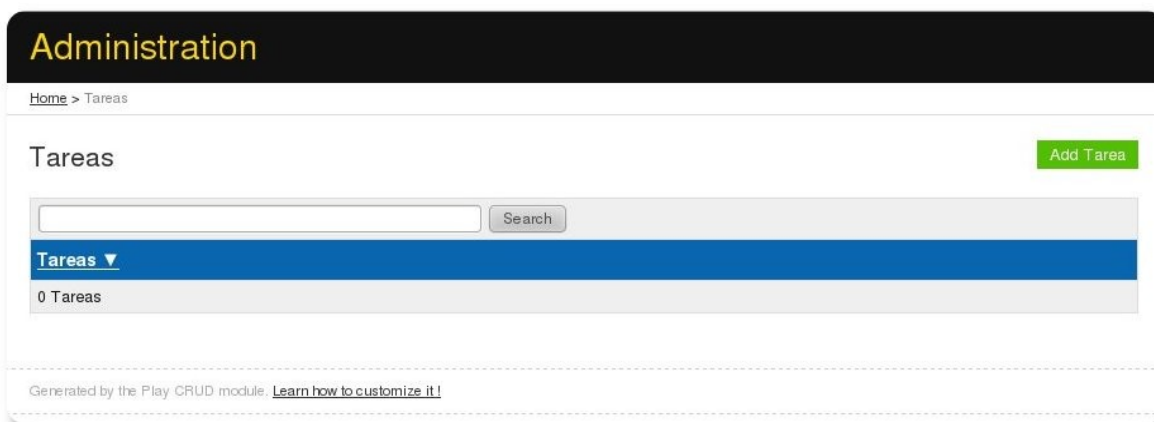


Figura 4. CRUD en funcionamiento

Ahora podemos crear tareas de forma simple sin haber invertido tiempo innecesario en tareas triviales como creación, modificación y borrado de Tareas. Además el módulo aplica todas las validaciones del elemento de dominio, por ejemplo, si intentamos crear una Tarea sin título tendremos el siguiente error:



Figura 5. Validaciones

El módulo CRUD provee las siguientes características:

- ✓ Búsqueda de elementos por contenido (todos los campos String en su clase de dominio).
- ✓ Listado maestro de los elementos.
- ✓ Paginación automática.
- ✓ Creación de elementos de dominio aplicando sus validaciones definidas.
- ✓ Verificación de consistencia referencial entre elementos del dominio, es decir, no permite eliminar elementos relacionados/enlazados desde otro elemento de dominio.
- ✓ Personalización, se pueden internacionalizar los mensajes, se pueden modificar las plantillas y aplicar contenidos diferentes.



Figura 6. Tareas creadas usando CRUD

Enviando a producción

En el caso que quisiéramos publicar nuestra aplicación Tareas(lo cual no recomiendo, al menos no en su estado actual ;-)) debemos realizar las siguientes operaciones:

1. En **conf/application.conf**
2. Cambiar el ambiente en Play cambiar **application.mode=dev** por **application.mode=prod**
3. Usar una base de datos en memoria por una real (Postgres por ejemplo), cambiar **db=mem** por **db=postgres://postgres:secret@superhost/tareas**
4. Agregar la librería de conexión de base de datos JDBC para Postgres (en base al punto anterior) en la carpeta **lib**.
5. Generar el archivo WAR para nuestra aplicación con **play war -o /carpeta_externa/Tareas.war**
6. Desplegar el archivo WAR en Tomcat, Glassfish o Jboss.

Consideraciones importantes

Viendo el código usado para nuestra aplicación pueden darse ciertas interrogantes muy importantes si es que usted tiene experiencia en el entorno Java:

- ✓ Porqué usar variables públicas? Donde están los getters y setters?
- ✓ Son generados en tiempo de

compilación por Play.

- ✓ Porqué usan métodos estáticos en los controladores? Esto no rompe el principio de ejecución para peticiones concurrentes?
- ✓ Play solamente usa métodos estáticos donde sean aplicables:
- ✓ Controladores, los cuales no mantienen estado y definen su comportamiento en base a las rutas configuradas y los parámetros que reciben.
- ✓ Modelo, en los métodos factory (fábrica) que no afectan ni rompen principios de modificación concurrente.

Quiere usar un IDE?

A pesar de que es suficiente con usar un editor de código, puede usted verse en la necesidad de aplicar las bondades de los entornos integrados de desarrollo (IDE de sus siglas en inglés) con Play, para este cometido Play ofrece las siguientes utilidades a través de su consola de ejecución:

- ✓ **play ant**: Genera un archivo de construcción Ant para la aplicación.
- ✓ **play eclipsify**: Genera los archivos de configuración para Eclipse.
- ✓ **play netbeansify**: Genera los archivos de configuración para Netbeans.
- ✓ **play idealize**: Genera los archivos de configuración para IntelliJ.

Qué fué lo que hicimos?

Recopilemos lo que hicimos para nuestra aplicación Tareas:

1. Creamos la aplicación base.
2. Agregamos el elemento de dominio Tarea usando el componente base Model.
3. Habilitamos una base de datos en memoria a través de la configuración **db=mem**
4. Aplicamos el módulo CRUD a la aplicación definiendo un componente controlador base denominado Tareas y configuramos su ruta de mapeo HTTP en **/conf/routes**
5. Verificamos la creación, modificación y borrado de tareas usando las interfaces generadas a través del módulo CRUD.

6. Empaquetamos nuestra aplicación para un ambiente de producción.

Conclusiones

De forma sencilla y sin mucho esfuerzo definimos una aplicación sencilla pero lo suficientemente completa para mostrar las características fundamentales de Play, su simplicidad y convenciones de trabajo.

Comparado a un ambiente JEE se tiene un ahorro de líneas de código efectivas, de forma que esta aplicación no solo es más corta, sino que es más legible y por tanto mantenible debido al uso de principios de desarrollo desde el inicio. Esto hace de Play una opción atractiva para desarrolladores del mundo Java que prefieran usar un entorno integrado más ligero y simplificado que los actuales.

Referencias

- [1] <http://www.playframework.org/documentation/1.2.5/home>
- [2] <http://www.playframework.org/documentation/1.2.5/guide8>
- [3] <http://stackoverflow.com/questions/5192904/play-framework-uses-a-lot-of-statics>

Autor



Timoteo Ponce

Ingeniero de Software – Consultor Técnico

timo@gmail.com



Trabajando con Binary Large Object (BLOB) en PostgreSQL con MonoDevelop

Los BLOB (Binary Large Objects, objetos binarios grandes) son elementos utilizados en las bases de datos para almacenar datos de gran tamaño que cambian de forma dinámica. Generalmente, estos datos son imágenes, archivos de sonido y otros objetos multimedia; a veces se almacenan como BLOB código de binarios.



Introducción

En la actualidad existen aplicaciones que además de almacenar y procesar datos basados en caracteres, requieren también almacenar y procesar archivos de gran tamaño de tipo binario o texto, archivos como los multimedia (gráficas, imágenes, audio, vídeo) o los generados por programas de oficina (documentos, presentaciones, hojas de calculo), para estos casos en el estándar SQL están definidos dos tipos de datos para guardar grandes cantidades de información: el CLOB (CHARACTER LARGE OBJECT) que se utiliza para información de tipo texto y el BLOB (BINARY LARGE OBJECT) que se utiliza para información de tipo binario, para este último PostgreSQL proporciona dos formas de trabajar: una es mediante las funciones `lo_import()` y `lo_export()` incorporadas dentro del servidor de base de datos y la otra mediante columnas del tipo de dato `bytea`, que pueden ser utilizadas por las

aplicaciones para el almacenamiento de archivos binarios.

En este documento se muestra el uso de las clases de .NET para la escritura y la lectura de datos binarios mediante un proyecto de consola con MonoDevelop.

Como leer y escribir objetos binarios en PostgreSQL con MonoDevelop

Para leer y escribir tipos de datos binarios de PostgreSQL con .NET, el data provider de PostgreSQL

(<http://npgsql.projects.pgfoundry.org/>) proporciona la clase `NpgsqlDataReader`, esta clase representa un cursor rápido de solo lectura (read-only) para un recordset (conjunto de registros) obtenido de la base de datos mediante una consulta SQL, esto lo hace ideal para grandes cantidades de datos y para registros binarios de gran tamaño ya que obtiene la información mediante un flujo (stream) de bytes un registro a la vez de todo el conjunto de datos, esta característica puede combinarse con un objeto POCO (plain old CLR object) para poder crear objetos que sean equivalentes a las entidades en la base de datos.

Para demostrar su utilización creamos una base de datos en PostgreSQL con una tabla llamada Books.

```
CREATE TABLE Books
( bookid serial primary key NOT NULL,
  title varchar(512) NOT NULL,
```

```
publishyear smallint null,
  picture bytea );
```

A continuación ejecutamos Monodevelop y creamos un proyecto de consola, utilizando el lenguaje de programación C#.

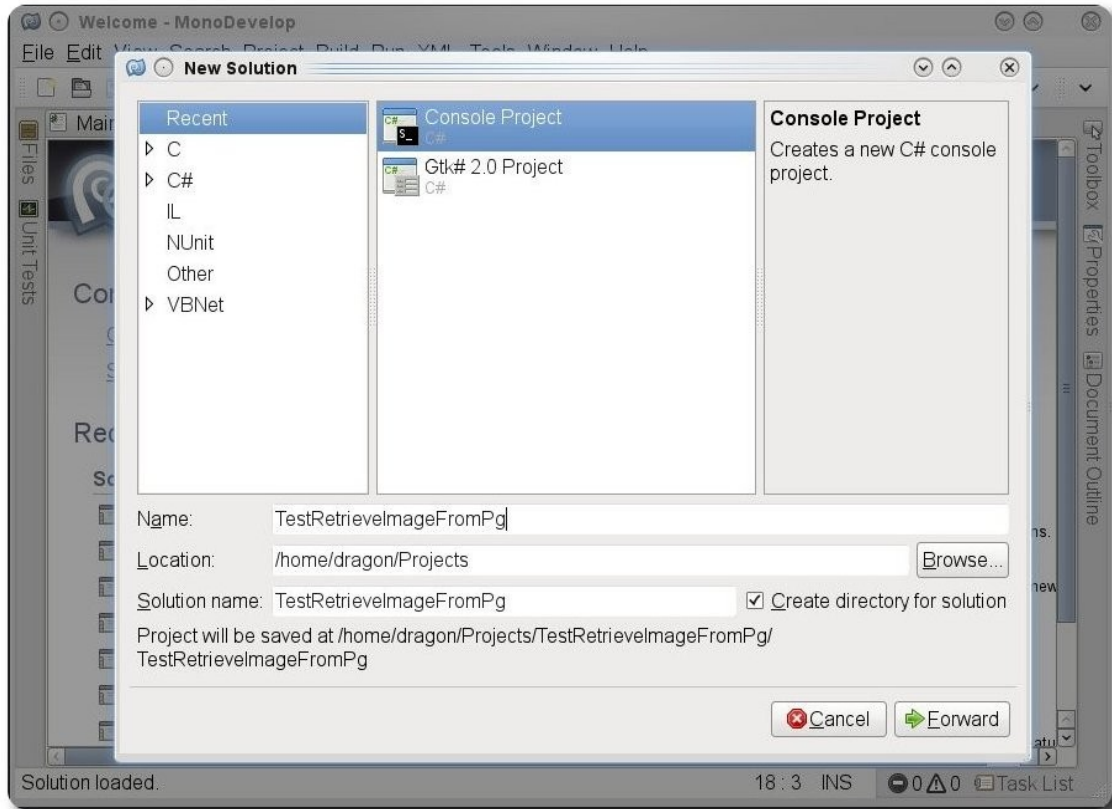


Figura 1.

Este proyecto se compone de tres clases: Main, Book y BooksManagerDAC. Main es la clase principal, Book es el objeto que representa un registro en la base de datos y BooksManagerDAC es donde se encuentra toda la funcionalidad para la comunicación con la base de datos.

El código de la clase Book.

```
using System;
namespace TestRetriveImageFromPg
{
    public class Book{
        public int BookId{set;get;}
        public string Title{ set; get;}
        public short PublisherYear {set;get;}
        public string ImagePath {set;get;}
        public byte[] ImageBytes {set;get;}
    }
}
```

El código de la clase BooksManagerDAC donde se concentra toda la funcionalidad de comunicación con la base de datos.

```

using System;
using System.IO;
using Npgsql;
using NpgsqlTypes;
using System.Collections.Generic;
using System.Data;

namespace TestRetriveImageFromPg
{
    public class BooksManagerDAC
    {
        static string strConnString =
"Server=127.0.0.1;Port=5432;Database=BookSample;User ID=postgres;Password=Pa$$W0rd";
        public static void Create (Book b)
        {
            string commandText = "Insert into
books(title,publishyear,picture)Values(:title,:publishyear,:picture)";
            byte[] bytesFromImage = GetPhoto(b.ImagePath);
            using(NpgsqlConnection conn = GetConnection())
            {
                using(NpgsqlCommand cmd = new
NpgsqlCommand(commandText,conn))
                {
                    var paramTitle = new NpgsqlParameter("title",
NpgsqlDbType.Varchar);
                    paramTitle.SourceColumn = "title";
                    paramTitle.Value = b.Title;
                    cmd.Parameters.Add(paramTitle);

                    var paramPubYear = new NpgsqlParameter("publishyear",
NpgsqlDbType.Smallint);
                    paramPubYear.SourceColumn = "publishyear";
                    paramPubYear.Value = b.PublisherYear;
                    cmd.Parameters.Add(paramPubYear);

                    var pPicture = new NpgsqlParameter("picture", NpgsqlDbType.Bytea);
                    pPicture.SourceColumn = "picture";
                    pPicture.Value = bytesFromImage;
                    cmd.Parameters.Add(pPicture);
                    int r = cmd.ExecuteNonQuery();
                    Console.WriteLine("{0} affected",r);
                }
            }

            public static Book SelectById(int id,string fileName){
                string commandText = "SELECT bookid,title,publishyear,picture "
+
                " FROM Books WHERE bookid = " + id.ToString();
                Book b = null;
                using(NpgsqlDataReader reader = GetReader(commandText))
                {
                    int colBookId = reader.GetOrdinal("bookid");
                    int colTitle = reader.GetOrdinal("title");
                    int colPublishYear = reader.GetOrdinal("publishyear");
                    int colPicture = reader.GetOrdinal("picture");

                    while(reader.Read()){
                        b = new Book{
                            BookId = reader.GetInt32(colBookId),
                            Title = reader.GetString(colTitle)
                        };
                    }
                }
            }
        }
    }
}

```



```

        if(!reader.IsDBNull(colPublishYear))
            b.PublisherYear =
reader.GetInt16(colPublishYear);

        if(!reader.IsDBNull(colPicture))
        {
            Console.WriteLine("Retrieving
image...");
            RetrieveImage(reader,colPicture,fileName);
        }
    }
    return b;
}

static void RetrieveImage(NpgsqlDataReader reader,int
columnImage,string fileName){
    byte[] result = (byte[])reader.GetValue(columnImage);
    using(FileStream fis = new
FileStream(fileName, FileMode.OpenOrCreate, FileAccess.Write))
    {
        using(BinaryWriter writer = new BinaryWriter(fis))
        {
            writer.Write(result);
            writer.Flush();
        }
    }
}

static byte[] GetPhoto(string filename){
    byte[] photo = null;
    using(FileStream fis = new
FileStream(filename, FileMode.Open, FileAccess.Read))
    {
        BinaryReader reader = new BinaryReader(fis);
        photo = reader.ReadBytes((int) fis.Length);
        reader.Close();
    }
    return photo;
}

public static List<Book> SelectAll(){
    var resp = new List<Book>();
    Book b = null;
    string commandText = "SELECT bookid,title FROM Books ";
    using(NpgsqlDataReader reader = GetReader(commandText))
    {
        while(reader.Read()){
            b = new Book();
            b.BookId = Convert.ToInt32(reader["bookid"]);
            b.Title = reader["title"].ToString();
            resp.Add(b);
        }
    }
    return resp;
}

static NpgsqlConnection GetConnection(){
    NpgsqlConnection conn = new NpgsqlConnection(strConnString);
    conn.Open();
    return conn;
}

```

```

        static NpgsqlDataReader GetReader(string commandText){
            NpgsqlDataReader resp = null;
            NpgsqlConnection conn = GetConnection();
            using (NpgsqlCommand cmd = new
NpgsqlCommand(commandText, conn))
            {
                resp =
cmd.ExecuteReader(CommandBehavior.CloseConnection | CommandBehavior.SequentialAcce
ss);
            }
            return resp;
        }
    }
}

```

Y finalmente la clase Main

```

using System;

namespace TestRetriveImageFromPg
{
    class MainClass
    {
        public static void Main (string[] args)
        {
            try{
                Add();
                Console.WriteLine("Press any key to query the database.");
                Console.ReadLine();
                Query();
            }catch(Exception ex){
                Console.WriteLine(ex.Message);
            }
            Console.ReadLine();
        }

        static void Query(){
            Console.WriteLine(Environment.NewLine);
            Console.WriteLine(" Querying database .....");
            BooksManagerDAC.SelectAll().ForEach(b =>
Console.WriteLine("{0}\t{1}",
b.BookId,b.Title));

            Console.WriteLine(Environment.NewLine);
            Console.WriteLine("Retrieve database record.....");
            Console.Write("Id ? ");
            int id = Convert.ToInt32(Console.ReadLine());
            Console.Write("Filename? ");
            string fileName = Console.ReadLine();
            Book myBook = BooksManagerDAC.SelectById(id,fileName);
            Console.WriteLine(myBook.Title);
            Console.WriteLine(myBook.PublisherYear);
            Console.WriteLine("Done.");
        }
    }
}

```

```

static void Add(){
    Book b = new Book();
    Console.WriteLine("Title ? ");
    b.Title = Console.ReadLine();
    Console.WriteLine("Publish year ? ");
    b.PublisherYear = Convert.ToInt16(Console.ReadLine());
    Console.WriteLine("Path to image ? ");
    b.ImagePath = Console.ReadLine();
    BooksManagerDAC.Create(b);
    Console.WriteLine("Done.");
}
}

```

Al ejecutar la clase dentro de Monodevelop o bien desde una terminal, la aplicación realiza las siguientes acciones: solicita los datos de un libro para almacenar, muestra los registros que ya están almacenados y solicita una llave primaria o id para obtener el registro.

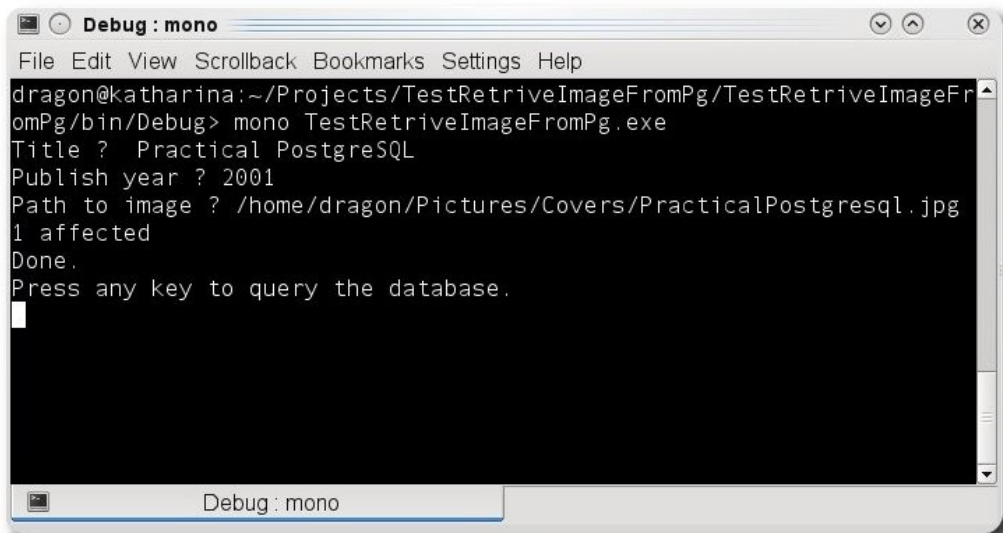


Figura 2.

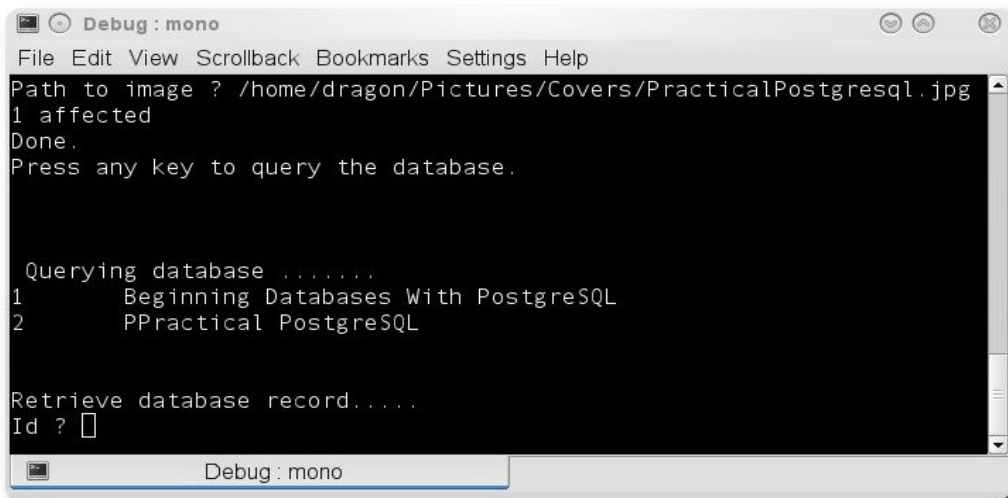


Figura 3.

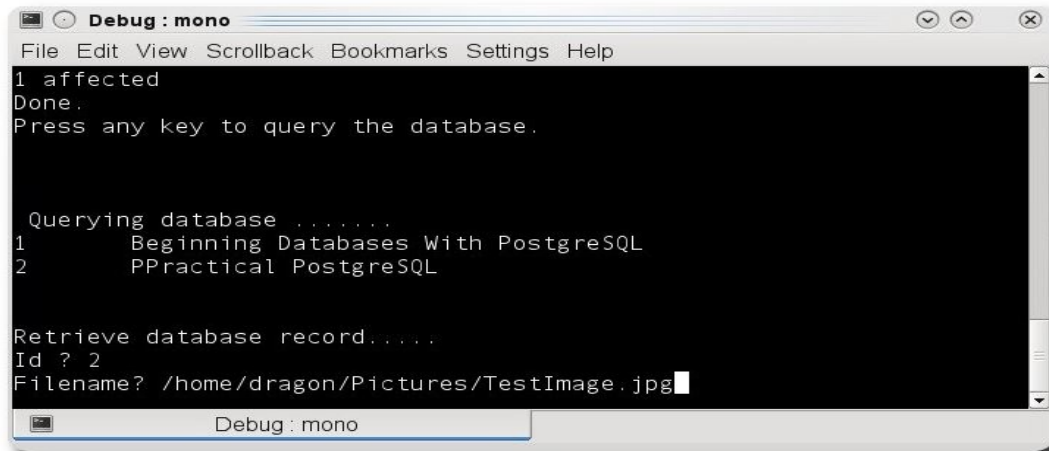


Figura 4.

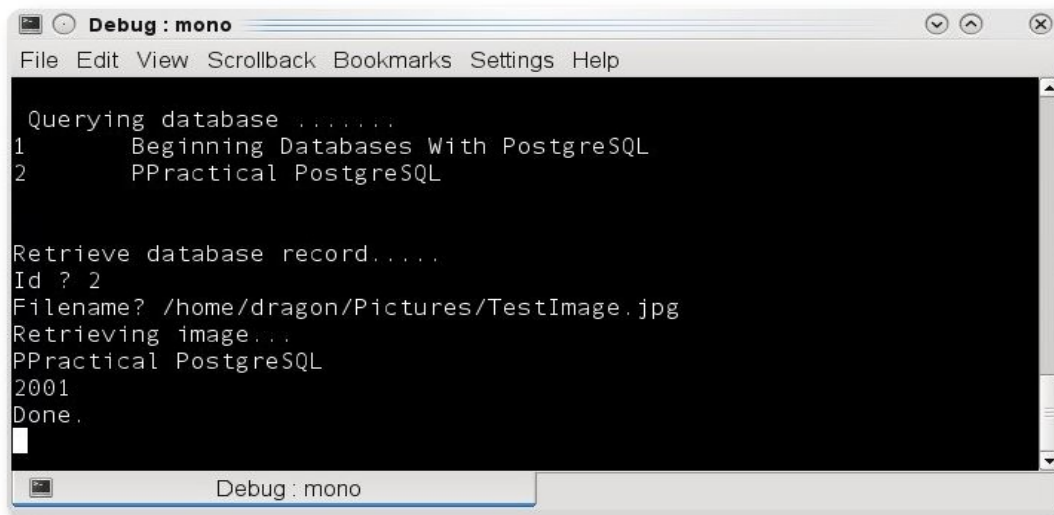


Figura 5.

Ahora la explicación de las funcionalidades que se encuentran en la clase **BooksManagerDAC**.

Mediante el método **GetPhoto()** la aplicación toma un archivo del sistema de archivos y lo convierte a una matriz de bytes

```
static byte[] GetPhoto(string filename){
    byte[] photo = null;
    using(FileStream fis = new
FileStream(filename, FileMode.Open, FileAccess.Read))
    {
        BinaryReader reader = new BinaryReader(fis);
        photo = reader.ReadBytes((int) fis.Length);
        reader.Close();
    }
    return photo;
}
```

Esta matriz de bytes se pasa como parámetro al comando **Insert** dentro del método **Create** (Book b) para enviarla hacia la base de datos.


```
byte[] bytesFromImage = GetPhoto(b.ImagePath);
var pPicture = new NpgsqlParameter("picture", NpgsqlDbType.Bytea);
pPicture.SourceColumn = "picture";
pPicture.Value = bytesFromImage;
```

Al obtener el registro mediante el método **SelectById(int id,string fileName)** se utiliza el método **RetrieveImage** únicamente para la columna **picture** en donde se convierte la matriz de bytes de la tabla a un archivo en el sistema de archivos.

```
static void RetrieveImage(NpgsqlDataReader reader,int columnImage,string fileName){
    byte[] result = (byte[])reader.GetValue(columnImage);
    using(FileStream fis = new
FileStream(fileName,FileMode.OpenOrCreate,FileAccess.Write))
    {
        using(BinaryWriter writer = new BinaryWriter(fis))
        {
            writer.Write(result);
            writer.Flush();
        }
    }
}
```

Si consultamos el contenido de la tabla, mediante PgAdmin veremos como la columna **picture** se muestra como un tipo de dato binario.

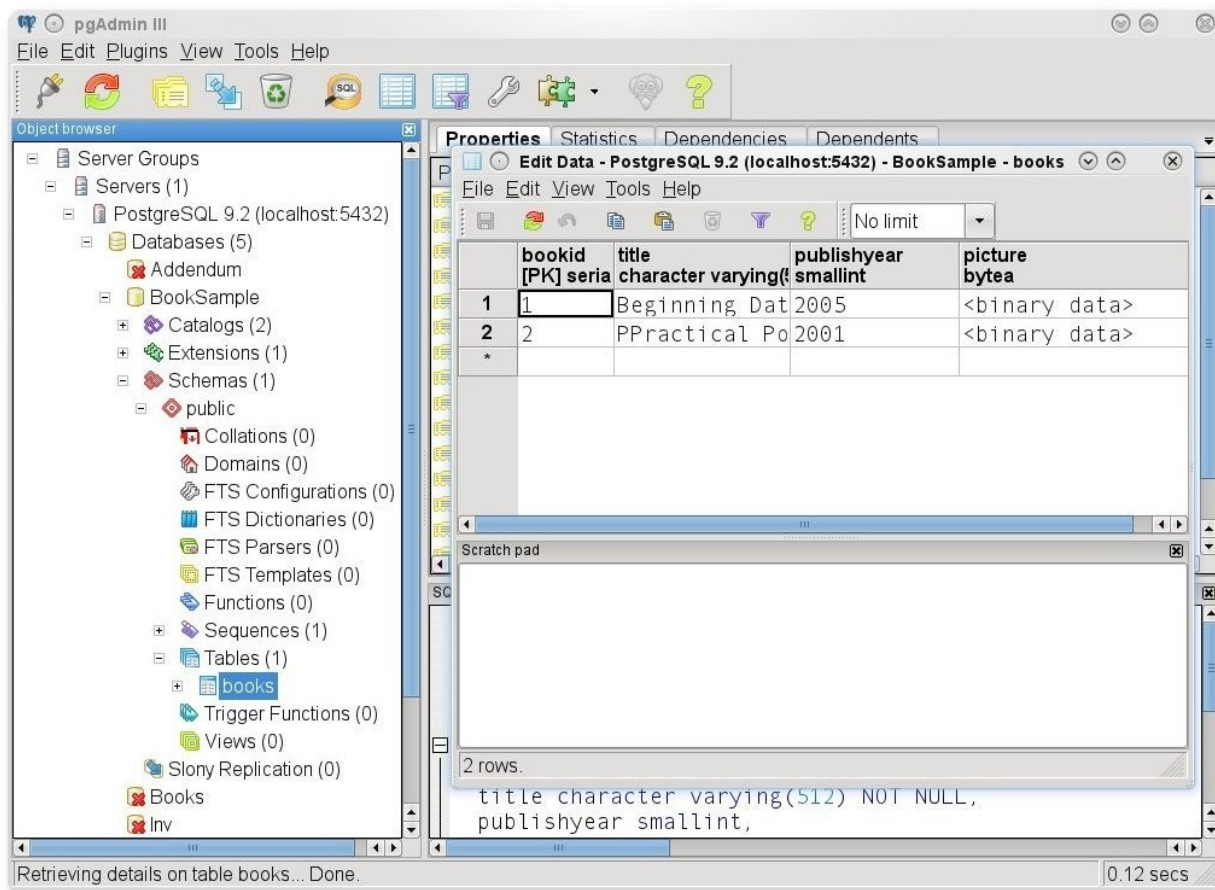


Figura 6.

Conclusiones

- ✓ PostgreSQL es una de las más completas base de datos relacionales de fuente abierta (open source) , el soporte al almacenamiento de archivos binarios hace que sea considerada como una buena alternativa de almacenamiento que combinado con las capacidades de .NET tenemos una propuesta sólida para la construcción de aplicaciones empresariales.
- ✓ Los ejemplos pueden ser descargados de <http://xomalli.blogspot.mx/>
- ✓ Este documento está protegido bajo la licencia de documentación libre Free Documentation License del Proyecto GNU, para consulta ver el sitio <http://www.gnu.org/licenses/fdl.txt> , toda persona que lo desee está autorizada a usar, copiar y modificar este documento según los puntos establecidos en la «Licencia FDL»

Referencias

- [1] Matthew Neil, Stones Richard; Beginning Databases with PostgreSQL: From Novice to Professional; 2005; editorial Apress.
- [2] Douglas Korry, Douglas Susan; PostgreSQL the comprehensive guide to building, programming and administering PostgreSQL; 2005; Sams Publishing
- [3] Momjian Bruce; PostgreSQL Introduction and Concepts; 2001; Addison-Wesley

Autor



I.S. Martín Márquez

xomalli@gmail.com



Luz, Cámara y Orientémoslo al Móvil

Un panorama del desarrollo de aplicaciones web para dispositivos móviles

Incluso en las interfaces de usuario modo texto podemos elaborar aplicaciones elegantes y llamativas.



Introducción

En la actualidad se puede apreciar un crecimiento constante de personas que acceden a la web haciendo uso de dispositivos móviles, lo que pone en evidencia que existe la necesidad de interactuar con la web sin importar el lugar y menos aún los medios.

Según reportes de la compañía Gartner Inc., líder mundial en investigación de tecnología de información y consultoría, se estima que en el 2013 los teléfonos móviles superarán a las PCs como los dispositivos más utilizado para el acceso a la web y que para el 2015 más del 80% de los teléfonos serán Smartphones (Gartner, 2012).

El escenario vigente pone de manifiesto que durante el proceso de diseño de sitios web, se debe considerar aspectos necesarios para lograr un despliegue adecuado en los dispositivos móviles, lo cual ha originado un debate sobre cuál debería ser el tipo de solución software a ser desarrollado para este tipo de aparatos, existiendo para su

elección: aplicaciones web optimizadas para móviles, aplicaciones móviles nativas y aplicaciones híbridas.

La elección de alguna de estas soluciones se encontrará en función al objetivo de la aplicación, así, se hace recomendable la elaboración de una aplicación nativa cuando es preciso identificar diferente contenido, diseño e interacción basada en dispositivo, localización, velocidad de conexión, así como, las capacidades técnicas del dispositivo; en otro caso tal vez sería suficiente disponer una versión web del sitio. Para ver las ventajas y desventajas de estas alternativas se puede consultar el documento emitido por KendoUI: Mobile Developer Guidance (UI, 2012).

En cualquier caso, el desarrollo móvil involucra muchos aspectos, desde las necesidades de los clientes hasta planificadores UX (User eXperience) además de tomar en cuenta las recomendaciones de accesibilidad para brindar la posibilidad de interacción a personas con alguna discapacidad. Este escenario en ocasiones ha convergido en el desarrollo de diferentes versiones de un mismo sitio web lo que implica el consumo adicional de distintos recursos.

En el caso que se adopte una solución orientada a la web, se plantea como solución la elaboración de un diseño web sensible (responsive web design) en base a HTML5 y CSS3, que permita a un sitio web desplegarse de manera adecuada ante diversos dispositivos y tamaño de pantallas,

siendo la mejor parte de ésto que las técnicas están todas implementadas sin la necesidad de soluciones que estén basadas en el lado del servidor.

Algunos conceptos a considerar

- ✓ **Screen size:** Tamaño físico de despliegue que tiene un dispositivo.
- ✓ **Viewport:** Es el área donde un sitio web se despliega sin considerar los componentes del navegador (excluyendo menus, toolbars, tabs, etc.)
- ✓ **Fixed web design:** Un sitio web con diseño fijo tiene dimensiones definidas, para el cual, el cambio de tamaño del navegador o viéndolos en diferentes dispositivos no afectan la manera en que el sitio web se despliega. Su navegación requiere el uso de la barra de scroll horizontal y el sitio no puede desplegarse adecuadamente en Smartphones o tabletas.
- ✓ **Fluid web design:** Un sitio web con diseño fluido considera el uso de porcentajes para el ancho del sitio, como resultado las columnas son relativas a las dimensiones del navegador permitiendo que el contenido se escale de manera fluida.
- ✓ **Adaptive web design:** Un sitio web con diseño adaptable considera el uso de anchos fijos o relativos y “media queries” para tomar en cuenta el tamaño específico de los dispositivos en los que se desplegará, ya sean pequeños monitores, tabletas o Smartphones. Están basados por lo general en conjunto de reglas definidas sobre las capacidades de los dispositivos, así como el tamaño de las pantallas, como resultado, el cambio del diseño puede ser radical con muchas variaciones. Si se identifica a los dispositivos, se puede “adaptar” una página web para incluir diversas versiones de imágenes o

remover video cuando se trata de pantallas pequeñas. Los diseños adaptables más sofisticados usan scripts para identificar al dispositivo que visita y de acuerdo a ello despliega la mejor versión del sitio.

Como ejemplo se puede dar a conocer el sitio web www.isobar.com, en el cual existe tres diferentes diseños (layouts) dirigidos a tres diferentes y específicos tamaños de pantalla, estos puntos (breakpoints) no son arbitrarios, sino que corresponden a tamaños de dispositivos populares.

- ✓ **Responsive web design:** Un sitio web con diseño sensible está construido en base a anchos relativos, imágenes flexibles y hace uso de “media queries” para controlar el diseño y su contenido.

Un ejemplo de este tipo de diseño es el sitio web www.bostonglobe.com, en el cual se puede apreciar que el layout se acomoda en el navegador a medida que este va cambiando de tamaño, de tal modo que muestra la disposición del contenido de acuerdo al espacio en el que se encuentra visualizado.

HTML5

Esta versión de HTML presenta una evolución considerable, siendo las mejoras más destacables:

- ✓ **Simplificación:** Ofrece formas más sencillas en la especificación de parámetros y piezas de código.
- ✓ **Contenido multimedia:** Reproducción de audio y video sin necesidad de plug-ins.
- ✓ **Animaciones:** Tiene la capacidad de mostrar contenidos de manera similar a Adobe Flash, sin embargo ya no hace uso de este complemento.
- ✓ **Almacenamiento de datos del lado del cliente:** Permite almacenar y procesar información en el cliente,

logrando que una aplicación web tenga el mismo comportamiento que una de escritorio.

- ✓ **Efectos y nueva versión de hojas de estilo CSS:** CSS3 engloba las nuevas posibilidades de formato, como por ejemplos: la implementación de sombras, bordes redondeados, etc., ventajas que se traducen en una mejora de la velocidad y rendimiento de un sitio, así como en nuevas e ilimitadas opciones de diseño.
- ✓ **Geo-locación:** Gracias a esta característica, los sitios web podrán saber la ubicación física de los usuarios que lo visitan.
- ✓ **Tipografías no estándar:** Se tiene la posibilidad de utilizar tipografías no estándar en sitios web.

Las especificaciones sobre HTML5, pueden ser consultadas en la siguiente dirección: <http://dev.w3.org/html5/spec/Overview.html>

Herramientas para simplificar el desarrollo en HTML5 y CSS3

- ✓ **HTML5 Boilerplate**
<http://html5boilerplate.com/>
Provee una plantilla base de HTML, CSS y JS con las normas vigentes de HTML5, CSS3, normalización de CSS y JavaScript.
- ✓ **Modernizr**
<http://www.modernizr.com/>
Librería de JavaScript que ayuda a implementar las nuevas ventajas del HTML5 y CSS3.
- ✓ **Initializr**
<http://www.initializr.com/>
Initializr es un sitio que genera y permite personalizar plantillas para HTML5.
- ✓ **52framework**
<http://52framework.com/>

Framework que ayuda en el uso de HTML5, CSS3 y Javascript.

- ✓ **HTML5 Reset**
<http://html5reset.org/>
Framework para HTML5 que acumula las mejores prácticas y técnicas para utilizar esta tecnología.
- ✓ **SwitchToHTML5**
<http://switchtohtml5.com/>
Sitio que proporciona opciones necesarias para generar plantillas en base a HTML5.
- ✓ **CSS Preprocessors**
Son lenguajes desarrollados con el propósito de adicionar características interesantes e ingeniosas para las hojas de estilo, sin que exista incompatibilidad en los diversos navegadores. Entre los más populares se encuentran:
 - ✓ **SASS - Syntactically Awesome Stylesheets**
sass-lang.com
 - ✓ **LESS - The Dynamic Stylesheet language**
lesscss.org
 - ✓ **STYLUS - Expressive, dynamic, robust CSS**
learnboost.github.com/stylus/

Estrategias de desarrollo orientado al móvil

- ✓ **Graceful degradation (degradación elegante):** Significa crear un sitio web orientado a navegadores modernos para posteriormente asegurarse de que navegadores antiguos o aquellos que tengan ciertas limitaciones puedan tener una experiencia aceptable, aplicándose para ello "trucos" o simplemente eliminando las partes que no funcionan.
- ✓ **Progressive enhancement (mejora progresiva):** Va en sentido contrario a la estrategia anterior, ya que tiene su

inicio en el uso de estándares web mediante lo cual asegura que un sitio debería ser usado por todos los navegadores (con independencia de tecnologías como Javascript o incluso CSS), a partir de ello se mejora la experiencia en los demás navegadores por medio de CSS y Javascript.

Este concepto trata de hacer inclusivo a un sitio web, es decir obtener la mejor experiencia para la más amplia audiencia posible, sin importar si los usuarios están viendo el sitio en un iPhone, una computadora de escritorio, un “ebook reader” como es Kindle, o escuchándolo mediante un lector de pantallas, la experiencia debería ser tan completa y funcional como fuese posible.

Esta estrategia se constituye en la base para lo que se denomina como la técnica “Mobile First”, que recomienda las bases para la elaboración de sitios web orientados a dispositivos móviles.

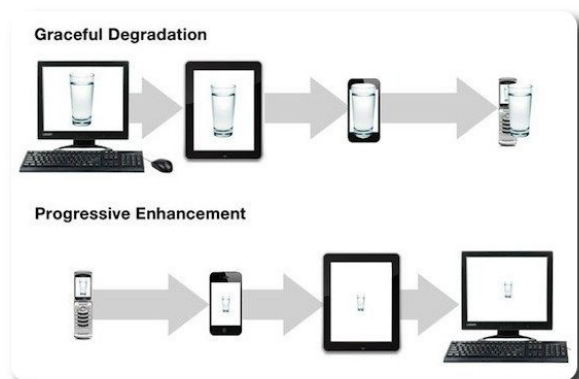


Figura 1.

Responsive HTML5 design

(Frain, 2012) Una definición alterna al diseño sensible se refiere a la presentación del contenido en la manera más accesible por cualquier viewport por el que se accede, esto responde a los cambios en el tamaño del navegador por medio de un “ajuste fluido” del ancho para acomodarse de manera cabal al espacio disponible. De esta forma mientras se cambia el tamaño de la ventana del

navegador, ya sea agrandándolo o reduciéndolo, se puede apreciar que el diseño del sitio cambia en tiempo real.

El concepto de “responsive HTML5 design” involucra al proceso de diseño y desarrollo de sitios web que son aptos para responder ante las acciones de los usuarios y detectar el medio donde el sitio esta siendo desplegado actualmente, esto para proporcionar la mejor experiencia posible para los usuarios, en lo que se refiere a la facilidad de navegación y lectura del contenido.

El diseño sensible en HTML5 no solo se refiere a la reducción del tamaño de las letras (fuentes) ni al ajuste de las imágenes para hacer que se muestren de manera cabal en un nuevo formato, este proceso requiere un proceso reflexivo donde, tanto los diseñadores y desarrolladores trabajan juntos para determinar como redistribuir los elementos de acuerdo a la resolución, que elementos pueden ser eliminados y como mantener el concepto mientras se simplifica la estructura.

En términos de diseño del lado del usuario, el diseño web sensible tiene tres componentes:

1. Flexible grid-based layout
2. Flexible images & media
3. Media queries

1. Flexible, grid based layout

Significa que el diseño de una página web debería estar definido en valores relativos (porcentajes), permitiendo al sitio web ser sensible ante diferentes dimensiones de dispositivos. Una fórmula simple y consistente para convertir dimensiones fijas (píxeles) a porcentajes es:

$$\text{Resultado} = \text{objetivo} \div \text{contexto}$$

Por ejemplo, se tiene la siguiente estructura asociada a un CSS expresadas en unidades fijas:

```
<div id="wrapper">
  <header>
  </header>
</div>

#wrapper {
  width: 960px;
}
header {
  width: 940px;
}
```

Si se desea transformar a dimensiones de porcentajes, se debe proceder de la siguiente manera:

- ✓ Para el contenedor (wrapper) buscar un valor en porcentaje lo más próximo a la dimensión fija, en este caso supongamos 96%.
- ✓ Para el encabezado (header) y todas las secciones contenidas dentro de wrapper, se debe utilizar la fórmula mencionada anteriormente, obteniéndose:

```
Resultado = Ancho de header ÷ ancho de wrapper
= 940 ÷ 960
= 97.9166667
```

Los estilos quedarían de la siguiente manera:

```
#wrapper {
  width: 96%; /*Porcentaje más aproximado*/
}
header {
  width: 97.9166667%; /* 940 ÷ 960 */
}
```

Para el tamaño de las fuentes se debe realizar la misma operación, considerando que todos los navegadores modernos adoptan 16px como el tamaño de fuente por defecto, además de la recomendación de utilizar como unidad de medida "em" (se refiere a la letra "M" y se toma como parámetro para las fuentes ya que es la más ancha de todas las letras), se procede de la siguiente manera:

```
Contexto = font-size: 100%;
= font-size: 16px;
= font-size: 1em;
```

Por ejemplo se desea transformar el siguiente tamaño de fuente:

```
font-size: 48px;
```

El resultado sería:

```
font-size: 3em; /* 48 ÷ 16 = 3*/
```

2. Flexible images & media

Hace referencia a que tanto las imágenes como aquellos elemento multimedia se puedan acomodar a los diversos contenidos, por lo que se ha identificado la siguiente propiedad CSS mediante la cual se previene que en algún momento puedan ser más grandes que sus contenedores:

```
img, embed, object, video {
  max-width: 100%;
}
```

3. Media queries

Permiten determinar si ciertas expresiones son ciertas en relación al navegador, si lo son, se puede cargar un bloque específico de estilos. La estructura de una media query tiene cuatro componentes básicos (Kadlec, 2013):

```
@media [not|only] type [and] (expr) {
  rules
}
```

Media types

Especifica el tipo de dispositivo, pudiendo ser:

- ✓ **all**: Todos los medios definidos.
- ✓ **braille**: Dispositivos táctiles que emplean el sistema Braille.
- ✓ **embossed**: Impresoras Braille
- ✓ **handheld**: Dispositivos de mano como móviles y PDA
- ✓ **print**: Impresoras y navegadores en el modo "Vista Previa para Imprimir".

- ✓ **projection:** Proyector y dispositivos para presentaciones.
- ✓ **screen:** Pantallas de ordenador.
- ✓ **speech:** Sintetizadores para navegadores de voz utilizados por personas discapacitadas.
- ✓ **tty:** Dispositivos textuales limitados como teletipos y terminales de texto.
- ✓ **tv:** Televisores y dispositivos con resolución baja.

Media expressions

Permite evaluar las características de un dispositivo. Ejemplo: determinar si el ancho de el viewport es más grande que 320px:

```
@media screen and (min-width: 320px) {
}
```

Las características de los dispositivos que se pueden evaluar son:

- ✓ **width:** describe el ancho del área de visualización del dispositivo
- ✓ **height:** describe la altura de la zona de visualización del dispositivo
- ✓ **device-width:** describe el ancho de la superficie de renderización del dispositivo.
- ✓ **device-height:** describe la altura de la superficie de renderización del dispositivo.
- ✓ **orientation:** indica la orientación del dispositivo, si se encuentra en disposición vertical (portrait) u horizontal (landscape).
- ✓ **aspect-ratio:** relación entre el valor del ancho con respecto al valor de la altura.
- ✓ **device-aspect-ratio:** relación entre el valor device-width y el valor del device-height.
- ✓ **color:** Número de bits por componentes de color en el dispositivo (retorna cero si el dispositivo no soporta colores).

- ✓
- ✓ **color-index:** Número de entradas en la tabla de consulta de colores para el dispositivo.
- ✓ **monochrome:** Número de bits por pixel en un dispositivo monocromático (devuelve cero si el dispositivo soporta colores).
- ✓ **resolution:** Resolución (densidad de pixeles) del dispositivo (resolución puede ser expresada en puntos por pulgada [dpi] o puntos por centímetro [dpcm]).
- ✓ **scan:** Proceso de búsqueda de dispositivos "TV".
- ✓ **grid:** Indica si el dispositivo está basado en grid (1) o en mapa de bits (0).

Logical keywords

Son palabras claves que permiten crear expresiones más complejas, así se puede mencionar a las siguientes:

- ✓ AND
- ✓ NOT
- ✓ OR
- ✓ ONLY

Rules

Estilos básicos que ajustan el modo de visualización. Son las reglas básicas de CSS. Lo que los hace especiales es que estos se encuentran dentro de una media query:

```
@media only screen and (min-width: 320px)
{
  a{
    color: blue;
  }
}
```

Herramientas para Test

Recursos en línea: En la web existen diversas herramientas que ayudan a realizar

pruebas para ver la sensibilidad de los sitios web en relación a múltiples tamaños de pantallas, a continuación se da a conocer algunos recursos disponibles:

- ✓ The Responsinator responsinator.com
- ✓ Responsive.is responsive.is
- ✓ Responsivepx responsivepx.com
- ✓ QuirkTools quirktools.com/screenfly/
- ✓ Pixel Perfect Responsive Design Testing Tool screenqueri.es
- ✓ Screenfly quirktools.com/screenfly/
- ✓ Demonstrating Responsive Design www.jamus.co.uk/demos/rwd-demonstrations/
- ✓ Media Query Bookmarklet seesparkbox.com/foundry/media_query_bookmarklet/
- ✓ Matt Kersley's Responsive Tool mattkersley.com/responsive/

Extensiones para navegadores

- ✓ Windows Resizer - Chrome <https://chrome.google.com/webstore/detail/kkelicaakdanhinjdeam>

mmilcgefonfh

- ✓ Firesizer - Firefox <https://addons.mozilla.org/en-US/firefox/addon/firesizer/>
- ✓ Developer Toolbar - Microsoft Internet Explorer <http://www.microsoft.com/download/en/details.aspx?id=18359>

Sitios web de inspiración

A continuación algunos ejemplos disponibles en la web donde podemos ver cómo funciona el diseño sensible:

- ✓ mediaqueri.es
- ✓ www.warface.co.uk/
- ✓ reverseburo.com/
- ✓ atlason.com/
- ✓ www.hangingupthemoon.com/

Conclusiones

- ✓ El diseño sensible proporciona una mejor y más inclusiva experiencia del usuario en relación que un diseño fijo.
- ✓ En la web se tienen gran cantidad de recursos que posibilitan realizar la adopción de los sitios web sin el excesivo consumo de recursos.

Recursos consultados

- [1] Frain, B. (2012). Responsive Web Design with HTML5 and CSS3. Packt Publishing.
- [2] Gartner. (23 de 10 de 2012). Gartner Identifies the Top 10 Strategic Technology Trends for 2013. Obtenido de <http://www.gartner.com/it/page.jsp?id=2209615>
- [3] Hallander, E. (02 de 04 de 2012). visualjazzisobar. Obtenido de http://www.visualjazz.com.au/en/news/2012/4/2/A_plea_for_device_agnostic_web_development_part1/
- [4] Kadlec, T. (2013). Implementing Responsive Design. New Riders.
- [5] UI, K. (2012). Mobile Developer Guidance. Obtenido de <http://www.kendoui.com/Libraries/Documents/mobile-developer.sflb.ashx>
- [6] <http://dev.w3.org/html5/decision-policy/html5-2014-plan.html>

Autor



Marco Antonio Avendaño Ajata

marcoviaweb@gmail.com



Herramientas de un Desarrollador I: tmux

Tmux es un multiplexor de terminal: posibilita que una o más terminales, cada una ejecutando un programa por separado, puedan crearse, accederse y controlarse desde una sola pantalla.



Introducción

Hace seis años aproximadamente empecé a usar el sistema operativo GNU/Linux, casi el mismo tiempo en el que me inicié en la programación, cuando aun cursaba el bachillerato. Hoy soy estudiante de ingeniería, próximo a graduarme, puedo decir que GNU/Linux es el mejor entorno de desarrollo que he conocido.

A lo largo de estos años he aprendido a utilizar el SO y algunas herramientas que me han ayudado a realizar mi trabajo de manera más fácil y eficiente, y que considero, debería ser el kit básico de todo desarrollador.

En este tiempo también he conocido a algunos programadores, la mayoría de ellos usuarios de Windows, dependientes de las GUIs y de los IDEs, que demoran demasiado en tareas que bien se podrían automatizar empleando las herramientas adecuadas. En la edición 2012 del CONACUP¹ un concursante que terminó por sentarse a lado mío, no pudo siquiera intentar resolver los problemas propios del evento debido a que pasó la mayor parte del tiempo cambiándose de máquina en máquina

porque eclipse se le bloqueaba. (Verídico).

Tiempo atrás leí una pregunta muy popular en StackOverflow, C++ IDE for Linux?, muchas respuestas y comentarios, de entre ellos destacó uno: UNIX is an IDE. All of it., lo cual me hizo recordar cual fue el propósito inicial de UNIX, un sistema para programadores.

Estamos iniciando el año y he decidido comenzar una serie de artículos que titulo "Herramientas de un desarrollador" como son:

- ✓ Multiplexor de terminal: tmux
- ✓ Línea de comandos: Bash
- ✓ Editor de texto: Vim y/o emacs
- ✓ Utilerías: grep, find, sed, etc.
- ✓ Sistema de control de versiones: git

Lo único que se necesita para seguir estos artículos es un emulador de terminal(xterm, gnome-terminal, konsole, etc.) y ganas de aprender. Puesto que vamos a dedicar mucho tiempo trabajando en línea de comandos, es vital una herramienta que nos permita administrar terminales en forma eficiente, esta herramienta se llama tmux y es el tema de esta primer entrega.

¿Qué es tmux?

En electrónica un multiplexor es un dispositivo que permite transmitir varios mensajes o señales de manera simultánea

sobre un canal de comunicación. Tmux es un multiplexor de terminal, posibilita que una o más terminales(o ventanas), cada una ejecutando un programa por separado, puedan accederse y controlarse desde una sola pantalla.

En mis actividades diarias empleo muy a menudo una o más terminales, cuando programo utilizo como mínimo dos, en una tengo abierto vim y en la otra ejecuto y pruebo el programa. Antes yo solía tener dos terminales abiertas en dos ventanas, usando Ctrl + Tab para moverme entre ellas, con el tiempo esto se volvió tedioso y mi vista se cansaba fácilmente debido a los constantes destellos del cambio de ventanas. Investigando un poco conocí GNU Screen, no me gustó. Encontré a tmux y fue exactamente lo que andaba buscando. Pero bueno, menos charla y más acción.

```

24 if (S[r][c] != '.') {
25     if (c < 9)
26         return backtrack(r, c + 1);
27     else
28         return backtrack(r + 1, 1);
29 } else {
30     int r0 = (r < 4) ? 1 : (r < 7) ? 2 : 3;
31     int c0 = (c < 4) ? 1 : (c < 7) ? 2 : 3;
32     int index = 3 * (r0 - 1) + c0;
33
34     for (int i = 1; i <= 9; i++) {
35         if (SQ[index][i]) continue;
36         SQ[index][i] = true;
37
38         if (!R[r][i] && !C[c][i]) {
39             S[r][c] = i + '0';

```

39,8 26%

```

[22:58:20]rendon@intel-$ cd code
[22:58:22]rendon@intel-$ g++ -Wall -o loj_1397 loj_1397.cpp
[22:58:26]rendon@intel-$ ./loj_1397

```

Figura 1.

```

24 if (S[r][c] != '.') {
25     if (c < 9)
26         return backtrack(r, c + 1);
27     else
28         return backtrack(r + 1, 1);
29 } else {
30     int r0 = (r < 4) ? 1 : (r < 7) ? 2 : 3;
31     int c0 = (c < 4) ? 1 : (c < 7) ? 2 : 3;
32     int index = 3 * (r0 - 1) + c0;
33
34     for (int i = 1; i <= 9; i++) {
35         if (SQ[index][i]) continue;
36         SQ[index][i] = true;
37
38         if (!R[r][i] && !C[c][i]) {
39             S[r][c] = i + '0';
40             R[r][i] = true;

```

38,8 27%

```

grind = moler, rechinar, afilar, pulverizar, rutina, trabajo
[22:25:18]rendon@intel-$ cd code
[22:25:49]rendon@intel-$ g++ -Wall -o loj_1397 loj_1397.cpp
[22:26:00]rendon@intel-$ ./loj_1397
intel 0:~$
Fri Dec 28 22:54:44 CST 2012

```

Figura 2.

Instalación

Para los usuarios de distribuciones Debian, Ubuntu, CentOS, Fedora, etc. es muy probable que puedan instalar tmux directamente con el gestor de paquetes o bien pueden compilarlo desde código fuente, nada complicado.

- ✓ Con el gestor de paquetes: Debian y Ubuntu
- ✓ \$ sudo apt-get install tmux
- ✓ Desde código fuente

Ir al sitio oficial de tmux(<http://tmux.sourceforge.net/>) y descargar el código fuente, yo estoy empleando la versión 1.6. Tmux depende de **libevent** y **ncurses**.

Instalar dependencias en debian

```

$ sudo apt-get install libevent-dev
libncurses5-dev
$
$ cd ~/Downloads/
$ tar xvf tmux-1.6.tar.gz
$ cd tmux-1.6/
$ ./configure
$ make
$ sudo make install
$

```

Prueba la instalación

```
$ tmux -V
```

Lo básico

Para empezar a usar tmux abre una terminal y ejecuta **tmux**, teclea el comando **exit** para salir.

Tmux al igual que vim es modal, el modo por defecto es el que nos permite interactuar con la terminal. Los otros modos son:

- ✓ **output mode**: Entramos a este modo cuando un comando que produce una salida, tal como **list-keys**, es ejecutado con una combinación de teclas.
- ✓ **copy mode**: Este modo permite copiar texto de las ventanas, del historial o de la salida producida por otros programas.

El prefijo de comandos

El prefijo de comandos es una combinación de teclas que avisa a tmux que estamos a punto de indicarle una acción que debe realizar, por defecto esta combinación es **ctrl + b**, PREFIX de aquí en adelante. La instrucción PREFIX + c se traduce en presionar las teclas **ctrl** y **b** simultáneamente, soltar las teclas y después presionar el carácter **c**.

Tmux tiene un comando para cada acción y una serie de keybindings para ejecutar algunos de ellos con una combinación de teclas, de esto hablaremos más adelante. Para entrar en modo de comandos presiona PREFIX + : y teclea el comando.

Copy mode

Como ya se mencionó, en este modo podemos navegar por el contenido que se ha generado en la terminal y copiar texto si así lo queremos. Presiona PREFIX + [para entrar en este modo.

Paneles

Tmux permite dividir el área de trabajo en varias partes, llamadas paneles, estas pueden ser horizontales o verticales.

Crea un panel horizontal.

Combinación Comando

PREFIX + " (split-window)

Crea un panel vertical.

PREFIX + % (split-window -h)

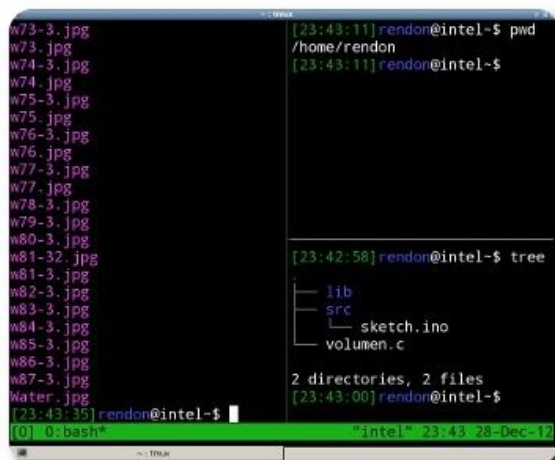


Figura 3.

Moverse al siguiente panel.

PREFIX + o (select-pane -t :.+)

Moverse al al panel superior, inferior, izquierdo y derecho.

PREFIX + UP (select-pane -U)
 PREFIX + DOWN (select-pane -D)
 PREFIX + LEFT (select-pane -L)
 PREFIX + RIGHT (select-pane -R)

Una combinación muy útil para moverse entre paneles es PREFIX + q, que enumera cada uno de los paneles por un momento, presiona el número del panel al cual te quieres trasladar y listo.

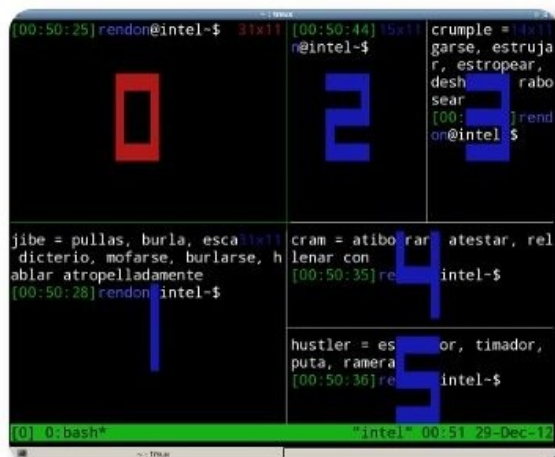


Figura 4.

Si necesitas reducir o aumentar el área de un panel lo puedes hacer con las siguientes combinaciones.

Aumentar o reducir el panel verticalmente.

```
PREFIX + C-UP      (resize-pane -U)
PREFIX + C-DOWN    (resize-pane -D)
```

Aumentar o reducir el panel horizontalmente.

```
PREFIX + C-LEFT    (resize-pane -L)
PREFIX + C-RIGHT   (resize-pane -R)
```

Para cerrar un panel tenemos dos opciones, terminar la sesión en la terminal con el comando `exit` o presionando **PREFIX + x**, que cierra el panel con previa confirmación.

Organizando paneles

Tmux proporciona cinco plantillas que nos permiten organizar nuestros paneles.

- ✓ **even-horizontal**: Alinea todos los paneles horizontalmente, de izquierda a derecha.
- ✓ **even-vertical**: Alinea todos los paneles verticalmente, de arriba a abajo.
- ✓ **main-horizontal**: Crea un panel en la parte superior que ocupa casi todo el espacio de la pantalla y alinea el resto de los paneles en la parte inferior.
- ✓ **main-vertical**: Crea un panel en el lado izquierdo que ocupa casi todo el espacio de la pantalla y alinea el resto de los paneles a la derecha.
- ✓ **tiled**: Alinea todos los paneles en la pantalla con igual cantidad de espacio.

Presiona **PREFIX + Espacio** para probar cada plantilla.

Ventanas

Cuando los paneles no son suficientes o deseamos agrupar las terminales según su actividad contamos con las ventanas.

Crear una ventana.

```
PREFIX + c
```

Las ventanas tienen nombre y lo podemos modificar con:

```
PREFIX + ,
```

Moverse entre ventanas.

```
PREFIX + n # Siguiente
PREFIX + p # Anterior
```

El listado de ventanas activas se muestra en la parte inferior de la pantalla, cada ventana tiene un número que la identifica. Una forma fácil de cambiar de ventana es presionando **PREFIX + número_de_ventana**. Si tienes demasiadas ventanas también es posible buscarlas por su nombre con **PREFIX + f** o bien presionar **PREFIX + w** para mostrar el listado de ventanas.



Figura 5.

Sesiones

Una sesión es una colección de pseudo terminales (ver `man 7 pty`) que son administradas por tmux. Cada sesión tiene una o más ventanas asociadas a ella. Las sesiones son persistentes y mantendrán su estado en caso de una desconexión. Aquí va un punto importante, si un programa se está ejecutando, cuando la sesión se interrumpe éste continuará su ejecución cuando la sesión sea reanudada.

En otras palabras las sesiones sirven para crear entornos de trabajo personalizados, por ejemplo, si están trabajando en un proyecto y

tienes abiertos varios archivos, un terminal interactivo de Ruby o Python, varias ventanas, etc. Si por alguna razón tienes que interrumpir tus actividades, resultaría un poco molesto tener que cerrar archivos, programas, paneles y ventanas, asimismo restablecer el entorno cuando se vuelva a trabajar. La solución es crear una sesión y le asignamos un nombre acorde, si trabajas en más de un proyecto o tarea simplemente crea otra sesión y mantendrás tus áreas de trabajo organizadas.

Aunque no le indiquemos explícitamente, tmux crea una sesión automáticamente cada vez que lo ejecutamos.

Crear una sesión:

```
$ tmux new -s nombre_de_sesion
```

Cerrar sesión:

```
PREFIX + d
```

o bien:

```
$ tmux kill-session -t nombre_de_sesion
```

Listar sesiones existentes:

```
$ tmux list-sessions
```

Abrir sesión cuando solo existe una:

```
$ tmux attach
```

Abrir una sesión en específico:

```
$ tmux attach -t nombre_de_sesion
```

Configuración y personalización

Hasta ahora hemos cubierto lo esencial, usando las configuraciones por defecto, sin embargo éstas no siempre son las más cómodas para todos, es por eso que tmux nos permite personalizar usando el archivo `~/.tmux.conf`. A continuación muestro algunas de las opciones disponibles:

Dar soporte para 256 colores.

```
set -g default-terminal "screen-256color"
```

Cambiar **PREFIX** a **Ctrl + a**, más cómodo.

```
set-option -g prefix C-a
unbind-key C-b
bind-key C-a send-prefix
```

Como dividir la pantalla de forma más intuitiva.

| en vez de %

```
bind | split-window -h
```

- en vez de "

```
bind - split-window -v
```

Cambiando el atajo para entrar al modo copy.

```
bind-key e copy-mode
```

Tmux soporta los modos vi y emacs para moverse dentro de la aplicación, por defecto el modo emacs es activado. Activar modo vi y algunos atajos.

```
setw -g mode-keys vi
bind-key -t vi-copy 'v' begin-selection
bind-key -t vi-copy 'y' copy-selection
```

Con estas opciones activadas, hacer lo siguiente: para copiar contenido dentro de tmux, 1) entra en modo copy, 2) presiona la tecla v para iniciar la selección y muevete con las teclas hjkl, como en vim, 3) presiona la tecla y para copiar la selección.

Moverse entre los paneles de manera similar a vim con los splits.

```
# Presiona PREFIX + [jkh]l
```

para cambiar de panel

```
unbind-key j
bind-key j select-pane -D
unbind-key k
bind-key k select-pane -U
unbind-key h
bind-key h select-pane -L
unbind-key l
bind-key l select-pane -R
```

Cambia el modo de redimensionar paneles.

```
# Presiona Ctrl + a + [jklh]
```

para redimensionar panel

```
bind-key C-a-j resize-pane -D 2
bind-key C-a-k resize-pane -U 2
bind-key C-a-h resize-pane -L 4
bind-key C-a-l resize-pane -R 4
```

Configurar la barra de estado.

Color de fondo y de letra

```
set -g status-bg black
set -g status-fg white
```

#Leyendas que se deben mostrar

#A la izquierda el nombre del host

#y a la derecha la fecha

```
set -g status-left '[fg=green]#H'
set -g status-right '[fg=yellow]#(date)'
```

Color de la ventana activa

```
set-window-option -g window-status-
current-bg blue
```

Color del panel activo

```
set-option -g pane-active-border-fg
colour027
```

Longitud del historial.

```
set -g history-limit 10000
```

Recargar configuración con **PREFIX + r**.

```
unbind r
bind r source-file ~/.tmux.conf
```

Existen muchas más opciones de configuración pero con esto es suficiente para comenzar.

Algunos hacks

Iniciar tmux automáticamente al abrir una terminal

Hemos visto que para utilizar tmux abrimos una terminal y entonces ejecutamos, para hacer que tmux se ejecute automáticamente cuando abrimos una terminal agrega las siguientes líneas al archivo `~/.bashrc`.

```
[[ $- != *i* ]] && return
[[ $TERM != screen* ]] && exec tmux -2
```

Si estamos ejecutando el interprete de comandos en modo interactivo y tenemos soporte para 256 colores entonces ejecutamos tmux.

Copiar texto al portapapeles del sistema

Estando en el modo copy es posible copiar texto para ser usado en otros paneles o ventanas dentro de tmux, pero no en otras aplicaciones, es decir, el texto copiado en tmux no se puede pegar en firefox, por citar un ejemplo. La siguiente línea soluciona el problema. Es necesario tener instalado el programa **xclip**.

```
bind C-c run "tmux save-buffer - | xclip
-i -selection clipboard"
```

Funciona de la siguiente manera, entra en modo copy y copia el texto en cuestión tal como ya se ha visto. Sal del modo copy y presiona **PREFIX + Ctrl-c** para colocar el contenido en el portapapeles del sistema.

Una desventaja que he notado con este truco es que una vez que lo usas vim ya no puede trabajar con el portapapeles del sistema(""), aun no he encontrado una solución.

Agregar una tecla control adicional

Cuando reasigne **PREFIX** a **Ctrl + a** mencioné que ésta combinación era más

cómoda, y lo es en comparación a Ctrl + b, pero en realidad para mi es más accesible porque he reasignado la tecla de la ventana para que actúe como una tecla control adicional, esta asignación trabaja a nivel de sistema y no tiene relación directa con tmux pero a mi me resulta muy útil, en especial con vim.

Para realizar la reasignación emplea el comando **xmodmap** y agrega lo siguiente en el archivo `~/.Xmodmap`.

```
remove Control = Control_L Control_R
remove mod4 = Super_L Super_R
add Control = Control_L Super_L
```

Algunos entornos de escritorio ejecutan este archivo al iniciar, si no es tu caso, agrega el

comando **xmodmap ~/.Xmodmap** al archivo de inicio de tu entorno de escritorio, para openbox el archivo es `~/.config/openbox/autostart.sh`.

Conclusión

Tmux al igual que las otras herramientas que vamos a tratar no son el tipo de programas que se aprenden a usar en 5 minutos, requieren cierta cantidad de tiempo, yo la llamo una inversión porque el tiempo que dediquen para aprender a utilizarlas lo verán remunerado en poco tiempo en forma de eficiencia y productividad.

Referencias

- [1] <http://tmux.sourceforge.net>
- [2] tmux Productive Mouse-Free Development, Brian P. Hogan
- [3] <http://stackoverflow.com/questions/24109/c-ide-for-linux>
- [4] <http://unix.stackexchange.com/questions/15715/getting-tmux-to-copy-a-buffer-to-the-clipboard>

Autor



Rafael Rendón Pablo

Estudiante de Ingeniería en Sistemas Computacionales

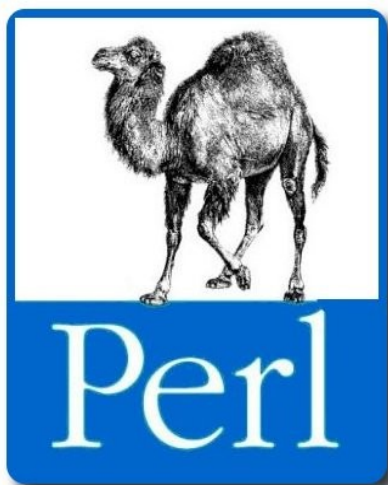
smart.rendon@gmail.com



Tratamiento de Textos en Perl

Primero pasos rumbo al Textmining

El tratamiento de textos durante la última década ha cobrado mucho interés por la relevancia del procesamiento, búsqueda y extracción de información previamente desconocida y potencialmente útil de un cuerpo de datos textuales desestructurados.



Introducción

Se calcula que entre un 75% y 85% de la información disponible en Internet se encuentra en formato no estructurado, es decir, en textos del tipo relato. Cientos de miles de artículos, resúmenes, notas, etc. están disponibles en la web, pero el acceso a ellos se produce de manera bastante básica a partir de la búsqueda de palabras en sus contenidos. Este alto porcentaje y el continuo incremento de este tipo de información ha permitido que se investiguen y desarrollen formás que permitan o faciliten su procesado y análisis con diversos objetivos.

Text Mining

En text mining, cada documento es representado como un vector, cuya dimensión es aproximadamente el número de palabras claves (keywords) distintas que se encuentran en él. Este número, puede ser considerablemente grande. Uno de los mayores retos en text mining es clasificar estos datos textuales de tal "dimensionalidad". Además de la "dimensionalidad", los algoritmos de text mining deben lidiar con la ambigüedad de las palabras como pronombres, sinónimos, errores de ortografía, abreviaturas, acrónimos y errores de semántica.

El text mining puede ser visto como una tarea de descubrir patrones desconocidos en base de datos textuales escritos en lenguaje natural.

Las tareas inmersas buscan descubrir conocimiento que no sea conocido por los expertos humanos, es decir, la información o patrones descubiertos deben ser de utilidad en la toma de decisiones.

Tratamiento de textos

El tratamiento de textos esta referido al procesamiento, búsqueda y extracción de información previamente desconocida y potencialmente útil de un cuerpo de datos textuales desestructurados, asimismo .

representa dar los primeros pasos rumbo al text mining

Características y objetivos del tratamiento de textos

El tratamiento de archivos de texto persigue varios objetivos y presenta varias características, de entre las que destacan:

Objetivos

- ✓ Explorar los datos almacenados en los archivos de texto
- ✓ Manipular y/o procesar el contenido de los archivos de texto.
- ✓ Realizar un análisis de los datos almacenados

Características

- ✓ Permite realizar recorridos exhaustivos del contenido de uno o varios archivos de texto
- ✓ Permite realizar búsquedas de coincidencias y reemplazos de patrones de texto bajo ciertos criterios.

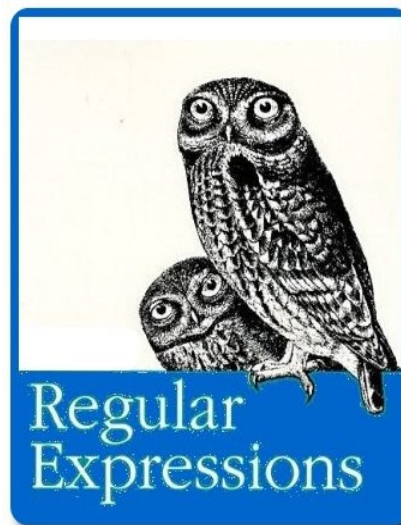
Forma de hacerlo

Existen varias formas de hacer el tratamiento de archivos de texto, una de las más sencillas y prácticas es utilizar el poder de las expresiones regulares de algún lenguaje de programación, para nuestro caso de estudio haremos uso del lenguaje Perl.

Expresión regular

Una expresión regular es un modelo o una forma de comparar con una cadena de caracteres. Esta comparación es conocida con el nombre de pattern matching o reconocimiento de patrones, permite identificar las ocurrencias del modelo en los datos tratados.

La utilización principal de las expresiones regulares en Perl consiste en la identificación de cadenas de caracteres para la búsqueda, modificación y extracción de palabras clave.



Tipos de expresiones regulares

- ✓ Expresiones regulares de búsqueda y/o comparación
- ✓ Expresiones regulares de sustitución
- ✓ Expresiones regulares de traducción.

Requisitos

Para cumplir el objetivo inicial de este artículo, precisaremos:

- ✓ Tener instalado el lenguaje Perl.
- ✓ Tener una idea básica de programación

Tratamiento de textos

Para nuestro cometido, iremos comentando los conceptos esenciales y mostraremos una aplicación de su practicidad de tal forma que se comprenda el concepto y la forma de hacer uso del mismo

Acceso a un archivo

Si deseamos procesar el contenido de un archivo, inicialmente precisamos tener acceso a él y a cada elemento componente (líneas, palabras, caracteres, números, caracteres especiales, etc)

Ejemplo 1. Dado el archivo de texto `fichero01.txt`, mostrar su contenido línea por línea.

`fichero01.txt`

```
Jose Esteban Saavedra Lopez          (591)72450061
Kathleen Diane Saavedra Ayarde       (591)72570001
Stepnahie Belen Saavedra Ayarde      (591)72682468
Scarlet Luciana Saavedra Ayarde      (591)72336912
Francisco Mauro Montecinos Collao    (591)72587981
Gabriel Rene Jimenez Mendoza         (591)25234510
David Marcelo Cartagena Guillen      (591)25246868
Franz Williams Pacheco Perez        (591)25236930
```

Ejemplo 1. Mostrar el contenido de un archivo de texto.

```
#!/usr/bin/perl
my $filename = 'fichero01.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    # Incluir aquí debajo el código para procesar la línea
    print $línea . "\n";
}
close ARCHIVO;
```

Expresiones regulares de búsqueda y/o comparación

Este tipo de expresiones regulares nos permiten evaluar si un patrón de búsqueda se encuentra en una cadena de caracteres, de modo que mediante este tipo de expresiones regulares obtendremos un valor lógico verdadero o falso según se encuentre el patrón deseado

Ejemplo 2. Mostrar las líneas que contengan el patrón buscado (Saavedra)

```
#!/usr/bin/perl
my $filename = 'fichero01.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    if($línea =~ m/Saavedra/)
    {
        print $línea . "\n";
    }
}
```

```

    }
}
close ARCHIVO;

```

Resultado 2.

```

Jose Esteban Saavedra Lopez      (591)72450061
Kathleen Diane Saavedra Ayarde   (591)72570001
Stepnahie Belen Saavedra Ayarde  (591)72682468
Scarlet Luciana Saavedra Ayarde  (591)72336912

```

Los operadores =~ y !~

Dentro de una expresión regular de comparación se permite definir si el patrón buscado o comparado esta o no presente dentro de la cadena original mediante =~ y !~ .

Ejemplo 3. Mostrar las líneas que no coincidan con el patrón buscado (Saavedra)

```

#!/usr/bin/perl
my $filename = 'fichero01.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    if($línea !~ m/Saavedra/)
    {
        print $línea . "\n";
    }
}
close ARCHIVO;

```

Resultado 3.

```

Francisco Mauro Montecinos Collao      (591)72587981
Gabriel Rene Jimenez Mendoza           (591)25234510
David Marcelo Cartagena Guillen         (591)25246868
Franz Williams Pacheco Perez          (591)25236930

```

Elementos de una expresión regular

Dentro de una expresión regular están presentes los siguientes elementos:

Átomos

Son los elementos más pequeños de una expresión regular, pueden representar una letra, un número, un caracter especial.

Modificador	Característica
.	especifica cualquier caracter salvo el retorno de carro
\n	retorno de carro (nueva línea)
\r	retorno de carro (vuelta al principio de línea)
\t	tabulador
\d	un dígito (igual que [0-9])
\D	un caracter que no sea dígito
\w	un caracter alfanumérico (como [0-9a-z_A-Z])

\W	un caracter no alfanumérico
\s	un caracter de espaciado (como [\t\r\n])
\S	un caracter que no sea de espaciado
\O	seguido de un número corresponde con ese número en octal
\x	seguido de un número hexadecimal corresponde con ese número en hexadecimal

Modificadores

Permiten variar la evaluación de una expresión regular.

Modificador	Característica
^	que indica que debe coincidir al principio de la cadena
\$	que indica que debe coincidir al final de la cadena
\b	que especifica los límites de una palabra
\B	que especifica los límites fuera de una palabra

Cuantificadores

Permiten cuantificar o expresar la cantidad de elementos de búsqueda dentro una cadena

Modificador	Característica
{n,m}	el átomo debe aparecer entre n y m veces
{n,}	el átomo debe aparecer al menos n veces
{n}	el átomo debe aparecer exactamente n veces
*	el átomo debe aparecer 0 o más veces (como {0,})
+	el átomo debe aparecer 1 o más veces (como {1,})
?	el átomo debe aparecer 0 o 1 veces (como {0,1})

Ejemplo 4. Mostrar las líneas que empiecen con cierto caracter (S)

```
#!/usr/bin/perl
my $filename = 'fichero01.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO>) {
    chomp($línea);
    if($línea =~ /^S/)
    {
        print $línea . "\n";
    }
}
close ARCHIVO;
```

Resultado 4.

```
Stepnahie Belen Saavedra Ayarde (591)72682468
Scarlet Luciana Saavedra Ayarde (591)72336912
```

Ejemplo 5. Mostrar las líneas que terminen con cierto caracter (0)

```
#!/usr/bin/perl
my $filename = 'fichero01.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    if($línea =~ /0$/)
    {
        print $línea . "\n";
    }
}
close ARCHIVO;
```

Resultado 5.

Gabriel Rene Jimenez Mendoza	(591)25234510
Franz Williams Pacheco Perez	(591)25236930

Clases o conjuntos

Una clase es un conjunto de caracteres que son utilizados dentro una expresión regular, haciendo referencia la coincidencia de uno de los caracteres de la clase o conjunto dentro la cadena original. Una clase o conjunto está encerrado entre [].y puede ser afectado por un modificador o por un cuantificador.

Ejemplo 6. Mostrar las líneas que inicien con un grupo de caracteres (Fra)

```
#!/usr/bin/perl
my $filename = 'fichero01.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    if($línea =~ /^[Fra]/)
    {
        print $línea . "\n";
    }
}
close ARCHIVO;
```

Resultado 6.

Francisco Mauro Montecinos Collao	(591)72587981
Franz Williams Pacheco Perez	(591)25236930

Ejemplo 7. Mostrar las líneas que terminen con un grupo de caracteres especificados (68)

```
#!/usr/bin/perl
my $filename = 'fichero01.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    if($línea =~ /[68]$/)
    {
        print $línea . "\n";
    }
}
```

```
}
}
close ARCHIVO;
```

Resultado 7.

```
Stepnahie Belen Saavedra Ayarde (591)72682468
David Marcelo Cartagena Guillen (591)25246868
```

Agrupaciones

Dentro las expresiones regulares muchas veces es preciso agrupar un conjunto de átomos para utilizarlos en búsquedas, sustituciones o traducciones, ésto es posible encerrando los átomos a agrupar entre ().

Extracciones

Cuando evaluamos una expresión regular, ésta puede estar compuesta de varias partes, cada parte agrupada por (), esto permite que cada parte pueda ser referenciada desde dentro o fuera de la expresión regular. Una extracción es representada por \$1, \$2, \$3, etc.

Tras el uso de una expresión regular, podemos utilizar las variables especiales \$`, \$&, \$', (las tildes esas son las que están junto con la [y con la }) que guardan lo que hay antes, lo encontrado y lo que hay después en la cadena

Referencias

Estas están ligadas a las extracciones, ya que nos permiten hacer referencia a las extracciones realizadas, pero esta vez dentro de la misma expresión regular. La forma de referenciarlas es \1, \2, \3, etc

Ejemplo 8. Mostrar las líneas que contengan letras repetidas y continuas (aa, ee, ll, rr, etc)

```
#!/usr/bin/perl
my $filename = 'fichero01.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    if($línea =~ m/(\w)\1+/)
    {
        print $línea . "\n";
    }
}
close ARCHIVO;
```

Resultado 8.

```
Jose Esteban Saavedra Lopez (591)72450061
Kathleen Diane Saavedra Ayarde (591)72570001
Stepnahie Belen Saavedra Ayarde (591)72682468
Scarlet Luciana Saavedra Ayarde (591)72336912
Francisco Mauro Montecinos Collao (591)72587981
David Marcelo Cartagena Guillen (591)25246868
Franz Williams Pacheco Perez (591)25236930
```


Expresiones regulares de sustitución

Las expresiones regulares de sustitución permiten cambiar los patrones de búsqueda por caracteres nuevos definidos por el usuario.

Ejemplo 9. Sustituir un patrón por otro (reemplazar la letra **a** por un *****)

```
#!/usr/bin/perl
my $filename = 'fichero01.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    if($línea =~ s/a/*/)
    {
        print $línea . "\n";
    }
}
close ARCHIVO;
```

Resultado 9.

Jose Esteb*n Saavedra Lopez	(591)72450061
K*thleen Diane Saavedra Ayarde	(591)72570001
Stepn*hie Belen Saavedra Ayarde	(591)72682468
Sc*rlet Luciana Saavedra Ayarde	(591)72336912
Fr*ncisco Mauro Montecinos Collao	(591)72587981
G*briel Rene Jimenez Mendoza	(591)25234510
D*vid Marcelo Cartagena Guillen	(591)25246868
Fr*nz Williams Pacheco Perez	(591)25236930

Ejemplo 10. De forma idéntica al ejercicio anterior, pero esta vez cambiamos todas las ocurrencias (observar el efecto de la opción **/g** que indica que no solo se limite a la primera ocurrencia sino realice un reemplazo global)

```
#!/usr/bin/perl
my $filename = 'fichero01.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    if($línea =~ s/a/*/g)
    {
        print $línea . "\n";
    }
}
close ARCHIVO;
```

Resultado 10.

Jose Esteb*n S**vedr* Lopez	(591)72450061
K*thleen Di*ne S**vedr* Ay*rde	(591)72570001
Stepn*hie Belen S**vedr* Ay*rde	(591)72682468
Sc*rlet Luci*n* S**vedr* Ay*rde	(591)72336912
Fr*ncisco M*uro Montecinos Coll*o	(591)72587981
G*briel Rene Jimenez Mendoz*	(591)25234510
D*vid M*rcelo C*rt*gen* Guillen	(591)25246868
Fr*nz Willi*ms P*checo Perez	(591)25236930

Expresiones regulares de Traducción

Las expresiones regulares de traducción, tienen una manera de trabajar muy parecida a las expresiones regulares de sustitución. En este caso se trata de comparar uno a uno los caracteres del patrón de búsqueda con los de la cadena de sustitución, de modo que cada vez que se encuentra una ocurrencia que coincide con uno de los caracteres del patrón se intercambia por su correspondiente en la cadena del patrón de sustitución.

Ejemplo 11. Cambiar el contenido de un archivo de texto en mayúsculas.

```
#!/usr/bin/perl
my $filename = 'fichero.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    $línea =~ tr/a-z/A-Z/;
    print $línea . "\n";
}
close ARCHIVO;
```

Resultado 11.

JOSE ESTEBAN SAAVEDRA LOPEZ	(591)72450061
KATHLEEN DIANE SAAVEDRA AYARDE	(591)72570001
STEPNAHIE BELEN SAAVEDRA AYARDE	(591)72682468
SCARLET LUCIANA SAAVEDRA AYARDE	(591)72336912
FRANCISCO MAURO MONTECINOS COLLAO	(591)72587981
GABRIEL RENE JIMENEZ MENDOZA	(591)25234510
DAVID MARCELO CARTAGENA GUILLEN	(591)25246868
FRANZ WILLIAMS PACHECO PEREZ	(591)25236930

Ejemplo12. Realizar un simple cifrado de un archivo de texto (reemplazar las a por b y b por c, y así sucesivamente, de similar forma con los números) . Para el ejemplo suponemos que el archivo fuente se encuentra en minúsculas.

```
#!/usr/bin/perl
my $filename = 'fichero.txt';
open ARCHIVO,$filename;
my $línea;
while ( $línea = <ARCHIVO> ) {
    chomp($línea);
    $línea =~ tr/a-z1234567890/bcdefghijklmnopqrstuvwxyz2345678901/;
    print $línea . "\n";
}
close ARCHIVO;
```

Resultado 12

kptf ftufcbo tbbwfesb mpqfa	(602)83561172
lbui mffo ejbof tbbwfesb bzbsef	(602)83681112
tufqobijf cfmfo tbbwfesb bzbsef	(602)83793579
tdbsmfu mvdjbob tbbwfesb bzbsef	(602)83447023
gsbodjtdp nbvsp npoufdjopt dpmmmbp	(602)83698092
hbcsjfm sfof kjnfofa nfoepab	(602)36345621

ebwje nbsdfmp dbsubhfob hvjmmfo	(602) 36357979
gsboa xjmmjbnt qbdifdp qfsfa	(602) 36347041

Importante

Todos los ejemplos descritos en este artículo, fueron desarrollados con objetivos netamente académicos de la forma más simple y sencilla, especialmente orientado a las personas que recién se inician en este tema. Seguros estamos que existe siempre la posibilidad de poder mejorarlos y desarrollarlos de forma óptima.

Próximamente

En las próximas entregas profundizaremos los conceptos de tratamiento de textos, text mining y expresiones regulares mediante ejemplos diversos.

Autor



Esteban Saavedra López

Presidente de la Fundación AtixLibre (Bolivia)

Activista de Software Libre en Bolivia

jesaavedra@opentelematics.org

esteban.saavedra@atixlibre.org

<http://jesaavedra.opentelematics.org>

<http://www.atixlibre.org>



Exceptions

Clases de errores incorporados

El módulo `exception` define los errores incorporados usados en toda la biblioteca estándar y por el intérprete.



Introducción

En el pasado, Python ha soportado mensajes simples, cadenas de texto como excepciones, así como clases. Desde 1.5, todos los módulos de la biblioteca estándar utilizan clases como excepciones. A partir de Python 2.5, las excepciones de cadenas de texto dan como resultado una **DeprecationWarning**, el soporte para excepciones como cadena de texto se eliminará en el futuro.

Clases base

Las clases de excepción se definen en una jerarquía, descrita en la documentación de la biblioteca estándar. Además de las ventajas obvias de la organización, la herencia de excepciones es útil porque las excepciones relacionadas pueden ser atrapadas por la captura de su clase base. En la mayoría de los casos, estas clases base no están destinadas a ser elevadas directamente.

BaseException

Clase base para todas las excepciones. Implementa lógica para crear una representación textual de la excepción usando `str()` en base a los argumentos dados al constructor.

Exception

Clase base para excepciones que no terminan la ejecución de la aplicación. Todas las excepciones definidas por el usuario deberían usar **Exception** como clase base.

StandardError

Clase base para las excepciones incorporadas en la biblioteca estándar.

ArithmeticError

Clase base para los errores relacionados a la matemática.

LookupError

Clase base para errores elevados cuando algo no puede ser encontrado.

EnvironmentError

Clase base para errores que provienen de fuera de Python (el sistema operativo, el sistema de archivos, etc.)

Excepciones

AssertionError

Un **AssertionError** se produce cuando una sentencia **assert** falla.

```
assert False, 'La afirmación falló'

$ python exceptions_AssertionError_assert.py
Traceback (most recent call last):
  File "exceptions_AssertionError_assert.py", line 12, in <module>
    assert False, 'La afirmación falló'
AssertionError: La afirmación falló
```

También se usa en el módulo **unittest** en métodos como **failIf()**.

```
import unittest

class AssertionExample(unittest.TestCase):

    def test(self):
        self.failUnless(False)

unittest.main()
```

```
$ python exceptions_AssertionError_unittest.py

F
=====
FAIL: test (__main__.AssertionExample)
-----
Traceback (most recent call last):
  File "exceptions_AssertionError_unittest.py", line 17, in test
    self.failUnless(False)
AssertionError: False is not true
-----

Ran 1 test in 0.000s

FAILED (failures=1)
```

AttributeError

Cuando una referencia a un atributo o una asignación falla, **AttributeError** se produce. Por ejemplo, cuando se trata de hacer referencia a un atributo que no existe.

```
class SinAtributos(object):
    pass

o = SinAtributos()
print o.atributo
```



```
$ python exceptions_AttributeError.py
Traceback (most recent call last):
  File "exceptions_AttributeError.py", line 16, in <module>
    print o.atributo
AttributeError: 'SinAtributos' object has no attribute 'atributo'
```

O cuando se intenta modificar un atributo que es sólo de lectura:

```
class MiClase(object):
    @property
    def atributo(self):
        return 'Este es el valor del atributo'

o = MiClase()
print o.atributo
o.atributo = 'Valor nuevo'
```

```
$ python exceptions_AttributeError_assignment.py

This is the attribute value
Traceback (most recent call last):
  File "exceptions_AttributeError_assignment.py", line 20, in <module>
    o.attribute = 'Valor nuevo'
AttributeError: can't set attribute
```

EOFError

Un **EOFError** se produce cuando una función incorporada como `input()` o `raw_input()` lee datos antes de encontrar el final del flujo de entrada. Métodos de archivo como `read()` devuelven una cadena vacía al final del archivo.

```
while True:
    data = raw_input('prompt:')
    print 'READ:', data
```

```
$ echo hello | python PyMOTW/exceptions/exceptions_EOFError.py
prompt:READ: hello
prompt:Traceback (most recent call last):
  File "PyMOTW/exceptions/exceptions_EOFError.py", line 13, in <module>
    data = raw_input('prompt:')
EOFError: EOF when reading a line
```

FloatingPointError

Se produce por operaciones de punto flotante que resultan en errores, cuando el control de excepciones de punto flotante (`fpect1`) está activo. Habilitar `fpect1` requiere de un intérprete compilado con `--with-fpect1`. El uso de `fpect1` es desalentado en la documentación de la biblioteca estándar.

GeneratorExit

Se produce en el interior de un generador, cuando en método `close()` es ejecutado.

```
def mi_generador():
    try:
        for i in range(5):
            print 'Produciento', i
            yield i
    except GeneratorExit:
        print 'Terminando'

g = mi_generador()
print g.next()
g.close()
```

```
$ python exceptions_GeneratorExit.py

Produciento 0
0
Terminando
```

IOError

Se produce cuando la entrada o salida fallan, por ejemplo cuando un disco se llena o el archivo de entrada no existe.

```
f = open('/no/existe', 'r')
$ python exceptions_IOError.py
Traceback (most recent call last):
  File "exceptions_IOError.py", line 12, in <module>
    f = open('/no/existe', 'r')
IOError: [Errno 2] No such file or directory: '/no/existe'
```

ImportError

Se produce cuando un módulo, o un miembro de un módulo, no pueden ser importados. Existen un par de condiciones en las que un **ImportError** se puede producir.

- ✓ Si el módulo no existe.

```
import modulo_no_existe

$ python exceptions_ImportError_nomodule.py

Traceback (most recent call last):
  File "exceptions_ImportError_nomodule.py", line 12, in <module>
    import modulo_no_existe
ImportError: No module named modulo_no_existe
```

- ✓ Si `from X import Y` es usado y `Y` no puede ser encontrado dentro del módulo `X`, un **ImportError** es creado.

```
from exceptions import NombreInventado
$ python exceptions_ImportError_missingname.py

Traceback (most recent call last):
  File "exceptions_ImportError_missingname.py", line 12, in <module>
    from exceptions import NombreInventado
ImportError: cannot import name NombreInventado
```

IndexError

Un **IndexError** se produce cuando una referencia a una secuencia está fuera de rango.

```
mi_seq = [ 0, 1, 2 ]
print mi_seq[3]

$ python exceptions_IndexError.py

Traceback (most recent call last):
  File "exceptions_IndexError.py", line 13, in <module>
    print mi_seq[3]
IndexError: list index out of range
```

KeyError

De manera similar, un **KeyError** se produce cuando un valor no se encuentra como una llave de un diccionario.

```
d = { 'a':1, 'b':2 }
print d['c']

$ python exceptions_KeyError.py

Traceback (most recent call last):
  File "exceptions_KeyError.py", line 13, in <module>
    print d['c']
KeyError: 'c'
```

KeyboardInterrupt

Un **KeyboardInterrupt** ocurre cuando el usuario presiona **ctrl-c** (o Supr) para detener un programa en ejecución. A diferencia de la mayoría de las otras excepciones, **KeyboardInterrupt** hereda directamente de **BaseException** para evitar ser atrapada por un controlador de excepciones globales que captura **Exception**.

```
try:
    print 'Presiona Return o Ctrl-C:',
    ignored = raw_input()
except Exception, err:
    print 'Exception capturada:', err
except KeyboardInterrupt, err:
    print 'KeyboardInterrupt capturada'
else:
    print 'Ninguna excepción'
Presionar Ctrl-C an ocasiona una excepción ``KeyboardInterrupt``.
```

```
$ python exceptions_KeyboardInterrupt.py
Press Return or Ctrl-C: ^CCaught KeyboardInterrupt
```

MemoryError

Si tu programa se queda sin memoria es posible recuperarse (eliminado algunos objetos, por ejemplo) un **MemoryError** se produce.

```
import itertools

# Intenta ocasionar un MemoryError allocando un montón de memoria
l = []
for i in range(3):
    try:
        for j in itertools.count(1):
            print i, j
            l.append('' * (2**30))
    except MemoryError:
        print '(error, descartando lista existente)'
        l = []
```

```
$ python exceptions_MemoryError.py
python(49670) malloc: *** mmap(size=1073745920) failed (error code=12)
*** error: can't allocate region
*** set a breakpoint in malloc_error_break to debug
python(49670) malloc: *** mmap(size=1073745920) failed (error code=12)
*** error: can't allocate region
*** set a breakpoint in malloc_error_break to debug
python(49670) malloc: *** mmap(size=1073745920) failed (error code=12)
*** error: can't allocate region
*** set a breakpoint in malloc_error_break to debug
0 1
0 2
0 3
(error, descartando lista existente)
1 1
1 2
1 3
(error, descartando lista existente)
2 1
2 2
2 3
(error, descartando lista existente)
```

NameError

NameErrors se producen cuando tu código hace referencia a un nombre que no existe en el ámbito actual. Por ejemplo, un nombre de variable no calificado.

```
def func():
    print nombre_desconocido

func()

$ python exceptions_NameError.py

Traceback (most recent call last):
  File "exceptions_NameError.py", line 15, in <module>
    func()
  File "exceptions_NameError.py", line 13, in func
    print nombre_desconocido
NameError: global name 'nombre_desconocido' is not defined
```

NotImplementedError

Clases base definidas por el usuario pueden producir **NotImplementedError** para indicar que un método o funcionamiento necesitan ser definidos por una sub clase, simulando una interfaz.

```

class ClaseBase(object):
    """Define la interfaz"""
    def __init__(self):
        super(ClaseBase, self).__init__()
    def haz_algo(self):
        """La interfaz, sin implementación"""
        raise NotImplementedError(self.__class__.__name__ + '.haz_algo')

class SubClase(ClaseBase):
    """Implementa la interfaz"""
    def haz_algo(self):
        """hace algo de verdad"""
        print self.__class__.__name__ + ' haciendo algo!'

SubClase().haz_algo()
BaseClass().haz_algo()

```

```

$ python exceptions_NotImplementedError.py

SubClase haciendo algo!
Traceback (most recent call last):
  File "exceptions_NotImplementedError.py", line 27, in <module>
    ClaseBase().haz_algo()
  File "exceptions_NotImplementedError.py", line 18, in do_something
    raise NotImplementedError(self.__class__.__name__ + '.haz_algo')
NotImplementedError: ClaseBase.haz_algo

```

OSError

OSError sirve como clase de errores para el módulo os, se produce cuando una operación específica al sistema operativo genera un error.

```

import os

for i in range(10):
    print i, os.ttyname(i)

```

```

$ python exceptions_OSError.py

0 /dev/pts/0
1 /dev/pts/0
2 /dev/pts/0
3
Traceback (most recent call last):
  File "exceptions_OSError.py", line 15, in <module>
    print i, os.ttyname(i)
OSError: [Errno 9] Bad file descriptor

```

OverflowError

Cuando una operación aritmética supera los límites del tipo de variable, un **OverflowError** se produce. Enteros largos asignan más espacio a manera que el valor crece, así que terminan produciendo **MemoryError**. El manejo de excepciones de punto flotante no está estandarizado, así que no son controlados. Enteros regulares son convertidos en enteros largos según sea necesario.

```
import sys

print 'Entero regular: (maxint=%s)' % sys.maxint
try:
    i = sys.maxint * 3
    print 'Sin desborde para ', type(i), 'i =', i
except OverflowError, err:
    print 'Desborde en ', i, err

print
print 'Entero largo:'
for i in range(0, 100, 10):
    print '%2d' % i, 2L ** i

print
print 'Valores de punto flotante:'
try:
    f = 2.0**i
    for i in range(100):
        print i, f
        f = f ** 2
except OverflowError, err:
    print 'Desborde después de ', f, err
```

```
$ python exceptions_OverflowError.py

Entero regular: (maxint=9223372036854775807)
Sin desborde para <type 'long'> i = 27670116110564327421

Entero largo:
0 1
10 1024
20 1048576
30 1073741824
40 1099511627776
50 1125899906842624
60 1152921504606846976
70 1180591620717411303424
80 1208925819614629174706176
90 1237940039285380274899124224
```

```
Valores de punto flotante:
0 1.23794003929e+27
1 1.53249554087e+54
2 2.34854258277e+108
3 5.5156522631e+216
Desborde después de 5.5156522631e+216 (34, 'Numerical result out of range')
```

ReferenceError

Cuando un proxy weakref es usado para acceder a un objeto que ya ha sido recolectado, un **ReferenceError** se produce.

```
import gc
import weakref

class ObjetoCaro(object):
    def __init__(self, name):
        self.name = name
    def __del__(self):
        print '(Eliminando %s)' % self
```



```
obj = ObjetoCaro('obj')
p = weakref.proxy(obj)

print 'ANTES:', p.name
obj = None
print 'DESPUÉS:', p.name
```

```
$ python exceptions_ReferenceError.py

ANTES: obj
(Eliminando <__main__.ExpensiveObject object at 0x100d9e950>)
DESPUÉS:
Traceback (most recent call last):
  File "exceptions_ReferenceError.py", line 26, in <module>
    print 'DESPUÉS:', p.name
ReferenceError: weakly-referenced object no longer exists
```

RuntimeError

Una excepción **RuntimeError** se usa cuando no se aplica otra excepción más específica. El intérprete no produce esta excepción muy a menudo, pero algún código de usuario lo hace.

StopIteration

Cuando un iterador ha terminado, su método `next()` produce **StopIteration**. Esta excepción no es considerada un error.

SyntaxError

Un **SyntaxError** ocurre siempre que el analizador (parser) encuentra código fuente que no entiende. Esto puede ser mientras importa un módulo, invoca `exec` o ejecuta `eval()`. Los atributos de la excepción pueden ser usados para encontrar exactamente qué parte del texto de entrada ocasionó la excepción.

```
try:
    print eval('five times three')
except SyntaxError, err:
    print 'Syntax error %s (%s-%s): %s' % \
        (err.filename, err.lineno, err.offset, err.text)
    print err
```

```
$ python exceptions_SyntaxError.py

Syntax error <string> (1-10): five times three
invalid syntax (<string>, line 1)
```

SystemError

Cuando ocurre un error en el propio intérprete y hay cierta oportunidad de continuar la ejecución con éxito se produce un **SystemError**. **SystemErrors** probablemente indican un error en el intérprete y deberían ser reportados al mantenedor.

SystemExit

Cuando se invoca `sys.exit()`, se produce **SystemExit** en lugar de terminar inmediatamente. Ésto permite ejecutar código de limpieza en bloques `try: finally` y entornos especiales (como depuradores a marcos de pruebas) atrapar la excepción y evitar terminar.

TypeError

TypeError se producen combinando tipos equivocados de objetos, o ejecutando funciones con el tipo equivocado de objetos.

```
result = ('tuple',) + 'string'
```

```
$ python exceptions_TypeError.py
Traceback (most recent call last):
  File "exceptions_TypeError.py", line 12, in <module>
    result = ('tuple',) + 'string'
TypeError: can only concatenate tuple (not "str") to tuple
```

UnboundLocalError

Un **UnboundLocalError** es un tipo de **NameError** específico para nombres de variables locales.

```
def ocasiona_name_error_global():
    print nombre_global_desconocido

def ocasiona_unbound_local():
    valor_local = valor_local
    print valor_local

try:
    ocasiona_name_error_global()
except NameError, err:
    print 'Name error global:', err
```

```
try:
    ocasiona_unbound_local()
except UnboundLocalError, err:
    print 'Name error local:', err
```

La diferencia entre el **NameError** global y el **UnboundLocalError** es la manera en que el nombre se usa. El nombre `valor_local` es interpretado como un nombre local de variable porque aparece en el lado izquierdo de una expresión.

```
$ python exceptions_UnboundLocalError.py
Name error global: global name 'nombre_global_desconocido' is not defined
Name error local: local variable 'valor_local' referenced before assignment
```

UnicodeError

UnicodeError es una sub clase de **ValueError** y se produce cuando un problema de unicode ocurre. Hay sub clases para **UnicodeEncodeError**, **UnicodeDecodeError** y **UnicodeTranslateError**

ValueError

Un **ValueError** se usa cuando una función recibe un valor que tiene el tipo correcto pero un valor inválido.

```
print chr(1024)
```

```
$ python exceptions_ValueError.py
Traceback (most recent call last):
  File "exceptions_ValueError.py", line 12, in <module>
    print chr(1024)
ValueError: chr() arg not in range(256)
```

ZeroDivisionError

Cuando cero aparece como denominador en una operación de división, un **ZeroDivisionError** se produce.

```
print 1/0
$ python exceptions_ZeroDivisionError.py
Traceback (most recent call last):
  File "exceptions_ZeroDivisionError.py", line 12, in <module>
    print 1/0
ZeroDivisionError: integer division or modulo by zero
```

Categorías de advertencia

También hay varias excepciones definidas para el uso con el módulo warnings.

Warning

La clase base de todas las advertencias.

UserWarning

Clase base para advertencias que provienen del código del usuario

DeprecationWarning

Usada para características que ya no se mantienen.

PendingDeprecationWarning

Usada para características que van a ser obsoletas pronto.

SyntaxWarning

Usada para sintaxis cuestionable.

RuntimeWarning

Usada para los eventos que suceden en tiempo de ejecución que podrían causar problemas.

FutureWarning

Advertencia sobre cambios en en lenguaje o la biblioteca que llegarán en un momento posterior.

ImportWarning

Advertencia sobre problemas importando un módulo.

UnicodeWarning

Advertencia sobre problemas con texto Unicode.

Autor



Ernesto Rico Schmidt

Usuario de Linux y Software Libre desde 1994
e.rico.schmidt@gmail.com





Willay News

GNU/LINUX

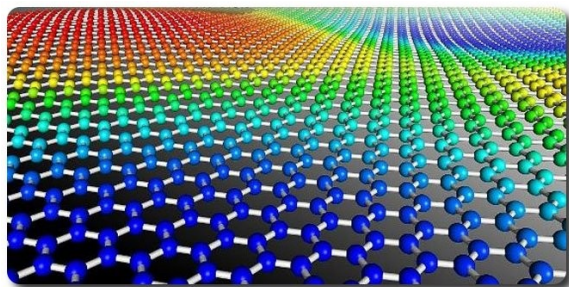
TECNOLOGÍA

CONOCIMIENTO

Novedades Tecnológicas

Grafeno

El grafeno, uno de los materiales más finos, flexibles, fuertes y con mayor conductividad que existen, está llamado a revolucionar el futuro, con importantes cambios en la industria de la telefonía móvil y las telecomunicaciones .



Los teléfonos móviles y otros dispositivos serán flexibles y se plegarán y desplegarán según nuestras necesidades gracias al nuevo material .



Uno de los cambios más inmediatos será el de las pantallas táctiles, tan de moda en tablets y smartphones durante los últimos años. Y todo gracias a que posee una extraordinaria flexibilidad mecánica y una resistencia química sin igual, muy superior a la de los materiales que se usan en la actualidad. Según los autores, las primeras pantallas táctiles de grafeno verán la luz en un plazo que va de los tres a los cinco años.

Se estima que el grafeno abrirá toda una nueva era de «dispositivos flexibles». Una revolución tecnológica comparable a la que supuso el paso de las lámparas a los transistores, o de éstos a los circuitos electrónicos. En algo más de una década, los dispositivos electrónicos ya no serán rígidos, como sucede con los actuales, sino elásticos, lo cual les permitirá cambiar de configuración (de forma) y también de funciones según las necesidades de cada momento.



En un futuro cercano los dispositivos móviles podrían ser una especie de lámina de plástico transparente, flexible y desplegable, de forma que podamos, a voluntad, llevarla en el bolsillo o desplegarla varias veces hasta que tenga el tamaño estándar de la pantalla de un ordenador. Cada usuario elegirá si quiere utilizar su dispositivo para hablar por teléfono, para ver una película, para trabajar o para compartir documentos con sus contactos.



Reemplazo del GPS

China tiene su propio sistema GPS que rivaliza con el de Estados Unidos .



El gobierno de China lanzó en la región de Asia-pacífico los servicios públicos y comerciales de su propio sistema satelital de navegación, diseñado y construido como competidor del estadounidense Sistema de Posicionamiento Global (GPS).



El sistema Beidou comenzó a proporcionar servicios a civiles en toda la región, y de acuerdo con la prensa estatal se espera que para el 2020 pueda ofrecer cobertura global.

Se trata del último logro de China en tecnología espacial. El país se propone construir una estación espacial para el fin de la presente década y eventualmente enviar una misión tripulada a la luna.

Autor



Jenny Saavedra López

Diseño y Edición Revista Atix
jenny.saavedra@atixlibre.org

Libres para pensar, libres para decidir, libres para crear

Arte Libre

Te ofrecemos este espacio para mostrar tu Creatividad



Envíanos tus diseños y creaciones para publicarlos



Camión al éxito

Contacto

Para solicitar cualquier información, puedes contactar a:

- ✓ Esteban Saavedra López (esteban.saavedra@atixlibre.org)
- ✓ Jenny Saavedra (jenny.saavedra@atixlibre.org)

Visita nuestro sitio web y descarga todos los números

- ✓ <http://revista.atixlibre.org>

Envío de Artículos

Te invitamos a participar de la Revista Atix enviándonos artículos referidos a las siguientes áreas :

- ✓ Instalación y personalización de aplicaciones
- ✓ Scripting
- ✓ Diseño gráfico
- ✓ Programación y desarrollo de aplicaciones
- ✓ Administración de servidores
- ✓ Seguridad
- ✓ y cualquier tema enmarcado dentro del uso de Software Libre
- ✓ Conocimiento Libre
- ✓ Tecnología Libre
- ✓ Cultura Libre
- ✓ Trucos y recetas.
- ✓ Noticias.

Te falta algun número de

atix

obtenlos de nuestra web



Hacia un Futuro Innovador



<http://www.atixlibre.org>

Por un Mundo Ético, Libre y Justo



Por un Mundo Ético, Libre y Justo

<http://revista.atixlibre.org>